

Statistical inference in generative models using scoring rules



Lorenzo Pacchiardi
St. Peter's College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy
Trinity 2022

Statement of Originality

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. My personal contributions are as outlined in the authorship forms at the end of each chapter. This dissertation is my own work except as specified in the text, acknowledgements, forms and papers.

Lorenzo Pacchiardi
Trinity 2022

This thesis is dedicated to Valeria
and to my family: Elena, Andrea, Marta, Giovanni and Rita.

Acknowledgements

First of all, I owe much to my supervisors, who supported me throughout my PhD journey and were an endless source of inspiration. Specifically, I am grateful to Ritabrata Dutta for his constant availability, for sharing with me his enthusiasm for new ideas, and for his friendship; I thank Geoff Nicholls for his all-round mentorship during my time at Oxford and for transmitting part of his vast knowledge of statistics to me.

Next, I would like to thank Pierre Jacob and Christian Robert for hosting me in Paris and Antonietta Mira for inviting me twice to spend time in Lugano. Discussing research ideas with them gave me stimulating new perspectives. I am also grateful to Chris Holmes, Judith Rousseau and Arnaud Doucet for acting as examiners for the intermediate milestones of my PhD and offering constructive feedback on my work. Thanks also to Patrick Rebeschini and Massimiliano Pontil, with whom I carried out a short project during my first year: working with them helped me develop a good research practice from the very beginning. Looking back, I am indebted to my MSc mentor Florent Krzakala for encouraging me to embark on this journey.

If I learnt a lot during my PhD, it is also because of the many group meetings, reading groups and seminar series I attended. I thank all the participants and organisers (too many to be listed here) for providing me with food for thought.

I am aware that, if I have been able to fully concentrate on my studies, it is also due to the great work done by the administrative staff in the Department of Statistics, whom I thank.

I gratefully acknowledge the Engineering and Physical Sciences Research Council and the Medical Research Council for funding my doctoral studies.

My time at Oxford has been a wonderful experience thanks to the incredible people I have met: first of all, the friends who made the countless

hours spent in the Department of Statistics interesting and enjoyable and that shared with me the ups and downs of PhD life (special mentions to Déborah Sulem, Anna Menacher, Romain Fournier, Shahine Bouabid, Yiorgos Kalantzis, Alan Chau, Sahra Ghalebikesabi, Kamélia Daudel, Natalia García Martín, Luis Torada Aguilera, Bobby He). I also thank my housemates Anri Asagumo, Eugenio Clerico and Adrien Sprumont for sharing many meals, pizza nights and Star Wars nights (but no double-blitz!). I thank all the members of St. Peter's College Boat Club for providing a great way to unwind from the stress of research. Finally, I am grateful to all those not listed here who have offered me their friendship, mentorship or collaboration at any point of my journey.

Thanks to my good friends (or *'mbari*) Stefano Falletta, Andrea Pizzi and Enrica Racca, for our friendship that has lasted since our time at PoliTo across space and time; their advice and support are always most valuable to me, as coming from people sharing similar experiences and ambitions.

My parents, Elena and Andrea, believed in me longer than anyone else. I owe them a large part of what I achieved and who I am today. I thank my brother Giovanni for the stimulating discussions and for keeping my ambitions high, my sister Marta for her affection and simplicity, and my grandmother Rita for her kindness, care and hospitality. I also owe a lot to my second family (Paola, Roberto and Maura), who have always supported me with their welcomeness and availability.

Above all, I want to thank Valeria for always being by my side, even if often separated by thousands of kilometres, and for being my companion of adventures. Your love kept me afloat in the darkest moments, and I am privileged to share my life with you.

Abstract

Statistical models which allow generating simulations without providing access to the density of the distribution are called simulator models. They are commonly developed by scientists to represent natural phenomena and depend on physically meaningful parameters. Analogously, generative networks produce samples from a probability distribution by transforming draws from a noise (or latent) distribution via a neural network; as for simulator models, the density is unavailable. These two frameworks, developed independently from different communities, can be grouped into the class of *generative models*; compared to statistical models that explicitly specify the density, they are more powerful and flexible.

For generative networks, typically, a single point estimate for the parameters (or weights) is obtained by minimizing an objective function through gradient descent enabled by automatic differentiation. In contrast, for simulator models, samples from a probability distribution for the parameters are usually obtained via some statistical algorithm. Nevertheless, in both cases, the inference methods rely on common principles that exploit simulations. In this thesis, I follow the principle of assessing how a probabilistic model matches an observation by *Scoring Rules*. This generalises common statistical practices based on the density function and, with specific Scoring Rules, allows tackling generative models.

After a detailed introduction and literature review in Chapter 1, the first part of this thesis (Chapters 2 and 3) is concerned with methods to infer probability distributions for the parameters of simulator models. Specifically, Chapter 2 contributes to the traditional *Bayesian Likelihood-Free Inference* literature with a new way to learn summary statistics, defined as the sufficient statistics of the best exponential family approximation to the simulator model. In contrast, Chapter 3 departs from tradition by defining a new posterior distribution based on the *generalised Bayesian inference framework*, rather than motivated as an approximation to the

standard posterior. The posterior is defined through Scoring Rules computable for simulator models and is robust to outliers.

In the second part of the thesis (Chapters 4 and 5), I study *Scoring Rule Minimization* to determine the weights of generative networks; for specific choices of Scoring Rules, this approach better captures the variability of the data than popular alternatives. I apply generative networks trained in this way to uncertainty-sensitive tasks: in Chapter 4 I use them to provide a probability distribution over the parameters of simulator models, thus falling back to the theme of Chapters 2 and 3; instead, in Chapter 5, I consider probabilistic forecasting, also establishing consistency of the training objective with dependent training data.

Finally, I conclude in Chapter 6 with some final thoughts and directions for future work.

Contents

1	Introduction and Literature review	1
1.1	Inference for generative models	3
1.1.1	Bayesian Likelihood-Free Inference	4
1.1.1.1	Approximate Bayesian Computation methods	4
1.1.1.2	Surrogate Likelihood methods	8
1.1.1.3	Neural network approximations	8
1.1.2	Inference for generative networks	11
1.1.2.1	Generative Adversarial Networks	12
1.2	Background	15
1.2.1	Generalised Bayesian inference	15
1.2.1.1	Properties and issues of standard Bayesian inference	15
1.2.1.2	Loss-based generalised Bayesian inference	17
1.2.2	Scoring Rules	18
1.2.2.1	Scoring Rules and probabilistic forecasting	19
1.2.2.2	Examples of Scoring Rules for continuous distributions	21
1.3	Contributions and thesis outline	24
1.3.1	Score Matched Neural Exponential Families for Likelihood-Free Inference	24
1.3.2	Generalised Bayesian Likelihood-Free Inference Using Scoring Rules Estimators	25
1.3.3	Training generative networks via Scoring Rule minimization .	26
2	Score Matched Neural Exponential Families for Likelihood-Free In- ference	28
3	Generalized Bayesian Likelihood-Free Inference Using Scoring Rules Estimators	97

4	Likelihood-Free Inference with Generative Neural Networks via Scoring Rule Minimization	156
5	Probabilistic Forecasting with Generative Networks via Scoring Rule Minimization	188
6	Conclusions and discussion	247
6.1	Summary	247
6.2	Extensions	248
	Bibliography	251

Chapter 1

Introduction and Literature review

Increasing scientific knowledge and computational power allow scientists to develop more complex computer simulations of natural phenomena. These *simulator models* often include stochastic components to account for uncertainty in the simulation; therefore, running the simulation twice under the same conditions produces different outputs. Moreover, achieving a good representation of reality typically requires tuning some *parameters* from real-world observations.

This setup presents all the ingredients necessary for the practice of statistics: a stochastic description of some process, real-world observations, and parameters to determine. For complex simulator models, however, the exact form of the *density* of the probability distribution from which simulations are generated when running the simulator (and hence the *likelihood*) is unavailable, which makes most traditional statistical methods inapplicable. Thus, statistical inference for simulator models requires dedicated *likelihood-free* techniques. Despite the research effort on these methods, they still present some drawbacks: for instance, Approximate Bayesian Computation (ABC, Sec. 1.1.1.1) and Surrogate Likelihood methods (Sec. 1.1.1.2) target approximate posterior distributions by employing a large number of model simulations and typically summarise the data to low-dimensional *summary statistics*, whose specification is difficult. Recent neural-network-based approaches (Sec. 1.1.1.3) are more efficient in terms of model simulations (Lueckmann et al., 2021) but are limited to specific neural network architectures. Most instances of the above methods, moreover, are highly sensitive to outliers in the observed data (Schmitt et al., 2022; Cannon et al., 2022).

In parallel, deep learning has produced extraordinary results (Silver et al., 2016; Brown et al., 2020; Jumper et al., 2021) by using neural networks stacking a growing number of matrix multiplications and non-linear functions. Specifically, *generative*

(neural) networks underpin the successes of deep learning in generating realistic images (Gui et al., 2021), text (Brown et al., 2020) and video (Clark et al., 2019). In a nutshell, generative networks transform random noise into a meaningful sample via a neural network; in this way, they parametrize a probability distribution on the space of samples without providing access to its density. Generative networks are trained by iteratively comparing generated samples to training data and adjusting the neural network weights accordingly. In the popular *adversarial* framework (Sec. 1.1.2), the comparison is carried out by another *discriminator* neural network, which is trained alongside the first. Unfortunately, while this leads to realistic samples, the learnt generative distribution is often overconfident, thus not representing the full variability of the data (Isola et al., 2017; Goodfellow, 2016; Bellemare et al., 2017; Arora et al., 2018; Richardson and Weiss, 2018), which makes adversarial methods unsuitable to reliably quantify the uncertainty in a process. Furthermore, adversarial training is notoriously unstable (Salimans et al., 2016) and theoretically unmotivated for dependent training data.

Although they were independently developed and present different features, both generative networks and simulator models allow sample generation without density evaluation. Indeed, Mohamed and Lakshminarayanan (2017) discusses how common principles underlie inference methods for both approaches, which they group into the class of *(implicit) generative models*.

In this thesis, I develop new inference approaches for generative networks and simulator models. To do so, I build on *Scoring Rules* (Sec. 1.2.2), which provide general ways of evaluating probabilistic statements for observations. Importantly, some Scoring Rules only require the ability to generate samples and are thus naturally applicable to generative models. Nevertheless, this common framework must be articulated in different methods taking into account the specific properties of generative networks and simulator models.

My proposed approaches address some of the issues mentioned above: in Chapter 2, I build new summary statistics for ABC by exploiting the sufficient statistics of an exponential family trained to minimize a Scoring Rule; in Chapter 3, I define an outlier-robust distribution on parameters of simulator models using a generalised Bayesian formulation (Sec. 1.2.1) based on Scoring Rules. Finally, in Chapters 4 and 5, I train generative networks to minimize Scoring Rules, which yields distributions better representing the data variability, and use this approach for uncertainty-sensitive tasks: Bayesian Likelihood-Free Inference in Chapter 4 and probabilistic forecasting

in Chapter 5 (for which I derive a theoretically sound objective for dependent training data).

In the rest of this chapter, I review state-of-the-art inference methods for simulator models and generative networks in Section 1.1. Then, in Section 1.2, I introduce the tools on which my proposed methods in Chapters 2 to 5 build, namely generalised Bayesian inference in Section 1.2.1 and Scoring Rules in Section 1.2.2. Finally, Section 1.3 presents in more detail the contribution of each of the subsequent Chapters.

1.1 Inference for generative models

In this thesis, by “generative model” I mean a statistical model P on some space $\mathcal{Y} \subseteq \mathbb{R}^d$ from which it is possible to simulate $x \sim P$ but of which, in general, it is not possible to evaluate the density¹ $p(y)$ for a given observation $y \in \mathcal{Y}$. Works employing generative models rely on model simulations to compare distributions following a few principles (Mohamed and Lakshminarayanan, 2017). They can be grouped into two different categories according to the task they address and the features of the generative model they employ:

- **parameter estimation for simulator models:** a *simulator model* is a generative model assumed to depend on relatively few (at most a few hundred) physically meaningful parameters $\theta \in \Theta \subseteq \mathbb{R}^p$; I will write $P(\cdot|\theta)$ for the model and $p(\cdot|\theta)$ for its (inaccessible) density function; also, the *likelihood* of θ at y is the function $\theta \rightarrow p(y|\theta)$. Given the data $\mathbf{y}_{1:n} = (y_1, y_2, \dots, y_n)$, the interest is in estimating θ as its value conveys scientific information. Many works develop algorithms to approximate the Bayesian posterior distribution $\pi(\theta|\mathbf{y}_{1:n})$ by relying on model simulations (as direct sampling with Markov-Chain Monte Carlo is impossible due to missing likelihood). This class of strategies is called *Bayesian Likelihood-Free Inference (LFI)* (fewer works attempt to provide a point estimate $\hat{\theta}$, see for instance Cranmer et al., 2015; Kajihara et al., 2018). Typically, a simulator model consists of a computer code implementing a sequence of operations involving θ and a possibly varying number of random variables.
- **Generative modelling with generative networks:** this line of works, instead, consider a class of generative models $P_\phi, \phi \in \Phi \subseteq \mathbb{R}^q$, and aims to obtain a point estimate $\hat{\phi}$ such that $P_{\hat{\phi}}$ is as close as possible to observed (or *training*)

¹Unless otherwise stated, I will consider densities to be with respect to the Lebesgue measure.

data; the value $\hat{\phi}$ is not interesting in itself as ϕ does not have physical meaning. Typically, P_ϕ is induced by transforming a latent random variable (or noise) $Z \sim P_Z, Z \in \mathcal{Z} \subseteq \mathbb{R}^{d_z}$ with a function h_ϕ parametrised by a neural network with weights ϕ . Given a draw $z \sim P_Z$, $h_\phi(z)$ is a sample from the generative model P_ϕ . This construction is called *generative (neural) networks*. Here, a typical inference strategy uses draws from the generative network and gradients obtained by automatic differentiation to obtain $\hat{\phi}$ by optimising an objective function. Instead, works inferring a distribution on ϕ are rare (with an exception being Saatci and Wilson, 2017). By considering h_ϕ to additionally depend on a conditioning variable θ , a conditional generative distribution $P_\phi(\cdot|\theta)$ can be obtained.

The two categories identified above constitute an imperfect classification of recent research efforts: for instance, some works exploit gradient information in the first setup (Moreno et al., 2016), while others use generative neural networks to solve problems of interest to the first community (see, among others, Papamakarios and Murray, 2016; Papamakarios et al., 2019; Ramesh et al., 2022).

This thesis contributes with methods for both tasks above. Specifically, in Chapters 2 and 3, I give new Bayesian Likelihood-Free Inference methods for parameter estimation of simulator models. Instead, in Chapters 4 and 5, I contribute to generative modelling with a method to fit generative networks; in Chapter 4, interestingly, this is applied to solve a Bayesian Likelihood-Free Inference task. To better frame these contributions, Sections 1.1.1 and 1.1.2 below respectively review the literature on Bayesian Likelihood-Free Inference and generative networks, focusing on foundational works and recent developments relevant to the subsequent chapters.

1.1.1 Bayesian Likelihood-Free Inference

The quantity of interest in Bayesian inference is the Bayesian posterior:

$$\pi(\theta|y) = \frac{\pi(\theta)p(y|\theta)}{p(y)}; \quad (1.1)$$

typically, samples from the above can be obtained by Markov-Chain Monte Carlo (MCMC) methods (Robert and Casella, 2005); alternatively, an approximation can be obtained with Variational Inference (Blei et al., 2017). However, both approaches cannot be applied to directly target Eq. (1.1.1) when the likelihood $p(y|\theta)$ is inaccessible, as it is the case for a simulator model $P(\cdot|\theta)$. As mentioned above, Bayesian Likelihood-Free Inference (LFI) methods replace likelihood evaluation with simulations from the generative model and target approximations of the posterior in

Eq. (1.1.1). The largest research effort has gone towards the Approximate Bayesian Computation (ABC) family of algorithms (Beaumont, 2019; Lintusaari et al., 2017), which are based on exploring the parameter space by evaluating the similarity between observation and simulations; I discuss them in Section 1.1.1.1. Other algorithms replace the intractable likelihood with a surrogate estimated from model simulations (Drovandi et al., 2018; Thomas et al., 2020); those are discussed in Section 1.1.1.2. Finally, a recent class of algorithms has focused on exploiting neural networks to represent the likelihood or posterior and trained neural network weights from simulations from the model (Alsing et al., 2019; Papamakarios and Murray, 2016; Lueckmann et al., 2017; Papamakarios et al., 2019; Greenberg et al., 2019; Lueckmann et al., 2019; Radev et al., 2020; Durkan et al., 2020); I review some methods in this family in Section 1.1.1.3.

1.1.1.1 Approximate Bayesian Computation methods

Approximate Bayesian Computation (ABC) compares the observation y with simulations from the generative model; intuitively, the closer a simulation x is to y , the more likely the parameter value used to simulate x was to generate y . Specifically, the comparison is done by computing some summary statistics $s(y)$ and $s(x)$ for both observation and simulation, and checking $d(s(y), s(x)) < \epsilon$, for some distance d and a threshold ϵ . The simplest ABC algorithm is based on rejection sampling; a version generating a fixed number of draws with a prespecified ϵ can be found in Algorithm 1. Often, however, a computational budget is fixed and ϵ is determined as a quantile of the obtained distances between observation and simulations.

The rejection-ABC algorithm generates draws from the *ABC posterior*:

$$\pi^\epsilon(\theta|s(y)) \propto \pi(\theta) \int \mathbb{1}[d(s(y), s(x)) \leq \epsilon] p(x|\theta) dx, \quad (1.2)$$

where $\mathbb{1}[\cdot]$ denotes the indicator function². Intuitively, smaller values of ϵ lead to closer match to the observation and therefore to a better posterior approximation. Indeed, if $\epsilon \rightarrow 0$ and if s is a set of sufficient statistics³ for $P(\cdot|\theta)$, then the ABC posterior converges to the exact posterior Beaumont (2019). However, the rejection-ABC algorithm is inefficient for small ϵ as most simulations are rejected. More efficient ABC algorithms have therefore been developed, based for instance on MCMC (Marjoram

²In general, $\mathbb{1}[d(s(y), s(x)) \leq \epsilon]$ can be replaced with $\mathbb{K}_\epsilon(d(s(y), s(x)))$, where \mathbb{K}_ϵ is a kernel which goes to 0 for argument going to $\pm\infty$ and whose width decreases when ϵ decreases; see for instance Beaumont (2019).

³See Appendix A in Chapter 2 for the definition of sufficient statistics.

Algorithm 1 Rejection ABC (generating N draws from the ABC posterior).

Require: Prior distribution π , simulator model $P(\cdot|\theta)$, threshold ϵ , distance function d , statistics function s , observed data y .

```
1: while  $j = 1 \leq N$  do
2:   Draw  $\theta' \sim \pi$ 
3:   Simulate a dataset  $x$  from  $P(\cdot|\theta')$ 
4:   Compute some statistics  $s$  of the simulated and observed datasets
5:   if  $d(s(x), s(y)) < \epsilon$  then
6:     Set  $\theta_j \leftarrow \theta'$ 
7:      $j \leftarrow j + 1$ 
8:   else
9:     Discard  $\theta'$ 
10:  end if
11: end while
12: return  $\{\theta_j\}_{j=1}^N$ .
```

et al., 2003), Population Monte Carlo (Beaumont, 2010) or Sequential Monte Carlo (Del Moral et al., 2012). See also the review paper Beaumont (2019) and the Handbook of Approximate Bayesian Computation Sisson et al. (2018) for more background information.

Choosing s to be the identity function gives a well-grounded ABC algorithm. However, finding a simulation x that matches y for a fixed threshold ϵ becomes harder as the dimension of the data increases. For high-dimensional data, therefore, using a low-dimensional summary allows reducing ϵ at a lower computational cost. Even if sufficient statistics are unavailable (as is most often the case), this gain in algorithmic efficiency may outweigh the approximation induced by conditioning on non-sufficient statistics. The best trade-off between computational gain and loss of information depends on the considered model and the computational budget (in particular, it does not make sense to add more summaries if the approximation due to non-sufficiency is already as small as the Monte Carlo error). Possible ways of selecting a subset of existing summaries according to some principle are reviewed in Prangle (2018), which also discusses this issue in more detail. Other works instead focused on automatically extracting statistics from raw data; I review the prominent approaches in this second strand of the literature below. Other methods are surveyed in Prangle (2018).

Learning summary statistics for ABC In the seminal work Fearnhead and Prangle (2012), the authors argue for using the summary statistic $s(x) = \mathbb{E}[\theta|x]$ as that leads to an ABC posterior whose mean best matches the true posterior mean

under the ℓ_2 loss. $\mathbb{E}[\theta|x]$ is unknown; however, note that:

$$\mathbb{E}[\theta|x] = \min_s \mathbb{E}_{\theta \sim \pi} \mathbb{E}_{X \sim P(\cdot|\theta)} \|s(X) - \theta\|_2^2. \quad (1.3)$$

Therefore, the authors of Fearnhead and Prangle (2012) propose to generate parameter-simulation pairs $(\theta_j, x_j)_{j=1}^N$, $\theta_j \sim \pi(\theta)$, $x_j \sim P(\cdot|\theta_j)$, and to find

$$\hat{\beta} = \arg \min_{\beta} \frac{1}{N} \sum_{j=1}^N \|s_{\beta}(x_j) - \theta_j\|_2^2, \quad (1.4)$$

where s_{β} is a function in a set indexed by β . Eq. (1.1.1.1) is the empirical equivalent of Eq. (1.1.1.1). In Fearnhead and Prangle (2012), the authors considered s_{β} a linear function of β , so that solving Eq. (1.1.1.1) amounts to linear regression. In Jiang et al. (2017); Wiqvist et al. (2019); Akesson et al. (2021), s_{β} is instead parametrised by neural networks trained to minimize Eq. (1.1.1.1) by Stochastic Gradient Descent (SGD).

Other works have revisited the idea of defining the statistics function as the solution of an optimisation problem over a set of parameter-simulations pairs; different objective functions lead to summary statistics with different properties and theoretical motivations. For example, Chen et al. (2021) aims to maximize the mutual information between learnt statistics and parameters; Pacchiardi et al. (2020) aims to learn statistics s so that $\|s(x_i) - s(x_j)\| \approx \|\theta_i - \theta_j\|$ for all pairs of simulations and corresponding parameter values, by using distance learning techniques. My contribution in Chapter 2 can be placed in this line of research; there, I define summary statistics as the sufficient statistics of the best exponential family approximation to the unavailable likelihood.

An extension of the approach in Fearnhead and Prangle (2012) is presented in Forbes et al. (2021), where a mixture-of-Gaussian regression model is fit to the parameter-samples pair as a first approximation to the posterior. From this, the posterior expectation can be analytically obtained for each x , returning the approach in Fearnhead and Prangle (2012); additionally, the variance and other quantiles can be obtained and added to the set of summaries. Alternatively, it is possible to replace the distance between the summary statistics by a discrepancy between the Gaussian mixture approximation for the observation and the simulation, which amounts to using functional summary statistics.

Finally, Chen et al. (2021) proposes to depart from the two-step approach introduced by Fearnhead and Prangle (2012) by alternating steps of sequential ABC methods with improvements to the learnt statistics. Instead, Bharti et al. (2022)

consider a Bayesian experimental design framework in which an expert knowledge is sequentially used to select amongst a set of pre-specified summary statistics (which could be obtained with automatic methods). If the expert can exclude misspecified summaries, the resulting inference performs better in the presence of model misspecification.

ABC with statistical divergences To overcome the issue of non-sufficient statistics, the distance $d(s(y), s(x))$ in the ABC posterior (Eq. (1.1.1.1)) can be replaced by an empirical estimate of a discrepancy measure D between the true distribution P_0 and the model at the considered value of θ : $D(P_0||P(\cdot|\theta))$. In practice, an estimate of the above quantity requires multiple observations y_1, y_2, \dots, y_n from P_0 and multiple simulations x_1, x_2, \dots, x_m from $P(\cdot|\theta)$ for each considered θ , and consists of a function $\hat{D}(\{y_i\}_{i=1}^n, \{x_j\}_{j=1}^m)$ that converges to $D(P_0||P(\cdot|\theta))$ when $n, m \rightarrow \infty$.

Several choices of D and the associated empirical estimate have been considered: Park et al. (2016) considered the kernel Maximum Mean Discrepancy and Nguyen et al. (2020) the Energy Distance; however, their methods rely on assumptions on the moments of P_0 and may not produce reliable results if there are outliers in the data (Drovandi and Frazier, 2022). Outlier-robust methods are the γ -divergence of Fujisawa et al. (2021) and the Hellinger and Cramer von-Mises distances of Frazier (2020). Bernton et al. (2019) used the Wasserstein distance and Nadjahi et al. (2020) its sliced version; the Wasserstein distance and the Energy distance of Nguyen et al. (2020) are the only methods for which theoretical guarantees with fixed n and m have been given. Finally, Wang et al. (2021) uses a probabilistic classifier trained to distinguish between observations and simulations from the model at a given θ to provide alternative estimates of some divergences mentioned above; this extends the method in Gutmann et al. (2018), where the ABC distance was replaced by the accuracy of a classifier trained as in Wang et al. (2021). See Drovandi and Frazier (2022) for a comparison of some of these approaches in the case of univariate data.

1.1.1.2 Surrogate Likelihood methods

ABC uses an implicit estimate of the likelihood based on rejection sampling. Alternatively, an explicit likelihood approximation can be built by considering a parametric surrogate density and estimating its parameters using simulations from $P(\cdot|\theta)$.

Bayesian Synthetic Likelihood (BSL, Price et al., 2018) is the most popular instance of this idea: here, the unknown likelihood is replaced by a surrogate multivariate normal distribution for the statistics, whose mean vector μ_θ and covariance

matrix Σ_θ depend on θ . For each considered value of θ , estimates $\hat{\mu}_\theta$ and $\hat{\Sigma}_\theta$ are obtained from m simulations from $P(\cdot|\theta)$ and the density of the observations is evaluated. Then, this can be inserted into a Metropolis-Hastings MCMC scheme (Price et al., 2018), leading to a biased pseudomarginal MCMC (Andrieu et al., 2009) whose target depends on the number of simulations m and recovers the exact BSL posterior as $m \rightarrow \infty$.

Several variations of the surrogate likelihood have been proposed: Fasiolo et al. (2018) proposed a different likelihood estimator to be inserted in the MCMC scheme; An et al. (2019) employs a covariance matrix estimator with lasso regularisation which requires fewer simulations than the one employed in Price et al. (2018) to achieve similar performance; An et al. (2020) considers a semiparametric surrogate likelihood with a Gaussian copula. Other approaches are reviewed in Drovandi et al. (2018).

Some works instead focused on improving computational performance: for instance, Ong et al. (2018a) and Ong et al. (2018b) employ mean-field variational inference rather than pseudomarginal MCMC, and Picchini et al. (2022) advocates for the use of correlated pseudomarginal MCMC schemes in order to improve the mixing of the chains. The theoretical performances of BSL have also been explored: Frazier et al. (2022) establishes asymptotic properties, while Frazier and Drovandi (2021) and Frazier et al. (2021) studies they behaviour with misspecified models.

Another surrogate likelihood method is Ratio Estimation (Thomas et al., 2020), where the ratio between the marginal density of the data and the likelihood at a given θ is estimated using logistic regression; the estimate is then inserted into MCMC; more details are given in Section 2.2 in Chapter 2.

1.1.1.3 Neural network approximations

I review now some works employing normalising flows (Papamakarios et al., 2021) and other neural networks to perform Bayesian LFI; this section focusses on the case of continuous variables, as research into applying these methods to discrete ones is ongoing (see, for instance, Papamakarios et al., 2021).

Normalising flows Normalising flows (Papamakarios et al., 2021) are generative neural networks P_ϕ where the transformation h_ϕ is invertible and differentiable (a *diffeomorphism*). Therefore, the density of the distribution P_ϕ is given by the change-of-variable formula:

$$p_\phi(y) = p_Z(z) \left| \det J_{h_\phi^{-1}}(y) \right|, \quad z = h_\phi^{-1}(y), \quad (1.5)$$

where p_Z is the probability density of the latent variable (or noise) $Z \sim P_Z$ and $J_{h_\phi^{-1}}(y)$ is the Jacobian matrix of the inverse transformation evaluated in y . Evaluating the density requires explicitly computing the inverse h_ϕ^{-1} ; in contrast, drawing samples from the normalising flow employs h_ϕ (as for generative networks). The ability to cheaply compute the inverse is a stronger constraint than invertibility. As such, if sampling is not required, it is more convenient to directly represent the invertible function h_ϕ^{-1} by a neural network for which the inverse may be inaccessible or expensive. If both sampling and density evaluations are required, then the architecture is restricted to those providing efficient inverse computation (and the neural network can represent either h_ϕ or its inverse).

Further, cheaply evaluating the density in Eq. (1.1.1.3) requires the determinant of the Jacobian (which is a $d \times d$ matrix, where d is the number of components in y) to be computable at a small cost; for example, this can be done at a $\mathcal{O}(d)$ cost for transformations with a triangular Jacobian matrix, some of which are reviewed in Papamakarios et al. (2021).

Due to the availability of the density, normalising flows are versatile and can be applied to tasks such as density estimation and variational inference (see Section 6 in Papamakarios et al., 2021). In the case of density estimation, if samples $\{y_i\}_{i=1}^N$ from a distribution P_0 are available, a normalising flow can be trained by Maximum Likelihood Estimation (MLE):

$$\max_{\phi} \sum_{i=1}^N p_{\phi}(y_i),$$

which is the empirical problem corresponding to minimizing the forward KL divergence $D_{KL}(P_0 \| P_{\phi})$; note that this does not require drawing samples from P_{ϕ} .

To parametrise conditional probabilities, a conditioning variable θ can be introduced in the transformation h_ϕ : given θ and noise z , a sample from the conditional distribution can be obtained by $h_\phi(z; \theta)$ (Papamakarios et al., 2021). The conditional density can now be obtained by adapting Eq. (1.1.1.3) to:

$$p_{\phi}(y|\theta) = p_Z(z) \left| \det J_{h_\phi^{-1}(\cdot; \theta)}(y) \right|, \quad z = h_\phi^{-1}(y; \theta),$$

where the Jacobian and the inverse are referred to the first argument of the function.

Normalising flows for LFI Several works employed conditional normalising flows for Bayesian LFI. For instance, in Radev et al. (2020), the authors considered a normalising flow parametrising an approximate posterior density $\pi_{\phi}(\cdot|y)$ for all values

of y ; given parameter-simulation samples $(\theta_j, x_j)_{j=1}^N$, $\theta_j \sim \pi(\theta)$, $x_j \sim P(\cdot|\theta_j)$, the approximation is obtained by:

$$\arg \max_{\phi} \sum_{j=1}^N \pi_{\phi}(\theta_j|x_j), \quad (1.6)$$

which is the empirical equivalent of (Radev et al., 2020):

$$\arg \max_{\phi} \iint p(x, \theta) \log \pi_{\phi}(\theta|x) dx d\theta = \arg \min_{\phi} \mathbb{E}_{X \sim P} [D_{KL}(\Pi(\cdot|X) \|\Pi_{\phi}(\cdot|X))],$$

where $\Pi(\cdot|X)$ and $\Pi_{\phi}(\cdot|X)$ denote the true and approximate posterior distributions, $p(x, \theta)$ is the true joint density over observations and parameters, and P is the true marginal distribution over observations.

Therefore, the approach in Radev et al. (2020) amounts to generating a dataset of parameter-simulation pairs and training a normalising flow that provides an approximate posterior for any observation y with positive marginal probability density. As such, this method is said to be *amortised* over observations. Samples from the approximate posterior can be obtained by drawing from the normalising flow directly if the used architecture supports it.

However, the amortised approach is wasteful (in terms of the number of model simulations) if inference is required for a single value of y . In fact, most θ_j 's drawn from the prior may be in regions of low posterior density for that y and would provide little information to approximate the posterior. Sequential approaches (Papamakarios and Murray, 2016; Lueckmann et al., 2017; Greenberg et al., 2019) iteratively fit multiple posterior approximations, using the current one as proposal to sample new values of θ 's (and associated simulations x 's), from which the approximation for the next round is fit. Therefore, the ability to sample from the normalising flow is required. This strategy ensures that simulation resources are concentrated onto regions of high posterior density. However, naively minimizing the equivalent of Eq. (1.1.1.3) on draws from a proposal different from the prior leads to π_{ϕ} approximating the wrong target. To correct for this, Lueckmann et al. (2017) introduces importance weights in the minimization objective, while Papamakarios and Murray (2016) and Greenberg et al. (2019) rely on correction factors that are tractable when π_{ϕ} is a mixture of Gaussians; alternatively, Greenberg et al. (2019) uses a proposal supported on a finite set of discrete atoms, in which case the correction factor can be computed for any normalising flow. If, at each round, the atoms of the proposal are sampled from a distribution whose support covers the posterior, then the approximate posterior π_{ϕ} still converges to the exact one.

A complementary approach is to use normalising flows to approximate the likelihood function (Papamakarios et al., 2019; Lueckmann et al., 2019; Alsing et al., 2019). Denoting as $P_\phi(\cdot|\theta)$ the approximation to the statistical model and $p_\phi(\cdot|\theta)$ its corresponding density, ϕ is chosen to minimize $\mathbb{E}_{\theta \sim \tilde{\Pi}} [D_{KL}(P(\cdot|\theta) \| P_\phi(\cdot|\theta))]$ for a proposal distribution $\tilde{\Pi}$. As long as $\tilde{\Pi}$ has the same support as the prior and the model is well-specified, the above will be minimized when $p(x|\theta) = p_\phi(x|\theta)$ for all θ such that $\pi(\theta) > 0$. Therefore, no corrections are needed when a sequential approach is used and sample generation from the normalising flow is not required. On the other hand, sampling from the posterior requires now MCMC.

Wiqvist et al. (2021) and Glöckler et al. (2022) propose to exploit separate normalising flows to learn a likelihood and a posterior approximation at the same time; the likelihood is learnt sequentially as in Papamakarios et al. (2019) and Lueckmann et al. (2019) (so without correction approaches) and the explicit posterior approximation enables inference without MCMC.

Approaches based on other neural networks The architectures of normalising flows are strongly constrained. As such, it is difficult to design architectures suitable for high-dimensional and structured data (such as images); Chen et al. (2021) shows how, indeed, some of the methods discussed above perform poorly on such data. Therefore, one can abandon normalising flows and employ generative networks, which impose no constraints on h_ϕ . As explicit density evaluation is now impossible, training the neural network requires different strategies, as those discussed in Section 1.1.2. For instance, Ramesh et al. (2022) uses generative networks to represent the posterior, but this often leads to overconfident posterior approximations; in Chapter 4, I build on this work and employ a different training mechanism from the traditional one, yielding better calibrated posterior approximations.

Other works instead use classifier networks to approximate the ratio between the data marginal and the likelihood at all values of θ , which can then be inserted in MCMC or similar likelihood-based sampling methods (Hermans et al., 2020; Rozet and Louppe, 2021). This idea extends Thomas et al. (2020) and is analogue to the frequentist approach in Cranmer et al. (2015).

1.1.2 Inference for generative networks

In this Section, I review some fundamental methods to train generative networks. In contrast to Bayesian Likelihood-Free Inference (Section 1.1.1), methods for generative networks usually aim to obtain a point estimate $\hat{\phi}$ of the network weights. This is

done by minimizing an objective function by Stochastic Gradient Descent enabled by the automatic differentiation capabilities of neural networks.

Similarly to normalising flows, applying this method to a discrete setting is difficult due to the use of gradients to obtain $\hat{\phi}$. Few works have attempted to circumvent this issue (see, for instance, Montahaiei et al., 2021; Zhang et al., 2022); across this work, I will thus focus on generative networks for continuous data.

1.1.2.1 Generative Adversarial Networks

The original Generative Adversarial Networks Consider a generative network h_ϕ with associated probability distribution P_ϕ . Having observed $(y_i)_{i=1}^n$ from a distribution P_0 , the task is to adjust ϕ so that $P_\phi \approx P_0$ in some sense. The key idea in Goodfellow et al. (2014) is to introduce a *discriminator* (or critic) $c : \mathcal{Y} \rightarrow [0, 1]$ whose output expresses confidence in the input coming from the true distribution rather than being generated from P_ϕ . For fixed P_ϕ , the discriminator aims therefore to maximize

$$V(c, h_\phi) = \mathbb{E}_{Y \sim P_0} [\log c(Y)] + \mathbb{E}_{X \sim P_\phi} [\log(1 - c(X))],$$

which attributes a large penalty if $c(Y)$ is close to 0 for observed Y , and similarly if $c(X)$ is close to 1 for simulated X . For fixed ϕ , the optimal discriminator (see Proposition 1 in Goodfellow et al., 2014 and also Gutmann and Hyvärinen, 2012) is:

$$c^*(\phi)(x) = \frac{p_0(x)}{p_0(x) + p_\phi(x)},$$

which, by Theorem 1 in Goodfellow et al. (2014), makes

$$V(c^*(\phi), h_\phi) = -\log(4) + 2 \cdot D_{JS}(P_0 || P_\phi),$$

where D_{JS} is the Jensen-Shannon divergence. Therefore, by defining

$$\phi^* = \arg \min_{\phi} V(c^*(\phi), h_\phi) = \arg \min_{\phi} \sup_c V(c, h_\phi), \quad (1.7)$$

P_{ϕ^*} is the closest model to P_0 in Jensen-Shannon divergence (and $P_{\phi^*} = P_0$ if the model is well specified).

In practice, c is parametrised by a discriminator neural network c_ψ (ψ being the weights of the neural network), so that the problem in Eq. (1.1.2.1) is replaced by the surrogate:

$$\min_{\phi} \max_{\psi} \mathbb{E}_{Y \sim P_0} [\log c_\psi(Y)] + \mathbb{E}_{X \sim P_\phi} [\log(1 - c_\psi(X))]. \quad (1.8)$$

The above problem is typically tackled using stochastic gradient optimisation, alternating one (or a few) steps over ψ with one (or a few) steps for ϕ . At each gradient step, the expectations over P_0 and P_ϕ are estimated by, respectively, using a batch of observations and by generating samples from the generative network. This method is termed *Generative Adversarial Network (GAN)* as the generative network P_ϕ aims to trick the discriminator to believe that the generated samples come from the true distribution. Algorithm 2 shows a possible training algorithm, where a single element of the observed dataset and a single draw from P_ϕ are used in each gradient optimisation step.

Algorithm 2 GAN training.

Require: Generative network h_ϕ , critic network c_ψ , learning rates ϵ, γ .

for each observation y_i **do**

 Sample $z \sim Q$

 Obtain $\hat{x}_i^\phi = h_\phi(z)$

 Set $\psi \leftarrow \psi + \gamma \cdot \nabla_\psi \left[\log c_\psi(y_i) + \log(1 - c_\psi(\hat{x}_i^\phi)) \right]$

 Set $\phi \leftarrow \phi - \epsilon \cdot \nabla_\phi \left[\log(1 - c_\psi(\hat{x}_i^\phi)) \right]$

end for

Other adversarial training approaches Inspired by the original GAN approach introduced in Goodfellow et al. (2014), many works develop related approaches maintaining the adversarial framework but modifying the training objective and relaxing the assumption of the discriminator output being in $[0, 1]$ (Nowozin et al., 2016; Arjovsky et al., 2017; Bellemare et al., 2017; Li et al., 2017). Some works introduce additional regularization terms in the objective to help with training (Gulrajani et al., 2017; Li et al., 2017; Arjovsky and Bottou, 2017; Grover et al., 2018), while others focus on designing generator and discriminator architectures suitable for specific tasks (Isola et al., 2017; Yoon et al., 2019; Koochali et al., 2021; Ravuri et al., 2021; Xing, 2021). See Gui et al. (2021) for a recent review.

In Appendix B.1 in Chapter 5, I overview two popular extensions of the original GAN: f -GAN Nowozin et al. (2016) replaces the Jensen-Shannon divergence with a generic f -divergence; by exploiting a variational formulation of the f -divergence relying on the Fenchel conjugate of convex functions, they obtain an adversarial formulation similar to Eq. (1.1.2.1). In Arjovsky et al. (2017), instead, a variational formulation of the 1-Wasserstein distance is used to derive an adversarial training objective, called Wasserstein GAN (or WGAN); here, the discriminator has real (scalar)

output but is constrained to have a bounded Lipschitz constant, which is achieved by clipping the weights to a given range.

Li et al. (2015) and Dziugaite et al. (2015) instead train generative networks to minimize an unbiased estimate of the Maximum Mean Discrepancy (MMD) between P_0 and P_ϕ . The unbiased estimate is obtained by applying a kernel to observations and draws from P_ϕ without introducing a discriminator network. To improve performance, some works (Li et al., 2017; Sutherland et al., 2017) compute the kernel on the embedding induced by an additional NN trained to maximize the MMD; this again amounts to an adversarial setting. I discuss this in more detail in Appendix B.1.3 in Chapter 5.

Conditional setting All the methods mentioned above can be extended to settings where the observed data consist of pairs of variables, say $(y_i, \theta_i)_{i=1}^n$, where θ_i is a conditioning variable. As mentioned at the beginning of Sec. 1.1, a conditional generative distribution $P_\phi(\cdot|\theta)$ is obtained using a neural network $h_\phi : \mathcal{Z} \times \Theta \rightarrow \mathcal{Y}$, where the first input is the noise variable and the second is the conditioning variable. A sample from $P_\phi(\cdot|\theta)$ is obtained by computing $h_\phi(z, \theta)$ for $z \sim P_z$. Similarly, the discriminator network will, in general, take as input the (generated or real) y and the value θ associated to it.

The GAN training objective in Eq. (1.1.2.1), for instance, generalizes to:

$$\min_{\phi} \max_{\psi} \mathbb{E}_{\Theta \sim \Pi} [\mathbb{E}_{Y \sim P_0(\cdot|\Theta)} [\log c_\psi(Y, \Theta)] + \mathbb{E}_{X \sim P_\phi(\cdot|\Theta)} [\log(1 - c_\psi(X, \Theta))]] ,$$

where Π is the true marginal distribution over the conditioning variable Θ . As before, this problem can be related to the minimization of the Jensen-Shannon divergence between the joint true and generative distributions over Θ and Y . A similar procedure can be adopted for f -GAN and WGAN; see Appendix B.1 in Chapter 5 for more details.

Alternating optimization and mode collapse Solving the min-max problem of Eq. (1.1.2.1) by alternating optimisation over ψ and ϕ is difficult: for instance, the two neural networks may fail to converge to an equilibrium. Thus, it is necessary to carefully tune the number of optimisation steps to apply to the generator before switching to the discriminator (and back); additionally, learning rates and other hyperparameters for two neural networks must be chosen. Therefore, a substantial research effort has been dedicated to improving adversarial training; see Salimans et al. (2016) for a review.

Furthermore, samples from generative adversarial networks are often realistic but have been observed (Goodfellow, 2016; Isola et al., 2017; Bellemare et al., 2017; Arora et al., 2018; Richardson and Weiss, 2018) not to represent the full variability of the data (in the conditional case, generative networks sometimes ignore the noise variable altogether, degenerating into a deterministic distribution, see Isola et al., 2017); this phenomenon is termed *mode collapse*.

Some works attempted explaining mode collapse by theoretically studying adversarial training. As observed in Arora et al. (2017), theoretical guarantees on adversarial training assume infinitely many training data and an optimal discriminator; clearly, that is not the case in practice. For instance, Arora et al. (2017) shows how, with a discriminator with finite capacity (even with infinitely many data), the objectives of GAN and WGAN cannot distinguish between the data distribution and a distribution with small support. Additionally, the stochastic gradients used to train the generator are biased as they are obtained with a suboptimal discriminator trained on finite data (Bellemare et al., 2017; Bińkowski et al., 2018); Bellemare et al. (2017) shows an instance where this can lead to mode collapse.

In some applications, generating realistic samples is more important than capturing the full (conditional) data distribution. However, the possibility of mode collapse makes generative adversarial networks ill-suited in settings where reliable uncertainty quantification is paramount. In Chapters 4 and 5, I discuss an adversarial-free training method for generative networks that yields better calibrated generative distributions and apply it to Bayesian Likelihood-Free Inference and probabilistic forecasting.

1.2 Background

In this Section, I give background information on the Generalized Bayesian inference framework in Section 1.2.1 and on Scoring Rules in Section 1.2.2, on which I will rely to introduce new methods in Chapters 2 to 5.

1.2.1 Generalised Bayesian inference

In recent years, ways to update probability distributions generalising Bayesian inference have been proposed. These approaches retain some features of Bayesian inference, while discarding others which can be inconvenient in specific settings (for both computational and fundamental reasons). In this Section, I overview a generalisation which Chapter 3 builds upon.

1.2.1.1 Properties and issues of standard Bayesian inference

Recall the Bayesian posterior introduced in Eq. (1.1.1):

$$\pi(\theta|\mathbf{y}_{1:n}) = \frac{\pi(\theta)p(\mathbf{y}_{1:n}|\theta)}{p(\mathbf{y}_{1:n})} = \frac{\pi(\theta)\prod_{i=1}^n p(y_i|\theta)}{p(\mathbf{y}_{1:n})},$$

where π reflects the prior belief in θ , $p(y|\theta)$ is the likelihood of θ at y and $\mathbf{y}_{1:n} = (y_1, y_2, \dots, y_n)$. Due to its multiplicative structure, the posterior is invariant to re-ordering of $\mathbf{y}_{1:n}$. Furthermore, the same distribution is recovered when inference is performed in a two-step procedure, with a subset of $\mathbf{y}_{1:n}$, say $\mathbf{y}_{1:m} = (y_1, \dots, y_m)$, $m < n$, used to update the prior $\pi(\theta)$ to $\pi(\theta|\mathbf{y}_{1:m})$ and the remaining data $\mathbf{y}_{m+1:n} = (y_{m+1}, \dots, y_n)$ then used to update $\pi(\theta|\mathbf{y}_{1:m})$ to $\pi(\theta|\mathbf{y}_{1:n})$. This property is called (*order-*)*coherence* (Bissiri et al., 2016).

Assume now that the data $\mathbf{y}_{1:n}$ are generated by a distribution P_0 with density p_0 and that there exists a parameter value θ_0 such that $p_0(\cdot) = p(\cdot|\theta_0)$ (that is, the model is *well specified*). As n increases, under some assumptions (see for instance Section 4.1.2 in Ghosh et al., 2006), the posterior converges almost surely to a multivariate normal distribution centred in θ_0 , with covariance matrix decreasing as $1/n$ and asymptotically equivalent to the sampling covariance of the Maximum Likelihood Estimator (MLE) of θ . This result is known as a Bernstein-von Mises theorem.

Therefore, for a well-specified model, the Bayesian posterior provides a coherent strategy to accumulate information about the true parameter value θ_0 (to which it converges in the limit of infinite data); further, credible regions obtained from the posterior have correct asymptotic coverage.

When the model is misspecified, there is no true parameter value; still, asymptotic normality of the Bayesian posterior holds: under some assumptions (Kleijn and van der Vaart, 2012), the posterior concentrates around

$$\theta^* = \arg \min_{\theta} D_{\text{KL}}(P_0||P(\cdot|\theta)), \quad (1.9)$$

where D_{KL} is the Kullback-Leibler (KL) divergence. θ^* is also called “pseudo-truth” and is the parameter value to which the MLE converges as $n \rightarrow \infty$; however, the sampling covariance of the MLE is now asymptotically different from the covariance matrix of the posterior, so that posterior credible regions do not have correct asymptotic coverage of the pseudo-truth.

Therefore, in a misspecified setting, the Bayesian posterior accumulates information about θ^* defined by the KL divergence. As there is no true parameter value, this choice of divergence is arbitrary and the pseudo-truth could as well be defined

via a different one. Furthermore, the KL divergence leads to undesirable behaviour in some cases: for instance, consider the outlier contamination model

$$p_0(x) = 0.1 h(x) + 0.9 p(x|\theta_0),$$

where $h(x)$ is a density with heavier tails than $p(x|\theta_0)$. As the KL gives large importance to the tail behaviour of distributions, θ^* can be far away from θ_0 (Jewson et al., 2018). Other divergences may, instead, be more robust to outliers in the data and therefore produce parameter estimates closer to θ_0 .

In Section 1.2.1.2 below, I discuss a generalised Bayesian posterior which learns about parameter values minimizing different quantities from the KL divergence.

1.2.1.2 Loss-based generalised Bayesian inference

In place of θ^* defined in Eq. (1.2.1.1), consider:

$$\theta_\ell^* = \arg \min \int \ell(\theta, y) p_0(y) dy,$$

where the “parameter” and observation values θ and y are related by a loss function $\ell(\theta, y)$. The aim is to update an original belief π_ℓ for θ_ℓ^* using observations $\mathbf{y}_{1:n} = (y_1, \dots, y_n)$. The only coherent (in the sense discussed in Sec. 1.2.1.1) way to do so requires the prior π_ℓ to be specified independently of the data and defines the posterior belief as (Bissiri et al., 2016):

$$\pi_{\ell,w}(\theta|\mathbf{y}_{1:n}) \propto \pi_\ell(\theta) \cdot \exp\left(-w \sum_{i=1}^n \ell(\theta, y_i)\right), \quad (1.10)$$

where $w > 0$ is a free scalar that balances the influence of loss and prior terms⁴. This is in general different from using $p(y|\theta) = e^{-w \cdot \ell(\theta, y)}$ in the standard posterior, as the latter can be unnormalized with respect to y .

A Bernstein-von Mises result can be obtained for the generalised posterior, under some assumptions on ℓ (see Miller, 2021): with an increasing number of observations, the general posterior is asymptotically normal centred in θ_ℓ^* and has an asymptotic covariance matrix generalising the Fisher information. Therefore, the generalised posterior provides a coherent strategy to learn about θ_ℓ^* in the same way as the original posterior learnt about θ^* in Eq. (1.2.1.1).

⁴In some works (Jewson et al., 2018; Matsubara et al., 2022b), the authors define a generalised posterior $\pi_D(\theta) \cdot \exp(-w D_n(\theta, \mathbf{y}_{1:n}))$ for a loss $D_n(\theta, \mathbf{y}_{1:n})$ depending on θ and the full dataset $\mathbf{y}_{1:n}$, which recovers Eq. (1.2.1.2) when $D_n(\theta, \mathbf{y}_{1:n})$ is additive in $\mathbf{y}_{1:n}$. If this is not the case, this posterior is not coherent in the sense of Sec. 1.2.1.1 and cannot be motivated in the same way as Eq. (1.2.1.2).

The appropriate ℓ depends on the problem at hand: for instance, some choices may provide robustness to outliers (Matsubara et al., 2022b), thus avoiding the problem mentioned in Section 1.2.1.1. Computational benefits can also be obtained: in Chapter 3 I introduce a generalised Bayesian formulation for generative models (Section 1.1.1), while Matsubara et al. (2022b) and Matsubara et al. (2022a) do the same for continuous and discrete likelihoods with an intractable normalising constant. In all of these cases, the standard Bayesian posterior is inaccessible to traditional MCMC, while the generalised one is amenable to computation.

In general, the belief update in Eq. (1.2.1.2) does not even require to specify a probabilistic model, as θ is defined via its relation to the data through the loss ℓ . Choosing $\ell(\theta, y) = \|\theta - y\|_1$, for example, allows learning about the median of p_0 .

Setting $w = 1$ and $\ell(\theta, y) = -\log p(y|\theta)$ in Eq. (1.2.1.2) recovers the standard Bayesian posterior. Some works advocate for using $w < 1$ and $\ell(\theta, y) = -\log p(y|\theta)$ for some forms of misspecification (Grünwald and Van Ommen, 2017; Holmes and Walker, 2017); the specific value of w must be chosen according to some criteria (some are reviewed in Wu and Martin, 2020).

For generic ℓ , w loses the reference value $w = 1$ that, with $\ell(\theta, y) = -\log p(y|\theta)$, is optimal for well-specified models (Zellner, 1988). Several principled approaches to tune w have been developed in this case (Bissiri et al., 2016; Lyddon et al., 2019; Syring and Martin, 2019; Matsubara et al., 2022a). A brief review can be found in Section 2.4 in Chapter 3.

If we set w by using (part of) the observed data, this breaks coherence of the overall procedure for updating beliefs. If coherence is required, a possible way to restore it is by fixing w using, for instance, simulated data if θ is associated to a statistical model.

1.2.2 Scoring Rules

In this Section, I give background information on Scoring Rules, upon which my proposed methods in Chapters 2 to 5 rely.

In statistics, the suitability of a distribution P in modelling an observation $y \in \mathcal{Y} \subseteq \mathbb{R}^d$ is commonly evaluated through its density function $p(y)$. For parameter inference with a model $P(\cdot|\theta)$, for instance, the Bayesian paradigm employs the *likelihood* $p(y|\theta)$ to define the posterior as in Eq. (1.1.1), while the frequentist paradigm maximizes $p(y|\theta)$ with respect to θ to obtain the Maximum Likelihood Estimator.

Scoring Rules are more general ways to assess how well a probability distribution P describes an observation y . Specifically, for a Scoring Rule (SR) S , $S(P, y)$ represents

a “penalty” received when P is used to model y . Of course, not all such functions are useful. I thus recall here some properties: a SR S is said to be proper for the class of distributions $\mathcal{P}(\mathcal{Y})$ on the space \mathcal{Y} if its expected value⁵ $S(P, Q) := \mathbb{E}_{Y \sim Q} S(P, Y)$ is minimized when $P = Q$:

$$S(Q, Q) \leq S(P, Q) \quad \forall P, Q \in \mathcal{P}(\mathcal{Y}).$$

If additionally $P = Q$ is the only minimum, then S is said to be *strictly proper*:

$$S(Q, Q) < S(P, Q) \quad \forall P, Q \in \mathcal{P}(\mathcal{Y}) \text{ s.t. } P \neq Q.$$

A detailed survey of the properties of SRs is available in Gneiting and Raftery (2007). For a proper S , the quantity

$$D(P, Q) := S(P, Q) - S(Q, Q) \geq 0 \tag{1.11}$$

is called the *divergence* function associated with S . $P = Q$ implies $D(P, Q) = 0$ but the converse only holds when S is strictly proper. In this case, D is a *statistical divergence*, following the common nomenclature in the statistics literature⁶.

Let us now consider a class of distributions $P_\phi, \phi \in \Phi$, and define:

$$\phi^\star := \arg \min_{\phi \in \Phi} S(P_\phi, Q); \tag{1.12}$$

if Q belongs to the model class (i.e. the model is well specified), any strictly proper SR S yields $P_{\phi^\star} = Q$. However, when the model is misspecified, the solution to Eq. (1.2.2) depends on the specific SR used.

In practice, however, we may only have access to observations (y_1, y_2, \dots, y_n) , $y_i \sim Q$. We can then consider:

$$\hat{\phi} := \arg \min_{\phi \in \Phi} \frac{1}{n} \sum_{i=1}^n S(P_\phi, y_i), \tag{1.13}$$

which is the empirical equivalent of Eq. (1.2.2). Under some assumptions, it can be shown (Dawid and Musio, 2014; Dawid et al., 2016) that $\hat{\phi}$ converges to ϕ^\star as $n \rightarrow \infty$ and is asymptotically normally distributed. These results generalise the standard asymptotics of the Maximum Likelihood Estimator (MLE), which is recovered from Eq. (1.2.2) by using $S(P_\phi, y) = -\log p_\phi(y)$ (the *log-score*, see Section 1.2.2.2).

⁵Following previous literature, I overload notation by using S to denote both the SR and its expected value according to whether its second argument is an observation or a distribution.

⁶Differently from the common notation in statistics, the second argument to D , rather than the first, is here the true distribution. Additionally, some statistical divergences (such as the Wasserstein distance) cannot be decomposed as in Eq. (1.2.2); see Section 2.2 in Gneiting et al. (2007).

In Chapters 4 and 5, I use an objective related to Eq. (1.2.2) to train generative networks. Analogously to how Eq. (1.2.2) generalises the MLE, in Chapter 3 I generalise standard Bayesian inference based on the log-score to other Scoring Rules, by exploiting the construction presented in Section 1.2.1.2.

1.2.2.1 Scoring Rules and probabilistic forecasting

Above, I discussed the use of scoring rules to address parameter estimation problems; however, Scoring Rules originated in the context of probabilistic forecasting (Gneiting and Katzfuss, 2014), which is the task of producing probabilistic statements for future events. Forecasts can, for example, assume the form of credible intervals or full probability distributions.

In probabilistic forecasting, Scoring Rules are used to evaluate (or *verify*) forecasts once the actual value of the predicted quantity becomes available.

Using strictly proper Scoring Rules ensures that forecasters issue honest forecasts (Winkler, 1977; Dawid, 2006; Gneiting et al., 2007): in fact, if a forecaster has a subjective probabilistic belief Q for a future observation but provides a forecast P , the expected penalty she believes to receive is $S(P, Q) = \mathbb{E}_{Y \sim Q} S(P, Y)$, which is minimized if and only if she chooses $P = Q$ (by Eq. 1.2.2).

Additionally, the average score obtained by evaluating a set of probabilistic forecasts with a proper SR can be decomposed into separate contributions related to *calibration* and *resolution* of the forecasts. Calibration measures the degree to which probabilistic statements obtained from the forecasts match the long-run frequency of the realised values. Several definitions are available (Gneiting et al., 2007; Bröcker, 2009); for instance, 50% credible regions obtained from *probabilistically* calibrated forecasts would contain the realisation 50% of the times⁷. Resolution instead relates to how much the distribution of the observations differs when conditioned on different forecast values (Stephenson, 2012); it can be intended as the ability of the forecasts to sort events into groups that differ from each other (Wilks, 2019). Other attributes of forecast performance exist (Wilks, 2019; Stephenson, 2012); notably, *sharpness* measures the concentration of forecasts and is independent of the realisations (Gneiting et al., 2007); for continuous variables, it can be measured by, for example, the width of 90% credible intervals of the forecast distribution. For perfectly calibrated forecasts, sharpness and resolution are identical concepts.

⁷The notions of calibration of probabilistic forecasts are related but different from those for inference procedures returning a distribution on the parameters of a statistical model; see Cockayne et al. (2022) for a detailed discussion.

In Chapter 5, I train the weights of generative networks by minimizing an average SR over a set of forecasts; therefore, this bridges the two perspectives of forecast verification and parameter estimation. In the following, I give an example of calibration-resolution decomposition. Dawid (2006) and Bröcker (2009) give a generic decomposition for proper SRs for binary and categorical forecasts respectively; I am unaware of a similar result concerning SRs for continuous forecasts.

Calibration-resolution decomposition of Brier score for binary forecasts

Consider the task of forecasting binary observations $y_i \in \{0, 1\}$, $i = 1, 2, \dots, N$; for each i , the probabilistic forecast amounts to a subjective probability p_i for $y_i = 1$ stated before y_i is observed. The Brier Score (Dawid, 2006; Gneiting and Raftery, 2007) is a strictly proper SR for binary probabilistic forecasting and is defined by:

$$S(p, y) = (y - p)^2.$$

Assume now, for simplicity, that the forecasts p_i take on a finite set of values \tilde{p}_j , $j = 1, 2, \dots, M$; forecasting therefore consists of *sorting* the events in different groups and *labelling* them with probabilities \tilde{p}_j (Dawid, 2006). In this case, the average Brier Score can be written as:

$$\bar{S} := \frac{1}{N} \sum_{i=1}^N (y_i - p_i)^2 = \frac{1}{N} \sum_{j=1}^M \sum_{i:p_i=\tilde{p}_j} (y_i - \tilde{p}_j)^2 = \frac{1}{N} \sum_{j=1}^M n_j [\rho_j (1 - \tilde{p}_j)^2 + (1 - \rho_j) \tilde{p}_j^2],$$

where $n_j = \#\{i : p_i = \tilde{p}_j\}$ is the size of the j -th group and $\rho_j = \#\{i : p_i = \tilde{p}_j, y_i = 1\}/n_j$. Thus:

$$\bar{S} = S_1 + S_2, \text{ where } S_1 := \frac{1}{N} \sum_{j=1}^M n_j (\rho_j - \tilde{p}_j)^2, \quad S_2 := \frac{1}{N} \sum_{j=1}^M n_j \rho_j (1 - \rho_j).$$

$S_1 \geq 0$ with equality if and only if, for all j 's, $\tilde{p}_j = \rho_j$: in plain terms, this requires that, in the j -th group, exactly a fraction \tilde{p}_j has realisation $y_i = 1$. Therefore, it is a measure of the calibration of the forecasts.

S_2 , instead, depends on how the frequency of $y_i = 1$ varies among the different groups, but not on the specific probability \tilde{p}_j assigned to each of them; it decreases with increasing group homogeneity and is minimized whenever each group contains only $y_i = 1$ or $y_i = 0$ ⁸. Thus, S_2 penalises poor resolution (Dawid, 2006).

⁸As $\rho(1 - \rho)$ is concave and symmetric around $\rho = 0.5$ and is minimized by $\rho = 1$ or $\rho = 0$, which are achieved by perfectly homogeneous groups.

The average score \bar{S} , thus, trades off calibration with resolution. For calibrated forecasts, additionally, resolution corresponds to sharpness. In fact, if $\rho_j = \tilde{p}_j$:

$$NS_2 = \sum_{j=1}^M n_j \tilde{p}_j (1 - \tilde{p}_j),$$

which is independent of the realised values and only depends on the forecast via n_j . As $\sum_{j=1}^M n_j = N$, S_2 is minimized by forecasts that allocate all events to groups with \tilde{p}_j 's closer to 0 or 1; this amounts therefore to producing sharp forecasts.

1.2.2.2 Examples of Scoring Rules for continuous distributions

I report here some examples of strictly proper Scoring Rules, focussing on the case of continuous variables. Examples for different sample spaces can be found in Gneiting and Raftery (2007). Notice how some SRs are defined through the density p of P (the log-score and the Hyvarinen score), others through its Cumulative Density Function (the Continuous Ranked Probability Score), and others through expectations (the Energy and Kernel Scores).

Log-score The *log-score* is the negative log-density p of the distribution P :

$$S_{\log}(P, y) = -\log p(y);$$

it is a strictly proper SR and its corresponding divergence is the Kullback-Leibler divergence. The log-score is the only proper SR that only evaluates the density at the observation value y (it is thus said to be *local*, Parry et al., 2012).

Hyvarinen score The Hyvarinen score involves the first and second derivatives of the density p of P :

$$S_H(P, y) = \Delta \log p(y) + \frac{1}{2} \|\nabla \log p(y)\|^2, \quad (1.14)$$

where ∇ represents the gradient vector and Δ the Laplace operator. Under some integrability assumptions (see Section 3.2 in Chapter 2), integration by parts yields the following form for the associated divergence:

$$D_F(P, Q) = \frac{1}{2} \int q(y) \|\nabla \log q(y) - \nabla \log p(y)\|^2 dy,$$

which is usually termed ‘‘Fisher divergence’’.

As it depends only on the first and second derivatives of the log-density, the Hyvarinen score in Eq. (1.2.2.2) is invariant to multiplying p by a scalar value $c > 0$;

as such, it can be computed without knowing the normalising constant of p . In Chapter 2, I rely on this property and exploit the Hyvärinen score to fit a parametric density with unknown normalising constant to some data; this procedure is called *score matching* (Hyvärinen, 2005).

Continuous Ranked Probability Score (CRPS) For a continuous scalar variable, the CRPS (Gneiting and Raftery, 2007) is defined by considering the cumulative distribution function F_P of P and by computing:

$$S_{\text{CRPS}}(P, y) = \int_{-\infty}^{\infty} (F_P(x) - \mathbb{1}\{x \geq y\})^2 dx. \quad (1.15)$$

In general, analytically computing the integral in Eq. (1.2.2.2) is impractical, except in simple cases such as Gaussian distributions. However, the following alternative formulation (Eq. 17 in Székely and Rizzo, 2005) can be estimated using samples from P :

$$S_{\text{CRPS}}(P, y) = 2 \cdot \mathbb{E}[\|X - y\|] - \mathbb{E}[\|X - X'\|], \quad X \perp\!\!\!\perp X' \sim P. \quad (1.16)$$

From Eq. (1.2.2.2), it is clear how the CRPS is a specific case of the Energy Score (introduced below and used in Chapters 3 to 5) for scalar variables. The CRPS is proper for all probability distributions and is strictly proper for Borel probability distributions with finite first moment (Gneiting and Raftery, 2007). Additionally, the CRPS is the integral of the Brier score (Eq. 1.2.2.1) for binary forecasts of the event $\{x \geq y\}$ at all threshold values x . Calibration-resolution decompositions for the CRPS analogue to that given in Section 1.2.2.1 for the Brier score are given in Hersbach (2000) and Candille and Talagrand (2005).

Energy and Kernel Scores In Chapters 3 to 5, I extensively employ two Scoring Rules known as Energy and Kernel Scores. The Energy Score is given by:

$$S_{\text{E}}^{(\beta)}(P, y) = 2 \cdot \mathbb{E}[\|X - y\|_2^\beta] - \mathbb{E}[\|X - X'\|_2^\beta], \quad X \perp\!\!\!\perp X' \sim P,$$

where $\beta \in (0, 2)$. This is a strictly proper Scoring Rule for the class of probability measures P for which $\mathbb{E}_{X \sim P} \|X\|^\beta < \infty$ (Gneiting and Raftery, 2007) and is a generalisation of the CRPS to multivariate random variables. The related divergence is the square of the energy distance, which is a metric between probability distributions (Rizzo and Székely, 2016; see Appendix C.1 in Chapter 3)⁹.

⁹The probabilistic forecasting literature (Gneiting and Raftery, 2007) uses a different convention for the Energy Kernel Scores, which amounts to multiplying my definitions by 1/2. I follow here the convention used in the statistical inference literature (Rizzo and Székely, 2016; Chérif-Abdellatif and Alquier, 2020; Nguyen et al., 2020).

The kernel Score (Gneiting and Raftery, 2007) is instead defined as:

$$S_k(P, y) = \mathbb{E}[k(X, X')] - 2 \cdot \mathbb{E}[k(X, y)], \quad X \perp\!\!\!\perp X' \sim P,$$

where $k(\cdot, \cdot)$ is a positive definite kernel. The corresponding divergence is the squared Maximum Mean Discrepancy (MMD, Gretton et al., 2012) relative to k (see Appendix C.2 in Chapter 3). The Kernel Score is proper for the class of probability distributions for which $\mathbb{E}[k(X, X')]$ is finite (by Theorem 4 in Gneiting and Raftery, 2007). Additionally, it is strictly proper under conditions that ensure that the MMD is a metric (see Appendix C.2 in Chapter 3). These conditions are satisfied, among others, by the Gaussian kernel:

$$k(x, y) = \exp\left(-\frac{\|x - y\|_2^2}{2\gamma^2}\right),$$

in which γ is a scalar bandwidth. Notice that the Energy Score is recovered from the Kernel Score by setting $k(x, y) = -\|x - y\|_2^\beta$ (Gneiting and Raftery, 2007).

Assume we have draws $\mathbf{x}_j \sim P, j = 1, \dots, m$. In this case, for the Energy Score, unbiased estimates can be obtained by unbiasedly estimating the expectations in $S_E^{(\beta)}(P, x)$:

$$\hat{S}_E^{(\beta)}(\{x_j\}_{j=1}^m, y) = \frac{2}{m} \sum_{j=1}^m \|x_j - y\|_2^\beta - \frac{1}{m(m-1)} \sum_{\substack{j,k=1 \\ k \neq j}}^m \|x_j - x_k\|_2^\beta.$$

Similarly, we obtain an unbiased estimate of $S_k(P, x)$ by:

$$\hat{S}_k(\{x_j\}_{j=1}^m, y) = \frac{1}{m(m-1)} \sum_{\substack{j,k=1 \\ k \neq j}}^m k(x_j, x_k) - \frac{2}{m} \sum_{j=1}^m k(x_j, y).$$

1.3 Contributions and thesis outline

This integrated thesis contains four independent works in their complete form, each addressing open problems with inference methods for generative models. Specifically, Chapter 2 designs a new method for learning summary statistics for Approximate Bayesian Computation. Chapter 3 develops a novel outlier-robust posterior for Likelihood-Free Inference based on Scoring Rules and generalised Bayesian inference. Chapters 4 and 5 train generative neural networks to minimize Scoring Rules, yielding better calibrated generative distributions than the adversarial approach. This enables using generative networks for Bayesian Likelihood-Free Inference (Chapter 4)

and probabilistic forecasting (Chapter 5). In the rest of this Section, I briefly overview the objective, the methodology and the significance of each chapter. In Chapter 6, I provide a summary of the work and outline directions for future exploration.

1.3.1 Score Matched Neural Exponential Families for Likelihood-Free Inference

In Chapter 2, I consider the task of learning summary statistics for Approximate Bayesian Computation (ABC, see Sec. 1.1.1.1). To this aim, recall that the most general family of distributions with sufficient statistics of fixed size¹⁰ is the exponential family $p(y|\theta) \propto \exp(\eta(\theta)^T f(y))h(y)$, where η and f are the natural parameters and sufficient statistics of the exponential family and h is its base measure. Therefore, I propose to approximate the intractable likelihood with an exponential family and use f as summary statistics in ABC.

In practice, I represent f and η with two neural networks, which respectively receive as input data and parameters (h is represented by an additional output of the same neural network used for f). I then fit neural network weights using parameter-simulation pairs $(\theta_j, x_j)_{j=1}^N$ $\theta_j \sim \pi(\theta)$, $x_j \sim P(\cdot|\theta_j)$, as in the regression approach to learn summary statistics in Fearnhead and Prangle (2012); Jiang et al. (2017); Wiqvist et al. (2019) (Sec. 1.1.1.1). As the normalising constant of the exponential family is intractable, I resort to score matching (Hyvärinen, 2005) and its sliced version (Song et al., 2020), which allow fitting unnormalized models. As mentioned in Sec. 1.2.2.2, Score Matching corresponds to minimizing the empirical Hyvarinen score.

While the regression approach gives a single summary per dimension of θ , my framework allows a flexible number of summaries; further, thanks to the parameter transformation η , summaries learnt with my method are independent of the specific model parametrisation.

In ABC, I find the exponential family statistics to perform competitively with the ones learnt by regression. Additionally, the full exponential family approximation to the likelihood can be used to draw samples from an approximate posterior without generating additional simulations from the model. This is conceptually similar to Radev et al. (2020) and Hermans et al. (2020). As the normalising constant of the exponential family is intractable, I resort to MCMC for doubly intractable targets (Murray et al., 2012). This requires running a nested MCMC over the data space at each MCMC step over θ and is therefore inefficient for high-dimensional data space. A

¹⁰Provided that the domain of the probability distribution does not vary with the parameter.

recent work (Glaser et al., 2022) builds on my approach and develops a more efficient custom MCMC sampler. However, it is a promising approach for expensive simulator models with low-dimensional output.

1.3.2 Generalised Bayesian Likelihood-Free Inference Using Scoring Rules Estimators

In Chapter 3, I derive a novel posterior for the parameters θ of a simulator model $P(\cdot|\theta)$ from the loss-based generalised posterior in Eq. (1.2.1.2), by choosing the loss to be a Scoring Rule (Sec. 1.2.2). Notice how the standard Bayesian posterior can be written in this form by using the log-score (Sec. 1.2.2.2), which is however unavailable for simulator models as it requires the likelihood function. Therefore, in the generalised posterior, I employ Scoring Rules admitting empirical estimators based on samples from $P(\cdot|\theta)$. The proposed sampling strategy proceeds by running an MCMC on θ where, at each step, simulations from the model are used to estimate the target (it is an instance of the pseudomarginal MCMC of Andrieu et al., 2009).

My approach subsumes previous Bayesian Likelihood-Free Inference methods such as Bayesian synthetic likelihood (BSL; see Sec. 1.1.1.2). However, BSL and other popular methods (such as Approximate Bayesian Computation, see Sec. 1.1.1.1) are motivated as approximations to the true posterior, while my posterior is derived from the generalised formulation. I study theoretically and via computational experiments the features of this new posterior: with strictly proper S and a well-specified model, the posterior concentrates on the true parameter value as the number of observation increases; this highlights some issues with BSL, which relies on a non-strictly proper Scoring Rule. Further, importantly, with some of the Scoring Rules I employ, the resulting posterior is robust to outliers in the data, while the standard posterior and its approximations are not; my method is therefore well suited to parameter inference for simulator models using outlier-contaminated data.

1.3.3 Training generative networks via Scoring Rule minimization

As discussed in Section 1.1.2, generative networks are typically trained adversarially, which often leads to underestimating the variance of the data distribution. Although in many applications of generative networks (such as image generation) calibrated uncertainty estimation is not essential, this is required in key tasks in statistics. Therefore, in Chapters 4 and 5, I investigate training generative networks via Scoring Rule

Minimization. If the chosen Scoring Rule can be unbiasedly estimated using simulations, adversarial-free minimization is possible, leading to better calibrated generative distributions and easier training and hyperparameter tuning than the adversarial approach. Scoring Rule Minimization was used sparsely before (Bouchacourt et al., 2016; Gritsenko et al., 2020; Harakeh and Waslander, 2021), but a rigorous formalisation such as the one I give in Chapters 4 and 5 was not present in the literature.

In Chapter 4, I use Scoring Rule Minimization to train a conditional generative network to approximate the posterior distribution for a Bayesian Likelihood-Free Inference (LFI) task. The resulting approach belongs to the class of neural network-based methods to Bayesian LFI discussed in Section 1.1.1.3 and is closely related to Ramesh et al. (2022), which trained a generative network for the same task using an adversarial approach. In contrast to the methods based on normalising flows, employing a generative network more easily scales to high-dimensional and structured data. In simulation studies, Scoring Rule Minimization yields better results than the adversarial approach of Ramesh et al. (2022) with easier and cheaper training.

Finally, an original contribution of Chapter 5 is the application of Scoring Rule Minimization to probabilistic forecasting, namely the task of stating a probability distribution for the future step of a temporal process based on past information (Gneiting and Katzfuss, 2014). As in Chapter 4, I exploit conditional generative networks to this aim: however, here the conditioning variable is a window of past observations, and the output of the generative network is a (stochastic) prediction. In this setting, the data are *not* independent: the realisation at one time step enters the observation window at the next. Therefore, I extend the Scoring Rule Minimization formulation to a *prequential* (predictive-sequential, Dawid, 1984) training objective and prove a form of consistency of the minimizing distribution, provided that the temporal process has some stationarity and memoryless property. In contrast, adversarial training is formulated for independent data. In simulation studies, Scoring Rule Minimization achieves better performances (especially in terms of calibration) than adversarial methods.

Chapter 2

Score Matched Neural Exponential Families for Likelihood-Free Inference

Pacchiardi, Lorenzo, and Ritabrata Dutta. “Score Matched Neural Exponential Families for Likelihood-Free Inference.” *J. Mach. Learn. Res.* 23 (2022): 38-1.

Score Matched Neural Exponential Families for Likelihood-Free Inference

Lorenzo Pacchiardi
Department of Statistics
University of Oxford
Oxford, OX1 3LB
United Kingdom

LORENZO.PACCHIARDI@STATS.OX.AC.UK

Ritabrata Dutta
Department of Statistics
University of Warwick
Coventry, CV4 7AL
United Kingdom

RITABRATA.DUTTA@WARWICK.AC.UK

Abstract

Bayesian Likelihood-Free Inference (LFI) approaches allow to obtain posterior distributions for stochastic models with intractable likelihood, by relying on model simulations. In Approximate Bayesian Computation (ABC), a popular LFI method, summary statistics are used to reduce data dimensionality. ABC algorithms adaptively tailor simulations to the observation in order to sample from an approximate posterior, whose form depends on the chosen statistics. In this work, we introduce a new way to learn ABC statistics: we first generate parameter-simulation pairs from the model independently on the observation; then, we use Score Matching to train a neural conditional exponential family to approximate the likelihood. The exponential family is the largest class of distributions with fixed-size sufficient statistics; thus, we use them in ABC, which is intuitively appealing and has state-of-the-art performance. In parallel, we insert our likelihood approximation in an MCMC for doubly intractable distributions to draw posterior samples. We can repeat that for any number of observations with no additional model simulations, with performance comparable to related approaches. We validate our methods on toy models with known likelihood and a large-dimensional time-series model.

Code for reproducing the experiments is available at <https://github.com/LoryPack/SM-ExpFam-LFI>.

Keywords: Likelihood-Free Inference, Score Matching, Approximate Bayesian Computation, MCMC for doubly intractable distributions, Exponential Family.

1. Introduction

Stochastic simulator models are used to simulate realizations of physical phenomena; usually, a set of parameters θ govern the simulation output. As the model is stochastic, repeated simulations with fixed θ yield different outputs; their distribution is the model's *likelihood*. Simple models provide an analytic expression of the likelihood, which is however unavailable for more complex ones.

Upon observing a real-world realization of the phenomenon the model is describing, researchers may want to obtain a *posterior distribution* over parameters. If the likelihood is known, standard Bayesian inference tools (such as MCMC or variational inference) allow

to get posterior samples; if otherwise the likelihood is missing, Likelihood-Free Inference (LFI) techniques are the only viable solution. LFI has been applied in several domains, including genomics (Tavaré et al., 1997), biological science (Dutta et al., 2018), meteorology (Hakkarainen et al., 2012), geological science (Pacchiardi et al., 2020), genomics (Tavaré et al., 1997; Toni et al., 2009; Marttinen et al., 2015), and epidemiology (McKinley et al., 2018; Minter and Retkute, 2019; Dutta et al., 2021a).

In some LFI techniques (Wood, 2010; Thomas et al., 2020; Price et al., 2018), the missing likelihood is replaced with an explicit approximation built from model simulations at each parameter value. Alternatively, Approximate Bayesian Computation (ABC) (Marin et al., 2012; Lintusaari et al., 2017) circumvents the unavailability of the likelihood by comparing model simulations with the observation according to some notion of distance. To reduce data dimensionality, ABC usually relies on summary statistics, whose choice is not straightforward. Recently, the expressive capabilities of Neural Networks (NNs) have been leveraged to learn ABC statistics (Jiang et al., 2017; Wiqvist et al., 2019; Pacchiardi et al., 2020; Akesson et al., 2021). These techniques train NNs parametrizing the statistics by minimizing a suitable loss on a training set of parameter-simulation pairs generated from the model.

In the present work, we propose a new way to learn ABC statistics with NNs. Most previous works (Jiang et al., 2017; Wiqvist et al., 2019; Akesson et al., 2021) trained a single NN with the regression loss introduced in Fearnhead and Prangle (2012). Instead, we consider an exponential family with two NNs parametrizing respectively the sufficient statistics and natural parameters and fit this to the model. As the exponential family is the most general class of distributions with sufficient statistics of fixed size (Appendix A), it makes intuitive sense to use the learned sufficient statistics in ABC. Indeed, this approach empirically yields superior or equivalent performance with respect to state-of-the-art approaches.

As in previous approaches, we consider a training set of parameter-simulation pairs; however, we train the NNs with Score Matching (SM) or its Sliced approximation (SSM), which do not require evaluating the normalizing constant of the exponential family. We extend the SM and SSM objectives to the setting of conditional densities, thus fitting the likelihood approximation for all values of data and parameters.

In contrast to related approaches (Jiang et al., 2017; Wiqvist et al., 2019; Pacchiardi et al., 2020; Akesson et al., 2021), our method provides a full likelihood approximation. We test therefore direct sampling from the corresponding posterior (in place of ABC) with an MCMC algorithm for doubly intractable distributions. This approach achieves satisfactory performance with no additional model simulations and is therefore a valid alternative to standard LFI schemes for expensive simulator models. The computational gain is even larger when inference is performed for several observations, as the same likelihood approximation can be used.

The rest of our paper is organized as follows. In Section 2, we briefly review some LFI methods. Next, in Section 3 we introduce the neural conditional exponential family and show how to fit it with SM or SSM. In Section 4 we discuss how to exploit the exponential family to extract ABC statistics or for direct sampling. We extensively validate our proposed approaches in Section 5. Finally, we discuss related works in Section 6 and conclude in Section 7, where we also highlight directions for future research.

1.1 Notation

We set here notation for the rest of our manuscript. We will denote respectively by $\mathcal{X} \subseteq \mathbb{R}^d$ and $\Theta \subseteq \mathbb{R}^p$ the data and parameter space. Upper case letters will denote random variables while lower case ones will denote observed (fixed) values. Subscripts will denote vector components and superscript in brackets sample indices, while $\|\cdot\|$ will denote the ℓ_2 norm.

2. Likelihood-Free Inference

Let us consider a model which allows to generate a simulation $x \in \mathcal{X}$ at any parameter value $\theta \in \Theta$, but for which it is not possible to evaluate the likelihood $p_0(x|\theta)$. Given an observation x^0 and a prior on the parameters $\pi(\theta)$, Bayesian inference obtains the posterior distribution $\pi_0(\theta|x^0) = \frac{\pi(\theta)p_0(x^0|\theta)}{p_0(x^0)}$. However, obtaining that explicitly (or even sampling from it with Markov Chain Monte Carlo, MCMC) is impossible without having access to the likelihood, at least up to a normalizing constant that is independent on x .

Likelihood-Free Inference techniques yield approximations of the posterior distribution by replacing likelihood evaluations with model simulations. Broadly, they can be split into two kinds of approaches: Surrogate Likelihood, which explicitly builds a likelihood function, and Approximate Bayesian Computation (ABC), which instead uses discrepancy between simulated and observed data. The latter approach has already been described in detail in Sec. 1.1.1.1 of the introduction of the present thesis. The next subsection instead discusses two instances of Surrogate Likelihood.

2.1 Surrogate Likelihood

Surrogate likelihood approaches exploit simulations to build an explicit likelihood approximation, which is then inserted in standard likelihood-based sampling schemes (say, MCMC; Wood 2010; Thomas et al. 2020; Fasiolo et al. 2018; An et al. 2019, 2020; Price et al. 2018). Here, we discuss two methods that fall under this framework.

Synthetic Likelihood. Synthetic Likelihood (SL, Wood 2010) replaces the exact likelihood with a normal distribution for the summary statistics $s = s(x)$; specifically, it assumes $S|\theta \sim \mathcal{N}(\mu_\theta, \Sigma_\theta)$, where the mean vector μ_θ and covariance matrix Σ_θ depend on θ . For each θ , an estimate of μ_θ and Σ_θ is built with model simulations, and the likelihood of the observed statistics $s(x^0)$ is evaluated.

Using the statistics likelihood to define a posterior distribution yields a Bayesian SL (BSL). In Price et al. (2018), MCMC is used to sample from the BSL posterior; as in ABC, this approach requires generating simulations for each considered θ . However, one single simulation per parameter value is usually sufficient in ABC, while estimating $\mu_\theta, \Sigma_\theta$ in BSL requires multiple simulations. Nevertheless, a parametric likelihood approximation seemingly allows scaling to larger dimensional statistics space (Price et al., 2018).

Ratio Estimation. Thomas et al. (2020) tackle LFI by estimating the ratio between likelihood and data marginal: $r(x, \theta) = p(x|\theta)/p(x)$. As $r(x, \theta) \propto p(x|\theta)$ with respect to θ , this can be inserted in a likelihood-based sampling scheme in order to sample from the posterior, similarly to BSL. This method is termed Ratio Estimation (RE).

In practice, for each value of θ , an estimate $\hat{r}(x, \theta)$ is built by generating simulations from the model $p_0(x|\theta)$ and from the marginal¹ $p(x)$, and then fitting a logistic regression discriminating between the two sets of samples. In fact, logistic regression attempts to find a function $\hat{h}(x; \theta)$ for which $e^{\hat{h}(x; \theta)} \approx r(x; \theta)$.

In Thomas et al. (2020), \hat{h} is chosen in the class of functions $h_\beta(x; \theta) = \sum_{i=1}^k \beta_i(\theta) \psi_i(x) = \beta(\theta)^T \psi(x)$, where ψ is a vector of summary statistics and $\beta(\theta)$ are coefficients determined independently for each θ ; this therefore boils down to a linear logistic regression. Viewed differently, this approach approximates the likelihood with an exponential family, as in fact it assumes $p(x|\theta) \propto \hat{r}(x, \theta) = \exp(\hat{\beta}(\theta)^T \psi(x))$, for some coefficients $\hat{\beta}$ determined by data; it is thus a more general likelihood assumption than SL (as the normal distribution belongs to the exponential family).

Remark 1 (Reducing the number of simulations.) *Several approaches have attempted to reduce the number of simulations required in SL, RE and ABC. Specifically, Gaussian Processes can be exploited to replace evaluations of simulation-based quantities with emulated values, while at the same time providing an uncertainty quantification used to guide the next model simulations; that has been done for both ABC (Meeds and Welling, 2014; Wilkinson, 2014; Gutmann et al., 2016; Järvenpää et al., 2019; Jarvenpaa et al., 2020) and SL (Meeds and Welling, 2014; Wilkinson, 2014; Moores et al., 2015; Järvenpää et al., 2021). These approaches all consider a fixed observation x and emulate over parameter values.*

In Hermans et al. (2020), a classifier more powerful than logistic regression is used to learn the likelihood ratio estimate for all (x, θ) using parameter-simulation pairs (similarly to what we propose in Section. 3.3), amortizing therefore RE with respect to the observation.

3. Likelihood approximation with the exponential family

In this Section, we first introduce the parametric family which we use to approximate the likelihood. Then, we describe Score Matching (SM) and Sliced Score Matching (SSM), which allow us to fit distributions with unknown normalizing constants to data. Finally, we discuss how to extend SM and SSM to the setting of conditional densities, to obtain a likelihood approximation valid for all data and parameter values.

3.1 Conditional exponential family

A probability distribution belongs to the exponential family if it has a density of the following form:²

$$p(x|\theta) = \frac{e^{\eta(\theta)^T f(x)} h(x)}{Z(\theta)}, \tag{1}$$

where $x \in \mathcal{X}$, $\theta \in \Theta$. Here, $f : \mathcal{X} \rightarrow \mathbb{R}^{d_s}$ is a function of the data (sufficient statistics), $\eta : \Theta \rightarrow \mathbb{R}^{d_s}$ is a function of the parameters (natural parameters) and $h(x) : \mathcal{X} \rightarrow \mathbb{R}$ is a non-negative scalar function of data (base measure), whose support has non-zero measure.

1. Simulating from the marginal can be done by drawing $\theta^j \sim p(\theta)$, $x^j \sim p(x|\theta^j)$ and discarding θ^j .
 2. As we are concerned with continuous random variables, across the work we use the Lebesgue measure as a base measure, without explicitly referring to it.

The normalizing constant $Z(\theta) = \int_{\mathcal{X}} \exp\{\eta(\theta)^T f(x)\} h(x) dx$ is intractable and assumed to be finite $\forall \theta \in \Theta$; we will discuss later (Sec. 3.3) why this assumption is not an issue.

Across this work, we refer to Eq. (1) as *conditional* exponential family to stress that we learn an approximation valid for all x 's and θ 's by selecting functions f , η and h .

In the following, we rewrite Eq. (1) as: $p(x|\theta) = \exp(\bar{\eta}(\theta)^T \bar{f}(x))/Z(\theta)$, where $\bar{\eta}(\theta) = (\eta(\theta), 1) \in \mathbb{R}^{d_s+1}$ and $\bar{f}(x) = (f(x), \log h(x)) \in \mathbb{R}^{d_s+1}$. For simplicity, we will drop the bar notation, using as convention that $\eta(\theta)$ contains the natural parameters of the exponential family plus an additional constant term, and that the last component of $f(x)$ is $\log h(x)$.

Neural conditional exponential family. Let now f_w and η_w denote two Neural Networks (NNs) with collated weights w (in practice the two NNs do not share parameters); then, the neural conditional exponential family is defined as:

$$p_w(x|\theta) = \frac{e^{\eta_w(\theta)^T f_w(x)}}{Z_w(\theta)}. \quad (2)$$

Remark 2 *Ratio Estimation (Sec. 2.1) parametrizes the likelihood as an exponential family with user-specified statistics and natural parameters β independently learned for each $\theta \in \Theta$. In contrast, here we learn both the $f_w(x)$ and the $\eta_w(\theta)$ transformations over all \mathcal{X} and Θ at once by selecting the best w .*

Remark 3 (Identifiability) *For a family of distributions indexed by a parameter ϕ , identifiability means that $p_\phi(x|\theta) = p_{\phi'}(x|\theta) \forall x, \theta \implies \phi = \phi'$. The weights w in the neural conditional exponential family are not identifiable for two reasons: first, NNs have many intrinsic symmetries. Secondly, replacing f_w in Eq. (2) with $A \cdot f_w$ and η_w with $(A^T)^{-1} \cdot \eta_w$ does not change the probability density. In Khemakhem et al. (2020), two suitable concepts of function identifiability up to some linear transformations are defined. f_w and η_w in Eq. (2) are identifiable according to those definitions, under strict conditions on the architectures of the Neural Networks parametrizing them (Khemakhem et al., 2020). Moreover, they empirically verify that NNs not satisfying the above assumptions result in approximately identifiable f_w and η_w , according to their definition. More details are given in Appendix B.1.*

Remark 4 (Universal approximation) *Using larger d_s in Eq. (1) increases the expressive power of the approximating family. Khemakhem et al. (2020) proved that, by considering a freely varying d_s and generic f and η , the conditional exponential family has universal approximation capabilities for the set of conditional probability densities; we give more details in Appendix B.2. This result does not consider the practicality of fitting the approximating family to data, which arguably becomes harder when d_s increases.*

3.2 Score Matching

In order to fit a parametric density p_w to data, the standard approach is finding the Maximum Likelihood Estimator (MLE) for w ; in the limit of infinite data, that corresponds to minimizing the Kullback-Leibler divergence from the data distribution. MLE requires however the normalizing constant of p_w to be known, which is not the case for our approximating family (Eq. 2).

Score Matching (SM, Hyvärinen 2005) is a possible way to bypass the intractability of the normalizing constant. In this Subsection, we review SM for unconditional densities, discuss a faster version and provide an extension for distributions with bounded domain.

The original Score Matching (SM) Let us discard now the conditional dependency on θ and, following Hyvärinen (2005), consider a random variable X distributed according to $p_0(x)$. We want to use samples from p_0 to fit a generic model $p_w(x) = \tilde{p}_w(x)/Z_w$, where \tilde{p}_w is unnormalized and Z_w is intractable.

Definition 5 *Score Matching (SM) corresponds to finding:*

$$\arg \min_w D_F(p_0 \| p_w),$$

where the Fisher Divergence D_F is defined as:

$$D_F(p_0 \| p_w) = \frac{1}{2} \int_{\mathcal{X}} p_0(x) \|\nabla_x \log p_0(x) - \nabla_x \log p_w(x)\|^2 dx. \quad (3)$$

The Fisher divergence depends only on the logarithmic derivatives $\nabla_x \log p_0(x)$ and $\nabla_x \log p_w(x)$, which are termed *scores*³; computing D_F does not require therefore knowing the normalizing constant, as in fact:

$$\nabla_x \log p_w(x) = \nabla_x (\log \tilde{p}_w(x) - \log Z_w) = \nabla_x \log \tilde{p}_w(x).$$

Nevertheless, a Monte Carlo estimate of D_F in Eq. (3) using samples from p_0 would require knowing the logarithmic gradient of p_0 ; for this reason, Eq. (3) is usually termed *implicit* Fisher divergence. If $\mathcal{X} = \mathbb{R}^d$ and under mild conditions, Theorem 1 in Hyvärinen (2005) obtains an equivalent form for D_F in which p_0 only appears as the distribution over which expectation needs to be computed. We give here a similar result holding for more general $\mathcal{X} = \bigotimes_{i=1}^d (a_i, b_i)$, where $a_i, b_i \in \mathbb{R} \cup \{\pm\infty\}$; specifically, we consider the following assumptions⁴:

- A1** $p_0(x) \frac{\partial}{\partial x_i} \log p_w(x) \rightarrow 0$ when $x_i \searrow a_i$ and $x_i \nearrow b_i, \forall w, i$,
- A2** $\mathbb{E}_{p_0} [\|\nabla_x \log p_0(X)\|^2] < \infty, \mathbb{E}_{p_0} [\|\nabla_x \log p_w(X)\|^2] < \infty, \forall w$,
- A3** $\mathbb{E}_{p_0} \left| \frac{\partial^2}{\partial x_i \partial x_j} \log p_w(X) \right| < \infty, \forall w, \forall i, j = 1, \dots, d$.⁵

With the above, we can state the following:

-
3. In most of the statistics literature, *score* usually refer to the derivative of the log-likelihood with respect to the parameter; here, the nomenclature is slightly different.
 4. We stress how we do not explicitly enforce these assumptions for the neural exponential family, as they require controlling its behavior for diverging x . However, we argue below (end of Sec. 3.3) that this does not matter in practice when performing score matching on a finite set of observations $\{x_j^{(j)}\}$, as we can formally replace the neural exponential family with a distribution proportional to it for $x : \|x\| < \min_j \|x^{(j)}\|$ and converging to 0 for diverging x .
 5. We remark that this assumption was not present in Hyvärinen (2005), but it is necessary to apply Fubini-Tonelli theorem in the proof (see Appendix C.1), as already discussed in Yu et al. (2019).

Theorem 6 Let $\mathcal{X} = \bigotimes_{i=1}^d (a_i, b_i)$, where $a_i, b_i \in \mathbb{R} \cup \{\pm\infty\}$. If $p_0(x)$ is differentiable and $p_w(x)$ doubly differentiable over \mathcal{X} , then, under assumptions **A1**, **A2**, **A3**, the objective in Eq. (3) can be rewritten as:

$$D_F(p_0\|p_w) = \int_{\mathcal{X}} p_0(x) \sum_{i=1}^d \left[\frac{1}{2} \left(\frac{\partial \log p_w(x)}{\partial x_i} \right)^2 + \left(\frac{\partial^2 \log p_w(x)}{\partial x_i^2} \right) \right] dx + C. \quad (4)$$

Here, C is a constant w.r.t. p_w and x_i is the i -th coordinate of x .

A proof is given in Appendix C.1. We refer to Eq. (4) as the *explicit* Fisher Divergence, as it is now immediate to build an unbiased Monte Carlo estimate using samples from p_0 .

From Eq. (3), it is clear that $D_F(p_0\|p_w) \geq 0 \forall w$, and that choosing $p_w = p_0$ implies $D_F(p_0\|p_w) = 0$; however, the converse can not be said in general unless p_0 is supported over all \mathcal{X} , as stated in the following theorem:

Theorem 7 (Theorem 2 in Hyvärinen, 2005) Assume $p_0(x) > 0 \forall x \in \mathcal{X}$. Then:

$$D_F(p_0\|p_w) = 0 \iff p_0(x) = p_w(x) \forall x \in \mathcal{X}.$$

We prove Theorem 7 in Appendix C.2. If $p_0(x)$ is zero for some $x \in \mathcal{X}$, there could be a distribution p_w such that $D_F(p_0\|p_w) = 0$ even if $p_w \neq p_0$ (as discussed in Appendix C.6).

Sliced Score Matching (SSM) In the explicit Fisher Divergence (Eq. 4), the second derivatives of the log-density with respect to all components of x are required. When using the neural conditional exponential family (Eq. 2), this amounts to evaluating the second derivatives of f_w with respect to its input (see Appendix C.10). Practically, automatic differentiation libraries to obtain the derivatives effortlessly; however, the computational cost of the second derivatives is substantial, as it requires a number of forward and backward passes proportional to the dimension of the data x (see Appendix D.1)⁶.

Some approaches to reducing the computational burden have been proposed; we review some in Section 6. Here, we consider Sliced Score Matching (SSM, Song et al. 2020), which considers projections of the scores on random directions; matching the projections on *all* random directions ensures the two distributions are identical (under the same conditions as the original SM). More precisely, let $v \in \mathcal{V} \subseteq \mathbb{R}^d$ be a noise vector with a distribution q ; SSM is defined as follows:

Definition 8 *Sliced Score Matching (SSM) corresponds to finding:*

$$\arg \min_w D_{FS}(p_0\|p_w),$$

where the *Sliced Fisher Divergence* is defined as:

$$D_{FS}(p_0\|p_w) = \frac{1}{2} \int_{\mathcal{V}} q(v) \int_{\mathcal{X}} p_0(x) (v^T \nabla_x \log p_0(x) - v^T \nabla_x \log p_w(x))^2 dx dv. \quad (5)$$

6. Computing the derivatives during the forward pass offers some speedup, as automatic differentiation repeats some computations several times. However, this approach requires custom NN implementation (see Appendix D.1.1) and is thus not scalable to complex architectures. More importantly, the computational gain is limited with respect to what is achieved by the Sliced SM version introduced next.

We will require the noise distribution $q(v)$ to satisfy the following Assumption:

A4 For the random vector $V \sim q$, the covariance matrix $\mathbb{E}[VV^T] \succ 0$ is positive definite and $\mathbb{E}\|V\|_2^2 < \infty$.

As for SM, we can obtain an *explicit* formulation from the *implicit* one in Eq. (5). This is done in the following Theorem, which we prove for convenience in Appendix C.3:

Theorem 9 (Theorem 1 in Song et al. (2020)) *Let $\mathcal{X} = \bigotimes_{i=1}^d (a_i, b_i)$, where $a_i, b_i \in \mathbb{R} \cup \{\pm\infty\}$. If $p_0(x)$ is differentiable and $p_w(x)$ doubly differentiable over \mathcal{X} , then, under assumptions **A1**, **A2**, **A3**, **A4**, the objective in Eq. (5) can be rewritten as:*

$$D_{FS}(p_0\|p_w) = \int_{\mathcal{V}} q(v) \int_{\mathcal{X}} p_0(x) \left[v^T (\mathbb{H}_x \log p_w(x)) v + \frac{1}{2} (v^T \nabla_x \log p_w(x))^2 \right] dx dv + C, \quad (6)$$

where $\mathbb{H}_x \log p_w(x)$ denotes the Hessian matrix of $\log p_w(x)$ with respect to components of x and C is a constant w.r.t. p_w .

Assumption **A4** is satisfied by, among others, multivariate standard Gaussian and multivariate Rademacher random variables (Song et al., 2020). Additionally, these two distributions allow to compute explicitly the expectation with respect to v of $(v^T \nabla_x \log p_0(x))^2$ in Eq. (6), which leads to:

$$D_{FS}(p_0\|p_w) = \int_{\mathcal{X}} p_0(x) \left\{ \int_{\mathcal{V}} q(v) [v^T (\mathbb{H}_x \log p_w(x)) v] dv + \frac{1}{2} \|\nabla_x \log p_w(x)\|_2^2 \right\} dx + C; \quad (7)$$

in Song et al. (2020), the Monte Carlo estimate of the latter expression is found to perform better than the one for Eq. (6); across this work, we will therefore consider Eq. (7) with the Rademacher noise when using SSM.

Analogously to SM, a non-negative p_0 ensures that the sliced Fisher divergence is zero if and only if $p_w = p_0$:

Theorem 10 (Lemma 1 in Song et al. (2020)) *Assume Assumption **A4** holds and that $p_0(x) > 0 \forall x \in \mathcal{X}$. Then:*

$$D_{FS}(p_0\|p_w) = 0 \iff p_0(x) = p_w(x) \forall x \in \mathcal{X}.$$

The above result is proven in Appendix C.4.

SSM has a reduced computational cost with respect to SM when automatic differentiation is used. In fact, computing the second derivatives in Eq. (4) requires d backward propagations, while the quadratic form involving the Hessian in Eq. (7) only requires two backward propagations independently on the dimension of x (see Appendix D.1).

SM and SSM over transformed domain. If the domain \mathcal{X} is unbounded ($a_i = -\infty$, $b_i = +\infty \forall i$), **A1** is usually satisfied. Instead, it is easy for that assumption to be violated if \mathcal{X} is bounded: for instance, if $p_0(x)$ converges to a constant at the boundary of \mathcal{X} , **A1** requires $\nabla_x \log p_w(x)$ to go to 0. Further, if $p_0(x)$ diverges, $\nabla_x \log p_w(x)$ has to converge to 0 faster than some rate, which is in general a strong requirement.

To apply SM to distributions with bounded domain \mathcal{X} under looser conditions, we then transform \mathcal{X} to $\mathcal{Y} = \mathbb{R}^d$ with a suitable bijection mapping t ; this creates a new random variable $Y = t(X)$, with distributions $p_0^Y(y)$ and $p_w^Y(y)$ induced by p_0 and p_w on \mathcal{X} .

Definition 11 We define respectively as *Transformed Score Matching (TranSM)* and *Transformed Sliced Score Matching (TranSSM)* the procedures:

$$\arg \min_w D_F(p_0^Y \| p_w^Y) \quad \text{and} \quad \arg \min_w D_{FS}(p_0^Y \| p_w^Y).$$

TranSM and TranSSM enjoy similar properties as SM and SSM, as stated in the following Theorem, which mirrors Theorem 7 and Theorem 10:

Theorem 12 Let $Y = t(X) \in \mathcal{Y}$ for a bijection t . Assume Assumption **A4** is satisfied, $p_0(x) > 0 \forall x \in \mathcal{X}$, and let D denote either D_F or D_{FS} ; then:

$$D(p_0^Y \| p_w^Y) = 0 \iff p_w(x) = p_0(x) \forall x \in \mathcal{X}.$$

We prove this Theorem (in Appendix C.5) by relying on the equivalence between distributions for the random variables Y and X .

Motivated by Theorem 12, across this work we will apply TranSM and TranSSM whenever \mathcal{X} is bounded; precisely, we adopt the same bijections as in the `Stan` package for statistical modeling (Appendix C.8, Carpenter et al. 2017).

Remark 13 Another way to extend SM to distributions with bounded support involves multiplying the scores in the implicit Fisher divergence by a correction factor that allows to obtain an explicit form under looser assumptions. This *Corrected SM (CorrSM)* approach was first proposed for non-negative random variables in Hyvärinen (2007), which also remarked how CorrSM and TranSM are equivalent (Appendix C.7). TranSM is however more practically viable, as a single SM implementation is needed, while CorrSM requires separate implementations for different kinds of domains. Additionally, the transformations can be straightforwardly applied to SSM, while (to the best of our knowledge) a correction approach for SSM has not yet been proposed.

Remark 14 Across this work we restrict to domains \mathcal{X} defined by independent constraints across the coordinates, i.e. $\mathcal{X} = \bigotimes_{i=1}^d (a_i, b_i)$. However, TranSM, TranSSM and CorrSM can be potentially applied to distributions with more general supports. We briefly review this and other extensions in Appendix C.9.

3.3 Score Matching for conditional densities

We now go back to conditional densities $p_0(\cdot|\theta)$ and define an expected Fisher divergence (Arbel and Gretton, 2018) by considering $\theta \sim \pi(\theta)$:

$$\begin{aligned} D_F^E(p_0 \| p_w) &= \int_{\Theta} \pi(\theta) D_F(p_0(\cdot|\theta) \| p_w(\cdot|\theta)) d\theta \\ &= \frac{1}{2} \int_{\Theta} \int_{\mathcal{X}} p_0(x|\theta) \pi(\theta) \|\nabla_x \log p_0(x|\theta) - \nabla_x \log p_w(x|\theta)\|^2 dx d\theta. \end{aligned} \tag{8}$$

Analogously, we define an expected sliced Fisher divergence $D_{FS}^E(p_0 \| p_w)$ from D_{FS} .

Note that $D_F^E(p_0 \| p_w) \geq 0$; moreover, under the assumption that $p_0(x|\theta)$ is supported on the whole domain $\mathcal{X} \forall \theta$, the above objective is equal to 0 if and only if $p_0(\cdot|\theta) = p_w(\cdot|\theta)$

$\pi(\theta)$ -almost everywhere (Arbel and Gretton, 2018). In fact, $D_F(p_0(\cdot|\theta)||p_w(\cdot|\theta)) \geq 0 \forall \theta$, with equality holding if and only if the two conditional distributions are the same, as long as they are both supported on the whole \mathcal{X} (by Theorem 7). Exploiting Theorem 10, a similar result can be obtained for D_{FS}^E .

Requiring $p_0(x|\theta) > 0 \forall x \in \mathcal{X}, \forall \theta \in \Theta$ means that the model is capable of generating all possible values of $x \in \mathcal{X}$ with non-zero probability for all $\theta \in \Theta$; otherwise, there may be distributions $p_w \neq p_0$ minimizing the objective (see Appendix C.6).

Analogously to Eqs. (4) and (6), we can obtain explicit formulations⁷ of D_F^E and D_{FS}^E :

$$D_F^E(p_0||p_w) = \underbrace{\int_{\Theta} \int_{\mathcal{X}} p_0(x|\theta) \pi(\theta) \sum_{i=1}^d \left[\frac{1}{2} \left(\frac{\partial \log p_w(x|\theta)}{\partial x_i} \right)^2 + \left(\frac{\partial^2 \log p_w(x|\theta)}{\partial x_i^2} \right) \right]}_{J(w)} dx d\theta + C, \quad (9)$$

$$D_{FS}^E(p_0||p_w) = \underbrace{\int_{\mathcal{V}} \int_{\Theta} \int_{\mathcal{X}} q(v) p_0(x|\theta) \pi(\theta) \left[v^T (\mathbf{H}_x \log p_w(x|\theta)) v + \frac{1}{2} \|\nabla_x \log p_w(x|\theta)\|_2^2 \right]}_{J_S(w)} dx d\theta dv + C,$$

where C denotes constants with respect to w . For these two expressions, it is immediate to obtain Monte Carlo estimates using samples $(\theta^{(j)}, x^j)_{j=1}^N, \theta^{(j)} \sim \pi$ and $x^{(j)} \sim p(\cdot|\theta^{(j)})$, and draws from the noise model $\{v^{(j,k)}\}_{1 \leq j \leq N, 1 \leq k \leq M}$:

$$\begin{aligned} \hat{J}(w) &= \frac{1}{N} \sum_{j=1}^N \left[\sum_{i=1}^d \left(\frac{1}{2} \left(\frac{\partial \log p_w(x^{(j)}|\theta^{(j)})}{\partial x_i} \right)^2 + \left(\frac{\partial^2 \log p_w(x^{(j)}|\theta^{(j)})}{\partial x_i^2} \right) \right) \right], \\ \hat{J}_S(w) &= \frac{1}{NM} \sum_{j=1}^N \sum_{k=1}^M \left[v^{(j,k),T} (\mathbf{H}_x \log p_w(x^{(j)}|\theta^{(j)})) v^{(j,k)} + \frac{1}{2} \|\nabla_x \log p_w(x^{(j)}|\theta^{(j)})\|_2^2 \right]. \end{aligned} \quad (10)$$

As we discussed before, computing the square bracket term in $\hat{J}_S(w)$ only requires two backward propagations, while the square bracket term in $\hat{J}(w)$ requires d . However, $\hat{J}_S(w)$ sums over $M \cdot N$ terms, while $\hat{J}(w)$ sums over N . Nevertheless, when using $\hat{J}_S(w)$ to train a neural network, good results can be obtained by using a single different noise realization for each training sample $(\theta^{(j)}, x^j)$ at each epoch, thus leading to smaller computational cost with respect to $\hat{J}(w)$ (Song et al., 2020).

7. Requiring $p_0(x|\theta)$ to be differentiable and $p_w(x|\theta)$ doubly differentiable over \mathcal{X} for all θ and:

- A1.** $p_0(x|\theta) \frac{\partial}{\partial x_i} \log p_w(x|\theta) \rightarrow 0$ for $x_i \searrow a_i$ and $x_i \nearrow b_i, \forall w, i, \theta$,
- A2.** $\mathbb{E}_{p_0} [\|\nabla_x \log p_0(X|\theta)\|^2] < \infty, \mathbb{E}_{p_0} [\|\nabla_x \log p_w(X|\theta)\|^2] < \infty, \forall w, \theta$,
- A3.** $\mathbb{E}_{p_0} \left| \frac{\partial^2}{\partial x_i \partial x_j} \log p_w(X|\theta) \right| < \infty, \forall w, \theta, \forall i, j = 1, \dots, d$.

Score Matching for conditional exponential family. Both the implicit and explicit versions $D_F^E(p_0||p_w)$ and $D_{FS}^E(p_0||p_w)$ are well defined only if p_w is a proper density, which requires $Z_w(\theta) < \infty$. In practice, we are interested in:

$$\hat{w} = \arg \min \hat{J}(w) \quad \text{or} \quad \hat{w} = \arg \min \hat{J}_S(w). \quad (11)$$

When we compute $\hat{J}(w)$ or $\hat{J}_S(w)$, we only need to evaluate f_w in a finite set of points $\{x^{(j)}\}_{j=1}^N$ which belong to a bounded subset of \mathcal{X} , say $A \subset \mathcal{X}$. We can thus redefine the approximating family as follows:

$$p'_w(x|\theta) = \frac{\tilde{p}'_w(x|\theta)}{Z'_w(\theta)}, \quad \tilde{p}'_w(x|\theta) = \begin{cases} \exp(\eta_w(\theta)^T f_w(x)) & \text{if } x \in A, \\ g_w(x, \theta) & \text{otherwise,} \end{cases}$$

where g_w is such that $Z'_w(\theta) < \infty$, and can always be chosen such that $\tilde{p}'_w(x|\theta)$ is a smooth and continuous function of x .

Replacing p_w in Eq. (10) with p'_w does not change the value of $\hat{J}(w)$ and $\hat{J}_S(w)$; however, as p'_w is normalized, Eqs. (8) and (9) are now well defined for all w . Additionally, we do not need to specify A or g_w explicitly as we never evaluate the normalizing constant $Z'_w(\theta)$ (which depends on them). In the following we will thus use interchangeably p_w and p'_w .

Remark 15 (Notation) *For notational simplicity, in the rest of the work we drop the hat in \hat{w} , denoting by p_w the approximation obtained by one of the strategies in Eq. (11), and by f_w and η_w the corresponding sufficient statistics and natural parameters networks.*

4. Inference using the likelihood approximation

By fitting the neural conditional exponential family (Eq. 2) with SM or SSM to parameter-simulation pairs generated from the model, we obtain an approximation of the likelihood up to the normalization constant $Z_w(\theta)$: for each fixed θ , the function $x \mapsto \exp(\eta_w(\theta)^T f_w(x))$ is approximately proportional to $p_0(x|\theta)$. We exploit p_w in two ways: first, by using f_w as summaries in ABC; secondly, using the full approximation to draw samples from the posterior with MCMC for doubly intractable distributions. Both approaches are illustrated in Figure 1 and discussed next.

4.1 ABC with neural conditional exponential family statistics

The exponential family is the most general set of distributions with sufficient statistics of a given size (see Appendix A). Using f_w as summaries in ABC is therefore intuitively appealing: f_w represents in fact the sufficient statistics of the best exponential family approximation to p_0 , according to the (sliced) Fisher divergence. If p_0 belongs to the exponential family, $D_F^E(p_0||p_w) = D_{FS}^E(p_0||p_w) = 0$ if f_w and η_w are sufficient statistics and natural parameters for p_0 .

To use f_w in ABC, some practicalities are needed: first, we discard the last component of f_w , which represents the base measure $h_w(x)$. Secondly, as discussed in Section 3.1, f_w is identifiable only up to a scale parameter, so that the magnitude of the different components of f_w can vary significantly. Before using ABC, then, we rescale the different components

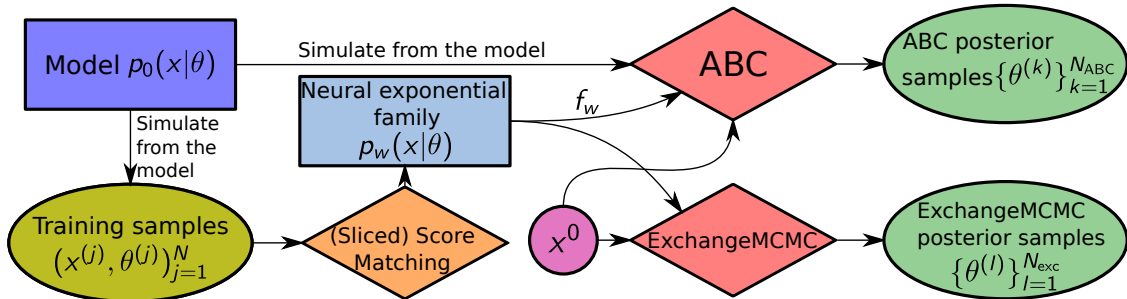


Figure 1: **Diagram showing the roles of the different components in the inference routines.** SSM or SM are used to train the neural exponential family on model simulations. Then, the sufficient statistics f_w are used in ABC or, alternatively, the neural exponential family is inserted in ExchangeMCMC. Notice how additional model simulation are needed for ABC but not for ExchangeMCMC.

by their standard deviation on new samples generated from the model, to prevent the ones with larger magnitude from dominating the ABC distance.

In the following, we will call the approach described here ABC with Score Matching statistics, for short ABC-SM, or ABC-SSM in the Sliced Score Matching case.

4.2 ExchangeMCMC with neural conditional exponential family

In contrast to other statistics learning methods (Jiang et al., 2017; Wiquist et al., 2019; Pacchiardi et al., 2020; Akesson et al., 2021), our technique provides a full likelihood approximation; it is therefore tempting to sample from the corresponding posterior directly, bypassing in this way ABC (and the additional model simulations required) altogether.

Unfortunately, p_w is known only up to the normalizing constant $Z_w(\theta)$; therefore, standard MCMC cannot be directly exploited, and methods for *doubly-intractable* distributions are required (see Park and Haran, 2018 for a review). Here, we use ExchangeMCMC (Murray et al. (2012), Algorithm 1 in Appendix D.4), an MCMC where, for each proposed θ' , a simulation from the distribution $p_w(x|\theta')$ is used within a Metropolis-Hastings rejection step.

In our case, we cannot generate samples from $p_w(\cdot|\theta')$ directly; to circumvent such issue, Murray et al. (2012) suggested to run an MCMC targeting $p_w(\cdot|\theta')$ for each ExchangeMCMC step; if the chain is long enough, the last step can be considered as a draw from p_w . Empirical results (Caimo and Friel, 2011; Alquier et al., 2016; Everitt, 2012; Liang, 2010) show that a relatively small number of inner MCMC steps are enough for good performance and that initializing the inner chain at the observation improves convergence; we employ therefore this strategy in our work.

Further, a variant of ExchangeMCMC employs Annealed Importance Sampling to improve the mixing of the chain (Murray et al., 2012). Specifically, a sequence of K auxiliary variables are drawn from Metropolis-Hastings kernels targeting some intermediate distributions, and the ExchangeMCMC acceptance rate is modified accordingly. This approach, termed *bridging*, decreases the number of rejections in ExchangeMCMC due to poor auxiliary variables. See Appendix D.4 for more details.

In the following, we will refer to using ExchangeMCMC with our likelihood approximation as Exc-SM or Exc-SSM, according to whether SM or SSM are used to obtain the likelihood approximation. Exc-SM and Exc-SSM avoid additional model simulations (beyond the ones required to train p_w) at the cost of running a nested MCMC. However, the number of steps in the inner MCMC required to obtain good performance increases with the dimension of \mathcal{X} . Exc-SM and Exc-SSM are therefore ideally applied to models with moderate dimension x (up to a few hundred), for which simulation is expensive. The same likelihood approximation can be used to perform inference on several observations, which makes the computational gain even greater in this case.

4.3 Tuning choices

Our proposals involve some tuning choices: to define the neural exponential family, the architecture of the two neural networks needs to be chosen. Typically, the parameters of a simulator model are low-dimensional, so a simple fully-connected network suffices. The choice of architecture for the statistics network is more delicate; however, modern architectures can be leveraged to design networks suitable for specific kinds of data (such as images or temporal data), which can lead powerful summaries. Additionally, this design choice also pertains to all statistics learning methods employing neural networks (such as the regression one in Jiang et al., 2017).

Next, the dimension of the sufficient statistics needs to be fixed. We recommend to initially set this to the number of components of θ ; if the performance of the learnt statistics is unsatisfactory, the number of statistics can be increased. We remark how other popular statistics learning methods (Jiang et al., 2017) do not allow this, which yields increased flexibility.

Finally, if ExchangeMCMC is used, it needs to be tuned as well. As this avoids however further model simulations, tuning ExchangeMCMC can be a modest price to pay in cases where the simulator model is expensive to run.

Remark 16 (Model misspecification) *The neural exponential family approximation is only valid close to where training data was distributed. Specifically, if x^0 is far from that region, $p_{\tilde{w}}(x^0|\theta)$ may be assigned a large value rather than a (correct) small one. This could happen when the model p_0 is unable to generate data resembling the observation for any parameter value, such that standard Bayesian inference in presence of the likelihood would also perform poorly. To get better inference in such scenarios, we could use the generalized posterior introduced in Matsubara et al. (2022), which is robust to outliers and suitable for doubly-intractable distributions (as the exponential family).*

In Exc-SM and Exc-SSM, we may wonder whether running MCMC over x targeting $p_w(x|\theta)$ for a fixed θ can fail due to what we discussed above. We believe this is not the case: if the MCMC is initialized close to training data and $p_w(\cdot|\theta)$ is a good representation of $p_0(\cdot|\theta)$ in that region, $p_w(\cdot|\theta)$ is small for values of x close to the boundary of the region where training data was distributed. Then, the MCMC is “trapped” inside that region and has no way of reaching regions of \mathcal{X} where p_w may behave irregularly.

5. Simulation Studies

We perform simulation studies with our proposed approaches (Exc-SM, Exc-SSM, ABC-SM and ABC-SSM) and we compare with:

- ABC with statistics learned with the rejection approach discussed in Sec. 1.1.1.1 of the introduction of the present thesis (Fearnhead and Prangle, 2012; Jiang et al., 2017), which we term ABC-FP.
- Population Monte Carlo (PMC, Cappé et al. 2004) with Ratio Estimation (PMC-RE).
- PMC with Synthetic Likelihood, using the robust covariance matrix estimator developed in Ledoit and Wolf (2004) to estimate Σ_θ . We will denote this as PMC-SL.

Specifically, we consider three exponential family models and two time-series models (AR(2) and MA(2)) for which the exact likelihood is available, as well as a large-dimensional model with unknown likelihood (the Lorenz96 model, Lorenz 1996; Wilks 2005).

Exc-SM and Exc-SSM do not require additional simulations and run an MCMC, in contrast to sequential algorithms for the other methods, which we run with parallel computing. Comparing the computational cost is therefore not easy; in Appendix F.3, we report the number of simulations needed by the different methods to reach the same performance achieved by Exc-SM for the models with known likelihood.

Choice of neural network architecture In our exponential family approximation, we fix d_s to the number of parameters in each model. We added a Batch Normalization layer (see Appendix D.2) on top of the neural network representing η_w to reduce the unidentifiability introduced by the dot product between $f_w(x)$ and $\eta_w(\theta)$ (as discussed in Appendix C.10). For all techniques, we use 10^4 training samples. All NNs use SoftPlus nonlinearity (NNs using the more common ReLU nonlinearity cannot be used with SM and SSM as they have null second derivative with respect to the input).

For all models, η_w is represented by fully connected NNs. For the exponential family models, fully connected NNs are also used for f_w and s_β . For the time series and Lorenz96 models, we parametrize f_w and s_β with Partially Exchangeable Networks (PENs, Wiquvist et al. 2019). The output of an r -PEN is invariant to input permutations which do not change the probability density of data distributed according to a Markovian model of order r (see Appendix D.3). As AR(2) is a 2-Markovian model, we use a 2-PEN; similarly, a 1-PEN is used for the Lorenz96 model, which is 1-Markovian. Finally, we use a 10-PEN for the MA(2) model; albeit not being Markovian, Wiquvist et al. (2019) argued that MA(2) can be considered as “almost” Markovian so that the loss of information in imposing a PEN structure of high enough order is negligible. Further details are given in Appendix E.

Choice of inferential parameters. For ABC, we employ Simulated annealing ABC (SABC, Albert et al. 2015), which considers a set of particles and updates their position in parameter space across several iterations. We use 100 iterations and 1000 particles (posterior samples), corresponding to 1000 model simulations per iteration.

In Exc-SM and Exc-SSM, we run Exchange MCMC for 20000 steps, of which the first 10000 are burned-in. During burn-in, at intervals of 100 outer steps, we tune the proposal sizes according to the acceptance rate in the previous 100 steps; this adaptation is not

<i>Model</i>	Beta		Gamma ⁸		Gaussian		AR(2)		MA(2)		Lorenz96			
<i>Parameter</i>	α	β	k	θ	μ	σ	θ_1	θ_2	θ_1	θ_2	b_0	b_1	σ_e	ϕ
<i>Lower bound</i>	1	1	1	1	-10	1	-1	-1	-1	0	1.4	0	1.5	0
<i>Upper bound</i>	3	3	3	3	10	10	1	0	1	1	2.2	1	2.5	1

Table 1: **Bounds of uniform priors for the considered models.**

done after burn-in, which makes the chain ergodic. Our implementation of the Exchange algorithm is detailed in Appendix D.4.1. For the exponential family and time-series models, we test different numbers of inner MCMC steps, and eventually use 30 for the former and 100 for the latter, above which there was no noticeable performance improvement (more details in Appendix F.1.3 and Appendix F.2).

In PMC-SL and PMC-RE, we run the PMC algorithm with 10 iterations, 1000 posterior samples and respectively 100 (with SL) and 1000 (with RE) simulations per parameter value in order to estimate the approximate likelihood; such a large number of simulations (respectively 10^5 and 10^6 for each iteration) is required for the likelihood estimate to be numerically stable. For the exponential family models, we use the true sufficient statistics; for AR(2) and MA(2), we instead use autocovariances with lag 1 and 2 (as for instance in Marin et al. 2012). For PMC-RE, the cross-product of the statistics is also added to the list of statistics, as PMC-SL implicitly uses it.

5.1 Exponential family models

First, we consider three models for which a sample is defined as a 10-dimensional independent and identical distributed (i.i.d.) draw from either a Gaussian, Gamma or Beta distribution (all belonging to the exponential family). We put uniform priors on the parameters, with bounds given in Table 1. These models have different data ranges: unbounded for Gaussian, and respectively lower bounded by 0 and bounded in $[0, 1]$ for Gamma and Beta. Therefore, we directly apply SM and SSM to the Gaussian model, while TranSM and TranSSM are applied to Gamma and Beta.

Inferred sufficient statistics and natural parametrization. With these models, our exponential family approximation is well specified. Thus, we compare the learned f_w and η_w with the exact sufficient statistics and natural parameters using the Mean Correlation Coefficient (MCC, Appendix B.1.1), which ranges in $[0, 1]$ and measures how well a multivariate function is recovered. We report results obtained with SM in Table 2; the values are close to 1, indicating that the sufficient statistics and natural parameters are recovered quite well by our method. In Appendix F.1, we report values corresponding to SSM (Table 9), as well as comparisons of exact and learned embeddings in Figures 9 and 10.

Inferred posterior distribution. Figure 2 shows posteriors obtained with the proposed methods, for a possible observation θ for each model; we see that all approximate posteriors

8. The scale parameter of the Gamma distribution is usually called θ ; in contrast, in all the rest of this work we use θ to denote all parameters. Please beware of the difference.

<i>Model</i>	<i>MCC weak in</i>	<i>MCC weak out</i>	<i>MCC strong in</i>	<i>MCC strong out</i>
Beta (statistics)	0.964	0.958	0.723	0.723
Beta (nat. par.)	0.990	0.991	0.807	0.812
Gamma (statistics)	0.911	0.924	0.894	0.883
Gamma (nat. par.)	0.967	0.967	0.872	0.873
Gaussian (statistics)	0.944	0.937	0.808	0.824
Gaussian (nat. par.)	0.974	0.974	0.970	0.972

Table 2: **MCC for exponential family models between exact embeddings and those learned with SM.** We show weak and strong MCC values; MCC is between 0 and 1 and measures how well an embedding is recovered up to permutation and rescaling of its components (strong) or linear transformation (weak); the larger, the better. “in” denotes MCC on training data used to find the best transformation, while “out” denote MCC on test data. We used 500 samples in both training and test data sets.

are remarkably close to the exact one; moreover, the results with SM and SSM are indistinguishable. For all methods, we also estimate the Wasserstein distance between true and approximate posterior and compute the Root Mean Squared Error (RMSE) between mean of the true and approximate posterior; this is repeated for 100 simulated observations, with results reported in Figure 3. ABC-FP is here the worst method; additionally, ABC-SM and ABC-SSM perform similarly to PMC-SL and PMC-RE. Finally, Exc-SM and Exc-SSM are marginally worse.

5.2 Time series models

The Moving Average model of order 2, or MA(2), and the AutoRegressive model of order 2, or AR(2), are special cases of the ARMA time-series model. The MA(2) model is defined by:

$$X_1 = \xi_1, \quad X_2 = \xi_2 + \theta_1 \xi_1, \quad X_j = \xi_j + \theta_1 \xi_{j-1} + \theta_2 \xi_{j-2}, \quad j = 3, \dots, t,$$

while the AR(2) is defined as:

$$X_1 = \xi_1, \quad X_2 = \xi_2 + \theta_1 X_1, \quad X_j = \xi_j + \theta_1 X_{j-1} + \theta_2 X_{j-2}, \quad j = 3, \dots, t;$$

in both, ξ_j 's are i.i.d. standard normal error terms (unobserved). We take here $t = 100$ and we put uniform priors on the parameters of the two models, with bounds given in Table 1. For these models, the true likelihood can be evaluated, but they do not belong to the exponential family⁹.

Inferred posterior distribution. Figure 4 shows the posterior obtained with our proposed methods, for a possible observation for each model; again, our approximations are close to the exact posterior, with Exc-SM and Exc-SSM leading to slightly broader posteriors. Again, we assess performance with the Wasserstein distance from the true posterior

9. More precisely, they cannot be written as an exponential family with embedding dimension fixed with data size; in fact, MA(2) can be written as a Gaussian distribution with $t \times t$ covariance matrix, which is an exponential family whose embedding dimension increases with data size.

SCORE MATCHED NEURAL EXPONENTIAL FAMILIES FOR LFI

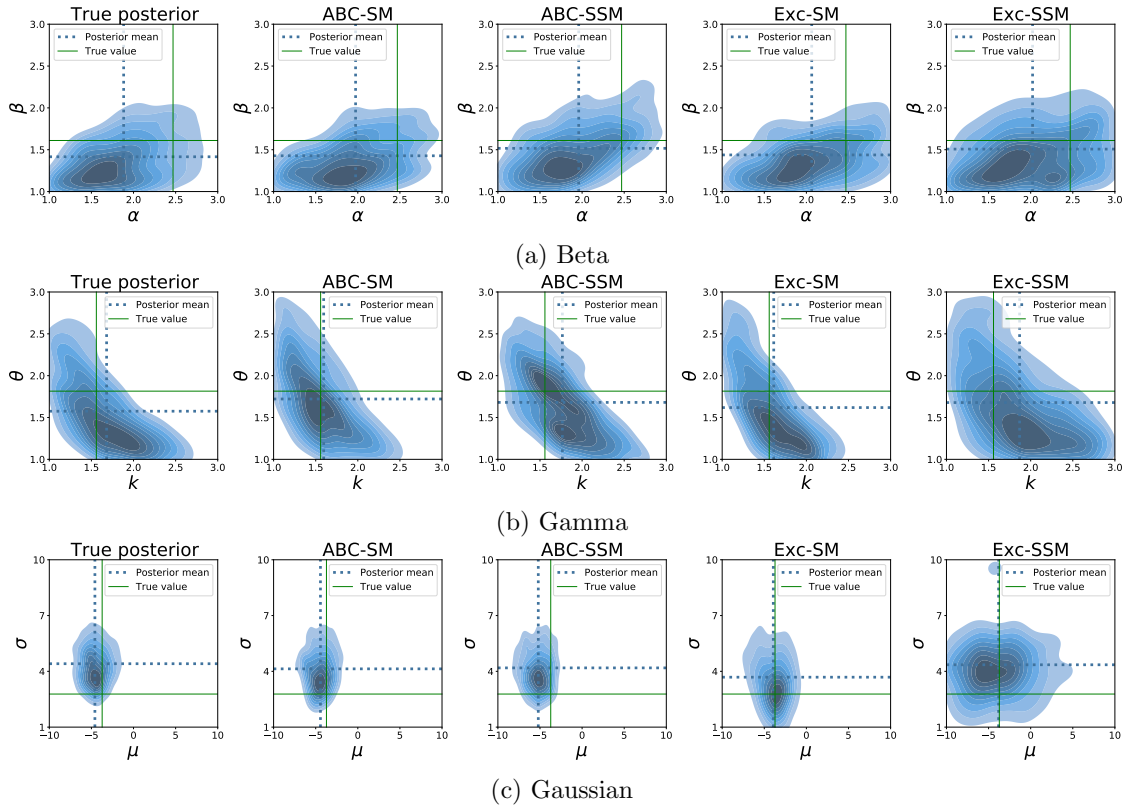


Figure 2: **True and approximate posteriors for exponential family models**, for a single observation per model. Dashed line represents posterior mean, while green solid line represents the exact parameter value.

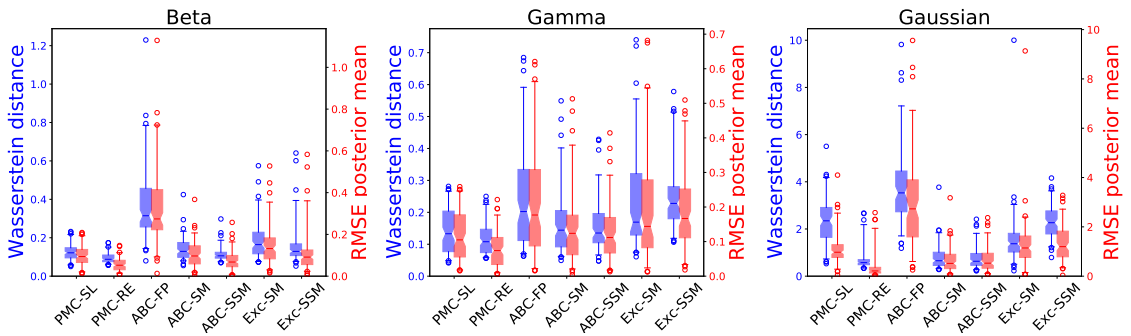


Figure 3: **Performance of the different techniques for exponential family models**. Wasserstein distance from the exact posterior and RMSE between exact and approximate posterior means are reported for 100 observations using boxplots. Boxes span from 1st to 3rd quartile, whiskers span 95% probability density region and horizontal line denotes median. The numerical values are not comparable across examples, as they depend on the range of parameters. Here, SL and RE used the true sufficient statistics.

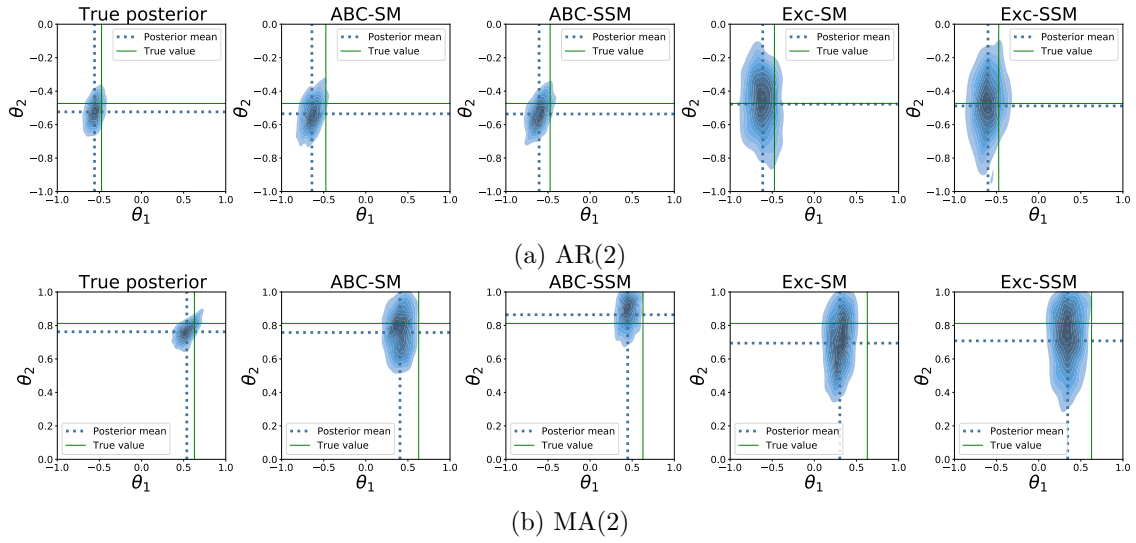


Figure 4: **True and approximate posteriors for AR(2) and MA(2) models**, for a single observation per model. Dashed line represents posterior mean, while green solid line represents the exact parameter value.

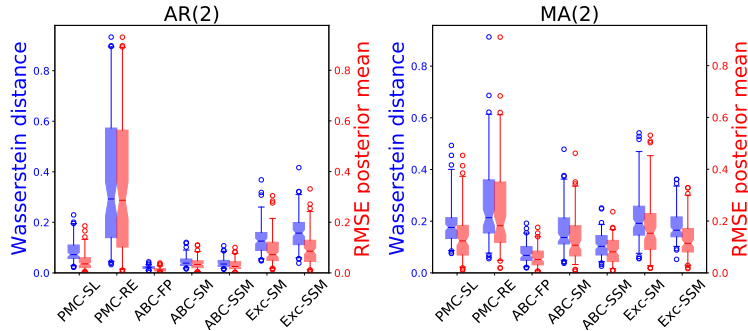


Figure 5: **Performance of the different techniques for AR(2) and MA(2) models**. Wasserstein distance from the exact posterior and RMSE between exact and approximate posterior means are reported for 100 observations using boxplots. Boxes span from 1st to 3rd quartile, whiskers span 95% probability density region and horizontal line denotes median.

and the RMSE between the means of true and approximate posterior for all methods, over 100 fresh observations. The results are reported in Figure 5; here, ABC-FP is the best method, with ABC-SM and ABC-SSM marginally worse. Exc-SM and Exc-SSM follow and perform better or comparably to PMC-RE and PMC-SL.

5.3 Lorenz96 meteorological model

The Lorenz96 model (Lorenz, 1996) is a toy model of chaotic atmospheric behaviour, including interacting slow and fast variables. We use here a modified version (Wilks, 2005)

where the effect of the fast variables on the slow ones is replaced by a stochastic effective term depending on a set of parameters. Specifically, this model is defined by the following coupled Differential Equations (DEs):

$$\frac{dy_k}{dt} = -y_{k-1}(y_{k-2} - y_{k+1}) - y_k + 10 - g(y_k, t; \theta); \quad k = 1, \dots, K,$$

where $y_k(t)$ is the value at time t of the k -th variable and indices are cyclic (index $K + 1$ corresponds to 1, and so on). The effective term g depends on $\theta = (b_0, b_1, \sigma_e, \phi)$, and is defined upon discretizing the DEs with a timestep Δt :

$$g(y, t; \theta) = \underbrace{b_0 + b_1 y}_{\text{linear deterministic term}} + \underbrace{\phi \cdot r(t - \Delta t) + \sigma_e(1 - \phi^2)^{1/2} \eta(t)}_{\text{zero mean AR(1)}},$$

where $\eta(t) \sim \mathcal{N}(0, 1)$. We numerically integrate the model using 4th order Runge-Kutta method with $\Delta t = 1/30$ in an interval $[0, T]$; further, we fix the initial condition to a value $y(0)$ which is independent on parameters. Here, the true likelihood is inaccessible, so that sampling from the exact posterior is impossible.

We consider the model in a small and large configuration: in the small one, we take $K = 8$ and $T = 1.5$, which lead to 45 timesteps and a data dimension of $d = 45 \cdot 8 = 360$. In the large one, we take instead $K = 40$ and $T = 4$, corresponding to 120 timesteps and a data dimension of $d = 120 \cdot 40 = 4800$.

Inferred posterior distribution. For both configurations, we perform inference with ABC-SSM and ABC-FP; additionally, we use Exc-SSM for the small configuration (as running the exchange algorithm in the large one is too costly). In ExchangeMCMC, we used 500 inner MCMC steps and 200 bridging steps for each outer step. Figure 6 reports posteriors for a single observation in both setups. The ABC-FP posterior is narrower than the one for ABC-SSM, but concentrated around similar parameter values. The posterior for Exc-SSM looks slightly different: it concentrates on similar parameter values as the other two for θ_1 and θ_2 , but is broader for σ_e and ϕ .

Posterior predictive validation. We assess the performance of the posterior predictive distribution $p(x|x^0) = \int p(x|\theta)\pi(\theta|x^0)d\theta$, in which $\pi(\theta|y)$ is the posterior obtained with one of the considered techniques. Specifically, we use the Kernel and Energy Scoring Rules (SRs) to evaluate how well the predictive $p(x|x^0)$ predicts the observation x^0 . A SR (Gneiting and Raftery, 2007) is a function of a distribution and an observation, and assesses the mismatch between them (the smaller, the better). More details on Scoring Rules are given in Sec. 1.2.2 in the introduction of the present thesis, with details on those used here.

As the Lorenz96 model returns a (multivariate) time-series, we compare predictive distribution and observation at each timestep independently. We repeat this validation with 100 different observations and corresponding posteriors, for all techniques and both model configurations. We obtain cumulative scores by summing the scores over timesteps; median and some quantile values over the 100 observations are shown in Fig. 7. In both model configurations, the posterior predictive generated by ABC-SSM and Exc-SSM marginally outperform ABC-FP, according to both SRs. Score values at each timestep, together with more details on the validation technique, are reported in Appendix F.4.

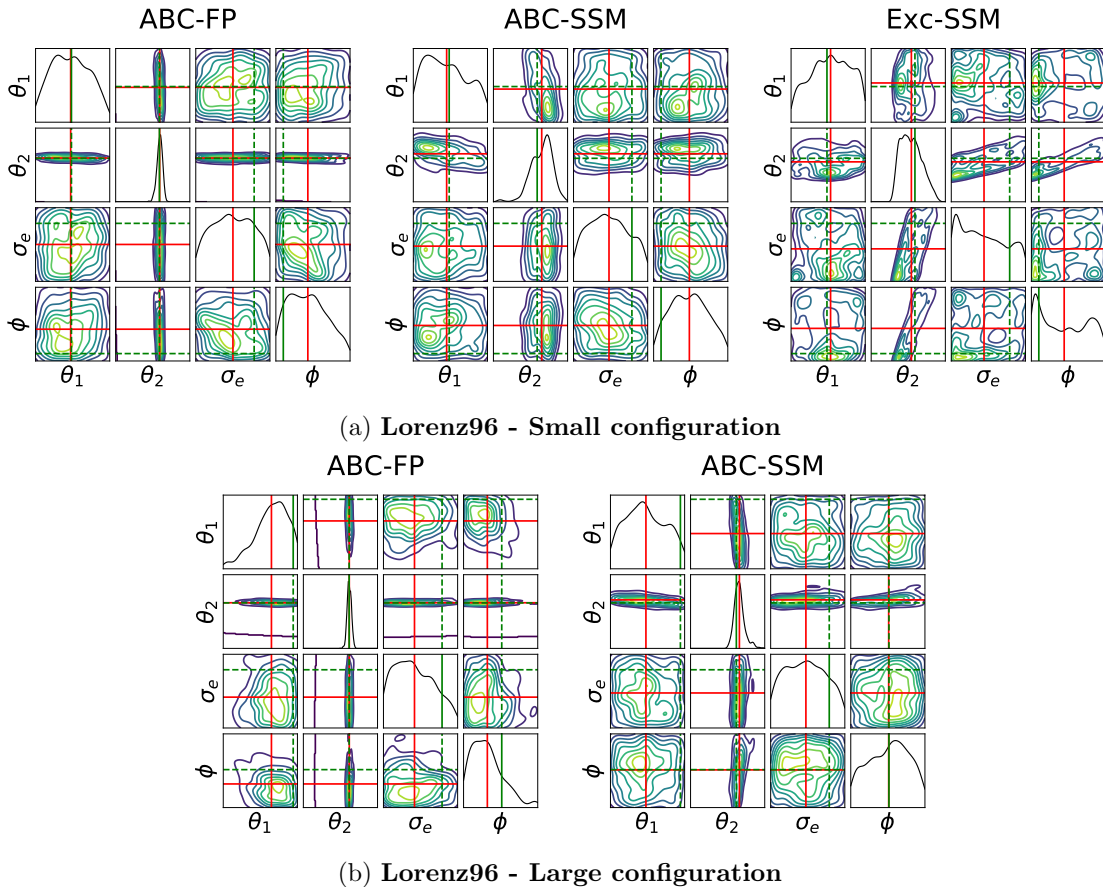


Figure 6: **Example of posterior inference for a single Lorenz96 observation with different approaches**, for both small (first row) and large (second row) configurations. In each panel, the diagonal plots represent the univariate marginal distributions, while the off-diagonal ones are bivariate density contour plots. Moreover, the green and red lines represent respectively exact parameter value and posterior mean. All axes span full prior range (see Table 1).

6. Related works

ABC statistics with NN. Using NNs for learning statistics for ABC has been previously suggested, first with the regression approach discussed in Sec. 1.1.1.1 in the introduction of the present thesis (used in Jiang et al. 2017; Wiqvist et al. 2019; Akesson et al. 2021). In Pacchiardi et al. (2020), a NN is trained so that the distance between pairs of simulated statistics best reproduces the distance between the corresponding parameter values. Chen et al. (2021) instead fit a NN by maximizing an estimate of the mutual information between statistics and parameters.

ABC statistics from auxiliary models. Some similarities with our work can be found in Gleim and Pigorsch (2013) and Ruli et al. (2016), which consider respectively an auxiliary

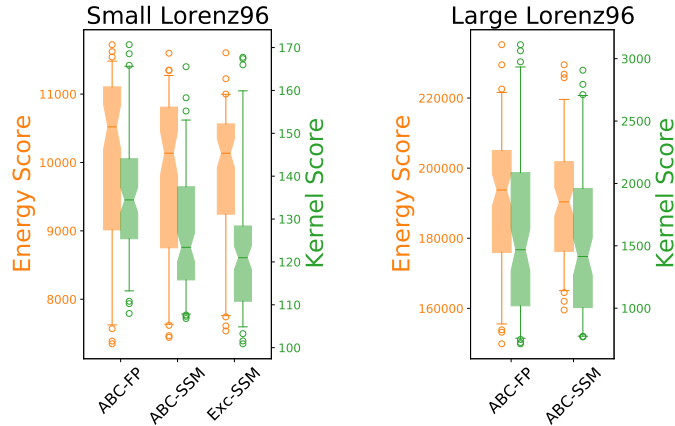


Figure 7: **Predictive performance of the different methods according to the Kernel and Energy Scoring Rules.** Each boxplot represents cumulative (i.e., summed over the time index) scoring rule value for a given method for the small (left) and large (right) Lorenz96 configuration. Boxes span from 1st to 3rd quartile, whiskers span 95% probability density region and horizontal line denotes median.

model and a composite likelihood alongside the simulator model and obtain summary statistics from them; similarly, our approach can be seen as building an auxiliary exponential family model with easily accessible summary statistics.

Statistics-free ABC. Recent ABC methods (Park et al., 2016; Bernton et al., 2019; Drovandi and Frazier, 2021; Wang et al., 2021) replace the summary statistics with the estimate of a statistical divergence. However, to estimate the latter, these methods require multiple observations and multiple simulations (some tens) from the model for each value of θ . This makes them inapplicable when a single observation is present and when the cost of running the simulator model is substantial.

LFI with NNs. The idea of sampling directly from the approximate posterior defined by p_w is related to a suite of LFI methods using NNs. A large part of these works use normalizing flows (NNs implementing invertible transformations, suitable for efficiently representing probability densities; see Papamakarios et al., 2021 for a review); specifically, Papamakarios et al. (2019); Lueckmann et al. (2019) use them to learn an approximation of the likelihood, while Papamakarios and Murray (2016); Lueckmann et al. (2017); Greenberg et al. (2019); Radev et al. (2020) learn the posterior. Most of the above approaches focus on inference for a single observation, tailoring the simulations to better approximate the posterior for the relevant parameter values at lower computational cost; instead, Radev et al. (2020) and Lueckmann et al. (2019) propose to amortize across observations, similarly to our approach. Besides normalizing flows, Klein et al. (2020) casts the LFI problem as a distributional regression one. Finally, Tabak et al. (2020) does not employ NNs but rather solves a Wasserstein barycenter problem to model conditional maps which allow sampling from the distribution of an observation conditional on some covariates.

Fitting unnormalized models. Several techniques besides Score Matching have been proposed for fitting unnormalized models: MCMC-MLE (Geyer, 1991) exploits MCMC to estimate the normalizing constant for different values of the parameter and uses that in MLE. Contrastive Divergence (CD, Hinton 2002) instead uses MCMC to obtain a stochastic approximation of the gradient of the log-likelihood; this requires a smaller number of MCMC steps with respect to MCMC-MLE, but the stochastic gradient estimate is biased. Minimum Probability Flow (MPF, Sohl-Dickstein et al. 2011) considers a dynamics from data to model distribution, and minimizes the Kullback Leibler divergence between the data distribution and the one obtained by running the dynamics for a short time; however, the efficacy of MPF depends significantly on the considered dynamics. Noise Contrastive Estimation (NCE, Gutmann and Hyvärinen 2012) converts the parameter estimation problem to ratio estimation between data distribution and a suitable noise distribution; in practice, NCE uses logistic regression to discriminate the observed data and data generated from the noise; in the loss, the normalizing constant appears explicitly and can be estimated independently. To be effective, NCE requires the noise distribution to overlap well with the data density while being easy to sample from and to evaluate, which is not easy to get. Finally, some works (Dai et al., 2019a,b) use the dual formulation of MLE to avoid estimating the normalizing constant at the price of introducing dual variables to be jointly estimated.

Fitting unnormalized conditional models. The approaches above running dynamics to estimate quantities (CD, MCMC-MLE, MPF) cannot easily be applied to the conditional setting. NCE has been instead used in Ton et al. (2021) with a single noise distribution for all values of the conditioning variable θ ; it however requires an independent NN to parametrize the normalizing constant $Z(\theta)$. SM can be instead easily applied to the conditional setting, as previously done in Arbel and Gretton (2018) and further demonstrated in our work. To the best of our knowledge, SSM was not applied to a conditional setting before, although the extension straightforwardly follows what is done for SM.

Fast approximations of SM. Besides SSM, some approximations to SM only requiring first derivatives have been investigated. For instance, Denoising Score Matching (Vincent, 2011) computes the Fisher divergence between the model and a kernel density estimate of the data distribution, which is equal to a quantity independent on the second derivative. Similarly, Kernel Stein Discrepancy (Liu and Wang, 2016) intrinsically depends on first derivatives only. For both techniques, however, several samples for each θ are required in the conditional setting; generating them would increase the number of simulations required to fit the neural exponential family. In contrast, our chosen SSM approximation only relies on first derivatives and does not need multiple simulations for each θ . Furthermore, SSM requires less tuning choices than both KSD and Denoising Score Matching.

Finally, Wang et al. (2020) exploits a connection between the Fisher divergence and gradient flows in the 2-Wasserstein space to develop an approximation that only relies on first-order derivatives.

Kernel Conditional Exponential Families (KCEFs). In KCEFs (Arbel and Gretton, 2018), the summary statistics and natural parameters are functions in a Reproducing Kernel Hilbert Space, whose properties allow to evaluate the density although an infinite-dimensional embedding space is used (using the *kernel trick*). In Arbel and Gretton (2018), SM was used to fit KCEFs instead of our neural conditional exponential families. They

build on Sriperumbudur et al. (2017), which first used SM to perform density estimation with (non-conditional) Kernel Exponential Families (KEFs). In Wenliang et al. (2019) NNs are used to parametrize the kernel in a KEF and trained with the SM loss. However, as KCEF’s do not have finite-dimensional sufficient statistics, they are unsuitable for learning ABC statistics. Additionally, KCEF’s have a worse complexity in terms of training dataset size with respect to NN-based methods. KCEF’s have also been used with a dual MLE objective in Dai et al. (2019a).

7. Conclusions and extensions

We proposed a technique to approximate the likelihood using a neural conditional exponential family, trained via (Sliced) Score Matching to handle the intractable normalizing constant.

We tested this approximation in two setups: first, by using the exponential family sufficient statistics as ABC statistics, which is intuitively appealing as the exponential family is the largest class of distributions with fixed-size sufficient statistics. We empirically showed this to be comparable or outperform ABC with summaries built via the state-of-the-art regression approach (Fearhead and Prangle, 2012; Jiang et al., 2017).

Secondly, we used MCMC for doubly-intractable distributions to sample from the posterior corresponding to the likelihood approximation, which we found to have performance comparable to the other approaches. This can be repeated for any new observation without additional model simulations, making it advantageous for expensive simulator models.

Our proposed direct sampling approach based on exponential family likelihood approximation and ExchangeMCMC could be improved as follows:

- we used ExchangeMCMC (Murray et al., 2012) to handle double intractability, but other algorithms could be more efficient, (for instance the one in Liang et al. (2016), which makes use of parallel computing). Alternatively, we could exploit the generalized posterior introduced in Matsubara et al. (2022), which allows standard MCMC to be used for double intractable distributions and is robust to outliers.
- To infer the posterior for a single observation, approximating the likelihood for all x ’s and θ ’s as we do now is suboptimal. Similar performance may in fact be obtained with fewer simulations tailored to the observation. Sequential schemes implementing such ideas have been introduced for LFI using normalizing flows (see for instance Papamakarios et al. 2019) and could be extended to our setup.
- The motivation for the current work was learning ABC statistics, hence the exponential family formulation. However, the dot-product structure between f_w and η_w is not beneficial for the direct sampling approach. An *energy-based model*, which employs a single NN with input (x, θ) in the exponent, may be more expressive and easier to train.
- An energy-based posterior approximation $\pi_w(\theta|x)$ could be trained by minimizing the expectation over the data marginal $p(x)$ of the (sliced) Fisher divergence with respect to the true posterior. This is complementary to the strategy employed in this work

to fit $p_w(x|\theta)$. Interestingly, $\pi_w(\theta|x)$ would be known up to a normalizing constant depending on x only, making use of standard MCMC possible.

Acknowledgments

We acknowledge fruitful discussion with prof. Christian Robert, prof. Antonietta Mira and prof. Geoff Nicholls. LP is supported by the EPSRC and MRC through the OxWaSP CDT programme (EP/L016710/1), which also funds the computational resources used to perform this work. RD is funded by EPSRC (grant nos. EP/V025899/1, EP/T017112/1) and NERC (grant no. NE/T00973X/1).

All summary statistics learning strategies used in this manuscript (the regression approach and the exponential family with SM and SSM) have been implemented in the Python library `ABCpy` (Dutta et al., 2021b), which exploits `Pytorch` (Paszke et al., 2019) to train the NNs. `ABCpy` was also used to run PMC-SL, PMC-RE and the ABC experiments.

A. Sufficient statistics

Consider a conditional probabilistic model $p(x|\theta)$; moreover, abusing notation, we will also denote as $p(x|t)$ the density $p_{X|T}(X = x|T = t)$, as well as $p(x|t; \theta)$ the density $p_{X|T, \Theta}(X = x|T = t; \Theta = \theta)$, where here Θ denotes the random variable which takes values θ . Finally, we use π to denote distributions over the parameter values θ ; specifically, $\pi(\theta|x)$ denotes the standard posterior, and $\pi(\theta|t)$ is an abuse of notation for the density $\pi_{\Theta|T}(\Theta = \theta|T = t)$.

Definition 17 *A statistic $t = t(x)$ is sufficient if $p(x|t; \theta) = p(x|t)$, where θ is a parameter of the distribution. Alternatively, we have, in the Bayesian setting:*

$$\pi(\theta|x) = \pi(\theta|t(x)),$$

for any (non-degenerate) choice of prior distribution $\pi(\theta)$.

The existence of sufficient statistics implies a precise form of the distribution:

Theorem 18 (Fisher–Neyman factorization theorem): *A statistic is sufficient $\iff p(x|\theta) = h(x)g(t(x)|\theta)$, where h and g are non-negative functions.*

It is clear from the above theorem that $f(x)$ in the exponential family is sufficient.

A stronger result regarding exponential family also exists, which goes under the name of Pitman–Koopman–Darmois theorem. This theorem says that the exponential family is the most general family of distributions for which there is a sufficient statistics whose size is fixed with the number of samples, provided that the domain of the probability distribution does not vary with the parameter θ (Koopman, 1936).

B. Properties of the conditional exponential family

B.1 Linear identifiability

Let us consider the exponential family model $p_w(x|\theta) = \exp(\eta_w(\theta)^T f_w(x))/Z_w(\theta)$, where here η_w and f_w are not restricted to be Neural Networks. Khemakhem et al. (2020) studies the identifiability properties of the above family; specifically, they consider identifiability properties of the feature extractors η_w and f_w . Identifiability of representations is useful as it means that two different models from the above family learn similar representations when trained with different initialization on the same data set. In the framework of our likelihood-free inference task, it also means that if the true model belongs to the family we are considering, it is theoretically possible to recover the natural parameters and the true sufficient statistics.

If we consider now η_w and f_w to be Neural Networks, this causes problems as they are not identifiable in the standard sense. In fact, many different parameter configurations lead to the same function (as there are many symmetries in how the transformations in a Neural Network layer are defined). Therefore, Khemakhem et al. (2020) introduces, following previous works, two more general notions of identifiable representations. Subsequently, let \mathcal{W} denote the space of the possible Neural Network weights w .

Definition 19 Weak identifiability (Section 2.2 in Khemakhem et al., 2020). Let \sim_w^f and \sim_w^η be equivalence relations on \mathcal{W} defined as:

$$\begin{aligned} w \sim_w^f w' &\iff f_w(x) = Af_{w'}(x) + c \\ w \sim_w^\eta w' &\iff \eta_w(\theta) = B\eta_{w'}(\theta) + e \end{aligned}$$

where A and B are $(d_s \times d_s)$ -matrices of rank at least $\min(d_s, d)$ and $\min(d_s, p)$ respectively, and c and e are vectors.

Definition 20 Strong identifiability (Section 2.2 in Khemakhem et al., 2020). Let \sim_s^f and \sim_s^η be equivalence relations on \mathcal{W} defined as:

$$\begin{aligned} w \sim_s^f w' &\iff \forall i, f_{i,w}(x) = a_i f_{\sigma(i),w'}(x) + c_i \\ w \sim_s^\eta w' &\iff \forall i, \eta_{i,w}(\theta) = b_i \eta_{\gamma(i),w'}(\theta) + e_i \end{aligned}$$

where σ and γ are permutations of $[[1, n]]$, a_i and b_i are non-zero scalars and c_i and e_i are scalars.

Weak identifiability means that two parameters are equivalent if the corresponding feature extractors are the same up to linear transformation. Strong identifiability is a specific case of the weak one, in which the linear transformation is restricted to be a scaled permutation.

After introducing these concepts, Khemakhem et al. (2020) provides two theorems (Theorem 1 and 2) in which weak or strong identifiability of the representations are implied if different parameter values lead to same distributions, i.e. $p_w(x|\theta) = p_{w'}(x|\theta) \forall x, \theta \implies w \sim_w^f w'$ or $p_w(x|\theta) = p_{w'}(x|\theta) \forall x, \theta \implies w \sim_s^f w'$ (and similar for \sim_w^η and \sim_s^η). These two results hold under some strict conditions on the functional form of the feature extractors f and η (concerning differentiability, rank of the Jacobian and other properties).

Then, they verify explicitly these conditions for a simple fully connected Neural Network architecture, in which they restrict the activation functions to be LeakyReLUs and the layer widths to be monotonically increasing or decreasing. Of course, this architecture is quite restrictive; it is in fact impossible to study theoretically the properties of more complex architectures and to show that they satisfy the necessary conditions for identifiability to hold. However, they show empirically that the conditional structure, even with more complex architectures, is helpful in increasing identifiability of the representations (as computed by the MCC, see Appendix B.1.1).

Therefore, even if the architectures which are used throughout this work do not satisfy the assumptions needed to explicitly show identifiability of the representations, we argue that the presence of such results is a hint towards the fact that identifiability (according to the above definitions) is actually achievable.

B.1.1 MEAN CORRELATION COEFFICIENT (MCC)

In order to evaluate empirically whether the above identifiability results are satisfied, we need a way to measure similarity between two embeddings of the same set of data. By following Khemakhem et al. (2020), we describe here the Mean Correlation Coefficient

(MCC), which is a simple measure to do that. In the main text, we used this technique to assess how well our approximating family recovered the exact sufficient statistics and natural parameters, in the case where the data generating model p_0 belongs to the exponential family.

In the following, we describe two versions of MCC, which are directly linked to the weak and strong identifiability definitions in Appendix B.1.

Strong MCC. Let us consider two sets of embeddings $\{y_i \in \mathbb{R}^{d_s}\}_{i=1}^n$ and $\{z_i \in \mathbb{R}^{d_s}\}_{i=1}^n$, which can be thought of as samples from two multivariate random variables $Y = t_1(X)$ and $Z = t_2(X)$, for two functions t_1, t_2 and a random variable X ; in general, the order of the components of these two vectors is arbitrary, so that we cannot say, for instance, that the 1st component of Y corresponds to the 1st of Z . Then, MCC computes all the correlation coefficients between each pair of components of Y and Z . Next, it solves a linear sum assignment problem which identifies each component of Y with a component of Z aiming to maximize the sum of the absolute value of the corresponding correlation coefficient. In this way, it tries to couple the embeddings which are most linear one to the other. Finally, the MCC is computed as the mean of the absolute correlation coefficients after the right permutation of elements of the vector.

MCC is therefore a metric between 0 and 1 which measures how well each component of the original embedding (say, Y) has been recovered independently by the other one (say, Z). Moreover, as it relies on the correlation coefficient, it is not sensitive to rescaling or translation of each of the embeddings (in fact, the correlation coefficient of two univariate random variables is ± 1 when a deterministic linear relationship between the two exists, unless the relationship is perfectly horizontal or vertical).

Finally, in order to get a better estimate of the MCC, we can split the set of embeddings in two: one which will be used to determine the permutation and which will give an in-sample estimate of the MCC, and the other one to which the previous permutation will be applied and will give an out-of-sample estimate of MCC.

Weak MCC. The above definition of MCC aims to estimate how well each single element of the embedding is recovered, independently on the other; we call it, following Khemakhem et al. (2020), strong MCC. However, there may be cases where the recovered embedding is equal to the correct one up to a more generic linear transformation A . In that case, we would like to have a way to measure, up to a linear transformation, how far are two embeddings. Following Khemakhem et al. (2020), we therefore apply Canonical Correlation Analysis (CCA) Hotelling (1936) to learn A , and after that compute MCC, which we will call weak MCC. Specifically, CCA is a way to compute linear transformation A so that the correlations between corresponding components of $A \cdot Y$ and Z is maximized. The so-defined weak MCC is therefore a number between 0 and 1 which measures how close is Y to Z after the best linear transformation is applied to Y . Similarly as before, a part of data is used to learn the best embedding; we can therefore use fresh samples to get an out-of-sample estimate along the in-sample one.

B.2 Universal approximation capability

Khemakhem et al. (2020) provide a result (Theorem 3) in which they prove universal approximation capability of the conditional exponential family. Specifically, they show that, considering the dimension of the representations d_s as an additional parameter, it is possible to find an arbitrarily good approximation of any conditional probability density $p_0(x|\theta)$, provided that \mathcal{X} and Θ are compact Hausdorff spaces. In practice, good approximation may be achieved with a value of d_s larger than d or p . Moreover, as remarked in the main text, this result is not concerned with the way the approximating family is fit; indeed, we expect that this task becomes (both statistically and computationally) more challenging when d_s increases.

C. Some properties of Score Matching

C.1 Proof of Theorem 6

Our Theorem 6 extends Theorem 1 in Hyvärinen (2005), that considers the case $\mathcal{X} = \mathbb{R}^d$, and which is recovered in the case $a_i = -\infty$ and $b_i = +\infty \forall i$. Our proof follows Hyvärinen (2005), which however explicitly stated only Assumptions **A1** and **A2**. Following Yu et al. (2019), we add the additional Assumption **A3** which is required for Fubini-Tonelli theorem to apply.

In order to prove Theorem 6, Assumption **A3** can be weakened to $\mathbb{E}_{p_0} \left| \frac{\partial^2}{\partial x_i^2} \log p_w(X) \right| < \infty, \forall w, \forall i = 1, \dots, d$. We state the more general one in the main text as that allows Theorem 9 for SSM to be proved as well.

Proof Let $s_0(x) = \nabla_x \log p_0(x)$ denote the score of the data distribution, and analogously $s(x; w) = \nabla_x \log p_w(x)$. Then, Eq. (3) can be rewritten as:

$$\begin{aligned} D_F(p_0 \| p_w) &= \frac{1}{2} \int_{\mathcal{X}} p_0(x) \|s_0(x) - s(x; w)\|^2 dx \\ &= \frac{1}{2} \int_{\mathcal{X}} p_0(x) [\|s_0(x)\|^2 + \|s(x; w)\|^2 - 2s_0(x)^T s(x; w)] dx \\ &= \underbrace{\frac{1}{2} \int_{\mathcal{X}} p_0(x) \|s_0(x)\|^2 dx}_C + \underbrace{\frac{1}{2} \int_{\mathcal{X}} p_0(x) \|s(x; w)\|^2 dx}_A - \underbrace{\int_{\mathcal{X}} p_0(x) s_0(x)^T s(x; w) dx}_B \end{aligned}$$

Note that we can split the integral into the three parts as the first two are assumed to be finite in **A2**; as a consequence, B is also finite thanks to $|2ab| \leq a^2 + b^2$.

The first element does not depend on w , so that we can safely ignore that. The second one appears as it is in the final Eq. (4). Therefore, we will focus on the last term, which we can write as:

$$B = - \sum_{i=1}^d \int_{\mathcal{X}} p_0(x) s_{0,i}(x) s_i(x; w) dx; \quad (12)$$

Let's now consider a single i , which we can rewrite as:

$$- \int_{\mathcal{X}} p_0(x) \frac{\partial \log p_0(x)}{\partial x_i} s_i(x; w) dx = - \int_{\mathcal{X}} \frac{p_0(x)}{p_0(x)} \frac{\partial p_0(x)}{\partial x_i} s_i(x; w) dx = - \int_{\mathcal{X}} \frac{\partial p_0(x)}{\partial x_i} s_i(x; w) dx.$$

Now, let's consider the integral over x_i first, and apply partial integration to it. Doing this relies on the Fubini-Tonelli's theorem, which can be safely applied due to B being finite as discussed above. We now apply partial integration to the integral over x_i :

$$\begin{aligned}
 & - \int_{a_1}^{b_1} \int_{a_2}^{b_2} \dots \int_{a_d}^{b_d} \frac{\partial p_0(x)}{\partial x_i} s_i(x; w) dx = - \int_{a_1}^{b_1} \dots \int_{a_{i-1}}^{b_{i-1}} \int_{a_{i+1}}^{b_{i+1}} \dots \int_{a_d}^{b_d} \left[p_0(x) s_i(x; w) \Big|_{x_i \searrow a_i}^{x_i \nearrow b_i} \right. \\
 & \quad \left. - \int_{a_i}^{b_i} p_0(x) \frac{\partial s_i(x; w)}{\partial x_i} dx_i \right] dx_1 \dots dx_{i-1} dx_{i+1} \dots dx_d \\
 & = \int_{a_1}^{b_1} \dots \int_{a_{i-1}}^{b_{i-1}} \int_{a_{i+1}}^{b_{i+1}} \dots \int_{a_d}^{b_d} \left[\int_{a_i}^{b_i} p_0(x) \frac{\partial s_i(x; w)}{\partial x_i} dx_i \right] dx_1 \dots dx_{i-1} dx_{i+1} \dots dx_d \\
 & = \int_{\mathcal{X}} p_0(x) \frac{\partial s_i(x; w)}{\partial x_i} dx.
 \end{aligned}$$

where the second equality holds thanks to Assumption **A1**. For stating the last equality rigorously, we need to invoke Fubini-Tonelli's theorem again; this relies on the assumption:

$$\int_{\mathcal{X}} \left| p_0(x) \frac{\partial s_i(x; w)}{\partial x_i} \right| dx = \mathbb{E}_{p_0} \left| \frac{\partial s_i(X; w)}{\partial x_i} \right| < \infty,$$

which is equivalent to Assumption **A3**.

By repeating this argument for all terms in the sum in Eq. (12), we obtain that:

$$B = \int_{\mathcal{X}} p_0(x) \sum_{i=1}^d \frac{\partial s_i(x; w)}{\partial x_i} dx = \int_{\mathcal{X}} p_0(x) \sum_{i=1}^d \frac{\partial^2 \log p_w(x)}{\partial x_i^2} dx,$$

which concludes our proof. ■

C.2 Proof of Theorem 7

We give here an extended version of Theorem 7, which we then prove following Hyvärinen (2005).

Theorem 21 *Assume $\exists w^* : p_0(\cdot) = p_{w^*}(\cdot)$. Then:*

$$w = w^* \implies D_F(p_0 \| p_w) = 0.$$

Further, if $p_0(x) > 0 \forall x \in \mathcal{X}$, you also have:

$$D_F(p_0 \| p_w) = 0 \implies p_w(\cdot) = p_0(\cdot).$$

Finally, if no other value $w \neq w^$ gives a pdf p_w that is equal to p_{w^*} :*

$$D_F(p_0 \| p_w) = 0 \implies w = w^*.$$

Proof The first statement is straightforward. For the second one, if $D_F(p_0||p_w) = 0$ and $p_0(x) > 0 \forall x \in \mathcal{X}$, then $\nabla_x \log p_w(x) = \nabla_x \log p_0(x)$; this in turn implies that $\log p_w(x) = \log p_0(x) + c \forall x \in \mathcal{X}$ for a constant c , which however is 0 as both p_0 and p_w are pdf's.

The third statement follows from the second as, for the additional assumption, w^* is the only choice of w which gives $p_w = p_0$. \blacksquare

C.3 Proof of Theorem 9

Similarly to the SM case, our Theorem 9 extends Theorem 1 in Song et al. (2020), that considers the case $\mathcal{X} = \mathbb{R}^d$, and which is recovered in the case $a_i = -\infty$ and $b_i = +\infty \forall i$. Our proof follows Song et al. (2020), which however explicitly stated only Assumptions **A1**, **A2** and **A4**. Following Yu et al. (2019), we add the additional Assumption **A3** which is required for Fubini-Tonelli theorem to apply. The strategy of the proof is very similar to Theorem 6.

Proof As before, let $s_0(x) = \nabla_x \log p_0(x)$ denote the score of the data distribution, and analogously $s(x; w) = \nabla_x \log p_w(x)$. Then, Eq. (5) can be rewritten as:

$$\begin{aligned} D_{FS}(p_0||p_w) &= \frac{1}{2} \int_{\mathcal{V}} \int_{\mathcal{X}} q(v) p_0(x) [v^T s_0(x) - v^T s(x; w)]^2 dx dv \\ &= \frac{1}{2} \int_{\mathcal{V}} \int_{\mathcal{X}} q(v) p_0(x) [(v^T s_0(x))^2 + (v^T s(x; w))^2 - 2(v^T s_0(x))(v^T s(x; w))] dx dv \\ &= \int_{\mathcal{V}} \int_{\mathcal{X}} q(v) p_0(x) \left[\frac{1}{2} (v^T s(x; w))^2 - (v^T s_0(x))(v^T s(x; w)) \right] dx dv + C \end{aligned}$$

Note that we can split the integral into the three parts thanks to Assumptions **A2** and **A4**, which ensure that the expectation of each term is bounded. Additionally, we have absorbed in the constant C the term which does not depend on w .

In the last row of the above Equation, the first term in the square brackets appears in Eq. (6), which is what we want to prove. We focus therefore on the second term:

$$\begin{aligned} & - \int_{\mathcal{V}} \int_{\mathcal{X}} q(v) p_0(x) [(v^T s_0(x))(v^T s(x; w))] dx dv \\ &= - \int_{\mathcal{V}} \int_{\mathcal{X}} q(v) p_0(x) [(v^T \nabla_x \log p_0(x))(v^T \nabla_x \log p_w(x))] dx dv \\ &= - \int_{\mathcal{V}} \int_{\mathcal{X}} q(v) [(v^T \nabla_x p_0(x))(v^T \nabla_x \log p_w(x))] dx dv \tag{13} \\ &= - \sum_{i=1}^d \int_{\mathcal{V}} \int_{\mathcal{X}} q(v) \left[v_i \frac{\partial p_0(x)}{\partial x_i} (v^T \nabla_x \log p_w(x)) \right] dx dv \end{aligned}$$

We will now consider one single element of the sum for a chosen i , and consider the integral over x_i first. Swapping the order of integration relies on Fubini-Tonelli's theorem, which can be safely applied due to the above quantity being finite as discussed above. We then

apply partial integration to the integral over x_i :

$$\begin{aligned}
 & - \int_{\mathcal{V}} \int_{a_1}^{b_1} \int_{a_2}^{b_2} \cdots \int_{a_d}^{b_d} q(v) \left[v_i \frac{\partial p_0(x)}{\partial x_i} (v^T \nabla_x \log p_w(x)) \right] dx dv \\
 &= - \int_{\mathcal{V}} \int_{a_1}^{b_1} \cdots \int_{a_{i-1}}^{b_{i-1}} \int_{a_{i+1}}^{b_{i+1}} \cdots \int_{a_d}^{b_d} q(v) \left[v_i p_0(x) (v^T \nabla_x \log p_w(x)) \Big|_{x_i \searrow a_i}^{x_i \nearrow b_i} \right. \\
 & \quad \left. - \int_{a_i}^{b_i} v_i p_0(x) \left(v^T \frac{\partial}{\partial x_i} \nabla_x \log p_w(x) \right) dx_i \right] dx_{-i},
 \end{aligned}$$

where $dx_{-i} = dx_1 \dots dx_{i-1} dx_{i+1} \dots dx_d$. The first element in the square bracket goes to 0 thanks to Assumption **A1**. We are left therefore with:

$$\begin{aligned}
 & \int_{\mathcal{V}} \int_{a_1}^{b_1} \cdots \int_{a_{i-1}}^{b_{i-1}} \int_{a_{i+1}}^{b_{i+1}} \cdots \int_{a_d}^{b_d} \left[\int_{a_i}^{b_i} q(v) p_0(x) v_i \left(v^T \frac{\partial}{\partial x_i} \nabla_x \log p_w(x) \right) dx_i \right] dx_{-i} \\
 &= \int_{\mathcal{V}} \int_{\mathcal{X}} q(v) p_0(x) v_i \left(v^T \frac{\partial}{\partial x_i} \nabla_x \log p_w(x) \right) dx dv.
 \end{aligned}$$

The last equality again exploits Fubini-Tonelli theorem, which in this case requires:

$$\int_{\mathcal{V}} \int_{\mathcal{X}} q(v) p_0(x) \left| v_i \left(v^T \frac{\partial}{\partial x_i} \nabla_x \log p_w(x) \right) \right| dx dv = \mathbb{E}_{V \sim q} \mathbb{E}_{X \sim p_0} \left| V_i \left(V^T \frac{\partial}{\partial x_i} \nabla_x \log p_w(X) \right) \right| < \infty.$$

This is satisfied by combining Assumptions **A3** and **A4**, as in fact:

$$\begin{aligned}
 & \mathbb{E}_{V \sim q} \mathbb{E}_{X \sim p_0} \left| V_i \left(V^T \frac{\partial}{\partial x_i} \nabla_x \log p_w(X) \right) \right| = \mathbb{E}_{V \sim q} \mathbb{E}_{X \sim p_0} \left| V_i \sum_{j=1}^d V_j \frac{\partial^2}{\partial x_i \partial x_j} \log p_w(X) \right| \\
 & \leq \sum_{j=1}^d \mathbb{E}_{V \sim q} \mathbb{E}_{X \sim p_0} \left| V_i V_j \frac{\partial^2}{\partial x_i \partial x_j} \log p_w(X) \right| = \sum_{j=1}^d \mathbb{E}_{V \sim q} |V_i V_j| \cdot \mathbb{E}_{X \sim p_0} \left| \frac{\partial^2}{\partial x_i \partial x_j} \log p_w(X) \right| \\
 & \leq \sum_{j=1}^d \sqrt{\mathbb{E}_{V \sim q} V_i^2 \cdot \mathbb{E}_{V \sim q} V_j^2} \cdot \mathbb{E}_{X \sim p_0} \left| \frac{\partial^2}{\partial x_i \partial x_j} \log p_w(X) \right|,
 \end{aligned}$$

where the last inequality holds thanks to Cauchy-Schwarz inequality; Assumptions **A3** and **A4** ensure the last row is $< \infty$.

Back to Eq. (13), we have therefore:

$$\begin{aligned}
 & - \sum_{i=1}^d \int_{\mathcal{V}} \int_{\mathcal{X}} q(v) \left[v_i \frac{\partial p_0(x)}{\partial x_i} (v^T \nabla_x \log p_w(x)) \right] dx dv \\
 &= \int_{\mathcal{V}} \int_{\mathcal{X}} \sum_{i=1}^d q(v) p_0(x) v_i \left(v^T \frac{\partial}{\partial x_i} \nabla_x \log p_w(x) \right) dx dv \\
 &= \int_{\mathcal{V}} \int_{\mathcal{X}} \sum_{i=1}^d \sum_{j=1}^d q(v) p_0(x) v_i v_j \frac{\partial^2}{\partial x_i \partial x_j} \log p_w(x) dx dv \\
 &= \int_{\mathcal{V}} \int_{\mathcal{X}} q(v) p_0(x) [v^T (H_x \log p_w(x)) v] dx dv.
 \end{aligned}$$

■

C.4 Proof of Theorem 10

We give here an extended version of Theorem 10. This is a version of Lemma 1 in Song et al. (2020), whose proof we adapt.

Theorem 22 *Assume $\exists w^* : p_0(\cdot) = p_{w^*}(\cdot)$. Then:*

$$w = w^* \implies D_{FS}(p_0 \| p_w) = 0.$$

Further, if $p_0(x) > 0 \forall x \in \mathcal{X}$, you also have:

$$D_{FS}(p_0 \| p_w) = 0 \implies p_w(\cdot) = p_0(\cdot).$$

Finally, if no other value $w \neq w^$ gives a pdf p_w that is equal to p_{w^*} :*

$$D_{FS}(p_0 \| p_w) = 0 \implies w = w^*.$$

Proof The first statement is straightforward.

For the second one, if $D_F(p_0 \| p_w) = 0$ and $p_0(x) > 0 \forall x \in \mathcal{X}$, then:

$$\begin{aligned} & \int_{\mathcal{Y}} q(v) (v^T (\nabla_x \log p_0(x) - \nabla_x \log p_w(x)))^2 dv = 0 \\ \iff & \int_{\mathcal{Y}} q(v) v^T (\nabla_x \log p_0(x) - \nabla_x \log p_w(x)) (\nabla_x \log p_0(x) - \nabla_x \log p_w(x))^T v dv = 0 \\ \iff & (\nabla_x \log p_0(x) - \nabla_x \log p_w(x))^T E[VV^T] (\nabla_x \log p_0(x) - \nabla_x \log p_w(x)) = 0 \\ \stackrel{(\star)}{\iff} & \nabla_x \log p_0(x) - \nabla_x \log p_w(x) = 0 \\ \iff & \log p_w(x) = \log p_0(x) + c \quad \forall x \in \mathcal{X}, \end{aligned}$$

where in the third line above V is a random variable distributed according to q , for which therefore $\mathbb{E}[VV^T]$ is positive definite by Assumption **A4**, which ensures equivalence (\star) holds. Additionally, as both p_0 and p_w are pdf's (and therefore normalized), the constant $c = 0$.

The third statement follows from the second as, for the additional assumption, w^* is the only choice of w which gives $p_w = p_0$. ■

C.5 Proof of Theorem 12

We give here an extended version of Theorem 12, which we then prove.

Theorem 23 *Let $Y = t(X) \in \mathcal{Y}$ for a bijection t , and denote by p_0^Y and p_w^Y the distributions on \mathcal{Y} induced by the distributions p_0 and p_w on \mathcal{X} . Assume $\exists w^* : p_0(\cdot) = p_{w^*}(\cdot)$, and let D denote either D_F or D_{FS} . Then:*

$$w = w^* \implies D(p_0^Y \| p_w^Y) = 0.$$

Further, if $p_0(x) > 0 \forall x \in \mathcal{X}$ and Assumption **A4** holds, you also have:

$$D(p_0^Y \| p_w^Y) = 0 \implies p_w(\cdot) = p_0(\cdot).$$

Finally, if no other value $w \neq w^*$ gives a pdf p_w that is equal to p_{w^*} :

$$D(p_0^Y \| p_w^Y) = 0 \implies w = w^*.$$

Proof The proof relies on the equivalence between distributions for the random variables Y and X ; in fact, by fixing $y = t(x)$ and denoting by $|J_t(x)|$ the determinant of the Jacobian matrix of t evaluated in x , we have that $p_0^Y(y) = \frac{p_0(x)}{|J_t(x)|}$ and $p_w^Y(y) = \frac{p_w(x)}{|J_t(x)|}$, so that $p_0(\cdot) = p_w(\cdot) \iff p_0^Y(\cdot) = p_w^Y(\cdot)$. The first and third statements follow directly from this fact by applying Theorem 21 (if D is chosen to be D_F) or Theorem 22 (if $D = D_{FS}$) to $D(p_0^Y \| p_w^Y)$; for the second, notice also that $p_0(x) > 0 \forall x \in \mathcal{X} \implies p_0^Y(y) \forall y \in \mathcal{Y} > 0$ as t is a bijection; then, Theorem 21 or Theorem 22 imply that $p_0^Y(\cdot) = p_w^Y(\cdot) \implies p_0(\cdot) = p_w(\cdot)$. ■

C.6 Discussion on the positivity condition for SM

We follow here the discussion in Appendix D of Arbel and Gretton (2018).

Consider again the Fisher divergence in Eq. (3); we want to understand the conditions under which this is a divergence between probability measures, which essentially means it is 0 if and only if the probability distributions are the same.

Besides the fact that both p_0 and p_w need to be continuous (otherwise the gradient would be a delta function), it turns out that a necessary condition is that p_0 is positive on the whole space to which the random variable belong (let's say \mathcal{X}), if we don't put any restrictions on p_w . Otherwise, the following scenario may happen (see Appendix D in Arbel and Gretton, 2018): consider the case in which $p_0(x)$ is a mixture of two densities supported on disjoint sets: $p_0(x) = \alpha_A p_A(x) + \alpha_B p_B(x)$, with non-negative weights α_A, α_B , such that

$$p_A(x) > 0 \iff x \in \mathcal{X}_A, \quad p_B(x) > 0 \iff x \in \mathcal{X}_B, \quad \mathcal{X}_A, \mathcal{X}_B \in \mathcal{X}; \quad \mathcal{X}_A \cap \mathcal{X}_B = \emptyset.$$

Note that this implies $\exists x \in \mathcal{X} : p_0(x) = 0$. In this setting, any p_w of the form $p_w(x) = \beta_A p_A(x) + \beta_B p_B(x)$ will give $D_F(p_0 \| p_w) = 0$. This can be seen by computing the Fisher divergence directly:

$$\begin{aligned}
 D_F(p_0 \| p_w) &= \frac{1}{2} \int_{\mathcal{X}} p_0(x) \|\nabla_x \log p_0(x) - \nabla_x \log p_w(x)\|^2 dx \\
 &= \frac{1}{2} \int_{\mathcal{X}} p_0(x) \|\nabla_x \log(\alpha_A p_A(x) + \alpha_B p_B(x)) - \nabla_x \log(\beta_A p_A(x) + \beta_B p_B(x))\|^2 dx \\
 &= \frac{1}{2} \int_{\mathcal{X}_A} p_0(x) \|\nabla_x \log(\alpha_A p_A(x)) - \nabla_x \log(\beta_A p_A(x))\|^2 dx + \\
 &\quad \frac{1}{2} \int_{\mathcal{X}_B} p_0(x) \|\nabla_x \log(\alpha_B p_B(x)) - \nabla_x \log(\beta_B p_B(x))\|^2 dx \\
 &= \frac{1}{2} \int_{\mathcal{X}_A} p_0(x) \underbrace{\|\nabla_x \log p_A(x) - \nabla_x \log p_A(x)\|}_{=0}^2 dx + \\
 &\quad \frac{1}{2} \int_{\mathcal{X}_B} p_0(x) \underbrace{\|\nabla_x \log p_B(x) - \nabla_x \log p_B(x)\|}_{=0}^2 dx \\
 &= 0,
 \end{aligned}$$

where the third equality relies on splitting the integration domain over the two subsets on which p_A and p_B are supported (and the other is 0) and the fourth equality relies on the presence of the logarithmic derivatives, that removes the importance of the mixture weights.

Due to the above, in the case of the conditional Fisher divergence in Eq. (8), $p_0(x|\theta)$ needs to be supported on the whole \mathcal{X} for each θ in order for $D_F^E(p_0|p_w) = 0 \iff p_0(\cdot|\theta) = p_w(\cdot|\theta) \pi(\theta)$ -almost everywhere. This can be seen by considering the case of univariate parameter θ and by choosing $p_0(x|\theta) = p_A(x)H(\theta) + (1 - H(\theta))p_B(x)$, where p_A and p_B are as above and $H(\cdot)$ represents the Heaviside function. In this case, choosing $p_w(x|\theta) = q(x)$, where $q(x) = \beta_A p_A(x) + \beta_B p_B(x)$ will give $D_F^E(p_0|p_w) = 0$, as for each fixed θ , $D_F(p_0(\cdot|\theta)|p_w(\cdot|\theta))$ falls in the case considered above.

C.7 Equivalence of Correction Factor and Transformed Score Matching

As discussed in the main text (Section 3.2), the first extension of score matching to non-negative random variables was given in Hyvärinen (2007):

$$D_F^+(p_0 \| p_w) = \frac{1}{2} \int_{\mathbb{R}_+^d} p_0(x) \|\nabla_x \log p_0(x) \odot x - \nabla_x \log p_w(x) \odot x\|^2 dx, \quad (14)$$

where \odot denotes element wise product between vectors and \mathbb{R}_+^d is the positive octant of \mathbb{R}^d . The correction factor x relaxes assumption **A1** to $p_0(x)x_i^2 \frac{\partial}{\partial x_i} \log p_w(x) \rightarrow 0$, so that it is possible to get an explicit form of the above with looser assumptions. This is an example of Corrected Score Matching (CorrSM, 3.2), in which the issue arising due to distribution having a compact support is fixed by introducing a correction factor in the formulation of the objective.

Yu et al. (2019) proposed a more general score matching for non-negative random variables by allowing freedom of choice in the factor that is used in the integrand to correct for the integration by parts step (Appendix C.1), leading to the following objective:

$$D_F^+(p_0 \| p_w) = \frac{1}{2} \int_{\mathbb{R}_+^n} p_0(x) \|(\nabla_x \log p_0(x)) \odot \sqrt{h(x)} - (\nabla_x \log p_w(x)) \odot \sqrt{h(x)}\|^2 dx, \quad (15)$$

where $h(x)$ has the same dimension as x , and has positive elements almost surely.

The explicit formulation associated to Eq. (15) can be obtained under the following assumptions:

A1b $p_0(x)h_j(x_j)\partial_j \log p_w(x) \rightarrow 0$ for $x_i \searrow 0$ and $x_i \nearrow \infty$, $\forall w, i$,

A2b $\mathbb{E}_{p_0} \|\nabla_x \log p_0(X) \odot h^{1/2}(X)\|_2^2 < \infty$, $\mathbb{E}_{p_0} \|\nabla_x \log p_w(X) \odot h^{1/2}(X)\|_2^2 < \infty \forall w$,

A3b $\mathbb{E}_{p_0} \|(\nabla_x \log p_w(X) \odot h(X))'\|_1 < \infty \forall w$, where the prime symbol denotes element-wise differentiation.

Under the above assumption, Eq. (15) is equal to:

$$D_F^+(p_0 \| p_w) = \int_{\mathbb{R}_+^n} p_0(x) \sum_{i=1}^d \left[\frac{1}{2} h_i(x) \left(\frac{\partial \log p_w(x)}{\partial x_i} \right)^2 + h_i(x) \left(\frac{\partial^2 \log p_w(x)}{\partial x_i^2} \right) + h_i'(x) \frac{\partial \log p_w(x)}{\partial x_i} \right] dx + C,$$

where C is a constant with respect to p_w .

In Proposition 2 in Yu et al. (2019), they give a result similar to our Theorems 7 and 12 guaranteeing that minimization of $D_F^+(p_0 \| p_w)$ is a valid procedure for estimating a probabilistic model. When considering the finite-sample estimate of C.7, different choices of $h(x)$ may allow to focus more on smaller/larger values of x , which may in practice have better properties than the original form for non-negative data in Hyvärinen (2007), which is recovered for $h(x) = x^2$ (where the square is applied element-wise).

This formulation in Eq. (15), albeit originally considered for non-negative random variables only, can be extended to random variables with any bounded domain, by choosing a suitable function h and modifying **A1b** to hold for x_i going to the limits of the domain. In the next Sections, we therefore compare this approach with TranSM without specifying the domain; we will show that both the implicit and explicit formulation are the same with both TranSM and CorrSM, implying that the two are equivalent (we will show this in the specific case in which the transformation and the function h act independently on the different coordinates, but we believe this to be the case more in general; see Appendix C.9).

C.7.1 EQUIVALENCE OF THE IMPLICIT FORM

As mentioned in the main text (Section 3.2), another approach to apply Fisher divergence to distributions with bounded domain (on one side or both) is to transform the data space to the real line and then apply standard Fisher divergence; we called this Transformation Score Matching (TranSM). Let us denote t such a transformation, which we assume to be bijective. Then, starting from $p_0(x)$ and $p_w(x)$ we get $p_0(y) = \frac{p_0(x)}{|J_t(x)|}$ and $p_w(y) = \frac{p_w(x)}{|J_t(x)|}$ for $y = t(x)$, where J_t is the Jacobian matrix of t and $|\cdot|$ denotes here the determinant; here, differently from the main text, we use a lighter notation where $p_0(y)$ and $p_0(x)$ are two different densities associated to the different name of the argument (same for p_w). We investigate what is the Fisher divergence between the densities of the transformed distributions. Recall that $p_0(y)dy = p_0(x)dx$, for $y = t(x)$. Moreover, we also have that:

$$\nabla_y g(y) = J_{t^{-1}}(t(x)) \nabla_x g(t(x)) = (J_t(x))^{-1} \nabla_x g(t(x)),$$

where the second equality comes from the fact that $J_{t^{-1}}(t(x)) = (J_t(x))^{-1}$ due to the inverse function theorem. Then, we can compute the Fisher Divergence between $p_0(y)$ and $p_w(y)$ (corresponding to the TransSM objective):

$$\begin{aligned} D_F(p_0(y) \| p_w(y)) &= \frac{1}{2} \int p_0(y) \|\nabla_y \log p_0(y) - \nabla_y \log p_w(y)\|^2 dy \\ &= \frac{1}{2} \int p_0(x) \|(J_t(x))^{-1} [\nabla_x \log p_0(x) - \nabla_x \log |J_t(x)| - \nabla_x \log p_w(x) + \nabla_x \log |J_t(x)|]\|^2 dx \\ &= \frac{1}{2} \int p_0(x) \|(J_t(x))^{-1} [\nabla_x \log p_0(x) - \nabla_x \log p_w(x)]\|^2 dx. \end{aligned}$$

In the rather common case in which the transformation t acts on each component independently, the Jacobian matrix is diagonal; in this case, the latter is equivalent to Eq. (15) upon defining $h(x)$ to be a vector containing the squares of the diagonal elements of the Jacobian, i.e. putting $\sqrt{h_i(x)} = (J_t(x))_{ii}^{-1}$.

C.7.2 EQUIVALENCE OF THE EXPLICIT FORM

For both TransSM and CorrSM it is possible to get an explicit form of the objective (Eqs. 4 and C.7), in which the integrand does not depend on the data distribution p_0 . In case in which the transformation t acts on the different components independently, the original explicit divergence for the transformed variable $Y = t(X)$ is equivalent to the corrected explicit divergence for the original X , analogously to the implicit Fisher divergence form. In fact, by applying the definition of explicit Fisher divergence (Eq. 4) to the transformed Y , you get:

$$D_F(p_0(y) \| p_w(y)) = \int p_0(y) \underbrace{\sum_{i=1}^d \left[\frac{1}{2} \left(\frac{\partial \log p_w(y)}{\partial y_i} \right)^2 + \left(\frac{\partial^2 \log p_w(y)}{\partial y_i^2} \right) \right]}_{\star} dy + C$$

where C is a constant with respect to p_w . Considering only the term in square brackets, denoting $\partial_i = \frac{\partial}{\partial x_i}$ and setting $\sqrt{h_i(x)} = (J_t(x))_{ii}^{-1}$, we get:

$$\begin{aligned} \star &= \frac{1}{2} h_i(x) \left[(\partial_i \log p_w(x))^2 + \frac{1}{4} (\partial_i \log h_i(x))^2 + \partial_i \log p_w(x) \cdot \partial_i \log h_i(x) \right] \\ &+ \frac{1}{2} h'_i(x) \cdot \partial_i \log p_w(x) + \frac{1}{4} h'_i(x) \partial_i \log h_i(x) + h_i(x) \partial_i^2 \log p_w(x) + \frac{1}{2} h_i(x) \partial_i^2 \log h_i(x) \\ &= \frac{1}{2} h_i(x) (\partial_i \log p_w(x))^2 + \frac{1}{8} h_i(x) (\partial_i \log h_i(x))^2 \\ &+ h'_i(x) \cdot \partial_i \log p_w(x) + \frac{1}{4} h'_i(x) \partial_i \log h_i(x) + h_i(x) \partial_i^2 \log p_w(x) + \frac{1}{2} h_i(x) \partial_i^2 \log h_i(x). \end{aligned}$$

The blue terms are the same that appear in the CorrSM explicit formulation (Eq. C.7), while all other terms are constants with respect to p_w .

We have shown therefore that CorrSM and TransSM are equivalent in both the explicit and implicit formulation if the transformation is applied independently on the elements of

x . Therefore, the two approaches are completely equivalent when it comes to minimizing them.

C.8 Specific formulation of TransSM

We discuss here the transformations we apply in this work in TransSM; specifically, we only consider the case in which the support for the multivariate x is defined by an intersection of element-wise inequalities, i.e. $x \in \bigotimes_{i=1}^d (a_i, b_i)$, where a_i, b_i can take on the values $\pm\infty$ as well. In this case, then, a transformation can be applied independently on each element of x . We consider here the following transformations (which are also used in the Stan package Carpenter et al., 2017):

- When $X \in [0, \infty)^d$, the transformation we use is $y_i = \log(x_i) \in \mathbb{R}^d$. This corresponds to diagonal Jacobian with elements $(J_t(x))_{ii}^{-1} = x_i$, so that the above expression becomes the same as the original Fisher divergence for non-negative random variables discussed in Eq. (14).
- More generally, if $x_i \in [a_i, +\infty)$ for $|a_i| < \infty$, we can transform the data as $y_i = \log(x_i - a_i) \in \mathbb{R}$, while if $x_i \in (-\infty, b_i]$ for $|b_i| < \infty$, we simply reverse the transformation: $y_i = \log(b_i - x_i) \in \mathbb{R}$. These correspond to $(J_t(x))_{ii}^{-1} = x_i - a_i$ and $(J_t(x))_{ii}^{-1} = b_i - x_i$.
- Finally, if $x_i \in (a_i, b_i)$ for $|a_i|, |b_i| < \infty$, we can use the transformation defined as: $y_i = t(x_i) = \text{logit}\left(\frac{x_i - a_i}{b_i - a_i}\right)$ with inverse transformation $x_i = t^{-1}(y_i) = a + (b - a) \frac{e^{y_i}}{e^{y_i} + 1}$. This corresponds to $(J_t(x))_{ii}^{-1} = \frac{(x_i - a_i)(b_i - x_i)}{b_i - a_i}$.

C.9 Score matching for distributions with more general domain

As highlighted in the main text, across this work we are concerned with applying score matching to distributions whose support is defined by independent constraints on the different coordinates, as for instance $\mathcal{X} = \bigotimes_{i=1}^d (a_i, b_i)$. That is arguably the most common case in the literature. However, there have been some recent works which applied SM to more general cases. For instance, Mardia et al. (2016) devised a way to apply it to a directional distribution defined on an oriented Riemannian manifold (for instance, a sphere). It is interesting how their derivation of the explicit form from the implicit one relies on the classical divergence theorem (also known as Stokes' theorem), of which the partial integration trick used in Theorem 6 is a specific case. Liu and Kanamori (2019) introduced instead a way to apply score matching for a distribution on Euclidean space with complex truncation boundaries; their approach boils down to introducing a smart correction factor which goes to 0 at the boundary (thus allowing partial integration) but still being tractable; again, they need a more general version of Theorem 6 to obtain an objective for which the integrand does not depend on the data distribution.

In Appendix C.7, we established that CorrSM and TransSM are equivalent if the transformation is applied independently on the elements of x , which requires the domain to be defined by independent constraints on the coordinates. In the more general case of a irregular subset of Euclidean space (as in the setup of Liu and Kanamori, 2019), it is not

clear whether it is always possible to associate a correction factor to a transformation which maps the space to \mathbb{R}^d . That seems to be plausible if the domain satisfies some regularity conditions which may be related to convexity (for instance think of a triangle in \mathbb{R}^2 , which can be easily stretched to cover the full space). We are not aware however of any work investigating this.

C.10 Score matching with exponential family

We consider here the exponential family:

$$p_w(x|\theta) = \exp(\eta_w(\theta)^T f_w(x))/Z_w(\theta),$$

and want to find the value of w minimizing either $D_F^E(p_0||p_w)$ or $D_{FS}^E(p_0||p_w)$, which are defined in Eq. (9).

Under the conditions discussed in Section 3.3, if $\pi(\theta) > 0 \forall \theta$, then $D_F^E(p_0||p_w) = 0$ and $D_{FS}^E(p_0||p_w) = 0 \iff p_w(x|\theta) = p_0(x|\theta) \pi(\theta)$ -almost everywhere. In this case, if f_w and η_w satisfy the conditions required for the theorems mentioned in Appendix B.1 to hold, then f_w and η_w are respectively sufficient statistics and natural parameters of p_0 .

In order to find the value of the empirical estimate of the explicit for of both $D_F^E(p_0||p_w)$ and $D_{FS}^E(p_0||p_w)$, we insert the definition of the exponential family with in Eq. 10, which leads to:

$$\begin{aligned} \hat{J}(w) &= \frac{1}{N} \sum_{j=1}^N \left[\sum_{i=1}^d \left(\frac{1}{2} \left(\eta_w(\theta^{(j)})^T \frac{\partial}{\partial x_i} f_w(x^{(j)}) \right)^2 + \eta_w(\theta^{(j)})^T \frac{\partial^2}{\partial x_i^2} f_w(x) \right) \right], \\ \hat{J}_S(w) &= \frac{1}{NM} \sum_{j=1}^N \sum_{k=1}^M \left[v^{(j,k),T} \mathbf{H}_x(\eta_w(\theta^{(j)})^T f_w(x^{(j)})) v^{(j,k)} + \frac{1}{2} \sum_{i=1}^d \left(\eta_w(\theta^{(j)})^T \frac{\partial}{\partial x_i} f_w(x^{(j)}) \right)^2 \right]. \end{aligned}$$

Note that the objective does not change if you set $f_w(x)$ to $c + f_w(x)$, for a constant vector c ; in fact, this constant gets absorbed into the normalizing constant in $p_w(x|\theta)$.

Similarly, $\eta_w(\theta)^T f_w(x) = (1/c \cdot \eta_w(\theta))^T (c \cdot f_w(x))$ for some constant $c \neq 0$. Therefore, statistics and corresponding parameters are only defined up to a scale with respect to one another; if you use two Neural Networks to learn both of them, different network initializations may lead to different learned statistics and natural parameters, but their product should be fixed (up to translation of $f_w(x)$).

However, this degeneracy may make training the approximate likelihood p_w with the score matching approach harder. In order to improve training, we usually add a Batch Normalization layer on top of the η_w network. Basically, Batch Normalization fixes the scale of the output of η_w over a training batch, therefore removing this additional degree of freedom and making training easier. We discuss in more detail this in Appendix D.2.

D. Computational practicalities

D.1 Computational cost of SM and SSM

For SM, as discussed in Section 3.2, exploiting automatic differentiation libraries to compute the second derivatives of the log density requires d times more backward derivative computations with respect to the first derivatives. In fact, automatic differentiation libraries are

able to compute derivatives of a scalar with respect to several variables at once. One single call is therefore sufficient to obtain $\nabla_x \log p_w(x)$. However, d additional calls are required to obtain the second derivatives $\frac{\partial^2}{\partial x_i^2} \log p_w(x)$, $i = 1, \dots, d$, which are the diagonal elements of the Hessian matrix of $\log p_w(x)$; each additional call computes the gradient of $\frac{\partial}{\partial x_k} \log p_w(x)$ with respect to all components of x , for some $k \in [1, 2, \dots, d]$. Algorithm 2 in Song et al. (2020) gives a pseudocode implementation of this approach. Computational improvement can be obtained by implementing custom code which performs the gradient computation in the forward pass (i.e., along the computation of the neural network output for a given input x). This avoids repeating some computations multiple times, which is done when performing the backward step repeatedly; however, the implementation is tricky and needs custom code for each different neural network type (we discuss how it can be done for a fully connected neural network in Appendix D.1.1). Additionally, the computational speed-up achievable in this way is limited with respect to what is offered by, for instance, SSM.

SSM instead requires only two backward propagation steps independently on the input size of the network. This is possible by exploiting the vector-Hessian product structure and computing the linear products with v (which is independent on the input x , thus can be swapped with gradient computation) after the gradient has been computed once, so that you only ever require the gradient of a scalar quantity. See Algorithm 1 in Song et al. (2020) for a precise description of how that can be done.

D.1.1 FORWARD COMPUTATION OF DERIVATIVES

In the standard score matching approach, the first and second derivatives of Neural Network outputs with respect to the inputs are required, namely:

$$\frac{\partial f_w(x)}{\partial x_i} \quad \text{and} \quad \frac{\partial^2 f_w(x)}{\partial x_i^2}.$$

Neural network training is possible thanks to the use of autodifferentiation libraries, which allow to keep trace of the different operations and to automatically compute the gradients required for training. These libraries can be used to obtain the above derivatives.

However, as discussed previously, it is more efficient to compute the required derivatives during the forward pass of training (i.e. when the output of the Neural Network for a given input is computed). This requires additional coding effort specific to the chosen Neural Network architecture. For instance, Avrutskiy (2017) provide formulas to compute derivatives of any order recursively for fully connected Neural Networks. For more complex NN architectures, this approach is not viable as the coding effort becomes substantial. Additionally, with large d the improvement obtained by forward derivatives computation is much smaller than what offered, for instance, by SSM.

In the current work, the forward derivatives approach has been implemented for fully connected Neural Networks and Partially Exchangeable Networks (Appendix D.3). The computational advantage is evident for the computation of the second derivatives, as shown below. This approach allowed us to apply SM to relatively high dimensional data spaces (up to 100 dimensional for the MA(2) and AR(2) case), but not to the larger Lorenz96 model.

Forward computation of derivatives of fully connected NNs We revisit here the work in Avrutskiy (2017). Let us consider a fully connected Neural Networks with L layers, where the weights and biases of the l -th layer are denoted by W_l and b_l , $l = 1, \dots, L$. Let us denote by x the input of the Neural Network, and by z_l the hidden values after the l -th layer, before the activation function (denoted by σ) is applied. Specifically, the Neural Network hidden values are determined by:

$$z_1 = W_1 \cdot x + b_1, \quad z_l = W_l \cdot \sigma(z_{l-1}) + b_l, \quad l = 2, \dots, L,$$

where the activation function is applied element wise. Similar recursive expressions can be given for the first derivatives:

$$\frac{\partial z_1}{\partial x_i} = (W_1)_{i,\cdot}, \quad \frac{\partial z_l}{\partial x_i} = W_l \cdot \left[\sigma'(z_{l-1}) \odot \frac{\partial z_{l-1}}{\partial x_i} \right], \quad l = 2, \dots, L,$$

where $(W_1)_{i,\cdot}$ denotes the i -th column of W_1 , and \odot denotes element wise multiplication. Expressions for the second derivatives are instead:

$$\frac{\partial^2 z_1}{\partial x_i^2} = \mathbf{0}, \quad \frac{\partial^2 z_l}{\partial x_i^2} = W_l \cdot \left[\sigma''(z_{l-1}) \odot \left(\frac{\partial z_{l-1}}{\partial x_i} \right)^2 + \sigma'(z_{l-1}) \odot \frac{\partial^2 z_{l-1}}{\partial x_i^2} \right], \quad l = 2, \dots, L,$$

where here $\mathbf{0}$ denotes a 0 vector with the same size as z_1 .

When instead we are interested in cross terms of the form $\frac{\partial z_l}{\partial x_i \partial x_j}$, we can apply the following:

$$\frac{\partial z_1}{\partial x_i \partial x_j} = \mathbf{0}, \quad \frac{\partial z_l}{\partial x_i \partial x_j} = W_l \cdot \left[\sigma''(z_{l-1}) \odot \frac{\partial z_{l-1}}{\partial x_i} \odot \frac{\partial z_{l-1}}{\partial x_j} + \sigma'(z_{l-1}) \odot \frac{\partial^2 z_{l-1}}{\partial x_i \partial x_j} \right], \quad l = 2, \dots, L.$$

With respect to naively using auto-differentiation libraries, computing the derivatives in the forward step is much cheaper; specifically, for fully connected NNs, we found empirically the first to scale quadratically with the output size, while the second scales linearly (Figure 8).

D.2 Batch normalization

The exponential family form used as p_w depends on $\eta_w(\theta)^T f_w(x)$; if you multiply $f_w(x)$ with an invertible matrix A and multiply η with $(A^T)^{-1}$, the product does not change. In order to remove this additional degeneracy, we use a Batch Normalization (BatchNorm) layer (Ioffe and Szegedy, 2015) to normalize the outputs of $\eta_w(\theta)$. Essentially, BatchNorm rescales the different features to have always the same range across different batches. More specifically, BatchNorm performs the following operation on y :

$$\tilde{y} = \frac{y - \mathbb{E}[Y]}{\sqrt{\mathbb{V}[Y] + \epsilon}} * \gamma + b,$$

where ϵ is a small constant used for numerical stability, and γ and b are two (optionally learnable) sets of constants with dimension equal to y (set to 1 and 0 respectively by default). During training, the expectation \mathbb{E} and variance \mathbb{V} are estimated over the batch of training

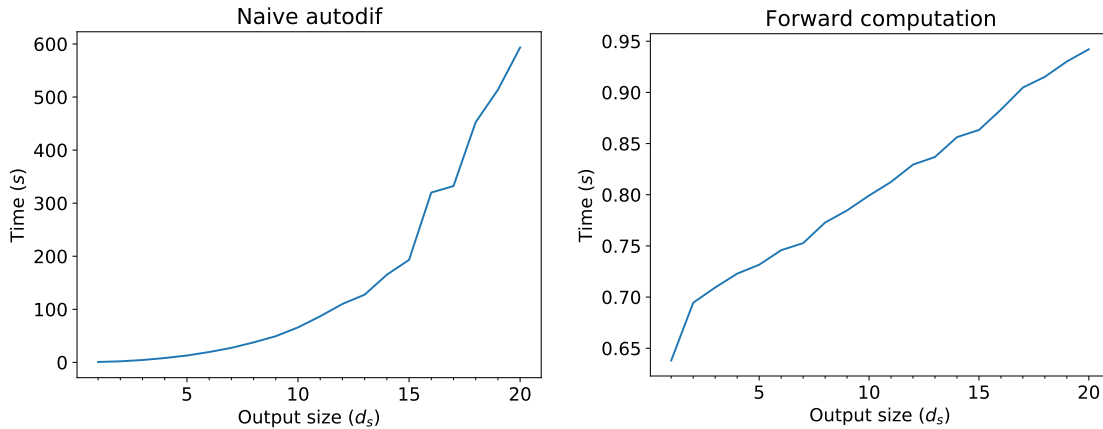


Figure 8: **Computational complexity for second order derivatives of Neural Network outputs** versus size of the output; we compare the forward computation of derivatives with naive autodifferentiation. Here, input size is fixed to 100, and one single batch of 5000 samples is fed to the network. Computations are done on a CPU machine with 8 cores.

samples fed to the Neural Network. In testing mode, BatchNorm rescales the features by using a fixed estimate of \mathbb{E} and \mathbb{V} ; this estimate is obtained as a running mean over the batches; let $s(Y)$ represent either the population expectation or variance. When the t -th batch is fed through the network in training mode, the running mean estimate is updated as follows:

$$\hat{s}_{new}(Y) = (1 - p) \cdot \hat{s}_{old}(Y) + p \cdot s_t(Y),$$

where s_t represent the estimate on the current batch, \hat{s}_{old} and \hat{s}_{new} respectively the old and updated running mean and p is a *momentum* constant which determines how quickly the running mean changes (the smaller it is, the slower the change of the estimate).

For instance, if the training of a Neural Network is quite unstable, the running estimate may not be a correct estimate of \mathbb{E} and \mathbb{V} , so that the test loss across training epochs may be very spiky, until network training stabilized. To solve this issue, you can either increase p (so that the running estimate “forgets” past information faster) or, alternatively, do a forward pass of the training data set (without computing gradients) before evaluating the test loss, so that the running estimate is more precisely estimated.

Across this work, we apply BatchNorm to $y = \eta_w(\theta)$. Moreover, we do not learn the translation parameters γ, b , and rather we fix them to be a vector of 1s and 0s.

D.3 Partially Exchangeable Networks

Partially Exchangeable Networks (PENs) were introduced in Wiqvist et al. (2019) as a Neural Network architecture that satisfies the probabilistic invariance of Markovian models.

Specifically, let consider the case in which $x = (x_1, \dots, x_d)$ comes from a Markovian model of order r , i.e.:

$$\begin{aligned} p(x|\theta) &= p(x_1|\theta)p(x_2|x_1;\theta)p(x_3|x_2, x_1;\theta) \prod_{i=4}^d p(x_i|x_1, \dots, x_{i-1};\theta) \\ &= p(x_1|\theta) \prod_{i=2}^d p(x_i|x_{i-r}, \dots, x_{i-1};\theta). \end{aligned}$$

This definition is an extension of the standard Markovianity assumption (of order 1), which corresponds to $p(x|\theta) = p(x_1|\theta) \prod_{i=2}^d p(x_i|x_{i-1};\theta)$, and it means that each element of x only depends on the last r elements. When $r = 0$, this corresponds to i.i.d. assumption.

When a model is r -Markovian, the probability density of an observation x is invariant to r -**block-switch transformation**, which is defined as follows:

Definition 24 r -block-switch transformation (Wiqvist et al., 2019) Let $x_{i:j}$ and $x_{k:l}$ be two non-overlapping blocks with $x_{i:(i+r)} = x_{k:(k+r)}$ and $x_{(j-r):j} = x_{(l-r):l}$. Then, denoting $b = (i, j, k, l)$, with $j - i \geq r$ and $l - k \geq r$:

$$\begin{array}{ccccccc} x = x_{1:i-1} & x_{i:j} & x_{(j+1):(k-1)} & x_{k:l} & x_{(l+1):M} \\ T_b^{(r)}(x) = x_{1:i-1} & x_{k:l} & x_{(j+1):(k-1)} & x_{i:j} & x_{(l+1):M}. \end{array}$$

Otherwise, if $x_{i:(i+r)} \neq x_{k:(k+r)}$ or $x_{(j-r):j} \neq x_{(l-r):l}$, then $T_b^{(r)}(x) = x$.

For instance, let us consider the case in which the data space is $\mathcal{X} = \{0, 1 \dots 9\}^{16}$, and $r = 2$. An example of the above transformation is the following:

$$\begin{array}{ccccccc} x = 1 & \boxed{7\ 2\ 3\ 6\ 4\ 5\ 8} & 1\ 7 & \boxed{7\ 2\ 9\ 5\ 8} & 1 \\ T_{(2,8,11,15)}^{(2)}(x) = 1 & \boxed{7\ 2\ 9\ 5\ 8} & 1\ 7 & \boxed{7\ 2\ 3\ 6\ 4\ 5\ 8} & 1. \end{array}$$

The authors of Wiqvist et al. (2019) provide a simple Neural Network structure which is invariant to the r -block-switch transformation, motivated by the following theorem:

Theorem 25 ((Wiqvist et al., 2019)) Let $f : \mathcal{X}^M \rightarrow A$ r -block-switch invariant. If \mathcal{X} is countable, $\exists \phi : \mathcal{X}^{r+1} \rightarrow \mathbb{R}$ and $\rho : \mathcal{X}^r \times \mathbb{R} \rightarrow A$ such that:

$$\forall x \in \mathcal{X}^M, f(x) = \rho \left(x_{1:r}, \sum_{i=1}^{M-r} \phi(x_{i:(i+r)}) \right). \quad (16)$$

In practice, ϕ and ρ are two independent Neural Networks (which we take to be fully connected in our case), giving rise to a PEN of order r .

In Wiqvist et al. (2019), the authors show that the posterior mean of a Markovian variable of order r needs to be invariant to the r -block-switch transformation. Therefore, this motivates using a PEN for learning a summary statistics as in the approach by Fearnhead and Prangle (2012). Here, we use PEN for parametrizing the statistics in the approximating exponential family as well; this choice implies that the approximating family has the same Markovianity property as the true distribution, as we discuss in the following.

D.3.1 RESULTS FOR THE EXPONENTIAL FAMILY

A deeper connection between r -Markovian probability models and r -block-switch transformation exists. We can in fact state the following result:

Lemma 26 *A probability model $p(x|\theta)$ is r -Markovian \iff the function $x \mapsto p(x|\theta)$ is r -block-switch invariant, i.e. $p(x|\theta) = p(T_b^{(r)}(x)|\theta) \forall T_b^{(r)}$.*

Proof The forward direction is straightforward and can be seen by considering the decomposition of an r -Markovian model.

The converse direction can be shown by contradiction; assume in fact that $x \mapsto p(x|\theta)$ is r -block-switch invariant but not Markovian. As it is not Markovian, $\exists x = (x_i, x_2, \dots, x_n)$ for which $x_{i:(i+r)} = x_{k:(k+r)}$ and $x_{(j-r):j} = x_{(l-r):l}$ such that, defining $b = (i, j, k, l)$, $p(x|\theta) \neq p(T_b^{(r)}(x)|\theta)$. This is however in contradiction with r -block-switch invariance, which leads to our result. \blacksquare

In the case where the model we consider has a sufficient statistic, we can write $p(x|\theta) = h(x)g(t(x)|\theta)$. We get therefore the following corollary, which can be seen by applying the above result:

Corollary 27 *Consider a distribution $p(x|\theta) = h(x)g(t(x)|\theta)$; if the function $h(x)$ and $t(x)$ are r -block-switch invariant, then $p(x|\theta)$ is r -Markovian.*

Without any further assumptions, this result is not enough to say that the sufficient statistics $t(x)$ for a Markovian model is r -block-switch invariant; in fact, the choice $t(x) = x$ always constitutes a sufficient statistic; moreover, in the decomposition $p(x|\theta) = h(x)g(t(x)|\theta)$, it may be that the function $t(x)$ is not r -block-switch invariant but $g(t(x)|\theta)$ is, or otherwise that the product $h(x)g(t(x)|\theta)$ is r -block-switch invariant even if the individual terms are not. We can however get the following result:

Lemma 28 *Consider a r -Markovian distribution $p(x|\theta) = h(x)g(t(x)|\theta)$; if $x \mapsto t(x)$ is not an injection mapping, then $x \mapsto h(x)$ is r -block-switch invariant.*

Proof If $x \mapsto t(x)$ is not an injection, $\exists x, x'$ such that $t(x) = t(x')$. If the density is not degenerate, moreover, $\exists \theta : p(x|\theta), p(x'|\theta) > 0$. Therefore, we consider the following ratio:

$$\frac{p(x|\theta)}{p(x'|\theta)} = \frac{h(x)}{h(x')} \cdot \frac{g(t(x)|\theta)}{g(t(x')|\theta)} = \frac{h(x)}{h(x')}.$$

Now, the left hand side is r -block-switch invariant with respect to both x and x' independently, implying that $h(x)$ is as well. \blacksquare

We remark that it does not seem possible in general to say anything about $t(x)$, as in fact it may be that the function $t(x)$ is not r -block-switch invariant but $g(t(x)|\theta)$ is. In the specific case of an exponential family, however, a more specific result can be obtained:

Lemma 29 *Consider an exponential family distribution $p(x|\theta) = h(x) \exp(\eta(\theta)^T f(x))/Z(\theta)$ which is r -Markovian distribution; if $x \mapsto f(x)$ is not an injection mapping, then $x \mapsto f(x)$ is r -block-switch invariant.*

Proof Without loss of generality, we consider the case in which all elements of $\eta(\theta)$ are not constant with respect to θ ; if this is not the case, in fact, you can redefine the exponential family by incorporating the elements of $f(x)$ corresponding to the constant ones of η in the $h(x)$ factor.

Now, $h(x)$ is r -block-switch invariant thanks to Lemma 28. Consider now the following decomposition:

$$\log p(x|\theta) = \log h(x) - \log Z(\theta) + \sum_i f_i(x) \eta_i(\theta);$$

as that needs to be r -block-switch invariant for any θ , this can happen only if each of the $f_i(x)$ elements are r -block-switch invariant. \blacksquare

Overall, these results imply that an exponential family in which f is parametrized with a PEN of order r is r -Markovian. Moreover, provided that f is not an injection mapping, all r -Markovian exponential families have f which satisfy the r -block-switch invariant property, which is imposed by using a PEN network of order r .

D.3.2 FORWARD COMPUTATION OF DERIVATIVES FOR PENS

We give here the derivation for the forward computation of derivatives with PENS. If we pick here ϕ and ρ to be fully connected Neural Networks, we can moreover apply the forward computation of derivatives for them and we are able to compute the derivatives for PENS at a much lower cost with respect to using automatic differentiation libraries.

In Eq. (16), let us denote for brevity $z = \sum_{i=1}^{M-r} \phi(x_{i:(i+r)})$. We are interested now in computing the derivative:

$$\frac{\partial f}{\partial x_j} = \frac{\partial}{\partial x_j} \rho(x_{1:r}, z),$$

where note that z depends in general on x_i . Therefore, in computing the above, we need to compute the derivative with respect to both arguments; let us denote by $\frac{\partial'}{\partial x_j}$ the derivative with respect to the first argument. Then, we have:

$$\frac{\partial f}{\partial x_j} = \frac{\partial'}{\partial x_j} \rho(x_{1:r}, z) \cdot \mathbb{1}[j \leq r] + \frac{\partial}{\partial z} \rho(x_{1:r}, z) \cdot \frac{\partial z}{\partial x_j},$$

where the second term is (note that z and ρ are multivariate, so that $\frac{\partial'}{\partial x_j} \rho(x_{1:r}, z)$ is a Jacobian matrix):

$$\frac{\partial z}{\partial x_j} = \frac{\partial}{\partial x_j} \sum_{i=1}^{M-r} \phi(x_{i:(i+r)}) = \sum_{\substack{i=j-r \\ i \geq 1}}^j \frac{\partial}{\partial x_j} \phi(x_{i:(i+r)}),$$

where all other terms of the sum disappear as they do not contain x_j .

Now, we are interested in obtaining the second derivative terms:

$$\begin{aligned} \frac{\partial^2 f}{\partial x_j^2} &= \frac{\partial}{\partial x_j} \frac{\partial f}{\partial x_j} = \frac{\partial}{\partial x_j} \left[\frac{\partial'}{\partial x_j} \rho(x_{1:r}, z) \cdot \mathbb{1}[j \leq r] + \sum_k \frac{\partial}{\partial z_k} \rho(x_{1:r}, z) \frac{\partial z_k}{\partial x_j} \right] \\ &= \frac{\partial'^2}{\partial x_j^2} \rho(x_{1:r}, z) \cdot \mathbb{1}[j \leq r] + 2 \sum_k \frac{\partial'}{\partial x_j} \frac{\partial}{\partial z_k} \rho(x_{1:r}, z) \cdot \frac{\partial z_k}{\partial x_j} \cdot \mathbb{1}[j \leq r] \\ &\quad + \sum_{k, k'} \frac{\partial^2}{\partial z_k \partial z_{k'}} \rho(x_{1:r}, z) \frac{\partial z_k}{\partial x_j} \frac{\partial z_{k'}}{\partial x_j} + \sum_k \frac{\partial}{\partial z_k} \rho(x_{1:r}, z) \frac{\partial^2 z_k}{\partial x_j^2}; \end{aligned}$$

in the above expression, \sum_k runs over the elements of z and $\frac{\partial'^2}{\partial x_j^2}$ denotes second derivative with respect to the first element. Note that all terms appearing in the above formulas contain first and second derivatives of the Neural Networks ϕ and ρ with respect to one single input, except for the terms highlighted in red. In order to compute that, obtaining the full hessian matrix of ρ is required. We remark that the latter can be very large in case the input dimension is large, therefore leading to memory overflow issues.

D.4 Exchange MCMC

For convenience, we describe here the ExchangeMCMC algorithm by Murray et al. (2012). We consider the task of sampling from a posterior distribution $\pi(\theta|x)$. We can evaluate an unnormalized version of the likelihood $\tilde{p}(x|\theta)$, and we denote the normalized version as $p(x|\theta) = \tilde{p}(x|\theta)/Z(\theta)$, $Z(\theta)$ being an intractable normalizing constant. We want to build an MCMC chain by using a proposal distribution $q(\cdot|\theta; x)$ (which optionally depends on the considered x as well). Usually, the standard Metropolis acceptance threshold for a proposal θ' is defined as:

$$\alpha = \frac{\pi(\theta'|x)q(\theta|\theta'; x)}{\pi(\theta|x)q(\theta'|\theta; x)} = \frac{\tilde{p}(x|\theta')q(\theta|\theta'; x)\pi(\theta')}{\tilde{p}(x|\theta)q(\theta'|\theta; x)\pi(\theta)} \cdot \frac{Z(\theta)}{Z(\theta')},$$

where the last factor cannot be evaluated, as we do not have access to the normalizing constant.

The ExchangeMCMC algorithm proposed by Murray et al. (2012) bypasses this issue by drawing an auxiliary observation $x' \sim p(\cdot|\theta')$ and defining the acceptance probability as:

$$\alpha = \frac{p(x|\theta')q(\theta|\theta'; x)\pi(\theta')}{p(x|\theta)q(\theta'|\theta; x)\pi(\theta)} \cdot \frac{p(x'|\theta)}{p(x'|\theta')} = \frac{\tilde{p}(x|\theta')\tilde{p}(x'|\theta)q(\theta|\theta'; x)\pi(\theta')}{\tilde{p}(x|\theta)\tilde{p}(x'|\theta')q(\theta'|\theta; x)\pi(\theta)} \cdot \frac{Z(\theta)}{Z(\theta')} \cdot \frac{Z(\theta')}{Z(\theta)}. \quad (17)$$

Here, all the normalizing constants cancel out, so that the acceptance threshold can be evaluated explicitly, at the expense of drawing a simulation from the likelihood for each MCMC step. Murray et al. (2012) showed that an MCMC chain using the above acceptance rate targets the correct posterior $\pi(\theta|x)$. The resulting algorithm is given in Algorithm 1:

Bridging. When considering more closely the acceptance rate in Eq. (17), it can be seen that it depends on two ratios: $\frac{p(x|\theta')}{p(x|\theta)}$ represents how well the proposed parameter value explains the observation with respect to the previous parameter value, while instead $\frac{p(x'|\theta)}{p(x'|\theta')}$ measures how well the auxiliary variable (generated using θ') can be explained with

Algorithm 1 Original exchangeMCMC algorithm (Murray et al., 2012).

Require: Initial θ , number of iterations T , proposal distribution q , observation x .

- 1: **for** $i = 1$ **to** T **do**
 - 2: Propose $\theta' \sim q(\theta'|\theta; x)$
 - 3: Generate auxiliary observation $x' \sim p(\cdot|\theta')$
 - 4: Compute acceptance threshold α as in Eq. (17)
 - 5: With probability α , set $\theta \leftarrow \theta'$
 - 6: **end for**
-

parameter θ . Therefore, even if the former is large and θ would be a suitable parameter value, α can still be small if the auxiliary random variable is not explained well by the previous parameter value. This can lead to slow mixing of the chain; to improve on this, Murray et al. (2012) proposed to sample a set of auxiliary variables $(x'_0, x'_1, \dots, x'_K)$ from intermediate distributions in the following way¹⁰: consider a set of densities

$$\tilde{p}_k(x|\theta, \theta') = \tilde{p}(x|\theta')^{\beta_k} \tilde{p}(x|\theta)^{1-\beta_k}, \quad \beta_k = \frac{K - k + 1}{K + 1};$$

x'_0 is generated from $p(\cdot|\theta')$ as before, and then each x'_k is generated from $R(\cdot|x'_{k-1}; \theta, \theta')$, which denotes a symmetric Metropolis-Hastings transition kernel starting from x'_{k-1} with stationary density $\tilde{p}_k(\cdot|\theta, \theta')$. Then, the acceptance rate is modified as follows:

$$\alpha = \frac{\tilde{p}(x|\theta')q(\theta|\theta'; x)\pi(\theta')}{\tilde{p}(x|\theta)q(\theta'|\theta; x)\pi(\theta)} \cdot \prod_{k=0}^K \frac{\tilde{p}_{k+1}(x'_k|\theta, \theta')}{\tilde{p}_k(x'_k|\theta, \theta')}. \quad (18)$$

The overall algorithm is given in Algorithm 2. Note that $K = 0$ recovers the original ExchangeMCMC. This procedure generally improves the acceptance rate as it basically introduces a sequence of intermediate updates to the auxiliary data which by smoothening out the difference between the two distributions.

Algorithm 2 ExchangeMCMC algorithm with bridging (Murray et al., 2012).

Require: Initial θ , number of iterations T , proposal distribution q , number of bridging steps K , observation x .

- 1: **for** $i = 1$ **to** T **do**
 - 2: Propose $\theta' \sim q(\theta'|\theta; x)$
 - 3: Generate auxiliary observation $x'_0 \sim p(\cdot|\theta')$
 - 4: **for** $k = 1$ **to** K **do** ▷ Bridging steps
 - 5: Generate $x'_k \sim R(\cdot|x'_{k-1}; \theta, \theta')$
 - 6: **end for**
 - 7: Compute acceptance threshold α as in Eq. (18)
 - 8: With probability α , set $\theta \leftarrow \theta'$
 - 9: **end for**
-

10. Differently from the rest of the work, here subscripts do not denote vector components, but rather different auxiliary variables.

ExchangeMCMC without perfect simulations. If, as in the setup considered across this work, we are not able to sample from $p(\cdot|\theta')$ as it is required in the ExchangeMCMC algorithm (line 3 in Alg. 1), Murray et al. (2012) suggested to run T_{in} steps of an MCMC chain on x targeting $p(\cdot|\theta')$ at each step of ExchangeMCMC; if T_{in} is large enough, the last sample can be considered as (approximately) drawn from $p(\cdot|\theta')$ itself and used in place of the unavailable perfect simulation. In practice, however, this only leads to an approximate ExchangeMCMC algorithm, as at each iteration of the inner chain a finite T_{in} is used, so that the inner chain would not perfectly converge to its target; for this reason, even an infinitely long outer chain would not target the right posterior for any finite T_{in} . Nonetheless, this approach was shown empirically to work satisfactorily in Caimo and Friel (2011); Everitt (2012); Liang (2010). Some theoretical guarantees (albeit under strong conditions), are given in Appendix B by Everitt (2012), which bounds the total variation distance between target of approximate ExchangeMCMC with finite T_{in} and the target of the exact one, and shows that they become equal when $T_{in} \rightarrow \infty$.

In Liang (2010), they argue that starting the inner chain from the observation value improves convergence; we adapt this approach in our implementation (Algorithm 3).

Algorithm 3 ExchangeMCMC algorithm (Murray et al., 2012) with inner MCMC.

Require: Initial θ , number of iterations T and T_{in} , proposal distributions q and q_x , observation x .

```

1: for  $i = 1$  to  $T$  do                                     ▷ Outer chain
2:   Propose  $\theta' \sim q(\theta'|\theta; x)$ 
3:   Set  $x' = x$                                              ▷ Start inner chain from the observation
4:   for  $j = 1$  to  $T_{in}$  do                                   ▷ Inner chain
5:     Propose  $x'' \sim q_x(x''|x')$ 
6:     With probability  $\frac{p_w(x''|\theta')q_x(x'|x'')}{p_w(x'|\theta')q_x(x''|x')}$ , set  $x' = x''$ 
7:   end for
8:   Compute acceptance threshold  $\alpha$  as in Eq. (17)     ▷ This uses last point of inner
   MCMC
9:   With probability  $\alpha$ , set  $\theta \leftarrow \theta'$ 
10: end for

```

Note that it is still possible to run bridging steps after the inner MCMC to sample from $p(\cdot|\theta')$; the algorithm combining bridging and inner MCMC, which is used across this work, is given in Algorithm 4).

Related algorithms. Algorithms for sampling from doubly-intractable targets which are suitable for parallel computing exist, for instance Caimo and Friel (2011) propose a parallel-chain MCMC algorithm, while Everitt et al. (2017) build instead an SMC-type algorithm which is also capable of recycling information from past simulations. However, in this work we stick to using ExchangeMCMC, which turned out to be relatively cheap to use and easy to implement.

Finally, we remark that Liang et al. (2016) proposed an algorithm which is inspired from ExchangeMCMC and, in the case of impossible perfect sampling, still targets the right invariant distribution. It works by considering a set of parallel chains targeting $p(\cdot|\theta^{(i)})$ for

Algorithm 4 ExchangeMCMC algorithm (Murray et al., 2012) with inner MCMC and bridging.

Require: Initial θ , number of iterations T and T_{in} , number of bridging steps K , proposal distributions q and q_x , observation x .

- 1: **for** $i = 1$ **to** T **do** ▷ Outer chain
- 2: Propose $\theta' \sim q(\theta'|\theta; x)$
- 3: Set $x' = x$ ▷ Start inner chain from the observation
- 4: **for** $j = 1$ **to** T_{in} **do** ▷ Inner chain
- 5: Propose $x'' \sim q_x(x''|x')$
- 6: With probability $\frac{p_w(x''|\theta')q_x(x'|x'')}{p_w(x'|\theta')q_x(x''|x')}$, set $x' = x''$
- 7: **end for**
- 8: Set $x'_0 = x'$
- 9: **for** $k = 1$ **to** K **do** ▷ Bridging steps
- 10: Generate $x'_k \sim R(\cdot|x'_{k-1}; \theta, \theta')$
- 11: **end for**
- 12: Compute acceptance threshold α as in Eq. (18)
- 13: With probability α , set $\theta \leftarrow \theta'$
- 14: **end for**

a fixed set of $\{\theta^{(i)}\}$ and iteratively updating those and the main chain over θ . The algorithm relies on some assumptions which are probably satisfied in practice (as discussed in Park and Haran, 2018). It also requires some hand-tuning and needs to keep in memory a large amount of data, which may hinder its applicability. For this reason, we do not investigate using that here.

D.4.1 IMPLEMENTATION DETAILS

Acceptance rate tuning. In our implementation of ExchangeMCMC, we discarded some burn-in steps to make sure the chain forgets its initial state. During burn-in, moreover, after each window of 100 outer steps, we tuned the proposal sizes considering the acceptance rate in the window, and increasing (respectively decreasing) if that is too large (small) with respect to a chosen interval (see below). Notice that we applied this strategy independently to the proposal size for the outer chain, the inner chain and the bridging step, when the latter is used. When tuning the proposal size for the inner chain or bridging steps, we considered the overall acceptance rate over each window of 100 outer steps. We remark how this was only done during burn-in, so that the convergence properties of the chain were not affected.

For inner and outer steps as well as bridging, we found that a target acceptance rate in the interval $[0.2, 0.5]$ lead to good performance; this is consistent with the recommended range for Metropolis-Hastings MCMC (Roberts et al., 1997).

MCMC on bounded space. When the inner (or outer) MCMC chain is run on a bounded domain, we apply the transformations discussed in Appendix C.8 to map it to an unbounded domain, and therefore run the MCMC on that space. Notice that the Jaco-

bian factor arising from the transformations has therefore to be taken into account when computing the acceptance rate.

E. Details on Neural Networks training

For all experiments, we used the `Pytorch` library (Paszke et al., 2019) to train NNs. In Tables 3, 4, 5, 6 and 7 we report the Neural Network architectures used in the different experiments. $\text{FC}(n, m)$ denotes a fully connected layer with n inputs and m outputs. For the time-series and Lorenz96 experiments, ϕ_w and ρ_w represent the two Neural Networks used to build the PEN network f_w as described in Eq. (16), and similarly ϕ_β and ρ_β represent the ones used in building s_β . Finally, $\text{BN}(p)$ represents a BatchNorm layer with momentum p (as described in Appendix D.2), with γ, b , fixed respectively to be vectors of 1s and 0s. We remark that the momentum value does not impact on the training of the network, but it modifies the evaluation of the test loss, which we use for early stopping, as discussed below.

<i>Network</i>	f_w	η_w	s_β
<i>Structure</i>	FC(10,30)	FC(2,15)	FC(10,30)
	FC(30,50)	FC(15,30)	FC(30,50)
	FC(50,50)	FC(30,30)	FC(50,50)
	FC(50,20)	FC(30,15)	FC(50,20)
	FC(20,3)	FC(15,2)	FC(20,2)
		BN(0.9)	

Table 3: **Architectures used for the exponential family models** (Gaussian, Gamma and Beta).

<i>Network</i>	f_w		η_w	s_β	
	ϕ_w	ρ_w		ϕ_β	ρ_β
<i>Structure</i>	FC(3,50)	FC(22,50)	FC(2,15)	FC(3,50)	FC(22,50)
	FC(50,50)	FC(50,50)	FC(15,30)	FC(50,50)	FC(50,50)
	FC(50,50)	FC(50,3)	FC(30,30)	FC(50,30)	FC(50,2)
	FC(30,20)		FC(30,15)	FC(30,20)	
			FC(15,2)		
			BN(0.9)		

Table 4: **Architectures used for the AR(2) model.**

In all experiments, stochastic gradient descent with a batch size of 1000 samples is used with Adam optimizer (Kingma and Ba, 2015), whose parameters are left to the default values as implemented in `Pytorch`. Finally, we evaluate the test loss on a test set with same size as the training set at intervals of T_{check} epochs and early-stop training if the test loss increased with respect to the last evaluation. In order to have a better running estimate of the quantities of interest for the BatchNorm layer, before each test epoch we perform a forward pass of the whole training data set, without computing gradients, as discussed in

<i>Network</i>	f_w		η_w	s_β	
	ϕ_w	ρ_w		ϕ_β	ρ_β
<i>Structure</i>	FC(11,50)	FC(30,50)	FC(2,15)	FC(11,50)	FC(30,50)
	FC(50,50)	FC(50,50)	FC(15,30)	FC(50,50)	FC(50,50)
	FC(50,30)	FC(50,3)	FC(30,30)	FC(50,30)	FC(50,2)
	FC(30,20)		FC(30,15)	FC(30,20)	
		FC(15,2)			
		BN(0.9)			

Table 5: Architectures used for the MA(2) model.

<i>Network</i>	f_w		η_w	s_β	
	ϕ_w	ρ_w		ϕ_β	ρ_β
<i>Structure</i>	FC(16,50)	FC(28,40)	FC(4,30)	FC(16,50)	FC(28,40)
	FC(50,100)	FC(40,90)	FC(30,50)	FC(50,100)	FC(40,90)
	FC(100,50)	FC(90,35)	FC(50,50)	FC(100,50)	FC(90,35)
	FC(50,20)	FC(35,5)	FC(50,30)	FC(50,20)	FC(35,4)
		FC(30,4)			
		BN(0.9)			

Table 6: Architectures used for the Lorenz96 model in the small setup.

<i>Network</i>	f_w		η_w	s_β	
	ϕ_w	ρ_w		ϕ_β	ρ_β
<i>Structure</i>	FC(80,120)	FC(60,80)	FC(4,30)	FC(80,120)	FC(60,80)
	FC(120,160)	FC(80,100)	FC(30,50)	FC(120,160)	FC(80,100)
	FC(160,120)	FC(100,80)	FC(50,50)	FC(160,120)	FC(100,80)
	FC(120,20)	FC(80,5)	FC(50,30)	FC(120,20)	FC(80,4)
		FC(30,4)			
		BN(0.9)			

Table 7: Architectures used for the Lorenz96 model in the large setup.

Appendix D.2. We do not perform early stopping before epoch T_{start} . The nets are trained for a maximum of T training epochs with N_{train} training samples. In some experiments, we used an exponential learning rate scheduler, which decreases progressively the learning rate by multiplying it by a factor $\zeta < 1$ at each epoch. We fixed $\zeta = 0.99$. The values of the parameters are reported in Table 8 for all the different models and setups, together with the values of the learning rates (lr) used and whether the scheduler was used or not (Sch).

We remark that, in order to make the comparison fair, the learning rates for the FP experiments were chosen by cross validation with several learning rates choices. Similarly, we hand-picked the best learning rate values for training the Neural Networks with SM and SSM, even if in that case it was not possible, due to computational constraints, to perform a full search on a large set of values of learning rates for all experiments. In fact, the computational cost of training the exponential family approximation with SM or SSM is larger than the cost of learning the summary statistics with the FP approach. Moreover, in this scenario we have two independent learning rates values to tune, as we train two networks simultaneously. For this reason, we did not spend too much time trying different lr values for models for which we already got satisfactory results.

<i>Model</i>	<i>Setup</i>	<i>Learning rates</i>	N_{train}	T	T_{start}	T_{check}	<i>Sch</i>
Gaussian	SM	$\text{lr}(f_w) = 0.0003, \text{lr}(\eta_w) = 0.003$	10^4	500	150	10	Yes
	SSM	$\text{lr}(f_w) = 0.001, \text{lr}(\eta_w) = 0.001$	10^4	500	200	10	Yes
	FP	$\text{lr}(s_\beta) = 0.01$	10^4	1000	300	25	No
Gamma	SM	$\text{lr}(f_w) = 0.001, \text{lr}(\eta_w) = 0.001$	10^4	500	200	10	Yes
	SSM	$\text{lr}(f_w) = 0.001, \text{lr}(\eta_w) = 0.001$	10^4	500	200	10	Yes
	FP	$\text{lr}(s_\beta) = 0.001$	10^4	1000	300	25	Yes
Beta	SM	$\text{lr}(f_w) = 0.001, \text{lr}(\eta_w) = 0.001$	10^4	500	200	10	Yes
	SSM	$\text{lr}(f_w) = 0.001, \text{lr}(\eta_w) = 0.001$	10^4	500	200	10	Yes
	FP	$\text{lr}(s_\beta) = 0.01$	10^4	1000	250	50	Yes
AR(2)	SM	$\text{lr}(f_w) = 0.001, \text{lr}(\eta_w) = 0.001$	10^4	500	100	25	Yes
	SSM	$\text{lr}(f_w) = 0.001, \text{lr}(\eta_w) = 0.001$	10^4	500	100	25	Yes
	FP	$\text{lr}(s_\beta) = 0.001$	10^4	1000	500	25	Yes
MA(2)	SM	$\text{lr}(f_w) = 0.001, \text{lr}(\eta_w) = 0.001$	10^4	500	100	25	Yes
	SSM	$\text{lr}(f_w) = 0.001, \text{lr}(\eta_w) = 0.001$	10^4	500	100	25	Yes
	FP	$\text{lr}(s_\beta) = 0.001$	10^4	1000	500	25	Yes
Lorenz96 Small	SSM	$\text{lr}(f_w) = 0.001, \text{lr}(\eta_w) = 0.001$	10^4	1000	500	50	Yes
	FP	$\text{lr}(s_\beta) = 0.001$	10^4	1000	200	25	Yes
Lorenz96 Large	SSM	$\text{lr}(f_w) = 0.001, \text{lr}(\eta_w) = 0.001$	10^4	1000	500	50	Yes
	FP	$\text{lr}(s_\beta) = 0.001$	10^4	1000	200	25	Yes

Table 8: **Hyperparameter values for NN training:** “FP” denotes the least squares regression by Fearnhead and Prangle (2012); Jiang et al. (2017) used for the ABC-FP experiment, while “SM” and “SSM” denote respectively exponential family trained with Score Matching and Sliced Score Matching.

F. Additional experimental results

F.1 Exponential family models

F.1.1 MEAN CORRELATION COEFFICIENTS FOR NEURAL NETWORKS TRAINED WITH SSM

We report in Table 9 the weak and strong Mean Correlation Coefficient (MCC, Appendix B.1.1) for Neural Networks trained with SSM, for the exponential family models. MCC is a metric in $[0, 1]$, with 1 denoting perfect recovery up to a linear transformation (weak) or permutation (strong). As it can be seen in Table 9, our method leads to values quite close to 1, particularly for the weak MCC, implying that our method is able to recover the embeddings up to a linear transformation, as expected.

<i>Model</i>	<i>MCC weak in</i>	<i>MCC weak out</i>	<i>MCC strong in</i>	<i>MCC strong out</i>
Beta (statistics)	0.990	0.986	0.982	0.979
Beta (nat. par.)	0.987	0.989	0.983	0.985
Gamma (statistics)	0.939	0.928	0.723	0.709
Gamma (nat. par.)	0.977	0.977	0.792	0.794
Gaussian (statistics)	0.874	0.844	0.623	0.638
Gaussian (nat. par.)	0.861	0.862	0.581	0.543

Table 9: **MCC for exponential family models between exact embeddings and those learned with SSM.** We show weak and strong MCC values; MCC is between 0 and 1 and measures how well an embedding is recovered up to permutation and rescaling of its components (strong) or linear transformation (weak); the larger, the better. “in” denotes MCC on training data used to find the best transformation, while “out” denote MCC on test data. We used 500 samples in both training and test data sets.!

F.1.2 LEARNED AND EXACT EMBEDDINGS FOR THE EXPONENTIAL FAMILY MODELS

We compare here the exact and learned sufficient statistics and natural parameters of the exponential family models; precisely, we draw samples $(x^{(j)}, \theta^{(j)})$ and then plot the learned statistics $f_w(x^{(j)})$ versus the exact one, and similarly for the natural parameters. Figure 9 reports the results for Neural Networks trained with SM, while Figure 10 reports the results for Neural Networks trained with SSM.

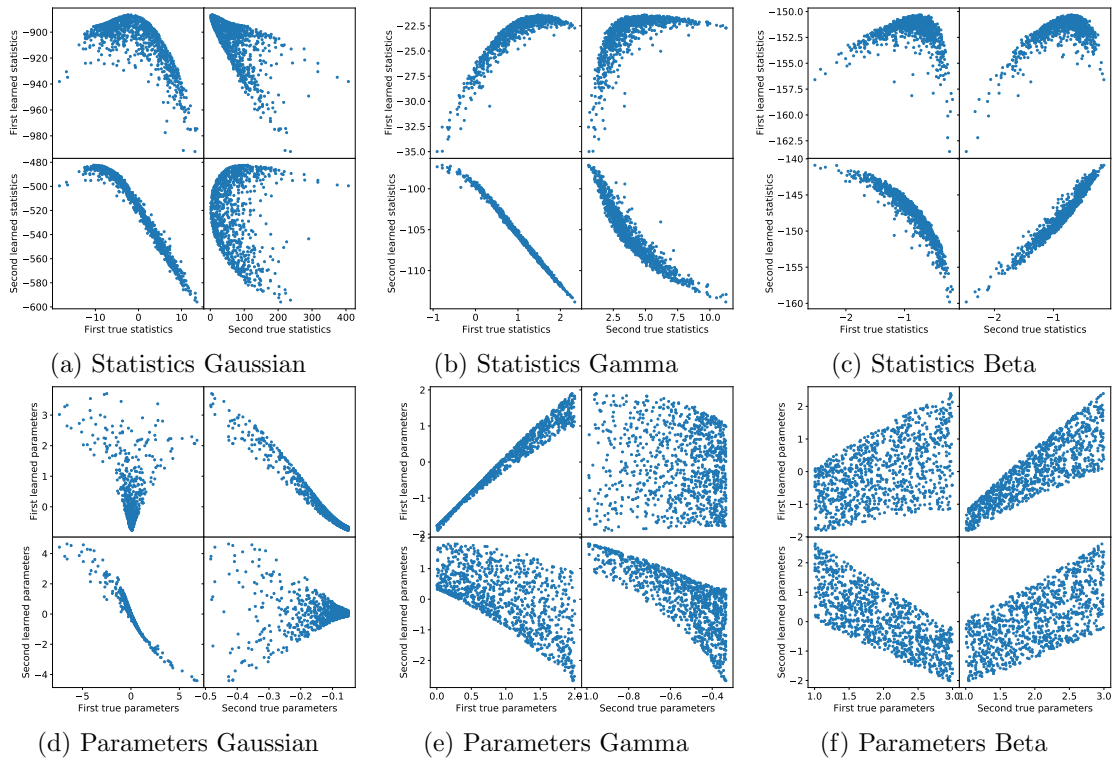


Figure 9: **Learned and exact embeddings for exponential family models obtained with SM.** Each point represents a different x (for the statistics) or θ (for the natural parameters); 1000 of each were used here.

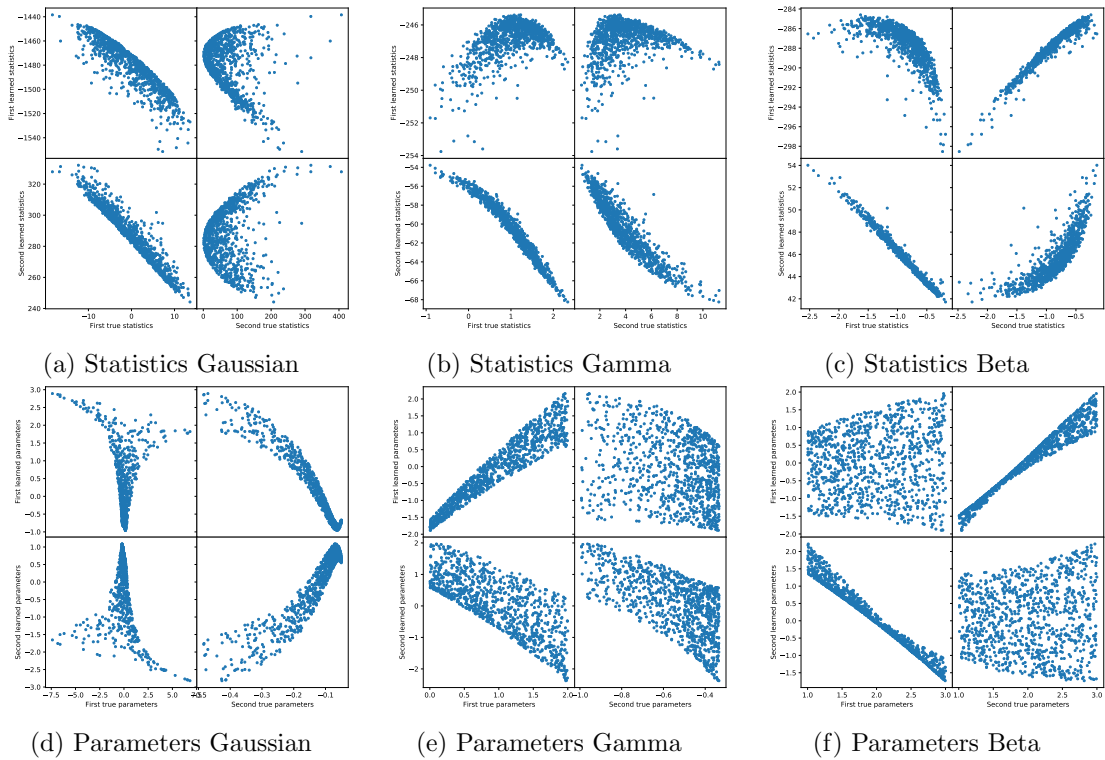


Figure 10: **Learned and exact embeddings for exponential family models obtained with SSM.** Each point represents a different x (for the statistics) or θ (for the natural parameters); 1000 of each were used here.

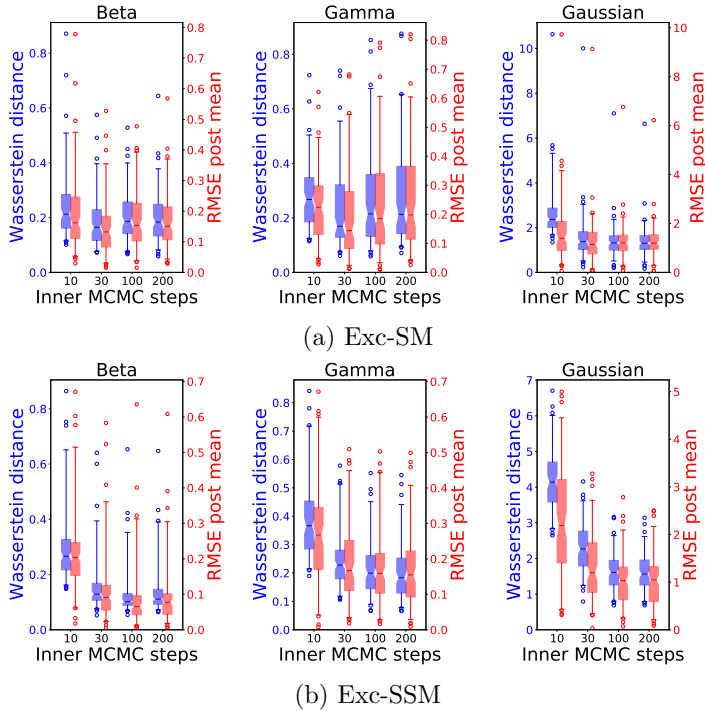


Figure 11: **Performance of Exc-SM with different number of inner MCMC steps, for exponential family models.** Wasserstein distance from the exact posterior and RMSE between exact and approximate posterior means are reported for 100 observations using boxplots. Boxes span from 1st to 3rd quartile, whiskers span 95% probability density region and horizontal line denotes median. The numerical values are not comparable across examples, as they depend on the range of parameters.

F.1.3 PERFORMANCE OF EXC-SM AND EXC-SSM

We study here the performance of Exc-SM and Exc-SSM with different numbers of inner steps in the ExchangeMCMC algorithm (Algorithm 4) for the Exponential family models. Specifically, we run the inference with 10, 30, 100 and 200 inner MCMC steps, and we evaluate the performance in these 4 cases (Figure 11); considering the different models, we observe that the performance with 30 steps is almost equivalent to the one with 100 and 200, albeit being faster (see the computational time in Table 10). In the main text, we therefore present results using 30 inner MCMC steps.

<i>Inner MCMC steps</i>	10	30	100	200
<i>Time (minutes)</i>	≈ 2	≈ 4	≈ 16	≈ 28

Table 10: **Approximate computational time of Exc-SM with different number of inner MCMC steps for the exponential family models.** These values were obtained by running on a single core.

<i>Inner MCMC steps</i>	10	30	100	200
<i>Time (minutes)</i>	≈ 2	≈ 5	≈ 17	≈ 29

Table 11: **Approximate computational time of Exc-SM with different number of inner MCMC steps, for the AR(2) and MA(2) models.** These values were obtained by running on a single core.

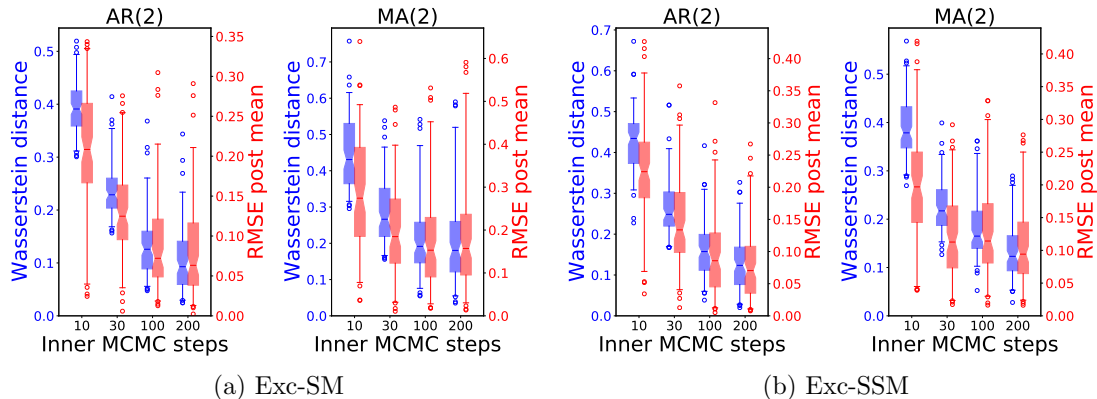


Figure 12: **Performance of Exc-SM with different number of inner MCMC steps, for AR(2) and MA(2) models.** Wasserstein distance from the exact posterior and RMSE between exact and approximate posterior means are reported for 100 observations using boxplots. Boxes span from 1st to 3rd quartile, whiskers span 95% probability density region and horizontal line denotes median.

F.2 AR(2) and MA(2) models

We study here the performance of Exc-SM and Exc-SSM with different numbers of inner steps in the ExchangeMCMC algorithm (Algorithm 4) for the AR(2) and MA(2) models. Specifically, we run the inference with 10, 30, 100 and 200 inner MCMC steps, and we evaluate the performance in these 4 cases (Figure 12); considering the different models, we observe that the performance with 30 steps is almost equivalent to the one with 100 and 200, albeit being faster (see the computational time in Table 11). In the main text, we therefore present results using 30 inner MCMC steps.

F.3 Simulation cost to reach equivalent performance as Exc-SM

Above, we have showed that Exc-SM and Exc-SSM are competitive with the other approaches, even if they require no additional model simulations. Here, we quantify how many model simulations are needed for the other techniques to reach the same performance of Exc-SM in the exponential family and time-series models. The same analysis could be done with respect to Exc-SSM but, as its performance is generally close to the one achieved by Exc-SM, we avoid repeating it.

We compute therefore the performance of ABC-SM, ABC-SSM, ABC-FP, PMC-SL, PMC-RE at each iteration for all 100 observations and find when their median performance

(as quantifies by the Wasserstein distance with respect to the true posterior) becomes comparable or better than the median one achieved by Exc-SM. The number of required simulations are reported in Table 12. Notice that some techniques are not able to outperform Exc-SM for some models; we highlight that by a dash in the Table. SL and RE reach similar performance to Exc-SM with one single iteration, when they are able to do so; in fact, we found empirically that the performance of them does not significantly improve with iterations. Still, we remark that one single iteration of SL and RE requires a very large number of model simulations.

In Table 12, we also give the number of simulations required for the preliminary NN training in the methods which use one; further, we compute the overall cost of inference in terms of model simulations (taking into account both NN training and inference) for different number of observations; we remark that, as discussed previously, our method requires no additional model simulations to perform inference after the NNs have been trained. From Table 12, it can be seen that, for all models, ABC-SM, ABC-FP, PMC-RE and PMC-SL require a number of simulations larger than the one needed to train the NNs in Exc-SM to reach the performance achieved by Exc-SM, which makes the latter an interesting option for models in which simulations are very expensive.

In Figures 13 and 14 we represent the performance attained by ABC-FP, ABC-SM, ABC-SSM, PMC-SL and PMC-RE at each iteration of the iterative algorithm. On the horizontal axis of all plots, we report the number of model simulations corresponding to the iteration of the algorithm.

F.4 Validation with Scoring Rules for Lorenz96 model

Recall that the Lorenz96 model (Section 5.3) is a multivariate time-series model. Therefore, we use the Scoring Rules (on which more details are given in Sec. 1.2.2 in the introduction of the present thesis) to evaluate the predictive performance at each timestep separately.

First, let us recall the definition of the posterior predictive density:

$$p(x|x^0) = \int p(x|\theta)\pi(\theta|x^0),$$

whose corresponding posterior predictive distribution we denote as $P(\cdot|x^0)$. In the above expression, $\pi(\theta|x^0)$ may represent the posterior obtained with any of the considered methods (Exc-SM, Exc-SSM, ABC-FP, ABC-SM, ABC-SSM).

Let $P^{(t)}(\cdot|x^0)$ denote the posterior predictive distribution at time t conditioned on an observation x^0 , and let $x^{0,(t)}$ denote the t -th timestep of the observation. Then, we are interested in:

$$S_E(P^{(t)}(\cdot|x^0), x^{0,(t)}) \quad \text{and} \quad S_k(P^{(t)}(\cdot|x^0), x^{0,(t)});$$

we estimate these using samples from $P^{(t)}(\cdot|x^0)$ with the unbiased estimators which can be easily derived for the Energy and Kernel Scores. In the Kernel Score, the bandwidth of the Gaussian kernel is set from simulations as described in Appendix F.4.1.

In this way, we obtain a score for an observation x^0 at each timestep of the model. This procedure is repeated for 100 different observations; the scores at each timestep are reported in Figure 15, while the summed scores over timesteps were reported in Figure 7 in the main text. In both Figures, we report the median value and various quantiles over the considered

Beta						
	Exc-SM	ABC-SM	ABC-SSM	ABC-FP	PMC-SL	PMC-RE
NN training	$2 \cdot 10^4$	$2 \cdot 10^4$	$2 \cdot 10^4$	$2 \cdot 10^4$	-	-
Inference	0	$2.5 \cdot 10^4$	$1.8 \cdot 10^4$	-	$1 \cdot 10^5$	$1 \cdot 10^6$
Total 1 obs	$2 \cdot 10^4$	$4.5 \cdot 10^4$	$3.8 \cdot 10^4$	-	$1 \cdot 10^5$	$1 \cdot 10^6$
Total 3 obs	$2 \cdot 10^4$	$9.5 \cdot 10^4$	$7.4 \cdot 10^4$	-	$3 \cdot 10^5$	$3 \cdot 10^6$
Total 100 obs	$2 \cdot 10^4$	$2.52 \cdot 10^6$	$1.82 \cdot 10^6$	-	$1 \cdot 10^7$	$1 \cdot 10^8$
Gamma						
	Exc-SM	ABC-SM	ABC-SSM	ABC-FP	PMC-SL	PMC-RE
NN training	$2 \cdot 10^4$	$2 \cdot 10^4$	$2 \cdot 10^4$	$2 \cdot 10^4$	-	-
Inference	0	$4.7 \cdot 10^4$	$2.9 \cdot 10^4$	-	$1 \cdot 10^5$	$1 \cdot 10^6$
Total 1 obs	$2 \cdot 10^4$	$6.7 \cdot 10^4$	$4.9 \cdot 10^4$	-	$1 \cdot 10^5$	$1 \cdot 10^6$
Total 3 obs	$2 \cdot 10^4$	$1.61 \cdot 10^5$	$1.07 \cdot 10^5$	-	$3 \cdot 10^5$	$3 \cdot 10^6$
Total 100 obs	$2 \cdot 10^4$	$4.72 \cdot 10^6$	$2.92 \cdot 10^6$	-	$1 \cdot 10^7$	$1 \cdot 10^8$
Gaussian						
	Exc-SM	ABC-SM	ABC-SSM	ABC-FP	PMC-SL	PMC-RE
NN training	$2 \cdot 10^4$	$2 \cdot 10^4$	$2 \cdot 10^4$	$2 \cdot 10^4$	-	-
Inference	0	$2.7 \cdot 10^4$	$2.6 \cdot 10^4$	-	-	$1 \cdot 10^6$
Total 1 obs	$2 \cdot 10^4$	$4.7 \cdot 10^4$	$4.6 \cdot 10^4$	-	-	$1 \cdot 10^6$
Total 3 obs	$2 \cdot 10^4$	$1.01 \cdot 10^5$	$9.8 \cdot 10^4$	-	-	$3 \cdot 10^6$
Total 100 obs	$2 \cdot 10^4$	$2.72 \cdot 10^6$	$2.62 \cdot 10^6$	-	-	$1 \cdot 10^8$
AR2						
	Exc-SM	ABC-SM	ABC-SSM	ABC-FP	PMC-SL	PMC-RE
NN training	$2 \cdot 10^4$	$2 \cdot 10^4$	$2 \cdot 10^4$	$2 \cdot 10^4$	-	-
Inference	0	$3.2 \cdot 10^4$	$3.1 \cdot 10^4$	$2.3 \cdot 10^4$	$1 \cdot 10^5$	-
Total 1 obs	$2 \cdot 10^4$	$5.2 \cdot 10^4$	$5.1 \cdot 10^4$	$4.3 \cdot 10^4$	$1 \cdot 10^5$	-
Total 3 obs	$2 \cdot 10^4$	$1.16 \cdot 10^5$	$1.13 \cdot 10^5$	$8.9 \cdot 10^4$	$3 \cdot 10^5$	-
Total 100 obs	$2 \cdot 10^4$	$3.22 \cdot 10^6$	$3.12 \cdot 10^6$	$2.32 \cdot 10^6$	$1 \cdot 10^7$	-
MA2						
	Exc-SM	ABC-SM	ABC-SSM	ABC-FP	PMC-SL	PMC-RE
NN training	$2 \cdot 10^4$	$2 \cdot 10^4$	$2 \cdot 10^4$	$2 \cdot 10^4$	-	-
Inference	0	$3.1 \cdot 10^4$	$2.1 \cdot 10^4$	$2.0 \cdot 10^4$	$1 \cdot 10^5$	-
Total 1 obs	$2 \cdot 10^4$	$5.1 \cdot 10^4$	$4.1 \cdot 10^4$	$4.0 \cdot 10^4$	$1 \cdot 10^5$	-
Total 3 obs	$2 \cdot 10^4$	$1.13 \cdot 10^5$	$8.3 \cdot 10^4$	$8.0 \cdot 10^4$	$3 \cdot 10^5$	-
Total 100 obs	$2 \cdot 10^4$	$3.12 \cdot 10^6$	$2.12 \cdot 10^6$	$2.02 \cdot 10^6$	$1 \cdot 10^7$	-

Table 12: **Model simulations needed for the different techniques**, for both NN training and inference; for ABC-FP, ABC-SM, ABC-SSM, PMC-SL and PMC-RE, we report simulations needed to obtain performance at least as good as Exc-SM; in case the approach does not reach the same performance as Exc-SM, we denote that by a dash. Notice that PMC-SL and PMC-RE do not require NN training before performing inference. We also show the total number of simulations needed to apply the different approaches on 1, 3 and 100 observations, by taking into account NN training and inference steps.

SCORE MATCHED NEURAL EXPONENTIAL FAMILIES FOR LFI

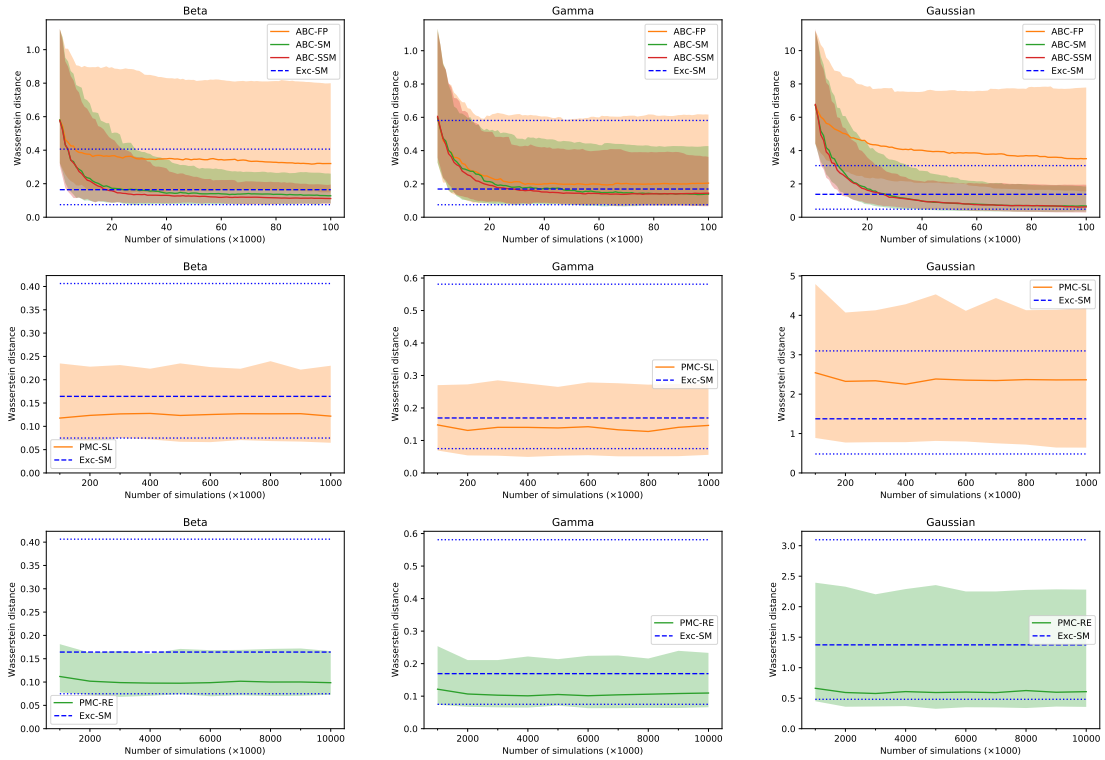


Figure 13: Wasserstein distance between approximate and exact posterior at different iterations of the sequential algorithms for the exponential family models, for 100 different observations. The solid line denotes median, while colored regions denote 95% probability density region; an horizontal line denoting the value obtained with Exc-SM is also represented, 95% probability density region denoted by dotted horizontal lines. The horizontal axis reports the number of model simulations corresponding to the iteration of the different algorithms.

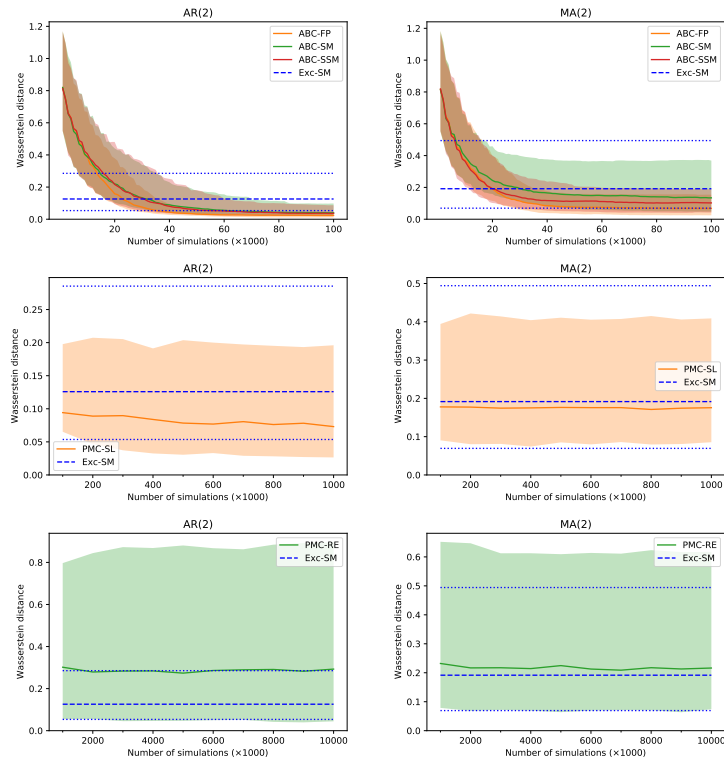


Figure 14: Wasserstein distance between approximate and exact posterior at different iterations of the sequential algorithms for the AR(2) and MA(2) models, for 100 different observations. The solid line denotes median, while colored regions denote 95% probability density region; an horizontal line denoting the value obtained with Exc-SM is also represented, 95% probability density region denoted by dotted horizontal lines. The horizontal axis reports the number of model simulations corresponding to the iteration of the different algorithms.

100 observations. It can be seen that ABC-FP is slightly outperformed by our proposed methods for both the large and small Lorenz96 configuration. Additionally, notice how, in the small configuration, the Energy score has an increasing trend over t , while the Kernel one instead decreases.

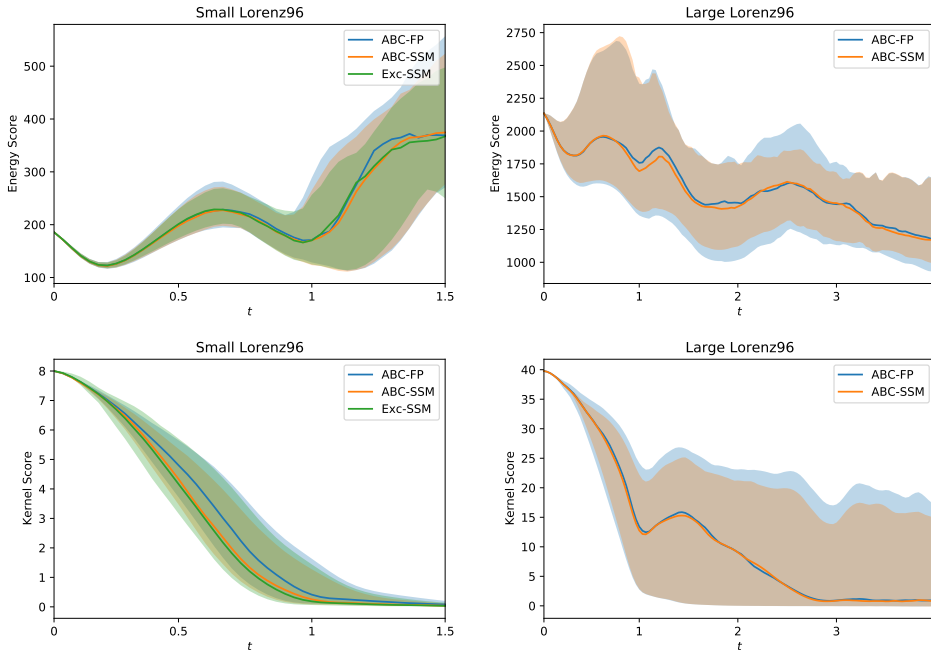


Figure 15: **Posterior predictive performance of the different methods at each timestep according to the Kernel and Energy Scores**; the smaller, the better. Samples from the posterior predictive were obtained for 100 observations, and both Scoring Rules estimated. The solid lines denote medians over the 100 observations, while colored regions denote 95% probability density region.

F.4.1 SETTING γ IN THE KERNEL SCORING RULE

In the kernel score S_k , the Gaussian kernel in Eq. (??) is used across this work. There, the kernel bandwidth γ is a free parameter. In order to ensure comparability between the Scoring Rule values for different observations and inference methods, the value of γ needs to be fixed independently on both.

Inspired by Park et al. (2016), we exploit an empirical procedure to set γ for the specific case of multivariate time-series models (of which our Lorenz96 model is an instance, see Section 5.3). Specifically, we use the following procedure:

1. Draw a set of parameter values $\theta_j \sim \pi(\theta)$ and simulations $x_j \sim p(\cdot|\theta_j)$, for $j = 1, \dots, m$.
2. Estimate the median of $\{\|x_j^{(t)} - x_k^{(t)}\|_2\}_{jk}^m$ and call it $\hat{\gamma}^{(t)}$, for all values of t .
3. Set the estimate for γ as the median of $\hat{\gamma}^{(t)}$ over all considered timesteps t .

Empirically, we use $m = 1000$. Note that the above strategy uses medians rather than means as those are more robust to outliers in the estimates. With this method, we obtain $\gamma \approx 1.54$ for the small version of the Lorenz96 model and $\gamma = 6.38$ for the large one.

References

- Mattias Akesson, Prashant Singh, Fredrik Wrede, and Andreas Hellander. Convolutional neural networks as summary statistics for approximate Bayesian computation. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2021.
- Carlo Albert, R. Künsch Hans, and Andreas Scheidegger. A simulated annealing approach to approximate Bayesian computations. *Statistics and Computing*, 25:1217–1232, 2015.
- Pierre Alquier, Nial Friel, Richard Everitt, and Aidan Boland. Noisy Monte Carlo: Convergence of Markov chains with approximate transition kernels. *Statistics and Computing*, 26(1-2):29–47, 2016.
- Ziwen An, Leah F South, David J Nott, and Christopher C Drovandi. Accelerating Bayesian synthetic likelihood with the graphical lasso. *Journal of Computational and Graphical Statistics*, 28(2):471–475, 2019.
- Ziwen An, David J Nott, and Christopher Drovandi. Robust Bayesian synthetic likelihood via a semi-parametric approach. *Statistics and Computing*, 30(3):543–557, 2020.
- Michael Arbel and Arthur Gretton. Kernel conditional exponential family. In *International Conference on Artificial Intelligence and Statistics*, pages 1337–1346. PMLR, 2018.
- V.I. Avrutskiy. Backpropagation generalized for output derivatives. *arXiv preprint arXiv:1712.04185*, 2017.
- Espen Bernton, Pierre E. Jacob, Mathieu Gerber, and Christian P. Robert. Approximate Bayesian computation with the Wasserstein distance. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 81(2):235–269, 2019. doi: <https://doi.org/10.1111/rssb.12312>. URL <https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/rssb.12312>.
- Alberto Caimo and Nial Friel. Bayesian inference for exponential random graph models. *Social Networks*, 33(1):41–55, 2011.
- Olivier Cappé, Arnaud Guillin, Jean-Michel Marin, and Christian P Robert. Population Monte Carlo. *Journal of Computational and Graphical Statistics*, 13(4):907–929, 2004.
- Bob Carpenter, Andrew Gelman, Matthew D Hoffman, Daniel Lee, Ben Goodrich, Michael Betancourt, Marcus Brubaker, Jiqiang Guo, Peter Li, and Allen Riddell. Stan: A probabilistic programming language. *Journal of statistical software*, 76(1), 2017.
- Yanzhi Chen, Dinghui Zhang, Michael U Gutmann, Aaron Courville, and Zhanxing Zhu. Neural approximate sufficient statistics for implicit models. In *Ninth International Conference on Learning Representations 2021*, 2021.
- Badr-Eddine Chérif-Abdellatif and Pierre Alquier. MMD-Bayes: Robust Bayesian estimation via maximum mean discrepancy. In *Symposium on Advances in Approximate Bayesian Inference*, pages 1–21. PMLR, 2020.
- Bo Dai, Hanjun Dai, Arthur Gretton, Le Song, Dale Schuurmans, and Niao He. Kernel exponential family estimation via doubly dual embedding. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2321–2330. PMLR, 2019a.
- Bo Dai, Zhen Liu, Hanjun Dai, Niao He, Arthur Gretton, Le Song, and Dale Schuurmans. Exponential family estimation via adversarial dynamics embedding. In *Advances in Neural Information Processing Systems*, pages 10979–10990, 2019b.

- Alexander Philip Dawid and Monica Musio. Theory and applications of proper scoring rules. *Metron*, 72(2):169–183, 2014.
- Christopher Drovandi and David T Frazier. A comparison of likelihood-free methods with and without summary statistics. *arXiv preprint arXiv:2103.02407*, 2021.
- Ritabrata Dutta, Bastien Chopard, Jonas Lätt, Frank Dubois, Karim Zouaoui Boudjeltia, and Antonietta Mira. Parameter estimation of platelets deposition: Approximate Bayesian computation with high performance computing. *Frontiers in physiology*, 9, 2018.
- Ritabrata Dutta, Susana N Gomes, Dante Kalise, and Lorenzo Pacchiardi. Using mobility data in the design of optimal lockdown strategies for the covid-19 pandemic. *PLoS Computational Biology*, 17(8):e1009236, 2021a.
- Ritabrata Dutta, Marcel Schoengens, Lorenzo Pacchiardi, Avinash Ummadisingu, Nicole Widmer, Pierre Künzli, Jukka-Pekka Onnela, and Antonietta Mira. ABCpy: A high-performance computing perspective to approximate bayesian computation. *Journal of Statistical Software*, 100(7):1–38, 2021b. doi: 10.18637/jss.v100.i07. URL <https://www.jstatsoft.org/index.php/jss/article/view/v100i07>.
- Richard G Everitt. Bayesian parameter estimation for latent Markov random fields and social networks. *Journal of Computational and graphical Statistics*, 21(4):940–960, 2012.
- Richard G Everitt, Dennis Prangle, Philip Maybank, and Mark Bell. Marginal sequential Monte Carlo for doubly intractable models. *arXiv preprint arXiv:1710.04382*, 2017.
- Matteo Fasiolo, Simon N Wood, Florian Hartig, and Mark V Bravington. An extended empirical saddlepoint approximation for intractable likelihoods. *Electronic Journal of Statistics*, 12(1):1544–1578, 2018.
- Paul Fearnhead and Dennis Prangle. Constructing summary statistics for approximate Bayesian computation: semi-automatic approximate Bayesian computation [with Discussion]. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 74(3):419–474, 2012. ISSN 1369-7412.
- Charles J Geyer. Markov chain Monte Carlo maximum likelihood. In *Computing science and statistics: Proceedings of 23rd Symposium on the Interface Interface Foundation, Fairfax Station, 1991*, pages 156–163, 1991.
- Alexander Gleim and Christian Pigorsch. Approximate Bayesian computation with indirect summary statistics. *Draft paper: <http://ect-pigorsch.mee.uni-bonn.de/data/research/papers>*, 2013.
- Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.
- David Greenberg, Marcel Nonnenmacher, and Jakob Macke. Automatic posterior transformation for likelihood-free inference. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2404–2414. PMLR, 09–15 Jun 2019. URL <http://proceedings.mlr.press/v97/greenberg19a.html>.
- Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.
- Michael U Gutmann and Aapo Hyvärinen. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13(Feb):307–361, 2012.
- Michael U Gutmann, Jukka Corander, et al. Bayesian optimization for likelihood-free inference of simulator-based statistical models. *Journal of Machine Learning Research*, 2016.

- J Hakkarainen, A Ilin, A Solonen, M Laine, H Haario, J Tamminen, E Oja, and H Järvinen. On closure parameter estimation in chaotic systems. *Nonlinear processes in Geophysics*, 19(1):127–143, 2012.
- Joeri Hermans, Volodimir Begy, and Gilles Louppe. Likelihood-free MCMC with amortized approximate ratio estimators. In *International Conference on Machine Learning*, pages 4239–4248. PMLR, 2020.
- Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.
- Harold Hotelling. Relations between two sets of variates. *Biometrika*, 28(3/4):321–377, 1936.
- Aapo Hyvärinen. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(Apr):695–709, 2005.
- Aapo Hyvärinen. Some extensions of score matching. *Computational statistics & data analysis*, 51(5):2499–2512, 2007.
- Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating deep network training by reducing internal covariate shift. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France, 07–09 Jul 2015. PMLR. URL <http://proceedings.mlr.press/v37/ioffe15.html>.
- Marko Järvenpää, Michael U Gutmann, Arius Pleska, Aki Vehtari, and Pekka Marttinen. Efficient acquisition rules for model-based approximate Bayesian computation. *Bayesian Analysis*, 14(2):595–622, 2019.
- Marko Jarvenpaa, Aki Vehtari, and Pekka Marttinen. Batch simulations and uncertainty quantification in gaussian process surrogate approximate Bayesian computation. In *Conference on Uncertainty in Artificial Intelligence*, pages 779–788. PMLR, 2020.
- Marko Järvenpää, Michael U Gutmann, Aki Vehtari, and Pekka Marttinen. Parallel Gaussian process surrogate Bayesian inference with noisy likelihood evaluations. *Bayesian Analysis*, 16(1):147–178, 2021.
- Bai Jiang, Tung-yu Wu, Charles Zheng, and Wing H Wong. Learning summary statistic for approximate Bayesian computation via deep neural network. *Statistica Sinica*, pages 1595–1618, 2017.
- Ilyes Khemakhem, Ricardo Pio Monti, Diederik P. Kingma, and Aapo Hyvärinen. ICE-BeeM: Identifiable conditional energy-based deep models. *CoRR*, abs/2002.11537, 2020. URL <https://arxiv.org/abs/2002.11537>.
- Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *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>.
- Nadja Klein, David J Nott, and Michael Stanley Smith. Marginally calibrated deep distributional regression. *Journal of Computational and Graphical Statistics*, pages 1–17, 2020.
- Bernard Osgood Koopman. On distributions admitting a sufficient statistic. *Transactions of the American Mathematical society*, 39(3):399–409, 1936.
- Olivier Ledoit and Michael Wolf. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of multivariate analysis*, 88(2):365–411, 2004.
- Faming Liang. A double Metropolis–Hastings sampler for spatial models with intractable normalizing constants. *Journal of Statistical Computation and Simulation*, 80(9):1007–1022, 2010.
- Faming Liang, Ick Hoon Jin, Qifan Song, and Jun S Liu. An adaptive exchange algorithm for sampling from distributions with intractable normalizing constants. *Journal of the American Statistical Association*, 111(513):377–393, 2016.

- Jarno Lintusaari, Michael U. Gutmann, Ritabrata Dutta, Samuel Kaski, and Jukka Corander. Fundamentals and recent developments in approximate Bayesian computation. *Systematic Biology*, 66(1):e66–e82, 2017. ISSN 1076836X. doi: 10.1093/sysbio/syw077. URL <https://doi.org/10.1093/sysbio/syw077>.
- Qiang Liu and Dilin Wang. Stein variational gradient descent: A general purpose Bayesian inference algorithm. In *Advances in neural information processing systems*, pages 2378–2386, 2016.
- Song Liu and Takafumi Kanamori. Estimating density models with complex truncation boundaries. *arXiv preprint arXiv:1910.03834*, 2019.
- Edward N Lorenz. Predictability: A problem partly solved. In *Proc. Seminar on predictability*, volume 1, 1996.
- Jan-Matthis Lueckmann, Pedro J Goncalves, Giacomo Bassetto, Kaan Öcal, Marcel Nonnenmacher, and Jakob H Macke. Flexible statistical inference for mechanistic models of neural dynamics. In *Advances in Neural Information Processing Systems*, pages 1289–1299, 2017.
- Jan-Matthis Lueckmann, Giacomo Bassetto, Theofanis Karaletsos, and Jakob H Macke. Likelihood-free inference with emulator networks. In *Symposium on Advances in Approximate Bayesian Inference*, pages 32–53. PMLR, 2019.
- Kanti V Mardia, John T Kent, and Arnab K Laha. Score matching estimators for directional distributions. *arXiv preprint arXiv:1604.08470*, 2016.
- Jean-Michel Marin, Pierre Pudlo, Christian P Robert, and Robin J Ryder. Approximate Bayesian computational methods. *Statistics and Computing*, 22(6):1167–1180, 2012.
- Pekka Marttinen, Nicholas J Croucher, Michael U Gutmann, Jukka Corander, and William P Hanage. Recombination produces coherent bacterial species clusters in both core and accessory genomes. *Microbial Genomics*, 1(5), 2015.
- Takuo Matsubara, Jeremias Knoblauch, François-Xavier Briol, and Chris J. Oates. Robust generalised Bayesian inference for intractable likelihoods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 84(3):997–1022, 2022. doi: <https://doi.org/10.1111/rssb.12500>. URL <https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/rssb.12500>.
- Trevelyan J McKinley, Ian Vernon, Ioannis Andrianakis, Nicky McCreesh, Jeremy E Oakley, Rebecca N Nsubuga, Michael Goldstein, Richard G White, et al. Approximate Bayesian computation and simulation-based inference for complex stochastic epidemic models. *Statistical science*, 33(1):4–18, 2018.
- Edward Meeds and Max Welling. GPS-ABC: Gaussian process surrogate approximate Bayesian computation. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, pages 593–602, 2014.
- Amanda Minter and Renata Retkute. Approximate Bayesian computation for infectious disease modelling. *Epidemics*, 29:100368, 2019.
- Matthew T Moores, Christopher C Drovandi, Kerrie Mengersen, and Christian P Robert. Pre-processing for approximate Bayesian computation in image analysis. *Statistics and Computing*, 25(1):23–33, 2015.
- Iain Murray, Zoubin Ghahramani, and David MacKay. MCMC for doubly-intractable distributions. *arXiv preprint arXiv:1206.6848*, 2012.
- Hien Duy Nguyen, Julyan Arbel, Hongliang Lü, and Florence Forbes. Approximate Bayesian computation via the energy statistic. *IEEE Access*, 8:131683–131698, 2020.
- Lorenzo Pacchiardi, Pierre Künzli, Marcel Schöngens, Bastien Chopard, and Ritabrata Dutta. Distance-learning for approximate Bayesian computation to model a volcanic eruption. *Sankhya B*, Jan 2020. ISSN 0976-8394. doi: 10.1007/s13571-019-00208-8. URL <https://doi.org/10.1007/s13571-019-00208-8>.

- George Papamakarios and Iain Murray. Fast ε -free inference of simulation models with Bayesian conditional density estimation. In *Advances in Neural Information Processing Systems*, pages 1028–1036, 2016.
- George Papamakarios, David Sterratt, and Iain Murray. Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *Proceedings of Machine Learning Research*, volume 89 of *Proceedings of Machine Learning Research*, pages 837–848. PMLR, 16–18 Apr 2019. URL <http://proceedings.mlr.press/v89/papamakarios19a.html>.
- George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021. URL <http://jmlr.org/papers/v22/19-1028.html>.
- Jaewoo Park and Murali Haran. Bayesian inference in the presence of intractable normalizing functions. *Journal of the American Statistical Association*, 113(523):1372–1390, 2018.
- Mijung Park, Wittawat Jitkrittum, and Dino Sejdinovic. K2-ABC: Approximate Bayesian computation with kernel embeddings. In *Artificial Intelligence and Statistics*, 2016.
- Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.
- Leah F Price, Christopher C Drovandi, Anthony Lee, and David J Nott. Bayesian synthetic likelihood. *Journal of Computational and Graphical Statistics*, 27(1):1–11, 2018.
- Stefan T Radev, Ulf K Mertens, Andreas Voss, Lynton Ardizzone, and Ullrich Köthe. BayesFlow: Learning complex stochastic models with invertible neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.
- Maria L Rizzo and Gábor J Székely. Energy distance. *Wiley interdisciplinary reviews: Computational statistics*, 8(1):27–38, 2016.
- Gareth O Roberts, Andrew Gelman, Walter R Gilks, et al. Weak convergence and optimal scaling of random walk Metropolis algorithms. *The annals of applied probability*, 7(1):110–120, 1997.
- Erlis Ruli, Nicola Sartori, and Laura Ventura. Approximate Bayesian computation with composite score functions. *Statistics and Computing*, 26(3):679–692, 2016.
- Jascha Sohl-Dickstein, Peter B Battaglino, and Michael R DeWeese. New method for parameter estimation in probabilistic models: minimum probability flow. *Physical review letters*, 107(22):220601, 2011.
- Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. Sliced score matching: A scalable approach to density and score estimation. In Ryan P. Adams and Vibhav Gogate, editors, *Proceedings of The 35th Uncertainty in Artificial Intelligence Conference*, volume 115 of *Proceedings of Machine Learning Research*, pages 574–584, Tel Aviv, Israel, 22–25 Jul 2020. PMLR. URL <http://proceedings.mlr.press/v115/song20a.html>.
- Bharath Sriperumbudur, Kenji Fukumizu, Arthur Gretton, Aapo Hyvärinen, and Revant Kumar. Density estimation in infinite dimensional exponential families. *The Journal of Machine Learning Research*, 18(1):1830–1888, 2017.
- Esteban G Tabak, Giulio Trigila, and Wenjun Zhao. Conditional density estimation and simulation through optimal transport. *Machine Learning*, pages 1–24, 2020.
- Simon Tavaré, David J Balding, Robert C Griffiths, and Peter Donnelly. Inferring coalescence times from DNA sequence data. *Genetics*, 145(2):505–518, 1997.

- Owen Thomas, Ritabrata Dutta, Jukka Corander, Samuel Kaski, Michael U Gutmann, et al. Likelihood-free inference by ratio estimation. *Bayesian Analysis*, 2020.
- Jean-Francois Ton, CHAN Lucian, Yee Whye Teh, and Dino Sejdinovic. Noise contrastive meta-learning for conditional density estimation using kernel mean embeddings. In *International Conference on Artificial Intelligence and Statistics*, pages 1099–1107. PMLR, 2021.
- Tina Toni, David Welch, Natalja Strelkowa, Andreas Ipsen, and Michael PH Stumpf. Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. *Journal of the Royal Society Interface*, 6(31):187–202, 2009.
- Pascal Vincent. A connection between score matching and denoising autoencoders. *Neural computation*, 23(7):1661–1674, 2011.
- Yuexi Wang, Tetsuya Kaji, and Veronika Ročková. Approximate bayesian computation via classification. *arXiv preprint arXiv:2111.11507*, 2021.
- Ziyu Wang, Shuyu Cheng, Li Yueru, Jun Zhu, and Bo Zhang. A Wasserstein minimum velocity approach to learning unnormalized models. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 3728–3738. PMLR, 26–28 Aug 2020. URL <https://proceedings.mlr.press/v108/wang20j.html>.
- Li Wenliang, Dougal Sutherland, Heiko Strathmann, and Arthur Gretton. Learning deep kernels for exponential family densities. In *International Conference on Machine Learning*, pages 6737–6746. PMLR, 2019.
- Richard Wilkinson. Accelerating ABC methods using Gaussian processes. In *Artificial Intelligence and Statistics*, pages 1015–1023. PMLR, 2014.
- Daniel S Wilks. Effects of stochastic parametrizations in the Lorenz’96 system. *Quarterly Journal of the Royal Meteorological Society*, 131(606):389–407, 2005.
- Samuel Wiqvist, Pierre-Alexandre Mattei, Umberto Picchini, and Jes Frelsen. Partially exchangeable networks and architectures for learning summary statistics in approximate Bayesian computation. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6798–6807. PMLR, 09–15 Jun 2019. URL <http://proceedings.mlr.press/v97/wiqvist19a.html>.
- Simon N Wood. Statistical inference for noisy nonlinear ecological dynamic systems. *Nature*, 466(7310):1102, 2010.
- Shiqing Yu, Mathias Drton, and Ali Shojaie. Generalized score matching for non-negative data. *Journal of Machine Learning Research*, 20(76):1–70, 2019. URL <http://jmlr.org/papers/v20/18-278.html>.

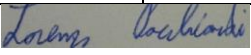
Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).

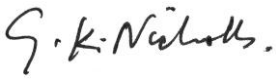
Title of Paper	Score Matched Neural Exponential Families for Likelihood-Free Inference
Publication Status	<input checked="" type="checkbox"/> Published <input type="checkbox"/> Accepted for Publication <input type="checkbox"/> Submitted for Publication <input type="checkbox"/> Unpublished and unsubmitted work written in a manuscript style
Publication Details	Pacchiardi, Lorenzo, and Ritabrata Dutta. "Score Matched Neural Exponential Families for Likelihood-Free Inference." <i>J. Mach. Learn. Res.</i> 23 (2022): 38-1.

Student Confirmation

Student Name:	Lorenzo Pacchiardi		
Contribution to the Paper	I am the first author of this paper. Prof. Dutta provided the original idea of a way to learn summary statistics using the exponential family. I came up with the precise formulation, the way to train the exponential family with score matching, and the possibility of using the resulting approximation as a stand-alone approximation scheme (with doubly-intractable MCMC). I further coded the method, ran the simulations and wrote most of the paper. Prof. Dutta advised along the way and corrected the paper draft.		
Signature		Date	1 st September 2022

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Professor Geoff Nicholls			
Supervisor comments			
Signature		Date	15-09-22

This completed form should be included in the thesis, at the end of the relevant chapter.

Chapter 3

Generalized Bayesian Likelihood-Free Inference Using Scoring Rules Estimators

Unpublished and unsubmitted work.

Generalized Bayesian Likelihood-Free Inference Using Scoring Rules Estimators

Lorenzo Pacchiardi^{1*}, Ritabrata Dutta²

¹*Department of Statistics, University of Oxford, UK*

²*Department of Statistics, University of Warwick, UK*

Abstract

We propose a framework for Bayesian Likelihood-Free Inference (LFI) based on Generalized Bayesian Inference. To define the generalized posterior, we use Scoring Rules (SRs), which evaluate probabilistic models given an observation. In LFI, we can sample from the model but not evaluate the likelihood; for this reason, we employ SRs which can be estimated empirically. Our framework includes novel approaches and popular LFI techniques (such as Bayesian Synthetic Likelihood) and enjoys posterior consistency in a well-specified setting when a *strictly-proper* SR is used (i.e., one whose expectation is uniquely minimized when the model corresponds to the data generating process). In general, our framework does *not* approximate the standard posterior; as such, it is possible to achieve outlier robustness, which we prove is the case for the Kernel and Energy Scores. We also discuss a strategy for tuning the learning rate in the generalized posterior suitable for the LFI setup. We run simulations studies with correlated pseudo-marginal Markov Chain Monte Carlo and compare with related approaches, which we show do not enjoy robustness and consistency.

1 Introduction

This work is concerned with performing inference for intractable-likelihood models, for which it is impossible or very expensive to evaluate the likelihood $p(y|\theta)$ for an observation y , but from which it is easy to simulate for any parameter value θ . Given y and a prior $\pi(\theta)$ on the parameters, the standard Bayesian posterior is $\pi(\theta|y) \propto \pi(\theta)p(y|\theta)$. However, obtaining that explicitly or sampling from it with Markov Chain Monte Carlo (MCMC) techniques is impossible without having access to the likelihood.

Standard Likelihood-Free Inference (LFI) techniques exploit model simulations to approximate the exact posterior distribution when the likelihood is unavailable. Broadly, two categories of approaches exist, differing for the kind of approximation: methods in the first category [Price et al., 2018, An et al., 2020, Thomas et al., 2020] replace the intractable likelihood with a surrogate one whose parameters are estimated from simulations. The second category is constituted by Approximate Bayesian Computation (ABC) methods [Lintusaari et al., 2017, Bernton et al., 2019], which implicitly approximate the likelihood by weighting parameter values according to the mismatch between observed and simulated data.

In this work, we extend the first category above by building on the generalized Bayesian inference setup [Bissiri et al., 2016, Jewson et al., 2018, Knoblauch et al., 2022]: given a generic loss $\ell(y, \theta)$ between a single observation y and parameter θ , the following update for beliefs on parameter values can be defined:

$$\pi(\theta|y) \propto \pi(\theta) \exp(-w \cdot \ell(y, \theta)); \quad (1)$$

this allows to learn about the parameter value minimizing the expected loss over the data generating process¹ and respects Bayesian additivity (i.e., the posterior obtained by sequentially updating the

*Corresponding author: lorenzo.pacchiardi@stats.ox.ac.uk.

¹Indeed setting $\ell(y, \theta) = -\log p(y|\theta)$ and $w = 1$ recovers the standard Bayes update, which learns about the parameter value minimizing the KL divergence [Bissiri et al., 2016].

belief with a set of observations does not depend on the order the observations are received). Here, the learning rate w controls speed of learning.

In the following, we will distinguish between a statistical model (or distribution) P_θ and its likelihood function $p(y|\theta)$. In general, the update in Eq. (1) does not require the quantity θ to parametrize a statistical model P_θ . In LFI, however, we cannot evaluate $p(y|\theta)$, but a model P_θ is present. Therefore, we propose to take $\ell(y, \theta)$ to be a Scoring Rule (SR) $S(P_\theta, y)$, which assesses the performance of P_θ for an observation y , thus obtaining the *Scoring Rule posterior* π_S . If $S(P_\theta, y)$ can be estimated with samples from P_θ , we can apply this approach in a LFI setting without worrying about the missing likelihood $p(y|\theta)$. Thus, our main contribution is: *we introduce an LFI framework based on Scoring Rules, which comprehends previously introduced LFI methods and novel ones. Additionally, for some Scoring Rules, the resulting posterior enjoys outlier robustness while asymptotically concentrating on the true parameter value in a well-specified setting.*

Specifically, π_S concentrates asymptotically on the minimizer of the expected Scoring Rule, provided such minimizer is unique; for well-specified models, the minimizer is the true parameter value if so-called *strictly proper* SRs are used. To state this, we rely on previous results in Miller [2021], for which we provide verifying assumptions for our setup. We also point out that the popular Bayesian Synthetic Likelihood (BSL, Price et al. 2018) is an instance of our framework but does not use a strictly proper SR, hence leading to some failures. Further, in general our framework does *not* approximate the standard Bayesian posterior, as it is the case for commonly-used LFI techniques; by relaxing this requirement, outlier robustness can be achieved, as we show for the Kernel and Energy Score. For the same SRs, we also establish a finite sample generalization bound.

Finally, we discuss a tuning strategy for the learning rate w which is suitable for the SR posterior in a LFI scenario and compare our Kernel and Energy Score posteriors with related approaches via simulation studies. To sample from an approximation of the SR posterior, we employ pseudo-marginal MCMC [Andrieu et al., 2009], inside which simulations from $P_{\theta'}$ are generated for each proposed θ' . To reduce the stickiness typical of pseudo-marginal chains, we exploit correlated pseudo-marginal MCMC [Dahlin et al., 2015, Deligiannidis et al., 2018], which was validated for BSL in Picchini et al. [2022]. We study posterior concentration with the g-and-k model (in both well-specified and misspecified case) and outlier robustness on a normal location example, as well as showcase the performance of our methods on other benchmarks.

The rest of this manuscript is organized as follows. In Sec. 2, which contains most of our contributions, we first review Scoring Rules and define the SR posterior; we then discuss theoretical properties, sampling method and our strategy to tune w . Next, we discuss the relation of our approach with previous works in Sec. 3, showing how some popular LFI approaches are instances of the SR posterior. Our simulation studies are presented in Sec. 4 and we conclude and suggest future directions in Sec. 5.

1.1 Notation

We will denote respectively by $\mathcal{X} \subseteq \mathbb{R}^d$ and $\Theta \subseteq \mathbb{R}^p$ the data and parameter space, which we assume to be Borel sets. We will assume the observations are generated by a distribution P_0 and use P_θ and $p(\cdot|\theta)$ to denote the distribution and likelihood of our model. Generic distributions will be indicated by P or Q , while S will denote a generic Scoring Rule. Other upper-case letters will denote random variables while lower-case ones will denote observed (fixed) values. We will denote by Y or y the observations (correspondingly random variables and realizations) and X or x the simulations. Subscripts will denote sample index and superscripts vector components. Also, we will respectively denote by $\mathbf{Y}_n = \{Y_i\}_{i=1}^n \in \mathcal{X}^n$ and $\mathbf{y}_n = \{y_i\}_{i=1}^n \in \mathcal{X}^n$ a set of random and fixed observations. Similarly, $\mathbf{X}_m = \{X_j\}_{j=1}^m \in \mathcal{X}^m$ and $\mathbf{x}_m = \{x_j\}_{j=1}^m \in \mathcal{X}^m$ denote a set of random and fixed model simulations. Finally, \perp will denote independence between random variables, while $X \sim P$ indicates a random variable distributed according to P .

2 Bayesian inference using Scoring Rules

A Scoring Rule (SR, Gneiting and Raftery, 2007) S is a function of a probability distribution over \mathcal{X} and of an observation in \mathcal{X} . In probabilistic forecasting, $S(P, y)$ represents the *penalty* incurred when

stating a forecast P for an observation y .²

Assuming that y is a realization of a random variable Y with distribution Q , the expected Scoring Rule is defined as:

$$S(P, Q) := \mathbb{E}_{Y \sim Q} S(P, Y),$$

where we overload notation in the second argument of S . The Scoring Rule S is *proper* relative to a set of distributions $\mathcal{P}(\mathcal{X})$ over \mathcal{X} if

$$S(Q, Q) \leq S(P, Q) \quad \forall P, Q \in \mathcal{P}(\mathcal{X}),$$

i.e., if the expected Scoring Rule is minimized in P when $P = Q$. Moreover, S is *strictly proper* relative to $\mathcal{P}(\mathcal{X})$ if $P = Q$ is the unique minimum:

$$S(Q, Q) < S(P, Q) \quad \forall P, Q \in \mathcal{P}(\mathcal{X}) \text{ s.t. } P \neq Q;$$

when minimizing an expected strictly proper Scoring Rule, a forecaster provides their true belief [Gneiting and Raftery, 2007].

The divergence related to a proper Scoring Rule [Dawid and Musio, 2014] can be defined as $D(P, Q) := S(P, Q) - S(Q, Q) \geq 0$. Notice that $P = Q \implies D(P, Q) = 0$, but there may be $P \neq Q$ such that $D(P, Q) = 0$. However, if S is strictly proper, $D(P, Q) = 0 \iff P = Q$, which is the commonly used condition to define a statistical divergence (as for instance the Kullback-Leibler, or KL, divergence). Therefore, each strictly proper Scoring Rule corresponds to a statistical divergence between probability distributions.

Consider now a set of independent and identically distributed observations $\mathbf{y}_n \in \mathcal{X}^n$ by a distribution P_θ . We introduce the SR posterior for S by setting $\ell(y, \theta) = S(P_\theta, y)$ in the general Bayes update in Eq. (1):

$$\pi_S(\theta | \mathbf{y}_n) \propto \pi(\theta) \exp \left\{ -w \sum_{i=1}^n S(P_\theta, y_i) \right\}. \quad (2)$$

The standard Bayes posterior is recovered from Eq. (2) by setting $w = 1$ and $S(P_\theta, y) = -\log p(y|\theta)$. Such choice of S is called the *log score*, is strictly proper, and corresponds to the Kullback-Leibler (KL) divergence. With the same S , $w \neq 1$ yields the fractional posterior [Holmes and Walker, 2017, Bhattacharya et al., 2019].

Remark 1 (Bayesian additivity). *The posterior in Eq. (2) satisfies Bayesian additivity (also called coherence, Bissiri et al. 2016): sequentially updating the belief with a set of observations does not depend on the order the observations are received. Some related approaches do not satisfy this property, notably the ABC posterior [Lintusaari et al., 2017, Bernton et al., 2019]. Additionally, methods building an estimate of the data generating density using all observations $\{y_i\}_{i=1}^n$ break additivity (as for instance the Hellinger posterior considered in Jewson et al. [2018]).*

In Sec. 2.1, we review some Scoring Rules which can be easily estimated with intractable-likelihood models. Some properties of the SR posterior are discussed in Sec. 2.2. In Sec. 2.3 and Sec. 2.4, we show how to sample from an approximation of the SR posterior using pseudo-marginal MCMC and propose a way to set the value of w .

2.1 Scoring Rules for LFI

Under the assumed LFI setup, we can sample from P_θ ; therefore, we are interested in SRs for which estimators can be easily obtained using generated samples. Specifically, we will estimate $S(P_\theta, y)$ with $\hat{S}(\mathbf{x}_m^{(\theta)}, y)$, where $\mathbf{x}_m^{(\theta)} = \{x_j^{(\theta)}\}_{j=1}^m$ is a set of samples $x_j^{(\theta)} \sim P_\theta$, and \hat{S} is such that $\hat{S}(\mathbf{X}_m^{(\theta)}, y) \rightarrow S(P_\theta, y)$ in probability as $m \rightarrow \infty$ (i.e., it estimates the SR consistently). Two such Scoring Rules are the Energy and Kernel Scores introduced in Sec. 1.2.2 in the introduction of the present thesis. Due to this and

²Some authors [Gneiting and Raftery, 2007] use the convention of $S(P, y)$ representing a *reward* rather than a penalty, which is equivalent up to change of sign.

their strict propriety, we will focus on those in studying our framework theoretically and empirically. Specifically, we will fix $\beta = 1$ for the Energy Score and use the Gaussian kernel in the Kernel Score.

Another scoring rule which allows for unbiased estimators and is at the core of a widely used LFI method is the Dawid–Sebastiani (DS) score, which is defined as:

$$S_{\text{DS}}(P, y) = \ln |\Sigma_P| + (y - \mu_P)^T \Sigma_P^{-1} (y - \mu_P), \quad (3)$$

where μ_P and Σ_P are the mean vector and covariance matrix of P . The DS score is the negative log-likelihood of a multivariate normal distribution with mean μ_P and covariance matrix Σ_P , up to some constants. Therefore, it is equivalent to the log score when P is a multivariate normal distribution. For a set of distributions $\mathcal{P}(\mathcal{X})$ with well-defined second moments, this SR is proper but not strictly so: several distributions of that class may yield the same score, as long as the two first moments match [Gneiting and Raftery, 2007]. It is strictly proper if distributions in $\mathcal{P}(\mathcal{X})$ are determined by their first two moments, as it is the case for the normal distribution. Using S_{DS} in Eq. (2) with $w = 1$ leads to the popular Bayesian Synthetic Likelihood approach (see Sec. 3). In an LFI setup, an empirical estimator \hat{S}_{DS} can be obtained by inserting any estimator of the mean and covariance matrix in Eq. (3).

Remark 2 (Non-invariance to change of data coordinates). *Violation of the likelihood principle implies that the SR posterior is in general not invariant to change of the coordinates used for representing the observations. This is a property common to loss-based frequentist estimators and to the generalized posterior obtained from them [Matsubara et al., 2022b]; see Appendix B for more details.*

2.2 Properties of the Scoring Rule posterior

2.2.1 Asymptotic normality

We first show that the SR posterior satisfies (under some conditions) a Bernstein-von Mises theorem ensuring asymptotic normality. The result we give is of independent interest as it generalizes the analogous result valid for the standard Bayesian posterior.

Without loss of generality, we fix here $w = 1$ (other values can be absorbed in the definition of S). The proof relies on the following assumptions:

A1 The expected Scoring Rule $S(P_\theta, P_0)$ is finite for all $\theta \in \Theta$; further, it has a unique minimizer:

$$\theta^* = \arg \min_{\theta \in \Theta} S(P_\theta, P_0) = \arg \min_{\theta \in \Theta} D(P_\theta, P_0).$$

Additionally, $H_\star := \nabla_\theta^2 S(P_\theta, P_0)|_{\theta=\theta^*}$ is positive definite.

A2 Let us denote $S'''(P_\theta, y)_{jkl} = \frac{\partial^3}{\partial \theta_j \partial \theta_k \partial \theta_l} S(P_\theta, Y)$. There exists an open neighborhood $E \subseteq \mathbb{R}^d$ of θ^* whose closure $\bar{E} \subseteq \Theta$ is such that, for all $j, k, l \in \{1, \dots, d\}$:

- $\theta \rightarrow S'''(P_\theta, y)_{jkl}$ is continuous in E and exists in \bar{E} for any fixed $y \in \mathcal{X}$,
- $y \rightarrow S'''(P_\theta, y)_{jkl}$ is measurable for any fixed $\theta \in \bar{E}$,
- $\mathbb{E}_{P_0} \sup_{\theta \in \bar{E}} |S'''(P_\theta, y)_{jkl}| < \infty$.

A3 For E defined above, there exists a compact $K \subseteq E$, with θ^* in the interior of K , such that:

$$P_0 \left\{ \liminf_n \inf_{\theta \in \Theta \setminus K} \frac{1}{n} \sum_{i=1}^n S(P_\theta, Y_i) > S(P_{\theta^*}, P_0) \right\} = 1.$$

A4 The prior has a density $\pi(\theta)$ with respect to Lebesgue measure; $\pi(\theta)$ is continuous and positive at θ^* .

Assumption **A3** is a regularity condition which can be replaced with clearer (but less general) Assumptions; see Appendix A.1.2. In Assumption **A1**, H_\star generalizes the standard Fisher information, which can be obtained by setting $S(P_\theta, y) = -\log p(y|\theta)$. Additionally, uniqueness of θ^\star is obtained by strictly proper S and a well-specified model (in which case observations were generated from P_{θ^\star}). If the model class is misspecified, a strictly proper S does not guarantee a unique minimizer (as in fact there may be pathological cases where multiple minimizers exist).

Theorem 1. *Let Assumptions **A1** to **A4** be true. Then, there is a sequence $\hat{\theta}^{(n)}(\mathbf{Y}_n)$ which converges almost surely to θ^\star as $n \rightarrow \infty$. Denote now by $\pi_S^*(\cdot|\mathbf{Y}_n)$ the density of $\sqrt{n}(\theta - \hat{\theta}^{(n)}(\mathbf{Y}_n))$ when $\theta \sim \pi_S(\cdot|\mathbf{Y}_n)$. Then as $n \rightarrow \infty$, with probability 1 over \mathbf{Y}_n :*

$$\int_{\mathbb{R}^p} |\pi_S^*(s|\mathbf{Y}_n) - \mathcal{N}(s|0, H_\star^{-1})| ds \rightarrow 0,$$

where $\mathcal{N}(\cdot|0, \Sigma)$ denotes the density of a multivariate normal distribution with zero mean vector and covariance matrix Σ .

Proof of Theorem 1 is reported in Appendix A.1.2 and uses a result for general Bayes posteriors with generic losses found in Miller [2021].

Theorem 1 implies that the SR posterior concentrates, with probability 1, on the parameter value minimizing the expected SR, if that minimizer is unique. If the model is well specified and S is strictly proper, the SR posterior concentrates therefore on the true parameter value; this property is usually referred to as *posterior consistency*.

Remark 3 (Non-invariance to change of data coordinates – continued). *Following on from Remark 2, notice that θ^\star depends on the data coordinates, unless the model is well specified and S is strictly proper. If that is not the case, SR posteriors using different data coordinates will concentrate on different parameter values in general. This property is coherent with the SR posterior learning about the parameter value which minimizes the expected Scoring Rule, which in turn depends on the chosen coordinate system. See Appendix B for more details.*

2.2.2 Finite sample generalization bound

We now consider the Energy and Kernel Score posteriors and their corresponding divergences, and provide a bound on the probability of deviation of the posterior expectation of the divergence from the minimum divergence achievable by the model. The bound holds with finite number of samples and does not require the model to be well specified nor the minimizer of the divergence to be unique. Such results are usually referred to as generalization bounds [Chérief-Abdellatif and Alquier, 2020]. For our bound to hold, we require the following *prior mass condition* with respect to a divergence D :

A5 The prior has density $\pi(\theta)$ (with respect to Lebesgue measure) which satisfies

$$\int_{B_n(\alpha_1)} \pi(\theta) d\theta \geq e^{-\alpha_2 \sqrt{n}}$$

for some constants $\alpha_1, \alpha_2 > 0$ and for all positive $n \in \mathbb{N}$, where we define the sets:

$$B_n(\alpha_1) := \{\theta \in \Theta : |D(P_\theta, P_0) - D(P_{\theta^\star}, P_0)| \leq \alpha_1 / \sqrt{n}\},$$

where $\theta^\star \in \arg \min_{\theta \in \Theta} D(P_\theta, P_0)$, which is assumed to be nonempty.

Assumption **A5** constrains the amount of prior mass given to D -balls with size decreasing as $n^{-1/2}$ to decrease slower than $e^{-\alpha_2 \sqrt{n}}$ for some α_2 . It is therefore a weak condition, as it bounds the mass by a quickly decreasing function while the radius is decreasing more slowly. Similar assumptions are taken in Chérief-Abdellatif and Alquier 2020, Matsubara et al. 2022b, where some examples of explicit verification can be found.

Our result (proved in Appendix A.2) further assumes either a bounded kernel k for the Kernel Score posterior, or bounded \mathcal{X} for the Energy Score posterior.

Theorem 2. *The following two statements hold for any $\epsilon > 0$:*

1. *Let the kernel k be such that $\sup_{x \in \mathcal{X}} k(x, x) \leq \kappa < \infty$, and let D_k be the divergence associated to S_k . Consider $\theta^* \in \arg \min_{\theta \in \Theta} D_k(P_\theta, P_0)$; if the prior $\pi(\theta)$ satisfies Assumption **A5** for D_k , we have for the kernel Score posterior π_{S_k} :*

$$P_0 \left(\left| \int_{\Theta} D_k(P_\theta, P_0) \pi_{S_k}(\theta | \mathbf{Y}_n) d\theta - D_k(P_{\theta^*}, P_0) \right| \geq \epsilon \right) \leq 2e^{-\frac{1}{2} \left(\frac{\sqrt{n}\epsilon - \alpha_1 - \alpha_2/w}{8\kappa} \right)^2}.$$

2. *Assume the space \mathcal{X} is bounded such that $\sup_{x, y \in \mathcal{X}} \|x - y\|_2 \leq B < \infty$, and let $D_E^{(\beta)}$ be the divergence associated with $S_E^{(\beta)}$. Consider $\theta^* \in \arg \min_{\theta \in \Theta} D_E^{(\beta)}(P_\theta, P_0)$; if the prior $\pi(\theta)$ satisfies Assumption **A5** for $D_E^{(\beta)}$, we have for the Energy score posterior $\pi_{S_E^{(\beta)}}$:*

$$P_0 \left(\left| \int_{\Theta} D_E^{(\beta)}(P_\theta, P_0) \pi_{S_E^{(\beta)}}(\theta | \mathbf{Y}_n) d\theta - D_E^{(\beta)}(P_{\theta^*}, P_0) \right| \geq \epsilon \right) \leq 2e^{-\frac{1}{2} \left(\frac{\sqrt{n}\epsilon - \alpha_1 - \alpha_2/w}{8B^\beta} \right)^2}.$$

As ϵ or n increases, the bound on the probability tends to 0; for $n \rightarrow \infty$, this implies that the SR posterior concentrates on those parameter values for which the model achieves minimum divergence from the data generating process P_0 , ensuring therefore consistency in the well-specified case. With respect to Theorem 1, Theorem 2 provides guarantees on the infinite sample behavior of the SR posterior even when θ^* is not unique; however, this result does not describe the specific form of the asymptotic distribution, which Theorem 1 instead does.

2.2.3 Global bias-robustness

We establish now robustness with respect to contamination in the dataset for the Kernel Score posterior with bounded kernel and the Energy Score posterior with bounded \mathcal{X} .

First, consider the empirical distribution of the observations $\hat{P}_n = \frac{1}{n} \sum_{i=1}^n \delta_{y_i}$. If we define:

$$L(\theta, \hat{P}_n) := \frac{1}{n} \sum_{i=1}^n S(P_\theta, y_i) = \mathbb{E}_{Y \sim \hat{P}_n} S(P_\theta, Y)$$

for a Scoring Rule S , the SR posterior in Eq. (2) can be rewritten as:

$$\pi_S(\theta | \mathbf{y}_n) = \pi_S(\theta | \hat{P}_n) \propto \pi(\theta) \exp \left\{ -wnL(\theta, \hat{P}_n) \right\}.$$

Next, consider the ϵ -contamination distribution $\hat{P}_{n, \epsilon, z} = (1 - \epsilon)\hat{P}_n + \epsilon\delta_z$, obtained by perturbing the fixed empirical distribution with an outlier z of weight ϵ . In this setup, the *posterior influence function* [Ghosh and Basu, 2016] can be defined as:

$$\text{PIF} \left(z, \theta, \hat{P}_n \right) := \left. \frac{d}{d\epsilon} \pi_S \left(\theta | \hat{P}_{n, \epsilon, z} \right) \right|_{\epsilon=0},$$

which measures the rate of change of the posterior in θ when an infinitesimal perturbation in z is added to the observations. We say the SR posterior is C -globally bias-robust if:

$$\sup_{\theta \in \Theta} \sup_{z \in \mathcal{X}} \left| \text{PIF} \left(z, \theta, \hat{P}_n \right) \right| \leq C$$

and globally bias-robust if $C < \infty$ [Matsubara et al., 2022b].

Theorem 3. *The following two independent statements hold:*

1. *Consider a kernel k such that $\sup_{x \in \mathcal{X}} k(x, x) \leq \kappa < \infty$; then, the Kernel Score posterior $\pi_{S_k}(\cdot | \mathbf{y}_n)$ is C -globally bias-robust with $C \leq 8wn\kappa e^{6wn\kappa} \sup_{\theta \in \Theta} \pi(\theta)$.*

2. Alternatively, assume the space \mathcal{X} is bounded such that $\sup_{x,y \in \mathcal{X}} \|x - y\|_2 \leq B < \infty$; then, the Energy Score posterior $\pi_{S_E^{(\beta)}}(\cdot | \mathbf{y}_n)$ is globally bias-robust with $C \leq 8wnB^\beta e^{2wnB^\beta} \sup_{\theta \in \Theta} \pi(\theta)$.

Proof is given in Appendix A.3. The Gaussian kernel (used across this work, see Sec. 2.1) is bounded. Our theoretical result does not hold for the Energy Score posterior when \mathcal{X} is unbounded. However, in practice we still find the Energy Score posterior to be robust to outliers in examples with unbounded \mathcal{X} (see Sec. 4.2).

2.3 Sampling the Scoring Rule posterior

In implementing our proposed approach, we use an MCMC where, for each proposed value of θ , we simulate $\mathbf{x}_m^{(\theta)} = \{x_j^{(\theta)}\}_{j=1}^m$ and estimate the target in Eq. (2) with:

$$\pi(\theta) \exp \left\{ -w \sum_{i=1}^n \hat{S}(\mathbf{x}_m^{(\theta)}, y_i) \right\}. \quad (4)$$

This procedure is an instance of pseudo-marginal MCMC [Andrieu et al., 2009], with target:

$$\pi_{\hat{S}}^{(m)}(\theta | \mathbf{y}_n) \propto \pi(\theta) p_{\hat{S}}^{(m)}(\mathbf{y}_n | \theta), \quad (5)$$

where:

$$\begin{aligned} p_{\hat{S}}^{(m)}(\mathbf{y}_n | \theta) &= \mathbb{E} \left[\exp \left\{ -w \sum_{i=1}^n \hat{S}(\mathbf{X}_m^{(\theta)}, y_i) \right\} \right] \\ &= \int \exp \left\{ -w \sum_{i=1}^n \hat{S}(\mathbf{x}_m^{(\theta)}, y_i) \right\} \prod_{j=1}^m p(x_j^{(\theta)} | \theta) dx_1^{(\theta)} dx_2^{(\theta)} \dots dx_m^{(\theta)}. \end{aligned}$$

For a single draw $\mathbf{x}_m^{(\theta)}$, the quantity in Eq. (4) is in fact a non-negative and unbiased estimate of the target in Eq. (5); this approach is similar to what is proposed in Drovandi et al. [2015] for inference with auxiliary likelihoods, which has also been used by Price et al. [2018] for BSL. As it was already the case for the latter, the target $\pi_{\hat{S}}^{(m)}(\theta | \mathbf{y}_n)$ is not the same as $\pi_S(\theta | \mathbf{y}_n)$ and depends on the number of simulations m ; in fact, in general:

$$\mathbb{E} \left[\exp \left\{ -w \sum_{i=1}^n \hat{S}(\mathbf{X}_m^{(\theta)}, y_i) \right\} \right] \neq \exp \left\{ -w \sum_{i=1}^n S(P_\theta, y_i) \right\},$$

even if $\hat{S}(\mathbf{x}_m^{(\theta)}, y)$ is an unbiased estimate of $S(P_\theta, y)$. However, it is possible to show that, as $m \rightarrow \infty$, $\pi_{\hat{S}}^{(m)}$ converges to π_S :

Theorem 4. *If $\hat{S}(\mathbf{X}_m^{(\theta)}, y_i)$ converges in probability to $S(P_\theta, y_i)$ as $m \rightarrow \infty$ for all $i = 1, \dots, n$, then, under some minor technical assumptions:*

$$\lim_{m \rightarrow \infty} \pi_{\hat{S}}^{(m)}(\theta | \mathbf{y}_n) = \pi_S(\theta | \mathbf{y}_n), \quad \forall \theta \in \Theta.$$

The above result is an extension of the one in Drovandi et al. [2015] for Bayesian inference with an auxiliary likelihood. Appendix A.4 gives the technical conditions explicitly (in Theorem 6) and proves the result.

Remark 4 (Properties of pseudo-marginal MCMC target). *Our results in Sec. 2.2 refer to the “exact” SR posterior π_S (Eq. 2), which, for finite m , is different from the pseudo-marginal MCMC target $\pi_{\hat{S}}^{(m)}$ in Eq. (5). Similarly to what was done in Frazier et al. [2022] for BSL, it could be possible to show asymptotic normality of $\pi_{\hat{S}}^{(m)}$ when both $n \rightarrow \infty$ and $m \rightarrow \infty$ at the same time; we leave this for future work. Additionally, while π_S satisfies Bayesian additivity (see Remark 1), $\pi_{\hat{S}}^{(m)}$ does not in general.*

2.4 Choice of w

In the generalized posterior distribution (Eq. 1), w represents the amount of information, with respect to prior information, one observation brings to the decision maker. For the standard Bayesian update, w is fixed to 1, which yields the optimal way to process information in a well-specified scenario [Zellner, 1988]. When the model is misspecified, some works have argued for the use of $w \neq 1$ in the standard Bayes update [Grünwald and Van Ommen, 2017, Holmes and Walker, 2017, Wu and Martin, 2020], and suggested criteria to select w . Many strategies for setting w for generalized Bayesian inference with a generic loss have also been proposed (see Bissiri et al. 2016, Syring and Martin 2019, Lyddon et al. 2019, Loaiza-Maya et al. 2021 and Matsubara et al. [2022a], among others). Most of these approaches are however inapplicable to the SR posterior for a LFI setting: Grünwald and Van Ommen [2017] and Syring and Martin [2019] require to obtain the posterior with different values of w , which is computationally costly; Holmes and Walker [2017], Lyddon et al. [2019] and Matsubara et al. [2022a] exploit derivatives of the loss with respect to θ ; Holmes and Walker [2017], Lyddon et al. [2019], Syring and Martin [2019] and Matsubara et al. [2022a] estimate the value of θ yielding the minimum expected loss, which cannot be obtained easily in LFI; finally, Loaiza-Maya et al. [2021] requires samples from the exact posterior.

We propose therefore a strategy to select w without repeated posterior inferences, and without relying on knowledge of the likelihood function. Notice that, as remarked by Bissiri et al. [2016]:

$$\log \underbrace{\left\{ \frac{\pi_S(\theta|y)}{\pi_S(\theta'|y)} / \frac{\pi(\theta)}{\pi(\theta')} \right\}}_{\text{BF}(\theta, \theta'; y)} = -w \{S(P_\theta, y) - S(P_{\theta'}, y)\} \iff w = -\frac{\log \text{BF}(\theta, \theta'; y)}{S(P_\theta, y) - S(P_{\theta'}, y)},$$

where $\text{BF}(\theta, \theta'; y)$ denotes the Bayes Factor of θ with respect to θ' for observation y . Therefore, w can be determined by fixing $\text{BF}(\theta, \theta'; y)$ for a single choice of θ, θ', y .

Assume the user has access to another posterior $\tilde{\pi}(\theta|y)$ which is obtained by means of a (in general misspecified) likelihood $\tilde{p}(y|\theta)$, with corresponding Bayes Factor $\widetilde{\text{BF}}$; for some θ, θ', y , setting:

$$w = -\frac{\log \widetilde{\text{BF}}(\theta, \theta'; y)}{S(P_\theta, y) - S(P_{\theta'}, y)},$$

would ensure $\widetilde{\text{BF}}(\theta, \theta'; y) = \text{BF}(\theta, \theta'; y)$. In practice, we have no reason to prefer a specific choice of (θ, θ') ; thus, we set w to be the median of $-\frac{\log \widetilde{\text{BF}}(\theta, \theta'; y)}{S(P_\theta, y) - S(P_{\theta'}, y)}$ over values of (θ, θ') sampled from the prior. The median (with respect to the mean) leads to a strategy which is robust to outliers in the computation of the above ratio. Additionally, if P_θ is an intractable-likelihood model, we estimate w by replacing $S(P_\theta, y)$ with $\hat{S}(\mathbf{x}_m^{(\theta)}, y)$, by generating data $\mathbf{x}_m^{(\theta)}$ for each considered values of θ .

Section 3.3. in Matsubara et al. [2022a] proposed to consider bootstrap datasets $\left\{ \mathbf{x}_i^{(b)} \right\}_{i=1}^n$, $b = 1, \dots, B$ and to find the loss minimizer $\theta_n^{(b)} = \arg \min_{\theta \in \Theta} D_n^{(b)}(\theta)$ for each of them; then, they find w by minimizing the Fisher divergence between the empirical measure of these minimizers and the generalized posterior for the same loss (you need the Fisher divergence to overcome the intractable normalizing constant of the generalised posterior). If the loss is twice differentiable with respect to the parameters, this leads to a closed form expression. This method is therefore embarrassingly parallelisable as the different minimizers can be computed independently. It is similar to Lyddon et al. [2019] in using the bootstrap samples, but it does not rely on asymptotic quantities and takes the prior into account. It is also somehow similar to Syring and Martin [2019] in trying to achieve some calibration; however it is much more computationally tractable. It is also similar to Jewson and Rossell [2021] in its use of the Fisher divergence.

Remark 5 (Posterior invariance with data rescaling). *Following on from Remark 2, we highlight here that the Kernel and Energy Score posteriors are invariant to an affine transformation of the data ($Z = a \cdot Y + b$ for $a, b \in \mathbb{R}$), albeit non-invariant to a generic transformation of the data coordinates. Specifically, the Kernel Score posterior with Gaussian kernel is invariant to such transformations with $w_Z = w_Y$, provided the kernel bandwidth is scaled too, while the Energy Score posterior is invariant when $w_Z \cdot a^\beta = w_Y$; both are ensured by our strategy for selecting w .*

3 Related approaches

As mentioned before, the popular LFI method Bayesian Synthetic Likelihood (BSL) corresponds to the SR posterior when using the Dawid-Sebastiani score (Eq. (3)). The Kernel Score posterior was already considered in Chérif-Abdellatif and Alquier [2020] under the name of MMD-Bayes. However, our work provides a broader perspective and a thorough evaluation of the Energy Score. Further, we study the robustness properties of the resulting posterior, which was not done in Chérif-Abdellatif and Alquier [2020], and consider more models in our simulation study. Moreover, in contrast to the variational approximation of Chérif-Abdellatif and Alquier [2020], we exploit pseudo-marginal MCMC.

Other two methods that fall under the SR framework are the semi-parametric BSL An et al. [2020] and the ratio-estimation methods Thomas et al. [2020]; we discuss these methods in Appendices C.2 and C.3.

Scoring Rules have been previously used to generalize Bayesian inference in Jewson et al. [2018], Giummolè et al. [2019], Loaiza-Maya et al. [2021]; no previous work, however, was concerned explicitly with LFI. Specifically, Giummolè et al. [2019] considered an update similar to ours, but fixed $w = 1$ and adjusted the parameter value (similarly to what was done in Pauli et al., 2011 and Ruli et al., 2016) so that the posterior has the same asymptotic covariance matrix as the frequentist minimum Scoring Rule estimator. Instead, Loaiza-Maya et al. [2021] considered a time-series setting in which the task is to learn about the parameter value which yields the best prediction, given the previous observations. Finally, Jewson et al. [2018] motivated Bayesian inference using general divergences (beyond the KL one which underpins standard Bayesian inference) to bypass the issues of standard Bayesian inference in an M-open setup (such as sensitivity to outliers). They discussed posteriors which employ estimators of the divergences from observed data; some of these estimators can be written using Scoring Rules (and thus satisfy Bayesian additivity), but they did not consider the Kernel and Energy score we study. Interestingly, Jewson et al. [2018] found divergences different from the KL to be robust to outliers, but no theory for why that is the case is provided.

A work released few weeks after the first version of the present manuscript [Matsubara et al., 2022b] investigates the generalized posterior obtained by using a Kernel Stein Discrepancy [Chwialkowski et al., 2016, Liu et al., 2016]. This posterior is shown to satisfy robustness and consistency properties, and is computationally convenient for doubly-intractable models (i.e., for which the likelihood is available, but only up to the normalizing constant). In contrast, our work focuses on models that do not have an explicit likelihood.

Finally, Dellaporta et al. [2022], published months after the release of the first version of our work, introduced a new LFI method which, similar to ours, enjoys outlier robustness and posterior consistency; however, their method is derived from the Bayesian non-parametric learning framework of Lyddon et al. [2018], Fong et al. [2019] rather than the generalized Bayesian posterior of Bissiri et al. [2016].

4 Simulation studies

We present here simulation studies to illustrate our approach. Precisely, we first study the concentration of different methods belonging to the SR posterior framework (the Energy and Kernel Scores, BSL and semiBSL) in Sec. 4.1; next, in Sec. 4.2, we consider the effect of outliers in the observed dataset; finally, in Sec. 4.3, we compare the posteriors obtained with different methods on two other benchmarks.

Throughout, we consider $\beta = 1$ in the Energy Score and define the Kernel Score via the Gaussian kernel with bandwidth set from simulations as illustrated in Appendix D. The LFI techniques are run using the ABCpy Python library [Dutta et al., 2021], while the PyMC3 library [Salvatier et al., 2016] is used to sample from the standard Bayes posterior when that is available (except for the M/G/1 example, where the custom strategy described in Shestopaloff and Neal, 2014 is exploited). Code for reproducing all results is available at https://github.com/LoryPack/GenBayes_LikelihoodFree_ScoringRules.

As mentioned in Sec. 2.3, we use pseudo-marginal MCMC to sample from an approximation of the

SR posterior. When the posterior is narrow, however, vanilla pseudo-marginal approaches [Andrieu et al., 2009] can have a “sticky” behavior, due to the variability in estimating the target. To reduce this issue, correlated pseudo-marginal MCMC has been suggested [Dahlin et al., 2015, Deligiannidis et al., 2018] and explicitly tested for BSL in Picchini et al. [2022]; the idea is to keep track of the random numbers used in simulating the model and reusing them for subsequent proposed parameter values. This correlates the target estimates at subsequent steps and reduces the chances of the chain getting stuck due to atypical random number draws. Specifically, the m simulations used in the posterior estimate (Eq. 4) are split in G groups; at each MCMC step, a new set of random numbers is proposed for the simulations in a randomly chosen group (alongside the proposed value for θ), and accepted or rejected in the standard way. Additionally, when Θ is bounded, we run the MCMC on a transformed unbounded space.

The algorithm described above is a valid pseudo-marginal MCMC and therefore targets Eq. (5) which, as mentioned before, is not the same as the original SR posterior defined in Eq. (2). However, as shown in Theorem 4, the former converges to the latter as $m \rightarrow \infty$. In Appendix F, we report experimental studies showing how, for m above some threshold (typically few hundreds), the MCMC target is roughly constant. On the contrary, very small values of m lead to unsatisfactory MCMC performance.

4.1 Concentration with the g-and-k model

Here, we study concentration of the posteriors belonging to the SR posterior framework with an increasing number of observations. We will consider both a well-specified and misspecified case; in the former, if a strictly proper SR (such as the Energy or Kernel Score) is used, the unique minimizer θ^* is the parameter value from which data are generated; the posterior will therefore concentrate on θ^* for large n (see Theorem 1). Indeed, in our experiments below we observe concentration for the Energy or Kernel Score posteriors but not for BSL and semiBSL. In the misspecified case, it is hard to verify analytically whether a unique θ^* exists, for both proper and strictly proper SRs; we proceed therefore by studying the behavior of the posterior with increasing n and deduce from this whether θ^* is unique or not for the different SRs. Additionally, we experience difficulties in sampling for BSL and semiBSL for large n , while this is not the case for the Energy and Kernel Score posteriors.

We consider the univariate g-and-k model and its multivariate extension; the univariate g-and-k distribution [Prangle, 2017] is defined in terms of the inverse of its cumulative distribution function F^{-1} . Given a quantile q , we define:

$$F^{-1}(q) = A + B \left[q + 0.8 \frac{1 - e^{-gz(q)}}{1 + e^{-gz(q)}} \right] (1 + z(q)^2)^k z(q),$$

where the parameters A, B, g, k are broadly associated to the location, scale, skewness and kurtosis of the distribution, and $z(q)$ denotes the q -th quantile of the standard normal distribution $\mathcal{N}(0, 1)$. Likelihood evaluation for this model is costly as it requires numerical inversion of F^{-1} ; instead, sampling is immediate by drawing $z \sim \mathcal{N}(0, 1)$ and inputing it in place of $z(q)$ in the expression above. A multivariate extension was first considered in the LFI literature in Drovandi and Pettitt [2011]; here we follow the setup of Jiang [2018]. Specifically, we consider drawing a multivariate normal $(Z^1, \dots, Z^5) \sim \mathcal{N}(0, \Sigma)$, where $\Sigma \in \mathbb{R}^{5 \times 5}$ has a sparse correlation structure: $\Sigma_{kk} = 1$, $\Sigma_{kl} = \rho$ for $|k - l| = 1$ and 0 otherwise; each component of Z is then transformed as in the univariate case. The sets of parameters are therefore $\theta = (A, B, g, k)$ for the univariate case and $\theta = (A, B, g, k, \rho)$ for the multivariate one. We use uniform priors on $[0, 4]^4$ for the univariate case and $[0, 4]^4 \times [-\sqrt{3}/3, \sqrt{3}/3]$ for the multivariate case.

In both setups, we perform inference with our SR methods, BSL and semiBSL (excluding semiBSL for the univariate g-and-k, as that is defined for multivariate distributions only) setting the number of simulations per parameter value to $m = 500$, $G = 500$ and run MCMC for 110000 steps, of which 10000 are burned in. We repeat this with 1, 5, 10, 15, 20 up to 100 observations spaced by 5.

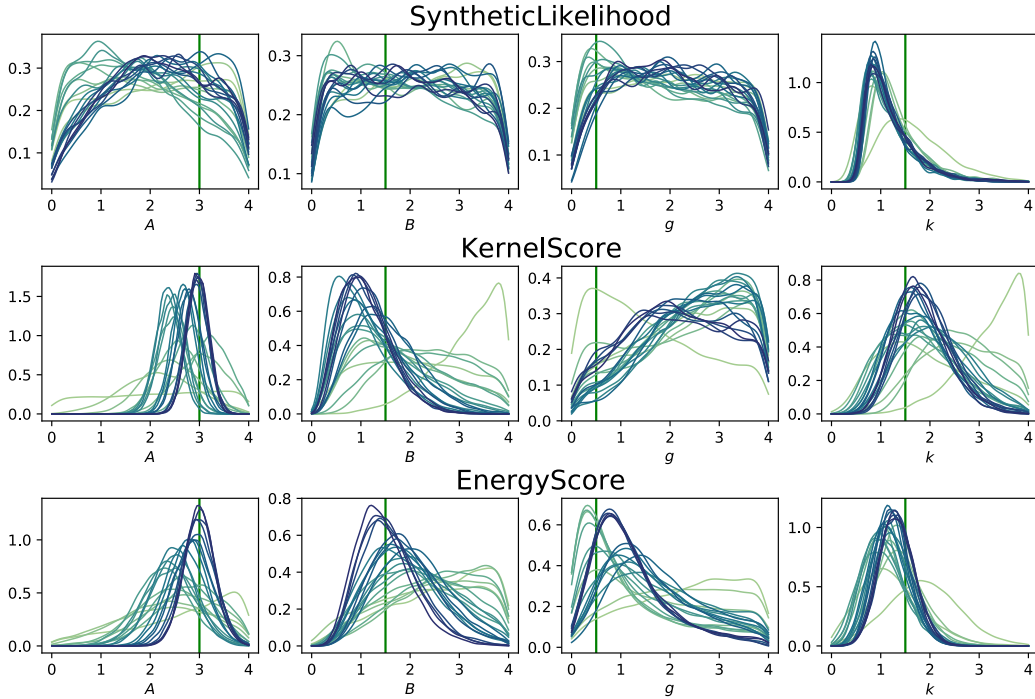


Figure 1: Marginal posterior distributions for the different parameters for the well-specified univariate g-and-k model, with increasing number of observations ($n = 1, 5, 10, 15, \dots, 100$). Darker (respectively lighter) colors denote a larger (smaller) number of observations. The densities are obtained by KDE on the MCMC output thinned by a factor 10. The Energy and Kernel Score posteriors concentrate around the true parameter value (green vertical line), while BSL does not.

4.1.1 Well-specified case

For both univariate and multivariate case, we consider synthetic observations generated from parameter values $A^* = 3$, $B^* = 1.5$, $g^* = 0.5$, $k^* = 1.5$ and $\rho^* = -0.3$ (notice ρ is not used in the univariate case). For the SR posteriors, we fix w by employing our suggested strategy with one single observation (Sec. 2.4), using as a reference BSL. The used values of w are reported in Appendix E.1.1, together with the proposal sizes for MCMC and the resulting acceptance rates.

For the univariate g-and-k, Fig. 1 reports the marginal posterior distributions for each parameter at different number of observations for the considered methods. With increasing n , the BSL posterior does not concentrate (except for the parameter k); the Energy Score posterior concentrates close to the true value for all parameters (green vertical line), while the Kernel Score posterior performs slightly worse, not being able to concentrate for the parameter g (albeit this may happen with an even larger n , which we did not consider here). The poor performance of BSL is due to violation of the underlying normality assumption (which is to say, the scoring rule used by BSL is not strictly proper for this example), while the concentration of the Energy and Kernel Score posteriors are in line with them being strictly proper SRs.

Similar results for the multivariate g-and-k are reported in Fig. 2. For this example, the MCMCs targeting the semiBSL and BSL posteriors do not converge beyond respectively 1 and 10 observations; those results are therefore deferred to Fig. 8 in Appendix. Instead, with the Kernel and Energy Scores we do not experience such a problem. The Energy Score concentrates well on the exact parameter value in this case too, while the Kernel Score is able to concentrate well for some parameters (g and k) and some concentration can be observed for ρ ; however, the Kernel Score posterior marginals for A and B are flatter and noisier (it may be that larger n leads to more concentrate posterior for A and B as well, but we did not research this further).

We investigate now the poor performance of semiBSL and BSL, by fixing $n = 20$ and running MCMC with 10 different initializations. The chains look “sticky” and, after a short transient, get stuck in different regions of Θ (see Fig. 9 in Appendix). This behavior can be explained by large variance

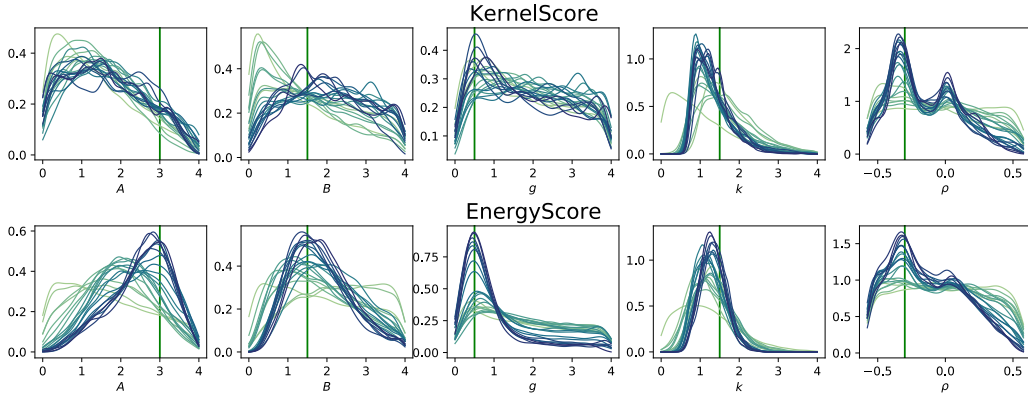


Figure 2: Marginal posterior distributions for the different parameters for the well-specified multivariate g -and- k model, with increasing number of observations ($n = 1, 5, 10, 15, \dots, 100$). Darker (respectively lighter) colors denote a larger (smaller) number of observations. The densities are obtained by KDE on the MCMC output thinned by a factor 10. The Energy Score posterior concentrates well around the true parameter value (green vertical line), with the Kernel Score one performing slightly worse.

in the estimates of the pseudo-marginal MCMC target. We repeat therefore the same experiments increasing the number of simulations m , as that decreases the variance of the target estimates (more details in Appendix E.1.2); larger values of m slightly increase the acceptance rate for BSL but do not almost change that for semiBSL. Additionally, while the BSL assumptions are unreasonable for this model, the multivariate g -and- k fulfills the assumptions underlying semiBSL: in fact, applying a one-to-one transformation to each component of a random vector does not change the copula structure, which is Gaussian in this case. It is therefore surprising that the performance of semiBSL degrades so rapidly when n increases.

4.1.2 Misspecified setup

Here, we consider as data generating process the Cauchy distribution, which has fatter tails than the g -and- k one. For the univariate case, the univariate Cauchy is used; for the multivariate case, the five components of each observation are drawn independently from the univariate Cauchy distribution (i.e., no correlation between components). For the SR posteriors, we use the values of w which were obtained with our strategy in the well-specified case; additional experimental details are reported in Appendix E.1.3.

For the univariate g -and- k , we report the marginal posteriors in Fig. 3. The Energy and Kernel Score posteriors concentrate on a similar parameter value; the BSL posterior concentrates as well (differently from the well-specified case), albeit on a slightly different parameter value (especially for B and k). Therefore, with this kind of misspecification, θ^* is unique both when using the strictly proper Kernel and Energy Scores, as well as the non-strictly proper Dawid–Sebastiani Score (corresponding to BSL).

To assess out-of-sample performance of the inferred posterior, we implement the following posterior predictive check: given draws from a posterior $\pi(\theta|\mathbf{y}_n)$, we generate simulations from the model for the corresponding parameter value, which are therefore samples from the posterior predictive:

$$p(y_{\text{new}}|\mathbf{y}_n) = \int p(y_{\text{new}}|\theta)\pi(\theta|\mathbf{y}_n)d\theta.$$

From these samples, we estimate the Energy and Kernel Score between the posterior predictive distribution and the observations \mathbf{y}_n , which assess how well the posterior predictive matches the original observation. Results for $n = 100$ observations are reported in Table 1: not surprisingly, the posterior predictive obtained for the Kernel Score posterior minimizes the Kernel Score, and similarly for the Energy Score. Indeed, the SR posterior concentrates on the parameter value which achieves the lowest expected SR.

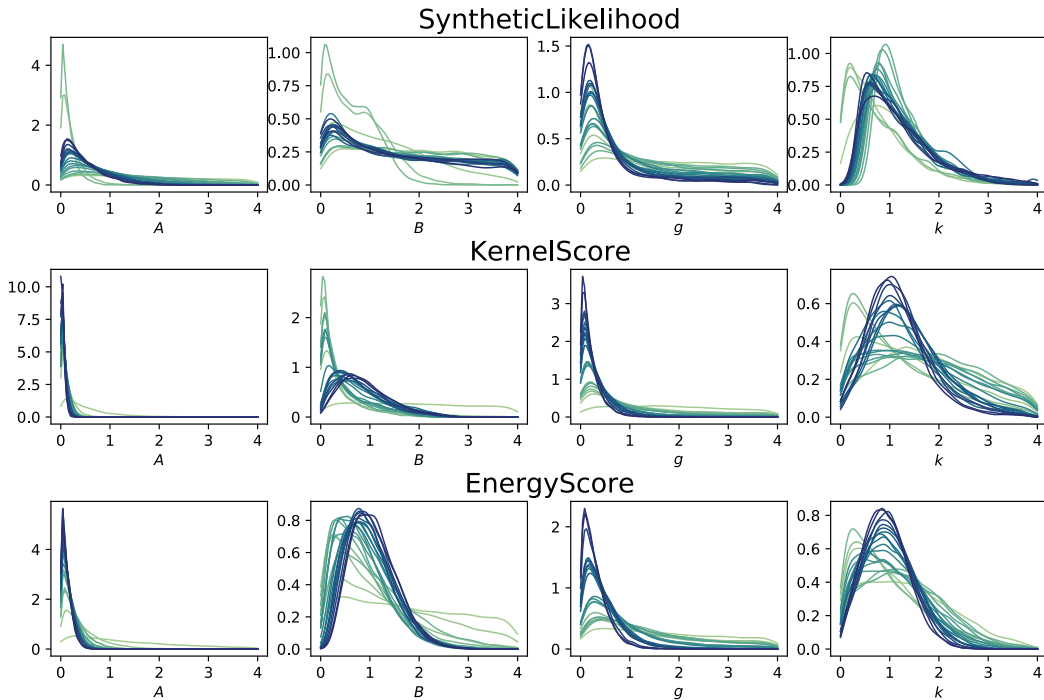


Figure 3: Marginal posterior distributions for the different parameters for the univariate g-and-k model, with increasing number of observations ($n = 1, 5, 10, 15, \dots, 100$) generated from the Cauchy distribution. Darker (respectively lighter) colors denote a larger (smaller) number of observations. The densities are obtained by KDE on the MCMC output thinned by a factor 10. The Energy and Kernel Score posteriors concentrate around the same parameter value, while BSL concentrates on slightly different one (specially for B and k).

For the multivariate g-and-k, we experienced the same issue with MCMC as in the well-specified case for BSL and semiBSL; therefore, we do not report those results. Marginals for the Energy and Kernel Score posteriors can be seen in Fig. 4; both posteriors concentrate for all parameters except for ρ (which describes correlation among different components in the observations, here absent). For the other parameters, the two methods concentrate on very similar parameter values, with slightly larger difference for k , for which the Kernel Score posterior does not concentrate very well.

As before, we report the posterior predictive check results for $n = 100$ observations in Table 1; here, the Energy Score posterior yields better posterior predictive according to both the Energy and Kernel Score. However, notice that the Kernel Score values for the two methods are very close.

	Misspecified univariate g-and-k			Misspecified multivariate g-and-k	
	BSL	Kernel Score	Energy Score	Kernel Score	Energy Score
Energy Score	49046.1828	41540.6385	36912.0548	243599.3863	227662.0929
Kernel Score	-7296.1355	-8088.4518	-8052.7533	-8856.8503	-8867.2898

Table 1: Posterior predictive check for misspecified univariate and multivariate g-and-k, for $n = 100$. Each column refers to a different posterior (column headline), while rows report the estimated Energy or Kernel Score between posterior predictive and the observations. Lower values are better, and bold denotes smallest values.

4.2 Robustness to outliers in normal location model

We now study the performance of the Kernel and Energy Score posteriors in presence of outliers. From our result in Theorem 3, we expect the Kernel Score to provide outlier robustness, which is empirically verified; however, even the Energy Score posterior shows some robustness.

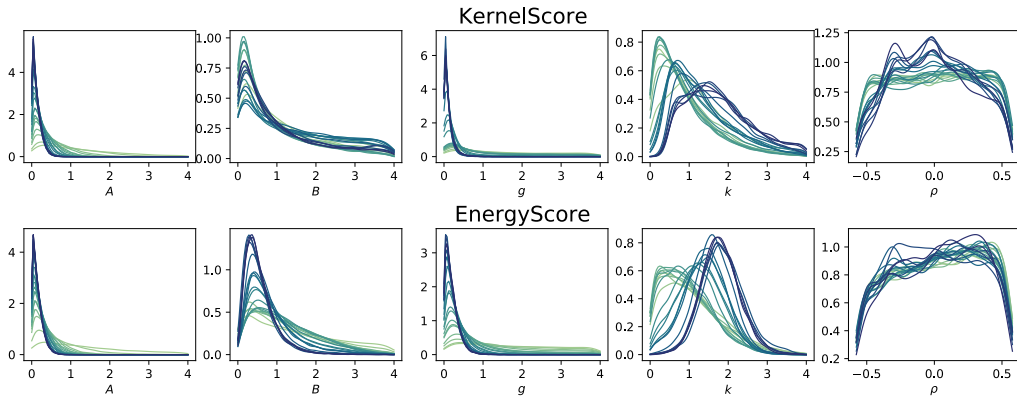


Figure 4: Marginal posterior distributions for the different parameters for the multivariate g-and-k model, with increasing number of observations ($n = 1, 5, 10, 15, \dots, 100$) generated from the Cauchy distribution. Darker (respectively lighter) colors denote a larger (smaller) number of observations. The densities are obtained by KDE on the MCMC output thinned by a factor 10. Both Energy and Kernel Score posteriors concentrate on a very similar parameter value, with slightly larger difference for k .

We consider a univariate normal model with fixed standard deviation $P_\theta = \mathcal{N}(\theta, 1)$. As was done in Matsubara et al. [2022b], we consider 100 observations, a proportion $1 - \epsilon$ of which is generated by P_θ with $\theta = 1$, while the remaining proportion ϵ is generated by $\mathcal{N}(z, 1)$ for some value of z . Therefore, ϵ and z control respectively the number and location of outliers. The prior distribution on θ is set to $\mathcal{N}(0, 1)$. To perform inference with our proposed SR posterior, we use $m = 500$, $G = 50$ and 60000 MCMC steps, of which 40000 are burned-in. Additionally, we perform standard Bayesian inference (as the likelihood is available here). For the SR posteriors, w is fixed in order to get approximately the same posterior variance as standard Bayes in the well-specified case ($\epsilon = 0$); values are reported in Appendix E.2, together with the proposal sizes for MCMC and the resulting acceptance rates.

We consider ϵ taking values in $(0, 0.1, 0.2)$ and z in $(1, 3, 5, 7, 10, 20)$; in Fig. 5, some results are shown. Results for all combinations of z and ϵ are available in Fig. 11 in Appendix. The Kernel Score posterior is highly robust with respect to outliers, while the Energy Score posterior performs slightly worse. As expected, the standard Bayes posterior shifts significantly when either ϵ or z are increased. We highlight that Theorem 3 only ensures robustness for small values of ϵ and all values of z for the Kernel Score posterior, which is in fact experimentally verified (the robustness result for the Energy Score posterior does not apply here as \mathcal{X} is unbounded); however, we find empirically that both SR posteriors are more robust than the standard Bayes one, when both z and ϵ are increased.

Finally, this example satisfies the BSL assumptions, which should therefore recover the standard Bayes posterior. However, our simulation results with BSL were unsatisfactory; specifically, BSL is able to reproduce the standard Bayes posterior when no outliers are present or when z is close to 1; in all other cases, MCMC does not converge and presents a sticky behavior, similar to what was already mentioned in Sec. 4.1. Further details on this are given in Appendix E.2.

4.3 Comparison of posteriors in MA(2) and M/G/1 models

We now show posteriors obtained with different methods for a single observation on the MA(2) and M/G/1 models [Marin et al., 2012, An et al., 2020]. We consider a well specified setting, and also report the standard Bayes posterior, which is available for these models. We stress again that the SR posterior does not approximate the standard Bayes posterior, but rather is defined as a generalized Bayesian update; for this reason, it is misleading to evaluate the SR posterior (with respect to, say, BSL) by assessing the mismatch from the standard Bayes one. Still, it is insightful to show the posterior obtained with our methods alongside the true posterior and its approximations BSL and semiBSL. With both models, we find that the SR posteriors are located on similar regions of the parameter space as the true posterior, but are centered on slightly different parameter values.

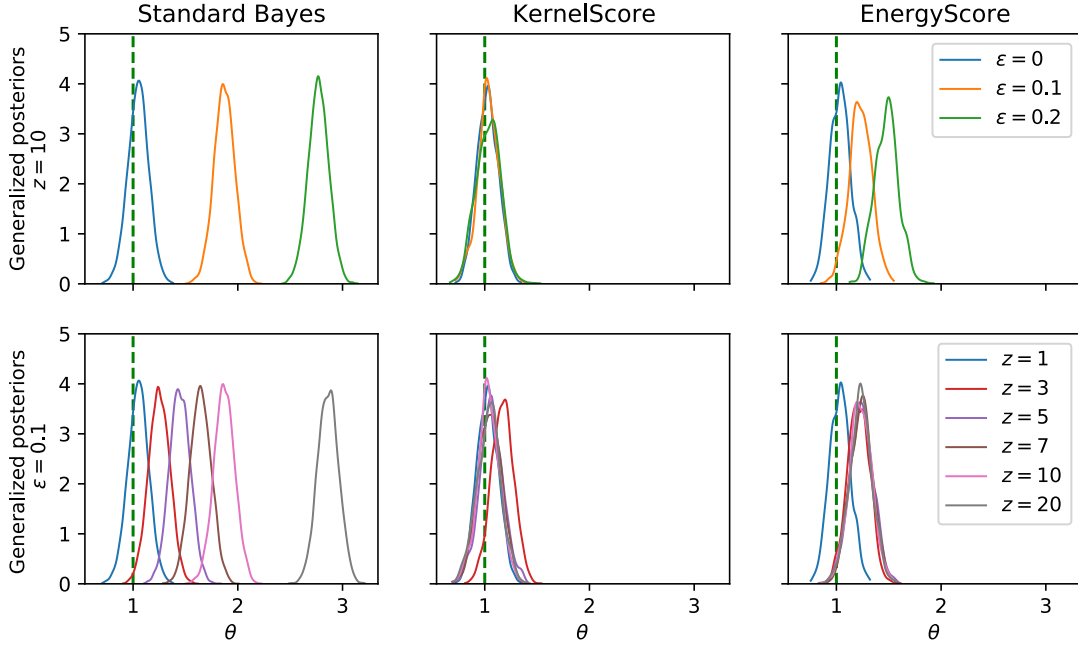


Figure 5: Posterior distribution for the misspecified normal location model, following experimental setup introduced in Matsubara et al. [2022b]. First row: fixed outliers location $z = 10$ and varying proportion ϵ ; second row: fixed outlier proportion ϵ , varying location z . From both rows, it can be seen that both Kernel and Energy score are more robust with respect to Standard Bayes. The densities are obtained by KDE on the MCMC output thinned by a factor 10.

4.3.1 The MA(2) model

The Moving Average model of order 2, or MA(2), is a time-series model for which simulation is easy and the likelihood is available in analytical form; it has 2 parameters $\theta = (\theta^1, \theta^2)$. Sampling from the model is achieved with the following recursive process:

$$x^1 = \xi^1, \quad x^2 = \xi^2 + \theta^1 \xi^1, \quad x^t = \xi^t + \theta^1 \xi^{t-1} + \theta^2 \xi^{t-2}, \quad t = 3, \dots, 50,$$

where ξ^t 's are i.i.d. samples from the standard normal distribution (recall here superscripts do not represent power but vector indices). The vector random variable $X \in \mathbb{R}^{50}$ has a multivariate normal distribution with sparse covariance matrix; therefore, this model satisfies the assumptions of both BSL and semiBSL. We set the prior distribution over the parameters to be uniform in the triangular region defined through the following inequalities: $-1 < \theta^2 < 1$, $\theta^1 + \theta^2 > -1$, $\theta^1 - \theta^2 < 1$. We consider an observation generated from $\theta^* = (0.6, 0.2)$; further, we use $G = 50$ groups, $m = 500$ simulations and 30000 MCMC steps, of which 10000 are burned-in, in order to sample from the different methods.

Exact posterior samples are obtained with MCMC using the exact MA(2) likelihood with 6 parallel chains with 20000 steps, of which 10000 are burned in. For the SR posteriors, we set w with our tuning strategy (Sec. 2.4). For all methods, we report the posteriors in Fig. 6. As expected, both BSL and semiBSL recover the true posterior well. The Energy Score and Kernel Score posterior perform similarly and are centered around the same parameter value as the true posterior, which is however narrower. This of course depends on the chosen value for w . To better understand the impact of that, we obtained the SR posteriors for different values of w : unsurprisingly, larger values of w lead to more concentrate posteriors (see Appendix E.3 for graphical illustrations and other experimental details).

4.3.2 The M/G/1 model

The M/G/1 model is a single-server queuing system with Poisson arrivals and general service times. Specifically, the distribution of the service time is Uniform in (θ^1, θ^2) and the interarrival times have exponential distribution with parameter θ^3 ; the set of parameters is therefore $\theta = (\theta^1, \theta^2, \theta^3)$. The observed data is the logarithm of the first 50 interdeparture times; as shown in An et al. [2020], the

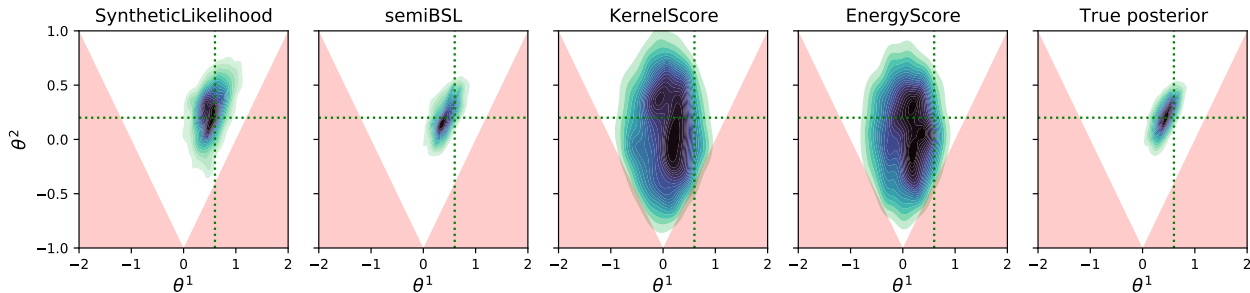


Figure 6: Contour plot for the posterior distributions for the MA(2) model, with darker colors denoting larger posterior density, and dotted line denoting true parameter value. The posterior densities are obtained by KDE on the MCMC output thinned by a factor 10. Here, the Energy and Kernel Score posteriors are similar and broader than the true posterior; notice that they do not approximate the true posterior but rather provide a general Bayesian update. BSL and semiBSL reproduce the true posterior well, as expected for this model. The prior distribution is uniform on the white triangular region.

distribution of simulated data does not resemble any common distributions; we give more details on the model and how to simulate from it in Appendix E.4.2. We set a Uniform prior on the region $[0, 10] \times [0, 10] \times [0, 1/3]$ for $(\theta^1, \theta^2 - \theta^1, \theta^3)$ and generate observations from $\theta^* = (1, 5, 0.2)$. We use $G = 50$ groups, $m = 1000$ simulations and 30000 MCMC steps, of which 10000 are burned-in, in order to sample from the different methods. To sample from the true posterior distribution, we employ the custom procedure described in Shestopaloff and Neal [2014].

Again, we set w for the SR posteriors with our tuning strategy (Sec. 2.4). For all methods, we report bivariate marginals of the posterior in Fig. 7. As already noticed in An et al. [2020], semiBSL is able to recover the true posterior quite well, while BSL performs worse. The Kernel and Energy Score posteriors are centered on slightly different parameter values from the true posterior, highlighting the fact that the SRs focus on different features in the data. However, we remark here that all posteriors are close in parameter space (notice that the axis in Fig. 7 do not span the full prior range). Notice that both the true posterior and the SR posteriors are guaranteed (by our Theorem 1) to concentrate on the exact parameter value as $n \rightarrow \infty$. For the used value of w , the Kernel Score posterior is much narrower than the Energy Score one. To better understand the impact of w , we provide result over a range of values in Appendix E.4, which also reports experimental details.

5 Conclusion

We introduced a way to perform Likelihood-Free Inference based on Generalized Bayesian Inference using Scoring Rules (SR), which includes popular LFI approaches [Price et al., 2018, Thomas et al., 2020, Chérif-Abdellatif and Alquier, 2020] as special cases. The SR posterior is computationally convenient for intractable-likelihood models when the chosen SR can be easily estimated from samples. For some SRs, it asymptotically recovers the exact parameter value in a well-specified setting.

A limitation of the SR posterior is that it does *not* approximate the standard Bayes posterior. This makes the interpretation of the posterior challenging, as it learns about the parameter value minimizing the expected SR. Importantly, outlier robustness is achieved as a consequence of this relaxation. However, our posterior may be of limited practical interest particularly when the model is well-specified and other LFI methods perform well.

However, we believe this connection between the Generalized Bayesian and LFI frameworks may inspire new research directions. Here, we suggest the following extensions:

- many more Scoring Rules beyond the Kernel and Energy score exist [Gneiting and Raftery, 2007, Dawid and Musio, 2014, Ziel and Berk, 2019], some of which may be successfully applied for LFI.

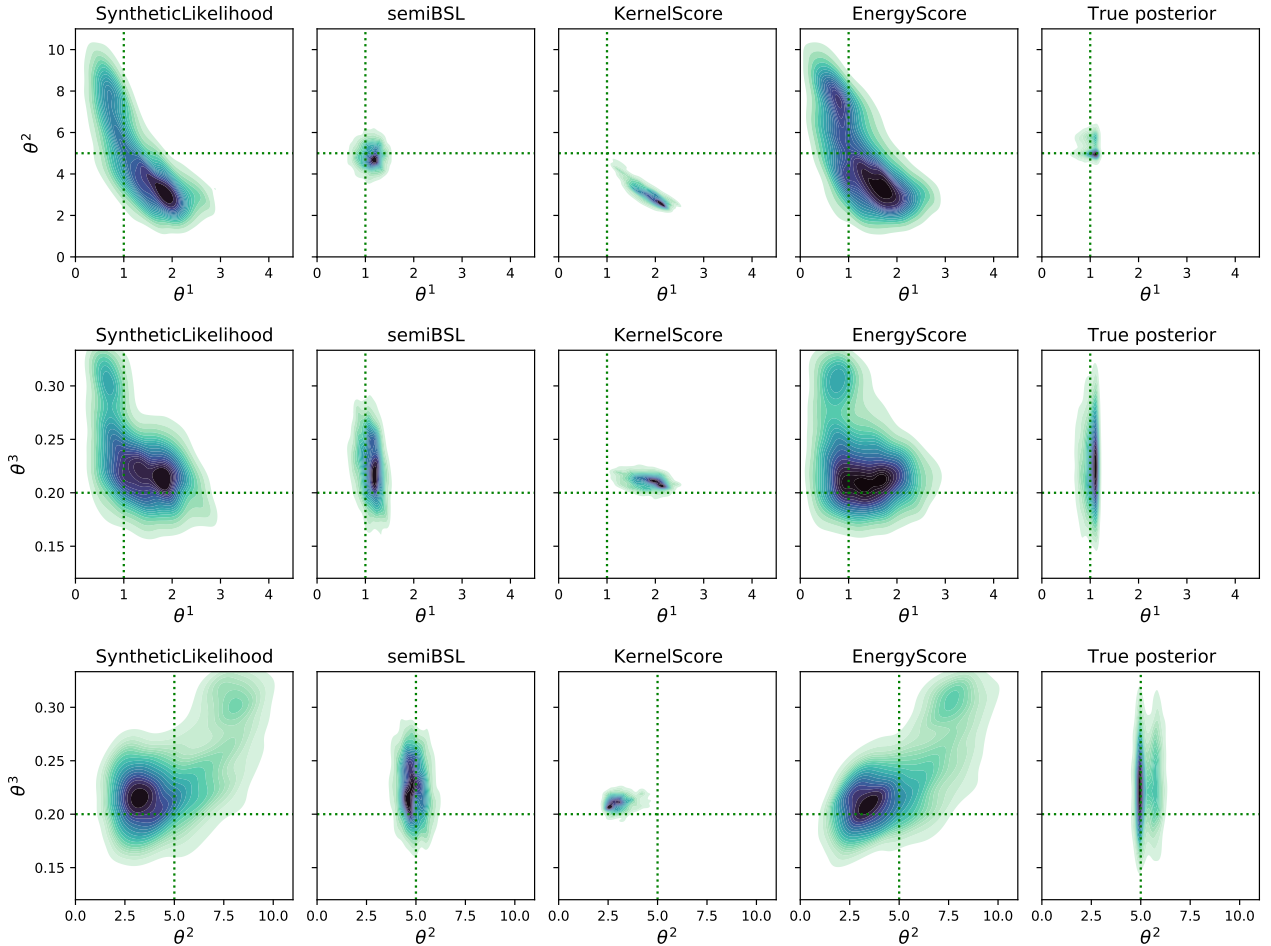


Figure 7: Posterior distributions for the M/G/1 model, with each row showing contour plots of the bivariate marginals for a different pair of parameters; darker colors denote larger posterior density, and dotted lines denote true parameter value. The posterior densities are obtained by KDE on the MCMC output thinned by a factor 10. All posteriors are close in parameter space (as the axis do not span the full prior range); however, the Energy and Kernel Score posteriors are different from each other as well as from the BSL and true posteriors. We remark that the SR posteriors do not approximate the true one but rather provide a general Bayesian update. As already noted in An et al. [2020], semiBSL recovers the true posterior well, while BSL performs worse.

- Our proposal to tune w (Sec. 2.4) has a low computational cost. However, the amplitude of the resulting generalized posterior is sometimes drastically different from that of the reference one. Moreover, the requirement of a reference posterior in the first place is an hindrance. Other strategies to tune w beyond our proposal in Sec. 2.4 could be designed. A promising strategy is that of Matsubara et al. [2022a], which could be efficiently implemented using automatic-differentiation libraries to compute the gradient of the scoring rule with respect to θ .
- Even with the correlated strategy, the pseudo-marginal MCMC used in our simulations had a low acceptance rate in some cases (Sec. 4.1). The SR posterior could be alternatively sampled using random weight particle filters [Del Moral et al., 2007, Fearnhead et al., 2008]. Alternatively, variational inference methods could be exploited, similarly to what was done in Ong et al. [2018], Chérif-Abdellatif and Alquier [2020] and Frazier et al. [2021b] for related methods.
- Generalized Bayesian approaches are often motivated with robustness arguments with respect to model misspecification, as the standard Bayes posterior may perform poorly in this setup [Bissiri et al., 2016, Jewson et al., 2018, Knoblauch et al., 2022]. Most LFI techniques are approximations of the true posterior, and as such are unsuited to a misspecified setup (albeit an

emerging literature investigating the effect of misspecification in LFI exists, see Ridgway [2017], Frazier et al. [2020b], Frazier [2020], Frazier et al. [2020a], Fujisawa et al. [2021]). In the present work, we studied robustness result to outlier contamination; however, it would be interesting to consider other forms of misspecification such as the distance in Prokhorov metric studied in [Briol et al., 2019] or the adversarial contamination method in Chérief-Abdellatif and Alquier [2022].

Acknowledgment

LP is supported by the EPSRC and MRC through the OxWaSP CDT programme (EP/L016710/1), which also funds the computational resources used to perform this work. RD is funded by EPSRC (grant nos. EP/V025899/1, EP/T017112/1) and NERC (grant no. NE/T00973X/1).

We thank Jeremias Knoblauch, François-Xavier Briol, Takuo Matsubara, Geoff Nicholls and Sebastian Schmon for valuable feedback and suggestions on earlier versions of this work. We also thank Alex Shestopaloff for providing code for exact MCMC for the M/G/1 model.

References

- Z. An, D. J. Nott, and C. Drovandi. Robust Bayesian synthetic likelihood via a semi-parametric approach. *Statistics and Computing*, 30(3):543–557, 2020.
- C. Andrieu, G. O. Roberts, et al. The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, 37(2):697–725, 2009.
- E. Bernton, P. E. Jacob, M. Gerber, and C. P. Robert. Approximate Bayesian computation with the Wasserstein distance. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 81(2):235–269, 2019. doi: <https://doi.org/10.1111/rssb.12312>. URL <https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/rssb.12312>.
- A. Bhattacharya, D. Pati, and Y. Yang. Bayesian fractional posteriors. *The Annals of Statistics*, 47(1):39–66, 2019.
- P. Billingsley. *Convergence of probability measures*. John Wiley & Sons, 2nd edition, 1999.
- P. G. Bissiri, C. C. Holmes, and S. G. Walker. A general framework for updating belief distributions. *Journal of the Royal Statistical Society. Series B, Statistical methodology*, 78(5):1103, 2016.
- F.-X. Briol, A. Barp, A. B. Duncan, and M. Girolami. Statistical inference for generative models with maximum mean discrepancy. *arXiv preprint arXiv:1906.05944*, 2019.
- B.-E. Chérief-Abdellatif and P. Alquier. MMD-Bayes: Robust Bayesian estimation via maximum mean discrepancy. In *Symposium on Advances in Approximate Bayesian Inference*, pages 1–21. PMLR, 2020.
- B.-E. Chérief-Abdellatif and P. Alquier. Finite sample properties of parametric MMD estimation: robustness to misspecification and dependence. *Bernoulli*, 28(1):181–213, 2022.
- K. Chwialkowski, H. Strathmann, and A. Gretton. A kernel test of goodness of fit. In *International conference on machine learning*, pages 2606–2615. PMLR, 2016.
- J. Dahlin, F. Lindsten, J. Kronander, and T. B. Schön. Accelerating pseudo-marginal Metropolis-Hastings by correlating auxiliary variables. *arXiv preprint arXiv:1511.05483*, 2015.
- A. P. Dawid and M. Musio. Theory and applications of proper scoring rules. *Metron*, 72(2):169–183, 2014.
- P. Del Moral, A. Doucet, and A. Jasra. Sequential Monte Carlo for Bayesian computation. *Bayesian statistics*, 8(1):34, 2007.
- G. Deligiannidis, A. Doucet, and M. K. Pitt. The correlated pseudomarginal method. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 80(5):839–870, 2018.
- C. Dellaporta, J. Knoblauch, T. Damoulas, and F.-X. Briol. Robust bayesian inference for simulator-based models via the MMD posterior bootstrap. In *International Conference on Artificial Intelligence and Statistics*, pages 943–970. PMLR, 2022.
- C. C. Drovandi and A. N. Pettitt. Likelihood-free Bayesian estimation of multivariate quantile distributions. *Computational Statistics & Data Analysis*, 55(9):2541–2556, 2011.

- C. C. Drovandi, A. N. Pettitt, and A. Lee. Bayesian indirect inference using a parametric auxiliary model. *Statistical Science*, 30(1):72–95, 2015.
- R. Dutta, M. Schoengens, L. Pacchiardi, A. Ummadisingu, N. Widmer, P. Künzli, J.-P. Onnela, and A. Mira. ABCpy: A high-performance computing perspective to approximate bayesian computation. *Journal of Statistical Software*, 100(7): 1–38, 2021. doi: 10.18637/jss.v100.i07. URL <https://www.jstatsoft.org/index.php/jss/article/view/v100i07>.
- P. Fearnhead, O. Papaspiliopoulos, and G. O. Roberts. Particle filters for partially observed diffusions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 70(4):755–777, 2008.
- E. Fong, S. Lyddon, and C. Holmes. Scalable nonparametric sampling from multimodal posteriors with the posterior bootstrap. In *International Conference on Machine Learning*, pages 1952–1962. PMLR, 2019.
- D. T. Frazier. Robust and efficient approximate Bayesian computation: A minimum distance approach. *arXiv preprint arXiv:2006.14126*, 2020.
- D. T. Frazier, C. Drovandi, and R. Loaiza-Maya. Robust approximate Bayesian computation: An adjustment approach. *arXiv preprint arXiv:2008.04099*, 2020a.
- D. T. Frazier, C. P. Robert, and J. Rousseau. Model misspecification in approximate bayesian computation: consequences and diagnostics. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 82(2):421–444, 2020b. doi: <https://doi.org/10.1111/rssb.12356>. URL <https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/rssb.12356>.
- D. T. Frazier, C. Drovandi, and D. J. Nott. Synthetic likelihood in misspecified models: Consequences and corrections. *arXiv preprint arXiv:2104.03436*, 2021a.
- D. T. Frazier, R. Loaiza-Maya, G. M. Martin, and B. Koo. Loss-based variational Bayes prediction. *arXiv preprint arXiv:2104.14054*, 2021b.
- D. T. Frazier, D. J. Nott, C. Drovandi, and R. Kohn. Bayesian inference using synthetic likelihood: Asymptotics and adjustments. *Journal of the American Statistical Association*, 0(0):1–12, 2022. doi: 10.1080/01621459.2022.2086132. URL <https://doi.org/10.1080/01621459.2022.2086132>.
- M. Fujisawa, T. Teshima, I. Sato, and M. Sugiyama. γ -ABC: Outlier-robust approximate Bayesian computation based on a robust divergence estimator. In *International Conference on Artificial Intelligence and Statistics*, pages 1783–1791. PMLR, 2021.
- A. Ghosh and A. Basu. Robust Bayes estimation using the density power divergence. *Annals of the Institute of Statistical Mathematics*, 68(2):413–437, 2016.
- J. K. Ghosh and R. Ramamoorthi. *Bayesian nonparametrics*. Springer Science & Business Media, 2003.
- F. Giummolè, V. Mameli, E. Ruli, and L. Ventura. Objective Bayesian inference with proper scoring rules. *Test*, 28(3): 728–755, 2019.
- T. Gneiting and A. E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.
- P. Grünwald and T. Van Ommen. Inconsistency of Bayesian inference for misspecified linear models, and a proposal for repairing it. *Bayesian Analysis*, 12(4):1069–1103, 2017.
- C. Holmes and S. Walker. Assigning a value to a power likelihood in a general Bayesian model. *Biometrika*, 104(2): 497–503, 2017.
- J. Jewson and D. Rossell. General bayesian loss function selection and the use of improper models. *arXiv preprint arXiv:2106.01214*, 2021.
- J. Jewson, J. Q. Smith, and C. Holmes. Principles of Bayesian inference using general divergence criteria. *Entropy*, 20(6):442, 2018.
- B. Jiang. Approximate Bayesian computation with Kullback-Leibler divergence as data discrepancy. In *International Conference on Artificial Intelligence and Statistics*, pages 1711–1721, 2018.
- J. Knoblauch, J. Jewson, and T. Damoulas. An optimization-centric view on bayes’ rule: Reviewing and generalizing variational inference. *Journal of Machine Learning Research*, 23(132):1–109, 2022.
- J. Lintusaari, M. U. Gutmann, R. Dutta, S. Kaski, and J. Corander. Fundamentals and recent developments in approximate Bayesian computation. *Systematic Biology*, 66(1):e66–e82, 2017. ISSN 1076836X. doi: 10.1093/sysbio/syw077. URL <https://doi.org/10.1093/sysbio/syw077>.

- Q. Liu, J. Lee, and M. Jordan. A kernelized Stein discrepancy for goodness-of-fit tests. In *International conference on machine learning*, pages 276–284. PMLR, 2016.
- R. Loaiza-Maya, G. M. Martin, and D. T. Frazier. Focused Bayesian prediction. *Journal of Applied Econometrics*, 36(5):517–543, 2021.
- S. Lyddon, S. Walker, and C. C. Holmes. Nonparametric learning from Bayesian models with randomized objective functions. *Advances in Neural Information Processing Systems*, 31, 2018.
- S. Lyddon, C. Holmes, and S. Walker. General Bayesian updating and the loss-likelihood bootstrap. *Biometrika*, 106(2):465–478, 2019.
- J.-M. Marin, P. Pudlo, C. P. Robert, and R. J. Ryder. Approximate Bayesian computational methods. *Statistics and Computing*, 22(6):1167–1180, 2012.
- T. Matsubara, J. Knoblauch, F.-X. Briol, C. Oates, et al. Generalised bayesian inference for discrete intractable likelihood. *arXiv preprint arXiv:2206.08420*, 2022a.
- T. Matsubara, J. Knoblauch, F.-X. Briol, and C. J. Oates. Robust generalised Bayesian inference for intractable likelihoods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 84(3):997–1022, 2022b. doi: <https://doi.org/10.1111/rssb.12500>. URL <https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/rssb.12500>.
- C. McDiarmid. On the method of bounded differences. *Surveys in combinatorics*, 141(1):148–188, 1989.
- J. W. Miller. Asymptotic normality, concentration, and coverage of generalized posteriors. *Journal of Machine Learning Research*, 22(168):1–53, 2021.
- B. Nelson. *Foundations and methods of stochastic simulation: a first course*. Springer Science & Business Media, 2013.
- H. D. Nguyen, J. Arbel, H. Lü, and F. Forbes. Approximate Bayesian computation via the energy statistic. *IEEE Access*, 8:131683–131698, 2020.
- V. M.-H. Ong, D. J. Nott, M.-N. Tran, S. A. Sisson, and C. C. Drovandi. Likelihood-free inference in high dimensions with synthetic likelihood. *Computational Statistics & Data Analysis*, 128:271–291, 2018.
- M. Park, W. Jitkrittum, and D. Sejdinovic. K2-ABC: Approximate Bayesian computation with kernel embeddings. In *Artificial Intelligence and Statistics*, 2016.
- F. Pauli, W. Racugno, and L. Ventura. Bayesian composite marginal likelihoods. *Statistica Sinica*, pages 149–164, 2011.
- U. Picchini, U. Simola, and J. Corander. Sequentially Guided MCMC Proposals for Synthetic Likelihoods and Correlated Synthetic Likelihoods. *Bayesian Analysis*, pages 1 – 31, 2022. doi: 10.1214/22-BA1305. URL <https://doi.org/10.1214/22-BA1305>.
- D. Prangle. gk: An R package for the g-and-k and generalised g-and-h distributions. *arXiv preprint arXiv:1706.06889*, 2017.
- L. F. Price, C. C. Drovandi, A. Lee, and D. J. Nott. Bayesian synthetic likelihood. *Journal of Computational and Graphical Statistics*, 27(1):1–11, 2018.
- J. Ridgway. Probably approximate Bayesian computation: nonasymptotic convergence of ABC under misspecification. *arXiv preprint arXiv:1707.05987*, 2017.
- M. L. Rizzo and G. J. Székely. Energy distance. *Wiley interdisciplinary reviews: Computational statistics*, 8(1):27–38, 2016.
- E. Ruli, N. Sartori, and L. Ventura. Approximate Bayesian computation with composite score functions. *Statistics and Computing*, 26(3):679–692, 2016.
- J. Salvatier, T. V. Wiecki, and C. Fonnesbeck. Probabilistic programming in Python using PyMC3. *PeerJ Computer Science*, 2:e55, 2016.
- H. Scheffé. A useful convergence theorem for probability distributions. *The Annals of Mathematical Statistics*, 18(3):434–438, 1947.
- A. Y. Shestopaloff and R. M. Neal. On Bayesian inference for the M/G/1 queue with efficient MCMC sampling. *arXiv preprint arXiv:1401.5548*, 2014.
- N. Syring and R. Martin. Calibrating general posterior credible regions. *Biometrika*, 106(2):479–486, 2019.

- O. Thomas, R. Dutta, J. Corander, S. Kaski, M. U. Gutmann, et al. Likelihood-free inference by ratio estimation. *Bayesian Analysis*, 2020.
- P.-S. Wu and R. Martin. A comparison of learning rate selection methods in generalized bayesian inference. *arXiv preprint arXiv:2012.11349*, 2020.
- A. Zellner. Optimal information processing and Bayes’s theorem. *The American Statistician*, 42(4):278–280, 1988.
- F. Ziel and K. Berk. Multivariate forecasting evaluation: On sensitive and strictly proper scoring rules. *arXiv preprint arXiv:1910.07325*, 2019.

A Proofs of theoretical results

A.1 Proof and more details on Theorem 1

A.1.1 Discussion and comparison with related results

Discussion on assumptions The uniqueness of the minimizer of the expected scoring rule θ^* (in Assumption **A1**) is satisfied in a well specified setup if S is a strictly proper scoring rule (in which case $P_{\theta^*} = P_0$). If the model class is not well specified, a strictly proper S does not guarantee the minimizer to be unique (as in fact there may be pathological cases where multiple minimizers exist).

Additionally, it may be the case that, for a specific P_0 and misspecified model class P_θ , the minimizer of $S(P_\theta, P_0)$ is unique even if S is not strictly proper; in fact, in general, being not strictly proper means that there exist at least one pair of values $\theta^{(1)}, \theta^{(2)}$ for which $S(P_{\theta^{(1)}}, P_{\theta^{(2)}}) = S(P_{\theta^{(1)}}, P_{\theta^{(1)}})$, but it may be that the $\arg \min_{\theta \in \Theta} S(P_\theta, P_0)$ is unique for that specific choice of P_0 , as the minimizer is in a region of the parameter space for which there are no other parameter values which lead to the same value of the scoring rule.

Our proof below builds on Theorem 5 in Miller [2021]; to do so, we require regularity conditions on the third order derivatives of the SR (in Assumptions **A2** or, alternatively, **A2bis** below). It may be possible however to relax these assumptions to assuming $\theta \rightarrow S(P_\theta, y)$ can be locally written as a quadratic function of θ , with bounded coefficient for the third order term; this is usually called a Locally Asymptotically Normal (LAN) condition. With such, it would be possible to apply Theorem 4 in Miller [2021] (more general than Theorem 5) to show our result.

Related results Appendix A in Loaiza-Maya et al. [2021] provides a result which holds with non-i.i.d. (independent and identically distributed) data, with a generalized posterior based on Scoring Rules with a similar formulation to ours. Additionally, they replace our assumptions on differentiability (which ensure the existence of the Taylor series expansion in the proof below) with assuming the difference of the cumulative scoring rules have a LAN form. Finally, they only show convergence in probability.

Another related result can be found in Matsubara et al. [2022b], which studies a generalized posterior based on Kernel Stein Discrepancy; similarly to us, they build on Miller [2021], and provide almost sure convergence. However, they exploit Theorem 4 in Miller [2021], while we rely on Theorem 5. In Matsubara et al. [2022b], third order differentiability conditions are assumed, analogously to our Assumption **A2**. The remaining assumptions in Matsubara et al. [2022b] are similar to ours, including prior continuity and uniqueness of the minimizer θ^* .

Finally, we remark that, if multiple minimizers of $S(P_\theta, P_0)$ exist (in finite number), it may be possible to obtain an asymptotic *fractional* normality result, which ensures the SR posterior converges to a mixture of normal distributions centered in the different minimizers; see for instance [Frazier et al., 2021a] for an example of such results in the setting of BSL. We leave this for future work.

A.1.2 Alternative statements and proof

First, let us reproduce Theorem 5 in Miller [2021], on which our proof is based, for ease of reference. Here, convergence and boundedness for vectors $v \in \mathbb{R}^p$, matrices $M \in \mathbb{R}^{p \times p}$ and tensors $T \in \mathbb{R}^{p \times p \times p}$ are

defined with respect to Euclidean-Frobenius norms, that is: $|v| = \left(\sum_j v_j^2\right)^{1/2}$, $\|M\| = \left(\sum_{jk} M_{jk}^2\right)^{1/2}$ and $\|T\| = \left(\sum_{jkl} T_{jkl}^2\right)^{1/2}$.

Theorem 5 (Theorem 5 in Miller [2021]). *Let $\Theta \subseteq \mathbb{R}^p$. Let $E \subseteq \Theta$ be open (in \mathbb{R}^p) and bounded. Fix $\theta^* \in E$ and let $\pi : \Theta \rightarrow \mathbb{R}$ be a probability density with respect to Lebesgue measure. Consider the following family of distributions:*

$$\pi_n(\theta) = \frac{\pi(\theta) \exp(-nf_n(\theta))}{\int_{\Theta} \pi(\theta) \exp(-nf_n(\theta))},$$

where $f_n : \Theta \rightarrow \mathbb{R}$ is a family of functions. Under the following conditions:

C1 π is continuous at θ^* and $\pi(\theta^*) > 0$,

C2 f_n have continuous third derivatives in E ,

C3 $f_n \rightarrow f$ pointwise for some $f : \Theta \rightarrow \mathbb{R}$,

C4 $f''(\theta^*)$ is positive definite,

C5 f_n''' is uniformly bounded in E ,

C6 Either one of the following holds:

- (a) for some compact $K \subseteq E$, with θ^* in the interior of K , $f(\theta) > f(\theta^*) \forall \theta \in K \setminus \{\theta^*\}$ and $\liminf_n \inf_{\theta \in \Theta \setminus K} f_n(\theta) > f(\theta^*)$, or
- (b) each f_n is convex and $f'(\theta^*) = 0$;

then, there is a sequence $\theta_n \rightarrow \theta^*$ such that $f_n'(\theta_n) = 0$ for all n sufficiently large, $f_n(\theta_n) \rightarrow f(\theta^*)$ and, letting q_n be the density of $\sqrt{n}(\theta - \theta_n)$ when $\theta \sim \pi_n$:

$$\int |q_n(s) - \mathcal{N}(s|0, (f''(\theta^*))^{-1})| ds \rightarrow 0 \text{ as } n \rightarrow \infty,$$

that is, q_n converges to $\mathcal{N}(0, (f''(\theta^*))^{-1})$ in total variation. Additionally, **C6b** implies **C6a** under the other conditions.

Notice that Theorem 5 considers deterministic f_n and f . In order to prove our result, therefore, we will verify the different conditions hold almost surely, which implies almost sure convergence.

Besides the assumptions considered in the main text (i.e. **A1-A4**), it is possible to prove the asymptotic normality result in Theorem 1 under alternative sets of assumptions. For this reason, we introduce the following:

A2bis The parameter space Θ is open, convex, and bounded; the function $\theta \rightarrow S(P_\theta, y)$, for any fixed $y \in \mathcal{X}$, can be extended to the closure $\bar{\Theta}$. Let us denote $S'''(P_\theta, y)_{jkl} = \frac{\partial^3}{\partial \theta_j \partial \theta_k \partial \theta_l} S(P_\theta, Y)$. For all $j, k, l \in \{1, \dots, d\}$:

- $\theta \rightarrow S'''(P_\theta, y)_{jkl}$ is continuous in Θ and exists in $\bar{\Theta}$ for any fixed $y \in \mathcal{X}$,
- $y \rightarrow S'''(P_\theta, y)_{jkl}$ is measurable for any fixed $\theta \in \bar{\Theta}$,
- $\mathbb{E}_{P_0} \sup_{\theta \in \bar{\Theta}} |S'''(P_\theta, y)_{jkl}| < \infty$.

A3bis For each $y \in \mathcal{X}$, the function: $\theta \rightarrow S(P_\theta, y)$ is convex.

The following extended form of Theorem 1 includes the formulation in the main text as well as two alternative sets of assumptions.

Theorem 1 - extended version. *Under either one of the following sets of assumptions:*

1. **A1, A2, A3, A4** (the set originally used in the main text),
2. **A1, A2, A3bis, A4**,
3. **A1, A2bis, A4**,

the statement of Theorem 1 in the main text holds.

We next move to proving our result.

Assumption **A2** is used in the original set of assumptions to ensure the second part of Condition **C6a** holds almost surely. Under set of assumptions 2, convexity of the scoring rules (Assumption **A3bis**) is used to show Condition **C6b**; alternatively, with set of assumptions 3, the constraints on Θ are used to imply the second part of Condition **C6a** with probability 1 using Theorem 7 in Miller [2021]. In both cases, Assumption **A2** is not explicitly needed anymore – as in fact it is implied by the remaining assumptions. However, we are unable to remove Assumption **A2** under no constraints on Θ or the convexity of $\theta \rightarrow S(P_\theta, y)$.

We now give our proof:

Proof of Theorem 1 - extended version. In order to obtain our result, we identify

$$f_n(\theta) = \frac{1}{n} \sum_{i=1}^n S(P_\theta, Y_i), \quad f(\theta) = S(P_\theta, P_0);$$

this implies that f_n is now a random quantity: as such, we show the conditions for Theorem 5 hold almost surely over the stochasticity induced by \mathbf{Y}_n .

All three sets of assumptions include Assumptions **A1** and **A4**; therefore, under all three sets of assumptions:

- **C1** corresponds to our Assumption **A4**,
- **C3** holds almost surely thanks to the strong law of large numbers, as $S(P_\theta, P_0)$ is finite $\forall \theta \in \Theta$ by Assumption **A1**,
- **C4** is implied by Assumption **A1**.

Therefore, we are left with establishing **C2**, **C5** and **C6** separately for the different sets of assumptions.

Set of assumptions 1 (used in main text):

- **C2** is implied by our Assumption **A2** to hold with probability 1 for all n .
- In order to show **C5**, we proceed in similar manner as in Theorem 13 in Miller [2021]. For any $j, k, l \in \{1, \dots, d\}$, Assumption **A2** implies that, with probability 1, $f_n'''(\theta)_{jkl} = \frac{1}{n} \sum_{i=1}^n S'''(P_\theta, Y_i)_{jkl}$ is uniformly bounded on \bar{E} by the uniform law of large number (Theorem 1.3.3 in Ghosh and Ramamoorthi 2003). Letting $C_{jkl}(Y_1, Y_2, \dots)$ be such a uniform bound for each j, k, l , we have that with probability 1, for all $n \in \mathbb{N}$, $\theta \in \bar{E}$, $\|f_n'''(\theta)\|^2 = \sum_{jkl} (f_n'''(\theta)_{jkl})^2 \leq \sum_{jkl} C_{jkl}(Y_1, Y_2, \dots) < \infty$. Thus, $f_n'''(\theta)$ is almost surely uniformly bounded on \bar{E} , and hence on E .
- The first part of **C6a** is implied by Assumption **A1**, while the second part holds almost surely by Assumption **A3**.

Set of assumptions 2: The only difference here is that we replace Assumption **A3** with the stronger convexity Assumption **A3bis**; therefore, **C2** and **C5** are shown in the same way as with set of assumptions 1.

Next, consider **C6b**: the first part is implied to hold with probability 1 for all n by Assumption **A3bis**, as the sum of convex functions is convex. The second part is instead consequence of θ^* being a stationary point of f due to Assumption **A1**, and of $f'(\theta^*)$ existing due to **C4**.

Set of assumptions 3: Under these assumptions, we fix $E = \Theta$ in the statement of Theorem 5, as we consider Θ to be open and bounded. With that, we can exploit Assumption **A2bis** and follow the same steps as with set of assumptions 1 to show that, over Θ , **C2** and **C5** hold with probability 1.

The first part of **C6a** is implied by Assumption **A1**, for any choice of K ; it now remains to show the second part. First, Theorem 7 in Miller [2021] implies that $f_n \rightarrow f$ uniformly almost surely, as in fact f_n have continuous third derivatives by **C2**, f_n''' is uniformly bounded with probability 1 by **C5**, and $f_n \rightarrow f$ with probability 1 due to **C3** holding with probability 1.

Therefore, with probability 1:

$$\liminf_n \inf_{\theta \in \Theta \setminus K} f_n(\theta) = \inf_{\theta \in \Theta \setminus K} f(\theta) > \inf_{\theta \in \Theta} f(\theta) = f(\theta^*),$$

where the first equality is due to uniform convergence allowing to “swap” the infimum and the limit. \square

A.2 Proof of Theorem 2

First, we prove a finite sample generalization bound which is valid for the generalized Bayes posterior with a generic loss, assuming a concentration property and prior mass condition. Next, we will use this Lemma to prove Theorem 2 reported in the main body of the paper (in Section 2.2.2), by first proving concentration results for Kernel and Energy Scores.

We remark that our Theorem 2 is similar to Theorem 1 in Matsubara et al. [2022b] for the kernel Stein Discrepancy (KSD) posterior, but provides a tighter probability bound. As the kernel used in KSD is unbounded, in fact, Matsubara et al. [2022b] had to rely on weaker results with respect to the ones used to prove Theorem 2. With a similar approach, a result for unbounded k or \mathcal{X} may be obtained in our case; we leave this for future exploration.

A.2.1 Lemma for generalized Bayes posterior with generic loss

In this Subsection, we consider the following generalized Bayes posterior:

$$\pi_L(\theta | \mathbf{y}_n) \propto \pi(\theta) \exp \{-wnL(\theta, \mathbf{y}_n)\}, \quad (6)$$

where $\mathbf{y}_n = \{y_i\}_{i=1}^n$ denote the observations, π is the prior and $L(\theta, \mathbf{y}_n)$ is a generic loss function (which does not need to be additive in y_i). Here, the SR posterior for the scoring rule S corresponds to choosing:

$$L(\theta, \mathbf{y}_n) = \frac{1}{n} \sum_{i=1}^n S(P_\theta, y_i).$$

First, we state a result concerning this form of the posterior which we will use later (taken from Knoblauch et al. 2022), and reproduce here the proof for convenience:

Lemma 1 (Theorem 1 in Knoblauch et al. [2022]). *Provided that $\int_{\Theta} \pi(\theta) \exp \{-wnL(\theta, \mathbf{y}_n)\} d\theta < \infty$, $\pi_L(\cdot | \mathbf{y}_n)$ in Eq. (6) can be written as the solution to a variational problem:*

$$\pi_L(\cdot | \mathbf{y}_n) = \arg \min_{\rho \in \mathcal{P}(\Theta)} \{wn\mathbb{E}_{\theta \sim \rho} [L(\theta, \mathbf{y}_n)] + \text{KL}(\rho \| \pi)\}, \quad (7)$$

where $\mathcal{P}(\Theta)$ denotes the set of distributions over Θ , and KL denotes the KL divergence.

Proof. We follow here (but adapt to our notation) the proof given in Knoblauch et al. [2022], which in turn is based on the one for the related result contained in Bissiri et al. [2016].

Notice that the minimizer of the objective in Eq. (7) can be written as:

$$\begin{aligned} \pi^*(\cdot | \mathbf{y}_n) &= \arg \min_{\rho \in \mathcal{P}(\Theta)} \left\{ \int_{\Theta} \left[\log(\exp \{wnL(\theta, \mathbf{y}_n)\}) + \log \left(\frac{\rho(\theta)}{\pi(\theta)} \right) \right] \rho(\theta) d\theta \right\} \\ &= \arg \min_{\rho \in \mathcal{P}(\Theta)} \left\{ \int_{\Theta} \left[\log \left(\frac{\rho(\theta)}{\pi(\theta) \exp \{-wnL(\theta, \mathbf{y}_n)\}} \right) \right] \rho(\theta) d\theta \right\}. \end{aligned}$$

As we are only interested in the minimizer $\pi^*(\cdot|\mathbf{y}_n)$ (and not in the value of the objective), it holds that, for any constant $Z > 0$:

$$\begin{aligned}\pi^*(\cdot|\mathbf{y}_n) &= \arg \min_{\rho \in \mathcal{P}(\Theta)} \left\{ \int_{\Theta} \left[\log \left(\frac{\rho(\theta)}{\pi(\theta) \exp \{-wnL(\theta, \mathbf{y}_n)\} Z^{-1}} \right) \right] \rho(\theta) d\theta - \log Z \right\} \\ &= \arg \min_{\rho \in \mathcal{P}(\Theta)} \left\{ \text{KL}(\rho(\theta) \parallel \pi(\theta) \exp \{-wnL(\theta, \mathbf{y}_n)\} Z^{-1}) \right\}.\end{aligned}$$

Now, we can set $Z = \int_{\Theta} \pi(\theta) \exp \{-wnL(\theta, \mathbf{y}_n)\} d\theta$ (which is finite by assumption) and notice that we get:

$$\pi^*(\cdot|\mathbf{y}_n) = \arg \min_{\rho \in \mathcal{P}(\Theta)} \left\{ \text{KL}(\rho \parallel \pi_L(\cdot|\mathbf{y}_n)) \right\},$$

which yields $\pi^*(\cdot|\mathbf{y}_n) = \pi_L(\cdot|\mathbf{y}_n)$ as the KL is minimized uniquely if the two arguments are the same. \square

Next, we prove a finite sample (as it holds for fixed number of samples n) generalization bound. Our statement and proof generalize Lemma 8 in Matsubara et al. [2022b] (as we consider a generic loss function $L(\theta, \mathbf{y}_n)$, while they consider the Kernel Stein Discrepancy only).

In order to do this, let J be a function of the parameter θ , with $J(\theta)$ representing some loss (of which we will assume $L(\theta, \mathbf{y}_n)$ is a finite sample estimate; the meaning of J will be made clearer in the following and when applying this result to the SR posterior).

We will assume the following *prior mass condition*, which is more generic with respect to the one considered in the main body of this manuscript (Assumption **A5**):

A5bis Denote $\theta^* \in \arg \min_{\theta \in \Theta} J(\theta)$, which is supposed to be non-empty. The prior has a density $\pi(\theta)$ (with respect to Lebesgue measure) which satisfies

$$\int_{B_n(\alpha_1)} \pi(\theta) d\theta \geq e^{-\alpha_2 \sqrt{n}}$$

for some constants $\alpha_1, \alpha_2 > 0$, where we define the sets

$$B_n(\alpha_1) := \left\{ \theta \in \Theta : |J(\theta) - J(\theta^*)| \leq \alpha_1 / \sqrt{n} \right\}.$$

Assumption **A5bis** constrains the minimum amount of prior mass which needs to be given to J -balls with decreasing size, and is in general quite a weak condition (similar assumptions are taken in Chérif-Abdellatif and Alquier [2020], Matsubara et al. [2022b]).

Next, we state our result, which as mentioned above generalizes Lemma 8 in Matsubara et al. [2022b]:

Lemma 2. Consider the generalized posterior $\pi_L(\theta|\mathbf{y}_n)$ defined in Eq. (6), and assume that:

- (concentration) for all $\delta \in (0, 1]$:

$$P_0 \left\{ |L(\theta, \mathbf{Y}_n) - J(\theta)| \leq \epsilon_n(\delta) \right\} \geq 1 - \delta, \quad (8)$$

where $\epsilon_n(\delta) \geq 0$ is an approximation error term;

- $J(\theta^*) = \min_{\theta \in \Theta} J(\theta)$ is finite;
- Assumption **A5bis** holds.

Then, for all $\delta \in (0, 1]$, with probability at least $1 - \delta$:

$$\int_{\Theta} J(\theta) \pi_L(\theta|\mathbf{Y}_n) d\theta \leq J(\theta^*) + \frac{\alpha_1 + \alpha_2/w}{\sqrt{n}} + 2\epsilon_n(\delta),$$

where the probability is taken with respect to realisations of the dataset $\mathbf{Y}_n = \{Y_i\}_{i=1}^n, Y_i \stackrel{iid}{\sim} P_0$ for $i = 1, \dots, n$; this also implies the following statement:

$$P_0 \left(\left| \int_{\Theta} J(\theta) \pi_L(\theta|\mathbf{Y}_n) d\theta - J(\theta^*) \right| \geq \frac{\alpha_1 + \alpha_2/w}{\sqrt{n}} + 2\epsilon_n(\delta) \right) \leq \delta.$$

This result ensures that, with high probability, the expectation over the posterior of $J(\theta)$ is close to the minimum $J(\theta^*)$, provided that the distribution of $L(\theta, \mathbf{Y}_n)$ (where $\mathbf{Y}_n \sim P_0^n$ is a random variable) satisfies a concentration bound, which constrains how far $L(\theta, \mathbf{Y}_n)$ is distributed from the loss function $J(\theta)$. Notice that this result does not require the minimizer of J to be unique.

Typically the approximation error term $\epsilon_n(\delta)$ is such that $\epsilon_n(\delta) \xrightarrow{\delta \rightarrow 0} +\infty$ and $\epsilon_n(\delta) \xrightarrow{n \rightarrow \infty} 0$. If the second limit is verified, the posterior concentrates, for large n , on the values of θ which minimize J . In practical cases (as for instance for the SR posterior), it is common to have $J(\theta) = D(\theta, P_0)$, i.e., corresponding to a loss function relating θ with the data generating process P_0 .

We now prove the result.

Proof of Lemma 2. Due to the absolute value in Eq. (8), the following two inequalities hold simultaneously with probability (w.p.) at least $1 - \delta$:

$$J(\theta) \leq L(\theta, \mathbf{Y}_n) + \epsilon_n(\delta), \quad (9)$$

$$L(\theta, \mathbf{Y}_n) \leq J(\theta) + \epsilon_n(\delta). \quad (10)$$

Taking expectation with respect to the generalized posterior on both sides of Eq. (9) yields, w.p. $\geq 1 - \delta$:

$$\int_{\Theta} J(\theta) \pi_L(\theta | \mathbf{Y}_n) d\theta \leq \int_{\Theta} L(\theta, \mathbf{Y}_n) \pi_L(\theta | \mathbf{Y}_n) d\theta + \epsilon_n(\delta).$$

We now want to apply the identity in Eq. (7); therefore, we add $(wn)^{-1} \text{KL}(\pi_L(\cdot | \mathbf{Y}_n) \| \pi) \geq 0$ in the right hand side such that, w.p. $\geq 1 - \delta$:

$$\int_{\Theta} J(\theta) \pi_L(\theta | \mathbf{Y}_n) d\theta \leq \frac{1}{wn} \left\{ \int_{\Theta} wn L(\theta, \mathbf{Y}_n) \pi_L(\theta | \mathbf{Y}_n) d\theta + \text{KL}(\pi_L(\cdot | \mathbf{Y}_n) \| \pi) \right\} + \epsilon_n(\delta).$$

Now by Eq. (7):

$$\begin{aligned} \int_{\Theta} J(\theta) \pi_L(\theta | \mathbf{Y}_n) d\theta &\leq \frac{1}{wn} \inf_{\rho \in \mathcal{P}(\Theta)} \left\{ \int_{\Theta} wn L(\theta, \mathbf{Y}_n) \rho(\theta) d\theta + \text{KL}(\rho \| \pi) \right\} + \epsilon_n(\delta) \\ &= \inf_{\rho \in \mathcal{P}(\Theta)} \left\{ \int_{\Theta} L(\theta, \mathbf{Y}_n) \rho(\theta) d\theta + \frac{1}{wn} \text{KL}(\rho \| \pi) \right\} + \epsilon_n(\delta), \end{aligned} \quad (11)$$

where $\mathcal{P}(\Theta)$ denotes the space of probability distributions over Θ . Putting now Eq. (10) in Eq. (11) we have, w.p. $\geq 1 - \delta$:

$$\int_{\Theta} J(\theta) \pi_L(\theta | \mathbf{Y}_n) d\theta \leq \inf_{\rho \in \mathcal{P}(\Theta)} \left\{ \int_{\Theta} J(\theta) \rho(\theta) d\theta + \frac{1}{wn} \text{KL}(\rho \| \pi) \right\} + 2\epsilon_n(\delta),$$

and using the trivial bound $J(\theta) \leq J(\theta^*) + |J(\theta) - J(\theta^*)|$ we get:

$$\int_{\Theta} J(\theta) \pi_L(\theta | \mathbf{Y}_n) d\theta \leq J(\theta^*) + \inf_{\rho \in \mathcal{P}(\Theta)} \left\{ \int_{\Theta} |J(\theta) - J(\theta^*)| \rho(\theta) d\theta + \frac{1}{wn} \text{KL}(\rho \| \pi) \right\} + 2\epsilon_n(\delta).$$

Finally, we upper bound the infimum term by exploiting the prior mass condition in Assumption **A5bis**. Specifically, letting $\Pi(B_n) = \int_{B_n} \pi(\theta) d\theta$, we take $\rho(\theta) = \pi(\theta) / \Pi(B_n)$ for $\theta \in B_n$ and $\rho(\theta) = 0$ otherwise. By Assumption **A5bis**, we have therefore $\int_{B_n} |J(\theta) - J(\theta^*)| \rho(\theta) d\theta \leq \alpha_1 / \sqrt{n}$ and that $\text{KL}(\rho \| \pi) = \int_{\Theta} \log(\rho(\theta) / \pi(\theta)) \rho(\theta) d\theta = \int_{B_n} -\log(\Pi(B_n)) \pi(\theta) d\theta / \Pi(B_n) = -\log \Pi(B_n) \leq \alpha_2 \sqrt{n}$. Thus, we have:

$$\int_{\Theta} J(\theta) \pi_L(\theta | \mathbf{Y}_n) d\theta \leq J(\theta^*) + \frac{\alpha_1 + \alpha_2/w}{\sqrt{n}} + 2\epsilon_n(\delta),$$

as claimed in the first statement.

In order to obtain the second statement, notice that:

$$J(\theta) - J(\theta^*) \geq 0, \quad \forall \theta \in \Theta \implies \int_{\Theta} J(\theta) \pi_L(\theta | \mathbf{Y}_n) d\theta - J(\theta^*) \geq 0;$$

thus:

$$P_0 \left(\left| \int_{\Theta} J(\theta) \pi_L(\theta | \mathbf{Y}_n) d\theta - J(\theta^*) \right| \leq \frac{\alpha_1 + \alpha_2/w}{\sqrt{n}} + 2\epsilon_n(\delta) \right) \geq 1 - \delta;$$

taking the complement yields the result. \square

A.2.2 Case of Kernel and Energy Score posteriors

We now state and prove concentration results of the form in Eq. (8) for the Kernel and Energy Scores. Here, we will assume $X \perp\!\!\!\perp X'$. To this regards, notice that the kernel SR posterior can be written as:

$$\begin{aligned} \pi_{S_k}(\theta|\mathbf{y}_n) &\propto \pi(\theta) \exp \left\{ -w \sum_{i=1}^n \left[\mathbb{E}_{X, X' \sim P_\theta} k(X, X') - 2\mathbb{E}_{X \sim P_\theta} k(X, y_i) \right] \right\} \\ &\propto \pi(\theta) \exp \left\{ -w \sum_{i=1}^n \left[\mathbb{E}_{X, X' \sim P_\theta} k(X, X') - 2\mathbb{E}_{X \sim P_\theta} k(X, y_i) + \frac{1}{n-1} \sum_{\substack{j=1 \\ j \neq i}}^n k(y_i, y_j) \right] \right\}, \end{aligned}$$

as in fact the terms $k(y_i, y_j)$ are independent of θ . From the second line in the above expression and the form of the generalized Bayes posterior with generic loss in Eq. (6), we can identify:

$$L(\theta, \mathbf{y}_n) = \mathbb{E}_{X, X' \sim P_\theta} k(X, X') - \frac{2}{n} \sum_{i=1}^n \mathbb{E}_{X \sim P_\theta} k(X, y_i) + \frac{1}{n(n-1)} \sum_{\substack{i, j=1 \\ i \neq j}}^n k(y_i, y_j). \quad (12)$$

Similarly, the Energy Score posterior can be obtained by identifying in Eq. (6):

$$L(\theta, \mathbf{y}_n) = \frac{2}{n} \sum_{i=1}^n \mathbb{E}_{X \sim P_\theta} \|X - y_i\|_2^\beta - \frac{1}{n(n-1)} \sum_{\substack{i, j=1 \\ i \neq j}}^n \|y_i - y_j\|_2^\beta - \mathbb{E}_{X, X' \sim P_\theta} \|X - X'\|_2^\beta; \quad (13)$$

this can be obtained by simply setting $k(x, y) = -\|x - y\|_2^\beta$ in Eq. (12), as the Kernel SR with that choice of kernel recovers the Energy SR.

For both SRs, $L(\theta, \mathbf{Y}_n)$ is an unbiased estimator (with respect to $Y_i \sim P_0$) of the associated divergences; in fact, considering $X \perp\!\!\!\perp X' \sim P_\theta$ and $Y \perp\!\!\!\perp Y' \sim P_0$, the associated divergence for Kernel SR is the squared MMD (see Section 2.1):

$$D_k(P_\theta, P_0) = \mathbb{E}k(X, X') + \mathbb{E}k(Y, Y') - 2\mathbb{E}k(X, Y), \quad (14)$$

while, for the Energy SR, the associated divergence is the squared Energy Distance:

$$D_E^{(\beta)}(P_\theta, P_0) = 2\mathbb{E}\|X - Y\|_2^\beta - \mathbb{E}\|X - X'\|_2^\beta - \mathbb{E}\|Y - Y'\|_2^\beta. \quad (15)$$

In order to prove our concentration results, we will exploit the following Lemma:

Lemma 3 (McDiarmid's inequality, McDiarmid 1989). *Let g be a function of n variables $\mathbf{y}_n = \{y_i\}_{i=1}^n$, and let*

$$\delta_i g(\mathbf{y}_n) := \sup_{z \in \mathcal{X}} g(y_1, \dots, y_{i-1}, z, y_{i+1}, \dots, y_n) - \inf_{z \in \mathcal{X}} g(y_1, \dots, y_{i-1}, z, y_{i+1}, \dots, y_n),$$

and $\|\delta_i g\|_\infty := \sup_{\mathbf{y}_n \in \mathcal{X}^n} |\delta_i g(\mathbf{y}_n)|$. If Y_1, \dots, Y_n are independent random variables:

$$P(g(Y_1, \dots, Y_n) - \mathbb{E}g(Y_1, \dots, Y_n) \geq \varepsilon) \leq e^{-2\varepsilon^2 / \sum_{i=1}^n \|\delta_i g\|_\infty^2}.$$

We are now ready to prove two concentration results of the form of Eq. (8). The first holds for the Kernel SR assuming a bounded kernel, while the latter holds for the Energy SR assuming a bounded \mathcal{X} . Let us start with a simple equality stated in the following Lemma:

Lemma 4. *For $L(\theta, \mathbf{Y}_n)$ defined in Eq. (12) and $D_k(P_\theta, P_0)$ defined in Eq. (14), we have:*

$$L(\theta, \mathbf{Y}_n) - D_k(P_\theta, P_0) = g(Y_1, Y_2, \dots, Y_n) - \mathbb{E}[g(Y_1, Y_2, \dots, Y_n)]$$

for

$$g(Y_1, Y_2, \dots, Y_n) = \frac{1}{n(n-1)} \sum_{\substack{i, j=1 \\ i \neq j}}^n k(Y_i, Y_j) - \frac{2}{n} \sum_{i=1}^n \mathbb{E}_{X \sim P_\theta} k(X, Y_i). \quad (16)$$

Similar expression holds for $L(\theta, \mathbf{Y}_n)$ defined in Eq. (13) and $D_E^{(\beta)}(P_\theta, P_0)$ defined in Eq. (15), by setting $k(x, y) = -\|x - y\|_2^\beta$.

Proof. First, notice that, for $L(\theta, \mathbf{Y}_n)$ defined in Eq. (12) and $D_k(P_\theta, P_0)$ defined in Eq. (14):

$$\begin{aligned}
L(\theta, \mathbf{Y}_n) - D_k(P_\theta, P_0) &= \mathbb{E}_{X, X' \sim P_\theta} k(X, X') - \frac{2}{n} \sum_{i=1}^n \mathbb{E}_{X \sim P_\theta} k(X, Y_i) + \frac{1}{n(n-1)} \sum_{\substack{i,j=1 \\ i \neq j}}^n k(Y_i, Y_j) + \\
&\quad - \mathbb{E}_{X, X' \sim P_\theta} k(X, X') - \mathbb{E}_{Y, Y' \sim P_0} [k(Y, Y')] + 2\mathbb{E}_{X \sim P_\theta, Y \sim P_0} [k(X, Y)] \\
&= \frac{1}{n(n-1)} \sum_{\substack{i,j=1 \\ i \neq j}}^n k(Y_i, Y_j) - \frac{2}{n} \sum_{i=1}^n \mathbb{E}_{X \sim P_\theta} k(X, Y_i) + \\
&\quad - (\mathbb{E}_{Y, Y' \sim P_0} [k(Y, Y')] - 2\mathbb{E}_{X \sim P_\theta, Y \sim P_0} [k(X, Y)]) \\
&= g(Y_1, Y_2, \dots, Y_n) - \mathbb{E}[g(Y_1, Y_2, \dots, Y_n)],
\end{aligned}$$

where the expectation in the last line is with respect to $Y_i \sim P_0$, $i = 1, \dots, n$, and where we set g as in Eq. (16). \square

Now, we give the concentration result for the kernel SR:

Lemma 5. Consider $L(\theta, \mathbf{y}_n)$ defined in Eq. (12) (corresponding to the loss function defining the Kernel Score posterior) and $D_k(P_\theta, P_0)$ defined in Eq. (14); if the kernel is such that $\sup_{x,y \in \mathcal{X}} |k(x, y)| \leq \kappa < \infty$, we have:

$$P_0 \left(|L(\theta, \mathbf{Y}_n) - D_k(P_\theta, P_0)| \leq \sqrt{-\frac{32\kappa^2}{n} \log \frac{\delta}{2}} \right) \geq 1 - \delta.$$

Proof. First, we write:

$$L(\theta, \mathbf{Y}_n) - D_k(P_\theta, P_0) = g(Y_1, Y_2, \dots, Y_n) - \mathbb{E}[g(Y_1, Y_2, \dots, Y_n)],$$

where g is defined in Eq. (16) in Lemma 4. Next, notice that:

$$\begin{aligned}
P_0(|g(\mathbf{Y}_n) - \mathbb{E}[g(\mathbf{Y}_n)]| \geq \epsilon) &\leq P_0(g(\mathbf{Y}_n) - \mathbb{E}[g(\mathbf{Y}_n)] \geq \epsilon) + P_0(g(\mathbf{Y}_n) - \mathbb{E}[g(\mathbf{Y}_n)] \leq -\epsilon) \\
&= P_0(g(\mathbf{Y}_n) - \mathbb{E}[g(\mathbf{Y}_n)] \geq \epsilon) + P_0(-g(\mathbf{Y}_n) - \mathbb{E}[-g(\mathbf{Y}_n)] \geq \epsilon)
\end{aligned}$$

by the union bound. We use now McDiarmid's inequality (Lemma 3) to prove the result. Consider first $P_0(g(\mathbf{Y}_n) - \mathbb{E}[g(\mathbf{Y}_n)] \geq \epsilon)$; thus:

$$\begin{aligned}
|\delta_i g(\mathbf{Y}_n)| &= \left| \sup_z \left\{ \frac{2}{n(n-1)} \sum_{\substack{j=1 \\ j \neq i}}^n k(z, Y_j) - \frac{2}{n} \mathbb{E}_{X \sim P_\theta} k(X, z) \right\} - \inf_z \left\{ \frac{2}{n(n-1)} \sum_{\substack{j=1 \\ j \neq i}}^n k(z, Y_j) - \frac{2}{n} \mathbb{E}_{X \sim P_\theta} k(X, z) \right\} \right| \\
&= \left| \sup_z \left\{ \frac{2}{n(n-1)} \sum_{\substack{j=1 \\ j \neq i}}^n k(z, Y_j) - \frac{2}{n} \mathbb{E}_{X \sim P_\theta} k(X, z) \right\} + \sup_z \left\{ \frac{2}{n} \mathbb{E}_{X \sim P_\theta} k(X, z) - \frac{2}{n(n-1)} \sum_{\substack{j=1 \\ j \neq i}}^n k(z, Y_j) \right\} \right| \\
&\leq \sup_z \left| \frac{2}{n(n-1)} \sum_{\substack{j=1 \\ j \neq i}}^n k(z, Y_j) - \frac{2}{n} \mathbb{E}_{X \sim P_\theta} k(X, z) \right| + \sup_z \left| \frac{2}{n} \mathbb{E}_{X \sim P_\theta} k(X, z) - \frac{2}{n(n-1)} \sum_{\substack{j=1 \\ j \neq i}}^n k(z, Y_j) \right| \\
&= 2 \cdot \frac{2}{n} \sup_z \left| \frac{1}{n-1} \sum_{\substack{j=1 \\ j \neq i}}^n k(z, Y_j) - \mathbb{E}_{X \sim P_\theta} k(X, z) \right| \leq \frac{4}{n} \sup_z \left\{ \frac{1}{n-1} \sum_{\substack{j=1 \\ j \neq i}}^n |k(z, Y_j)| + \mathbb{E}_{X \sim P_\theta} |k(X, z)| \right\} \\
&\leq \frac{4}{n} \left\{ \frac{1}{n-1} \sum_{\substack{j=1 \\ j \neq i}}^n \underbrace{\sup_z |k(z, Y_j)|}_{\leq \kappa} + \mathbb{E}_{X \sim P_\theta} \underbrace{\sup_z |k(X, z)|}_{\leq \kappa} \right\} \leq \frac{4}{n} \left\{ \frac{1}{n-1} \cdot (n-1)\kappa + \kappa \right\} = \frac{8\kappa}{n}
\end{aligned}$$

As the bound does not depend on \mathbf{Y}_n , we have that $\|\delta_i g\|_\infty \leq \frac{8\kappa}{n}$, from which McDiarmid's inequality (Lemma 3) gives:

$$P_0(g(\mathbf{Y}_n) - \mathbb{E}[g(\mathbf{Y}_n)] \geq \epsilon) \leq \exp\left(-\frac{2\epsilon^2}{n \cdot \frac{64\kappa^2}{n^2}}\right) = e^{-\frac{n\epsilon^2}{32\kappa^2}}.$$

For the bound on the other side, notice that $\|\delta_i(-g)\|_\infty = \|\delta_i g\|_\infty$; therefore, we also have

$$P_0(-g(\mathbf{Y}_n) - \mathbb{E}[-g(\mathbf{Y}_n)] \geq \epsilon) \leq e^{-\frac{n\epsilon^2}{32\kappa^2}},$$

from which:

$$P_0(|g(\mathbf{Y}_n) - \mathbb{E}[g(\mathbf{Y}_n)]| \geq \epsilon) \leq 2e^{-\frac{n\epsilon^2}{32\kappa^2}}.$$

Defining the right hand side of the bound as δ , we get:

$$P_0\left(|g(\mathbf{Y}_n) - \mathbb{E}[g(\mathbf{Y}_n)]| \geq \sqrt{-\frac{32\kappa^2}{n} \log \frac{\delta}{2}}\right) \leq \delta,$$

from which the result is obtained taking the complement. \square

We now give the analogous result for the Energy Score:

Lemma 6. Consider $L(\theta, \mathbf{y}_n)$ defined in Eq. (13) (corresponding to the loss function defining the Energy Score posterior) and $D_E^{(\beta)}(P_\theta, P_0)$ defined in Eq. (15); assume that the space \mathcal{X} is bounded such that $\sup_{x,y \in \mathcal{X}} \|x - y\|_2 \leq B < \infty$; therefore, we have:

$$P_0\left(\left|L(\theta, \mathbf{Y}_n) - D_E^{(\beta)}(P_\theta, P_0)\right| \leq \sqrt{-\frac{32B^{2\beta}}{n} \log \frac{\delta}{2}}\right) \geq 1 - \delta.$$

Proof. We rely on Lemma 5; in fact, recall that the Kernel Score recovers the Energy Score for $k(x, y) = -\|x - y\|_2^\beta$. With this choice of k , Eqs. (12) and (14) (considered in Lemma 5) respectively recover Eqs. (13) and (15).

Additionally, assuming \mathcal{X} to be bounded ensures that $|k(x, y)| = \|x - y\|_2^\beta \leq B^\beta$; therefore, we can apply Lemma 5 with $\kappa = B^\beta$, from which the result follows. \square

We are finally ready to prove our generalization bound:

Proof of Theorem 2. The proof consists in verifying the assumptions of Lemma 2, for both the Energy and Kernel Score posteriors. First, notice that **A5** is a specific case of **A5bis** by identifying $J(\theta) = D_k(P_\theta, P_0)$ or $J(\theta) = D_E(P_\theta, P_0)$. We therefore need to verify the first and second assumptions only.

Let us first consider the Kernel Score posterior (part 1 of Theorem 2). Recall that, for positive-definite, Cauchy-Schwarz inequality holds:

$$|k(x, y)| \leq \sqrt{k(x, x)k(y, y)}.$$

Hence, the boundedness assumption stated in part 1 of Theorem 2 implies that in Lemma 5:

$$\sup_{x,y \in \mathcal{X}} |k(x, y)| \leq \sup_{x,y \in \mathcal{X}} \sqrt{k(x, x)k(y, y)} \leq \kappa.$$

Also, the Kernel Score posterior corresponds to the generalized Bayes posterior in Eq. (6) by choosing $L(\theta, \mathbf{Y}_n)$ defined in Eq. (12); with this choice of $L(\theta, \mathbf{Y}_n)$, Lemma 5 holds, which corresponds to the first assumption of Lemma 2 with $J(\theta) = D_k(P_\theta, P_0)$ (D_k being the divergence related to the kernel SR, defined in Eq. (14)) and:

$$\epsilon_n(\delta) = \sqrt{-\frac{32\kappa^2}{n} \log \frac{\delta}{2}}.$$

Finally, we have that $D_k(P_\theta, P_0) \geq 0$, which ensures the second assumption of Lemma 2. Thus, we have, from Lemma 2:

$$P_0 \left(\left| \int_{\Theta} D_k(P_\theta, P_0) \pi_{S_k}(\theta | \mathbf{Y}_n) d\theta - D_k(P_{\theta^*}, P_0) \right| \geq \frac{1}{\sqrt{n}} \left(\alpha_1 + \frac{\alpha_2}{w} + 8\kappa \sqrt{-2 \log \frac{\delta}{2}} \right) \right) \leq \delta;$$

by defining the deviation term as ϵ and inverting the relation, we obtain the result for the kernel Score Posterior.

The same steps can be taken for the the Energy Score posterior; specifically, we notice that it corresponds to the generalized Bayes posterior in Eq. (6) by choosing $L(\theta, \mathbf{Y}_n)$ defined in Eq. (13); with this choice of $L(\theta, \mathbf{Y}_n)$, Lemma 6 holds, which corresponds to the first assumption of Lemma 2 with $J(\theta) = D_E^{(\beta)}(P_\theta, P_0)$ ($D_E^{(\beta)}$ being the divergence related to the Energy SR defined in Eq. (15)) and:

$$\epsilon_n(\delta) = \sqrt{-\frac{32B^{2\beta}}{n} \log \frac{\delta}{2}}.$$

Finally, we have that $D_E^{(\beta)}(P_\theta, P_0) \geq 0$, which ensures the second assumption of Lemma 2. Thus, we have, from Lemma 2:

$$P_0 \left(\left| \int_{\Theta} D_E^{(\beta)}(P_\theta, P_0) \pi_{S_E^{(\beta)}}(\theta | \mathbf{Y}_n) d\theta - D_E^{(\beta)}(P_{\theta^*}, P_0) \right| \geq \frac{1}{\sqrt{n}} \left(\alpha_1 + \frac{\alpha_2}{w} + 8B^\beta \sqrt{-2 \log \frac{\delta}{2}} \right) \right) \leq \delta;$$

by defining the deviation term as ϵ and inverting the relation, we obtain the result for the Energy Score Posterior. □

We remark here that Theorem 1 in Chérif-Abdellatif and Alquier [2020] proved a similar generalization bound for the Kernel Score posterior holding in expectation (rather than in high probability, as for our bounds), albeit under a slightly different prior mass condition.

A.3 Proof of Theorem 3

Global bias-robustness (for a generic constant $C < \infty$) was shown in Matsubara et al. [2022b] for their kernel Stein discrepancy (KSD) posterior. Here, we provide an upper bound for the constant C for both the kernel and energy score posteriors.

To prove our result, we first generalize Lemma 5 in Matsubara et al. [2022b] (our Lemma 7), which provides bounds on the constant for global bias-robustness for a generalized Bayes posterior depending on bounds on the loss function defining the posterior.

Across this Section, we define as $\hat{P}_n = \frac{1}{n} \sum_{i=1}^n \delta_{y_i}$ the empirical distribution given by the observations $\mathbf{y}_n = \{y_i\}_{i=1}^n$ (considered to be non-random here) and consider the generalized Bayes posterior:

$$\pi_L(\theta | \hat{P}_n) \propto \pi(\theta) \exp \left\{ -wnL(\theta, \hat{P}_n) \right\}, \quad (17)$$

from which the SR posterior in Eq. (2) with Scoring Rule S is recovered with:

$$L(\theta, \hat{P}_n) = \frac{1}{n} \sum_{i=1}^n S(P_\theta, y_i) = \mathbb{E}_{Y \sim \hat{P}_n} S(P_\theta, Y),$$

We remark that the notation is here slightly different from Appendix A.2, in which we considered L to be a function of θ and \mathbf{y}_n (compare Eq. 17 with Eq. 6). The reason of this will be clear in the following.

We start by stating the result we will rely on, to which we provide proof for ease of reference.

Lemma 7. *Let $\pi_L(\theta | \hat{P}_n)$ be the generalized posterior defined in Eq. (17) for fixed $n \in \mathbb{N}$, with a generic loss $L(\theta, \hat{P}_n)$ and prior $\pi(\theta)$. Let $\Delta_n = \sup_{\theta \in \Theta} L(\theta; \hat{P}_n) - \inf_{\theta \in \Theta} L(\theta; \hat{P}_n)$ and $DL(z, \theta, \hat{P}_n) = (d/d\epsilon)L(\theta, \hat{P}_{n,\epsilon,z}) \Big|_{\epsilon=0}$.*

Then,

$$\sup_{\theta \in \Theta} \sup_{z \in \mathcal{X}} \left| \text{PIF} \left(z, \theta, \hat{P}_n \right) \right| \leq 2wn e^{wn\Delta_n} \cdot \sup_{\theta \in \Theta} \sup_{z \in \mathcal{X}} \left| \text{DL} \left(z, \theta, \hat{P}_n \right) \right| \cdot \sup_{\theta \in \Theta} \pi(\theta).$$

Proof. First of all, Eq. (17) of Ghosh and Basu [2016] demonstrates that

$$\begin{aligned} \text{PIF} \left(z, \theta, \hat{P}_n \right) &= wn \pi_L(\theta | \hat{P}_n) \left(-\text{DL} \left(z, \theta, \hat{P}_n \right) + \int_{\Theta} \text{DL} \left(z, \theta', \hat{P}_n \right) \pi_L(\theta' | \hat{P}_n) d\theta' \right) \\ &\leq wn \pi_L(\theta | \hat{P}_n) \left(\sup_{\theta' \in \Theta} \text{DL} \left(z, \theta', \hat{P}_n \right) - \text{DL} \left(z, \theta, \hat{P}_n \right) \right), \end{aligned}$$

where PIF denotes the posterior influence function defined in Sec. 2.2.3 in the main text and where the inequality holds due to the mean of a random variable always being smaller than the maximum value the variable can get.

We can now get the following upper bound:

$$\begin{aligned} \sup_{\theta \in \Theta} \sup_{z \in \mathcal{X}} \left| \text{PIF} \left(z, \theta, \hat{P}_n \right) \right| &\leq wn \sup_{\theta \in \Theta} \left\{ \pi_L(\theta | \hat{P}_n) \left(\sup_{z \in \mathcal{X}} \left| \text{DL} \left(z, \theta, \hat{P}_n \right) \right| + \sup_{z \in \mathcal{X}} \sup_{\theta' \in \Theta} \left| \text{DL} \left(z, \theta', \hat{P}_n \right) \right| \right) \right\} \\ &\leq wn \sup_{\theta \in \Theta} \left\{ \pi_L(\theta | \hat{P}_n) \sup_{z \in \mathcal{X}} \left| \text{DL} \left(z, \theta, \hat{P}_n \right) \right| \right\} + wn \left\{ \sup_{\theta \in \Theta} \pi_L(\theta | \hat{P}_n) \cdot \sup_{z \in \mathcal{X}} \sup_{\theta' \in \Theta} \left| \text{DL} \left(z, \theta', \hat{P}_n \right) \right| \right\} \\ &\leq 2wn \left\{ \sup_{\theta \in \Theta} \pi_L(\theta | \hat{P}_n) \cdot \sup_{z \in \mathcal{X}} \sup_{\theta \in \Theta} \left| \text{DL} \left(z, \theta, \hat{P}_n \right) \right| \right\}. \end{aligned}$$

Recall now that

$$\begin{aligned} \pi_L(\theta | \hat{P}_n) &= \frac{\pi(\theta) \exp \left\{ -wnL \left(\theta; \hat{P}_n \right) \right\}}{\int_{\Theta} \pi(\theta) \exp \left\{ -wnL \left(\theta; \hat{P}_n \right) \right\} d\theta} \leq \frac{\pi(\theta) \exp \left\{ -wn \inf_{\theta \in \Theta} L \left(\theta; \hat{P}_n \right) \right\}}{\int_{\Theta} \pi(\theta) \exp \left\{ -wnL \left(\theta; \hat{P}_n \right) \right\} d\theta} \\ &\leq \frac{\pi(\theta) \exp \left\{ -wn \inf_{\theta \in \Theta} L \left(\theta; \hat{P}_n \right) \right\}}{\inf_{\theta \in \Theta} \exp \left\{ -wnL \left(\theta; \hat{P}_n \right) \right\}} = \frac{\pi(\theta) \exp \left\{ -wn \inf_{\theta \in \Theta} L \left(\theta; \hat{P}_n \right) \right\}}{\exp \left\{ -wn \sup_{\theta \in \Theta} L \left(\theta; \hat{P}_n \right) \right\}} \\ &= \pi(\theta) \exp \left\{ wn \left(\sup_{\theta \in \Theta} L \left(\theta; \hat{P}_n \right) - \inf_{\theta \in \Theta} L \left(\theta; \hat{P}_n \right) \right) \right\}, \end{aligned}$$

Let us now denote $\Delta_n = \sup_{\theta \in \Theta} L \left(\theta; \hat{P}_n \right) - \inf_{\theta \in \Theta} L \left(\theta; \hat{P}_n \right)$. From the upper bound above, we have:

$$\sup_{\theta \in \Theta} \sup_{z \in \mathcal{X}} \left| \text{PIF} \left(z, \theta, \hat{P}_n \right) \right| \leq 2wn e^{wn\Delta_n} \sup_{\theta \in \Theta} \pi(\theta) \sup_{z \in \mathcal{X}} \sup_{\theta \in \Theta} \left| \text{DL} \left(z, \theta, \hat{P}_n \right) \right|$$

as claimed. \square

Next, we give the explicit form for $\text{DL} \left(z, \theta, \hat{P}_n \right)$ in our case in the following Lemma:

Lemma 8. For $L \left(\theta, \hat{P}_{n,\epsilon,z} \right) = \mathbb{E}_{Y \sim \hat{P}_{n,\epsilon,z}} S(P_\theta, Y)$, we have:

$$\text{DL} \left(z, \theta, \hat{P}_n \right) = S(P_\theta, z) - \mathbb{E}_{Y \sim \hat{P}_n} S(P_\theta, Y);$$

further, setting $S = S_k$, where S_k is the kernel scoring rule with kernel k , we have:

$$\text{DL} \left(z, \theta, \hat{P}_n \right) = 2\mathbb{E}_{X \sim P_\theta} \left[\mathbb{E}_{Y \sim \hat{P}_n} k(X, Y) - k(X, z) \right];$$

finally, the form for the energy score can be obtained by setting $k(x, y) = -\|x - y\|_2^\beta$.

Proof. For the first statement, notice that:

$$\mathbb{E}_{Y \sim \hat{P}_{n,\epsilon,z}} S(P_\theta, Y) = (1 - \epsilon) \mathbb{E}_{Y \sim \hat{P}_n} S(P_\theta, Y) + \epsilon S(P_\theta, z),$$

from which differentiating with respect to ϵ gives the statement.

For the second statement, recall the form for the kernel SR:

$$S_k(P, z) = \mathbb{E}_{X, X' \sim P} [k(X, X')] - 2\mathbb{E}_{X \sim P} [k(X, z)],$$

from which:

$$\begin{aligned} DL(z, \theta, \hat{P}_n) &= S_k(P_\theta, z) - \mathbb{E}_{\hat{P}_n} S_k(P_\theta, Y) \\ &= \mathbb{E}_{X, X' \sim P_\theta} [k(X, X')] - 2\mathbb{E}_{X \sim P_\theta} [k(X, z)] - \mathbb{E}_{Y \sim \hat{P}_n} [\mathbb{E}_{X, X' \sim P_\theta} [k(X, X')] - 2\mathbb{E}_{X \sim P_\theta} [k(X, Y)]] \\ &= \mathbb{E}_{X, X' \sim P_\theta} [k(X, X')] - 2\mathbb{E}_{X \sim P_\theta} [k(X, z)] - \mathbb{E}_{X, X' \sim P_\theta} [k(X, X')] + 2\mathbb{E}_{Y \sim \hat{P}_n} \mathbb{E}_{X \sim P_\theta} [k(X, Y)] \\ &= 2\mathbb{E}_{X \sim P_\theta} [E_{Y \sim \hat{P}_n} k(X, Y) - k(X, z)]. \end{aligned}$$

□

Finally, we state the proof for Theorem 3:

Proof of Theorem 3. The proof consists in verifying the conditions necessary for Lemma 7 for the Kernel and Energy Score posteriors

First, let us consider the Kernel Score posterior; recall that, for positive-definite kernels, Cauchy-Schwarz inequality holds:

$$|k(x, y)| \leq \sqrt{k(x, x)k(y, y)}.$$

Hence, the boundedness assumption in Theorem 3 yields:

$$\sup_{x, y \in \mathcal{X}} |k(x, y)| \leq \sup_{x, y \in \mathcal{X}} \sqrt{k(x, x)k(y, y)} \leq \kappa.$$

Thus, we have:

$$\begin{aligned} |L(\theta, \hat{P}_n)| &\leq \frac{1}{n} \sum_{i=1}^n |S_k(P_\theta, y_i)| \leq \frac{1}{n} \sum_{i=1}^n \mathbb{E} [|k(X, X')| + |2k(X, y_i)|] \\ &\leq \frac{1}{n} \sum_{i=1}^n [\kappa + 2\kappa] = 3\kappa, \end{aligned}$$

where all expectations are over $X, X' \sim P_\theta$ and the bound exploits the fact that $|k(x, y)| \leq \kappa$. This implies that

$$\Delta_n = \sup_{\theta \in \Theta} L(\theta; \hat{P}_n) - \inf_{\theta \in \Theta} L(\theta; \hat{P}_n) \leq 6\kappa.$$

Using a similar argument as above, notice that, for the kernel SR (using Lemma 8):

$$\begin{aligned} |DL(z, \theta, \hat{P}_n)| &= 2 \left| \mathbb{E}_{X \sim P_\theta} \mathbb{E}_{Y \sim \hat{P}_n} [k(X, Y) - k(X, z)] \right| \\ &\leq 2\mathbb{E}_{X \sim P_\theta} \mathbb{E}_{Y \sim \hat{P}_n} [|k(X, Y)| + |k(X, z)|] \\ &\leq 2\mathbb{E}_{X \sim P_\theta} \mathbb{E}_{Y \sim \hat{P}_n} [\kappa + \kappa] = 4\kappa. \end{aligned}$$

Hence, by Lemma 7 we have, for the kernel score posterior

$$\sup_{\theta \in \Theta} \sup_{z \in \mathcal{X}} \left| \text{PIF}(z, \theta, \hat{P}_n) \right| \leq 8\omega n \kappa e^{\delta \omega n \kappa} \sup_{\theta \in \Theta} \pi(\theta),$$

as claimed.

For the statement about the Energy Score posterior, we proceed in similar manner. First, let us show that, under the assumptions of the Theorem, $L(\theta, \hat{P}_n)$ for the Energy Score $S_E^{(\beta)}$ is lower bounded; in fact:

$$\begin{aligned} L(\theta, \hat{P}_n) &= \frac{1}{n} \sum_{i=1}^n S_E^{(\beta)}(P_\theta, y_i) = \frac{1}{n} \sum_{i=1}^n \mathbb{E} \left[2\|X - y_i\|_2^\beta - \|X - X'\|_2^\beta \right] = \frac{2}{n} \sum_{i=1}^n \mathbb{E}\|X - y_i\|_2^\beta - \mathbb{E}\|X - X'\|_2^\beta \\ &= \underbrace{\frac{2}{n} \sum_{i=1}^n \mathbb{E}\|X - y_i\|_2^\beta - \mathbb{E}\|X - X'\|_2^\beta - \frac{1}{n^2} \sum_{i,j=1}^n \|y_i - y_j\|_2^\beta + \frac{1}{n^2} \sum_{i,j=1}^n \|y_i - y_j\|_2^\beta}_{=D_E^{(\beta)}(P_\theta, \hat{P}_n)}, \end{aligned}$$

where $D_E^{(\beta)}(P_\theta, \hat{P}_n)$ is the squared Energy Distance between P_θ and the empirical distribution \hat{P}_n ; as the Energy Distance is a distance between probability measures [Rizzo and Székely, 2016], $D_E^{(\beta)}(P_\theta, \hat{P}_n) \geq 0$, from which:

$$L(\theta, \hat{P}_n) = D_E^{(\beta)}(P_\theta, \hat{P}_n) + \frac{1}{n^2} \sum_{i,j=1}^n \|y_i - y_j\|_2^\beta \geq 0.$$

Additionally, recall that, as we assume \mathcal{X} to be bounded, there exists $B < \infty$ such that $\sup_{x,y \in \mathcal{X}} \|x - y\|_2 \leq B$. Thus:

$$L(\theta, \hat{P}_n) = \frac{1}{n} \sum_{i=1}^n \mathbb{E} \left[2\|X - y_i\|_2^\beta - \|X - X'\|_2^\beta \right] \leq \frac{2}{n} \sum_{i=1}^n \mathbb{E}\|X - y_i\|_2^\beta \leq 2B^\beta$$

Hence, we have

$$\Delta_n = \sup_{\theta \in \Theta} L(\theta; \hat{P}_n) - \inf_{\theta \in \Theta} L(\theta; \hat{P}_n) \leq 2B^\beta.$$

Moreover, for the Energy SR (using Lemma 8):

$$\begin{aligned} \left| DL(z, \theta, \hat{P}_n) \right| &= 2 \left| \mathbb{E}_{X \sim P_\theta} \mathbb{E}_{Y \sim \hat{P}_n} \left[\|X - z\|_2^\beta - \|X - Y\|_2^\beta \right] \right| \\ &\leq 2 \mathbb{E}_{X \sim P_\theta} \mathbb{E}_{Y \sim \hat{P}_n} \left[\left| \|X - z\|_2^\beta \right| + \left| \|X - Y\|_2^\beta \right| \right] \\ &\leq 2 \mathbb{E}_{X \sim P_\theta} \mathbb{E}_{Y \sim \hat{P}_n} \left[B^\beta + B^\beta \right] = 4B^\beta, \end{aligned}$$

where the last inequality is due to $z \in \mathcal{X}$ and the boundedness assumptions for \mathcal{X} . Hence, by Lemma 7 we have, for the energy score posterior

$$\sup_{\theta \in \Theta} \sup_{z \in \mathcal{X}} \left| \text{PIF}(z, \theta, \hat{P}_n) \right| \leq 8wnB^\beta e^{2wnB^\beta} \sup_{\theta \in \Theta} \pi(\theta),$$

as claimed. \square

A.4 Precise statement and proof of Theorem 4

We recall here for simplicity the useful definitions. We consider the SR posterior:

$$\pi_S(\theta | \mathbf{y}_n) \propto \pi(\theta) \exp \left\{ \underbrace{-w \sum_{i=1}^n S(P_\theta, y_i)}_{p_S(\mathbf{y}_n | \theta)} \right\}.$$

Further, we recall the form of the target of the pseudo-marginal MCMC:

$$\pi_{\hat{S}}^{(m)}(\theta | \mathbf{y}_n) \propto \pi(\theta) p_{\hat{S}}^{(m)}(\mathbf{y}_n | \theta),$$

where:

$$p_{\hat{S}}^{(m)}(\mathbf{y}_n | \theta) = \mathbb{E} \left[\exp \left\{ -w \sum_{i=1}^n \hat{S}(\mathbf{X}_m^{(\theta)}, y_i) \right\} \right] = \int \exp \left\{ -w \sum_{i=1}^n \hat{S}(\mathbf{x}_m^{(\theta)}, y_i) \right\} \prod_{j=1}^m p(x_j^{(\theta)} | \theta) dx_1 dx_2 \cdots dx_m.$$

The complete version of Theorem 4 is given in the following:

Theorem 6. *Assume the following:*

1. $\hat{S}(\mathbf{X}_m^{(\theta)}, y_i)$ converges in probability to $S(P_\theta, y_i)$ as $m \rightarrow \infty$ for all $i = 1, \dots, n$.
2. $\sup_m \mathbb{E} \left[\left| \exp\{-w \sum_{i=1}^n \hat{S}(\mathbf{X}_m^{(\theta)}, y_i)\} \right|^{1+\delta} \right] < \infty$ for some $\delta > 0$
3. $\inf_m \int_{\Theta} p_{\hat{S}}^{(m)}(\mathbf{y}_n|\theta)\pi(\theta)d\theta > 0$ and $\sup_{\theta \in \Theta} p_S(\mathbf{y}_n|\theta) < \infty$.

Then,

$$\lim_{m \rightarrow \infty} \pi_{\hat{S}}^{(m)}(\theta|\mathbf{y}_n) = \pi_S(\theta|\mathbf{y}_n).$$

A.5 Proof of Theorem 6

In order to prove Theorem 6, we extend the proof for the analogous result for Bayesian inference with an auxiliary likelihood [Drovandi et al., 2015]. Our setup is slightly more general as we do not constrain the update to be defined in terms of a likelihood; notice that the original setup in Drovandi et al. [2015] is recovered when we consider S being the negative log likelihood, for some auxiliary likelihood.

We begin by stating a useful property:

Lemma 9 (Theorem 3.5 in Billingsley [1999]). *If X_n is a sequence of uniformly integrable random variables and X_n converges in distribution to X , then X is integrable and $\mathbb{E}[X_n] \rightarrow \mathbb{E}[X]$ as $n \rightarrow \infty$.*

Remark 6 (Remark 1 in Drovandi et al. [2015]). *A simple sufficient condition for uniform integrability is that for some $\delta > 0$:*

$$\sup_n \mathbb{E}[|X_n|^{1+\delta}] < \infty.$$

The result in the main text is the combination of the following two Theorems, which respectively generalize Results 1 and 2 in Drovandi et al. [2015]:

Theorem 7 (Generalizes Result 1 in Drovandi et al. [2015]). *Assume that $p_{\hat{S}}^{(m)}(\mathbf{y}_n|\theta) \rightarrow p_S(\mathbf{y}_n|\theta)$ as $m \rightarrow \infty$ for all θ with positive prior support; further, assume $\inf_m \int_{\Theta} p_{\hat{S}}^{(m)}(\mathbf{y}_n|\theta)\pi(\theta)d\theta > 0$ and $\sup_{\theta \in \Theta} p_S(\mathbf{y}_n|\theta) < \infty$. Then*

$$\lim_{m \rightarrow \infty} \pi_{\hat{S}}^{(m)}(\theta|\mathbf{y}_n) = \pi_S(\theta|\mathbf{y}_n).$$

Furthermore, if $f : \Theta \rightarrow \mathbb{R}$ is a continuous function satisfying $\sup_m \int_{\Theta} |f(\theta)|^{1+\delta} \pi_S^{(m)}(\theta|\mathbf{y}_n)d\theta < \infty$ for some $\delta > 0$ then

$$\lim_{m \rightarrow \infty} \int_{\Theta} f(\theta) \pi_{\hat{S}}^{(m)}(\theta|\mathbf{y}_n)d\theta = \int_{\Theta} f(\theta) \pi_S(\theta|\mathbf{y}_n)d\theta.$$

Proof. The first part follows from the fact that the numerator of

$$\pi_{\hat{S}}^{(m)}(\theta|\mathbf{y}_n) = \frac{p_{\hat{S}}^{(m)}(\mathbf{y}_n|\theta)\pi(\theta)}{\int_{\Theta} p_{\hat{S}}^{(m)}(\mathbf{y}_n|\theta)\pi(\theta)d\theta}$$

converges pointwise and the denominator is positive and converges by the bounded convergence theorem.

For the second part, if for each $m \in \mathbb{N}$, θ_m is distributed according to $\pi_{\hat{S}}^{(m)}(\cdot|\mathbf{y}_n)$ and θ is distributed according to $\pi_S(\cdot|\mathbf{y}_n)$ then θ_m converges to θ in distribution as $m \rightarrow \infty$ by Scheffé's lemma [Scheffé, 1947]. Since f is continuous, $f(\theta_m)$ converges in distribution to $f(\theta)$ as $n \rightarrow \infty$ by the continuous mapping theorem and we conclude by application of Remark 6 and Lemma 9. \square

The following gives a convenient way to ensure $p_{\hat{S}}^{(m)}(\mathbf{y}_n|\theta) \rightarrow p_S(\mathbf{y}_n|\theta)$:

Theorem 8 (Generalizes Result 2 in Drovandi et al. [2015]). *Assume that $\exp\{-w \sum_{i=1}^n \hat{S}(\mathbf{X}_m^{(\theta)}, y_i)\}$ converges in probability to $p_S(\mathbf{y}_n|\theta)$ as $m \rightarrow \infty$. If*

$$\sup_m \mathbb{E} \left[\left| \exp\left\{-w \sum_{i=1}^n \hat{S}(\mathbf{X}_m^{(\theta)}, y_i)\right\} \right|^{1+\delta} \right] < \infty$$

for some $\delta > 0$ then $p_{\hat{S}}^{(m)}(\mathbf{y}_n|\theta) \rightarrow p_S(\mathbf{y}_n|\theta)$ as $m \rightarrow \infty$.

Proof. The proof follows by applying Remark 6 and Lemma 9. \square

We are finally ready to prove Theorem 6:

Proof of Theorem 6. First, notice how the convergence in probability of $\hat{S}(\mathbf{X}_m^{(\theta)}, y_i)$ to $S(P_\theta, y_i)$ (assumption 1 in Theorem 6) and the continuity of the exponential function imply convergence in probability of $\exp\{-w \sum_i \hat{S}(\mathbf{X}_m^{(\theta)}, y_i)\}$ to $p_S(\mathbf{y}_n|\theta)$. That, together with assumption 2 in Theorem 6, satisfy the requirements of Theorem 8. With the latter and assumption 3 in Theorem 6, Theorem 7 holds, which yields the result. \square

B Changing data coordinates

We give here some more details on the behavior of the SR posterior when the coordinate system used to represent the data is changed, as mentioned in Remark 2.

Frequentist estimator First, we investigate whether the minimum scoring rule estimator (for a strictly proper scoring rule) is affected by a transformation of the data. Specifically, considering a strictly proper S , we are interested in whether $\theta_Y^* = \arg \min_{\theta \in \Theta} S(P_\theta^Y, Q_Y) = \arg \min_{\theta \in \Theta} D(P_\theta^Y, Q_Y)$ is the same as $\theta_Z^* = \arg \min_{\theta \in \Theta} S(P_\theta^Z, Q_Z) = \arg \min_{\theta \in \Theta} D(P_\theta^Z, Q_Z)$, where $Z = f(Y) \implies Y \sim Q_Y \iff Z \sim Q_Z$ and $Y \sim P_\theta^Y \iff Z \sim P_\theta^Z$. If the model is well specified, $P_{\theta_Y^*}^Y = Q_Y, P_{\theta_Z^*}^Z = Q_Z \implies \theta_Y^* = \theta_Z^*$. If the model is misspecified, for a generic SR the minimizer of the expected SR may change according to the parametrization. We remark how this is not a drawback of the frequentist minimum SR estimator but rather a feature, as such estimator is the parameter value corresponding to the model minimizing the chosen expected scoring rule from the data generating process *in that coordinate system*, and is therefore completely reasonable for it to change when the coordinate system is modified.

Notice that a sufficient condition for $\theta_Y^* = \theta_Z^*$ is $S(P_\theta^Y, y) = a \cdot S(P_\theta^Z, z) + b$ for $a > 0, b \in \mathbb{R}$. This condition is verified when S is chosen to be the log-score, as in fact:

$$S(P_\theta^Z, f(y)) = -\ln p_Z(f(y)|\theta) = S(P_\theta^Z, y) + \ln |J_f(y)|,$$

where we assumed f to be a one-to-one function and we applied the change of variable formula to the density p_Z .

Generalized Bayesian posterior For a single observation, let π_S^Y denote the SR posterior conditioned on values of Y , while π_S^Z denote instead the posterior conditioned on values of $Z = f(Y)$ for some one-to-one function f ; in general, $\pi_S^Y(\theta|y) \neq \pi_S^Z(\theta|f(y))$. By denoting as w_Z (respectively w_Y) and P_θ^Z (respectively P_θ^Y) the weight and model distributions appearing in π_S^Z (resp. π_S^Y), the equality would in fact require $w_Z S(P_\theta^Z, f(y)) = w_Y S(P_\theta^Y, y) + C \forall \theta, y$ for some choice of w_Z, w_Y and for all transformations f , where C is a constant in θ . Notice that this is satisfied for the standard Bayesian posterior (i.e., with the log-score) with $w_Z = w_Y = 1$. Instead, for other scoring rules the above condition cannot be satisfied in general for any choice of w_Z, w_Y . For instance, consider the kernel SR:

$$S(P_\theta^Z, f(y)) = \mathbb{E}[k(Z, \tilde{Z})] - \mathbb{E}[k(Z, f(y))] = \mathbb{E}[k(f(Y), f(\tilde{Y}))] - \mathbb{E}[k(f(Y), f(y))];$$

for general kernels and functions f , the above is different from $S(P_\theta^Y, y) = \mathbb{E}[k(Y, \tilde{Y})] - \mathbb{E}[k(Y, f(x))]$ up to a constant, unless the kernel is redefined as well. Therefore, the posterior shape depends on the chosen data coordinates. Considering the expression for the kernel SR, it is clear that is a consequence of the fact that the likelihood principle is not satisfied (as the kernel SR does not only depend on the likelihood value at the observation). Similar argument holds for the Energy Score posterior as well.

We also remark that this is also the case for BSL [Price et al., 2018], as in that case the model is assumed to be multivariate normal, and changing the data coordinates impacts their normality (in fact it is common practice in BSL to look for transformations of data which yield distribution as close as possible to a normal one).

The theoretical semiBSL posterior [An et al., 2020], instead, is invariant with respect to one-to-one transformation applied independently to each data coordinate, which do not affect the copula structure. Notice however that different data coordinate systems may yield better empirical estimates of the marginal KDEs from model simulations.

C More details on related techniques

C.1 Energy Distance

The squared energy distance is a metric between probability distributions [Rizzo and Székely, 2016], and is defined by:

$$D_E^{(\beta)}(P, Q) = 2 \cdot \mathbb{E} \left[\|X - Y\|_2^\beta \right] - \mathbb{E} \left[\|X - X'\|_2^\beta \right] - \mathbb{E} \left[\|Y - Y'\|_2^\beta \right],$$

for $X \perp\!\!\!\perp X' \sim P$ and $Y \perp\!\!\!\perp Y' \sim Q$.

The probabilistic forecasting literature [Gneiting and Raftery, 2007] use a different convention of the energy score and distance, which amounts to multiplying our definitions by $1/2$. We follow here the convention used in the statistical inference literature [Rizzo and Székely, 2016, Chérif-Abdellatif and Alquier, 2020, Nguyen et al., 2020].

C.2 Semi-Parametric Synthetic Likelihood

Semi-parametric BSL (semiBSL, An et al. 2020) considers a Gaussian copula modeling the dependency between the different components in P_θ and poses no constraints on marginal densities. The semiBSL likelihood is thus:

$$p_{\text{semiBSL}}(y|\theta) = c_{R_\theta}(F_{\theta,1}(y^1), \dots, F_{\theta,d}(y^d)) \prod_{k=1}^d f_{\theta,k}(y^k), \quad (18)$$

where $f_{\theta,k}$ and $F_{\theta,k}$ are respectively the marginal density and Cumulative Density Functions (CDFs) for the k -th component of the model, and where $c_R(u)$ denotes the Gaussian copula density for $u \in [0, 1]^d$ and correlation matrix $R \in [-1, 1]^{d \times d}$.

As in BSL, An et al. [2020] used a pseudo-marginal MCMC where simulations from P_θ are used to obtain an estimate of the correlation matrix of the Gaussian copula \hat{R}_θ as well as Kernel Density Estimates (KDE) of the marginals $\hat{f}_{\theta,k}$ (from which $\hat{F}_{\theta,k}$ are obtained by integration). We can connect semiBSL to our framework by rewriting Eq. (18) as:

$$p_{\text{semiBSL}}(y|\theta) = \exp \left\{ - \sum_{k=1}^d S_{\log} \left(P_\theta^k, y^k \right) - S_{Gc}(C_\theta, (F_{\theta,1}(y^1), \dots, F_{\theta,d}(y^d))) \right\},$$

where P_θ^k is the distribution associated to the model for the k -th component, C_θ is the copula associated to P_θ and $S_{Gc}(C, u)$ is the copula Scoring Rule associated to the Gaussian copula; we show below that this is a proper, but not strictly so, Scoring Rule for copula random variables. Consistent estimators of the marginal and copula SRs can be obtained using \hat{R}_θ , $\hat{f}_{\theta,k}$ and $\hat{F}_{\theta,k}$.

Copula scoring rule Finally, we write down the explicit expression of the copula scoring rule S_{Gc} , associated to the Gaussian copula. We show that this is a proper, but not strictly so, scoring rule for copula distributions. Specifically, let C be a distribution for a copula random variable, and let $u \in [0, 1]^d$. We define:

$$S_{Gc}(C, u) = \frac{1}{2} \log |R_C| + \frac{1}{2} (\Phi^{-1}(u))^T (R_C^{-1} - \mathbf{I}_d) \Phi^{-1}(u),$$

where Φ^{-1} is applied element-wise to u , and R_C is the correlation matrix associated to C in the following way: define the copula random variable $V \sim C$ and its transformation $\Phi^{-1}(V)$; then, $\Phi^{-1}(V)$ will have a multivariate normal distribution with mean 0 and covariance matrix R_C .

Similarly to the Dawid–Sebastiani score (see Sec. 2.1), this scoring rule is proper but not strictly so as it only depends on the first 2 moments of the distribution of the random variable $\Phi^{-1}(V)$ (the first one being equal to 0). To show this, assume the copula random variable U has an exact distribution Q and consider the expected scoring rule:

$$S_{Gc}(C, Q) = \mathbb{E}_{U \sim Q} S_{Gc}(C, U) = \frac{1}{2} \log |R_C| + E_{U \sim Q} \left[(\Phi^{-1}(U))^T (R_C^{-1} - \mathbf{I}_d) \Phi^{-1}(U) \right];$$

now, notice that $\Phi^{-1}(U)$ is a multivariate normal distribution whose marginals are standard normals. Therefore, let us denote as R_Q the covariance matrix of $\Phi^{-1}(U)$, which is a correlation matrix. From the well-known form for the expectation of a quadratic form³, it follows that:

$$\begin{aligned} S_{Gc}(C, Q) &= \frac{1}{2} \log |R_C| + \frac{1}{2} \text{Tr} [(R_C^{-1} - \mathbf{I}_d) \cdot R_Q] \\ &= \frac{1}{2} \log |R_C| + \frac{1}{2} \text{Tr} [R_C^{-1} \cdot R_Q] - \frac{1}{2} \text{Tr} [R_Q] \\ &= \frac{1}{2} \underbrace{\left\{ \log \frac{|R_C|}{|R_Q|} - d + \text{Tr} [R_C^{-1} \cdot R_Q] \right\}}_{D_{KL}(Z_Q || Z_C)} + \frac{1}{2} \log R_Q + \frac{d}{2} - \frac{1}{2} \text{Tr} [R_Q], \end{aligned}$$

where $D_{KL}(Z_Q || Z_C)$ is the KL divergence between two multivariate normal distributions Z_Q and Z_C of dimension d , with mean 0 and covariance matrix R_Q and R_C respectively. Further, notice that the remaining factors do not depend on the distribution C . Therefore, $S_{Gc}(C, Q)$ is minimized whenever R_C is equal to R_Q ; this happens when $C = Q$, but also for all other choices of C which share the associated covariance matrix with Q . This implies that the Gaussian copula score is a proper, but not strictly so, scoring rule for copula distributions.

C.3 Ratio estimation

The standard Bayes posterior can be written as $\pi(\theta|y) = \pi(\theta) \cdot r(y; \theta)$, with $r(y; \theta) = \frac{p(y|\theta)}{p(y)}$. The Ratio Estimation (RE) approach [Thomas et al., 2020] builds an approximate posterior by estimating $\log r(y; \theta)$ with some function $\hat{h}^\theta(y)$ and considering $\pi_{\text{re}}(\theta|y) \propto \pi(\theta) \exp(\hat{h}^\theta(y))$.

Thomas et al. [2020] run an MCMC where, for each proposed θ , m samples $\mathbf{x}_m^{(\theta)}$ are generated from P_θ . These, together with a set of m reference samples $\mathbf{x}_m^{(r)} = \{x_j^{(r)}\}_{j=1}^m$ from the marginal data distribution⁴, are used to fit a logistic regression yielding $\hat{h}^\theta(y)$. Logistic regression is an optimization problem in which the best function of \mathcal{X} in distinguishing between the two sets of samples is selected. If $m \rightarrow \infty$ and all scalar functions are considered, the optimum h_*^θ is equal to $\log r(y; \theta)$. For finite data, however, the corresponding optimum \hat{h}_m^θ is only an approximation of the ratio. RE is therefore a specific case of our SR posterior framework with $w = 1$ and:

$$\hat{S}_{\text{RE}}(\mathbf{x}_m^{(\theta)}, \mathbf{x}_m^{(r)}, y) = -\hat{h}_m^\theta(y)$$

³ $\mathbb{E} [X^T \Lambda X] = \text{tr} [\Lambda \Sigma] + \mu^T \Lambda \mu$, for a symmetric matrix Λ , and where μ and Σ are the mean and covariance matrix of X (which in general does not need to be normal, but only needs to have well defined second moments).

⁴Which are obtained by drawing $\theta_j \sim p(\theta)$, $x_j \sim p(\cdot|\theta_j)$, and discarding θ_j .

In general, the number of reference samples and samples from the model can be different, see Appendix C.3; we make this choice here for the sake of simplicity.

which, differently from the other SR estimators considered previously, also depends on the reference samples. Due to what we discussed above, \hat{S}_{RE} converges in probability to the log-score (up to a constant term in θ) for $m \rightarrow \infty$.

The above argument relies on optimizing over all functions in logistic regression; in practice, the optimization is restricted to a set of functions \mathcal{H} (for instance, a linear combination of predictors). In this case, the infinite data optimum $h_{\mathcal{H}^*}^\theta(y)$ does not correspond to $\log r(y; \theta)$, but to the best possible approximation in \mathcal{H} in some sense. Therefore, Ratio Estimation with a restricted set of functions \mathcal{H} cannot be written exactly under our SR posterior framework. However, very flexible function classes (as for instance neural networks) can produce reasonable approximations to the log score for large values of m .

D Tuning the bandwidth of the Gaussian kernel

Consider the Gaussian kernel:

$$k(x, y) = \exp\left(-\frac{\|x - y\|_2^2}{2\gamma^2}\right);$$

inspired by Park et al. [2016], we fix the bandwidth γ with the following procedure:

1. Simulate a value $\theta_j \sim \pi(\theta)$ and a set of samples $x_{jk} \sim P_{\theta_j}$, for $k = 1, \dots, m_\gamma$.
2. Estimate the median of $\{\|x_{jk} - x_{jl}\|_2\}_{kl}^{m_\gamma}$ and call it $\hat{\gamma}_j$.
3. Repeat points 1) and 2) for $j = 1, \dots, m_{\theta, \gamma}$.
4. Set the estimate for γ as the median of $\{\hat{\gamma}_j\}_{j=1}^{m_{\theta, \gamma}}$.

Empirically, we use $m_{\theta, \gamma} = 1000$ and we set m_γ to the corresponding value of m for the different models.

E Further details on simulation studies reported in the main text

In all experiments, we use independent normal proposals on each component of the parameter space in MCMC, with σ denoting the standard deviation of the normal proposal distribution, which we report below. In all cases, whenever the parameter space is bounded, we run MCMC on a transformed unbounded space obtained via a logistic transformation. Therefore, the proposal sizes refer to that unbounded space.

E.1 The g-and-k model

E.1.1 Additional experimental details for well specified setup

We report here additional experimental details on the g-and-k model experiments (Sec. 4.1.1).

First, we discuss settings for the SR posteriors:

- For the Energy Score posterior, our strategy (Sec. 2.4) for setting w using BSL as a reference resulted in $w \approx 0.35$ for the univariate model and $w \approx 0.16$ for the multivariate one.
- For the Kernel Score posterior, we first fit the value of the Gaussian kernel bandwidth parameter as described in Appendix D, which resulted in $\gamma \approx 5.50$ for the univariate case and $\gamma \approx 52.37$ for the multivariate one. Then, the tuning strategy for w using BSL as a reference resulted in $w \approx 18.30$ for the univariate model and $w \approx 52.29$ for the multivariate one.

Next, we discuss the proposal sizes for MCMC; recall that we use independent normal proposals on each component of θ , with standard deviation σ . We report here the values for σ used in the experiments; we stress that, as the MCMC is run in the transformed unbounded parameter space (obtained applying a logit transformation), these proposal sizes refer to that space.

For the univariate g-and-k, the proposal sizes we use are the following:

N. obs.	Univariate g-and-k			Multivariate g-and-k			
	BSL	Kernel Score	Energy Score	BSL	semiBSL	Kernel Score	Energy Score
1	0.362	0.507	0.420	0.216	0.190	0.468	0.445
5	0.221	0.329	0.375	0.069	/	0.136	0.224
10	0.133	0.252	0.272	0.036	/	0.127	0.216
15	0.109	0.253	0.217	/	/	0.077	0.154
20	0.100	0.154	0.207	/	/	0.151	0.278
25	0.092	0.149	0.208	/	/	0.126	0.233
30	0.085	0.218	0.343	/	/	0.124	0.222
35	0.080	0.172	0.315	/	/	0.076	0.166
40	0.076	0.152	0.293	/	/	0.119	0.246
45	0.070	0.130	0.256	/	/	0.103	0.223
50	0.062	0.121	0.220	/	/	0.103	0.219
55	0.060	0.189	0.317	/	/	0.139	0.297
60	0.059	0.185	0.324	/	/	0.129	0.286
65	0.057	0.173	0.314	/	/	0.133	0.273
70	0.052	0.172	0.289	/	/	0.119	0.256
75	0.048	0.161	0.273	/	/	0.123	0.247
80	0.048	0.159	0.267	/	/	0.117	0.233
85	0.045	0.150	0.252	/	/	0.098	0.213
90	0.044	0.143	0.247	/	/	0.087	0.198
95	0.044	0.136	0.244	/	/	0.089	0.198
100	0.042	0.129	0.236	/	/	0.076	0.190

Table 2: Acceptance rates for the univariate and multivariate g-and-k experiments with different values of n , with the MCMC proposal sizes reported in Appendix E.1.1. “/” denotes experiments for which MCMC did not run satisfactorily.

- For BSL, we use $\sigma = 1$ for all values of n .
- For Energy and Kernel Scores, we take $\sigma = 1$ for n from 1 up to 25 (included), $\sigma = 0.4$ for n from 30 to 50, and $\sigma = 0.2$ for n from 55 to 100.

For the multivariate g-and-k:

- For BSL and semiBSL, we use $\sigma = 1$ for all values of n for which the chain converges. We stress that we tried decreasing the proposal size, but that did not solve the non-convergence issue (discussed in the main text in Sec. 4.1.1).
- For Energy and Kernel Scores, we take $\sigma = 1$ for n from 1 up to 15 (included), $\sigma = 0.4$ for n from 20 to 35, $\sigma = 0.2$ for n from 40 to 50 and $\sigma = 0.1$ for n from 55 to 100.

In Table 2, we report the acceptance rates the different methods achieve for all values of n , with the proposal sizes mentioned above. We denote by “/” the experiments for which we did not manage to run MCMC satisfactorily. We remark how the Energy Score achieves a larger acceptance rates in all experiments compared to the Kernel Score.

E.1.2 Poor MCMC performance for BSL and semiBSL

As discussed in the main text (Sec. 4.1.1), the correlated pseudo-marginal MCMC for BSL and semiBSL performed poorly for the multivariate g-and-k example, not being able to converge when using more than respectively 1 and 10 observations. In order to shed light on the reason for this behavior, we fix $n = 20$ and run MCMC with 10 different initializations, for 10000 MCMC steps with no burn-in, for BSL and semiBSL, with $m = 500$. For all runs, we found that, after a short transient, the

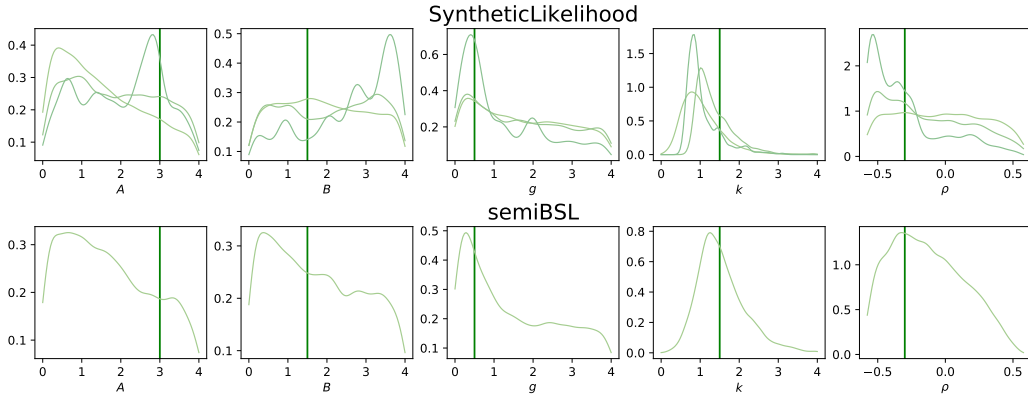


Figure 8: Marginal posterior distributions for the different parameters for the well specified multivariate g-and-k model. For BSL, we were able to run the inference for $n = 1, 5, 10$, while we were only able to do so for $n = 1$ for semiBSL. Darker (respectively lighter) colors denote a larger (smaller) number of observations. The densities are obtained by KDE on the MCMC output thinned by a factor 10.

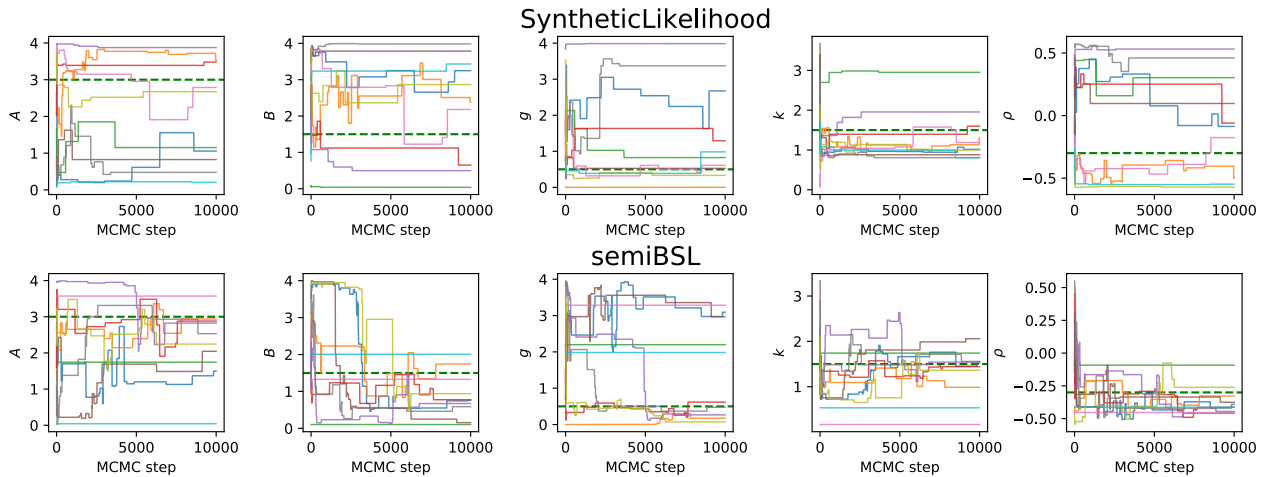


Figure 9: Traceplots for semiBSL and BSL for $n = 20$ for 10 different initializations (different colors), with 10000 MCMC steps (no burn-in); the green dashed line denotes the true parameter value. It can be seen that the chains are very sticky, and that they explore different parts of the parameter space.

different chains get stuck in different parameter values and have very “sticky” behavior. The results can be seen in Fig. 9.

In order to understand the reason for this result, we investigate whether the poor performance is due to large variance in the estimate of the target; as increasing the number of simulations reduces such variance, we study the effect of this on the MCMC performance. Therefore, we report here the results of a study increasing the number of simulations for a fixed number of observations $n = 20$ for the g-and-k model. Specifically, we tested $m = 500, 1000, 1500, 2000, 2500, 3000, 30000$; as discussed in Appendix E.1.1, we used a proposal size $\sigma = 0.4$, with which the Energy and Kernel Score posteriors performed well. We report traceplots in Fig. 10 and corresponding acceptance rates in Table 3; from this experiment, we note that BSL achieves acceptance rate as large as few percentage points with larger m values, but there is no constant trend (for instance, acceptance rate with $m = 3000$ is smaller than with $m = 2000$), which means that the method is still prone to getting stuck. For semiBSL, the acceptance rate is abysmal even for very large m .

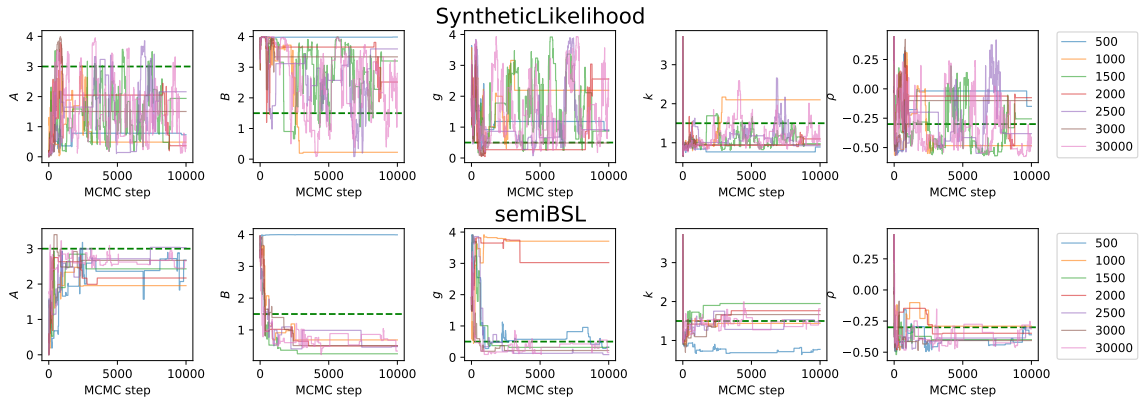


Figure 10: Traceplots for BSL and semiBSL and BSL for $n = 20$ using different number of simulations m , reported in the legend for each row; green dashed line denotes the true parameter value. There is no improvement in the mixing of the chain for increasing the number of simulations.

N. simulations m	500	1000	1500	2000	2500	3000	30000
Acc. rate BSL	$6.0 \cdot 10^{-3}$	$1.1 \cdot 10^{-2}$	$3.3 \cdot 10^{-2}$	$9.9 \cdot 10^{-3}$	$1.8 \cdot 10^{-2}$	$7.5 \cdot 10^{-3}$	$5.1 \cdot 10^{-2}$
Acc. rate semiBSL	$7.0 \cdot 10^{-3}$	$3.4 \cdot 10^{-3}$	$3.7 \cdot 10^{-3}$	$2.8 \cdot 10^{-3}$	$4.2 \cdot 10^{-3}$	$3.6 \cdot 10^{-3}$	$9.2 \cdot 10^{-3}$

Table 3: Acceptance rates for BSL and semiBSL and BSL for $n = 20$ using different number of simulations m ; there is no improvement in the acceptance rate for increasing number of simulations. We recall that we were not able to run semiBSL for $m = 30000$ due to its high computational cost.

E.1.3 Additional experimental details for misspecified setup

We report here additional experimental details on the misspecified g-and-k model experiments, where the observations were generated by a Cauchy distribution (Sec. 4.1.2).

In order to have coherent results with respect to the well specified case, we use here the values of w and γ determined in the well specified case (reported in Appendix E.1.1)

Next, we discuss the proposal sizes for MCMC (which is run with independent normal proposals on each component of θ with standard deviation σ , in the same way as in the well specified case, after applying a logit transformation to the parameter space).

- For the univariate g-and-k, for all methods (BSL, Energy and Kernel Scores), we take $\sigma = 1$ for n from 1 up to 25 (included), $\sigma = 0.4$ for n from 30 to 50, and $\sigma = 0.2$ for n from 55 to 100.
- For the multivariate g-and-k, recall that we did not report results for BSL and semiBSL here as we were not able to sample the posteriors with MCMC for large n , as already experienced in the well specified case. For the remaining techniques, we used the same values of σ as in the well specified experiments (Appendix E.1.3).

In Table 4, we report the acceptance rates the different methods achieve for all values of n , with the proposal sizes discussed above. We remark how the Energy Score achieves a larger acceptance rates in all experiments compared to the Kernel Score.

E.2 The misspecified normal location model

As mentioned in the main text (Sec. 4.2), we set the weight w such that the variance achieved by our SR posteriors is approximately the same as the one achieved by the standard Bayes distribution for the well specified case ($\epsilon = 0$). This resulted in $w = 1$ for the Energy Score posterior and $w = 2.8$ for the Kernel Score posterior. Additionally, the bandwidth for the Gaussian kernel was tuned to be $\gamma \approx 0.9566$ (with the strategy discussed in Appendix D).

N. obs.	Misspecified univariate g-and-k			Misspecified multivariate g-and-k	
	BSL	Kernel Score	Energy Score	Kernel Score	Energy Score
1	0.457	0.482	0.521	0.472	0.470
5	0.302	0.436	0.454	0.324	0.373
10	0.193	0.450	0.425	0.362	0.330
15	0.146	0.441	0.390	0.361	0.276
20	0.102	0.264	0.311	0.544	0.410
25	0.093	0.288	0.314	0.530	0.377
30	0.153	0.426	0.471	0.536	0.359
35	0.144	0.349	0.448	0.537	0.336
40	0.134	0.340	0.440	0.631	0.432
45	0.130	0.344	0.429	0.523	0.373
50	0.125	0.255	0.393	0.383	0.343
55	0.167	0.318	0.501	0.471	0.436
60	0.176	0.303	0.490	0.412	0.407
65	0.164	0.293	0.481	0.389	0.391
70	0.164	0.276	0.455	0.372	0.374
75	0.156	0.272	0.445	0.278	0.329
80	0.157	0.262	0.436	0.232	0.306
85	0.153	0.254	0.430	0.247	0.300
90	0.147	0.231	0.415	0.239	0.299
95	0.152	0.226	0.410	0.235	0.291
100	0.141	0.223	0.407	0.232	0.277

Table 4: Acceptance rates for the misspecified univariate and multivariate g-and-k experiments with different values of n , with the MCMC proposal sizes reported in Appendix E.1.3.

In Figure 11 we report the full set of posterior distributions for the different values of ϵ and z obtained with the standard Bayes posterior and with our SR posteriors.

In the MCMC with the SR posteriors, a proposal size $\sigma = 2$ is used for all values of ϵ and z . For all experiments, Table 5 reports acceptance rates obtained with the SR posteriors, while Table 6 reports the obtained posterior standard deviation with SR posteriors and for the standard Bayes distribution (for which we do not give the proposal size and acceptance rate as it was sampled using more advanced MCMC techniques than standard Metropolis-Hastings using the PyMC3 library [Salvatier et al., 2016]).

Setup	$\epsilon = 0$	$\epsilon = 0.1$					$\epsilon = 0.2$				
	-	$z = 3$	$z = 5$	$z = 7$	$z = 10$	$z = 20$	$z = 3$	$z = 5$	$z = 7$	$z = 10$	$z = 20$
Kernel Score	0.076	0.086	0.089	0.085	0.086	0.087	0.080	0.086	0.089	0.091	0.090
Energy Score	0.076	0.084	0.087	0.082	0.083	0.085	0.082	0.079	0.077	0.082	0.082

Table 5: Acceptance rates for MCMC targeting the Energy and Kernel Score posteriors for the different outlier setups, for the misspecified normal location model.

Finally, as mentioned in the main text (Sec. 4.2), we attempted using BSL in this scenario. As the model is Gaussian, we expected the BSL posterior to be very close to the standard posterior. Indeed, this is what we observed in the well specified case and for small z (Figure 12). When however z is increased, the MCMC targeting the BSL posterior does not perform satisfactorily (see the trace plots in Figure 13). Neither reducing the proposal size nor running the chain for a longer number of steps seems to solve this issues, which reminds of the issue discussed in Sec. 4.1.

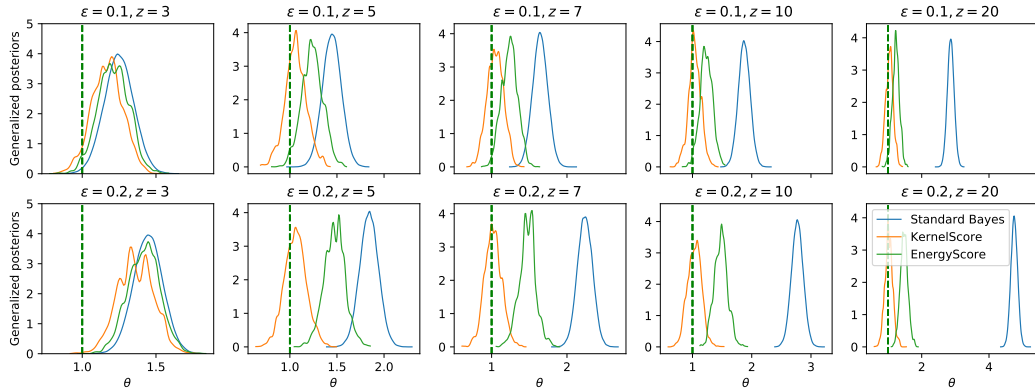


Figure 11: Posterior distribution obtained with the Scoring Rules and exact Bayes for the misspecified normal location model; each panel represents a different choice of ϵ and z . It can be seen that both Kernel and Energy score are more robust with respect to Standard Bayes, with the Kernel Score one being extremely robust. The densities are obtained by KDE on the MCMC output thinned by a factor 10.

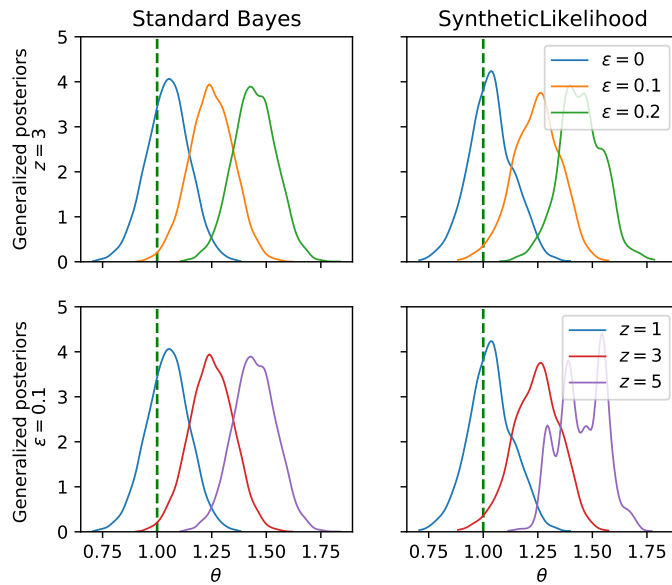


Figure 12: Standard Bayes and BSL posteriors for the normal location model, for different choices of ϵ and z . First row: fixed outliers location $z = 3$ and varying proportion ϵ ; second row: fixed outlier proportion ϵ , varying location z . As expected, the BSL posterior is very close to the standard Bayes posterior. The densities are obtained by KDE on the MCMC output thinned by a factor 10.

Setup	$\epsilon = 0$	$\epsilon = 0.1$					$\epsilon = 0.2$				
	-	$z = 3$	$z = 5$	$z = 7$	$z = 10$	$z = 20$	$z = 3$	$z = 5$	$z = 7$	$z = 10$	$z = 20$
Standard Bayes	0.100	0.100	0.099	0.099	0.099	0.100	0.099	0.099	0.100	0.099	0.099
Kernel Score	0.101	0.106	0.114	0.108	0.105	0.113	0.121	0.116	0.113	0.117	0.116
Energy Score	0.098	0.105	0.112	0.106	0.106	0.107	0.109	0.112	0.111	0.114	0.113

Table 6: Obtained posterior standard deviation for the standard Bayes and the Energy and Kernel Score posteriors, for the different outlier setups, for the misspecified normal location model.

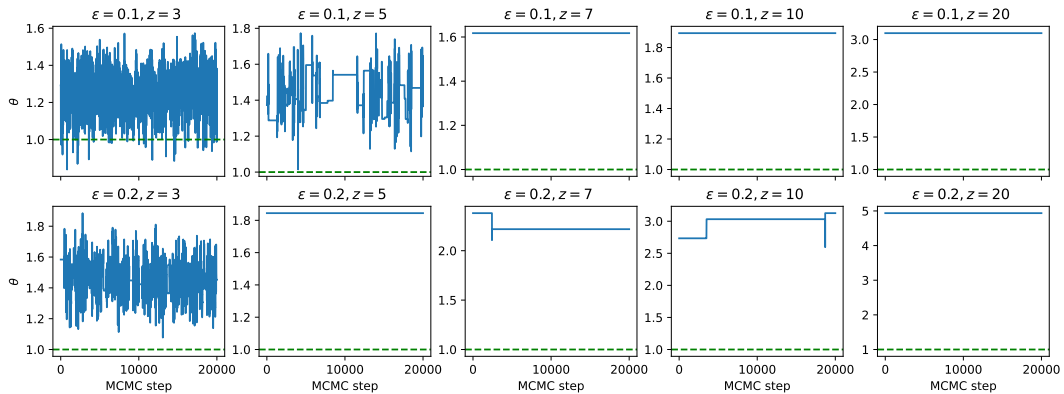


Figure 13: Trace plots for MCMC targeting the BSL posterior with different choices of z and ϵ , for the misspecified normal location model. We used here proposal size $\sigma = 2$ and 60000 MCMC steps, of which 40000 were burned in; reducing the proposal size or increasing the number of steps did not seem to solve this issue.

E.3 The MA(2) model

For the Kernel Score posterior with the Gaussian Kernel, we first fit the value of the bandwidth of the Gaussian kernel as described in Appendix D, which resulted in $\gamma \approx 12.77$.

Next, we used our strategy (Sec.2.4) to tune the value of the weight w for both the Kernel and Energy Score Posteriors; this resulted in $w \approx 12.97$ for the Energy Score and $w \approx 208$ for the Kernel Score.

In Table 7 we report the proposal sizes σ and the resulting acceptance rates and trace of the posterior covariance matrix Σ_{post} for BSL, semiBSL and the SR posteriors with the above values of w ; we also report the trace of Σ_{post} for the true posterior, for which we do not give the proposal size and acceptance rate as it was sampled using more advanced MCMC techniques than standard Metropolis-Hastings using the PyMC3 library [Salvatier et al., 2016].

Technique	BSL	semiBSL	True posterior	Kernel Score	Energy Score
Proposal size σ	1	0.2	/	1	1
Acceptance rate	0.16	0.16	/	0.51	0.46
$\text{Tr}[\Sigma_{post}]$	0.08595	0.05271	0.04483	0.3010	0.2600

Table 7: Proposal sizes and acceptance rates for the BSL, semiBSL, Kernel and Energy Score posteriors, and the true posterior for the MA2 model.

E.3.1 Different values of w

In order to understand the behavior of the SR posterior for different choices of w , we run an MCMC chain with 30000 steps of which 10000 are burned in, by using $m = 500$ and $G = 50$, for a range of

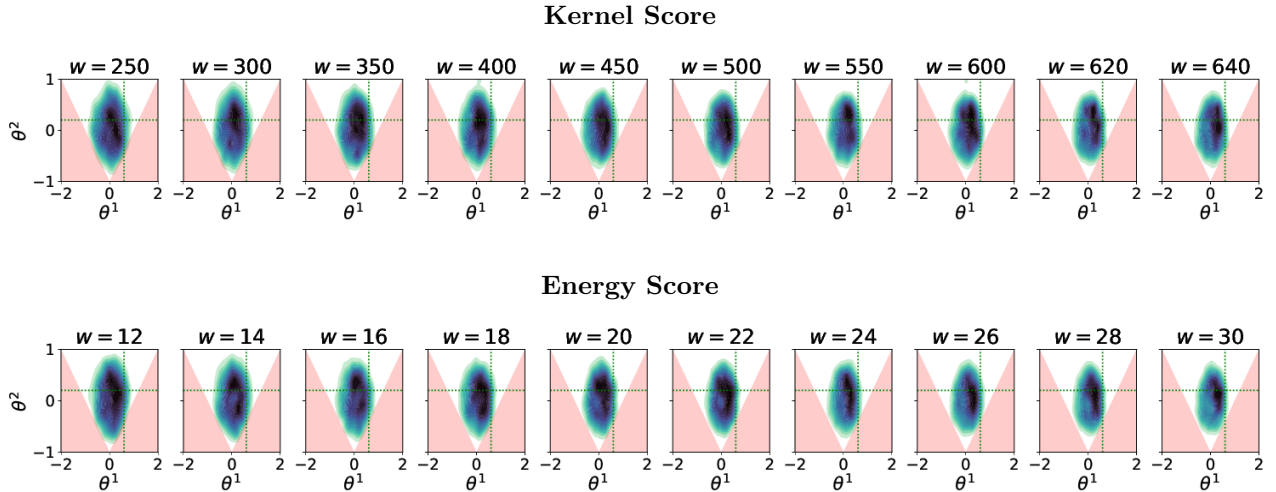


Figure 14: Contour plot for the posterior distributions for the MA(2) model with different values of w , with darker colors denoting larger posterior density and dotted line denoting true parameter value. The posterior densities are obtained by KDE on the MCMC output thinned by a factor 10. The prior distribution is uniform on the white triangular region. We remark how increasing w leads to narrower posteriors, as expected.

values of w . In Table 8, we report the proposal size, acceptance rate and the trace of the posterior covariance matrix for the different weights used, for both the Kernel and Energy Score posteriors. We highlight that increasing w leads to smaller posterior variance (as expected).

The posterior density plots for the different values of w are reported in Figure 14, where it can be seen that increasing w leads to narrower posteriors.

Kernel Score				Energy Score			
w	Prop. size σ	Acc. rate	Tr $[\Sigma_{\text{post}}]$	w	Prop. size σ	Acc. rate	Tr $[\Sigma_{\text{post}}]$
250	1	0.49	0.2828	12	0.3	0.67	0.2738
300	0.9	0.48	0.2653	14	0.3	0.65	0.2582
350	0.8	0.49	0.2537	16	0.3	0.63	0.2409
400	0.7	0.48	0.2437	18	0.3	0.62	0.2280
450	0.6	0.49	0.2273	20	0.15	0.69	0.2195
500	0.5	0.50	0.2192	22	0.15	0.67	0.2080
550	0.4	0.52	0.2164	24	0.15	0.64	0.2074
600	0.3	0.53	0.2160	26	0.15	0.63	0.2042
620	0.15	0.61	0.2068	28	0.1	0.65	0.1928
640	0.15	0.60	0.2037	30	0.1	0.63	0.1880

Table 8: Proposal size, acceptance rate and trace of the posterior covariance matrix for different weight values for MA2, for the Kernel and Energy Score posteriors.

E.4 The M/G/1 model

For the Kernel Score posterior with the Gaussian Kernel, we first fit the value of the Gaussian kernel bandwidth parameter as described in Appendix D, which resulted in $\gamma \approx 3.6439$.

Next, we used our strategy (Sec.2.4) to tune the value of the weight w for both the Kernel and Energy Score Posteriors; this resulted in $w \approx 10.98$ for the Energy Score and $w \approx 797$ for the Kernel Score.

In Table 9 we report the proposal sizes σ and the resulting acceptance rates and trace of the

posterior covariance matrix Σ_{post} for BSL, semiBSL and the SR posteriors with the above values of w ; we also report the trace of Σ_{post} for the true posterior, for which we do not give the proposal size and acceptance rate as it was sampled using more advanced MCMC techniques than standard Metropolis-Hastings [Shestopaloff and Neal, 2014].

Technique	BSL	semiBSL	True posterior	Kernel Score	Energy Score
Proposal size σ	1	0.2	/	0.01	1
Acceptance rate	0.12	0.11	/	0.2697	0.1477
Tr[Σ_{post}]	4.518	0.2726	0.2108	0.336	4.499

Table 9: Proposal sizes and acceptance rates for the BSL, semiBSL, Kernel and Energy Score posteriors, and the true posterior for the M/G/1 model.

E.4.1 Different values of w

In order to understand the behavior of the SR posterior for different choices of w , we run an MCMC chain with 30000 steps of which 10000 are burned in, by using $m = 1000$ and $G = 50$, for a range of values of w . In Table 10, we report the proposal size, acceptance rate and the trace of the posterior covariance matrix for the different weights used, for both the Kernel and Energy Score posteriors. We highlight here that the larger values of w lead to much smaller posterior variance than BSL, and almost as small as semiBSL and the true posterior.

The posterior density plots for the different value of w are reported in Figure 15, where it can be seen that increasing w leads to narrower posteriors.

Kernel Score				Energy Score			
w	Prop. size σ	Acc. rate	Tr [Σ_{post}]	w	Prop. size σ	Acc. rate	Tr [Σ_{post}]
50	1	0.20	5.2641	11	0.9	0.17	4.6423
100	1	0.11	4.5269	14	0.8	0.14	4.1235
150	0.4	0.16	4.0499	17	0.6	0.16	4.0349
200	0.3	0.13	3.2948	20	0.5	0.15	3.8567
250	0.2	0.17	4.6480	23	0.4	0.15	3.8503
300	0.1	0.21	2.7522	26	0.3	0.17	3.2135
350	0.07	0.25	3.9636	29	0.2	0.20	2.2610
400	0.05	0.26	2.5626	32	0.05	0.43	4.4932
450	0.05	0.23	3.3643	35	0.05	0.40	2.9258
500	0.05	0.22	3.7557	38	0.05	0.38	3.3495
550	0.02	0.31	0.6334	41	0.05	0.37	3.6486
600	0.02	0.29	0.4585	44	0.04	0.39	3.6295
700	0.01	0.31	0.3652	47	0.04	0.39	4.9781
800	0.01	0.27	0.2561	50	0.04	0.36	3.2886
900	0.01	0.12	0.1907	53	0.04	0.34	3.3311
-	-	-	-	56	0.01	0.56	0.5347

Table 10: Proposal size, acceptance rate and trace of the posterior covariance matrix for different weight values for M/G/1, for the Kernel and Energy Score posteriors.

E.4.2 Simulating the M/G/1 model

We give here two different recursive formulations of the M/G/1 model which can be used to generate samples from it.

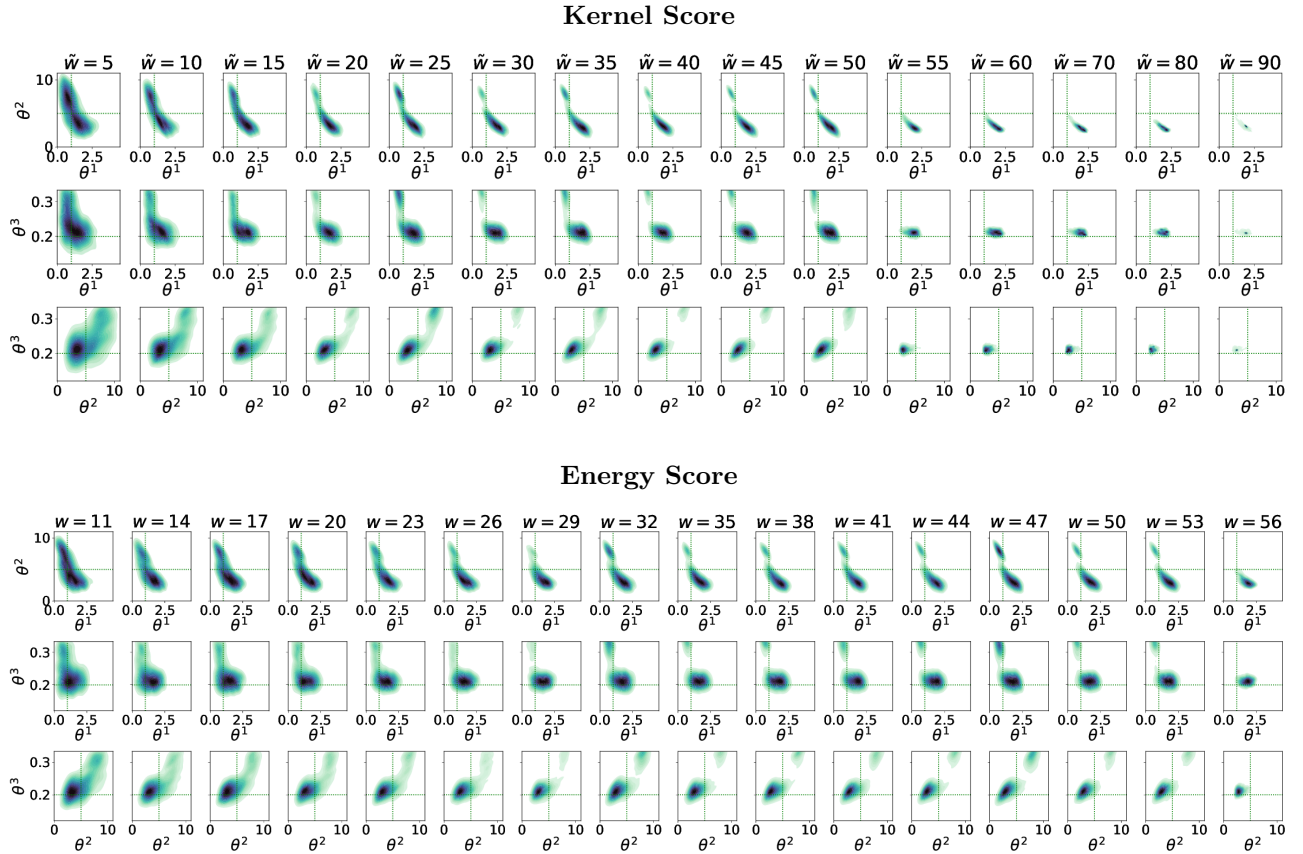


Figure 15: Posterior densities for the Kernel and Energy Score posteriors with different values of w for the M/G/1 model; for both methods, each row shows bivariate marginals for a different pair of parameters, with darker colors denoting larger posterior density and dotted line denoting true parameter value. In the figure for the Kernel Score posterior, we write $\tilde{w} = w/10$ for brevity. The posterior densities are obtained by KDE on the MCMC output thinned by a factor 10. Notice that the axis do not span the full prior range of the parameters. We remark how increasing w leads to narrower posteriors, as expected.

We follow the notation and the model description in Shestopaloff and Neal [2014]. Specifically, we consider customers arriving at a single server with independent interarrival times W_i distributed according to an exponential distribution with parameter θ_3 . The service time U_i is assumed to be $U_i \sim \text{Uni}(\theta_1, \theta_2)$; the observed random variables are the interdeparture times Y_i . In Shestopaloff and Neal [2014], Y_i is written using the following recursive formula:

$$Y_i = U_i + \max \left(0, \sum_{j=1}^i W_j - \sum_{j=1}^{i-1} Y_j \right) = U_i + \max(0, V_i - X_{i-1}), \quad (19)$$

where $V_i = \sum_{j=1}^i W_j$ and $X_i = \sum_{j=1}^i Y_j$ is the departure time are respectively the arrival and departure time of the i -th customer.

A different formulation of the same process is given in Chapter 4.3 in Nelson [2013] by exploiting Lindley's equation, and is of independent interest. We give it here and we show how the two formulations correspond. Specifically, this formulation considers an additional variable Z_i which denotes the waiting time of customer i . For this, a recursion can be obtained to be:

$$Z_i = \max(0, Z_{i-1} + U_{i-1} - W_i),$$

where $Z_0 = 0$ and $U_0 = 0$. Then, the interdeparture time is found to be:

$$Y_i = W_i + U_i - U_{i-1} + Z_i - Z_{i-1}; \quad (20)$$

this can be easily found as the absolute departure time for i -th client is $\sum_{j=1}^i W_j + U_i + Z_i$.

These two formulations are the same; indeed, the latter can be written as:

$$Y_i = U_i + \max(0, Z_{i-1} + U_{i-1} - W_i) - Z_{i-1} - U_{i-1} + W_i = U_i + \max(0, W_i - Z_{i-1} - U_{i-1}). \quad (21)$$

By comparing Eqs. (19) and (21), the two formulations are equal if the following equality is verified:

$$\max(0, W_i - Z_{i-1} - U_{i-1}) = \max \left(0, \sum_{j=1}^i W_j - \sum_{j=1}^{i-1} Y_j \right)$$

which is equivalent to:

$$W_i - Z_{i-1} - U_{i-1} = \sum_{j=1}^i W_j - \sum_{j=1}^{i-1} Y_j \iff Z_{i-1} + U_{i-1} = \sum_{j=1}^{i-1} (Y_j - W_j)$$

Now, from Eq. (20) we have:

$$\sum_{j=1}^{i-1} (Y_j - W_j) = \sum_{j=1}^{i-1} (U_j + Z_j - U_{j-1} - Z_{j-1}) = U_i + Z_i - U_0 - Z_0 = U_i + Z_i,$$

from which the chain of equalities are satisfied.

F Study on values of m

Here, we consider the univariate and multivariate g-and-k, both well specified and misspecified, the MA(2) and the M/G/1 models, and study the impact of varying m in the resulting MCMC target. As we span from very small to large values of m , we use here the vanilla pseudo-marginal MCMC of Andrieu et al. [2009] instead of the correlated pseudo-marginal MCMC which was used for all other simulations.

The choice of m has two different impacts on the MCMC:

1. first, it changes the pseudo-marginal MCMC target, as discussed in Section 2.3 in the main text; recall how, there, we proved that, for $m \rightarrow \infty$, the pseudo-marginal MCMC target converges to the original SR posterior defined in Eq. (2) in the main text. Therefore, we expect, for large enough m , the pseudo-marginal MCMC target to be roughly constant.
2. Additionally, smaller values of m imply that the target estimate has a larger variance. Therefore, we expect sampling to be harder for small m , in terms of acceptance rate of the MCMC, and easier for large m (albeit that is more computationally intensive).

In our simulation study below, we consider m values from 10 to 1000. Our results empirically verify our expectations above. In particular, we find that, for m larger than a threshold which is typically few hundreds, the pseudo-marginal MCMC target is roughly constant. Additionally, very small values of m (few tens) make sampling impractical.

Moreover, our empirical results suggest that larger values of m are required for the MCMC for semiBSL to be stable. For the other methods, the required m seem to be fairly similar, with slightly larger values for BSL for some models.

Typically, we found m values in the few hundreds to strike a good balance between larger computational cost and improved acceptance rate with larger m . Additionally, this consideration depends also on how quickly the simulation cost scales with m : even when not parallelizing model simulations across different processors, if the implementation is vectorized, the computational cost can scale sub-linearly in m , which means a better MCMC efficiency is reached for a larger m . A more extensive study considering for instance the effective sample size per CPU time could be carried out.

In all experiments, except where said otherwise, we use the value of w found via our tuning strategy (Section 2.4 in the main text) and reported above.

F.1 Univariate g-and-k

Here, we report results considering $n = 10$ observations.

m	BSL		Kernel Score		Energy Score	
	Acc. rate	Tr $[\Sigma_{\text{post}}]$	Acc. rate	Tr $[\Sigma_{\text{post}}]$	Acc. rate	Tr $[\Sigma_{\text{post}}]$
10	0.104	4.5245	0.011	3.6030	0.063	3.9822
20	0.122	4.4439	0.035	3.6679	0.115	3.9642
50	0.129	4.3778	0.098	3.3803	0.179	3.6105
100	0.134	4.4095	0.157	3.2220	0.219	3.5335
200	0.136	4.1753	0.204	3.1628	0.243	3.4730
300	0.135	4.2261	0.220	3.1181	0.252	3.3537
400	0.135	4.1769	0.229	3.0716	0.257	3.3553
500	0.132	4.1702	0.234	3.1079	0.262	3.4362
600	0.130	4.2095	0.239	3.0295	0.259	3.2612
700	0.133	4.2417	0.243	3.0536	0.265	3.3629
800	0.132	4.2421	0.247	3.0216	0.265	3.3077
900	0.132	4.1084	0.248	3.0477	0.267	3.3815
1000	0.137	4.2930	0.253	3.1181	0.269	3.3570

Table 11: Acceptance rate and trace of the posterior covariance matrix for different values of m for the well specified univariate g-and-k, for the BSL, Kernel and Energy Score posteriors.

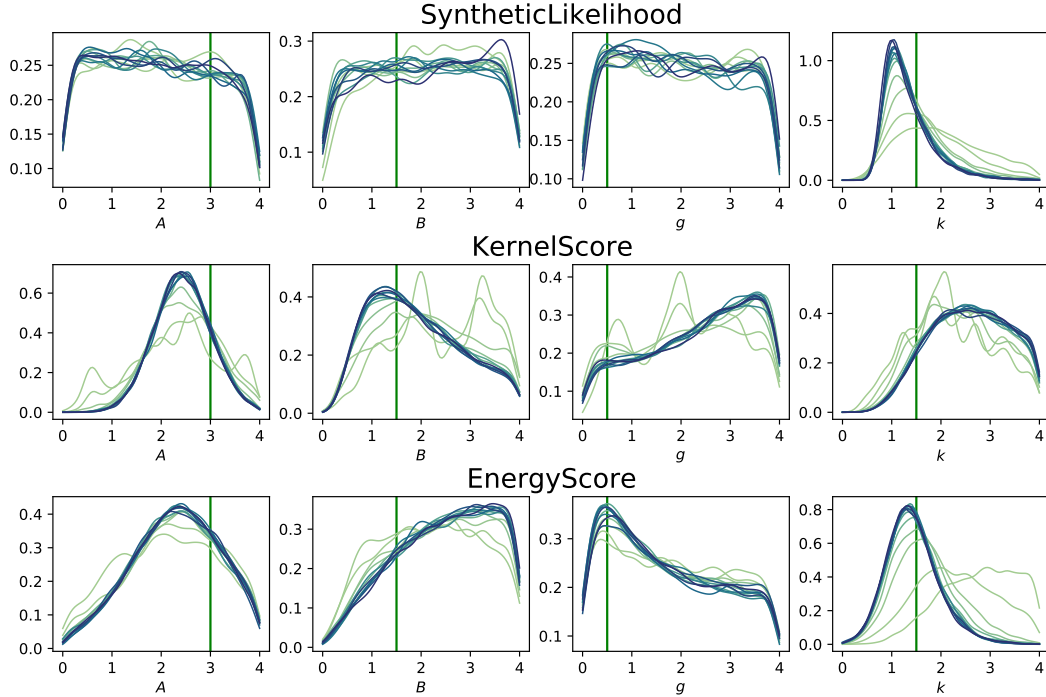


Figure 16: Univariate posterior marginals for different m values for the well specified univariate g-and-k distribution, for the BSL, Kernel and Energy Score posteriors. Lighter (respectively darker) colors denote smaller (resp. larger) values of m . For small values of m , the marginals are spiky, which is due to unstable MCMC. The densities are obtained by KDE on the MCMC output thinned by a factor 10.

F.2 Misspecified univariate g-and-k

Here, we report results considering $n = 10$ observations.

m	BSL		Kernel Score		Energy Score	
	Acc. rate	Tr $[\Sigma_{\text{post}}]$	Acc. rate	Tr $[\Sigma_{\text{post}}]$	Acc. rate	Tr $[\Sigma_{\text{post}}]$
10	0.038	3.3664	0.047	3.4141	0.164	3.8095
20	0.072	2.3207	0.069	3.2060	0.216	3.4900
50	0.130	1.9729	0.184	2.6690	0.306	2.9483
100	0.159	2.0145	0.298	2.4529	0.364	2.7232
200	0.179	1.8829	0.359	2.4037	0.391	2.7153
300	0.187	2.0198	0.389	2.3623	0.402	2.6055
400	0.188	1.9498	0.405	2.3403	0.410	2.6164
500	0.189	1.9092	0.412	2.3756	0.413	2.5579
600	0.191	1.8259	0.422	2.3461	0.414	2.5704
700	0.186	1.9207	0.430	2.3452	0.417	2.5484
800	0.184	1.9509	0.432	2.3810	0.419	2.6276
900	0.190	1.9475	0.434	2.4472	0.423	2.6468
1000	0.194	1.9763	0.436	2.3434	0.425	2.6386

Table 12: Acceptance rate and trace of the posterior covariance matrix for different values of m for the misspecified univariate g-and-k, for the BSL, Kernel and Energy Score posteriors.

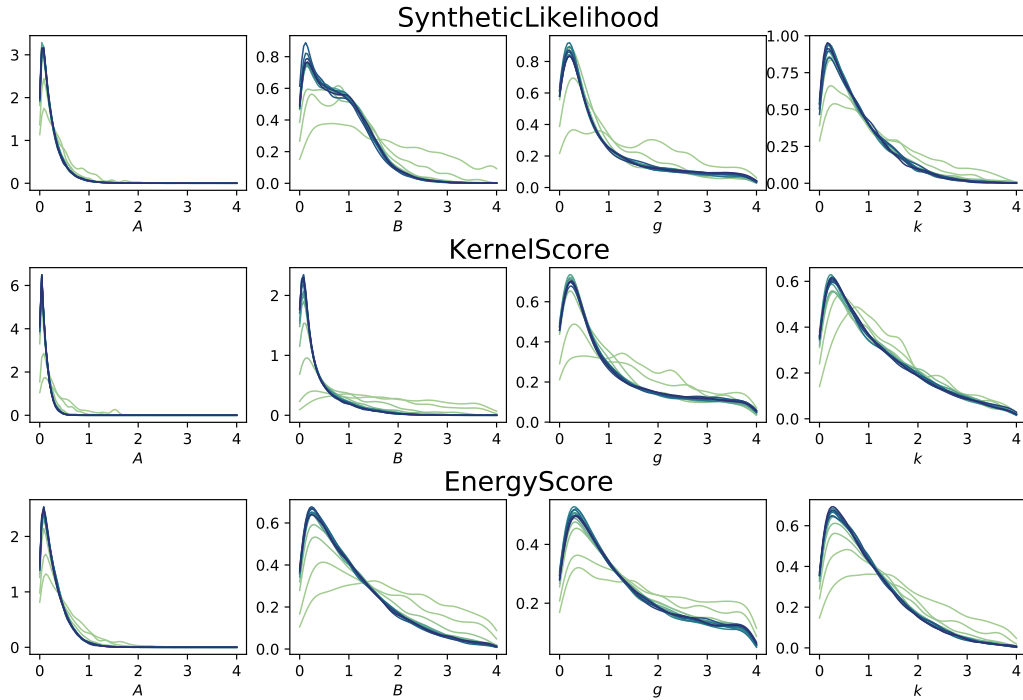


Figure 17: Univariate posterior marginals for different m values for the misspecified univariate g-and-k distribution, for the BSL, Kernel and Energy Score posteriors. Lighter (respectively darker) colors denote smaller (resp. larger) values of m . The densities are obtained by KDE on the MCMC output thinned by a factor 10.

F.3 Multivariate g-and-k

Here, we report results considering $n = 10$ observations.

For this model, small m lead to extremely small acceptance rates for BSL and semiBSL (Table 13); in those cases, the trace of the posterior covariance matrix is also very small due to the chain being almost still. Additionally, even large m values lead to small acceptance rate for semiBSL; that is consequence of the issues discussed in Section 4.1 in the main text and in Appendix E.1.2. We report nevertheless the results here.

m	BSL		semiBSL		Kernel Score		Energy Score	
	Acc. rate	Tr $[\Sigma_{\text{post}}]$	Acc. rate	Tr $[\Sigma_{\text{post}}]$	Acc. rate	Tr $[\Sigma_{\text{post}}]$	Acc. rate	Tr $[\Sigma_{\text{post}}]$
10	<0.001	1.0566	<0.001	0.4227	0.006	3.6061	0.070	4.5255
20	<0.001	0.3674	<0.001	0.6383	0.023	4.0455	0.123	3.9212
50	0.003	2.8320	<0.001	0.6331	0.055	3.8924	0.170	3.8571
100	0.002	2.3666	<0.001	0.6131	0.078	4.1250	0.194	3.8126
200	0.001	0.7140	0.001	0.8603	0.099	3.9624	0.206	3.7142
300	0.008	2.8229	0.002	2.2184	0.108	4.2766	0.208	3.9078
400	0.009	2.5694	0.001	0.6885	0.113	3.9710	0.212	3.8284
500	0.009	3.3583	0.002	1.2885	0.116	4.0250	0.217	3.8383
600	0.013	2.9646	0.005	1.3359	0.120	3.9632	0.216	3.7698
700	0.010	3.7043	0.005	0.6511	0.119	4.0173	0.214	3.7437
800	0.016	3.3017	0.006	0.6679	0.122	3.9607	0.214	3.7512
900	0.022	2.9915	0.005	0.6411	0.126	4.1293	0.216	3.9202
1000	0.017	3.1304	0.006	0.5892	0.122	3.9757	0.216	3.7959

Table 13: Acceptance rate and trace of the posterior covariance matrix for different values of m for the well specified multivariate g-and-k, for the BSL, semiBSL, Kernel and Energy Score posteriors.

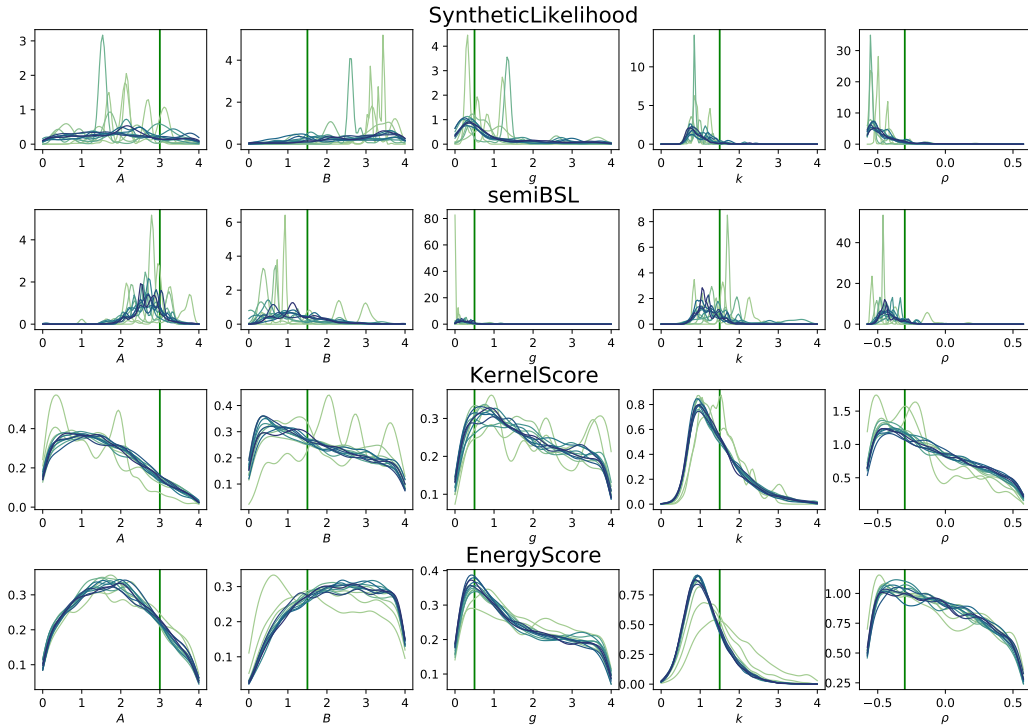


Figure 18: Univariate posterior marginals for different m values for the well specified multivariate g-and-k distribution, for the BSL, semiBSL, Kernel and Energy Score posteriors. Lighter (respectively darker) colors denote smaller (resp. larger) values of m . For small values of m , the marginals are spiky, which is due to unstable MCMC. The densities are obtained by KDE on the MCMC output thinned by a factor 10.

F.4 Misspecified multivariate g-and-k

Here, we report results considering $n = 10$ observations. We do not report results for BSL and semiBSL as those were unable to run satisfactorily for that number of observations, for all considered values of m .

m	Kernel Score		Energy Score	
	Acc. rate	Tr $[\Sigma_{\text{post}}]$	Acc. rate	Tr $[\Sigma_{\text{post}}]$
10	0.017	4.5045	0.174	3.4306
20	0.108	3.6950	0.252	3.2373
50	0.243	3.4612	0.300	3.0291
100	0.308	3.4759	0.316	3.0081
200	0.344	3.4666	0.323	2.9303
300	0.348	3.4583	0.321	2.9160
400	0.355	3.4158	0.331	3.0031
500	0.359	3.4047	0.332	2.9743
600	0.363	3.3847	0.330	2.9321
700	0.360	3.3485	0.329	2.9249
800	0.361	3.3505	0.332	2.9854
900	0.363	3.3627	0.331	3.0155
1000	0.363	3.3307	0.330	2.9277

Table 14: Acceptance rate and trace of the posterior covariance matrix for different values of m for the misspecified multivariate g-and-k, for the Kernel and Energy Score posteriors.

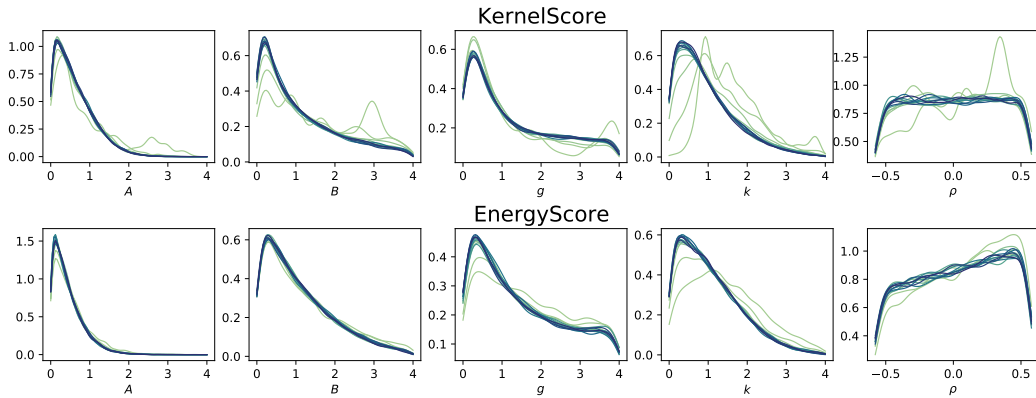


Figure 19: Univariate posterior marginals for different m values for the misspecified multivariate g-and-k distribution, for the Kernel and Energy Score posteriors. Lighter (respectively darker) colors denote smaller (resp. larger) values of m . For small values of m , the marginals are spiky, which is due to unstable MCMC. The densities are obtained by KDE on the MCMC output thinned by a factor 10.

F.5 MA2 model

Here, we consider a single observation ($n = 1$). Notice additionally how small m values lead to very small acceptance rates for all methods except BSL; in those cases, the trace of the posterior covariance matrix is also very small due to the chain being almost still. Moreover, running semiBSL failed altogether for $m = 10$ and $m = 50$ due to numerical errors in the estimate of some quantities. That problem does not happen for larger m values.

m	BSL		semiBSL		Kernel Score		Energy Score	
	Acc. rate	Tr $[\Sigma_{\text{post}}]$	Acc. rate	Tr $[\Sigma_{\text{post}}]$	Acc. rate	Tr $[\Sigma_{\text{post}}]$	Acc. rate	Tr $[\Sigma_{\text{post}}]$
10	0.031	0.1667	nan	nan	<0.001	<0.0001	<0.001	<0.0001
20	0.123	0.2149	<0.001	0.0151	0.008	0.3371	0.001	0.2041
50	0.178	0.1901	nan	nan	0.106	0.3288	0.041	0.2709
100	0.178	0.1392	0.003	0.0214	0.215	0.3189	0.111	0.2748
200	0.168	0.1128	0.041	0.0565	0.303	0.3042	0.203	0.2772
300	0.170	0.0958	0.081	0.0496	0.360	0.3062	0.251	0.2554
400	0.165	0.0890	0.123	0.0505	0.392	0.2974	0.303	0.2516
500	0.157	0.0860	0.155	0.0527	0.414	0.2940	0.330	0.2546
600	0.154	0.0817	0.165	0.0441	0.434	0.2956	0.347	0.2574
700	0.143	0.0749	0.191	0.0458	0.442	0.2975	0.372	0.2617
800	0.137	0.0722	0.209	0.0494	0.446	0.2900	0.375	0.2511
900	0.132	0.0678	0.222	0.0505	0.454	0.2886	0.384	0.2576
1000	0.144	0.0684	0.230	0.0481	0.463	0.2956	0.395	0.2611

Table 15: Acceptance rate and trace of the posterior covariance matrix for different values of m for MA2, for the BSL, semiBSL, Kernel and Energy Score posteriors. nan values corresponds to settings in which semiBSL failed.

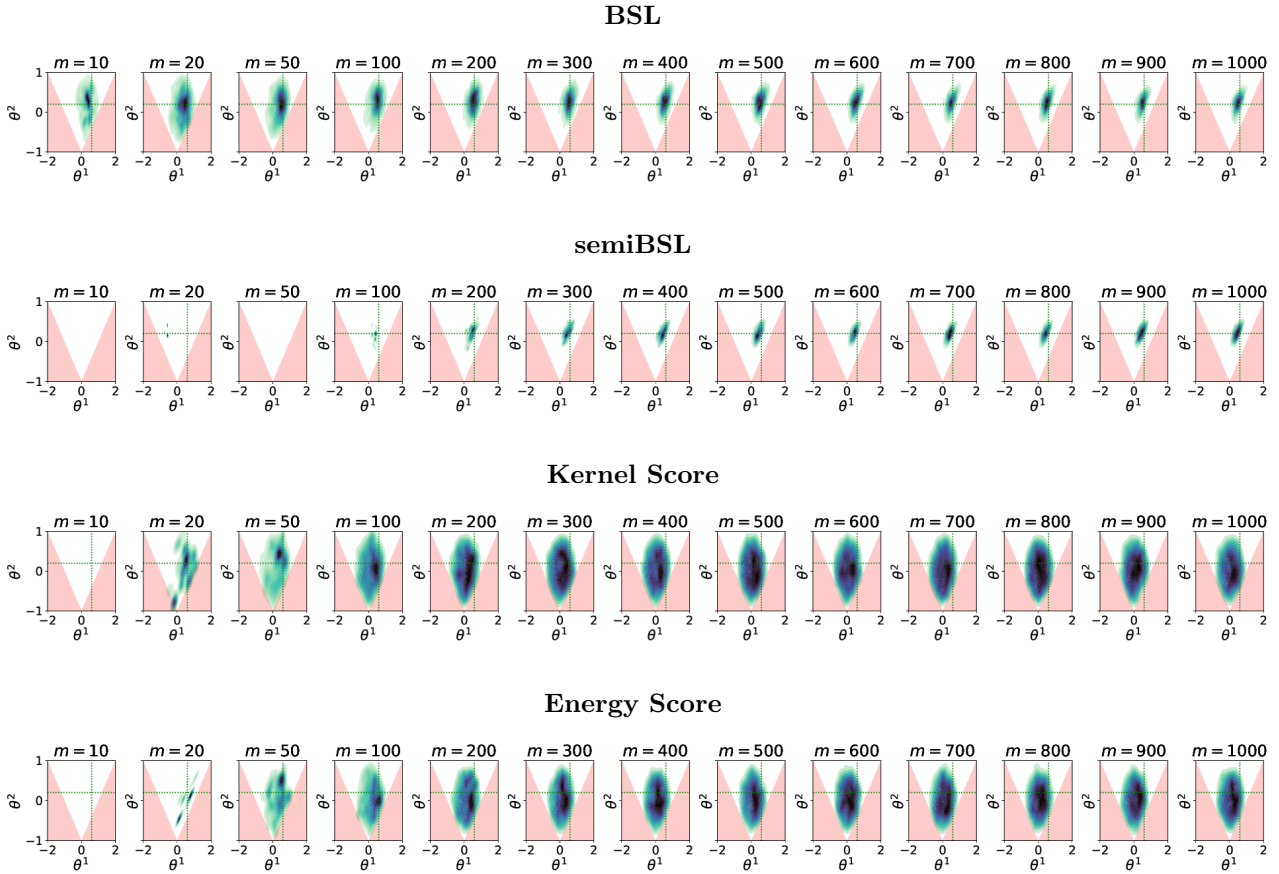


Figure 20: Bivariate posterior marginals for different m values for the MA(2) model, for the BSL, semiBSL, Kernel and Energy Score posteriors. Lighter (respectively darker) colors denote smaller (resp. larger) values of m . For small values of m , the marginals are spiky, which is due to unstable MCMC. Additionally, panels for $m = 10$ and $m = 50$ for semiBSL are empty as MCMC failed for those setups. The densities are obtained by KDE on the MCMC output thinned by a factor 10.

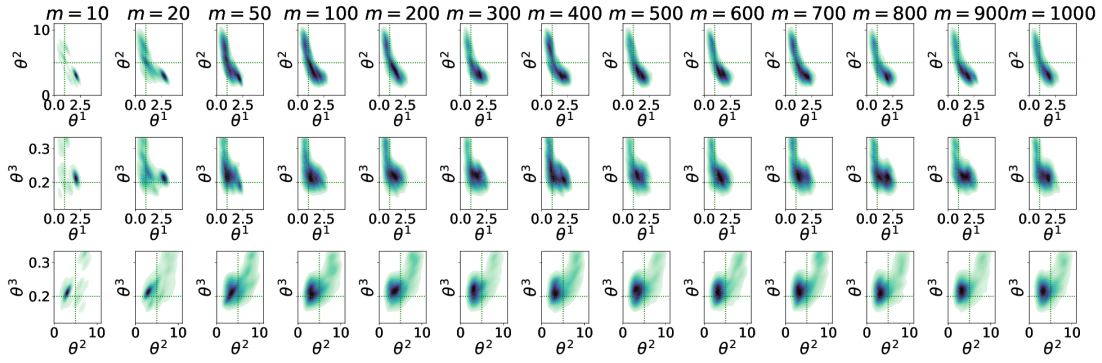
F.6 M/G/1 model

Here, we consider a single observation ($n = 1$). Additionally, we set $w = 200$ for the Kernel Score experiment, which leads to nicer visualizations. Notice additionally how small m values lead to very small acceptance rates for all methods except Energy Score; in some of those cases, the trace of the posterior covariance matrix is also very small due to the chain being almost still. Moreover, running semiBSL failed altogether for $m = 50$ due to numerical errors in the estimate of some quantities. That problem does not happen for larger m values.

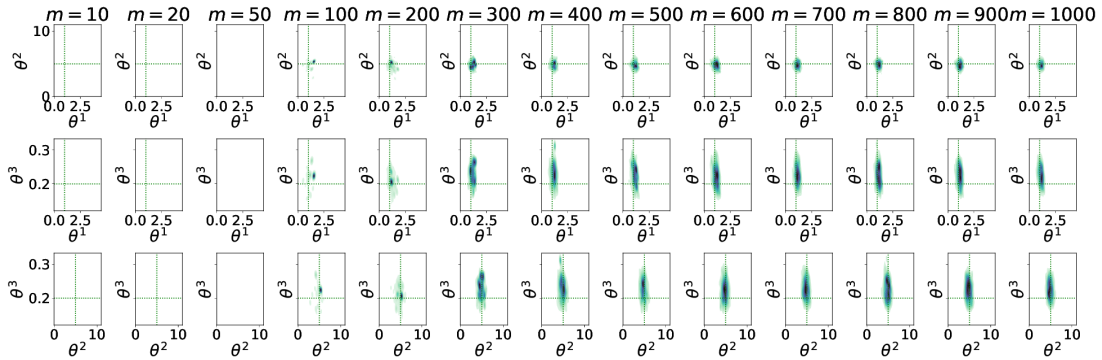
m	BSL		semiBSL		Kernel Score		Energy Score	
	Acc. rate	Tr $[\Sigma_{\text{post}}]$	Acc. rate	Tr $[\Sigma_{\text{post}}]$	Acc. rate	Tr $[\Sigma_{\text{post}}]$	Acc. rate	Tr $[\Sigma_{\text{post}}]$
10	0.004	3.4400	<0.001	1.1318	<0.001	0.0072	0.015	7.1089
20	0.039	4.7629	<0.001	<0.001	0.001	0.4960	0.032	6.9773
50	0.084	4.6943	nan	nan	0.007	0.7041	0.082	4.9889
100	0.100	5.0199	0.002	0.3024	0.048	4.1120	0.104	4.4941
200	0.113	4.9298	0.018	0.3448	0.059	2.4117	0.127	5.0424
300	0.109	4.2174	0.036	0.3350	0.067	1.2283	0.133	4.3495
400	0.115	4.9477	0.066	0.3468	0.099	3.2122	0.142	5.2047
500	0.119	4.5245	0.072	0.3241	0.100	2.9637	0.144	4.7920
600	0.113	4.7229	0.086	0.3383	0.109	3.7342	0.148	5.1760
700	0.112	4.9816	0.091	0.3082	0.120	4.5793	0.146	4.5703
800	0.114	4.7486	0.089	0.2843	0.114	3.1445	0.151	5.1457
900	0.113	4.6482	0.114	0.3123	0.110	2.5299	0.154	4.9276
1000	0.120	4.5183	0.106	0.2726	0.120	3.0236	0.147	4.7525

Table 16: Acceptance rate and trace of the posterior covariance matrix for different values of m for M/G/1, for the BSL, semiBSL, Kernel and Energy Score posteriors. **nan** values corresponds to settings in which semiBSL failed.

BSL



semiBSL



Kernel Score

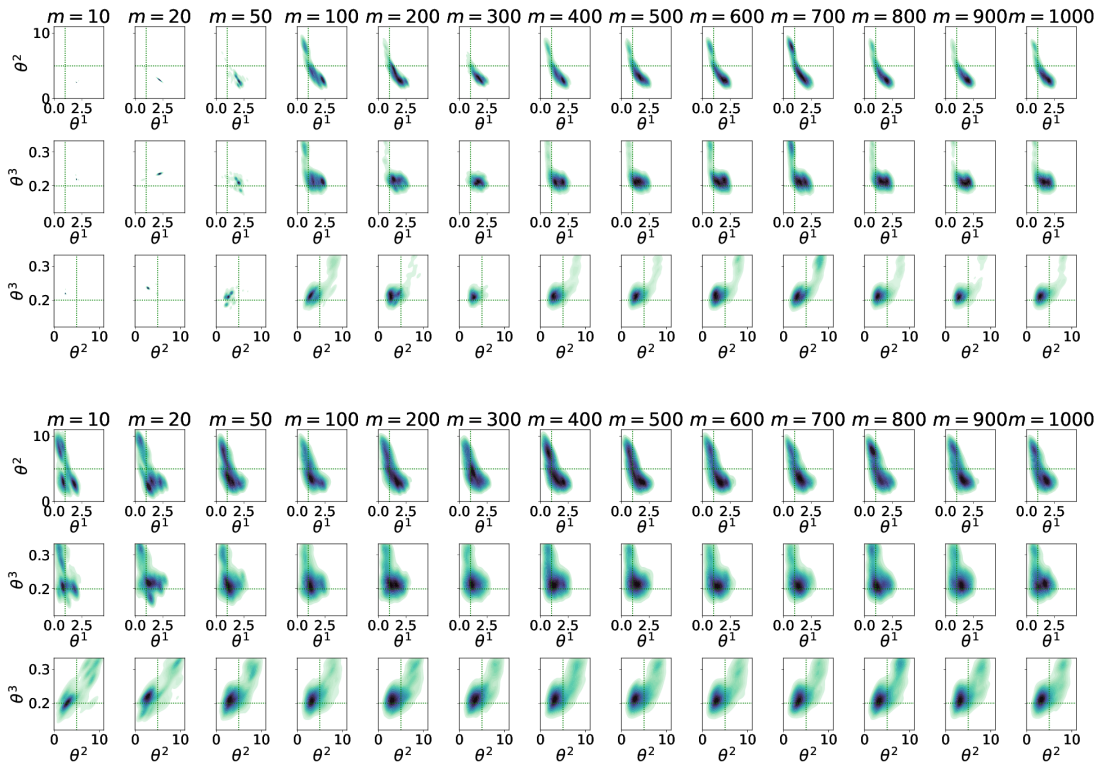


Figure 21: Bivariate posterior marginals for different m values for the M/G/1 model, for the BSL, semiBSL, Kernel and Energy Score posteriors. Lighter (respectively darker) colors denote smaller (resp. larger) values of m . For small values of m , the marginals are spiky, which is due to unstable MCMC. Additionally, panels for $m = 50$ for semiBSL are empty as MCMC failed for those setups. The densities are obtained by KDE on the MCMC output thinned by a factor 10.

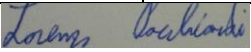
Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).

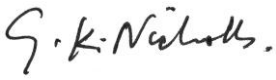
Title of Paper	Generalized Bayesian Likelihood-Free Inference Using Scoring Rules Estimators
Publication Status	<input type="checkbox"/> Published <input type="checkbox"/> Accepted for Publication <input type="checkbox"/> Submitted for Publication in a manuscript style <input checked="" type="checkbox"/> Unpublished and unsubmitted work written
Publication Details	Joint work with Prof. Ritabrata Dutta (University of Warwick).

Student Confirmation

Student Name:	Lorenzo Pacchiardi		
Contribution to the Paper	I am the first author of this paper. I had the original idea of using scoring rules for likelihood-free inference. Discussing with prof. Dutta, this was crystallized in the present form based on generalized Bayesian inference. I obtained the theoretical results, implemented the method, ran the simulations and wrote most of the paper. Prof. Dutta advised along the way and corrected the paper draft.		
Signature		Date	1 st September 2022

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Professor Geoff Nicholls			
Supervisor comments			
Signature		Date	15-09-22

This completed form should be included in the thesis, at the end of the relevant chapter.

Chapter 4

Likelihood-Free Inference with Generative Neural Networks via Scoring Rule Minimization

Unpublished and unsubmitted work.

Likelihood-Free Inference with Generative Neural Networks via Scoring Rule Minimization

Lorenzo Pacchiardi^{1*}, Ritabrata Dutta²

¹*Department of Statistics, University of Oxford, UK*

²*Department of Statistics, University of Warwick, UK*

Abstract

Bayesian Likelihood-Free Inference methods yield posterior approximations for simulator models with intractable likelihood. Recently, many works trained neural networks to approximate either the intractable likelihood or the posterior directly. Most proposals use normalizing flows, namely neural networks parametrizing invertible maps used to transform samples from an underlying base measure; the probability density of the transformed samples is then accessible and the normalizing flow can be trained via maximum likelihood on simulated parameter-observation pairs. A recent work [Ramesh et al., 2022] approximated instead the posterior with generative networks, which drop the invertibility requirement and are thus a more flexible class of distributions scaling to high-dimensional and structured data. However, generative networks only allow sampling from the parametrized distribution; for this reason, Ramesh et al. [2022] follows the common solution of adversarial training, where the generative network plays a min-max game against a “discriminator” network. This procedure is unstable and can lead to a learned distribution underestimating the uncertainty - in extreme cases collapsing to a single point. Here, we propose to approximate the posterior with generative networks trained by Scoring Rule minimization, an overlooked adversarial-free method enabling smooth training and better uncertainty quantification. In simulation studies, the Scoring Rule approach yields better performances with shorter training time with respect to the adversarial framework.

1 Introduction

Intractable-likelihood (or simulator) models are models for which it is impossible to evaluate the likelihood $p(\mathbf{y}|\boldsymbol{\theta})$ for an observation \mathbf{y} , but from which it is easy to simulate for any parameter value $\boldsymbol{\theta}$. Given \mathbf{y} and a prior $\pi(\boldsymbol{\theta})$, the standard Bayesian posterior is $\pi(\boldsymbol{\theta}|\mathbf{y}) \propto \pi(\boldsymbol{\theta})p(\mathbf{y}|\boldsymbol{\theta})$. However, obtaining that explicitly or sampling from it with Markov Chain Monte Carlo (MCMC) is impossible without having access to the likelihood.

Bayesian Likelihood-Free Inference (LFI) techniques exploit model simulations to approximate the exact posterior distribution when the likelihood is unavailable. Popular approaches include Approximate Bayesian Computation methods [Lintusaari et al., 2017, Bernton et al., 2019] and Synthetic Likelihood [Price et al., 2018, An et al., 2020].

A recent strand of literature [Papamakarios and Murray, 2016, Lueckmann et al., 2017, Papamakarios et al., 2019, Lueckmann et al., 2019, Greenberg et al., 2019, Durkan et al., 2020, Radev et al., 2020] has explored using neural networks to perform LFI¹. Many methods employ normalizing flows [Papamakarios et al., 2021]: invertible neural networks which parametrize complex probability distributions by transforming samples from a simple one (say, multivariate Gaussian). Normalizing flows allow evaluating the density of the distribution via the change-of-variables formula enabled by invertibility; using the latter, normalizing flows can be trained via maximum likelihood estimation on parameter-simulation pairs. They can be used to represent either the likelihood [Papamakarios et al., 2019, Lueckmann et al., 2019] or the posterior [Papamakarios and Murray, 2016, Lueckmann et al., 2017, Greenberg et al., 2019, Radev et al., 2020].

*Corresponding author: lorenzo.pacchiardi@stats.ox.ac.uk.

¹Check <https://neurallikelihoodfreeinference.github.io/> for an interactive up-to-date list.

However, imposing invertibility strongly constrains the network architecture. More general generative networks drop this requirement, thus gaining expressiveness and the ability to easily scale to large input and output sizes, but forgoing density evaluation: from a generative network, you can only obtain draws from the parametrized probability distribution. For this reason, maximum likelihood estimation of neural network weights is impossible and people use training methods based on generating samples from the generative network. A paradigmatic example is the popular Generative Adversarial Network, or GAN, framework of Goodfellow et al. [2014], where the generative network is trained in a min-max game against an additional *discriminator* network aiming at distinguishing between training samples and simulations from the generative network².

For LFI, Ramesh et al. [2022] used a generative network to represent a posterior approximation and trained it with an adversarial approach. From the trained network, approximate posterior samples can be directly obtained. Here, we build on Ramesh et al. [2022] by proposing a different training strategy based on minimizing values of Scoring Rules (SR, Gneiting and Raftery, 2007), which are functions assessing the match between a probability distribution and an observation. In contrast to the adversarial approach, often leading to mode collapse (in which the parametrized probability distribution collapses on a single point, see Arora et al., 2017, Bellemare et al., 2017, Arora et al., 2018, Richardson and Weiss, 2018), SR training has been found to better capture the full shape of the probability distribution in the setting of probabilistic forecasting (Chapter 5 of the present thesis). Additionally, the SR-minimization approach leads to simpler training with respect to the adversarial one, as it does not require a discriminator or solving a min-max problem.

The rest of the paper is organized as follows. Section 2 discusses how to use a generative network to represent and approximate posterior and reviews the training method employed in Ramesh et al. [2022]. Section 3 introduces SR-minimization training for LFI. Section 4 reports simulation results and Section 5 gives concluding remarks.

Notation We will denote respectively by $\mathcal{Y} \subseteq \mathbb{R}^d$ and $\Theta \subseteq \mathbb{R}^p$ the data and parameter space. We will use $P(\cdot|\boldsymbol{\theta})$ and $p(\cdot|\boldsymbol{\theta})$ to denote the distribution and likelihood (with respect to Lebesgue measure) of the considered likelihood-free model. Π and π will denote prior distribution and prior density on Θ , and $\Pi(\cdot|\mathbf{y})$ and $\pi(\cdot|\mathbf{y})$ will denote corresponding posterior quantities for observation \mathbf{y} . In general, we will use P or Q to denote distributions, while S will denote a generic Scoring Rule. Other upper-case letters (\mathbf{X}, \mathbf{Y} and \mathbf{Z}) will denote random variables while lower-case ones will denote observed (fixed) values. We will denote by \mathbf{Y} or \mathbf{y} the observations (correspondingly random variables and realizations). Bold symbols denote vectors, and subscripts to bold symbols denote sample index (for instance, \mathbf{y}_i). Instead, subscripts to normal symbols denote component indices (for instance, y_j is the j -th component of \mathbf{y} , and $y_{i,j}$ is the j -th component of \mathbf{y}_i). Finally, \perp will denote independence between random variables, while $\mathbf{Y} \sim P$ indicates a random variable distributed according to P and $\mathbf{y} \sim P$ a sample from such random variable.

2 Approximate posterior via generative network

We use a generative network to represent an approximate posterior distribution $Q_\phi(\cdot|\mathbf{y})$ on the parameter space Θ given an observation $\mathbf{y} \in \mathcal{Y}$. The density of $Q_\phi(\cdot|\mathbf{y})$ (with respect to the Lebesgue measure) will be denoted by $q_\phi(\cdot|\mathbf{y})$. The generative network is defined via a neural network $g_\phi: \mathcal{Z} \times \mathcal{Y} \rightarrow \Theta$ transforming samples from a probability distribution $P_{\mathbf{z}}$ over the space \mathcal{Z} conditionally on an observation $\mathbf{y} \in \mathcal{Y}$; ϕ represents neural network weights. Samples from $Q_\phi(\cdot|\mathbf{y})$ are therefore obtained by sampling $\mathbf{z} \sim P_{\mathbf{z}}$ and computing $\boldsymbol{\theta} = g_\phi(\mathbf{z}, \mathbf{y}) \sim Q_\phi(\cdot|\mathbf{y})$ ³

In the following, as it is standard in the LFI setup, we assume to have access to parameter-simulations pairs $(\boldsymbol{\theta}_i, \mathbf{y}_i)_{i=1}^n$ generated from the prior $\boldsymbol{\theta}_i \sim \Pi$ and the model $\mathbf{y}_i \sim P(\cdot|\boldsymbol{\theta}_i)$; critically, these can also be considered as being samples from the data marginal $\mathbf{y}_i \sim P$ and the posterior

²Notice that generative networks and adversarial training are older techniques than normalizing flows. We introduce them in this order by following their usage in the LFI context.

³Formally, $Q_\phi(\cdot|\mathbf{y})$ is the push-forward of $P_{\mathbf{z}}$ through the map $g_\phi(\cdot, \mathbf{y}): Q_\phi(\cdot|\mathbf{y}) = g_\phi(\cdot, \mathbf{y})\#P_{\mathbf{z}}$, which means that, for any set A belonging to the Borel σ -algebra $\sigma(\Theta)$, $Q_\phi(A|\mathbf{y}) = P_{\mathbf{z}}(\{\mathbf{z} \in \mathcal{Z} : g_\phi(\mathbf{z}, \mathbf{y}) \in A\})$.

$\theta_i \sim \Pi(\cdot|\mathbf{y}_i)$. Using these samples, we want to tune ϕ such that $Q_\phi(\cdot|\mathbf{y}) \approx \Pi(\cdot|\mathbf{y})$ for all values of \mathbf{y} ; this is therefore an *amortized* setting [Radev et al., 2020], namely the resulting posterior approximation is valid for multiple observations.

In the amortized setting, a single neural network has to map the observation into a posterior for all possible observations; intuitively, we expect this to work well for those cases where such inversion process is in some sense “generic”. In contrast, the amortized approach will perform poorly when the posterior distribution depends on the data in a non-linear way. Additionally, the amortized approach may be wasteful in terms of model simulations when inference for a single observation is needed, as the simulations from the likelihood-free model are drawn independently from it, so that many will be uninformative. In Sec 3.4, we discuss strategies for tailoring simulations to a specific observation.

2.1 Adversarial posterior inference

In Ramesh et al. [2022], the posterior approximation Q_ϕ was trained in an adversarial framework. This requires introducing a *discriminator* or *critic* neural network $c_\psi : \Theta \times \mathcal{Y} \rightarrow \mathbb{R}$ with weights ψ whose task is to distinguish draws from the approximate and true posteriors. The loss employed in Ramesh et al. [2022] is the conditional version of the original GAN loss from Goodfellow et al. [2014], which was originally discussed in Mirza and Osindero [2014]:

$$\begin{aligned} L(\phi, \psi) &= \mathbb{E}_{\theta \sim \Pi} \mathbb{E}_{\mathbf{Y} \sim P(\cdot|\theta)} \mathbb{E}_{\mathbf{Z} \sim P_{\mathbf{z}}} [\log c_\psi(\theta, \mathbf{Y}) + \log(1 - c_\psi(g_\phi(\mathbf{Z}, \mathbf{Y}), \mathbf{Y}))] \\ &= \mathbb{E}_{\mathbf{Y} \sim P} \left[\mathbb{E}_{\theta \sim \Pi(\cdot|\mathbf{Y})} (\log c_\psi(\theta, \mathbf{Y})) + \mathbb{E}_{\tilde{\theta} \sim Q_\phi(\cdot|\mathbf{Y})} \left(\log \left(1 - c_\psi(\tilde{\theta}, \mathbf{Y}) \right) \right) \right], \end{aligned} \quad (1)$$

whose saddle point solution

$$\min_{\phi} \max_{\psi} L(\phi, \psi) \quad (2)$$

leads to $Q_\phi(\cdot|\mathbf{y})$ being the exact posterior for all choices of \mathbf{y} for which $p(\mathbf{y}) > 0$ (provided g_ϕ and c_ψ have infinite expressive power; that in fact corresponds to minimizing the Jensen-Shannon divergence, see Appendix B.1 in Chapter 5 of the present thesis).

The practical training procedure employs Stochastic Gradient Descent (SGD): we replace the expectations in Eq. (1) with empirical means over (a mini-batch of) the training dataset and draws from the generative network and alternate maximization steps over ψ with minimization steps over ϕ . This alternating optimization is however unstable and requires careful hyperparameters tuning and specialized training routines [Salimans et al., 2016]. It has been empirically shown [Arora et al., 2018, Isola et al., 2017, Richardson and Weiss, 2018] that adversarial training can lead to *mode collapse*, in which the distribution parametrized by the generative network collapses onto a single point. Arora et al. [2017] showed how mode collapse can happen due to the finite capacity of the discriminator, while Bellemare et al. [2017] theoretically linked it to the use of biased gradient estimates for ϕ in optimizing Eq. (2) (in fact, we compute gradients with respect to ϕ relying on a value of ψ obtained by few optimization steps, rather than the value maximizing eq. 1).

Mode collapse may not be an issue in some applications of generative networks where uncertainty quantification is not important, but it can be detrimental for approximate posterior inference.

3 Posterior inference via Scoring Rules minimization

We discuss here the use of Scoring Rules to define an adversarial-free training objective for generative networks, focusing on the specific case of a generative network parametrizing an approximate posterior. In Chapter 5 of the present thesis, SR-training for probabilistic forecasting can be found. Other works employing SR training, albeit not for the LFI framework, are Bouchacourt et al. [2016], Gritsenko et al. [2020], Harakeh and Waslander [2021].

We first introduce Scoring Rules for a distribution P related to a generic random variable \mathbf{X} . A Scoring Rule (SR, Gneiting and Raftery, 2007) $S(P, \mathbf{x})$ is a function of P and of an observation \mathbf{x} of the random variable \mathbf{X} . If \mathbf{X} is distributed according to Q , the expected Scoring Rule is defined as:

$$S(P, Q) := \mathbb{E}_{\mathbf{X} \sim Q} S(P, \mathbf{X}),$$

The Scoring Rule S is *proper* relative to a set of distributions \mathcal{P} over \mathcal{X} if

$$S(Q, Q) \leq S(P, Q) \quad \forall P, Q \in \mathcal{P},$$

i.e., if the expected Scoring Rule is minimized in P when $P = Q$. Moreover, S is *strictly proper* relative to \mathcal{P} if $P = Q$ is the unique minimum:

$$S(Q, Q) < S(P, Q) \quad \forall P, Q \in \mathcal{P} \text{ s.t. } P \neq Q.$$

3.1 Patched SRs

In practice, we will use the Energy and Kernel Scores introduced in Sec. 1.2.2 in the introduction of the present thesis, fixing $\beta = 1$ for the Energy Score and use the Gaussian kernel in the Kernel Score. However, those scores consider \mathbf{X} as a multivariate variable, where the individual entries can be permuted. Some of the examples in Sec. 4 generate however data on a 1D or 2D grid and the Energy and Kernel Scores would be oblivious to such structure. To better represent it, we propose computing the SRs on localized *patches* across the grid and cumulate the score; in this way, short-scale correlations are given more importance. For a given SR S , therefore, the patched SR is:

$$S_p(P, \mathbf{x}) = w_1 S(P, \mathbf{x}) + w_2 \sum_{p \in \mathcal{P}} S(P|_p, \mathbf{x}|_p),$$

where $w_1, w_2 > 0$, $|_p$ denotes the restriction of a distribution or of a vector to a patch p and \mathcal{P} is a set of patches.

However, the resulting Scoring Rule is not strictly proper; to fix this, we add the SR computed over the full \mathbf{x} , which makes the overall SR strictly proper, as shown in the following Lemma (proven in Appendix A):

Lemma 1. *Consider two proper SRs S_1 and S_2 , and let $\alpha_1, \alpha_2 > 0$; the quantity:*

$$S_+(P, \mathbf{y}) = \alpha_1 \cdot S_1(P, \mathbf{y}) + \alpha_2 \cdot S_2(P, \mathbf{y})$$

is a proper SR. If at least one of S_1 and S_2 is also strictly proper, then S_+ is strictly proper.

In Chapter 5 of the present thesis, another SR capturing spatial information based on the variogram [Scheuerer and Hamill, 2015] is tested; as that performed worse than the patched one, we exclude it from our analysis.

3.2 Scoring Rule training

Let us now go back to the Bayesian LFI setting introduced at the start of the paper. Denoting by $Q_\phi(\cdot|\mathbf{y})$ the approximate posterior parametrized by the generative network, solving the following problem for a strictly proper S :

$$\arg \min_{\phi} \mathbb{E}_{\mathbf{Y} \sim P} \mathbb{E}_{\theta \sim \Pi(\cdot|\mathbf{Y})} S(Q_\phi(\cdot|\mathbf{Y}), \theta) = \arg \min_{\phi} \mathbb{E}_{\theta \sim \Pi} \mathbb{E}_{\mathbf{Y} \sim P(\cdot|\theta)} S(Q_\phi(\cdot|\mathbf{Y}), \theta) \quad (3)$$

leads to $q_\phi(\cdot|\mathbf{y}) = \pi(\cdot|\mathbf{y})$ for all values of \mathbf{y} for which $p(\mathbf{y}) > 0$.

An empirical version of Eq. (3) is obtained by replacing the expectations with empirical means over the training dataset:

$$\arg \min_{\phi} \frac{1}{n} \sum_{i=1}^n S(Q_\phi(\cdot|\mathbf{y}_i), \theta_i); \quad (4)$$

computing the objective directly is intractable as, in general, we do not have access to $S(Q_\phi(\cdot|\mathbf{y}), \theta)$. Notice, however, that in order to solve Eq. (4) via SGD it is enough to obtain unbiased estimates of $\nabla_{\phi} S(Q_\phi(\cdot|\mathbf{y}_i), \theta_i)$, which can be easily done whenever S admits an unbiased empirical estimator \hat{S} such that:

$$\mathbb{E} \left[\hat{S}(\{\tilde{\theta}_j^{(\mathbf{y})}\}_{j=1}^m, \theta) \right] = S(Q_\phi(\cdot|\mathbf{y}), \theta),$$

where the expectation is over $\tilde{\theta}_j^{(\mathbf{y})} \sim Q_\phi(\cdot|\mathbf{y})$. More details can be found in Appendix B. If S admits such an estimator, each step of SGD involves generating m simulations from the generative network $Q_\phi(\cdot|\mathbf{y}_i)$ for each \mathbf{y}_i in the training batch.

For the Energy and Kernel Scores introduced in Sec. 1.2.2 in the introduction of the present thesis, unbiased estimators are available (see Sec. 1.2.2 in the introduction). These estimators require $m > 1$; to train GAN, instead, a single draw from the generative network for each training sample at each SGD step is enough. In experiments, however, small values of m ($m = 10$ for instance) lead to satisfactory results. Additionally, as mentioned above, the SR approach does not require a discriminator network and has a smoother training process, which implies convergence is generally reached with less training epochs. These two factors lead to lower computational and memory cost with respect to adversarial training (see Section 4 for details).

3.3 Connection with normalizing flows

As mentioned in the introduction, normalizing flows are generative networks which impose invertibility of the map $g_\phi(\mathbf{z}, \mathbf{y})$ with respect to \mathbf{z} . As such, density evaluation of the resulting q_ϕ is possible via the change-of-variables formula, so that ϕ is usually trained via maximum likelihood [Papamakarios et al., 2021]. For instance, in Radev et al. [2020], the following problem was considered, where D_{KL} denotes the Kullback-Leibler divergence:

$$\begin{aligned} & \underset{\phi}{\operatorname{argmin}} \mathbb{E}_{\mathbf{Y} \sim P} [D_{KL}(\Pi(\cdot | \mathbf{Y}) \| Q_\phi(\cdot | \mathbf{Y}))] \\ &= \underset{\phi}{\operatorname{argmin}} \mathbb{E}_{\mathbf{Y} \sim P} \mathbb{E}_{\theta \sim \Pi(\cdot | \mathbf{Y})} [-\log q_\phi(\theta | \mathbf{Y})] \\ &= \underset{\phi}{\operatorname{argmin}} \mathbb{E}_{\theta \sim \Pi} \mathbb{E}_{\mathbf{Y} \sim P(\cdot | \theta)} [-\log q_\phi(\theta | \mathbf{Y})], \end{aligned}$$

which corresponds to our SR-based approach in Eq. (3) by identifying $S(Q_\phi(\cdot|\mathbf{y}), \theta) = -\log q_\phi(\theta|\mathbf{y})$, which is the strictly-proper logarithmic scoring rule [Gneiting and Raftery, 2007].

3.4 Sequential training

Up to this point, we have considered the training data from the simulator model $(\theta_i, \mathbf{y}_i)_{i=1}^n$ to be generated independently from the observation on which inference is performed; under this assumption, we have discussed ways to learn posterior approximations valid for all values of \mathbf{y} such that $p(\mathbf{y}) > 0$. Once the neural network is trained, therefore, inference can be performed for as many observations as we wish. This is a so-called *amortized* setup [Radev et al., 2020].

However, practitioners may require posterior inference for a single observation \mathbf{y}_o . What they are interested in, therefore, is the quality of the approximation for values of θ with large posterior density for the observed \mathbf{y}_o . In this case, generating training samples independently from \mathbf{y}_o may be wasteful: a more efficient method (in terms of simulations from the model $p(\cdot|\theta)$) would generate more training samples θ_i 's close to the modes of the true posterior, as those convey more information on the precise posterior shape. This can be done in a sequential fashion: given a small amount of training data, a first approximation Q_{ϕ_1} is obtained; from that, additional training samples (θ_i, \mathbf{y}_i) are generated by $\theta_i \sim Q_{\phi_1}(\cdot|\mathbf{y}_o)$, $\mathbf{y}_i \sim P(\cdot|\theta_i)$ and used to (re-)train an approximation Q_{ϕ_2} . This procedure is iterated several times, allowing the training samples to progressively focus around the posterior modes and thus refining the approximation [Lueckmann et al., 2017, Greenberg et al., 2019].

However, naively following that strategy is incorrect. To see this, assume that, at the second round, we just train on samples drawn from the approximate posterior $\tilde{\Pi} = Q_{\phi_1}(\cdot|\mathbf{y}_o)$ obtained at the first round. Such a sampled pair (θ_i, \mathbf{y}_i) was drawn from a joint density $\tilde{\pi}(\theta_i)p(\mathbf{y}_i|\theta_i) = \tilde{p}(\mathbf{y}_i)\tilde{\pi}(\theta_i|\mathbf{y}_i)$, where $\tilde{\pi}$ on the left-hand side of the equality is the density of the proposal $\tilde{\Pi}$ and the quantities on the right-hand side are univocally defined by the left-hand side. The optimal ϕ^* obtained via SR-minimization thus corresponds to $q_{\phi^*}(\cdot|\mathbf{y}) = \tilde{\pi}(\cdot|\mathbf{y})$, which is not the correct target.

The traditional way to fix this entails introducing importance weights in the training objective

(Eq. 3):

$$\mathbb{E}_{\boldsymbol{\theta} \sim \Pi} \mathbb{E}_{\mathbf{Y} \sim P(\cdot|\boldsymbol{\theta})} S(Q_\phi(\cdot|\mathbf{Y}), \boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{\theta} \sim \tilde{\Pi}} \frac{\pi(\boldsymbol{\theta})}{\tilde{\pi}(\boldsymbol{\theta})} \mathbb{E}_{\mathbf{Y} \sim P(\cdot|\boldsymbol{\theta})} S(Q_\phi(\cdot|\mathbf{Y}), \boldsymbol{\theta}).$$

As $\tilde{\pi}(\boldsymbol{\theta})$ cannot be evaluated, a solution is to fit a probabilistic classifier (at each round of the sequential procedure) to samples from $\pi(\boldsymbol{\theta})$ and $\tilde{\pi}(\boldsymbol{\theta})$ and use it to estimate the ratio $\frac{\pi(\boldsymbol{\theta})}{\tilde{\pi}(\boldsymbol{\theta})}$. This classifier is not required for the normalizing flows approaches, where the ratio can be evaluated explicitly [Lueckmann et al., 2017, Greenberg et al., 2019] (unless the prior π is also defined implicitly, as in the camera model example in Section 4). For the GAN approach, a similar importance weights approach requires additionally to estimate the ratio $\frac{\tilde{p}(\mathbf{y})}{p(\mathbf{y})}$ [Ramesh et al., 2022].

An alternative approach, which was applied to the GAN approach in Ramesh et al. [2022], involves correcting the distribution of the variable \mathbf{Z} which is transformed by the generative network. Specifically, Ramesh et al. [2022] showed that $\pi(\boldsymbol{\theta}|\mathbf{y}) = \tilde{\pi}(\boldsymbol{\theta}|\mathbf{y})w(\boldsymbol{\theta}, \mathbf{y}) \iff \tilde{\pi}(\boldsymbol{\theta}|\mathbf{y}) = \pi(\boldsymbol{\theta}|\mathbf{y})(w(\boldsymbol{\theta}, \mathbf{y}))^{-1}$, where $w(\boldsymbol{\theta}, \mathbf{y}) = \frac{\pi(\boldsymbol{\theta}) \tilde{p}(\mathbf{y})}{\tilde{\pi}(\boldsymbol{\theta}) p(\mathbf{y})}$. Therefore you can consider a modified approximation $\tilde{Q}_\phi(\cdot|\mathbf{Y})$ and a new training objective:

$$\mathbb{E}_{\mathbf{Y} \sim \tilde{P}} \mathbb{E}_{\boldsymbol{\theta} \sim \tilde{\Pi}(\cdot|\mathbf{Y})} S(\tilde{Q}_\phi(\cdot|\mathbf{Y}), \boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{\theta} \sim \tilde{\Pi}} \mathbb{E}_{\mathbf{Y} \sim P(\cdot|\boldsymbol{\theta})} S(\tilde{Q}_\phi(\cdot|\mathbf{Y}), \boldsymbol{\theta}) \quad (5)$$

whose minimization leads to $\tilde{Q}_\phi(\cdot|\mathbf{Y}) = \tilde{\Pi}(\cdot|\mathbf{Y})$. By setting

$$\tilde{Q}_\phi(\cdot|\mathbf{Y}) = Q_\phi(\cdot|\mathbf{Y})(w(\boldsymbol{\theta}, \mathbf{y}))^{-1},$$

you ensure $Q_\phi(\cdot|\mathbf{Y}) = \Pi(\cdot|\mathbf{Y})$. To train ϕ using the objective in Eq. (5), draws from $\tilde{Q}_\phi(\cdot|\mathbf{Y})$ are required; those can be obtained by sampling $\mathbf{z} \sim \tilde{P}_\mathbf{z}$, whose density is $\tilde{p}_\mathbf{z}(\mathbf{z}) = p_\mathbf{z}(\mathbf{z})(w(g_\phi(\mathbf{z}, \mathbf{y}), \mathbf{y}))^{-1}$, and computing $\boldsymbol{\theta} = g_\phi(\mathbf{z}, \mathbf{y})$, which is thus a sample from $\tilde{Q}_\phi(\cdot|\mathbf{Y})$. Compared to using importance weights, the variance of the training objective is here smaller. However, rejection sampling or MCMC are needed to sample from $\tilde{P}_\mathbf{z}$, and two ratios have to be estimated via probabilistic classifiers ($\frac{\tilde{p}(\mathbf{y})}{p(\mathbf{y})}$ and $\frac{\pi(\boldsymbol{\theta})}{\tilde{\pi}(\boldsymbol{\theta})}$), making this strategy more expensive than using importance weights

On the examples considered in Ramesh et al. [2022], the sequential approaches did not perform better than the amortized one, mainly due to the additional computational cost associated to estimating the ratios. As we employ the same examples here, we do not test these methods, but we discussed them anyway as they may turn out to be useful in other applications.

4 Simulation study

Following Ramesh et al. [2022], we present here results on two benchmark problems and two high-dimensional models, one of which has an implicitly defined prior. For all examples, we evaluate the performance of the different methods as in Ramesh et al. [2022]. Besides that, we assess the calibration of the approximate posteriors by the discrepancy between credible intervals in the approximate posteriors and the frequency with which the true parameter belongs to the credible interval itself (we call this metric *calibration error*). We also evaluate how close the posterior means are to the true parameter value by the *Normalized Root Mean-Square Error* (NRMSE) and the *coefficient of determination* R^2 ; these metrics were used for LFI in Radev et al. [2020]; we provide more detail in Appendix C. As all these metrics are for scalar variables, we compute their values independently for each component of $\boldsymbol{\theta}$ and report their average.

We compare our generative networks trained with SRs with the GAN-based one of Ramesh et al. [2022]; in both setups, we adapt the generative networks defined in Ramesh et al. [2022] for the different tasks. Additional training details for all models are reported in Appendix D. Notice how Ramesh et al. [2022] compared with additional LFI methods, concluding that the generative-network based one performs worse for the simple models but is competitive for the high-dimensional ones. Here, we do not compare with these other methods as the focus of our paper is to provide a different training strategy for the generative network approach.

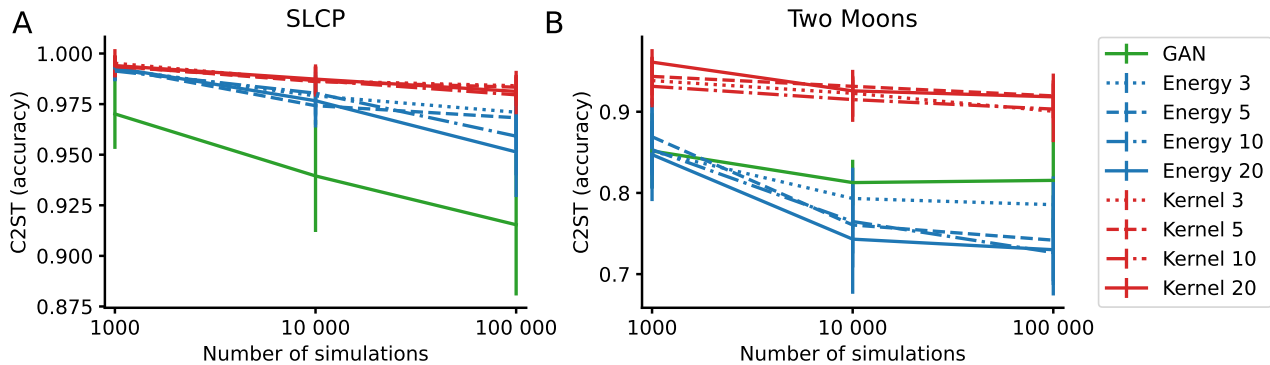


Figure 1: C2ST for SR and GAN methods for the SLCP and Two Moons benchmarks; larger values are worse. For the SR methods, we report results for different choices of the number of generative network samples m used in training. SLCP: GAN performs better, but poorly on an absolute scale. Two Moons: methods based on the Energy Score perform better.

4.1 Benchmark models

We consider here the “Simple Likelihood Complex Posterior” (SLCP) and the “Two Moons” benchmarks; in the former, a 5-dimensional θ defines the distribution of an 8-dimensional Gaussian \mathbf{y} in a nonlinear manner. In the Two Moons model, both \mathbf{y} and θ are 2-dimensional. We refer to Ramesh et al. [2022] and references therein for more details⁴ For both models, we train all methods on $n_{\text{train}} = 1000, 10000$ and 100000 posterior samples. We consider the SR methods with the Energy and Kernel Score trained with $m = 3, 5, 10$ or 20 samples from the generative network for each \mathbf{y}_i in a training batch. The SR methods are trained on a single CPU, while GAN is trained on an NVIDIA Tesla-V100 GPU. For the Two Moons model, we do not use early stopping for the SR methods; additionally, we employ the optimal configuration found in Ramesh et al. [2022] for GAN.

For these two models, samples from reference posteriors are available [Lueckmann et al., 2021]; therefore, as done in Ramesh et al. [2022], we assess the performance of the different methods via the discrimination ability of a classifier trained to distinguish samples from the reference and approximate posteriors (classification-based two-sample test, C2ST). If the classification accuracy is 0.5, the classifier is unable to distinguish between the two sets of samples, implying perfect posterior approximation.

In Figure 1, we report C2ST values for the GAN and SR methods for the different number of training simulations. For SLCP, GAN performs better (although the performance is poor on an absolute scale and worse than other LFI methods, see Ramesh et al., 2022). For the Two Moons example, methods based on the Energy Score perform better.

In Tables 1 and 2, we report other performance metrics, together with the runtime and the epoch at which training was early stopped, for GAN, Energy and Kernel Score, with $n_{\text{train}} = 100000$ and $m = 20$. Notice how the SR methods were trained in much shorter time (and on a single CPU). Additional results are reported in Appendices E.1 and E.2.

Table 1: SLCP: performance metrics, runtime and early stopping epoch for GAN, Energy and Kernel Score methods, with $n_{\text{train}} = 100000$ and $m = 20$. Notice how the SR methods were trained on a single CPU, while GAN was trained on a GPU. The maximum number of training epochs was 20000.

	C2ST ↓	NRMSE ↓	Cal. Err. ↓	R^2 ↑	Runtime (sec)	Early stopping epoch
GAN	0.92 ± 0.03	0.23 ± 0.05	0.06 ± 0.03	0.35 ± 0.30	30963	20000
Energy	0.95 ± 0.02	0.22 ± 0.06	0.07 ± 0.04	0.38 ± 0.32	1645	2100
Kernel	0.98 ± 0.01	0.22 ± 0.06	0.13 ± 0.10	0.37 ± 0.31	1210	1200

⁴These models are implemented in the `sbibm` Python package, whose accompanying paper [Lueckmann et al., 2021] provides additional details.

Table 2: Two Moons: performance metrics, runtime and early stopping epoch for GAN, Energy and Kernel Score methods, with $n_{\text{train}} = 100000$ and $m = 20$. Notice how the SR methods were trained on a single CPU, while GAN was trained on a GPU. Here, no early stopping was used (the maximum number of training epochs was 20000).

	C2ST ↓	NRMSE ↓	Cal. Err. ↓	R ² ↑	Runtime (sec)	Early stopping epoch
GAN	0.82 ± 0.07	0.20 ± 0.00	0.07 ± 0.02	0.51 ± 0.01	30232	20000
Energy	0.73 ± 0.04	0.20 ± 0.00	0.03 ± 0.00	0.51 ± 0.01	10805	20000
Kernel	0.92 ± 0.02	0.20 ± 0.00	0.03 ± 0.01	0.50 ± 0.01	10902	20000

4.2 Shallow water model

The shallow water model is obtained as the discretization of a PDE describing the propagation of an initial disturbance across the surface of a 1D shallow basin; the parameter $\theta \in \mathbb{R}^{100}$ represents the depth of the basin at equidistant points; the simulator outputs the evolution over 100 time-steps (producing a raw observation of size $100 \times 100 = 10000$); then, a Fourier transform is computed and the real and imaginary parts are concatenated and summed to Gaussian noise, leading to $\mathbf{y} \in \mathbb{R}^{20k}$. More details are given in Ramesh et al. [2022]. Besides the GAN method, we test here the Energy and Kernel score with $m = 10$ computed in three different configurations: 1) on the full parameter space, 2) with patch size 10 and step 5, and 3) with patch size 20 and step 10. Training is done on 100k samples on a NVIDIA Tesla-V100 GPU; additional details are discussed in Appendix D.2. Among the SR methods, the Energy Score with patch size 20 and step 10 performed better; therefore, we report only results for that method in the main body of the paper; results for the other configurations are given in Appendix E.3.

In Figure 2, we report posterior and posterior predictive samples for both methods, together with prior samples and the ground-truth depth profile. For the Energy Score, posterior samples better follow the ground truth profile and, similarly, posterior predictive samples better match the true observation.

In Table 3, we report the performance metrics, runtime and epoch of early stopping of the GAN and Energy Score method; notice how the calibration error is much smaller for the latter, whose training run faster. We also assess calibration via Simulation Based Calibration (Talts et al., 2018, details in Appendix C.2.2) in Figure 3. That as well highlights how the calibration of the Energy Score method is better than the one achieved by GAN.

Table 3: Shallow Water model: performance metrics, runtime and early stopping epoch for GAN and the Energy Score with patch size 20 and step 10. The latter method achieved better results with shorter training time. We do not train GAN from scratch but rather relied on the trained network obtained in Ramesh et al. [2022]. The training time we report here corresponds to what is mentioned in Ramesh et al. [2022], which used two GPUs for training (with respect to a single one for the SR methods). For the same reason, we do not report the epoch at which GAN training was early stopped.

	NRMSE ↓	Cal. Err. ↓	R ² ↑	Runtime (sec)	Early stopping epoch
Energy	0.05 ± 0.01	0.03 ± 0.02	0.89 ± 0.05	60017	12400
GAN	0.07 ± 0.01	0.12 ± 0.09	0.78 ± 0.05	≈345600	-

4.3 Noisy Camera model

Here, we consider $\theta \in \mathbb{R}^{28 \times 28}$ to be the images of the EMNIST dataset [Cohen et al., 2017], from which the data $\mathbf{y} \in \mathbb{R}^{28 \times 28}$ are generated by applying some blurring (see Ramesh et al., 2022 for

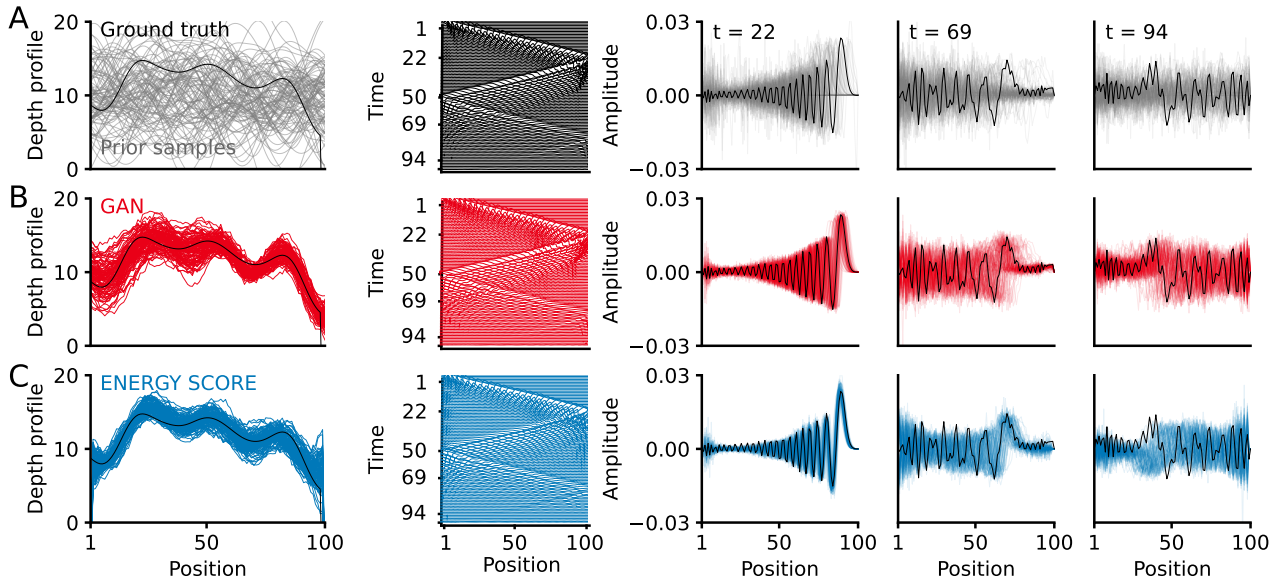


Figure 2: Shallow water model: inference results with GAN and Energy Score with patch size 20 and step 10. The figure structure closely follows that in Ramesh et al. [2022]. Row A: Ground truth, observation and prior samples. Left: ground-truth depth profile and prior samples. Middle: surface wave simulated from ground-truth profile as a function of position and time. Right: wave amplitudes at three different fixed times for ground-truth depth profile (black), and waves simulated from multiple prior samples (gray). Rows B and C refer respectively to GAN and Energy Score (with patch size 20 and step 10). For both, left represents posterior samples versus ground-truth (black) depth profiles, from which it can be seen how posterior samples for the Energy Score better follow the truth with respect to GAN; middle represents surface wave simulated from a single posterior sample; right represents wave amplitudes simulated from multiple posterior samples, at three different fixed times, with black line denoting the actual observation; again, Energy Score better follows the observation, except for $t = 94$.

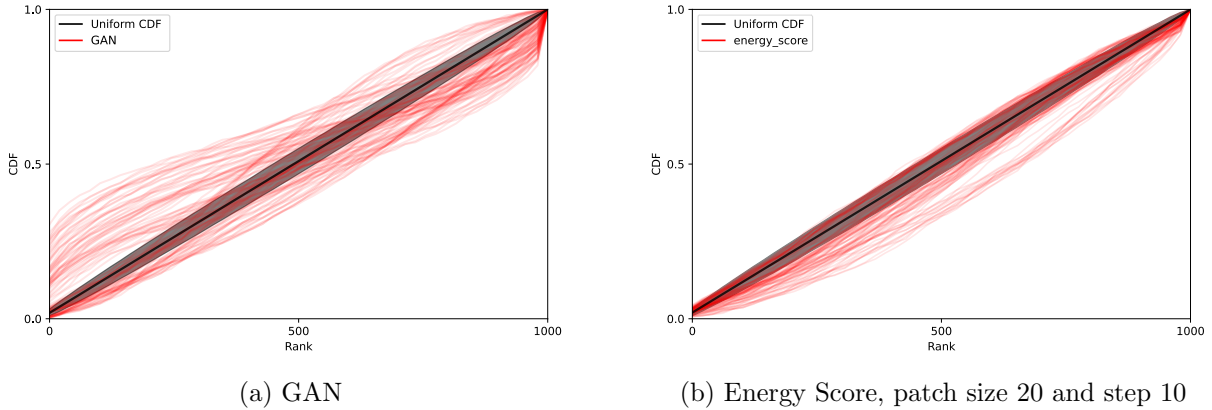


Figure 3: Shallow Water model: Simulation Based Calibration. Each line corresponds to a single dimension of θ and represents the CDF of the rank of the true parameter value with respect to a set of posterior samples. A calibrated posterior implies uniform CDF (diagonal black line, with associated 99% confidence region for that number of samples in gray).

details). Posterior inference corresponds therefore to Bayesian denoising. In this model, the dimension of parameter space is larger than in typical LFI applications; additionally, the prior is defined implicitly as we can only generate samples from it. This prevents the application of many standard LFI methods. Besides the GAN method, we test here the Energy and Kernel score with $m = 10$ in three different configurations: 1) on the full parameter space, 2) with patch size 14 and step 7, and 3) with patch size 8 and step 5. Training is done on 800 thousands samples on a NVIDIA Tesla-V100 GPU; additional details are discussed in Appendix D.3. Among the SR methods, those with patch size 8 and step 5 performed better; therefore, we report only results for the Kernel and Energy Score in that configuration in the main body of the paper; results for the other configurations are given in Appendix E.4.

In Figure 4, we report posterior mean and standard deviation for a set of observations for the different methods. SR methods lead to cleaner image reconstruction and more meaningful uncertainty quantification.

In Table 4, we report the performance metrics, runtime and epoch of early stopping of the GAN and SR methods; the latter lead to smaller calibration error, although that is still quite poor in absolute terms. The R^2 values here are also poor. We believe these low metric values are due to each pixel only taking a discrete set of values between 0 and 1, with white spaces assigned 0 and darkest pixels being assigned 1. The generative network outputs is bounded in $(0, 1)$ as it is obtained via a continuous transformation from \mathbb{R} . For the calibration error (see Appendix C.2.1), that means that a credible interval obtained from the generative network cannot contain the extreme values 0 or 1; similarly, the approximate posterior mean can never be smaller than 0 or larger than 1, thus decreasing the R^2 values (see Appendix C.1.2). Additionally, we report here the un-normalized RMSE, as computing the normalization would lead to infinite values for the pixels in which the true value is 0 for all training samples (see Appendix C.1.1).

Table 4: Noisy Camera model: performance metrics, runtime and early stopping epoch for GAN and for the Energy and Kernel Score with patch size 8 and step 5. The latter methods achieved better performance with shorter training time. All methods are trained on a single GPU.

	RMSE \downarrow	Cal. Err. \downarrow	R^2 \uparrow	Runtime (sec)	Early stopping epoch
GAN	0.25 ± 0.19	0.50 ± 0.00	-23.94 ± 366.08	45398	3600
Energy	0.06 ± 0.05	0.36 ± 0.12	-2.14 ± 55.86	22633	4000
Kernel	0.07 ± 0.05	0.36 ± 0.12	-10.29 ± 222.12	22545	3200



Figure 4: Noisy Camera model: ground truth and posterior inference with different methods, for a set of observations (each observation corresponds to a column). The first two rows represent the ground-truth values of θ and the corresponding observation y_o . The remaining rows represent mean and Standard Deviation (SD) for GAN and Energy and Kernel Score methods with patch size 8 and step 5. Notice how the posterior mean for the SR methods are neater than those obtained with GAN; additionally, the SD is larger close to the boundary of the reconstructed digit (notice the different color scale in the SD for GAN and for the SR methods).

5 Conclusions

We considered using a generative network to represent posterior distributions for Bayesian Likelihood-Free Inference, following Ramesh et al. [2022], and investigated training it via Scoring Rule minimization rather than in the adversarial setup of Ramesh et al. [2022]. The Scoring Rule approach is theoretically grounded and does not suffer from training instability and biased gradients, as does the adversarial approach. In simulation studies, and especially on high-dimensional tasks, we found that the scoring rule approach generally performed better and was substantially cheaper to train.

For the Scoring Rule approach, employing patched scores (Sec. 3.1) leads to a small performance improvement over the vanilla ones on the high-dimensional examples (see Appendix E.3 and E.4). While we designed the patches to capture the data structure, the improvement we observe could simply be due to computing the Energy and Kernel scores on lower-dimensional objects. To disentangle these two effects, we could define scoring rules using a random subset of components of θ of the same size as the patches used above. We leave this for future work.

Analogously to the patched scores, it may be that employing a patched discriminator [Isola et al., 2017] improves results with GAN; however, we believe this would not completely close the performance gap, which is mostly due to the harder optimization objective in GAN. To this point, more advanced adversarial training algorithms than the original GAN objective [Goodfellow et al., 2014] may lead to better results; however, for probabilistic forecasting, the results in Chapter 5 of the present thesis show Scoring Rule minimization to outperform state-of-the-art adversarial approach, while being cheaper and easier to train. We expect the same to hold for likelihood-free inference.

In the present work, we did not provide any theoretical guarantees for the Scoring Rule minimization approach; it could be of interest to prove a generalization bound between the empirical (Eq. 4) and population (Eq. 3) objectives, or a consistency result for the minimizer of Eq. (4), similarly to what done for probabilistic forecasting in Chapter 5. We leave these extensions for future work.

Finally, using a learned kernel to train a generative network via Kernel Score minimization could make the method more flexible. This could be done by learning the kernel adversarially (as in MMD GAN, Bińkowski et al., 2018), which however would break the ease of optimization which is the main advantage of the Scoring Rule approach. We hope that future research will address achieving both these goals together.

Acknowledgements

The authors thank Poornima Ramesh for helping to use the code used to create the results reported in Ramesh et al. [2022] and for providing additional result files.

LP is supported by EPSRC and MRC through the OxWaSP CDT programme (EP/L016710/1), which also funds the computational resources used to perform this work. RD is funded by EPSRC (grant nos. EP/V025899/1, EP/T017112/1) and NERC (grant no. NE/T00973X/1). The authors thank Geoff Nicholls for valuable feedback and suggestions.

References

- Z. An, D. J. Nott, and C. Drovandi. Robust Bayesian synthetic likelihood via a semi-parametric approach. *Statistics and Computing*, 30(3):543–557, 2020.
- M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.
- S. Arora, R. Ge, Y. Liang, T. Ma, and Y. Zhang. Generalization and equilibrium in generative adversarial nets (GANs). In *International Conference on Machine Learning*, pages 224–232. PMLR, 2017.
- S. Arora, A. Risteski, and Y. Zhang. Do GANs learn the distribution? some theory and empirics. In *International Conference on Learning Representations*, 2018. URL <https://openreview.net/forum?id=BJehNfW0->.
- M. G. Bellemare, I. Danihelka, W. Dabney, S. Mohamed, B. Lakshminarayanan, S. Hoyer, and R. Munos. The Cramer distance as a solution to biased Wasserstein gradients. *arXiv preprint arXiv:1705.10743*, 2017.
- E. Bernton, P. E. Jacob, M. Gerber, and C. P. Robert. Approximate Bayesian computation with the Wasserstein distance. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 81(2):235–269, 2019. doi: <https://doi.org/10.1111/rssb.12312>. URL <https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/rssb.12312>.
- M. Bińkowski, D. J. Sutherland, M. Arbel, and A. Gretton. Demystifying MMD GANs. In *International Conference on Learning Representations*, 2018.
- D. Bouchacourt, P. K. Mudigonda, and S. Nowozin. DISCO nets: DISsimilarity COefficient networks. *Advances in Neural Information Processing Systems*, 29:352–360, 2016.
- B.-E. Chérif-Abdellatif and P. Alquier. MMD-Bayes: Robust Bayesian estimation via maximum mean discrepancy. In *Symposium on Advances in Approximate Bayesian Inference*, pages 1–21. PMLR, 2020.
- J. Cockayne, M. M. Graham, C. J. Oates, T. J. Sullivan, and O. Teymur. Testing whether a learning procedure is calibrated. *Journal of Machine Learning Research*, 23(203):1–36, 2022. URL <http://jmlr.org/papers/v23/21-1065.html>.
- G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik. EMNIST: Extending MNIST to handwritten letters. In *2017 international joint conference on neural networks (IJCNN)*, pages 2921–2926. IEEE, 2017.

- C. Durkan, I. Murray, and G. Papamakarios. On contrastive learning for likelihood-free inference. In *International Conference on Machine Learning*, pages 2771–2781. PMLR, 2020.
- T. Gneiting and A. E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.
- T. Gneiting, F. Balabdaoui, and A. E. Raftery. Probabilistic forecasts, calibration and sharpness. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(2):243–268, 2007.
- I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.
- D. Greenberg, M. Nonnenmacher, and J. Macke. Automatic posterior transformation for likelihood-free inference. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2404–2414. PMLR, 09–15 Jun 2019. URL <http://proceedings.mlr.press/v97/greenberg19a.html>.
- A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.
- A. Gritsenko, T. Salimans, R. van den Berg, J. Snoek, and N. Kalchbrenner. A spectral energy distance for parallel speech synthesis. *Advances in Neural Information Processing Systems*, 33:13062–13072, 2020.
- A. Harakeh and S. L. Waslander. Estimating and evaluating regression predictive uncertainty in deep object detectors. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=YLewtnvKgR7>.
- P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.
- D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In Y. Bengio and Y. LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL <http://arxiv.org/abs/1312.6114>.
- J. Lintusaari, M. U. Gutmann, R. Dutta, S. Kaski, and J. Corander. Fundamentals and recent developments in approximate Bayesian computation. *Systematic Biology*, 66(1):e66–e82, 2017. ISSN 1076836X. doi: 10.1093/sysbio/syw077. URL <https://doi.org/10.1093/sysbio/syw077>.
- J.-M. Lueckmann, P. J. Goncalves, G. Bassetto, K. Öcal, M. Nonnenmacher, and J. H. Macke. Flexible statistical inference for mechanistic models of neural dynamics. In *Advances in Neural Information Processing Systems*, pages 1289–1299, 2017.
- J.-M. Lueckmann, G. Bassetto, T. Karaletsos, and J. H. Macke. Likelihood-free inference with emulator networks. In *Symposium on Advances in Approximate Bayesian Inference*, pages 32–53. PMLR, 2019.
- J.-M. Lueckmann, J. Boelts, D. Greenberg, P. Goncalves, and J. Macke. Benchmarking simulation-based inference. In A. Banerjee and K. Fukumizu, editors, *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 343–351. PMLR, 13–15 Apr 2021.
- M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

- H. D. Nguyen, J. Arbel, H. Lü, and F. Forbes. Approximate Bayesian computation via the energy statistic. *IEEE Access*, 8:131683–131698, 2020.
- S. Nowozin, B. Cseke, and R. Tomioka. f-GAN: Training generative neural samplers using variational divergence minimization. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 271–279, 2016.
- L. Pacchiardi, R. Adewoyin, P. Dueben, and R. Dutta. Probabilistic forecasting with conditional generative networks via scoring rule minimization. *arXiv preprint arXiv:2112.08217*, 2022.
- G. Papamakarios and I. Murray. Fast ε -free inference of simulation models with Bayesian conditional density estimation. In *Advances in Neural Information Processing Systems*, pages 1028–1036, 2016.
- G. Papamakarios, D. Sterratt, and I. Murray. Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. In K. Chaudhuri and M. Sugiyama, editors, *Proceedings of Machine Learning Research*, volume 89 of *Proceedings of Machine Learning Research*, pages 837–848. PMLR, 16–18 Apr 2019. URL <http://proceedings.mlr.press/v89/papamakarios19a.html>.
- G. Papamakarios, E. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021. URL <http://jmlr.org/papers/v22/19-1028.html>.
- A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. PyTorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.
- L. F. Price, C. C. Drovandi, A. Lee, and D. J. Nott. Bayesian synthetic likelihood. *Journal of Computational and Graphical Statistics*, 27(1):1–11, 2018.
- S. T. Radev, U. K. Mertens, A. Voss, L. Ardizzone, and U. Köthe. BayesFlow: Learning complex stochastic models with invertible neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.
- P. Ramesh, J.-M. Lueckmann, J. Boelts, Á. Tejero-Cantero, D. S. Greenberg, P. J. Goncalves, and J. H. Macke. GATSBI: Generative adversarial training for simulation-based inference. In *International Conference on Learning Representations*, 2022. URL <https://openreview.net/forum?id=kR1hC6j48Tp>.
- E. Richardson and Y. Weiss. On GANs and GMMs. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 5852–5863, 2018.
- M. L. Rizzo and G. J. Székely. Energy distance. *Wiley interdisciplinary reviews: Computational statistics*, 8(1):27–38, 2016.
- T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training GANs. *Advances in neural information processing systems*, 29, 2016.
- M. Scheuerer and T. M. Hamill. Variogram-based proper scoring rules for probabilistic forecasts of multivariate quantities. *Monthly Weather Review*, 143(4):1321–1334, 2015.
- S. Talts, M. Betancourt, D. Simpson, A. Vehtari, and A. Gelman. Validating Bayesian inference algorithms with simulation-based calibration. *arXiv preprint arXiv:1804.06788*, 2018.

Appendix

A Proof of Lemma 1

Proof. By the definition of proper SR, we have that:

$$\alpha_1 \cdot S_1(Q, Q) \leq \alpha_1 \cdot S_1(P, Q) \quad \forall P, Q \in \mathcal{P},$$

and similar for S_2 . By adding the two inequalities, we have therefore that:

$$\alpha_1 \cdot S_1(Q, Q) + \alpha_2 \cdot S_2(Q, Q) \leq \alpha_1 \cdot S_1(P, Q) + \alpha_2 \cdot S_2(P, Q) \quad \forall P, Q \in \mathcal{P},$$

which implies that S_+ is a proper SR.

Assume now additionally that S_1 , without loss of generality, is strictly proper, i.e.:

$$\alpha_1 \cdot S_1(Q, Q) < \alpha_1 \cdot S_1(P, Q) \quad \forall P, Q \in \mathcal{P};$$

then, summing the above with the corresponding inequality for S_2 gives that:

$$\alpha_1 \cdot S_1(Q, Q) + \alpha_2 \cdot S_2(Q, Q) < \alpha_1 \cdot S_1(P, Q) + \alpha_2 \cdot S_2(P, Q) \quad \forall P, Q \in \mathcal{P},$$

which implies that S_+ is a strictly proper SR. □

B Unbiased estimate of the training objective

We discuss here how we can obtain unbiased gradient estimates for the Scoring Rule training objective in Eq. (4) with respect to the parameters of the generative network ϕ .

Recall now that we want to solve:

$$\hat{\phi} := \arg \min_{\phi} J(\phi), \quad J(\phi) = \frac{1}{n} \sum_{i=1}^n S(Q_{\phi}(\cdot | \mathbf{y}_i), \boldsymbol{\theta}_i). \quad (6)$$

To do this, we exploit Stochastic Gradient Descent (SGD), which requires unbiased estimates of $J(\phi)$. Notice how, for all the Scoring Rules used across this work, as well as any weighted sum of those, we can write: $S(P, \mathbf{x}) = \mathbb{E}_{\tilde{\mathbf{X}}, \tilde{\mathbf{X}}' \sim P} [h(\tilde{\mathbf{X}}, \tilde{\mathbf{X}}', \mathbf{x})]$ for some function h ; namely, the SR is defined through an expectation over (possibly multiple) samples from P . That is the form exploited in Sec. 1.2.2 in the introduction of the present thesis to obtain unbiased SR estimates.

Now, we will use this fact to obtain unbiased estimates for the objective in Eq. (6).

$$J(\phi) = \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{\tilde{\boldsymbol{\theta}}, \tilde{\boldsymbol{\theta}}' \sim Q_{\phi}(\cdot | \mathbf{y}_i)} [h(\tilde{\boldsymbol{\theta}}, \tilde{\boldsymbol{\theta}}', \boldsymbol{\theta}_i)] = \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{\mathbf{Z}, \mathbf{Z}' \sim P_{\mathbf{z}}} [h(g_{\phi}(\mathbf{Z}, \mathbf{y}_i), g_{\phi}(\mathbf{Z}', \mathbf{y}_i), \boldsymbol{\theta}_i)],$$

where we used the fact that Q_{ϕ} is the distribution induced by a generative network with transformation g_{ϕ} ; this is called the reparametrization trick [Kingma and Welling, 2014]. Now:

$$\begin{aligned} \nabla_{\phi} J(\phi) &= \nabla_{\phi} \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{\mathbf{Z}, \mathbf{Z}' \sim P_{\mathbf{z}}} [h(g_{\phi}(\mathbf{Z}, \mathbf{y}_i), g_{\phi}(\mathbf{Z}', \mathbf{y}_i), \boldsymbol{\theta}_i)] \\ &= \frac{1}{n} \sum_{i=1}^n \mathbb{E}_{\mathbf{Z}, \mathbf{Z}' \sim P_{\mathbf{z}}} [\nabla_{\phi} h(g_{\phi}(\mathbf{Z}, \mathbf{y}_i), g_{\phi}(\mathbf{Z}', \mathbf{y}_i), \boldsymbol{\theta}_i)]. \end{aligned}$$

In the latter equality, the exchange between expectation and gradient is not a trivial step, due to the non-differentiability of functions (such as ReLU) used in g_{ϕ} . Fortunately, Theorem 5 in Bińkowski et al. [2018] proved that to be valid almost surely with respect to a measure on the space Φ to which the weights of the neural network ϕ belong, under mild conditions on the NN architecture.

We can now easily obtain an unbiased estimate of the above using samples $\mathbf{z}_{i,j} \sim P_{\mathbf{z}}, j = 1, \dots, m$, for each $i \in \{1, \dots, n\}$. Additionally, Stochastic Gradient Descent usually considers a small batch of training samples at each step, obtained by taking a random subset (or batch) $\mathcal{B} \subseteq \{1, 2, \dots, n\}$. Therefore, the following unbiased estimate of $\nabla_{\phi} J(\phi)$ can be obtained:

$$\widehat{\nabla_{\phi} J(\phi)} = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \frac{1}{m(m-1)} \sum_{\substack{j,k=1 \\ j \neq k}}^m \nabla_{\phi} h(g_{\phi}(\mathbf{z}_{i,j}; \mathbf{y}_i), g_{\phi}(\mathbf{z}_{i,k}; \mathbf{y}_i), \boldsymbol{\theta}_i).$$

In practice, the above is obtained by computing the gradient of the following unbiased estimate of $J(\phi)$ via autodifferentiation libraries (see for instance Paszke et al., 2019):

$$\hat{J}(\phi) = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \frac{1}{m(m-1)} \sum_{\substack{j,k=1 \\ j \neq k}}^m h(g_{\phi}(\mathbf{z}_{i,j}; \mathbf{y}_i), g_{\phi}(\mathbf{z}_{i,k}; \mathbf{y}_i), \boldsymbol{\theta}_i).$$

In Algorithm 1, we train a generative network for a single epoch using a scoring rule S for which unbiased estimators can be obtained by using $m > 1$ samples from Q_{ϕ} . Compare it with the adversarial approach reported in Algorithm 2 in the Introduction of the present thesis; in the SR approach, multiple samples from the generative networks are required at each step ($m > 1$), while a unique one is enough for the adversarial approach. Conversely, the SR approach does not require an additional critic network and learning rate γ and is simpler and faster to train (see the results in Sec. 4 and Chapter 5 for more details). As in Algorithm 2 in the Introduction of the present thesis, we use a single pair $(\boldsymbol{\theta}_i, \mathbf{y}_i)$ to estimate the gradient.

Algorithm 1 Single epoch generative-SR training.

Require: Parametric map g_{ϕ} , SR S , learning rate ϵ .

for each training pair $(\boldsymbol{\theta}_i, \mathbf{y}_i)$ **do**

 Sample **multiple** $\mathbf{z}_1, \dots, \mathbf{z}_m$

 Obtain $\tilde{\boldsymbol{\theta}}_{i,j}^{\phi} = g_{\phi}(\mathbf{z}_j, \mathbf{y}_i)$

 Obtain unbiased estimate $\hat{S}(Q_{\phi}(\cdot | \mathbf{y}_i), \boldsymbol{\theta}_i)$ from $\tilde{\boldsymbol{\theta}}_{i,j}^{\phi}$

 Set $\phi \leftarrow \phi - \epsilon \cdot \nabla_{\phi} \hat{S}(Q_{\phi}(\cdot | \mathbf{y}_i), \boldsymbol{\theta}_i)$

end for

C Details on performance measures

Here, we review the measures of performance used in the empirical studies. We follow Radev et al. [2020] in defining these measures and report them here for ease of reference. All these metrics are for univariate $\boldsymbol{\theta}$; when handling multivariate $\boldsymbol{\theta}$, we therefore compute them on each dimension separately and report the average.

C.1 Deterministic performance measures

We discuss two measures of performance of a deterministic forecast $\hat{\boldsymbol{\theta}}_i$ for a realization $\boldsymbol{\theta}_i$; across our work, we take $\hat{\boldsymbol{\theta}}_i$ to be the mean of the (univariate) probability distribution $Q_{\phi}(\cdot | \mathbf{y}_i)$.

C.1.1 RMSE

We first introduce the Root Mean-Square Error (RMSE) as:

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (\hat{\boldsymbol{\theta}}_i - \boldsymbol{\theta}_i)^2},$$

where we consider here for simplicity $i = 1, \dots, n$. From the above, we obtain the Normalized RMSE (NRMSE) as:

$$\text{NRMSE} = \frac{\text{RMSE}}{\max_i\{\boldsymbol{\theta}_i\} - \min_i\{\boldsymbol{\theta}_i\}}.$$

NRMSE = 0 implies $\hat{\boldsymbol{\theta}}_i = \boldsymbol{\theta}_i$ for all i 's. NRMSE $\in [0, 1]$ and allows to compare performance over different tasks. Notice however that, when $\max_i\{\boldsymbol{\theta}_i\} = \min_i\{\boldsymbol{\theta}_i\}$, NRMSE diverges; in that case, we consider the un-normalized RMSE.

C.1.2 Coefficient of determination

The coefficient of determination R^2 measures how much of the variance in $\{\boldsymbol{\theta}_i\}_{i=1}^n$ is explained by $\{\hat{\boldsymbol{\theta}}_i\}_{i=1}^n$. Specifically, it is given by:

$$R^2 = 1 - \frac{\sum_{i=1}^n (\boldsymbol{\theta}_i - \hat{\boldsymbol{\theta}}_i)^2}{\sum_{i=1}^n (\boldsymbol{\theta}_i - \bar{\boldsymbol{\theta}})^2},$$

where $\bar{\boldsymbol{\theta}} = \frac{1}{n} \sum_{i=1}^n \boldsymbol{\theta}_i$. $R^2 \leq 1$ and $R^2 = 1 \implies \hat{\boldsymbol{\theta}}_i = \boldsymbol{\theta}_i$ for all i 's.

C.2 Calibration measures

Here, we review two measures of calibration of a probabilistic forecast. Both measures consider the univariate marginals of the approximate posterior distribution $Q_\phi(\cdot|\mathbf{y}_i)$; for the component l , let us denote it by $Q_{\phi,l}(\cdot|\mathbf{y}_i)$.

C.2.1 Calibration error

The calibration error [Radev et al., 2020] quantifies how well the credible intervals of the approximate posteriors $Q_{\phi,l}(\cdot|\mathbf{y}_i)$ for different \mathbf{y}_i match the empirical distribution of $\theta_{i,l}$. Specifically, let $\alpha(l)$ be the proportion of times the verification $\theta_{i,l}$ falls into an α -credible interval of $Q_{\phi,l}(\cdot|\mathbf{y}_i)$, computed over all values of i . If the marginal forecast distribution is perfectly calibrated for component l , $\alpha(l) = \alpha$ for all values of $\alpha \in (0, 1)$.

Therefore, we define the calibration error as the median of $|\alpha(l) - \alpha|$ over 100 equally spaced values of $\alpha \in (0, 1)$. Therefore, the calibration error is a value between 0 and 1, where 0 denotes perfect calibration.

In practice, the credible intervals of the predictive are estimated using a set of samples from $Q_\phi(\cdot|\mathbf{y}_i)$.

The calibration error can be related to the *strong calibration* of Cockayne et al. [2022], which implies correct coverage for credible sets (see their Remark 2.9).

C.2.2 Simulation-Based Calibration (SBC)

SBC [Talts et al., 2018] tests a self-consistency property of the Bayesian posterior in a posterior approximation. In fact, the Bayesian posterior satisfies the following equality:

$$\pi(\boldsymbol{\theta}) = \int p(\boldsymbol{\theta}, \tilde{\boldsymbol{\theta}}, \tilde{\mathbf{y}}) d\tilde{\mathbf{y}} d\tilde{\boldsymbol{\theta}} = \int p(\boldsymbol{\theta}, \tilde{\mathbf{y}} | \tilde{\boldsymbol{\theta}}) \pi(\tilde{\boldsymbol{\theta}}) d\tilde{\mathbf{y}} d\tilde{\boldsymbol{\theta}} = \int \pi(\boldsymbol{\theta} | \tilde{\mathbf{y}}) p(\tilde{\mathbf{y}} | \tilde{\boldsymbol{\theta}}) \pi(\tilde{\boldsymbol{\theta}}) d\tilde{\mathbf{y}} d\tilde{\boldsymbol{\theta}} \quad (7)$$

in practice, this means that, if you sample from the prior $\tilde{\boldsymbol{\theta}} \sim \pi$, use that to generate a sample from the likelihood $\tilde{\mathbf{y}} \sim p(\cdot|\tilde{\boldsymbol{\theta}})$ and use the latter in turn to generate a posterior sample $\boldsymbol{\theta} \sim \pi(\cdot|\tilde{\mathbf{y}})$, $\boldsymbol{\theta}$ is distributed according to the prior $\pi(\boldsymbol{\theta})$. If you repeat the same procedure by sampling $\boldsymbol{\theta}$ from an *approximate* posterior, say $\boldsymbol{\theta} \sim Q_\phi(\cdot|\tilde{\mathbf{y}})$, then $\boldsymbol{\theta} \sim \pi$ is a necessary condition for $q_\phi(\cdot|\mathbf{y}) = \pi(\cdot|\mathbf{y})$, i.e. for the approximate posterior to be exact. Notice, however, how this is *not* a sufficient condition: the equality can be satisfied even if $q_\phi(\cdot|\mathbf{y})$ is different from the posterior (it is in fact trivially satisfied $q_\phi(\cdot|\mathbf{y}) = \pi$, i.e., when the approximate posterior corresponds to the prior).

A way to empirically test the above property involves, for a given prior sample $\tilde{\theta}$, drawing from the likelihood multiple times $\mathbf{y}_i \sim p(\cdot|\tilde{\theta}), i = 1, \dots, N$ and, for each of these, obtaining a single approximate posterior sample $\theta_i \sim q_\phi(\cdot|\mathbf{y}_i)$. Given these, you compute the rank of $\tilde{\theta}$: $r = \sum_{i=1}^N \mathbf{1}_{[\theta_i < \tilde{\theta}]}$ (this only makes sense if θ is univariate; otherwise, you compute the rank independently for each dimension of θ). If θ_i 's were effectively distributed from the prior, r is a uniform random variable on $\{1, 2, \dots, N\}$. Therefore, repeating this procedure for different prior samples $\tilde{\theta}$ and visualizing the distribution of the resulting r 's (for instance, through a histogram or by plotting the CDF) gives an indication of whether an equivalence such as Eq. (7) is satisfied for q_ϕ . See Algorithm 2 in Radev et al. [2020] for a precise description of this procedure, which goes under the name of Simulation-Based Calibration (SBC). SBC tests the *weak calibration* of Cockayne et al. [2022]; additionally, it is closely related to the concept of probabilistic calibration and rank histogram in the framework of probabilistic forecasting [Gneiting et al., 2007].

D Experimental details

Precise configuration details can be found in the code accompanying the paper.

D.1 Benchmark models

Except for the details reported in the main body of the paper, the training configuration for the two benchmark models is the same as in Ramesh et al. [2022]; of course, some hyperparameter values for the GAN training routine do not apply to the SR one (for example, all the hyperparameters related to the discriminator).

D.2 Shallow Water Model

We train all methods for at most 40k epochs on 100k training samples. For the SR method, we tried both $m = 3$ and $m = 10$, with the latter resulting in improved performance; all the results reported in the paper refer to $m = 10$.

GAN used a batch size of 125 (as in Ramesh et al., 2022), while SR methods used a batch size of 60 (otherwise, GPU memory overflow occurs).

Recall that the parameters $\theta \in \mathbb{R}^{100}$ are arranged along a 1D uniform grid. When using the patched SR configuration, we consider patches of size `patch_size` disposed at a distance `patch_step` from each other. Therefore, the number of patches is

$$\text{n_patches} = (100 - \text{patch_size})/\text{patch_step} + 1.$$

We used therefore the following patched SR configurations on the 1D grid:

1. `patch_size = 10` and `patch_step = 5`, resulting in `n_patches = 19`.
2. `patch_size = 20` and `patch_step = 10`, resulting in `n_patches = 9`.

The patched SR is added to the overall score over the full parameter space.

The training time (per epoch) is roughly constant in the un-patched and the two different patched configurations.

D.3 Camera Model

We train all methods for at most 10k epochs on 800k training samples. For the SR method, we tried both $m = 3$ and $m = 10$, with the latter resulting in better performance.

Both the SR and GAN methods used a batch size of 800 as in Ramesh et al. [2022].

Here, the parameters θ are on a 28×28 square grid. When using the patched SR configuration, we consider patches of size `patch_size` \times `patch_size` disposed at a distance `patch_step` from each other in both spatial dimensions. The number of patches is obtained as

$$\text{n_patches} = [(28 - \text{patch_size})/\text{patch_step} + 1]^2.$$

We used therefore the following patched SR configurations on the 2D grid:

1. `patch_size = 14` and `patch_step = 7`, resulting in `n_patches = 9`.
2. `patch_size = 8` and `patch_step = 5`, resulting in `n_patches = 25`.

The patched SR is added to the overall score over the full parameter space.

The training time (per epoch) is roughly constant in the un-patched and the two different patched configurations.

E Additional experimental results

E.1 SLCP

In Figure 5, we report the posterior samples obtained with the Energy Score with $m = 20$ and compare them with the samples from the reference posterior. In Figure 6, we report Simulation-Based Calibration results (see Appendix C.2.2): for each dimension of θ , the corresponding histogram represents the distribution of the rank of the true parameter value in a set of samples from the approximate posterior. We show that for GAN and for the Energy Score with $m = 20$.

Tables 5, 6, 7, 8, 9 and 10 report the different performance metrics, the runtime, and the early stopping epoch for all methods (columns) and all number of training samples (rows); for Energy and Kernel Score, the number in the column header denotes the number of draws from the generative network during training for each \mathbf{y}_i in the training batch.

Table 5: SLCP: classification-based two-sample test (C2ST); lower is better.

	GAN	Energy 3	Energy 5	Energy 10	Energy 20	Kernel 3	Kernel 5	Kernel 10	Kernel 20
1000	0.97 ± 0.02	0.99 ± 0.01	0.99 ± 0.01	0.99 ± 0.00	0.99 ± 0.01	1.00 ± 0.01	0.99 ± 0.01	0.99 ± 0.01	0.99 ± 0.01
10000	0.94 ± 0.03	0.98 ± 0.01	0.97 ± 0.01	0.98 ± 0.01	0.98 ± 0.01	0.99 ± 0.01	0.99 ± 0.01	0.99 ± 0.01	0.99 ± 0.01
100000	0.92 ± 0.03	0.97 ± 0.01	0.97 ± 0.02	0.96 ± 0.02	0.95 ± 0.02	0.98 ± 0.01	0.98 ± 0.01	0.98 ± 0.01	0.98 ± 0.01

Table 6: SLCP: NRMSE; lower is better.

	GAN	Energy 3	Energy 5	Energy 10	Energy 20	Kernel 3	Kernel 5	Kernel 10	Kernel 20
1000	0.24 ± 0.05	0.25 ± 0.05	0.25 ± 0.05	0.25 ± 0.05	0.25 ± 0.06	0.25 ± 0.05	0.25 ± 0.05	0.25 ± 0.05	0.25 ± 0.05
10000	0.23 ± 0.05	0.23 ± 0.05	0.23 ± 0.05	0.23 ± 0.05	0.23 ± 0.05	0.23 ± 0.05	0.23 ± 0.05	0.23 ± 0.05	0.23 ± 0.05
100000	0.23 ± 0.05	0.22 ± 0.05	0.22 ± 0.06	0.22 ± 0.06	0.22 ± 0.06	0.22 ± 0.06	0.22 ± 0.06	0.22 ± 0.05	0.22 ± 0.06

Table 7: SLCP: calibration error; lower is better.

	GAN	Energy 3	Energy 5	Energy 10	Energy 20	Kernel 3	Kernel 5	Kernel 10	Kernel 20
1000	0.13 ± 0.05	0.19 ± 0.07	0.20 ± 0.05	0.20 ± 0.05	0.22 ± 0.07	0.24 ± 0.09	0.23 ± 0.10	0.24 ± 0.08	0.24 ± 0.08
10000	0.08 ± 0.03	0.11 ± 0.05	0.10 ± 0.05	0.12 ± 0.07	0.10 ± 0.07	0.15 ± 0.10	0.13 ± 0.09	0.14 ± 0.10	0.16 ± 0.10
100000	0.06 ± 0.03	0.08 ± 0.07	0.08 ± 0.04	0.07 ± 0.05	0.07 ± 0.04	0.13 ± 0.11	0.13 ± 0.10	0.12 ± 0.08	0.13 ± 0.10

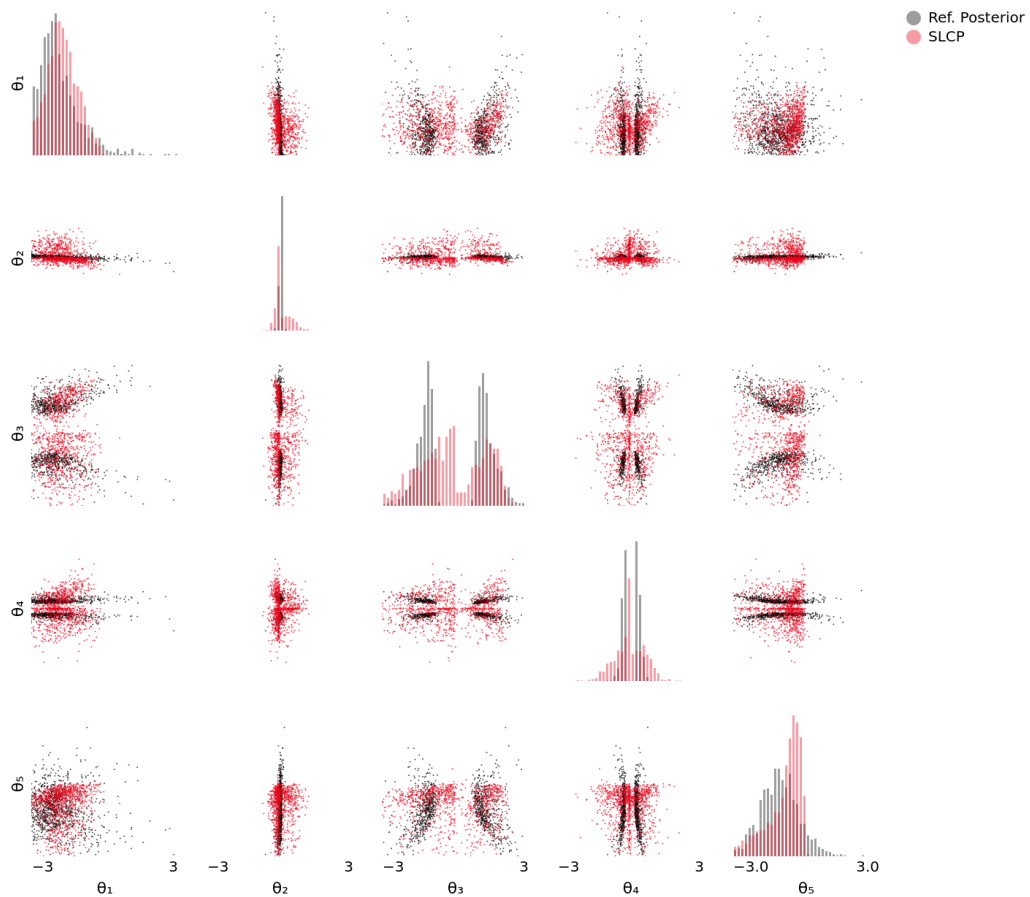
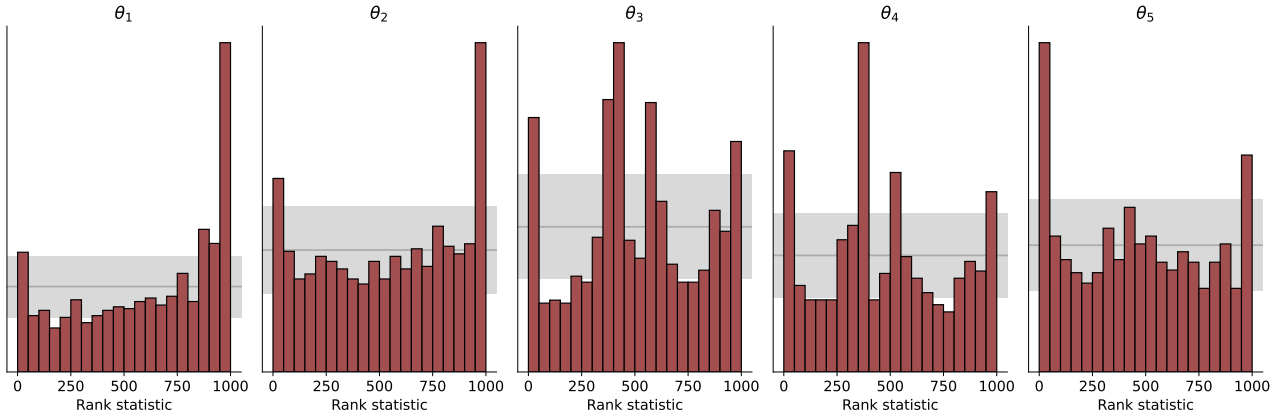
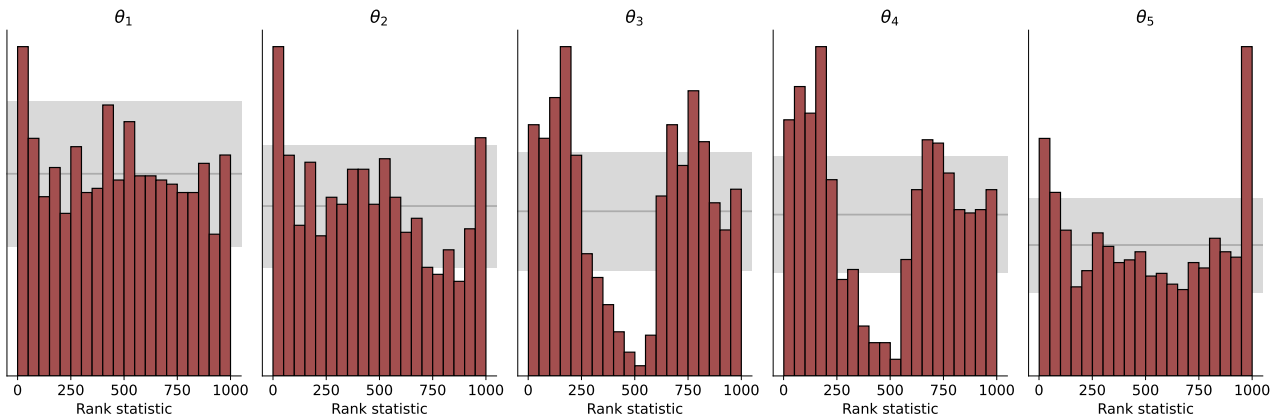


Figure 5: SLCP: posterior samples for Energy Score trained with $m = 20$ and reference posterior samples. Diagonal panels represent univariate marginals, while off-diagonals panels represent bivariate marginals. A similar graph for GAN can be found in the supplementary material in Ramesh et al. [2022].



(a) GAN



(b) Energy Score, $m = 20$

Figure 6: SLCP: Simulation-Based Calibration results represented as rank histograms; for each dimension of θ , the corresponding histogram represents the distribution of the rank of the true parameter value in a set of samples from the approximate posterior. If the approximate posterior is calibrated, histogram bars should be in the grey region with 99% probability.

Table 8: SLCP: R^2 ; larger is better.

	GAN	Energy 3	Energy 5	Energy 10	Energy 20	Kernel 3	Kernel 5	Kernel 10	Kernel 20
1000	0.25 ± 0.29	0.24 ± 0.30	0.22 ± 0.30	0.25 ± 0.31	0.18 ± 0.35	0.22 ± 0.30	0.22 ± 0.30	0.24 ± 0.30	0.23 ± 0.31
10000	0.35 ± 0.30	0.35 ± 0.30	0.35 ± 0.30	0.35 ± 0.30	0.34 ± 0.31	0.35 ± 0.29	0.35 ± 0.30	0.34 ± 0.30	0.34 ± 0.30
100000	0.35 ± 0.30	0.36 ± 0.30	0.37 ± 0.30	0.38 ± 0.32	0.38 ± 0.32	0.37 ± 0.31	0.36 ± 0.31	0.36 ± 0.30	0.37 ± 0.31

Table 9: SLCP: runtime in seconds; recall that GAN was trained on GPU while the SR methods were trained on a single CPU.

	GAN	Energy 3	Energy 5	Energy 10	Energy 20	Kernel 3	Kernel 5	Kernel 10	Kernel 20
1000	4796	654	692	620	885	515	531	682	1330
10000	9671	651	658	639	720	636	658	655	697
100000	30963	1060	1160	1305	1645	1245	1044	1057	1210

Table 10: SLCP: epoch at which early stopping occurred; the max number of training epochs was 20000.

	GAN	Energy 3	Energy 5	Energy 10	Energy 20	Kernel 3	Kernel 5	Kernel 10	Kernel 20
1000	20000	1000	1000	1000	1100	1100	1000	1000	1000
10000	20000	1100	1000	1100	1100	1100	1100	1000	1000
100000	20000	1000	1200	1500	2100	1600	1100	1000	1200

E.2 Two Moons

In Figure 7, we report posterior samples obtained with the Energy Score with $m = 20$ and compare them with samples from the reference posterior. In Figure 8, we report Simulation-Based Calibration results (see Appendix C.2.2): for each dimension of θ , the corresponding histogram represents the distribution of the rank of the true parameter value in a set of samples from the approximate posterior. We show that for GAN and for the Energy Score with $m = 20$.

Tables 11, 12, 13, 14, 15 and 16 report the different performance metrics, the runtime and the early stopping epoch for all methods (columns) and all number of training samples (rows); for Energy and Kernel Score, the number in the column header denotes the number of draws from the generative network during training for each \mathbf{y}_i in the training batch.

Table 11: Two Moons: classification-based two-sample test (C2ST); lower is better.

	GAN	Energy 3	Energy 5	Energy 10	Energy 20	Kernel 3	Kernel 5	Kernel 10	Kernel 20
1000	0.85 ± 0.05	0.85 ± 0.06	0.87 ± 0.05	0.85 ± 0.03	0.85 ± 0.04	0.94 ± 0.03	0.94 ± 0.02	0.93 ± 0.03	0.96 ± 0.02
10000	0.81 ± 0.03	0.79 ± 0.04	0.76 ± 0.05	0.76 ± 0.04	0.74 ± 0.07	0.92 ± 0.03	0.93 ± 0.01	0.91 ± 0.03	0.93 ± 0.01
100000	0.82 ± 0.07	0.79 ± 0.03	0.74 ± 0.06	0.73 ± 0.05	0.73 ± 0.04	0.90 ± 0.04	0.92 ± 0.03	0.90 ± 0.02	0.92 ± 0.02

Table 12: Two Moons: NRMSE; lower is better.

	GAN	Energy 3	Energy 5	Energy 10	Energy 20	Kernel 3	Kernel 5	Kernel 10	Kernel 20
1000	0.20 ± 0.00	0.20 ± 0.00	0.20 ± 0.00	0.20 ± 0.00	0.20 ± 0.00	0.21 ± 0.00	0.21 ± 0.00	0.21 ± 0.00	0.20 ± 0.00
10000	0.20 ± 0.00	0.20 ± 0.00	0.20 ± 0.00	0.20 ± 0.00	0.20 ± 0.00	0.20 ± 0.00	0.20 ± 0.00	0.20 ± 0.00	0.20 ± 0.00
100000	0.20 ± 0.00	0.20 ± 0.00	0.20 ± 0.00	0.20 ± 0.00	0.20 ± 0.00	0.20 ± 0.00	0.20 ± 0.00	0.20 ± 0.00	0.20 ± 0.00

Table 13: Two Moons: calibration error; lower is better.

	GAN	Energy 3	Energy 5	Energy 10	Energy 20	Kernel 3	Kernel 5	Kernel 10	Kernel 20
1000	0.07 ± 0.01	0.05 ± 0.01	0.09 ± 0.02	0.07 ± 0.01	0.06 ± 0.00	0.08 ± 0.01	0.11 ± 0.00	0.14 ± 0.02	0.12 ± 0.01
10000	0.06 ± 0.01	0.04 ± 0.02	0.03 ± 0.01	0.04 ± 0.03	0.03 ± 0.01	0.04 ± 0.00	0.03 ± 0.01	0.03 ± 0.02	0.03 ± 0.01
100000	0.07 ± 0.02	0.04 ± 0.01	0.03 ± 0.00	0.04 ± 0.02	0.03 ± 0.00	0.04 ± 0.00	0.03 ± 0.01	0.06 ± 0.01	0.03 ± 0.01

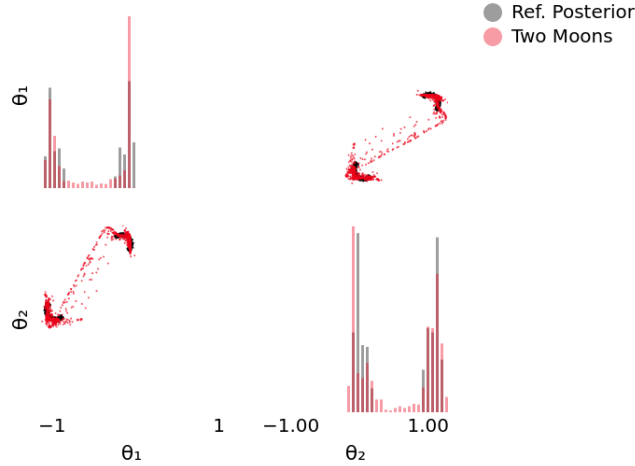


Figure 7: Two Moons: posterior samples for Energy Score trained with $m = 20$ and reference posterior samples. Diagonal panels represent univariate marginals, while off-diagonal panels represent bivariate marginals. A similar graph for GAN can be found in the supplementary material in Ramesh et al. [2022].

Table 14: Two Moons: R^2 ; larger is better.

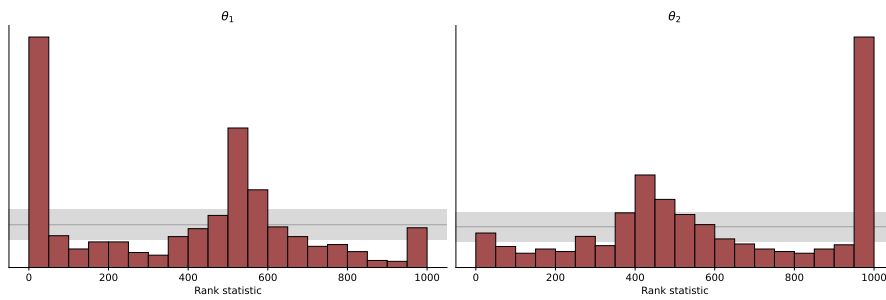
	GAN	Energy 3	Energy 5	Energy 10	Energy 20	Kernel 3	Kernel 5	Kernel 10	Kernel 20
1000	0.50 ± 0.01	0.49 ± 0.01	0.50 ± 0.01	0.50 ± 0.01	0.51 ± 0.01	0.48 ± 0.01	0.49 ± 0.01	0.48 ± 0.01	0.49 ± 0.01
10000	0.49 ± 0.01	0.50 ± 0.01	0.51 ± 0.01	0.51 ± 0.01	0.51 ± 0.01	0.50 ± 0.01	0.50 ± 0.01	0.50 ± 0.01	0.50 ± 0.01
100000	0.51 ± 0.01	0.50 ± 0.01	0.50 ± 0.01	0.50 ± 0.01	0.51 ± 0.01	0.50 ± 0.01	0.51 ± 0.01	0.50 ± 0.01	0.50 ± 0.01

Table 15: Two Moons: runtime in seconds; recall that GAN was trained on GPU while the SR methods were trained on a single CPU.

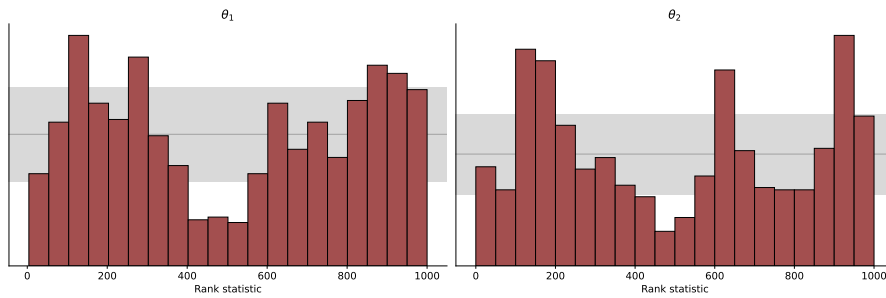
	GAN	Energy 3	Energy 5	Energy 10	Energy 20	Kernel 3	Kernel 5	Kernel 10	Kernel 20
1000	4799	578	690	759	896	585	613	651	852
10000	8163	1775	1917	2415	3228	1708	1883	2329	3267
100000	30232	9266	9388	9903	10805	9283	9479	9859	10902

Table 16: Two Moons: epoch at which early stopping occurred; the max number of training epochs was 20000.

	GAN	Energy 3	Energy 5	Energy 10	Energy 20	Kernel 3	Kernel 5	Kernel 10	Kernel 20
1000	20000	20000	20000	20000	20000	20000	20000	20000	20000
10000	20000	20000	20000	20000	20000	20000	20000	20000	20000
100000	20000	20000	20000	20000	20000	20000	20000	20000	20000



(a) GAN



(b) Energy Score, $m = 20$

Figure 8: Two Moons: Simulation-Based Calibration results represented as rank histograms; for each dimension of θ , the corresponding histogram represents the distribution of the rank of the true parameter value in a set of samples from the approximate posterior. If the approximate posterior is calibrated, histogram bars should be in the grey region with 99% probability.

E.3 Shallow Water Model

In Figure 9, we show results, analogously to what done in Figure 2, for all methods. Table 17 reports the different performance metrics, the runtime and the early stopping epoch for all methods. Finally, Figure 10 reports Simulation-Based Calibration results for all SR methods.

Table 17: Shallow Water model: performance metrics, runtime and early stopping epoch for all methods. We do not train GAN from scratch but rather relied on the trained network obtained in Ramesh et al. [2022]. The training time we report here is what is mentioned in Ramesh et al. [2022], which used two GPUs for training (in contrast, we used a single GPU for the SR methods). For the same reason, we do not report the epoch at which GAN training was early stopped.

	RMSE ↓	Cal. Err. ↓	R ² ↑	Runtime (sec)	Early stopping epoch
Energy	0.05 ± 0.01	0.03 ± 0.02	0.87 ± 0.05	51328	10400
Energy patched 10 20	0.05 ± 0.01	0.03 ± 0.02	0.89 ± 0.05	60017	12400
Energy patched 5 10	0.06 ± 0.01	0.03 ± 0.02	0.86 ± 0.06	49626	9600
Kernel	0.06 ± 0.01	0.11 ± 0.05	0.84 ± 0.06	39608	7800
Kernel patched 10 20	0.06 ± 0.01	0.09 ± 0.04	0.86 ± 0.06	47642	9000
Kernel patched 5 10	0.06 ± 0.01	0.09 ± 0.04	0.86 ± 0.06	44590	9200
GAN	0.07 ± 0.01	0.12 ± 0.09	0.78 ± 0.05	≈345600	-

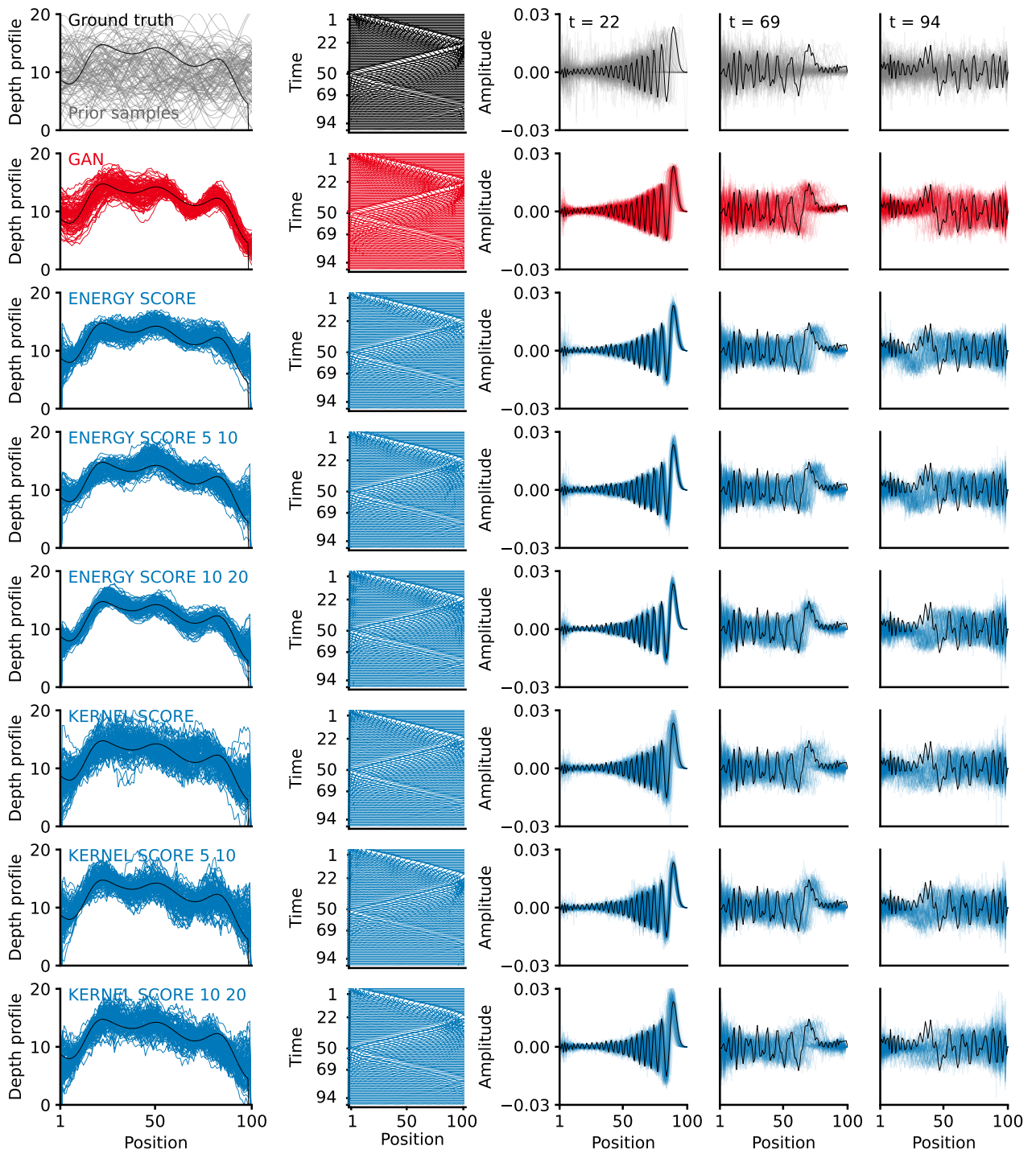
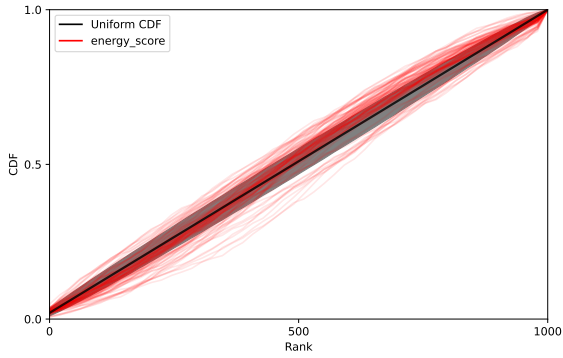
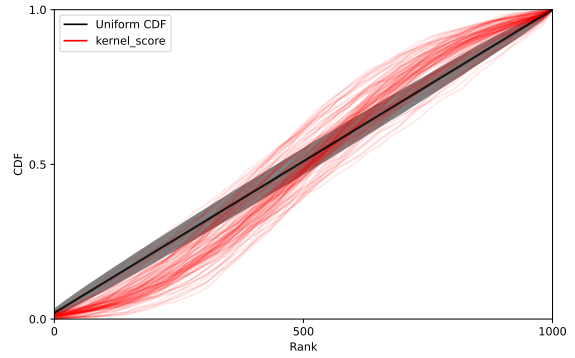


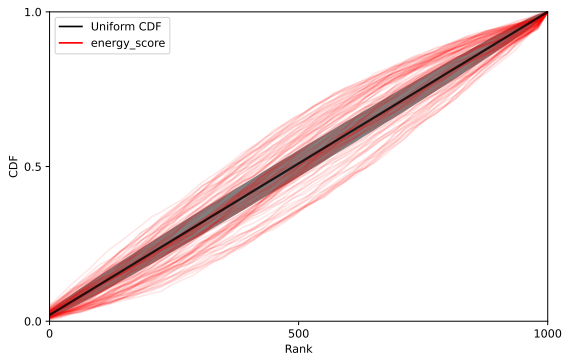
Figure 9: Shallow water model: inference results with all methods. See Figure 2 for a description of the different panels.



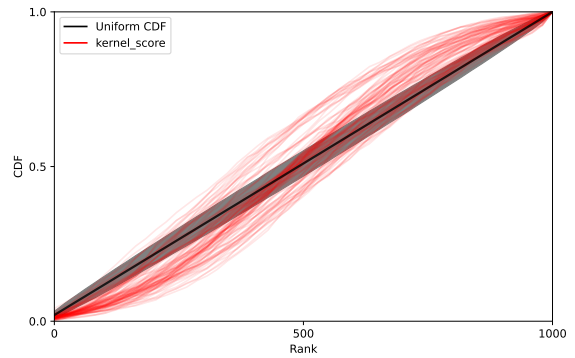
(a) Energy Score



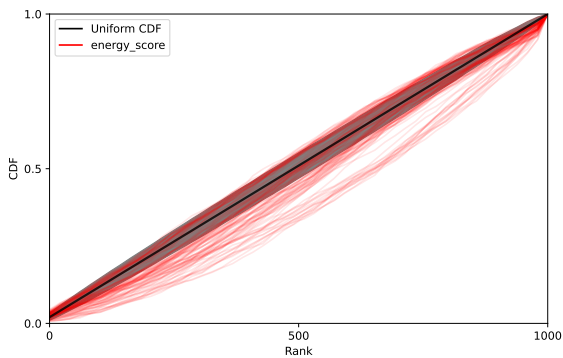
(b) Kernel Score



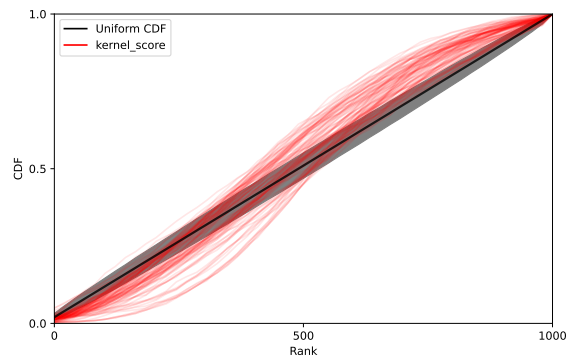
(c) Energy Score patched 5, 10



(d) Kernel Score patched 5, 10



(e) Energy Score patched 10, 20



(f) Kernel Score patched 10, 20

Figure 10: Shallow Water model: Simulation Based Calibration for all SR methods. Each line corresponds to a single dimension of θ and represents the CDF of the rank of the true parameter value with respect to a set of posterior samples. A calibrated posterior implies uniform CDF (diagonal black line, with associated 99% confidence region for the considered number of samples in gray).

E.4 Camera model

In Figure 11, we show results, analogously to what is done in Figure 4, for all methods. Table 18 reports the different performance metrics, runtime, and early stopping epoch for all methods.

Table 18: Noisy Camera model: performance metrics, runtime and early stopping epoch for all methods.

	RMSE ↓	Cal. Err. ↓	R ² ↑	Runtime (sec)	Early stopping epoch
GAN	0.25 ± 0.19	0.50 ± 0.00	-23.94 ± 366.08	45398	3600
Energy	0.08 ± 0.05	0.36 ± 0.12	-24.39 ± 450.13	24555	4200
Energy patched 5 8	0.06 ± 0.05	0.36 ± 0.12	-2.14 ± 55.86	22633	4000
Energy patched 7 14	0.07 ± 0.05	0.37 ± 0.12	-10.33 ± 227.38	24033	3600
Kernel	0.06 ± 0.05	0.32 ± 0.15	-7.22 ± 164.26	21862	3200
Kernel patched 5 8	0.07 ± 0.05	0.36 ± 0.12	-10.29 ± 222.12	22545	3200
Kernel patched 7 14	0.10 ± 0.06	0.38 ± 0.11	-144.56 ± 2952.80	20605	3600



Figure 11: Noisy Camera model: ground truth and posterior inference with all methods, for a set of observations (each observation corresponds to a column). The first two rows represent the ground truth values of θ and the corresponding observation y_o . The remaining rows represent the mean and Standard Deviation (SD) for all methods.

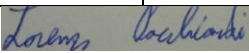
Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).


Title of Paper	Likelihood-Free Inference with Generative Neural Networks via Scoring Rule Minimization
Publication Status	<input type="checkbox"/> Published <input type="checkbox"/> Accepted for Publication <input type="checkbox"/> Submitted for Publication <input checked="" type="checkbox"/> Unpublished and unsubmitted work written in a manuscript style
Publication Details	Joint work with Prof. Ritabrata Dutta (University of Warwick).

Student Confirmation

Student Name:	Lorenzo Pacchiardi		
Contribution to the Paper	I am the first author of this paper. After I originally thought of using scoring rules to train generative networks, the idea of applying this method to Likelihood-Free Bayesian Inference was raised by Prof. Geoff Nicholls in a meeting with Prof. Dutta and myself. I then formalized the methodology, coded the method, run the simulations and wrote most of the paper. Prof. Dutta advised along the way and corrected the paper draft.		
Signature		Date	1 st September 2022

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Professor Geoff Nicholls			
Supervisor comments			
Signature		Date	15-09-22

This completed form should be included in the thesis, at the end of the relevant chapter.

Chapter 5

Probabilistic Forecasting with Generative Networks via Scoring Rule Minimization

Unpublished and unsubmitted work.

Probabilistic Forecasting with Generative Networks via Scoring Rule Minimization

Lorenzo Pacchiardi^{1*}, Rilwan A. Adewoyin^{2,3}, Ritabrata Dutta², Peter Dueben⁴

¹*Department of Statistics, University of Oxford, UK*

²*Department of Statistics, University of Warwick, UK*

³*Department of Computer Science and Engineering, Southern University of Science and Technology, China*

⁴*Earth System Modelling Section, ECMWF, UK*

Abstract

Generative networks are often trained to minimize a statistical divergence between the reference distribution and the generative one in an adversarial setting. Some works trained instead generative networks to minimize Scoring Rules, functions assessing how well the generative distribution matches each training sample individually. We show how the Scoring Rule formulation easily extends to the so-called prequential (predictive-sequential) score, whose minimization allows performing probabilistic forecasting with generative networks. This objective leads to adversarial-free training, therefore easily avoiding uncertainty underestimation due to mode collapse, which is a common issue in the adversarial setting and undesirable for probabilistic forecasting. We provide consistency guarantees for the minimizer of the prequential score and employ that to perform probabilistic forecasting for two chaotic dynamical models and a benchmark dataset of global weather observations. For this last example, we define scoring rules for spatial data by drawing from the relevant literature, with which we obtain better uncertainty quantification with little hyperparameter tuning compared to adversarial training.

1 Introduction

In many disciplines (for instance econometrics and meteorology), practitioners want to forecast the future state of a phenomenon. Providing prediction uncertainty (ideally by stating a full probability distribution) is often essential. This task is called *probabilistic forecasting* [Gneiting and Katzfuss, 2014] and is commonplace in Numerical Weather Prediction (NWP, Palmer, 2012), where physics-based models are run multiple times to obtain an ensemble of forecasts representing the possible evolution of the weather Leutbecher and Palmer [2008]. To assess the performance of NWP systems, people commonly use Scoring Rules (SRs, Gneiting and Raftery, 2007), functions quantifying the quality of a probabilistic forecast with respect to the observed outcome.

Here, we use *generative (neural) networks* to provide probabilistic forecasts. In a generative network, a neural network maps a latent random variable to the required output space; hence, samples on the latter are obtained by transforming latent variable draws. As the density is inaccessible, the distribution is implicitly defined and specialized techniques are necessary to train generative networks. Among those, the popular Generative Adversarial Networks (GANs, Goodfellow et al., 2014, Mirza and Osindero, 2014, Nowozin et al., 2016, Arjovsky et al., 2017) framework trains a generative network by defining a min-max game against a competitor, termed *critic*. However, adversarial training is unstable: it requires ad-hoc strategies [Gulrajani et al., 2017] and careful hyperparameter tuning [Salimans et al., 2016] but, even so, the trained generative network may not fully capture the data distribution, a phenomenon referred to as *mode collapse* [Goodfellow, 2016, Isola et al., 2017, Arora et al., 2017, Bellemare et al., 2017, Arora et al., 2018, Richardson and Weiss, 2018]. This prevents practitioners from reliably applying GANs to tasks where calibrated uncertainty quantification is paramount, such as probabilistic forecasting. Additionally, it is unclear how to extend the GAN training objective to

*Corresponding author: lorenzo.pacchiardi@stats.ox.ac.uk.

the temporal data considered in probabilistic forecasting. Indeed, the adversarial framework is derived from divergences between probability distributions and considers data as independent and identically distributed samples from one of those distributions.

Therefore, motivated by the use of scoring rules to evaluate traditional forecasting systems, we propose to train generative networks to minimize scoring rule values. Given a recorded temporal sequence of the phenomenon of interest, we use the generative network to forecast all steps of the sequence conditioned on the past. Then, our objective is the average over steps of the scoring rule between forecasts and realizations. In contrast to the adversarial framework, this so-called *prequential* (predictive-sequential, Dawid, 1984) scoring rule captures the temporal structure of the data. Additionally, the minimizer of the prequential scoring rule enjoys consistency under mild conditions on the temporal sequence. Furthermore, our proposal allows adversarial-free training through a reparametrization trick [Kingma and Welling, 2014] for SRs defined as expectations over the generative distribution. Training with our objective is therefore drastically easier than with GAN, requires less hyperparameter tuning and easily avoids mode collapse. More in detail, our contributions are:

- We introduce a novel training objective for probabilistic forecasting based on a prequential scoring rule.
- Under stationarity and mixing conditions of the time series, we prove that the minimizer of the prequential scoring rule coincides asymptotically with that of the expected prequential scoring rule. Importantly, the latter corresponds to the true parameter value if the distribution induced by the generative network is well-specified.
- We leverage previous works in meteorology [Gneiting and Raftery, 2007, Scheuerer and Hamill, 2015] and design training objectives for high-dimensional spatio-temporal data, enabling good performance with no need for a learnable data transformation.
- We test our method and state-of-the-art adversarial approaches on two chaotic models and a spatio-temporal weather dataset. We find our method to be more stable and perform better, particularly in terms of uncertainty quantification of the forecast.

The rest of the paper is organized as follows. In Sec. 2, we review the standard setup of generative networks training via divergence minimization, introduce the Scoring Rules formulation and connect to previous works which used it. In Sec. 3, which contains the main contributions of our work, we formalize the training objective for probabilistic forecasting and discuss SRs for time-series and spatial data. We discuss some related works in Sec. 4 and show simulation results in Sec. 5. We conclude in Sec. 6.

Notation We use upper case X, Y and Z to denote random variables, and their lower-case counterpart to denote observed values. Bold symbols denote vectors, and subscripts to bold symbols denote sample index (for instance, \mathbf{y}_t). Instead, subscripts to normal symbols denote component indices (for instance, y_i is the i -th component of \mathbf{y} , and $y_{t,j}$ is the j -th component of \mathbf{y}_t). Finally, we use notation $\mathbf{y}_{j:k} = (\mathbf{y}_j, \mathbf{y}_{j+1}, \dots, \mathbf{y}_{k-1}, \mathbf{y}_k)$, for $j \leq k$.

2 Background

2.1 Generative networks via divergence minimization

A generative network represents distributions on some space \mathcal{Y} via a map $h_\phi : \mathcal{Z} \rightarrow \mathcal{Y}$ transforming samples from a probability distribution Q over the space \mathcal{Z} ; the map is parametrized by a Neural Network (NN) with weights ϕ . Samples from P^ϕ are obtained by generating $\mathbf{z} \sim Q$ and computing $h_\phi(\mathbf{z}) \in \mathcal{Y}$; therefore, expectations $\mathbb{E}_{\mathbf{Y} \sim P^\phi}[g(\mathbf{Y})]$ can be computed by $\mathbb{E}_{\mathbf{z} \sim Q}[g(h_\phi(\mathbf{z}))]$.

Assume now we observe data from a distribution P^* and want to tune ϕ so that P^ϕ is as close as possible to P^* . A divergence $D(P^*||P^\phi)$ is a function of two distributions such that $D(P^*||P^\phi) \geq 0$

and $D(P^*||P^\phi) = 0 \iff P^* = P^\phi$. Therefore, for a given D , we can attempt solving:

$$\arg \min_{\phi} D(P^*||P^\phi). \quad (1)$$

Various proposed approaches differ according to (i) their choice of divergence D and (ii) how they estimate the optimal solution in Eq. (1) using samples from P^* and P^ϕ . A popular strategy is choosing D to be an f -divergence (termed f -GAN, Nowozin et al., 2016), in which case a variational lower bound can be obtained:

$$D_f(P^*||P^\phi) \geq \sup_{c \in \mathcal{C}} (\mathbb{E}_{\mathbf{Y} \sim P^*} c(\mathbf{Y}) - \mathbb{E}_{\mathbf{X} \sim P^\phi} f^*(c(\mathbf{X}))),$$

where f^* is the Fenchel conjugate of the function f (Appendix B.1.1) and \mathcal{C} is any set of functions from \mathcal{Y} to the domain of f^* . By representing the set \mathcal{C} by a neural network c_ψ (termed *critic*) with parameters $\psi \in \Psi$, the problem in Eq. (1) is:

$$\min_{\phi} \max_{\psi} (\mathbb{E}_{\mathbf{Y} \sim P^*} c_\psi(\mathbf{Y}) - \mathbb{E}_{\mathbf{X} \sim P^\phi} f^*(c_\psi(\mathbf{X}))).$$

The WGAN of Arjovsky et al. [2017], which uses the 1-Wasserstein distance, has a similar objective to Eq. (2), differing mainly in taking \mathcal{C} to be the set of 1-Lipschitz functions. Details in Appendix B.1.2.

The problem in Eq. (2) is solved by alternating optimization steps over ψ and ϕ ; the expectations are estimated via samples from both P^* (i.e., a minibatch of observations) and from P^ϕ . This approach is termed *adversarial* as P^ϕ and c_ψ respectively aim to minimize and maximize the same objective.

Adversarial training of generative networks is however unstable and difficult. A well-known consequence of unstable adversarial training is mode collapse [Goodfellow, 2016, Isola et al., 2017, Arora et al., 2017, Bellemare et al., 2017, Arora et al., 2018, Richardson and Weiss, 2018], in which the generative distribution underestimates uncertainty and, in extreme cases, can collapse to a single point. Mode collapse has been related to the approximations involved in adversarial training: Arora et al. [2017] showed that mode collapse can arise due to finite capacity of the critic c_ψ , while Bellemare et al. [2017] and Bińkowski et al. [2018] respectively linked it to using finite data and a finite number of steps in optimizing the c_ψ network and subsequently using it to obtain gradient estimates for ϕ , which are thus biased.

To avoid adversarial training altogether and bypass the above issues, Moment Matching Networks Li et al. [2015], Dziugaite et al. [2015] are trained by considering D to be the squared Maximum Mean Discrepancy (MMD) induced by a positive definite kernel k :

$$D_k(P^*, P^\phi) := \mathbb{E} [k(\mathbf{X}, \mathbf{X}') - 2k(\mathbf{X}, \mathbf{Y}) + k(\mathbf{Y}, \mathbf{Y}')], \quad \mathbf{X}, \mathbf{X}' \sim P^\phi, \quad \mathbf{Y}, \mathbf{Y}' \sim P^* \quad (3)$$

From Eq. (3), an empirical unbiased estimate of D_k and its gradients can be obtained without introducing a critic network. However, using a fixed kernel on raw data can yield small discriminative power (as in the case of images, where numerical values have little meaning), leading to poor fit of P^ϕ to P^* . Li et al. [2017] suggested therefore applying a learnable transformation before computing the kernel, with parameters trained to maximize the MMD. This approach, termed MMD-GAN, leads again to an adversarial setting and to the issues mentioned above. Details in Appendix B.1.3.

Conditional setting To represent a conditional distribution $P^\phi(\cdot|\theta)$, $\theta \in \Theta$, a map $h_\phi : \mathcal{Z} \times \Theta \rightarrow \mathcal{Y}$ can be used; similarly to above, samples from $P^\phi(\cdot|\theta)$ for fixed θ can be obtained via $h_\phi(\mathbf{z}; \theta)$, $\mathbf{z} \sim Q$. In this way, f -GAN, WGAN and MMD-GAN can all be easily extended to the setting in which we have data

$$(\theta_i, \mathbf{y}_i)_{i=1}^n, \text{ where } \theta_i \sim \Pi \text{ and } \mathbf{y}_i \sim P^*(\cdot|\theta_i), \quad (4)$$

and want $P^\phi(\cdot|\theta) = P^*(\cdot|\theta)$ Π -almost everywhere. For instance, the f -GAN objective in Eq. (2) becomes:

$$\min_{\phi} \max_{\psi} \mathbb{E}_{\theta \sim \Pi} (\mathbb{E}_{\mathbf{Y} \sim P^*(\cdot|\theta)} c_\psi(\mathbf{Y}; \theta) - \mathbb{E}_{\mathbf{Y} \sim P^\phi(\cdot|\theta)} f^*(c_\psi(\mathbf{Y}; \theta))),$$

where now $c_\psi : \mathcal{Y} \times \Theta \rightarrow \text{dom}_{f^*}$. More details can be found in Appendix B.1.

2.2 Generative networks via scoring rules minimization

Here, we review a formulation for training generative networks which, in some cases, is intrinsically adversarial-free, as for Moment Matching Networks. This different perspective is based on Scoring Rules (SR), which we introduce next, and will allow us to design objective functions suitable for probabilistic forecasting of time-series (Sec. 3.1) and for tackling spatial data (Sec. 3.2).

A SR S is a function of a distribution and an observation; specifically, $S(P^\phi, \mathbf{y})$ represents the penalty assigned to the distribution P^ϕ when \mathbf{y} is observed Gneiting and Raftery [2007], Dawid and Musio [2014]. If \mathbf{y} is the realization of a random variable $\mathbf{Y} \sim P^*$, the expected SR is: $S(P^\phi, P^*) := \mathbb{E}_{\mathbf{Y} \sim P^*} S(P^\phi, \mathbf{Y})$. S is said to be *proper* relative to a set of distributions \mathcal{P} if the expected Scoring Rule is minimized in P^ϕ when $P^\phi = P^*$:

$$S(P^*, P^*) \leq S(P^\phi, P^*) \quad \forall P^\phi, P^* \in \mathcal{P}.$$

Moreover, S is *strictly proper* relative to \mathcal{P} if $P^\phi = P^*$ is the unique minimum. In plain words, P^ϕ minimizes an expected proper SR S if it has some features corresponding to those of the data distribution P^* ; if S is strictly proper, P^* and P^ϕ coincide.

For a strictly proper SR S , the quantity $D(P^\phi || P^*) := S(P^\phi, P^*) - S(P^*, P^*)$ is a statistical divergence, as in fact $D(P^\phi || P^*) \geq 0$ and $D(P^\phi || P^*) = 0 \iff P^\phi = P^*$. However, not all divergences can be written in terms of a SR (that is not possible, e.g., for the Wasserstein distance).

SRs have been previously used to train conditional generative networks in Bouchacourt et al. [2016], Gritsenko et al. [2020], Harakeh and Waslander [2021], where the authors considered:

$$\min_{\phi} \mathbb{E}_{\theta \sim \Pi} \mathbb{E}_{\mathbf{Y} \sim P^*(\cdot | \theta)} S(P^\phi(\cdot | \theta), \mathbf{Y}); \quad (5)$$

for strictly proper S , the solution is $P^\phi(\cdot | \theta) = P^*(\cdot | \theta)$ Π -almost everywhere. With data as in Eq. (4), we can replace Eq. (5) with:

$$\min_{\phi} \frac{1}{n} \sum_{i=1}^n S(P^\phi(\cdot | \theta_i), \mathbf{y}_i), \quad (6)$$

i.e., we evaluate $P^\phi(\cdot | \theta_i)$ according to how well it predicts \mathbf{y}_i . The objective in Eq. (6) is an unbiased estimate of that in Eq. (5); therefore, to train P^ϕ via Stochastic Gradient Descent, it is enough to obtain unbiased estimates of $\nabla_{\phi} S(P^\phi(\cdot | \theta_i), \mathbf{y}_i)$. That is possible whenever S is defined via an expectation over P^ϕ , in which case we can train P^ϕ without adversarial setup. More details in Appendix C.

Eq. (5) is equivalent to $\min_{\phi} \mathbb{E}_{\theta \sim \Pi} D(P^\phi(\cdot | \theta) || P^*(\cdot | \theta))$, where D is the divergence associated to S . For instance, conditional Moment Matching Network corresponds to setting S in Eq. (5) to be the *Kernel Score* [Gneiting and Raftery, 2007]:

$$S_k(P^\phi, \mathbf{y}) := \mathbb{E}[k(\mathbf{X}, \mathbf{X}')] - 2 \cdot \mathbb{E}[k(\mathbf{X}, \mathbf{y})], \quad \mathbf{X}, \mathbf{X}' \sim P^\phi \quad (7)$$

where k is a positive definite kernel. We can obtain unbiased gradient estimates for Eq. (7) by replacing expectations with empirical means over simulations from P^ϕ . Under some conditions on P^ϕ and for a specific class of kernels, S_k is strictly proper (see Sec. 1.2.2 in the introduction of the present thesis). Additionally, the Energy Score used in Bouchacourt et al. [2016], Gritsenko et al. [2020], Harakeh and Waslander [2021] can be obtained from S_k by choosing $k(\mathbf{x}, \mathbf{y}) = -\|\mathbf{y} - \mathbf{x}\|^\beta$ for $\beta \in (0, 2)$ Gneiting and Raftery [2007]. See more details in Sec. 1.2.2 in the introduction of the present thesis. In the following, we will fix $\beta = 1$ for the Energy Score and use the Gaussian kernel in the Kernel Score.

3 Generative networks for spatio-temporal models via SR minimization

We will now extend the SR formulation to a training objective for probabilistic forecasting for time-series (Sec. 3.1). Later (Sec. 3.2), we will exploit the SR formulation to tackle high dimensional spatial data, by relying on scores studied in the probabilistic forecasting and meteorology literature Gneiting and Raftery [2007], Scheuerer and Hamill [2015]. The resulting objectives can be minimized without recurring to adversarial training.

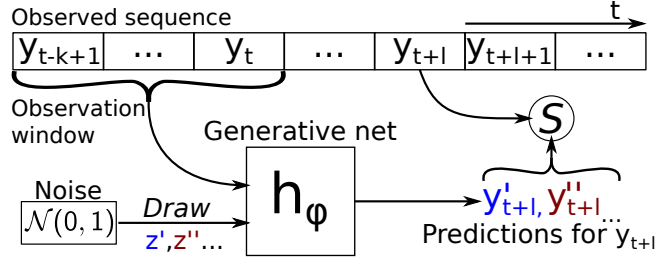


Figure 1: Estimation of the SR evaluating the forecast $P_{t+l}^\phi(\cdot|\mathbf{y}_{t-k+1:t})$ for the realization \mathbf{y}_{t+l} . The prequential SR is obtained by repeating this procedure for all t 's and summing the scores.

3.1 Time-series probabilistic forecasting via the prequential SR

Consider a temporal stochastic process $(\mathbf{Y}_1, \mathbf{Y}_2, \dots, \mathbf{Y}_t, \dots) = (\mathbf{Y}_t)_t \sim P^*$, where $\mathbf{Y}_t \in \mathcal{Y}$; in general, \mathbf{Y}_t 's are *not* independent. For a generic distribution P for $(\mathbf{Y}_t)_t$, we denote by P_t the marginal distribution for \mathbf{Y}_t , and by $P_{r:s}$ the marginal distribution for $\mathbf{Y}_{r:s}$; the conditional distribution for $\mathbf{Y}_t|\mathbf{y}_{u:v}$ will be denoted by $P_t(\cdot|\mathbf{y}_{u:v})$ and similar for $\mathbf{Y}_{r:s}$.

Having observed $\mathbf{y}_{1:t}$, we produce a *probabilistic forecast* for \mathbf{Y}_{t+l} for a given lead time l via a generative network conditioned on the last k observations, $P_{t+l}^\phi(\cdot|\mathbf{y}_{t-k+1:t})$. By repeating this procedure for all t 's in a recorded window of length T , P^ϕ induces a joint distribution over $\mathbf{Y}_{k+l:T}$ given $\mathbf{y}_{1:k+l-1}$, denoted $P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1})$ (as in fact the generative network cannot forecast the first elements of the sequence $\mathbf{y}_{1:k+l-1}$). For each t , we evaluate the forecast performance via $S(P_{t+l}^\phi(\cdot|\mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l})$ for a SR S (Fig. 1); summing this for all t 's, we obtain:

$$S_T(P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1}), \mathbf{y}_{k+l:T}) := \sum_{t=k}^{T-l} S(P_{t+l}^\phi(\cdot|\mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l}). \quad (8)$$

The above notation only makes sense as the distribution for \mathbf{Y}_{t+l} in $P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1})$ depends only on $\mathbf{Y}_{t-k+1:t}$; otherwise, $\mathbf{y}_{1:k+l-1}$ should also appear explicitly in the conditioning of P_{t+l}^ϕ . This is formalized by the following property (which recovers the standard k -Markov property for $l = 1$):

Definition 3.1. A probability distribution $P_{1:T}$ is k -Markovian with lag l if, assuming it has density $p_{1:T}$ with respect to some base measure, it can be decomposed as:

$$p_{1:T}(\mathbf{y}_{1:T}) = p_{1:k+l-1}(\mathbf{y}_{1:k+l-1}) \prod_{t=k}^{T-l} p_{t+l}(\mathbf{y}_{t+l}|\mathbf{y}_{t-k+1:t}).$$

We call S_T in Eq. (8) the *prequential* (or *predictive-sequential*) score [Dawid, 1984, Dawid and Musio, 2015], as it evaluates sequential predictions. S_T is a SR for distributions over $\mathbf{Y}_{k+l:T}|\mathbf{y}_{1:k+l-1}$ which are k -Markovian with lag l .

We propose therefore to learn ϕ by:

$$\hat{\phi}_T(\mathbf{y}_{1:T}) := \arg \min_{\phi} S_T(P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1}), \mathbf{y}_{k+l:T}), \quad (9)$$

which picks the best ϕ for which the average l -steps ahead forecast in the training data is optimal according to S . Operationally, Eq. (9) can be solved in the same way as Eq. (6), i.e., by simulating from P^ϕ for each observation window $\mathbf{y}_{t-k+1:t}$ in a training batch, unbiasedly estimating the SR S and descending the gradient. However, contrarily to the independent-data setting of Eq. (6), Eq. (9) cannot be motivated as an empirical estimate of an expected SR, as the different \mathbf{y}_t 's are dependent. Still, we show below how the empirical minimizer $\hat{\phi}_T(\mathbf{Y}_{1:T})$ converges, under some stationarity and mixing conditions of $(\mathbf{Y}_t)_t$, to a fixed quantity corresponding to the minimizer of a proper SR. First, Theorem 3.2 below (proven in Appendix A.1.2) establishes propriety properties of S_T :

Theorem 3.2. *If S is (strictly) proper, then S_T is (strictly) proper for distributions over $\mathbf{Y}_{k+l:T}|\mathbf{y}_{1:k+l-1}$ which are k -Markovian with lag l .*

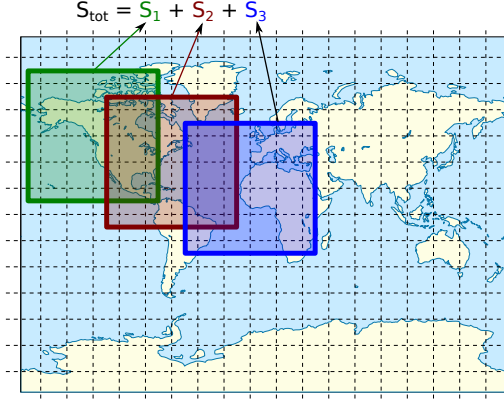


Figure 2: Patched SR: a SR for multivariate data is computed on localized patches, and the resulting values are summed.

Consider now the following two quantities:

$$\begin{aligned}\tilde{\phi}_T(\mathbf{y}_{1:k+l-1}) &:= \arg \min_{\phi} \mathbb{E}_{\mathbf{Y}_{k+l:T} | \mathbf{y}_{1:k+l-1}} S_T(P_{k+l:T}^{\phi}(\cdot | \mathbf{y}_{1:k+l-1}), \mathbf{Y}_{k+l:T}), \\ \phi_T^* &:= \arg \min_{\phi} \mathbb{E} S_T(P_{k+l:T}^{\phi}(\cdot | \mathbf{Y}_{1:k+l-1}), \mathbf{Y}_{k+l:T}).\end{aligned}$$

$\tilde{\phi}_T(\mathbf{y}_{1:k+l-1})$ is the minimizer of the expected prequential SR with respect to $\mathbf{Y}_{k+l:T} | \mathbf{y}_{1:k+l-1}$ which, from Theorem 3.2, is minimized when the true distribution for $\mathbf{Y}_{k+l:T} | \mathbf{y}_{1:k+l-1}$ is recovered. ϕ_T^* instead minimizes the expectation of S_T with respect to the full sequence $\mathbf{Y}_{1:T}$.

Now, each term in the sum defining S_T depends on a finite number of observations; therefore, if $(\mathbf{Y}_t)_t$ satisfies some mixing and stationarity properties, we expect $\tilde{\phi}_T(\mathbf{y}_{1:k+l-1})$ to not depend on $\mathbf{y}_{1:k+l-1}$ for large T ; similarly, we expect the empirical estimator $\hat{\phi}_T(\mathbf{y}_{1:T})$ to converge to a fixed quantity. The following Theorem proves such consistency of $\hat{\phi}_T(\mathbf{y}_{1:T})$ and $\tilde{\phi}_T(\mathbf{y}_{1:k+l-1})$ to ϕ_T^* .

Theorem 3.3. *Assume ϕ_T^* and $\tilde{\phi}_T(\mathbf{y}_{1:k+l-1})$ are unique for each fixed $\mathbf{y}_{1:k+l-1}$, and $(\mathbf{Y}_t)_t$ is asymptotically stationary and satisfies some mixing properties. Then, under some regularity conditions, it exists a metric d such that $d(\phi_T^*, \hat{\phi}_T(\mathbf{Y}_{1:T})) \rightarrow 0$ and $d(\tilde{\phi}_T(\mathbf{Y}_{1:k+l-1}), \hat{\phi}_T(\mathbf{Y}_{1:T})) \rightarrow 0$ when $T \rightarrow \infty$ almost surely with respect to $(\mathbf{Y}_t)_t \sim P^*$. It also follows that $d(\tilde{\phi}_T(\mathbf{Y}_{1:k+l-1}), \phi_T^*) \rightarrow 0$.*

The precise statement of Theorem 3.3 is given and proven in Appendix A.2; our proof holds when P_{t+l}^{ϕ} depends on t only through the value of the past observations, which is our case of interest as we use the same generative network for all t 's. In plain words, the mixing properties require \mathbf{Y}_{t-m} and \mathbf{Y}_t to become independent as $m \rightarrow \infty$, while asymptotic stationarity means that the average of the marginal distributions over different t converges to a constant distribution.

Under the assumptions of Theorem 3.3, with large enough T , $\hat{\phi}_T(\mathbf{y}_{1:T})$ and $\tilde{\phi}_T(\mathbf{y}_{1:k+l-1})$ will both be independent of the observed sequence $\mathbf{y}_{1:T}$ and will converge to ϕ_T^* . Therefore, minimizing the empirical prequential SR in Eq. (8) asymptotically recovers the minimizer of an expected proper SR, which additionally does not depend on the initial conditions of the sequence $\mathbf{y}_{1:k+l-1}$.

3.2 Scoring rules for spatial data

In contrast to multivariate, spatial data (such as the dataset used in Sec. 5.2) is structured: the relation between different entries depends on their spatial distance. Computing, say, the Kernel SR in Eq. (7) would not capture this structure; as discussed in Sec 3.1 in Chapter 4 of the present thesis, one way to address that is computing the SR on patches and cumulating it; see Fig. 2 for a representation. Another option is the following:

Variogram Score Say now $\mathcal{Y} \subseteq \mathbb{R}^d$. For any $p > 0$, the Variogram Score Scheuerer and Hamill [2015] is defined as:

$$S_v^{(p)}(P^\phi, \mathbf{y}) := \sum_{i,j=1}^d w_{ij} (|y_i - y_j|^p - \mathbb{E}_{\mathbf{X} \sim P^\phi} |X_i - X_j|^p)^2,$$

where $w_{ij} > 0$ are fixed scalars. Scheuerer and Hamill [2015] set w_{ij} to be inversely proportional to the distance of locations i and j to capture the spatial structure. However, $S_v^{(p)}$ is proper but not strictly so: it is invariant to change of sign and shift of all entries of \mathbf{X} by a constant, and only depends on the moments of P^ϕ up to order $2p$ Scheuerer and Hamill [2015]. We will fix $p = 1$ in the rest of our work.

Probabilistic forecasting for spatial data Inserting the spatial SRs discussed above in the prequential score in Eq. (8) enables probabilistic forecasting for spatial data using generative networks. For the patched SR, unbiased gradient estimates can be computed if the underlying SR admits unbiased gradient estimates (which is the case for the Energy and Kernel SRs, see Sec. 1.2.2 in the introduction of the present thesis). For the Variogram Score, instead, it is immediate to obtain an unbiased estimate by:

$$\hat{S}_v^{(p)}(\{\mathbf{x}_j\}_{j=1}^m, \mathbf{y}) = \sum_{i,j=1}^d w_{ij} \left(|y_i - y_j|^p - \frac{1}{m} \sum_{k=1}^m |x_{k,i} - x_{k,j}|^p \right)^2,$$

where $\mathbf{x}_j \sim P, j = 1, \dots, m$.

4 Related works

In the statistics literature, Dawid et al. [2016] studied a SR parameter estimator for independent and identically distributed data; instead, Dawid and Musio [2013] used SRs to infer parameters for spatial models, considering the conditional distribution in each location given all the others to be available; finally, Dawid and Musio [2015] considered model selection based on SRs and studied a prequential application. Properties of prequential losses for forecasting systems, such as our Eq. (8), were investigated in Skouras [1998], which also gave consistency results similar to our Theorem 3.3.

Kwon and Park [2019], Koochali et al. [2021], Bihlo [2021] and Ravuri et al. [2021], among others, all used GANs for forecasting. However, they all considered the training samples to be independent in their formulation and did not study theoretically the consequence of using dependent data. Bihlo [2021] considered a task similar to ours (which we privileged for ease of reproducibility as it is a standardized dataset) and found the GAN to underestimate uncertainty, so they considered a GANs ensemble increase the uncertainty. Instead, Ravuri et al. [2021] exploited GANs for a precipitation *nowcasting* task (i.e., predicting for small lead time), achieving good deterministic and probabilistic performance with large amount of architecture hand-tuning and computing power.

Deterministic forecasting with NNs for the WeatherBench dataset (Sec. 5.2) was studied extensively Dueben and Bauer [2018], Scher [2018], Scher and Messori [2019], Weyn et al. [2019]. Fewer studies tackled probabilistic forecasting: Scher and Messori [2021] combined deterministic NNs with ad-hoc strategies, not guaranteed to lead to the correct distribution. Clare et al. [2021] binned instead the data, thus mapping the problem to that of estimating a categorical distribution.

5 Simulation study

We first study two low-dimensional time-series model which allow exhaustive hyperparameter tuning and architecture comparison, but still have challenging dynamics due to their chaotic nature. We then move to a high-dimensional spatio-temporal meteorology dataset. For all examples, we train generative models with the Energy and the Kernel Scores (Sec. 1.2.2 in the introduction of the present thesis) and their sum, termed *Energy-Kernel Score* (a strictly proper SR due to Lemma 1 in Chapter 4 of the present thesis). Additional SRs, discussed later in Sec. 5.2, are used for the meteorology example. For the Kernel Score, we use the Gaussian kernel with bandwidth γ tuned from the validation set

(Appendix E.1). For all SR methods, we use 10 forecasts from the generator for each observation window to estimate SR values during training; however, performance does not degrade when using as few as 3 simulations (Appendix F.3.2), which lowers the computational cost (Appendix F.3.3). We compare with the original GAN Goodfellow et al. [2014] and WGAN with gradient penalties (WGAN-GP, Gulrajani et al., 2017). The latent variable \mathbf{Z} has independent components with standard normal distribution. To put in context the deterministic performance of the probabilistic methods, we compare with deterministic networks trained to minimize the standard regression loss.

All datasets consist of a long time series, which we split into training, validation and test set. We use the validation set for early stopping and hyperparameter tuning and report final performance on the test set. The adversarial methods do not allow early stopping or hyperparameter selection using the training objective, as the generator loss depends on the critic state. For these methods, therefore, we use other metrics to pick the best hyperparameters (see below).

On the test set, we assess the calibration of the probabilistic forecasts by the *calibration error* (the discrepancy between credible intervals in the forecast distribution and the actual frequencies). We also evaluate how close the means of the forecast distributions are to the observation by the *Normalized Root Mean-Square Error* (NRMSE) and the *coefficient of determination* R^2 ; we detail all these metrics in Appendix D. As all these metrics are for scalar variables, we compute their values independently for each component and report their average (standard deviation in Appendix F).

Our simulations show how the SR methods are easier to train and provide better uncertainty quantification. The adversarial methods require more hyperparameter tuning. We find the original GAN to be unstable and very poor at quantifying uncertainty due to mode collapse; WGAN-GP performs better, but has still inferior performance with respect to the SR approaches. Likely, ad-hoc adversarial training strategies could lead to better performance; however, the possibility of effortlessly training with off-the-shelf methods is an advantage of the SR approaches. Code for reproducing results is available here.

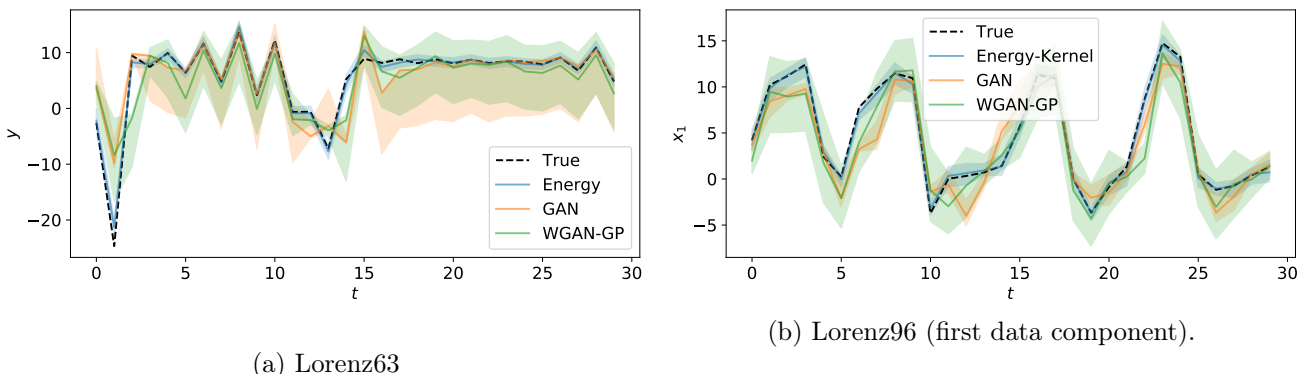


Figure 3: Results for selected methods for Lorenz63 and Lorenz96 (first data component): median forecasts (solid line) and 99% credible area (shaded area) for a part of the test set. For each t , forecasts are obtained using the previous observation window. Credible regions for GAN and WGAN-GP are broader but contain the truth less frequently.

5.1 Time-series models

We consider the Lorenz63 Lorenz [1963] and Lorenz96 Lorenz [1996] chaotic models (Appendices E.2.1 and E.3.1). The former is defined on a 3-dimensional variable, a single component of which we assume to observe. The latter contains two sets of variables; we observe only one of them, which is 8-dimensional. In both cases, we generate an observed trajectory from a long model integration, from which we take the first 60% as training set, the following 20% as validation and the remaining 20% as test.

We train the generative networks to forecast the next time-step ($l = 1$) from an observation window of size $k = 10$. We use recurrent NNs based on Gated Recurrent Units (GRU, Cho et al. [2014]; Appendices E.2.2 and E.3.2); we also tested fully connected networks but they had worse performance (see results in Appendix F.1 and F.2). For the SR methods, we select the best learning rate among 6

values according to the validation loss. For the adversarial methods, we consider instead 14 learning rates for both generator and critic; we also try two hidden dimensions for the GRU layers and four numbers of critic training steps for WGAN-GP; overall, we run 392 experiments for GAN and 1568 for WGAN-GP. As the validation loss is not a meaningful metric for adversarial approaches, we report results for 3 different configurations for GAN and WGAN-GP, maximizing either deterministic performance (1) or calibration (2), or striking the best balance between these two (3). More details are in Appendix E.2.3 and E.3.3). These experiments are run on CPU machines and take at most few minutes to complete.

In Table 1, we report performance metrics on the test set. The Kernel Score excels in deterministic forecast, getting close or outperforming the regression loss; however, all SR methods lead to combined great deterministic and probabilistic performance. On the other hand, adversarial methods are capable of good deterministic performance (1) or calibration (2) independently; but either of these two is at the expense of the other; the configuration with the best trade-off (3) is in fact much worse than the SR methods (with WGAN-GP better than GAN). In Fig. 3, we show observation and forecast for a part of the test set, for GAN and WGAN-GP in configuration 3, the Energy Score for Lorenz63 and the Energy-Kernel Score for Lorenz96. For the two SR methods, the median forecast is close to the observation and the credible region contains the true observation for most time-steps. For GAN and WGAN-GP, the match with the observation is worse and credible regions generally contain the truth less frequently albeit being wider. Additional results in Appendices F.1 and F.2.

Table 1: Performance on test set for the different methods, on the Lorenz63 and Lorenz96 models. Results with three hyperparameter configurations are reported for GAN and WGAN-GP, see text. Overall, SR methods perform well on both calibration and deterministic forecast metrics (NMRSE and R^2), while adversarial approaches are incapable of doing so.

	Lorenz63			Lorenz96		
	Cal. error ↓	NRMSE ↓	R^2 ↑	Cal. error ↓	NRMSE ↓	R^2 ↑
Regression	-	0.0079	0.9977	-	0.0198	0.9905
Energy	0.0380	0.0105	0.9960	0.0205	0.0166	0.9933
Kernel	0.0910	0.0083	0.9975	0.2196	0.0164	0.9935
Energy-Kernel	0.1000	0.0114	0.9953	0.0104	0.0173	0.9928
GAN (1)	0.4830	0.0274	0.9729	0.4644	0.0354	0.9696
GAN (2)	0.0860	0.2425	-1.1166	0.2671	0.1500	0.4537
GAN (3)	0.3590	0.0698	0.8245	0.3700	0.0763	0.8590
WGAN-GP (1)	0.4710	0.0398	0.9429	0.4134	0.0330	0.9736
WGAN-GP (2)	0.0270	0.1243	0.4440	0.0565	0.1081	0.7165
WGAN-GP (3)	0.2100	0.0914	0.6996	0.1648	0.0786	0.8502

5.2 Meteorological dataset

The WeatherBench dataset¹ for data-driven weather forecasting Rasp et al. [2020] contains hourly values of several atmospheric fields from 1979 to 2018 at different resolutions; we choose here a resolution of 5.625° over both longitude and latitude, corresponding to a 32×64 grid. We consider a single observation per day (12:00 UTC) and the 500 hPa geopotential (Z500) variable. We forecast with a lead of 3 days ($l = 3$) from a single observation ($k = 1$). We use the years from 1979 to 2006 as training set, 2007 to 2016 as validation test and 2017 to 2018 as test set.

In addition to the Energy, Kernel and Energy-Kernel Scores, we test the spatial SRs introduced in Sec 3.2. Specifically, we consider the Variogram Score with weights w inversely proportional to the distance on the globe (Appendix E.4.1) and sum it to the Energy (*Energy-Variogram*) or to the Kernel (*Kernel-Variogram*) Scores. We also consider the Patched Energy Score with patch size 8 and

¹Released under MIT license, see here.

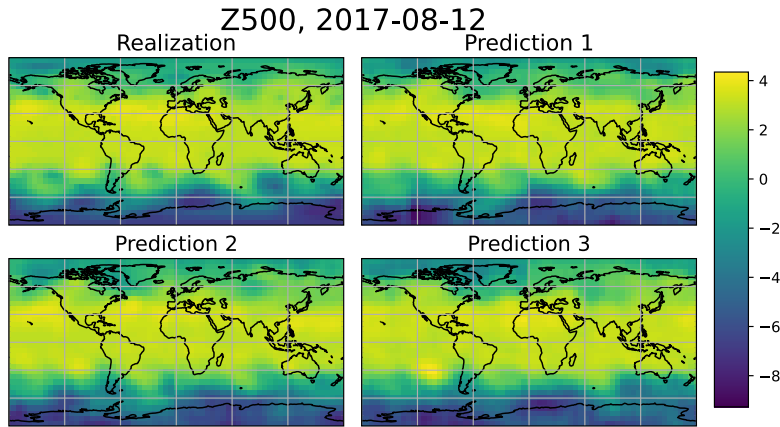


Figure 4: Realization and example of predictions obtained with the patched Energy Score (patch size 16) for a specific date in the test set for the WeatherBench dataset. The predictions capture the main features but are slightly different from each other.

16; to ensure the score is strictly proper, we add the overall Energy Score (summation weights in Appendix E.4.2). We also consider patched regression loss.

We employ a U-NET architecture [Olaf et al., 2015] for the generative network and a PatchGAN discriminator [Isola et al., 2017] for the critic (Appendix E.4.3). For the SR methods, we select the best learning rate among 6 values according to the validation loss; for the adversarial ones, we consider instead 7 values for both generator and critic, resulting in 49 experiments. We then pick the setups optimizing deterministic or calibration performance. For WGAN-GP, a single configuration optimizes both; for GAN, that did not happen. As for the time-series models, we report therefore results for setups maximizing either deterministic performance (1) or calibration (2), or striking the best balance between these two (3). All trainings are run on a single Tesla V100 GPU; computing times are reported in Appendix F.3.3.

Table 2 contains performance metrics on the test set. The Patched Energy Scores yield the best performance, with deterministic skill only slightly worse than the regression loss. Energy-Variogram and Kernel-Variogram are worse than the standard Energy and Kernel ones; moreover, as the Variogram Score requires quadratic memory in data size, a smaller batch size was needed for training (Appendix E.4.4). All GAN configurations have bad performance, while WGAN-GP is acceptable, but still worse than the SR methods. In Fig. 4 we show observation and three different predictions obtained with the Patched Energy Score for a date in the test set. More results in Appendix F.3.

6 Conclusions

We gave an overview of a formulation for generative networks training based on Scoring Rules and compared it to the standard one based on divergences, to which the former is complementary. The advantages of the Scoring Rule formulation are: (i) it provides a principled objective for probabilistic forecasting; (ii) it yields adversarial-free training, with which better uncertainty quantification is possible, as we show empirically; (iii) it enables leveraging the literature on SRs to define objectives for spatio-temporal datasets. These findings corroborate those reported in Chapter 4 of the present thesis for Bayesian Likelihood-Free Inference tasks, making Scoring Rules minimization an appealing method to train generative networks, particularly when uncertainty quantification in the approximate distribution is critical.

The resulting training method involves choosing the scoring rule (including scoring-rule-specific choices, such as the kernel, the patch size and the variogram weights); we argue however that this is an easier problem than designing a discriminator network, as it is required for adversarial approaches. Moreover, in contrast to adversarial methods, the use of a single neural network makes tuning the hyperparameter for the optimizer scheme drastically easier.

We highlight the following limitations of our work: first, our Theorem 3.3 relies on assumptions

Table 2: Performance on WeatherBench test set for different methods. Results with three hyperparameter configurations are reported for GAN, see text. SR methods perform well on both calibration and deterministic forecast metrics (NRMSE and R^2). WGAN-GP is worse and GAN is drastically worse.

	Cal. error ↓	NRMSE ↓	R^2 ↑
Regression	-	0.1162	0.5300
Patched Regression, 8	-	0.1147	0.5459
Patched Regression, 16	-	0.1144	0.5509
Energy	0.0863	0.1208	0.4968
Kernel	0.0797	0.1200	0.5097
Energy-Kernel	0.0794	0.1194	0.5150
Energy-Variogram	0.0899	0.1192	0.5177
Kernel-Variogram	0.1704	0.1203	0.5050
Patched Energy, 8	0.0550	0.1189	0.5217
Patched Energy, 16	0.0690	0.1186	0.5248
GAN (1)	0.4845	0.1573	0.1418
GAN (2)	0.3130	0.2487	-2.7970
GAN (3)	0.3625	0.1693	-0.0117
WGAN-GP	0.1009	0.1302	0.4340

which are hard to verify; however, we believe similar consistency properties hold provided the temporal process satisfies some generic stationarity and memory-less properties. Secondly, we do not experiment forecasting multiple time-steps at once as we preferred focusing on single time-step forecast tasks for the matter of analytical simplicity while developing our framework. Doing so would be however a useful extension of our work; in practice, SRs assessing temporal coherence analogous to what done with temporal discriminators in Ravuri et al. [2021] in the adversarial setting could be developed. Finally, training the generative networks with a fixed scoring rule, instead of a learned discriminator, may make the spatial structure of single samples from the generative distribution less realistic than those obtained by an adversarial method. While results on the considered datasets did not seem to present this issue at visual inspection, combining adversarial and SR training could address this issue; to get the best of both methods, this would need to preserve the smooth dynamics of SR training. Measuring progress on this front is however hindered by the lack of a good “realism” metric², which must be subject of future investigation.

Acknowledgment

LP is supported by the EPSRC and MRC through the OxWaSP CDT programme (EP/L016710/1), which also funds the computational resources used to perform this work. RD is funded by EPSRC (grant nos. EP/V025899/1, EP/T017112/1) and NERC (grant no. NE/T00973X/1). PD gratefully acknowledges funding from the Royal Society for his University Research Fellowship, as well as from the ESiWACE Horizon 2020 project (#823988) and the MAELSTROM EuroHPC Joint Undertaking project (#955513). We thank Geoff Nicholls, Christian Robert, Peter Watson, Matthew Chantry, Mihai Alexe and Eugenio Clerico for valuable feedback and suggestions.

References

M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

²Our evaluation metrics are univariate and thus not capturing the spatial structure of the samples.

- S. Arora, R. Ge, Y. Liang, T. Ma, and Y. Zhang. Generalization and equilibrium in generative adversarial nets (GANs). In *International Conference on Machine Learning*, pages 224–232. PMLR, 2017.
- S. Arora, A. Risteski, and Y. Zhang. Do GANs learn the distribution? some theory and empirics. In *International Conference on Learning Representations*, 2018. URL <https://openreview.net/forum?id=BJehNfW0->.
- M. G. Bellemare, I. Danihelka, W. Dabney, S. Mohamed, B. Lakshminarayanan, S. Hoyer, and R. Munos. The Cramer distance as a solution to biased Wasserstein gradients. *arXiv preprint arXiv:1705.10743*, 2017.
- A. Bihlo. A generative adversarial network approach to (ensemble) weather prediction. *Neural Networks*, 139:1–16, 2021.
- M. Bińkowski, D. J. Sutherland, M. Arbel, and A. Gretton. Demystifying MMD GANs. In *International Conference on Learning Representations*, 2018.
- D. Bouchacourt, P. K. Mudigonda, and S. Nowozin. DISCO nets: DISsimilarity COefficient networks. *Advances in Neural Information Processing Systems*, 29:352–360, 2016.
- R. C. Bradley. Basic properties of strong mixing conditions. a survey and some open questions. *Probability surveys*, 2:107–144, 2005.
- B.-E. Chérif-Abdellatif and P. Alquier. MMD-Bayes: Robust Bayesian estimation via maximum mean discrepancy. In *Symposium on Advances in Approximate Bayesian Inference*, pages 1–21. PMLR, 2020.
- K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar, Oct. 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-4012. URL <https://aclanthology.org/W14-4012>.
- M. C. Clare, O. Jamil, and C. J. Morcrette. Combining distribution-based neural networks to predict weather probabilities. *Quarterly Journal of the Royal Meteorological Society*, 147(741):4337–4357, 2021.
- A. P. Dawid. Present position and potential developments: Some personal views statistical theory the prequential approach. *Journal of the Royal Statistical Society: Series A (General)*, 147(2):278–290, 1984.
- A. P. Dawid and M. Musio. Estimation of spatial processes using local scoring rules. *ASTA Advances in Statistical Analysis*, 97(2):173–179, 2013.
- A. P. Dawid and M. Musio. Theory and applications of proper scoring rules. *Metron*, 72(2):169–183, 2014.
- A. P. Dawid and M. Musio. Bayesian model selection based on proper scoring rules. *Bayesian analysis*, 10(2):479–499, 2015.
- A. P. Dawid, M. Musio, and L. Ventura. Minimum scoring rule inference. *Scandinavian Journal of Statistics*, 43(1):123–138, 2016.
- I. Domowitz and H. White. Misspecified models with dependent observations. *Journal of Econometrics*, 20(1):35–58, 1982. ISSN 0304-4076. doi: [https://doi.org/10.1016/0304-4076\(82\)90102-6](https://doi.org/10.1016/0304-4076(82)90102-6). URL <https://www.sciencedirect.com/science/article/pii/0304407682901026>.
- P. D. Dueben and P. Bauer. Challenges and design choices for global weather and climate models based on machine learning. *Geoscientific Model Development*, 11(10):3999–4009, 2018.

- G. K. Dziugaite, D. M. Roy, and Z. Ghahramani. Training generative neural networks via maximum mean discrepancy optimization. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, pages 258–267, 2015.
- D. J. Gagne, H. M. Christensen, A. C. Subramanian, and A. H. Monahan. Machine learning for stochastic parameterization: Generative adversarial networks in the Lorenz’96 model. *Journal of Advances in Modeling Earth Systems*, 12(3):e2019MS001896, 2020.
- T. Gneiting and M. Katzfuss. Probabilistic forecasting. *Annual Review of Statistics and Its Application*, 1:125–151, 2014.
- T. Gneiting and A. E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.
- I. Goodfellow. Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*, 2016.
- I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.
- A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola. A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773, 2012.
- A. Gritsenko, T. Salimans, R. van den Berg, J. Snoek, and N. Kalchbrenner. A spectral energy distance for parallel speech synthesis. *Advances in Neural Information Processing Systems*, 33:13062–13072, 2020.
- I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved training of Wasserstein GANs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 5769–5779, 2017.
- A. Harakeh and S. L. Waslander. Estimating and evaluating regression predictive uncertainty in deep object detectors. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=YLewtvnvKgR7>.
- H. Hersbach. Decomposition of the continuous ranked probability score for ensemble prediction systems. *Weather and Forecasting*, 15(5):559–570, 2000.
- P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.
- D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In Y. Bengio and Y. LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL <http://arxiv.org/abs/1312.6114>.
- A. Koochali, A. Dengel, and S. Ahmed. If you like it, GAN it—probabilistic multivariate times series forecast with GAN. *Engineering Proceedings*, 5(1):40, Jul 2021. ISSN 2673-4591. doi: 10.3390/engproc2021005040. URL <http://dx.doi.org/10.3390/engproc2021005040>.
- Y.-H. Kwon and M.-G. Park. Predicting future frames using retrospective cycle gan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1811–1820, 2019.
- M. Leutbecher and T. N. Palmer. Ensemble forecasting. *Journal of computational physics*, 227(7): 3515–3539, 2008.
- C.-L. Li, W.-C. Chang, Y. Cheng, Y. Yang, and B. Póczos. MMD GAN: Towards deeper understanding of moment matching network. In *NIPS*, 2017.

- Y. Li, K. Swersky, and R. Zemel. Generative moment matching networks. In *International Conference on Machine Learning*, pages 1718–1727. PMLR, 2015.
- E. N. Lorenz. Deterministic nonperiodic flow. *Journal of atmospheric sciences*, 20(2):130–141, 1963.
- E. N. Lorenz. Predictability: A problem partly solved. In *Proc. Seminar on predictability*, volume 1, 1996.
- M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.
- H. D. Nguyen, J. Arbel, H. Lü, and F. Forbes. Approximate Bayesian computation via the energy statistic. *IEEE Access*, 8:131683–131698, 2020.
- S. Nowozin, B. Cseke, and R. Tomioka. f-GAN: Training generative neural samplers using variational divergence minimization. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 271–279, 2016.
- R. Olaf, F. Philipp, and B. Thomas. U-Net: Convolutional networks for biomedical image segmentation, 2015.
- T. Palmer. Towards the probabilistic Earth-system simulator: a vision for the future of climate and weather prediction. *Quarterly Journal of the Royal Meteorological Society*, 138(665):841–861, 2012.
- M. Park, W. Jitkrittum, and D. Sejdinovic. K2-ABC: Approximate Bayesian computation with kernel embeddings. In *Artificial Intelligence and Statistics*, 2016.
- A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. PyTorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d’Alché Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.
- B. M. Pötscher and I. R. Prucha. A uniform law of large numbers for dependent and heterogeneous data processes. *Econometrica: Journal of the Econometric Society*, pages 675–683, 1989.
- S. T. Radev, U. K. Mertens, A. Voss, L. Ardizzone, and U. Köthe. BayesFlow: Learning complex stochastic models with invertible neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.
- S. Rasp, P. D. Dueben, S. Scher, J. A. Weyn, S. Mouatadid, and N. Thuerey. WeatherBench: a benchmark data set for data-driven weather forecasting. *Journal of Advances in Modeling Earth Systems*, 12(11):e2020MS002203, 2020.
- S. Ravuri, K. Lenc, M. Willson, D. Kangin, R. Lam, P. Mirowski, M. Fitzsimons, M. Athanassiadou, S. Kashem, S. Madge, et al. Skilful precipitation nowcasting using deep generative models of radar. *Nature*, 597(7878):672–677, 2021.
- E. Richardson and Y. Weiss. On GANs and GMMs. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 5852–5863, 2018.
- M. L. Rizzo and G. J. Székely. Energy distance. *Wiley interdisciplinary reviews: Computational statistics*, 8(1):27–38, 2016.
- T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training GANs. *Advances in neural information processing systems*, 29, 2016.
- S. Scher. Toward data-driven weather and climate forecasting: Approximating a simple general circulation model with deep learning. *Geophysical Research Letters*, 45(22):12–616, 2018.

- S. Scher and G. Messori. Weather and climate forecasting with neural networks: using general circulation models (GCMs) with different complexity as a study ground. *Geoscientific Model Development*, 12(7):2797–2809, 2019.
- S. Scher and G. Messori. Ensemble methods for neural network-based weather forecasts. *Journal of Advances in Modeling Earth Systems*, 13(2), 2021.
- M. Scheuerer and T. M. Hamill. Variogram-based proper scoring rules for probabilistic forecasts of multivariate quantities. *Monthly Weather Review*, 143(4):1321–1334, 2015.
- K. Skouras. *On the optimal performance of forecasting systems: The prequential approach*. University of London, University College London (United Kingdom), 1998.
- J. A. Weyn, D. R. Durran, and R. Caruana. Can machines learn to predict weather? Using deep learning to predict gridded 500-hPa geopotential height from historical weather data. *Journal of Advances in Modeling Earth Systems*, 11(8):2680–2693, 2019.

Appendix

A Proofs of theoretical results

A.1 Propriety of the prequential SR

In this Section, let P^* denote the data generating distribution for $(\mathbf{Y}_1, \mathbf{Y}_2, \dots, \mathbf{Y}_t, \dots) = (\mathbf{Y}_t)_t$, and let P denote a generic distribution assigned to $(\mathbf{Y}_t)_t$. From the distribution on the full sequence P , conditional and marginals can be obtained, and denoted as follows: $P_{t+1}(\cdot|\mathbf{y}_{1:t})$ denotes the conditional distribution for \mathbf{Y}_{t+1} given $\mathbf{y}_{1:t}$, and $P_{1:t}$ the (marginal) distribution for $(\mathbf{Y}_1, \mathbf{Y}_2, \dots, \mathbf{Y}_t)$. Similar notation will be used for the conditional and marginals induced by P^* .

A.1.1 Generic 1-step ahead prequential SR

We first consider a simplified case in which we can access the marginal for \mathbf{Y}_1 and all subsequent conditionals from P . Given $\mathbf{y}_{1:t}$, we use the distribution P to construct a forecast distribution for \mathbf{Y}_{t+1} , namely $P_{t+1}(\cdot|\mathbf{y}_{1:t})$; we penalize the forecast, against the verifying observation \mathbf{y}_{t+1} , via a SR S :

$$S(P_{t+1}(\cdot|\mathbf{y}_{1:t}), \mathbf{y}_{t+1}).$$

From the above, we construct the *prequential* SR for the forecast $P_{1:T}$ as follows:

$$S_T(P_{1:T}, \mathbf{y}_{1:T}) = \frac{1}{T} \left[\sum_{t=1}^{T-1} S(P_{t+1}(\cdot|\mathbf{y}_{1:t}), \mathbf{y}_{t+1}) + S(P_1, \mathbf{y}_1) \right]; \quad (10)$$

the above assumes that at each time instant we obtain a probabilistic forecast $P_{t+1}(\cdot|\mathbf{y}_{1:t})$ from the distribution P and we verify it against the next observed element of the sequence \mathbf{y}_{t+1} . Additionally, at the first time step, we have not yet received any observation, so our forecast P_1 is unconditional. Also, let us define the expected prequential score as:

$$S_T(P_{1:T}, P_{1:T}^*) := \mathbb{E}_{\mathbf{Y}_{1:T} \sim P_{1:T}^*} S_T(P_{1:T}, \mathbf{Y}_{1:T}),$$

Theorem A.1. *If the scoring rule S is proper, then the prequential score S_T in Eq. (10) is proper for distributions over \mathcal{Y}^T , i.e.:*

$$S_T(P_{1:T}^*, P_{1:T}^*) \leq S_T(P_{1:T}, P_{1:T}^*).$$

Similarly, if S is strictly proper, the prequential score S_T is strictly proper, i.e. the equality only holds if $P_{1:T} = P_{1:T}^$.*

Proof. First, notice that $P_{1:T}$ is fully determined by the marginal P_1 and by the conditionals $P_{t+1}(\cdot|\mathbf{y}_{1:t})$, $1 \leq t \leq T-1$. In fact, if $P_{1:T}$ has densities, you can write:

$$p_{1:T}(\mathbf{y}_{1:T}) = p_1(\mathbf{y}_1) p_2(\mathbf{y}_2|\mathbf{y}_1) p_3(\mathbf{y}_3|\mathbf{y}_{1:2}) \dots p_{T-1}(\mathbf{y}_{T-1}|\mathbf{y}_{1:T-2}) p_T(\mathbf{y}_T|\mathbf{y}_{1:T-1}).$$

By definition of proper SR, we have that:

$$\mathbb{E}_{\mathbf{Y}_{t+1} \sim P_{t+1}^*(\cdot|\mathbf{y}_{1:t})} S(P_{t+1}^*(\cdot|\mathbf{y}_{1:t}), \mathbf{Y}_{t+1}) \leq \mathbb{E}_{\mathbf{Y}_{t+1} \sim P_{t+1}(\cdot|\mathbf{y}_{1:t})} S(P_{t+1}(\cdot|\mathbf{y}_{1:t}), \mathbf{Y}_{t+1})$$

for any conditional distribution $P_{t+1}(\cdot|\mathbf{y}_{1:t})$ and for any values $\mathbf{y}_{1:t}$.

Similarly, it holds:

$$\mathbb{E}_{\mathbf{Y}_1 \sim P_1^*} S(P_1^*, \mathbf{Y}_1) \leq \mathbb{E}_{\mathbf{Y}_1 \sim P_1} S(P_1, \mathbf{Y}_1), \quad (11)$$

for any distribution P_1 .

For the expected prequential SR, it holds that:

$$\begin{aligned} S_T(P_{1:T}, P_{1:T}^*) &= \mathbb{E}_{\mathbf{Y}_{1:T} \sim P_{1:T}^*} S_T(P_{1:T}, \mathbf{Y}_{1:T}) \\ &= \frac{1}{T} \left[\sum_{t=1}^{T-1} \mathbb{E}_{\mathbf{Y}_{1:T} \sim P_{1:T}^*} S(P_{t+1}(\cdot|\mathbf{Y}_{1:t}), \mathbf{Y}_{t+1}) + \mathbb{E}_{\mathbf{Y}_{1:T} \sim P_{1:T}^*} S(P_1, \mathbf{Y}_1) \right] \\ &= \frac{1}{T} \left[\sum_{t=1}^{T-1} \mathbb{E}_{\mathbf{Y}_{1:t+1} \sim P_{1:t+1}^*} S(P_{t+1}(\cdot|\mathbf{Y}_{1:t}), \mathbf{Y}_{t+1}) + \mathbb{E}_{\mathbf{Y}_1 \sim P_1^*} S(P_1, \mathbf{Y}_1) \right]; \end{aligned}$$

but now:

$$\begin{aligned} \mathbb{E}_{\mathbf{Y}_{1:t+1} \sim P_{1:t+1}^*} S(P_{t+1}(\cdot | \mathbf{Y}_{1:t}), \mathbf{Y}_{t+1}) &= \mathbb{E}_{\mathbf{Y}_{1:t} \sim P_{1:t}^*} \left[\mathbb{E}_{\mathbf{Y}_{t+1} \sim P_{t+1}^*(\cdot | \mathbf{Y}_{1:t})} S(P_{t+1}(\cdot | \mathbf{Y}_{1:t}), \mathbf{Y}_{t+1}) \right] \\ &\geq \mathbb{E}_{\mathbf{Y}_{1:t} \sim P_{1:t}^*} \left[\mathbb{E}_{\mathbf{Y}_{t+1} \sim P_{t+1}^*(\cdot | \mathbf{Y}_{1:t})} S(P_{t+1}^*(\cdot | \mathbf{Y}_{1:t}), \mathbf{Y}_{t+1}) \right], \end{aligned} \quad (12)$$

so that:

$$\begin{aligned} S_T(P_{1:T}, P_{1:T}^*) &\geq \frac{1}{T} \left[\sum_{t=1}^{T-1} \mathbb{E}_{\mathbf{Y}_{1:t+1} \sim P_{1:t+1}^*} S(P_{t+1}^*(\cdot | \mathbf{Y}_{1:t}), \mathbf{Y}_{t+1}) + \mathbb{E}_{\mathbf{Y}_1 \sim P_1^*} S(P_1^*, \mathbf{Y}_1) \right] \\ &= \frac{1}{T} \left[\sum_{t=1}^{T-1} \mathbb{E}_{\mathbf{Y}_{1:T} \sim P_{1:T}^*} S(P_{t+1}^*(\cdot | \mathbf{Y}_{1:t}), \mathbf{Y}_{t+1}) + \mathbb{E}_{\mathbf{Y}_{1:T} \sim P_{1:T}^*} S(P_1^*, \mathbf{Y}_1) \right] \\ &= S_T(P_{1:T}^*, P_{1:T}^*), \end{aligned} \quad (13)$$

which proves that S_T is proper.

To show that S_T is strictly proper if S is, notice that the equality in Eq. (13) holds if and only if the equalities in Eq. (11) and (12) are verified for all $1 \leq t \leq T$; if S is strictly proper, however, that requires that $P_1 = P_1^*$ and $P_{t+1}(\cdot | \mathbf{y}_{1:t}) = P_{t+1}^*(\cdot | \mathbf{y}_{1:t})$ for $1 \leq t \leq T-1$, which implies that $P_{1:T} = P_{1:T}^*$ due to distributions on $\mathbf{Y}_{1:T}$ being determined by the marginal for \mathbf{Y}_1 and the conditional on $\mathbf{Y}_{t+1} | \mathbf{Y}_{1:t}$. \square

A.1.2 l -steps ahead prequential SR (Theorem 3.2)

We now go back to the specific setting considered in the main body of the paper. By discarding the model parameter ϕ in the notation for simplicity, the generative network induces conditional distributions $P_{t+l}(\cdot | \mathbf{y}_{1:t})$ for \mathbf{Y}_{t+l} which only depend on the last k observations, i.e. $P_{t+l}(\cdot | \mathbf{y}_{1:t}) = P_{t+l}(\cdot | \mathbf{y}_{t-k+1:t})$. Therefore, the joint distribution for $\mathbf{Y}_{k+l:T}$ induced by the generative network satisfies the following property:

Definition A.2. A probability distribution $P_{1:T}$ is k -Markovian with lag l if it can be decomposed as follows, assuming it has density $p_{1:T}$ with respect to some base measure:

$$p_{1:T}(\mathbf{y}_{1:T}) = p_{1:k+l-1}(\mathbf{y}_{1:k+l-1}) \prod_{t=k}^{T-l} p_{t+l}(\mathbf{y}_{t+l} | \mathbf{y}_{t-k+1:t}).$$

Setting $l = 1$ recovers the standard definition of k -Markovian models.

Notice also that the set of distributions which are k -Markovian with lag l is a subset of $(k+l-1)$ -Markovian distributions, for which in fact:

$$\begin{aligned} p_{1:T}(\mathbf{y}_{1:T}) &= p_{1:k+l-1}(\mathbf{y}_{1:k+l-1}) \prod_{t=k+l}^T p_t(\mathbf{y}_t | \mathbf{y}_{t-k-l+1:t-1}) \\ &= p_{1:k+l-1}(\mathbf{y}_{1:k+l-1}) \prod_{t=k}^{T-l} p_{t+l}(\mathbf{y}_{t+l} | \mathbf{y}_{t-k+1:t+l-1}); \end{aligned}$$

the additional assumption in Definition A.2 with respect to $(k+l-1)$ -Markovian is that the conditional distribution for \mathbf{Y}_t is *not* influenced by the last $l-1$ elements.

In our setting, we can only access $P_{t+l}(\cdot | \mathbf{y}_{t-k+1:t})$ for $k \leq t \leq T-l$; the marginals $P_{1:k+l-1}$ are not available. Therefore, we consider the following quantity:

$$S_T^{k,l}(P_{k+l:T}(\cdot | \mathbf{y}_{1:k+l-1}), \mathbf{y}_{k+l:T}) := \frac{1}{T-l-k+1} \sum_{t=k}^{T-l} S(P_{t+l}(\cdot | \mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l}); \quad (14)$$

in contrast to Eq. (8) in the main text, we make explicit the dependence on k and l in the notation for $S_T^{k,l}$ and introduce a scaling constant for simplicity, which however does not impact the following

arguments. The notation in Eq. (14) only makes sense if P is a $(k + l - 1)$ -Markovian distribution, as otherwise $\mathbf{y}_{1:k+l-1}$ would also appear explicitly in the conditioning of P_{t+l} on the right hand-side. The notation therefore makes sense for P obtained from the generative network, as that is k -Markovian with lag l which, as mentioned above, is a specific case of $(k + l - 1)$ -Markovian.

As mentioned in the main text, $S_T^{k,l}$ is the prequential score and is a SR for distributions over $\mathbf{Y}_{k+l:T}|\mathbf{y}_{1:k+l-1}$ which are $(k + l - 1)$ -Markovian.

From Eq. (14), we can define the expected SR as:

$$S_T^{k,l}(P_{k+l:T}(\cdot|\mathbf{y}_{1:k+l-1}), P_{k+l:T}^*(\cdot|\mathbf{y}_{1:k+l-1})) := \mathbb{E}_{\mathbf{Y}_{k+l:T} \sim P_{k+l:T}^*(\cdot|\mathbf{y}_{1:k+l-1})} S_T^{k,l}(P_{k+l:T}(\cdot|\mathbf{y}_{1:k+l-1}), \mathbf{Y}_{k+l:T}).$$

For the scoring rule defined in Eq. (14), the following Theorem holds, which we state in more generality with respect to Theorem 3.2 in the main text:

Theorem A.3. *If the scoring rule S is proper, then, for all choices of $\mathbf{y}_{1:k+l-1}$, the prequential score $S_T^{k,l}$ in Eq. (14) is proper for distributions on $\mathbf{Y}_{k+l:T}|\mathbf{y}_{1:k+l-1}$ which are $(k + l - 1)$ -Markovian; namely, the following inequality holds:*

$$S_T^{k,l}(P_{k+l:T}^*(\cdot|\mathbf{y}_{1:k+l-1}), P_{k+l:T}^*(\cdot|\mathbf{y}_{1:k+l-1})) \leq S_T^{k,l}(P_{k+l:T}(\cdot|\mathbf{y}_{1:k+l-1}), P_{k+l:T}^*(\cdot|\mathbf{y}_{1:k+l-1})), \quad (15)$$

where $P_{1:T}$ and $P_{1:T}^*$ are $(k + l - 1)$ -Markovian.

If additionally S is strictly proper, then, for all choices of $\mathbf{y}_{1:k+l-1}$, $S_T^{k,l}$ is proper for distributions on $\mathbf{Y}_{k+l:T}|\mathbf{y}_{1:k+l-1}$ which are k -Markovian with lag l , i.e. the equality in Eq. (15) only holds if $P_{k+l:T}(\cdot|\mathbf{y}_{1:k+l-1}) = P_{k+l:T}^*(\cdot|\mathbf{y}_{1:k+l-1})$, where $P_{1:T}$ and $P_{1:T}^*$ are k -Markovian with lag l .

The prequential score S_T^{k+l} is non-strictly proper for distributions that are $(k + l - 1)$ -Markovian but not k -Markovian with lag l . In fact, it builds forecasts from $P_{k+l:T}^*(\cdot|\mathbf{y}_{1:k+l-1})$ with lead of l timesteps, meaning that the information included in observations $\mathbf{y}_{t+1:t+l-1}$ is not used in formulating the forecast for \mathbf{Y}_{t+l} . It is therefore unable to distinguish between different distributions for $\mathbf{Y}_{k+l:T}|\mathbf{y}_{1:k+l-1}$ which have the same conditionals at lead l , but for which the conditionals change if one takes into account $\mathbf{y}_{t+1:t+l-1}$ in forecasting \mathbf{Y}_{t+l} . Therefore, you need to restrict the class of distributions to those in which the value $\mathbf{y}_{t+1:t+l-1}$ does not impact the distribution for \mathbf{Y}_{t+l} in order to get strict propriety.

We now prove the Theorem.

Proof. The proof steps follow those of Theorem A.1.

By definition of proper SR, we have that, for all $t \geq k$:

$$\mathbb{E}_{\mathbf{Y}_{t+l} \sim P_{t+l}^*(\cdot|\mathbf{y}_{t-k+1:t})} S(P_{t+l}^*(\cdot|\mathbf{y}_{t-k+1:t}), \mathbf{Y}_{t+l}) \leq \mathbb{E}_{\mathbf{Y}_{t+l} \sim P_{t+l}^*(\cdot|\mathbf{y}_{t-k+1:t})} S(P_{t+l}(\cdot|\mathbf{y}_{t-k+1:t}), \mathbf{Y}_{t+l}) \quad (16)$$

for any conditional distribution $P_{t+l}(\cdot|\mathbf{y}_{t-k+1:t})$ and for any values $\mathbf{y}_{t-k+1:t}$.

For the expected prequential SR, it holds that:

$$\begin{aligned} & S_T^{k,l}(P_{k+l:T}^*(\cdot|\mathbf{y}_{1:k+l-1}), P_{k+l:T}^*(\cdot|\mathbf{y}_{1:k+l-1})) \\ &= \mathbb{E}_{\mathbf{Y}_{k+l:T} \sim P_{k+l:T}^*(\cdot|\mathbf{y}_{1:k+l-1})} S_T^{k,l}(P_{k+l:T}^*(\cdot|\mathbf{y}_{1:k+l-1}), \mathbf{Y}_{k+l:T}) \\ &= \mathbb{E}_{\mathbf{Y}_{1:T} \sim P_{1:T}^*(\cdot|\mathbf{y}_{1:k+l-1})} S_T^{k,l}(P_{k+l:T}^*(\cdot|\mathbf{Y}_{1:k+l-1}), \mathbf{Y}_{k+l:T}) \\ &= \frac{1}{T-l-k+1} \sum_{t=k}^{T-l} \mathbb{E}_{\mathbf{Y}_{1:T} \sim P_{1:T}^*(\cdot|\mathbf{y}_{1:k+l-1})} S(P_{t+l}^*(\cdot|\mathbf{Y}_{t-k+1:t}), \mathbf{Y}_{t+l}) \\ &= \frac{1}{T-l-k+1} \sum_{t=k}^{T-l} \mathbb{E}_{\mathbf{Y}_{1:t+l} \sim P_{1:t+l}^*(\cdot|\mathbf{y}_{1:k+l-1})} S(P_{t+l}^*(\cdot|\mathbf{Y}_{t-k+1:t}), \mathbf{Y}_{t+l}); \end{aligned}$$

the second equality in the Equation above is trivial but we use it to simplify notation in the following. Now:

$$\begin{aligned}
& \mathbb{E}_{\mathbf{Y}_{1:t+l} \sim P_{1:t+l}^*(\cdot | \mathbf{y}_{1:k+l-1})} S(P_{t+l}^*(\cdot | \mathbf{Y}_{t-k+1:t}), \mathbf{Y}_{t+l}) \\
&= \mathbb{E}_{\mathbf{Y}_{t-k+1:t} \sim P_{t-k+1:t}^*(\cdot | \mathbf{y}_{1:k+l-1})} \left[\mathbb{E}_{\mathbf{Y}_{t+l} \sim P_{t+l}^*(\cdot | \mathbf{Y}_{t-k+1:t}, \mathbf{y}_{1:k+l-1})} S(P_{t+l}^*(\cdot | \mathbf{Y}_{t-k+1:t}), \mathbf{Y}_{t+l}) \right] \\
&= \mathbb{E}_{\mathbf{Y}_{t-k+1:t} \sim P_{t-k+1:t}^*(\cdot | \mathbf{y}_{1:k+l-1})} \left[\mathbb{E}_{\mathbf{Y}_{t+l} \sim P_{t+l}^*(\cdot | \mathbf{Y}_{t-k+1:t})} S(P_{t+l}^*(\cdot | \mathbf{Y}_{t-k+1:t}), \mathbf{Y}_{t+l}) \right] \quad (17) \\
&\leq \mathbb{E}_{\mathbf{Y}_{t-k+1:t} \sim P_{t-k+1:t}^*(\cdot | \mathbf{y}_{1:k+l-1})} \left[\mathbb{E}_{\mathbf{Y}_{t+l} \sim P_{t+l}^*(\cdot | \mathbf{Y}_{t-k+1:t})} S(P_{t+l}(\cdot | \mathbf{Y}_{t-k+1:t}), \mathbf{Y}_{t+l}) \right] \\
&= \mathbb{E}_{\mathbf{Y}_{1:t+l} \sim P_{1:t+l}^*(\cdot | \mathbf{y}_{1:k+l-1})} S(P_{t+l}(\cdot | \mathbf{Y}_{t-k+1:t}), \mathbf{Y}_{t+l});
\end{aligned}$$

in the first equality above, we have marginalized over all components of $\mathbf{Y}_{1:t+l}$ which do not appear in the expected quantity and we have used the definition of conditional probability together with the tower property of expectations. In the second equality, we have exploited the $(k+l-1)$ -Markov property³ of P^* which ensures that the distribution for \mathbf{Y}_{t+l} does not depend on $\mathbf{Y}_{1:t-k}$. The inequality holds for any conditional distribution $P_{t+l}(\cdot | \mathbf{y}_{t-k+1:t})$ and for any values $\mathbf{y}_{t-k+1:t}$ thanks to Eq. (16). Finally, the last equality is obtained via the reverse of the argument used for the first one.

Now, we can write:

$$\begin{aligned}
& S_T^{k,l}(P_{k+l:T}^*(\cdot | \mathbf{y}_{1:k+l-1}), P_{k+l:T}^*(\cdot | \mathbf{y}_{1:k+l-1})) \\
&\leq \frac{1}{T-l-k+1} \sum_{t=k}^{T-l} \mathbb{E}_{\mathbf{Y}_{1:t+l} \sim P_{1:t+l}^*(\cdot | \mathbf{y}_{1:k+l-1})} S(P_{t+l}(\cdot | \mathbf{Y}_{t-k+1:t}), \mathbf{Y}_{t+l}) \\
&= \frac{1}{T-l-k+1} \sum_{t=k}^{T-l} \mathbb{E}_{\mathbf{Y}_{1:T} \sim P_{1:T}^*(\cdot | \mathbf{y}_{1:k+l-1})} S(P_{t+l}(\cdot | \mathbf{Y}_{t-k+1:t}), \mathbf{Y}_{t+l}) \quad (18) \\
&= \frac{1}{T-l-k+1} \sum_{t=k}^{T-l} \mathbb{E}_{\mathbf{Y}_{k+l:T} \sim P_{k+l:T}^*(\cdot | \mathbf{y}_{1:k+l-1})} S(P_{t+l}(\cdot | \mathbf{Y}_{t-k+1:t}), \mathbf{Y}_{t+l}) \\
&= S_T^{k,l}(P_{k+l:T}(\cdot | \mathbf{y}_{1:k+l-1}), P_{k+l:T}^*(\cdot | \mathbf{y}_{1:k+l-1})),
\end{aligned}$$

which proves that $S_T^{k,l}$ is proper for distributions over $\mathbf{Y}_{k+l:T} | \mathbf{y}_{1:k+l-1}$ which are $(k+l)$ -Markov.

Now, consider $P_{1:T}$ and $P_{1:T}^*$ to be k -Markovian with lag l . The equality in Eq. (18) holds if and only if the equality in Eq. (17) is verified for all $k \leq t \leq T-l$; if S is strictly proper, however, that requires that $P_{t+l}(\cdot | \mathbf{y}_{t-k+1:t}) = P_{t+l}^*(\cdot | \mathbf{y}_{t-k+1:t})$ for $k \leq t \leq T-l$, which implies that $P_{k+l:T}(\cdot | \mathbf{y}_{1:k+l-1}) = P_{k+l:T}^*(\cdot | \mathbf{y}_{1:k+l-1})$ due to the k -Markov with lag l property. \square

A.2 Proof and precise statement of the consistency result (Theorem 3.3)

We follow here the notation introduced at the start of Appendix A.1. Specifically, P^* denotes the data generating distribution for $(\mathbf{Y}_1, \mathbf{Y}_2, \dots, \mathbf{Y}_t, \dots) = (\mathbf{Y}_t)_t$.

We consider a model class parametrized by a set of parameters ϕ . For such models, we assume the conditional distributions $P_{t+l}^\phi(\cdot | \mathbf{y}_{1:t})$ for \mathbf{Y}_{t+l} only depends on the last k observations, i.e. $P_{t+l}^\phi(\cdot | \mathbf{y}_{1:t}) = P_{t+l}^\phi(\cdot | \mathbf{y}_{t-k+1:t})$. Additionally, we assume that the conditional distribution does not depend explicitly on t , such that $P_{t+l}^\phi(\cdot | \mathbf{y}_{t-k+1:t}) = P_{(l)}^\phi(\cdot | \mathbf{y}_{t-k+1:t})$, where the bracketed subscript denotes that the forecast is for l steps ahead. This is the setting considered in the main manuscript.

In this specific case, therefore, the scoring rule used to penalize the forecast $P_{(l)}^\phi(\cdot | \mathbf{y}_{t-k+1:t})$ against the verification \mathbf{y}_{t+l} (Eq. 14) becomes:

$$S(P_{(l)}^\phi(\cdot | \mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l}).$$

³Technically, you can relax the $(k+l-1)$ -Markov assumption for the full sequence to assuming $(k+l-1)$ -Markovianity for $\mathbf{Y}_{1:2k+l-1}$ and independence of $\mathbf{Y}_{2k+l:T}$ on $\mathbf{Y}_{1:k+l-1}$; this is however quite artificial.

Therefore, the prequential score defined in Eq. (14) becomes:

$$S_T^{k,l}(P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1}), \mathbf{y}_{k+l:T}) = \frac{1}{T-l-k+1} \sum_{t=k}^{T-l} S(P_{(l)}^\phi(\cdot|\mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l}); \quad (19)$$

notice that we introduce here a scaling constant for simplicity; that however does not impact any of the following arguments. Recall also the definition of the expected prequential score:

$$\begin{aligned} S_T^{k,l}(P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1}), P_{k+l:T}^\star(\cdot|\mathbf{y}_{1:k+l-1})) \\ := \mathbb{E}_{\mathbf{Y}_{k+l:T} \sim P_{k+l:T}^\star(\cdot|\mathbf{y}_{1:k+l-1})} S_T^{k,l}(P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1}), \mathbf{Y}_{k+l:T}), \end{aligned} \quad (20)$$

for which we will use the following notation for brevity:

$$\tilde{S}_T^{k,l}(P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1})) := S_T^{k,l}(P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1}), P_{k+l:T}^\star(\cdot|\mathbf{y}_{1:k+l-1}))$$

As discussed in Appendix A.1.2 and shown in Theorem A.3, provided that S is strictly proper, $S_T^{k,l}$ is a strictly proper SR for k -Markovian with lag l distributions over $\mathbf{Y}_{k+l:T}|\mathbf{y}_{1:k+l-1}$, for all values of $\mathbf{y}_{1:k+l-1}$.

We will also consider the minimizer of the expectation of the expected prequential SR in Eq. (20) with respect to the initial data $\mathbf{y}_{1:k+l-1}$, i.e.:

$$\begin{aligned} S_T^{k,l^\star}(P_{k+l:T}^\phi) &:= \mathbb{E}_{\mathbf{Y}_{1:k+l-1} \sim P_{1:k+l-1}^\star} S_T^{k,l}(P_{k+l:T}^\phi(\cdot|\mathbf{Y}_{1:k+l-1}), P_{k+l:T}^\star(\cdot|\mathbf{Y}_{1:k+l-1})) \\ &= \mathbb{E}_{\mathbf{Y}_{1:T} \sim P_{1:T}^\star} S_T^{k,l}(P_{k+l:T}^\phi(\cdot|\mathbf{Y}_{1:k+l-1}), \mathbf{Y}_{k+l:T}). \end{aligned} \quad (21)$$

Theorem 3.3 in the main text states that the value of ϕ minimizing the empirical prequential SR (Eq. (19)) converges to both the minimizer of the expected (with respect to $\mathbf{Y}_{k+l:T}|\mathbf{y}_{1:k+l-1}$ for fixed $\mathbf{y}_{1:k+l-1}$) SR in Eq. (20) and to the minimizer of the expected (with respect to $\mathbf{Y}_{k+l:T}$) SR in Eq. (21). We will split the original result in two separate statements, which hold under similar Assumptions.

We now set notation and introduce the relevant quantities. From now onwards, we will drop k and l for brevity in the definition of S_T ; all following results hold for each fixed value of k and l . We write therefore $S_T(P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1}), \mathbf{y}_{k+l:T}) = S_T^{k,l}(P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1}), \mathbf{y}_{k+l:T})$, $\tilde{S}_T(P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1})) = \tilde{S}_T^{k,l}(P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1}))$ and $S_T^\star(P_{k+l:T}^\phi) = S_T^{k,l^\star}(P_{k+l:T}^\phi)$. Next, we define the minimizers of the empirical and expected prequential scores:

$$\begin{aligned} \hat{\phi}_T(\mathbf{y}_{1:T}) &: S_T(P_{k+l:T}^{\hat{\phi}_T(\mathbf{y}_{1:T})}(\cdot|\mathbf{y}_{1:k+l-1}), \mathbf{y}_{k+l:T}) = \min_{\phi \in \Phi} S_T(P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1}), \mathbf{y}_{k+l:T}) \\ \tilde{\phi}_T(\mathbf{y}_{1:k+l-1}) &: \tilde{S}_T(P_{k+l:T}^{\tilde{\phi}_T(\mathbf{y}_{1:k+l-1})}(\cdot|\mathbf{y}_{1:k+l-1})) = \min_{\phi \in \Phi} \tilde{S}_T(P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1})). \\ \phi_T^\star &: S_T^\star(P_{k+l:T}^{\phi_T^\star}) = \min_{\phi \in \Phi} S_T^\star(P_{k+l:T}^\phi). \end{aligned}$$

Convergence of $\hat{\phi}_T$ to ϕ_T^\star We first introduce Assumptions and give the statement linking $\hat{\phi}_T(\mathbf{y}_{1:T})$ to ϕ_T^\star (Theorem A.4). We require the sequence $(\mathbf{Y}_t)_t$ to be stationary and to satisfy some mixing properties. Specifically, the following Assumptions are required. The precise definition of the mixing properties is postponed to later in Appendix A.2.2.

A1 Φ is compact.

A2 ϕ_T^\star is unique; additionally, there exist a metric d on Φ such that, for all $\epsilon > 0$:

$$\liminf_{T \rightarrow +\infty} \left\{ \min_{\phi: d(\phi, \phi_T^\star) \geq \epsilon} S_T^\star(P_{k+l:T}^\phi) - S_T^\star(P_{k+l:T}^{\phi_T^\star}) \right\} > 0$$

A3 (Asymptotic stationarity) Let G_t be the marginal distribution of $\mathbf{Y}_{t-k+1:t+l}$ for $t \geq k$; then, $(T-l-k+1)^{-1} \sum_{t=k}^{T-l} G_t$ converges weakly to some probability measure on \mathcal{Y}^{k+l} as $T \rightarrow \infty$.

A4 Both conditions below are satisfied:

- (a) (Mixing) Either one of the following holds:
 - i. $(\mathbf{Y}_t)_t$ is α -mixing with mixing coefficient of size $r/(2r - 1)$, with $r \geq 1$, or
 - ii. $(\mathbf{Y}_t)_t$ is φ -mixing with mixing coefficient of size $r/(r - 1)$ with $r > 1$.
- (b) (Moment boundedness) Define $H(\mathbf{y}_{t-k+1:t+l}) = \sup_{\phi \in \Phi} |S(P^\phi(\cdot|\mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l})|$; then,

$$\sup_{t \geq k} \mathbb{E} \left[H(\mathbf{Y}_{t-k+1:t+l})^{r+\delta} \right] < \infty$$

for some $\delta > 0$, for the value of r corresponding to the condition above which is satisfied.

S being strictly proper and $P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1})$ being a well specified model for $\mathbf{Y}_{k+l:T}|\mathbf{y}_{1:k+l-1}$ is a sufficient (but not necessary) condition for the uniqueness of ϕ_T^* in Assumption **A2** (see Lemma A.8 in Appendix A.2.1), provided that the parameters ϕ are identifiable. Notice that neural networks do not have identifiable parameters; we require however this assumption to prove the Theorem. In case the parameters are not identifiable, we believe it is possible to show asymptotic convergence of the distributions minimizing the empirical and expected prequential SR, instead of convergence of the parameters. Extending the proof to this setting is technically challenging, as the distance in Assumption **A1** needs to be replaced by a divergence between probability distributions. We leave this extension for future work.

The rest of Assumption **A2** is a standard condition ensuring that the function which we are minimizing does not get flatter and flatter around the optimal value as $T \rightarrow \infty$. The asymptotic stationarity condition in Assumption **A3** is implied by the stronger condition of the marginals G_t being the same for each t . Assumption **A4(a)** is a mixing condition, ensuring that the dependence between two different $\mathbf{Y}_t, \mathbf{Y}_{t'}$ decreases as $t - t' \rightarrow \infty$ (defined precisely in Appendix A.2.2). Finally, Assumption **A4(b)** is a boundedness condition; for the specific case of the Kernel and Energy SR, that can be verified by simpler conditions as discussed in Lemma A.9 in Appendix A.2.1.

We will now state our first result.

Theorem A.4. *If Assumptions **A1**, **A2**, **A3** and **A4** hold, and if $(\mathbf{y}_{t-k+1:t+l}, \phi) \rightarrow S(P^\phi(\cdot|\mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l})$ is continuous on $\mathcal{Y}^{k+l} \times \Phi$, then $d(\hat{\phi}_T(\mathbf{Y}_{1:T}), \phi_T^*) \rightarrow 0$ with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^*$.*

The Theorem above relies on a generic consistency result (discussed in Appendix A.2.3) for which a uniform law of large numbers is required. Such a uniform law of large numbers can be obtained under stationarity and mixing conditions; we report in Appendix A.2.4 a result ensuring this. We prove Theorem A.4 by combining the above two elements in Appendix A.2.5.

Convergence of $\tilde{\phi}_T$ to ϕ_T^* We now give the statement linking $\hat{\phi}_T(\mathbf{y}_{1:T})$ to $\tilde{\phi}_T(\mathbf{y}_{1:t})$ (Theorem A.6). We will require similar Assumptions to what considered above, but holding for fixed values of $\mathbf{y}_{1:k+l-1}$:

B1 $\tilde{\phi}_T(\mathbf{y}_{1:k+l-1})$ is unique; additionally, there exist a metric d on Φ such that, for all $\epsilon > 0$:

$$\liminf_{T \rightarrow +\infty} \left\{ \min_{\phi: d(\phi, \tilde{\phi}_T(\mathbf{y}_{1:k+l-1})) \geq \epsilon} \tilde{S}_T(P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1})) - \tilde{S}_T(P_{k+l:T}^{\tilde{\phi}_T(\mathbf{y}_{1:k+l-1})}(\cdot|\mathbf{y}_{1:k+l-1})) \right\} > 0$$

B2 (Asymptotic stationarity) Let \tilde{G}_t be the marginal distribution of $\mathbf{Y}_{t-k+1:t+l}|\mathbf{y}_{1:k+l-1}$ for $t \geq k$; then,

$$(T - l - k + 1)^{-1} \sum_{t=k}^{T-l} \tilde{G}_t$$

converges weakly to some probability measure on \mathcal{Y}^{k+l} as $T \rightarrow \infty$.

B3 Both conditions below are satisfied:

- (a) (Mixing) Let $(\mathbf{X}_t)_t \sim P^*(\cdot|\mathbf{y}_{1:k+l-1})$; then, either one of the following holds:

- i. $(\mathbf{X}_t)_t$ is α -mixing with mixing coefficient of size $r/(2r-1)$, with $r \geq 1$, or
 - ii. $(\mathbf{X}_t)_t$ is φ -mixing with mixing coefficient of size $r/(r-1)$ with $r > 1$.
- (b) (Moment boundedness) Define $H(\mathbf{y}_{t-k+1:t+l}) = \sup_{\phi \in \Phi} |S(P^\phi(\cdot|\mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l})|$; then,

$$\sup_{t \geq k} \mathbb{E}_{\mathbf{Y}_{t-k+1:t+l}|\mathbf{Y}_{1:k+l-1}} \left[H(\mathbf{Y}_{t-k+1:t+l})^{r+\delta} \right] < \infty$$

for some $\delta > 0$, for the value of r corresponding to the condition above which is satisfied.

We can therefore state the following:

Theorem A.5. *If Assumptions A1, B1, B2 and B3 hold, and if $(\mathbf{y}_{t-k+1:t+l}, \phi) \rightarrow S(P^\phi(\cdot|\mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l})$ is continuous on $\mathcal{Y}^{k+l} \times \Phi$, then $d(\hat{\phi}_T(\mathbf{Y}_{1:k+l-1}, \mathbf{Y}_{k+l:T}), \tilde{\phi}_T(\mathbf{y}_{1:k+l-1})) \rightarrow 0$ with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^*(\cdot|\mathbf{Y}_{1:k+l-1})$.*

Notice how now in $\hat{\phi}_T$ we split the dependence with respect to the fixed $\mathbf{y}_{1:k+l-1}$ and the random $\mathbf{Y}_{k+l:T}$.

Proof. Theorem A.5 is proven following the same steps as Theorem A.4 (given in Appendix A.2.5). Specifically, Corollary A.16 can be used to obtain a uniform Law of Large Numbers such as in Assumption A5. Then, an equivalent to Theorem A.13 can be shown following the exact same steps. That implies the result of Theorem A.5. \square

The above result is saying that, for the sequence $(\mathbf{Y}_t)_t$ conditioned on $\mathbf{y}_{1:k+l-1}$, if stationarity and mixing conditions hold for a fixed $\mathbf{y}_{1:k+l-1}$, then the empirical minimizer $\hat{\phi}_T$ converges to the minimizer $\tilde{\phi}$, both with fixed $\mathbf{y}_{1:k+l-1}$.

Clearly, if the above Assumptions hold for all values of $\mathbf{y}_{1:k+l-1}$, the statement also does. This is made precise by the following Corollary:

Corollary A.6. *If Assumptions A1, B1, B2 and B3 hold almost surely for $\mathbf{Y}_{1:k+l-1} \sim P_{1:k+l-1}^*$, and if $(\mathbf{y}_{t-k+1:t+l}, \phi) \rightarrow S(P^\phi(\cdot|\mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l})$ is continuous on $\mathcal{Y}^{k+l} \times \Phi$, then*

$$d(\hat{\phi}_T(\mathbf{Y}_{1:k+l-1}, \mathbf{Y}_{k+l:T}), \tilde{\phi}_T(\mathbf{Y}_{1:k+l-1})) \rightarrow 0$$

with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^*$.

Proof. If Assumptions A1, B1, B2 and B3 hold almost surely for $\mathbf{Y}_{1:k+l-1} \sim P_{1:k+l-1}^*$, and under the continuity condition, the following statement holds with probability 1 with respect to $\mathbf{Y}_{1:k+l-1} \sim P_{1:k+l-1}^*$: “ $d(\hat{\phi}_T(\mathbf{Y}_{1:k+l-1}, \mathbf{Y}_{k+l:T}), \tilde{\phi}_T(\mathbf{Y}_{1:k+l-1})) \rightarrow 0$ with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^*(\cdot|\mathbf{Y}_{1:k+l-1})$,” from which the result follows by considering that a statement holding with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^*(\cdot|\mathbf{Y}_{1:k+l-1})$, for each value $\mathbf{Y}_{1:k+l-1}$ takes, and with probability 1 with respect to $\mathbf{Y}_{1:k+l-1} \sim P_{1:k+l-1}^*$ holds almost surely with respect to $(\mathbf{Y}_t)_t \sim P^*$. \square

Putting the two results together Finally, we also have the following, which is a precise version of Theorem 3.3 in the main text:

Corollary A.7. *If Assumptions A1, A2, A3 and A4 hold, and if Assumptions B1, B2 and B3 hold almost surely for $\mathbf{Y}_{1:k+l-1} \sim P_{1:k+l-1}^*$, and if $(\mathbf{y}_{t-k+1:t+l}, \phi) \rightarrow S(P^\phi(\cdot|\mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l})$ is continuous on $\mathcal{Y}^{k+l} \times \Phi$, then*

1. $d(\hat{\phi}_T(\mathbf{Y}_{1:T}), \phi_T^*) \rightarrow 0$ with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^*$;
2. $d(\hat{\phi}_T(\mathbf{Y}_{1:T}), \tilde{\phi}_T(\mathbf{Y}_{1:k+l-1})) \rightarrow 0$ with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^*$;
3. $d(\phi_T^*, \tilde{\phi}_T(\mathbf{Y}_{1:k+l-1})) \rightarrow 0$ with probability 1 with respect to $\mathbf{Y}_{1:k+l-1} \sim P_{1:k+l-1}^*$.

Proof. Under the Assumptions, both Theorem A.4 and Corollary A.6 hold, from which the first two statements follow. For the last statement, applying the triangle inequality yields:

$$d(\phi_T^*, \tilde{\phi}_T(\mathbf{Y}_{1:k+l-1})) \leq d(\hat{\phi}_T(\mathbf{Y}_{1:T}), \tilde{\phi}_T(\mathbf{Y}_{1:k+l-1})) + d(\hat{\phi}_T(\mathbf{Y}_{1:T}), \phi_T^*) \rightarrow 0.$$

As the left-hand side above depends only on $\mathbf{Y}_{1:k+l-1}$, the result holds almost surely with respect to $\mathbf{Y}_{1:k+l-1} \sim P_{1:k+l-1}^*$. \square

In case in which all the Assumption hold, therefore, the minimizer of the expected prequential SR over $\mathbf{Y}_{k+l:T}|\mathbf{Y}_{1:k+l-1}$ converges to the minimizer of the expected prequential SR over $\mathbf{Y}_{1:T}$, which is a deterministic quantity. Therefore, this result is saying that for large T , $\tilde{\phi}_T$ does not depend on the initial conditions, as it is intuitive under mixing and stationarity of $(\mathbf{Y}_t)_t$. Indeed, the same holds for the empirical minimizer $\hat{\phi}_T$, in which no expectation at all is computed.

In the next Subsections, we will discuss how to verify the Assumptions in some specific cases, and then move to introducing preliminary results for proving Theorem A.4, which we do in Appendix A.2.5. As mentioned above, the proof of Theorem A.5 follows the same steps as the one for Theorem A.4, but with the corresponding set of Assumptions. For this reason, we do not give that in details.

A.2.1 Verifying the Assumptions in specific cases

Before delving into proving Theorem A.4, we here show sufficient conditions under which ϕ_T^* and $\tilde{\phi}_T(\mathbf{y}_{1:k+l-1})$ are unique and under which Assumption **A4(b)** holds. Specifically, for the former (Lemma A.8), we consider the model $P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1})$ to be a well specified model and the scoring rule S to be strictly proper; for the latter, we consider instead the Kernel and Energy SR and obtain more precise conditions, which are easily satisfied.

First, consider uniqueness of ϕ_T^* :

Lemma A.8. *If S is strictly proper and $P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1})$ is a well specified model for $\mathbf{Y}_{k+l:T}|\mathbf{y}_{1:k+l-1}$, for all values of T , then ϕ_T^* and $\tilde{\phi}_T(\mathbf{y}_{1:k+l-1})$ are unique for all values of T and $\mathbf{y}_{1:k+l-1}$.*

Proof. If P^ϕ is well specified, there exists a ϕ^* such that $P_{k+l:T}^*(\cdot|\mathbf{y}_{1:k+l-1}) = P_{k+l:T}^{\phi^*}(\cdot|\mathbf{y}_{1:k+l-1}) \forall T, \forall \mathbf{y}_{1:k+l-1}$. Notice that this implies that P^* is k -Markovian with lag l . If S is strictly proper, we have by Theorem A.3 that:

$$\phi^* = \arg \min_{\phi \in \Phi} S_T(P_{k+l:T}^\phi(\cdot|\mathbf{y}_{1:k+l-1}), P_{k+l:T}^*(\cdot|\mathbf{y}_{1:k+l-1}))$$

is unique, for all $\mathbf{y}_{1:k+l-1}$. Therefore, $\tilde{\phi}_T(\mathbf{y}_{1:k+l-1}) = \phi^*$ for all values of $\mathbf{y}_{1:k+l-1}$. Recalling now the definition of $S_T^*(P_{k+l:T}^\phi)$ in Eq. (21), notice that the quantity inside the expectation $\mathbb{E}_{\mathbf{Y}_{1:k+l-1} \sim P_{1:k+l-1}^*}$ is minimized uniquely by $\phi = \phi^*$, so that $S_T^*(P_{k+l:T}^\phi)$ is also uniquely minimized by $\phi_T^* = \phi^*$. \square

We now consider Assumption **A4(b)**:

Lemma A.9. *Assumption **A4(b)** is verified in the following cases:*

- **Kernel SR** When $S = S_k$ for a kernel k which satisfies either of the following:

1. with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^{*4}$, for all $t \geq k$ and ϕ , $\mathbb{E}_{\mathbf{X}, \mathbf{X}' \sim P_{(l)}^\phi(\cdot|\mathbf{Y}_{t-k+1:t})} |k(\mathbf{X}, \mathbf{X}')| < \infty$ and $\mathbb{E}_{\mathbf{X} \sim P_{(l)}^\phi(\cdot|\mathbf{Y}_{t-k+1:t})} |k(\mathbf{X}, \mathbf{Y}_{t+l})| < \infty$;
2. k is bounded, i.e. $|k(\mathbf{y}, \mathbf{x})| < +\infty \forall \mathbf{y}, \mathbf{x} \in \mathcal{Y}$ (this implies the above condition).

- **Energy SR** When $S = S_E^{(\beta)}$ and either of the following holds:

1. with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^*$, for all $t \geq k$ and ϕ , $\mathbb{E}_{\mathbf{X}, \mathbf{X}' \sim P_{(l)}^\phi(\cdot|\mathbf{Y}_{t-k+1:t})} \|\mathbf{X} - \mathbf{X}'\| < \infty$ and $\mathbb{E}_{\mathbf{X} \sim P_{(l)}^\phi(\cdot|\mathbf{Y}_{t-k+1:t})} \|\mathbf{X} - \mathbf{Y}_{t+l}\| < \infty$;

⁴Put simply, this condition means that the following has to be true for all observed sequences $(\mathbf{y}_t)_t$ which can be generated by the distribution P^* .

2. the space \mathcal{Y} is bounded, such that $\|\mathbf{y}\| \leq B < \infty \forall \mathbf{y} \in \mathcal{Y}$ (this implies the first condition);
3. $\beta \geq 1$, $\mathbb{E}\|\mathbf{Y}_{t+l}\|^{\beta(r+\delta)} < \infty$ for all t and, with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^\star$, for all t and ϕ , $\mathbb{E}_{\mathbf{X} \sim P_{(l)}^\phi(\cdot|\mathbf{y}_{t-k+1:t})}\|\mathbf{X}\|^\beta \leq B < \infty$.

Proof. First, notice that $\sup_{t \geq k} \mathbb{E} [H(\mathbf{Y}_{t-k+1:t+l})^{r+\delta}] < \infty \iff \mathbb{E} [H(\mathbf{Y}_{t-k+1:t+l})^{r+\delta}] < \infty \forall t \geq k$.

Kernel SR Consider the kernel SR $S = S_k$:

$$\begin{aligned} |S_k(P_{(l)}^\phi(\cdot|\mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l})| &= |\mathbb{E}_{\mathbf{X}, \mathbf{X}' \sim P_{(l)}^\phi(\cdot|\mathbf{y}_{t-k+1:t})}[k(\mathbf{X}, \mathbf{X}') - 2k(\mathbf{X}, \mathbf{y}_{t+l})]| \\ &\leq \mathbb{E}_{\mathbf{X}, \mathbf{X}' \sim P_{(l)}^\phi(\cdot|\mathbf{y}_{t-k+1:t})}|k(\mathbf{X}, \mathbf{X}') - 2k(\mathbf{X}, \mathbf{y}_{t+l})| \\ &\leq \mathbb{E}_{\mathbf{X}, \mathbf{X}' \sim P_{(l)}^\phi(\cdot|\mathbf{y}_{t-k+1:t})}[|k(\mathbf{X}, \mathbf{X}')| + 2|k(\mathbf{X}, \mathbf{y}_{t+l})|]. \end{aligned} \quad (22)$$

We first show why condition 1 yields the result. If, with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^\star$, for all $t \geq k$ and ϕ :

$$\mathbb{E}_{\mathbf{X}, \mathbf{X}' \sim P_{(l)}^\phi(\cdot|\mathbf{Y}_{t-k+1:t})}|k(\mathbf{X}, \mathbf{X}')| \leq \kappa_1 < \infty \quad \text{and} \quad \mathbb{E}_{\mathbf{X} \sim P_{(l)}^\phi(\cdot|\mathbf{Y}_{t-k+1:t})}|k(\mathbf{X}, \mathbf{Y}_{t+l})| \leq \kappa_2 < \infty,$$

we have that:

$$|S_k(P_{(l)}^\phi(\cdot|\mathbf{Y}_{t-k+1:t}), \mathbf{Y}_{t+l})| \leq \kappa_1 + 2\kappa_2 < \infty,$$

from which:

$$\begin{aligned} \mathbb{E} [H(\mathbf{Y}_{t-k+1:t+l})^{r+\delta}] &= \mathbb{E} \left[\left(\sup_{\phi \in \Phi} |S_k(P_{(l)}^\phi(\cdot|\mathbf{Y}_{t-k+1:t}), \mathbf{Y}_{t+l})| \right)^{r+\delta} \right] \\ &\leq \mathbb{E} \left[\left(\sup_{\phi \in \Phi} \kappa_1 + 2\kappa_2 \right)^{r+\delta} \right] = (\kappa_1 + 2\kappa_2)^{r+\delta} < \infty. \end{aligned}$$

Now, condition 2 implies condition 1. Therefore, condition 2 yields the result.

Energy SR Notice how the kernel SR recovers the Energy SR when $k(\mathbf{y}, \mathbf{x}) = -\|\mathbf{y} - \mathbf{x}\|^\beta$; condition 1 for the kernel SR corresponds therefore to condition 1 for the Energy SR; therefore, the result holds under condition 1.

For condition 2 for the Energy SR, notice that

$$|k(\mathbf{y}, \mathbf{x})| = \|\mathbf{y} - \mathbf{x}\|^\beta \leq (\|\mathbf{y}\| + \|\mathbf{x}\|)^\beta \leq (2B)^\beta,$$

where the first inequality comes from applying the triangle inequality and the second comes from condition 2 for the Energy SR. Therefore, condition 2 for the Energy SR implies condition 2 for the corresponding Kernel SR, from which the result follows.

Finally, an alternative route leads to condition 3. Specifically, for the Energy SR, Equation (22) becomes:

$$\begin{aligned} |S_E^{(\beta)}(P_{(l)}^\phi(\cdot|\mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l})| &\leq \mathbb{E}_{\mathbf{X}, \mathbf{X}' \sim P_{(l)}^\phi(\cdot|\mathbf{y}_{t-k+1:t})}[|\|\mathbf{X} - \mathbf{X}'\|^\beta + 2\|\mathbf{X} - \mathbf{y}_{t+l}\|^\beta|] \\ &\leq \mathbb{E}_{\mathbf{X}, \mathbf{X}' \sim P_{(l)}^\phi(\cdot|\mathbf{y}_{t-k+1:t})}[(\|\mathbf{X}\| + \|\mathbf{X}'\|)^\beta + 2(\|\mathbf{X}\| + \|\mathbf{y}_{t+l}\|)^\beta] \end{aligned}$$

by triangle inequality. Now, for any $\beta > 1$, $a, b > 0$, $(a + b)^\beta \leq 2^{\beta-1}(a^\beta + b^\beta)$,⁵ therefore,

$$|S_E^{(\beta)}(P_{(l)}^\phi(\cdot|\mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l})| \leq \mathbb{E}_{\mathbf{X}, \mathbf{X}' \sim P_{(l)}^\phi(\cdot|\mathbf{y}_{t-k+1:t})}[2^{\beta-1}(\|\mathbf{X}\|^\beta + \|\mathbf{X}'\|^\beta) + 2^\beta(\|\mathbf{X}\|^\beta + \|\mathbf{y}_{t+l}\|^\beta)].$$

⁵This inequality is well-known and can be shown by convexity.

From the above, we have that:

$$\begin{aligned}\mathbb{E} \left[H(\mathbf{Y}_{t-k+1:t+l})^{r+\delta} \right] &= \mathbb{E} \left[\left(\sup_{\phi \in \Phi} |S_E^{(\beta)}(P_{(l)}^\phi(\cdot | \mathbf{Y}_{t-k+1:t}), \mathbf{Y}_{t+l})| \right)^{r+\delta} \right] \\ &\leq \mathbb{E} \left[\left(\sup_{\phi \in \Phi} \mathbb{E}_{\mathbf{X}, \mathbf{X}' \sim P_{(l)}^\phi(\cdot | \mathbf{Y}_{t-k+1:t})} [2^{\beta-1}(\|\mathbf{X}\|^\beta + \|\mathbf{X}'\|^\beta) + 2^\beta(\|\mathbf{X}\|^\beta + \|\mathbf{Y}_{t+l}\|^\beta)] \right)^{r+\delta} \right].\end{aligned}$$

If, with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^*$, for all $t \geq k$ and ϕ , $\mathbb{E}_{\mathbf{X} \sim P_{(l)}^\phi(\cdot | \mathbf{Y}_{t-k+1:t})} \|\mathbf{X}\|^\beta \leq B < \infty$, we have therefore:

$$\begin{aligned}\mathbb{E} \left[H(\mathbf{Y}_{t-k+1:t+l})^{r+\delta} \right] &\leq \mathbb{E} \left[\left(2^{\beta-1}(B+B) + 2^\beta(B + \|\mathbf{Y}_{t+l}\|^\beta) \right)^{r+\delta} \right] \\ &= \mathbb{E} \left[\left(2^{\beta+1}B + 2^\beta \|\mathbf{Y}_{t+l}\|^\beta \right)^{r+\delta} \right].\end{aligned}$$

Now, denote $\delta' = r + \delta$; $\delta' > 1$ by assumption. It holds therefore, as above, $(a+b)^{\delta'} \leq 2^{\delta'-1}(a^{\delta'} + b^{\delta'})$ for $a, b > 0$; we have therefore that:

$$\mathbb{E} \left[H(\mathbf{Y}_{t-k+1:t+l})^{\delta'} \right] \leq 2^{-1} \left(2^{\beta+2}B \right)^{\delta'} + 2^{\delta'(\beta+1)-1} \mathbb{E} \|\mathbf{Y}_{t+l}\|^{\beta\delta'};$$

the above expression is therefore bounded whenever $\mathbb{E} \|\mathbf{Y}_{t+l}\|^{\beta(r+\delta)} < \infty$. \square

A.2.2 Defining the mixing conditions

Here, we give the precise definitions for the mixing conditions stated in Assumption **A4**(a). More background on the following definitions can be found, for instance, in Bradley, 2005.

Definition A.10 (Measures of dependence). Consider a probability space (Ω, \mathcal{F}, P) ; for any two sigma algebras $\mathcal{A} \subseteq \mathcal{F}$ and $\mathcal{B} \subseteq \mathcal{F}$, define:

$$\begin{aligned}\alpha_P(\mathcal{A}, \mathcal{B}) &:= \sup_{A \in \mathcal{A}, B \in \mathcal{B}} |P(A \cap B) - P(A)P(B)|, \\ \varphi_P(\mathcal{A}, \mathcal{B}) &:= \sup_{A \in \mathcal{A}, B \in \mathcal{B}: P(B) > 0} |P(B|A) - P(B)|.\end{aligned}$$

For $1 \leq r \leq s \leq \infty$, define the Borel σ -algebra of events generated from $(\mathbf{Y}_r, \mathbf{Y}_{r+1}, \dots, \mathbf{Y}_{s-1}, \mathbf{Y}_s)$ as \mathcal{G}_r^s . Then, we define:

$$\alpha^{\mathbf{Y}}(m) = \sup_{r \geq 1} \alpha_{P^*}(\mathcal{G}_1^r, \mathcal{G}_{r+m}^{+\infty}), \quad \varphi^{\mathbf{Y}}(m) = \sup_{r \geq 1} \varphi_{P^*}(\mathcal{G}_1^r, \mathcal{G}_{r+m}^{+\infty}).$$

Definition A.11. The random sequence $(\mathbf{Y}_t)_t$ is said α -mixing if $\alpha^{\mathbf{Y}}(m) \rightarrow 0$ as $m \rightarrow \infty$ and φ -mixing if $\varphi^{\mathbf{Y}}(m) \rightarrow 0$ as $m \rightarrow \infty$. It can be seen that φ -mixing implies α -mixing [Domowitz and White, 1982].

Definition A.12. We say that the mixing coefficients $\varphi^{\mathbf{Y}}(m)$ are of size s [Domowitz and White, 1982] if $\varphi^{\mathbf{Y}}(m) = \mathcal{O}(m^{-\lambda})$ for $\lambda > s$; similar definition can be given for the coefficients $\alpha^{\mathbf{Y}}(m)$.

In Bradley [2005], the definitions for the quantities above consider a sequence $(\mathbf{X}_t)_{t \in \mathbb{Z}}$, and defined:

$$\alpha^{\mathbf{X}}(m) = \sup_{r \in \mathbb{Z}} \alpha_P(\mathcal{G}_{-\infty}^r, \mathcal{G}_{r+m}^{+\infty}),$$

for some distribution P , and similar for $\varphi^{\mathbf{X}}(m)$. Our definition can be cast in this way by defining $\mathbf{X}_t = \mathbf{Y}_t \forall t \geq 1$ and $\mathbf{X}_t = 0 \forall t \leq 0$.

A.2.3 Generic consistency result

We consider here the following Assumption:

A5 (Uniform Law of Large Numbers.) The following holds with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^*$:

$$\sup_{\phi \in \Phi} \left| S_T(P_{k+l:T}^\phi(\cdot | \mathbf{Y}_{1:k+l-1}), \mathbf{Y}_{k+l:T}) - S_T^*(P_{k+l:T}^\phi) \right| \rightarrow 0.$$

We give here a consistency result more general than Theorem A.4, as in fact Assumption **A5** is more general than the stationarity and mixing conditions in Assumption **A3** and **A4**.

Theorem A.13 (Theorem 5.1 in Skouras, 1998). *If Assumptions **A2** and **A5** hold, then $d(\hat{\phi}_T(\mathbf{Y}_{1:T}), \phi_T^*) \rightarrow 0$ with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^*$.*

We report here a proof for ease of reference.

Proof. By the definition of \liminf , for a fixed $\epsilon > 0$, Assumption **A2** implies that there exists $T_1(\epsilon)$ such that:

$$\delta(\epsilon) := \left\{ \inf_{T > T_1(\epsilon)} \min_{\phi: d(\phi, \phi_T^*) \geq \epsilon} S_T^*(P_{k+l:T}^\phi) - S_T^*(P_{k+l:T}^{\phi_T^*}) \right\} > 0. \quad (23)$$

Due to Assumption **A5**, with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^*$, there exists $T_2((\mathbf{Y}_t)_t, \delta(\epsilon))$ such that, for all $T > T_2((\mathbf{Y}_t)_t, \delta(\epsilon))$:

$$\left| S_T(P_{k+l:T}^{\phi_T^*}(\cdot | \mathbf{Y}_{1:k+l-1}), \mathbf{Y}_{k+l:T}) - S_T^*(P_{k+l:T}^{\phi_T^*}) \right| < \delta(\epsilon)/2,$$

which implies:

$$S_T^*(P_{k:T}^{\phi_T^*}) > S_T(P_{k:T}^{\phi_T^*}(\cdot | \mathbf{Y}_{1:k+l-1}), \mathbf{Y}_{k+l:T}) - \delta(\epsilon)/2 \geq S_T(P_{k+l:T}^{\hat{\phi}_T(\mathbf{Y}_{1:T})}(\cdot | \mathbf{Y}_{1:k+l-1}), \mathbf{Y}_{k+l:T}) - \delta(\epsilon)/2, \quad (24)$$

where the second inequality is valid thanks to the definition of $\hat{\phi}_T(\mathbf{Y}_{1:T})$.

Similarly, by exploiting Assumption **A5** again, with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^*$, there exists $T_3((\mathbf{Y}_t)_t, \delta(\epsilon))$ such that, for all $T > T_3((\mathbf{Y}_t)_t, \delta(\epsilon))$:

$$\left| S_T^*(P_{k+l:T}^{\hat{\phi}_T(\mathbf{Y}_{1:T})}) - S_T(P_{k+l:T}^{\hat{\phi}_T(\mathbf{Y}_{1:T})}(\cdot | \mathbf{Y}_{1:k+l-1}), \mathbf{Y}_{k+l:T}) \right| < \delta(\epsilon)/2. \quad (25)$$

Then, with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^*$, for all $T > \max\{T_2((\mathbf{Y}_t)_t, \delta(\epsilon)), T_3((\mathbf{Y}_t)_t, \delta(\epsilon))\}$:

$$\begin{aligned} S_T^*(P_{k+l:T}^{\hat{\phi}_T(\mathbf{Y}_{1:T})}) - S_T^*(P_{k+l:T}^{\phi_T^*}) &\leq S_T^*(P_{k+l:T}^{\hat{\phi}_T(\mathbf{Y}_{1:T})}) - S_T(P_{k+l:T}^{\hat{\phi}_T(\mathbf{Y}_{1:T})}(\cdot | \mathbf{Y}_{1:k+l-1}), \mathbf{Y}_{k+l:T}) + \delta(\epsilon)/2 \\ &< \delta(\epsilon)/2 + \delta(\epsilon)/2 = \delta(\epsilon), \end{aligned} \quad (26)$$

where the first inequality is thanks to Eq. (24) and the second is thanks to Eq (25).

Now, Eq. (23) and Eq. (26) both hold with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^*$ for all $T > \max\{T_1(\delta(\epsilon)), T_2((\mathbf{Y}_t)_t, \delta(\epsilon)), T_3((\mathbf{Y}_t)_t, \delta(\epsilon))\}$. Notice that Eq. (26) ensures that the difference considered in Eq. (23) is smaller than $\delta(\epsilon)$ for $\phi = \hat{\phi}_T(\mathbf{Y}_{1:T})$; However, Eq. (23) states that the same difference is larger or equal than $\delta(\epsilon)$ for all $\phi : d(\phi, \phi_T^*) \geq \epsilon$, from which it follows that $d(\hat{\phi}_T(\mathbf{Y}_{1:T}), \phi_T^*) < \epsilon$ with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^*$. As ϵ is however arbitrary, it follows that, with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^*$:

$$d(\hat{\phi}_T(\mathbf{Y}_{1:T}), \phi_T^*) \rightarrow 0.$$

□

A.2.4 Uniform law of large numbers

We will here show how the Uniform Law of Large Numbers in Assumption **A5** can be obtained from the stationarity and mixing conditions in **A3** and **A4**. To this aim, we exploit a result in Pötscher and Prucha [1989].

We consider now a generic sequence of random variables $\mathbf{Z}_t \in \mathcal{Z}$, and a function $q : \mathcal{Z} \times \Phi \rightarrow \mathbb{R}$. Let us denote now by \mathcal{F} the Borel σ -algebra generated by the sequence $(\mathbf{Z}_t)_t$, $\Omega_{\mathbf{Z}}$ the space of realizations of $(\mathbf{Z}_t)_t$ and Q^* the probability distribution for it.

Consider the following Assumptions:

C1 (Dominance condition) For $D(\mathbf{z}) = \sup_{\phi \in \Phi} |q(\mathbf{z}, \phi)|$, there is some $\delta > 0$ such that

$$\sup_t \frac{1}{N} \sum_{t=1}^N \mathbb{E} \left[D(\mathbf{Z}_t)^{1+\delta} \right] < \infty.$$

C2 (Asymptotic stationarity) Let Q_t^* be the marginal distribution of \mathbf{Z}_t ; then, $N^{-1} \sum_{t=1}^N Q_t^*$ converges weakly to some probability measure F on \mathcal{Z} .

C3 (Pointwise law of large numbers) For some metric ρ on Φ , let:

$$\bar{q}(\mathbf{z}, \phi, \tau) := \sup_{\phi': \rho(\phi, \phi') < \tau} q(\mathbf{z}, \phi'), \quad \underline{q}(\mathbf{z}, \phi, \tau) := \inf_{\phi': \rho(\phi, \phi') < \tau} q(\mathbf{z}, \phi').$$

For all $\phi \in \Phi$, there exists a sequence of positive numbers $\tau_i(\phi)$ such that $\tau_i(\phi) \rightarrow 0$ as $i \rightarrow \infty$, and such that for each τ_i the random variables $\bar{q}(\mathbf{Z}_t, \phi, \tau_i)$ and $\underline{q}(\mathbf{Z}_t, \phi, \tau_i)$ satisfy a strong law of large numbers, i.e., as $N \rightarrow \infty$:

$$\begin{aligned} \frac{1}{N} \sum_{t=1}^N \{ \bar{q}(\mathbf{Z}_t, \phi, \tau_i) - \mathbb{E} [\bar{q}(\mathbf{Z}_t, \phi, \tau_i)] \} &\rightarrow 0 \\ \frac{1}{N} \sum_{t=1}^N \{ \underline{q}(\mathbf{Z}_t, \phi, \tau_i) - \mathbb{E} [\underline{q}(\mathbf{Z}_t, \phi, \tau_i)] \} &\rightarrow 0, \end{aligned}$$

where the two above equations hold with probability 1 with respect to $(\mathbf{Z}_t)_t \sim Q^*$.

Theorem A.14 (Theorem 2 in Pötscher and Prucha, 1989). *If Assumptions **A1**, **C1**, **C2** and **C3** hold and if $q(\mathbf{z}, \phi)$ is continuous on $\mathcal{Z} \times \Phi$, then:*

(i) *with probability 1 with respect to $(\mathbf{Z}_t)_t \sim Q^*$,*

$$\lim_{t \rightarrow \infty} \sup_{\phi \in \Phi} \left| \frac{1}{N} \sum_{t=1}^N \{ q(\mathbf{Z}_t, \phi) - \mathbb{E} [q(\mathbf{Z}_t, \phi)] \} \right| = 0;$$

(ii) *$\int q(\mathbf{z}, \phi) dF(\mathbf{z})$ exists and is finite, continuous on Φ and, with probability 1 with respect to $(\mathbf{Z}_t)_t \sim Q^*$,*

$$\lim_{t \rightarrow \infty} \sup_{\phi \in \Phi} \left| \frac{1}{N} \sum_{t=1}^N q(\mathbf{Z}_t, \phi) - \int q(\mathbf{z}, \phi) dF(\mathbf{z}) \right| = 0;$$

We now give sufficient conditions for Assumption **C3** to hold. In fact, sequences for which the dependence of \mathbf{Z}_t on a past observation \mathbf{Z}_{t-m} decreases to 0 quickly enough as $m \rightarrow \infty$ satisfy Assumption **C3**. This can be made more rigorous considering the definitions of α - and φ -mixing sequences given in Appendix A.2.2.

Given the sequence $(\mathbf{Z}_t)_t$, for $1 \leq r \leq s \leq \infty$, define the Borel σ -algebra of events generated from $(\mathbf{Z}_r, \mathbf{Z}_{r+1}, \dots, \mathbf{Z}_{s-1}, \mathbf{Z}_s)$ as \mathcal{F}_r^s . Then, we define the mixing coefficients for $(\mathbf{Z}_t)_t$ as:

$$\alpha^{\mathbf{Z}}(m) = \sup_{r \geq 1} \alpha_{Q^*}(\mathcal{F}_1^r, \mathcal{F}_{r+m}^{+\infty}), \quad \varphi^{\mathbf{Z}}(m) = \sup_{r \geq 1} \varphi_{Q^*}(\mathcal{F}_1^r, \mathcal{F}_{r+m}^{+\infty}).$$

Similarly to before, the random sequence $(\mathbf{Z}_t)_{t \in \mathbb{Z}}$ is said α -mixing if $\alpha^{\mathbf{Z}}(m) \rightarrow 0$ as $m \rightarrow \infty$ and φ -mixing if $\varphi^{\mathbf{Z}}(m) \rightarrow 0$ as $m \rightarrow \infty$. Additionally, we say that the mixing coefficients $\varphi^{\mathbf{Z}}(m)$ are of size s Domowitz and White [1982] if $\varphi^{\mathbf{Z}}(m) = \mathcal{O}(m^{-\lambda})$ for $\lambda > s$; similar definition can be given for the coefficients $\alpha^{\mathbf{Z}}(m)$.

Let us define now the following additional assumption:

C4 Both conditions below hold:

- (a) (Mixing) Either one of the following holds:
 - i. $(\mathbf{Z}_t)_t$ is α -mixing with mixing coefficient of size $r/(2r-1)$, with $r \geq 1$, or
 - ii. $(\mathbf{Z}_t)_t$ is φ -mixing with mixing coefficient of size $r/(r-1)$ with $r > 1$.
- (b) (Moment boundedness) $\sup_t \mathbb{E} [D(\mathbf{Z}_t)^{r+\delta}] < \infty$ for some $\delta > 0$, for the value of r corresponding to the condition above which is satisfied.

We give the following Lemma, which is contained in Corollary 1 in Pötscher and Prucha, 1989.

Lemma A.15 (Corollary 1 in Pötscher and Prucha, 1989). *Assumption C4 implies Assumptions C1 and C3.*

We can therefore state the following.

Corollary A.16. *If Assumptions A1, C2 and C4 hold and if $q(\mathbf{z}, \phi)$ is continuous on $\mathcal{Z} \times \Phi$, then the conclusions of Theorem A.14 are satisfied.*

A.2.5 Proving Theorem A.4

Here, we finally prove Theorem A.4 by combining the generic consistency result in Appendix A.2.3 with the uniform law of large number result reported in Appendix A.2.4.

Notice that, in stating Theorem A.14 and Corollary A.16, we have considered a generic sequence $(\mathbf{Z}_t)_t$. In the setting of our interest, however, we want to study the prequential scoring rule defined in Eq. (19), and use Corollary A.16 to state conditions under which Assumption A5, and therefore Theorem A.13, hold.

To this aim, we identify now $N = T - k - l + 1$, $\mathbf{Z}_t = \mathbf{Y}_{t:t+k+l-1}$ and $q(\mathbf{Z}_t, \phi) = S(P_{(l)}^\phi(\cdot | \mathbf{Y}_{t:t+k-1}), \mathbf{Y}_{t+k+l-1})$; which leads to:

$$\begin{aligned} \frac{1}{N} \sum_{t=1}^N q(\mathbf{Z}_t, \phi) &= \frac{1}{T - k - l + 1} \sum_{t=1}^{T-k-l+1} S(P_{(l)}^\phi(\cdot | \mathbf{Y}_{t:t+k-1}), \mathbf{Y}_{t+k+l-1}) \\ &= \frac{1}{T - k - l + 1} \sum_{t=k}^{T-l} S(P_{(l)}^\phi(\cdot | \mathbf{Y}_{t-k+1:t}), \mathbf{Y}_{t+l}) \\ &= S_T(P_{k+l:T}^\phi(\cdot | \mathbf{Y}_{1:k+l-1}), \mathbf{Y}_{k+l:T}). \end{aligned}$$

The distribution Q^* on $(\mathbf{Z}_t)_t$ considered in the previous section is induced therefore by P^* over $(\mathbf{Y}_t)_t$.

We want now to relate $\alpha^{\mathbf{Y}}(m)$ and $\varphi^{\mathbf{Y}}(m)$ to $\alpha^{\mathbf{Z}}(m)$ and $\varphi^{\mathbf{Z}}(m)$; in order to do so, notice that, as $\mathbf{Z}_t = \mathbf{Y}_{t:t+k+l-1}$, $\mathcal{F}_r^{\mathbf{Z}} = \mathcal{G}_r^{s+k+l-1}$. Therefore,

$$\begin{aligned} \alpha^{\mathbf{Z}}(m) &= \sup_{r \geq 1} \alpha^{\mathbf{Z}}(\mathcal{F}_1^r, \mathcal{F}_{r+m}^{+\infty}) = \sup_{r \geq 1} \alpha^{\mathbf{Y}}(\mathcal{G}_1^{r+k+l-1}, \mathcal{G}_{r+m}^{+\infty}) \\ &= \sup_{r \geq k+l} \alpha^{\mathbf{Y}}(\mathcal{G}_1^r, \mathcal{G}_{r+m-k-l+1}^{+\infty}) \leq \sup_{r \geq 1} \alpha^{\mathbf{Y}}(\mathcal{G}_1^r, \mathcal{G}_{r+m-k-l+1}^{+\infty}) = \alpha^{\mathbf{Y}}(m - k - l + 1), \end{aligned}$$

and, similarly, $\varphi^{\mathbf{Z}}(m) \leq \varphi^{\mathbf{Y}}(m - k - l + 1)$. As k is fixed, $\varphi^{\mathbf{Y}}(m) \rightarrow 0 \implies \varphi^{\mathbf{Z}}(m) \rightarrow 0$ as $m \rightarrow \infty$, which is to say, $(\mathbf{Y}_t)_t$ being φ -mixing implies $(\mathbf{Z}_t)_t$ is φ -mixing as well, and similar for α -mixing. Additionally, if the mixing coefficients for $(\mathbf{Z}_t)_t$ have a given size s , then the mixing coefficients for $(\mathbf{Y}_t)_t$ will have the same size, and viceversa. In fact, $\varphi^{\mathbf{Z}}(m) \leq \varphi^{\mathbf{Y}}(m - k - l + 1) = \mathcal{O}(m^{-\lambda})$ implies either $\varphi^{\mathbf{Y}}(m) = \mathcal{O}(m^{-\lambda})$ or $\varphi^{\mathbf{Y}}(m) = o(m^{-\lambda})$, and similar for α -mixing.

We are now ready to prove Theorem A.4.

Proof of Theorem A.4. Notice that, by identifying $\mathbf{Z}_t = \mathbf{Y}_{t:t+k+l-1}$ and $q(\mathbf{Z}_t, \phi) = S(P_{(l)}^\phi(\cdot | \mathbf{y}_{t:t+k-1}, \mathbf{y}_{t+k+l-1}))$, Assumption **A3** corresponds to Assumption **C2**, and Assumption **A4** implies Assumption **C4**, due to the conservation of size of the mixing coefficients discussed above.

Together with Assumption **A1** and the continuity condition, therefore, Corollary A.16 holds, from which you have that, with probability 1 with respect to $(\mathbf{Y}_t)_t \sim P^*$,

$$\lim_{T \rightarrow \infty} \sup_{\phi \in \Phi} \left| \frac{1}{T - k - l + 1} \sum_{t=k}^{T-l} \left\{ S(P_{(l)}^\phi(\cdot | \mathbf{Y}_{t-k+1:t}), \mathbf{Y}_{t+l}) - \mathbb{E} \left[S(P_{(l)}^\phi(\cdot | \mathbf{Y}_{t-k+1:t}), \mathbf{Y}_{t+l}) \right] \right\} \right| = 0;$$

which, recalling the definition of $S_T(P_{k+l:T}^\phi(\cdot | \mathbf{Y}_{1:k+l-1}), \mathbf{Y}_{k+l:T})$ and $S_T^*(P_{k+l:T}^\phi)$ in Eqs. (19) and (20), is the same as Assumption **A5**. Thanks to this and Assumption **A2**, therefore, Theorem A.13 holds, from which the result follows. \square

B More details on the different methods

B.1 Training generative networks via divergence minimization

B.1.1 f -GAN

The f -GAN approach is defined by considering an f -divergence in place of D in Eq. (1) in the main text:

$$D_f(P^* || P^\phi) = \int_{\mathcal{Y}} p^\phi(\mathbf{y}) f\left(\frac{p^*(\mathbf{y})}{p^\phi(\mathbf{y})}\right) d\mu(\mathbf{y}),$$

where $f: \mathbb{R}_+ \rightarrow \mathbb{R}$ is a convex, lower-semicontinuous function for which $f(1) = 0$, and where p^ϕ and p^* are densities of P^ϕ and P^* with respect to a base measure μ . Let now dom_f denote the domain of f . By exploiting the Fenchel conjugate $f^*(t) = \sup_{u \in \text{dom}_f} \{ut - f(u)\}$, Nowozin et al., 2016 obtain the following variational lower bound:

$$D_f(P^* || P^\phi) \geq \sup_{c \in \mathcal{C}} (\mathbb{E}_{\mathbf{Y} \sim P^*} c(\mathbf{Y}) - \mathbb{E}_{\mathbf{X} \sim P^\phi} f^*(c(\mathbf{X}))),$$

which holds for any set of functions \mathcal{C} from \mathcal{Y} to dom_{f^*} . By considering a parametric set of functions $\mathcal{C} = \{c_\psi: \mathcal{Y} \rightarrow \text{dom}_{f^*}, \psi \in \Psi\}$, a surrogate to the problem in Eq. (1) in the main text becomes:

$$\min_{\phi} \max_{\psi} (\mathbb{E}_{\mathbf{Y} \sim P^*} c_\psi(\mathbf{Y}) - \mathbb{E}_{\mathbf{X} \sim P^\phi} f^*(c_\psi(\mathbf{X}))).$$

In the conditional setting discussed in Section 2.1 in the main text, the above generalizes to:

$$\min_{\phi} \max_{\psi} \mathbb{E}_{\theta \sim \Pi} (\mathbb{E}_{\mathbf{Y} \sim P^*(\cdot | \theta)} c_\psi(\mathbf{Y}; \theta) - \mathbb{E}_{\mathbf{Y} \sim P^\phi(\cdot | \theta)} f^*(c_\psi(\mathbf{Y}; \theta))), \quad (28)$$

By denoting as $P_{\theta, \mathbf{Y}}^*$ and $P_{\theta, \mathbf{Y}}^\phi$ the joint distributions over $\Theta \times \mathcal{Y}$, Eq. (28) corresponds to the relaxation of $D_f(P_{\theta, \mathbf{Y}}^* || P_{\theta, \mathbf{Y}}^\phi)$ under the constraint that the marginal of $P_{\theta, \mathbf{Y}}^\phi$ for θ is equal to Π .

In order to solve the problem in Eq. (28), alternating optimization over ϕ and ψ can be performed; in Algorithm 1, we show a single epoch (i.e., a loop on the full training dataset) of conditional f -GAN training; for simplicity, we consider here using a single pair (θ_i, \mathbf{y}_i) to estimate the expectations in Eq. (28) (i.e., the batch size is 1), but using a larger number of samples is indeed possible. Notice how in Algorithm 1 we update the critic once every generator update; however, multiple critic updates can be done.

B.1.2 Wasserstein-GAN (WGAN)

Arjovsky et al., 2017 exploited the following expression for the 1-Wasserstein distance:

$$W(P^*, P^\phi) = \sup_{c: \|c\|_L \leq 1} \mathbb{E}_{\mathbf{Y} \sim P^*} [c(\mathbf{Y})] - \mathbb{E}_{\mathbf{X} \sim P^\phi} [c(\mathbf{X})], \quad (29)$$

Algorithm 1 Single epoch conditional f -GAN training.

Require: Parametric map h_ϕ , critic network c_ψ , learning rates ϵ, γ .

for each training pair $(\boldsymbol{\theta}_i, \mathbf{y}_i)$ **do**

 Sample $\mathbf{z} \sim Q$

 Obtain $\hat{\mathbf{x}}_i^\phi = h_\phi(\mathbf{z}, \boldsymbol{\theta}_i)$

 Set $\psi \leftarrow \psi + \gamma \cdot \nabla_\psi \left[c_\psi(\mathbf{y}_i, \boldsymbol{\theta}_i) - f^*(c_\psi(\hat{\mathbf{x}}_i^\phi, \boldsymbol{\theta}_i)) \right]$

 Set $\phi \leftarrow \phi - \epsilon \cdot \nabla_\phi \left[-f^*(c_\psi(\hat{\mathbf{x}}_i^\phi, \boldsymbol{\theta}_i)) \right]$

end for

where $\|c\|_L$ denotes the Lipschitz constant of the function c . The different notation here highlights how W is a symmetric function. Plugging Eq. (29) into Eq. (1) in the main text leads again to an adversarial setting; here, the Lipschitz constraint can be enforced by clipping the weights of the neural network to a given range Arjovsky et al. [2017]. Alternatively, this hard constraint can be relaxed to a soft one via gradient penalization Gulrajani et al. [2017].

B.1.3 MMD-GAN

A specific case of the MMD (Eq. 3 in the main text) is the Energy Distance:

$$\mathcal{E} \left(P^*, P^\phi \right) = \mathbb{E} \left[2\|\mathbf{X} - \mathbf{Y}\|_2^\beta - \|\mathbf{X} - \mathbf{X}'\|_2^\beta - \|\mathbf{Y} - \mathbf{Y}'\|_2^\beta \right],$$

where $\beta \in (0, 2)$ and $\|\cdot\|_2$ denotes the ℓ_2 norm. In Bellemare et al., 2017, the above is used to define an algorithm to train generative networks, termed Cramer-GAN.

In Li et al. [2017], the authors proposed to compute the kernel k in Eq. (3) in the main text on a learnable transformation c_ψ , whose weights are trained to maximize the discrepancy. Specifically, that leads to a new discrepancy measure:

$$\max_{\psi} \mathbb{E} \left[k(c_\psi(\mathbf{X}), c_\psi(\mathbf{X}')) - 2k(c_\psi(\mathbf{X}), c_\psi(\mathbf{Y})) + k(c_\psi(\mathbf{Y}), c_\psi(\mathbf{Y}')) \right],$$

which is a meaningful divergence between probability distributions [Li et al., 2017]. In this setting, again people recur to alternating maximization steps over ψ with minimization over ϕ . This, as mentioned in the main text, leads to biased estimates of gradients. However, for MMD-GANs, training is made easier by applying the gradient regularization techniques described in Gulrajani et al. [2017], as shown in Bińkowski et al. [2018].

Notice that, in minimizing Equations (3) in the main text with respect to ϕ , one could ignore the term involving \mathbf{Y}, \mathbf{Y}' ; however, when introducing c_ψ , this cannot be done as that term depends on ψ as well.

In the conditional setting, a natural approach for MMD-GAN is minimizing $\mathbb{E}_{\boldsymbol{\theta} \sim \Pi} [\text{MMD}^2(P^*(\cdot|\boldsymbol{\theta}), P^\phi(\cdot|\boldsymbol{\theta}))]$, as $\text{MMD}^2(P_{\boldsymbol{\theta}, \mathbf{Y}}^*, P_{\boldsymbol{\theta}, \mathbf{Y}}^\phi)$ would require computing kernel over $\Theta \times \mathcal{Y}$.

Notice however how, in estimating $\text{MMD}^2(P^*(\cdot|\boldsymbol{\theta}), P^\phi(\cdot|\boldsymbol{\theta}))$, multiple samples $\mathbf{Y}, \mathbf{Y}' \sim P^*(\cdot|\boldsymbol{\theta})$ are used (see Eq. 3 in the main text), but those are unavailable (empirical samples are of the form in Eq. 4 in the main text); as discussed before, however, $k(\mathbf{Y}, \mathbf{Y}')$ does not depend on ϕ , so that it can be discarded in the minimization process. However, if the data is transformed via c_ψ , $k(c_\psi(\mathbf{Y}), c_\psi(\mathbf{Y}'))$ cannot be dropped anymore, which makes the problem intractable. In Bellemare et al. [2017], this problem is solved by replacing $k(c_\psi(\mathbf{Y}), c_\psi(\mathbf{Y}'))$ with some other tractable terms; however, that approach leads to an ill-defined statistical divergence, as it can be minimized by two distributions which are not the same [Bińkowski et al., 2018].

C Unbiased estimate for gradient of S_T

We discuss here how we can get unbiased gradient estimates for the prequential SR in Eq. (8) in the main text with respect to the parameters of the generative network ϕ . This follows closely what is done in Chapter 4.

Recall now we want to solve:

$$\hat{\phi}_T(\mathbf{y}_{1:T}) := \arg \min_{\phi} S_T(P_{k+l:T}^{\phi}(\cdot|\mathbf{y}_{1:k+l-1}), \mathbf{y}_{k+l:T}),$$

where, for simplicity, we re-define S_T in Eq. (8) in the main text with an additional scaling constant:

$$S_T(P_{k+l:T}^{\phi}(\cdot|\mathbf{y}_{1:k+l-1}), \mathbf{y}_{k+l:T}) := \frac{1}{T-l-k+1} \sum_{t=k}^{T-l} S(P_{t+l}^{\phi}(\cdot|\mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l}). \quad (30)$$

In order to do this, we exploit Stochastic Gradient Descent (SGD), which requires unbiased estimates of $S_T(P_{k+l:T}^{\phi}(\cdot|\mathbf{y}_{1:k+l-1}), \mathbf{y}_{k+l:T})$ (notice we are not talking here of unbiased estimates with respect to the observed sequence $\mathbf{y}_{1:T}$).

Notice how, for all the Scoring Rules used across this work, as well as any weighted sum of those, we can write: $S(P, \mathbf{y}) = \mathbb{E}_{\mathbf{Y}, \mathbf{Y}' \sim P} [g(\mathbf{Y}, \mathbf{Y}', \mathbf{y})]$ for some function g ; namely, the SR is defined through an expectation over (possibly multiple) samples from P . That is the form exploited in Sec. 1.2.2 in the introduction of the present thesis to obtain unbiased SR estimates.

Now, we will use this fact to obtain unbiased estimates for the objective in Eq. (30). For brevity, let us now denote $J(\phi) = S_T(P_{k+l:T}^{\phi}(\cdot|\mathbf{y}_{1:k+l-1}), \mathbf{y}_{k+l:T})$, which we can rewrite as (letting $N = T-l-k+1$ for brevity):

$$\begin{aligned} J(\phi) &= \frac{1}{N} \sum_{t=k}^{T-l} \mathbb{E}_{\mathbf{Y}, \mathbf{Y}' \sim P^{\phi}(\cdot|\mathbf{y}_{t-k+1:t})} [g(\mathbf{Y}, \mathbf{Y}', \mathbf{y}_{t+l})] \\ &= \frac{1}{N} \sum_{t=k}^{T-l} \mathbb{E}_{\mathbf{Z}, \mathbf{Z}' \sim Q} [g(h_{\phi}(\mathbf{Z}; \mathbf{y}_{t-k+1:t}), h_{\phi}(\mathbf{Z}'; \mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l})], \end{aligned}$$

where we used the fact that P^{ϕ} is the distribution induced by a generative network with transformation h_{ϕ} ; this is called the reparametrization trick [Kingma and Welling, 2014]. Now:

$$\begin{aligned} \nabla_{\phi} J(\phi) &= \nabla_{\phi} \frac{1}{N} \sum_{t=k}^{T-l} \mathbb{E}_{\mathbf{Z}, \mathbf{Z}' \sim Q} [g(h_{\phi}(\mathbf{Z}; \mathbf{y}_{t-k+1:t}), h_{\phi}(\mathbf{Z}'; \mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l})] \\ &= \frac{1}{N} \sum_{t=k}^{T-l} \mathbb{E}_{\mathbf{Z}, \mathbf{Z}' \sim Q} [\nabla_{\phi} g(h_{\phi}(\mathbf{Z}; \mathbf{y}_{t-k+1:t}), h_{\phi}(\mathbf{Z}'; \mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l})]. \end{aligned}$$

In the latter equality, the exchange between expectation and gradient is not a trivial step, due to the non-differentiability of functions (such as ReLU) used in h_{ϕ} . Luckily, Theorem 5 in Bińkowski et al. [2018] proved the above step to be valid almost surely with respect to a measure on Φ , under mild conditions on the NN architecture.

We can now easily obtain an unbiased estimate of the above. Additionally, Stochastic Gradient Descent usually consider a small batch of training samples, obtained by considering a random subset $\mathcal{T} \subseteq \{k, k+1, \dots, n-l-1, n-l\}$. Therefore, the following unbiased estimator of $\nabla_{\phi} J(\phi)$ can be obtained, with samples $\mathbf{z}_{t,j} \sim Q, j = 1, \dots, m$:

$$\widehat{\nabla_{\phi} J(\phi)} = \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \frac{1}{m(m-1)} \sum_{\substack{i,j=1 \\ i \neq j}}^m \nabla_{\phi} g(h_{\phi}(\mathbf{z}_{t,i}; \mathbf{y}_{t-k+1:t}), h_{\phi}(\mathbf{z}_{t,j}; \mathbf{y}_{t-k+1:t}), \mathbf{y}_{t+l}).$$

In practice, we then use autodifferentiation libraries (see for instance Paszke et al., 2019) to compute the gradients in the above quantity.

In Algorithm 2, we train a generative network for a single epoch using a scoring rule S for which unbiased estimators can be obtained by using more than one sample from P^{ϕ} . As in Algorithm 1, we use a single pair (θ_i, \mathbf{y}_i) to estimate the gradient.

Algorithm 2 Single epoch generative-SR training.

Require: Parametric map h_ϕ , SR S , learning rate ϵ .

for each training pair $(\boldsymbol{\theta}_i, \mathbf{y}_i)$ **do**

 Sample **multiple** $\mathbf{z}_1, \dots, \mathbf{z}_m$

 Obtain $\hat{\mathbf{x}}_{i,j}^\phi = h_\phi(\mathbf{z}_j, \boldsymbol{\theta}_i)$

 Obtain unbiased estimate $\hat{S}(P^\phi(\cdot|\boldsymbol{\theta}_i), \mathbf{y}_i)$ from $\hat{\mathbf{x}}_{i,j}^\phi$

 Set $\phi \leftarrow \phi - \epsilon \cdot \nabla_\phi \hat{S}(P^\phi(\cdot|\boldsymbol{\theta}_i), \mathbf{y}_i)$

end for

D Performance measures for probabilistic forecast

D.1 Deterministic performance measures

We discuss two measures of performance of a deterministic forecast \hat{y}_{t+l} for a realization y_{t+l} ; across our work, we take \hat{y}_{t+l} to be the mean of the probability distribution $P^\phi(\cdot|\mathbf{y}_{t-k+1:t})$.

D.1.1 Normalized RMSE

We first introduce the Root Mean-Square Error (RMSE) as:

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{t=1}^N (\hat{y}_{t+l} - y_{t+l})^2},$$

where we consider here for simplicity $t = 1, \dots, N$. From the above, we obtain the Normalized RMSE (NRMSE) as:

$$\text{NRMSE} = \frac{\text{RMSE}}{\max_t\{y_{t+l}\} - \min_t\{y_{t+l}\}}.$$

NRMSE = 0 means that $\hat{y}_{t+l} = y_{t+l}$ for all t 's.

D.1.2 Coefficient of determination

The coefficient of determination R^2 measures how much of the variance in $\{y_{t+l}\}_{t=1}^N$ is explained by $\{\hat{y}_{t+l}\}_{t=1}^N$. Specifically, it is given by:

$$R^2 = 1 - \frac{\sum_{t=1}^N (y_{t+l} - \hat{y}_{t+l})^2}{\sum_{t=1}^N (y_{t+l} - \bar{y})^2},$$

where $\bar{y} = \frac{1}{N} \sum_{t=1}^N y_{t+l}$. When $R^2 = 1$, $\hat{y}_{t+l} = y_{t+l}$ for all t 's.

D.2 Calibration error

We review here a measure of calibration of a probabilistic forecast; this measure considers the univariate marginals of the probabilistic forecast distribution $P^\phi(\cdot|\mathbf{y}_{t-k+1:t})$; for component i , let us denote that by $P_{\phi,i}(\cdot|\mathbf{y}_{t-k+1:t})$.

The calibration error [Radev et al., 2020] quantifies how well the credible intervals of the probabilistic forecast $P_{\phi,i}(\cdot|\mathbf{y}_{t-k+1:t})$ match the distribution of the verification $Y_{t+l,i}$. Specifically, let $\alpha^*(i)$ be the proportion of times the verification $y_{t+l,i}$ falls into an α -credible interval of $P_{\phi,i}(\cdot|\mathbf{y}_{t-k+1:t})$, computed over all values of t . If the marginal forecast distribution is perfectly calibrated for component i , $\alpha^*(i) = \alpha$ for all values of $\alpha \in (0, 1)$.

We define therefore the calibration error as the median of $|\alpha^*(i) - \alpha|$ over 100 equally spaced values of $\alpha \in (0, 1)$. Therefore, the calibration error is a value between 0 and 1, where 0 denotes perfect calibration.

In practice, the credible intervals of the predictive are estimated using a set of samples from $P^\phi(\cdot|\mathbf{y}_{t-k+1:t})$.

E Additional experimental details

E.1 Tuning γ in the Gaussian kernel

We define the kernel Score by using the Gaussian kernel:

$$k(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{\|\mathbf{x} - \mathbf{y}\|_2^2}{2\gamma^2}\right).$$

Similar to what was suggested for instance in Park et al., 2016, we set γ in the Gaussian kernel to be the median of the pairwise distances $\|\mathbf{y}_i - \mathbf{y}_j\|$ over all pairs of observations $\mathbf{y}_i, \mathbf{y}_j, i \neq j$ in the validation window.

E.2 Lorenz63 model

E.2.1 Model definition

The Lorenz63 model [Lorenz, 1963] is defined by the following differential equations:

$$\begin{aligned}\frac{dx}{dt} &= \sigma(y - x), \\ \frac{dy}{dt} &= x(\rho - z) - y, \\ \frac{dz}{dt} &= xy - \beta z.\end{aligned}$$

To generate our dataset, we consider $\sigma = 10$, $\rho = 28$, $\beta = 2.667$ and integrate the model using Euler scheme with $dt = 0.01$ starting from $x = 0, y = 1, z = 1.05$. We discard the first 10 time units and integrate the model for additional 9000 time units, during which we record the value of y every $\Delta t = 0.3$ and discard the values of x and z .

E.2.2 Neural Networks architecture

We experiment with a Fully Connected Neural Network (FCNN) and a Recurrent Neural Network (RNN). The latter performs better as it is more suitable to capture the temporal structure in the data; we therefore report results for that in the main text, while results with FCNN are reported in Appendix F.1.2.

FCNN The generative network is a Fully Connected NN with 5 hidden layers which takes as input the concatenation of the values in the observation window and a latent variable \mathbf{Z} with size 1, and outputs a forecast for the next timestep. For the deterministic setting trained with the regression loss, the architecture is analogous, the only difference being that no latent variable \mathbf{Z} is concatenated to the input.

In the adversarial settings, the critic is a fully connected NN again, which takes as input the concatenation of the values in the observation window and the observation/forecast. In the GAN case, the critic outputs a value between 0 and 1 indicating how confident the critic believes that is a fake sample. In the WGAN-GP case, the critic output is a real number.

RNN For the generative network, the observation window is passed through a Gated Recurrent Units (GRU, Cho et al. [2014]) layer with depth 1 and hidden size 8 or 16 (that is a tuning hyperparameter, the choice of which we discuss below). The output of the GRU layer is then concatenated to the latent variable \mathbf{Z} with size 1 and passed through 3 fully connected layers, which output a forecast for the next timestep. For the deterministic setting trained with the regression loss, the architecture is analogous, the only difference being that no latent variable \mathbf{Z} is concatenated to the output of the GRU layer.

In the adversarial settings, the critic has a GRU layer with depth 1 that, analogously to the generative net, processes the information in the past observation window. As above, we try hidden

sizes 8 and 16. Then, the output of the GRU layer and the observation/forecast are concatenated and transformed by 3 fully connected layers. In the GAN case, the critic outputs a value between 0 and 1 indicating how confident the critic believes that is a fake sample. In the WGAN-GP case, the critic output is a real number.

E.2.3 Training hyperparameters

For the experiments on Lorenz63, we considered the batch size to be 1000. For the SR and deterministic approaches, we used Adam optimizer and tested the following learning rate values: 10^{-i} for $i = 1, \dots, 6$ for the SR methods and 10^{-i-1} and $3 \cdot 10^{-i-1}$ for $i = 1, \dots, 3$ for regression. For the RNN setting, we fix the GRU hidden size to 8. We report then the performance achieved with the learning rate yielding lower loss on the validation set, which is indicated in Table 3.

Table 3: Optimal learning rate values for SR and regression (deterministic) approaches for Lorenz63.

	Energy	Kernel	Energy-Kernel	Regression
FCNN	0.01	0.01	0.01	0.003
RNN	0.01	0.001	0.01	0.001

For the GAN and WGAN-GP approach, we used Adam optimizer and we tested the following learning rate values for both critic and generative network: 10^{-i} and $3 \cdot 10^{-i}$ for $i = 1, \dots, 7$. In total, those are 14 learning rate values. For the FCNN case, we fixed additionally the number of critic training steps each generator training steps to 1 for GAN and 5 for WGAN-GP. Overall, therefore, we run $14^2 = 196$ experiments for GAN and WGAN-GP respectively. For the RNN, we hidden size 8 and 16; further, we experiment with 4 number of critic training steps for WGAN-GP (1, 3, 5, 10), in order to have the best possible results to compare with our SR methods, while we left the number of critic training steps to 1 for GAN. Overall, therefore, we had $2 \cdot 14^2 = 392$ experiments for GAN and $2 \cdot 4 \cdot 14^2 = 1568$ for WGAN-GP; notice the extremely larger number number of experiments for the adversarial approaches with respect to SR ones, which highlights an advantage of our approach. We stress that such a number of trials could be possible only for the low-dimensional setting of the Lorenz63 and Lorenz96 models, in which training is cheap, but not in real-life applications.

Additionally, the adversarial approaches do not allow to select hyperparameters according to loss on a validation set, as the generator loss depends on the current state of the discriminator (i.e., there is no absolute loss scale). Therefore, we report results for 3 different configurations for GAN and WGAN-GP, maximizing either deterministic performance (1) or calibration (2), or striking the best balance between these two (3). The resulting learning rates are in Table 4 below.

Table 4: Optimal hyperparameter values for adversarial approaches for Lorenz63 model.

		GAN (1)	GAN (2)	GAN (3)	WGAN-GP (1)	WGAN-GP (2)	WGAN-GP (3)
FCNN	Generator l.r.	0.03	0.00003	0.0001	0.0003	0.0003	0.003
	Critic l.r.	0.01	0.03	0.0003	0.01	0.003	0.01
RNN	Generator l.r.	0.0003	0.001	0.0001	0.003	0.0003	0.0003
	Critic l.r.	0.03	0.01	0.001	0.001	0.1	0.03
	GRU hidden size	16	8	8	8	8	8
	Critic training steps	1	1	1	5	5	5

E.3 Lorenz96 model

E.3.1 Model definition

The Lorenz96 model [Lorenz, 1996] is a toy representation of atmospheric behavior containing slow (\mathbf{x}) and fast (\mathbf{y}) evolving variables.

Specifically, the evolution of the variables is determined by the following differential equations:

$$\begin{aligned}\frac{dx_k}{dt} &= -x_{k-1}(x_{k-2} - x_{k+1}) - x_k + F - \frac{hc}{b} \sum_{j=J(k-1)+1}^{kJ} y_j; \\ \frac{dy_j}{dt} &= -cby_{j+1}(y_{j+2} - y_{j-1}) - cy_j + \frac{hc}{b} X_{\text{int}[(j-1)/J]+1}.\end{aligned}$$

where $k = 1, \dots, K$, and $j = 1, \dots, JK$, and cyclic boundary conditions are assumed, so that index $k = K + 1$ corresponds to $k = 1$ and similarly for j . The above equations connect the fast and slow variables in a cyclic way. Additionally, x_k reciprocally depends on J fast variables.

Following Gagne et al., 2020, we take $K = 8$, $J = 32$, $h = 1$, $b = 10$, $c = 10$ and $F = 20$. We then integrate the above equations with RK4 scheme with $dt = 0.001$, starting from $x_k = y_j = 0$ for $k = 2, \dots, K$ and $j = 2, \dots, JK$ and $x_1 = y_1 = 1$. We discard the first 2 time units and record the values of \mathbf{x} every $\Delta t = 0.2$ (which corresponding to roughly one atmospheric day with respect to predictability, Gagne et al., 2020). We do this for additional 4000 time units, and split the resulting dataset in training, validation and test according to the proportions 60%, 20% and 20%.

E.3.2 Neural Networks architecture

We experiment with a Fully Connected Neural Network (FCNN) and a Recurrent Neural Network (RNN). The latter performs better as it is more suitable to capture the temporal structure in the data; we therefore report results for that in the main text, while results with FCNN are reported in Appendix F.2.2.

FCNN The generative network is a Fully Connected NN with 5 hidden layers which takes as input the concatenation of the values in the observation window (flattened to an $8 \cdot 10 = 80$ dimensional vector) and a latent variable \mathbf{Z} with size 8, and outputs a forecast for the next timestep. For the deterministic setting trained with the regression loss, the architecture is analogous, the only difference being that no latent variable \mathbf{Z} is concatenated to the input.

In the adversarial settings, the critic is a fully connected NN again, which takes as input the concatenation of the flattened values in the observation window and the observation/forecast. In the GAN case, the critic outputs a value between 0 and 1 indicating how confident the critic believes that is a fake sample. In the WGAN-GP case, the critic output is a real number.

RNN For the generative network, the observation window is passed through a Gated Recurrent Units (GRU, Cho et al. [2014]) layer with depth 1 and hidden size 32 or 64 (that is a tuning hyperparameter, the choice of which we discuss below). The output of the GRU layer is then concatenated to the latent variable \mathbf{Z} with size 1 and passed through 3 fully connected layers, which output a forecast for the next timestep. For the deterministic setting trained with the regression loss, the architecture is analogous, the only difference being that no latent variable \mathbf{Z} is concatenated to the output of the GRU layer.

In the adversarial settings, the critic has a GRU layer with depth 1 that, analogously to the generative net, processes the information in the past observation window. As above, we try hidden sizes 8 and 16. Then, the output of the GRU layer and the observation/forecast are concatenated and transformed by 3 fully connected layers. In the GAN case, the critic outputs a value between 0 and 1 indicating how confident the critic believes that is a fake sample. In the WGAN-GP case, the critic output is a real number.

E.3.3 Training hyperparameters

For the experiments on Lorenz96, we considered the batch size to be 1000. For the SR and deterministic approaches, we used Adam optimizer and tested the following learning rate values: 10^{-i} for $i = 1, \dots, 6$ for the SR methods and 10^{-i-1} and $3 \cdot 10^{-i-1}$ for $i = 1, \dots, 3$ for regression. For the RNN setting, we fix the GRU hidden size to 32. We report then the performance achieved with the learning rate yielding lower loss on the validation set, which is indicated in Table 5.

Table 5: Optimal learning rate values for SR and regression (deterministic) approaches for Lorenz96.

	Energy	Kernel	Energy-Kernel	Regression
FCNN	0.001	0.001	0.001	0.0003
RNN	0.01	0.001	0.001	0.003

For the GAN and WGAN-GP approach, we used Adam optimizer and we tested the following learning rate values for both critic and generative network: 10^{-i} and $3 \cdot 10^{-i}$ for $i = 1, \dots, 7$. In total, those are 14 learning rate values. For the FCNN case, we fixed additionally the number of critic training steps each generator training steps to 1 for GAN and 5 for WGAN-GP. Overall, therefore, we run $14^2 = 196$ experiments for GAN and WGAN-GP respectively. For the RNN, we hidden size 8 and 16; further, we experiment with 4 number of critic training steps for WGAN-GP (1, 3, 5, 10), in order to have the best possible results to compare with our SR methods, while we left the number of critic training steps to 1 for GAN. Overall, therefore, we had $2 \cdot 14^2 = 392$ experiments for GAN and $2 \cdot 4 \cdot 14^2 = 1568$ for WGAN-GP; notice the extremely larger number number of experiments for the adversarial approaches with respect to SR ones, which highlights an advantage of our approach. We stress that such a number of trials could be possible only for the low-dimensional setting of the Lorenz63 and Lorenz96 models, in which training is cheap, but not in real-life applications.

Additionally, the adversarial approaches do not allow to select hyperparameters according to loss on a validation set, as the generator loss depends on the current state of the discriminator (i.e., there is no absolute loss scale). Therefore, we report results for 3 different configurations for GAN and WGAN-GP, maximizing either deterministic performance (1) or calibration (2), or striking the best balance between these two (3). The resulting learning rates are in Table 6 below. Notice that, for GAN, there was no configuration leading to intermediate performance between (1) and (2), so that the column for (3) is left empty.

Table 6: Optimal hyperparameter values for adversarial approaches for Lorenz96 model.

		GAN (1)	GAN (2)	GAN (3)	WGAN-GP (1)	WGAN-GP (2)	WGAN-GP (3)
FCNN	Generator l.r.	0.01	0.0001	-	0.0001	0.0001	0.0001
	Critic l.r.	0.001	0.001	-	0.003	0.03	0.01
RNN	Generator l.r.	0.01	0.0001	0.0001	0.001	0.00003	0.0001
	Critic l.r.	0.001	0.003	0.001	0.001	0.1	0.01
	GRU hidden size	64	32	64	64	64	64
	Critic training steps	1	1	1	10	1	5

E.4 WeatherBench dataset

E.4.1 Variogram Score

For the Variogram Score, we use a weight matrix which is inversely proportional to the Haversine distance, which measures the angular distance between two points on the surface of a sphere. Specifically,

by denoting the longitude and latitude (in radians) of component i of \mathbf{y} as $\text{lon}_i, \text{lat}_i$, the Haversine distance is defined as:

$$d_{ij} = 2 \arcsin \left[\sqrt{\sin^2((\text{lat}_i - \text{lat}_j)/2) + \cos(\text{lat}_i) \cos(\text{lat}_j) \sin^2((\text{lon}_i - \text{lon}_j)/2)} \right]$$

The physical distance along the sphere can be computed by multiplying the above by Earth’s radius (approximately 6371 km). However, that is just a scaling constant, therefore we ignore it in defining the variogram, which we take to be $w_{ij} = 1/d_{ij}$.

E.4.2 Choice of weights for summed scores

In the summed Scores (Energy-Variogram, Kernel-Variogram, Energy-Kernel and Patched Energy Score), we need to select the weights for the two addends. Notice that, in the Patched Energy Score, we consider the Energy Score computed on the full data to be the first addend, and the sum of the Energy Scores computed on each patch to be the second addend.

We fix the weights such that the two addends have roughly the same magnitude. This results, for the Energy-Variogram, Kernel-Variogram, Energy-Kernel, in the choices reported in the following Table:

Table 7: Weights for summed Scores.

	Energy-Kernel	Energy-Variogram	Kernel-Variogram
α_1	1/70	1	1
α_2	1	$6.94 \cdot 10^{-7}$	$1.3 \cdot 10^{-8}$

For the Patched Energy Score, we use the following two setups in our experiments:

- Patches of size 16 separated by 8 grid points: this leads to 32 patches. As the Energy Score scales as the data dimensionality, each of the $16 \times 16 = 256$ patches has relative magnitude with respect to Energy Score computed on the full WeatherBench grid $256/2048 = 0.125$, where $32 \times 64 = 2048$ is the size of the WeatherBench grid. However, we sum the Score for each of the 32 patches, which leads to a quantity with magnitude 4 times the one of the overall Energy Score.
- Patches of size 8 separated by 4 grid points: this leads to 128 patches. Following the argument above, each $8 \times 8 = 64$ patch gives a Score with relative magnitude $64/2048 = 0.03125$. As there are 128 patches, again the cumulative patched score has magnitude 4 times the overall one.

In both cases, we leave therefore $\alpha_1 = \alpha_2 = 1$, as the patched and overall components are already of similar magnitude (they just differ by a factor 4).

E.4.3 Neural Networks architecture

For the generative network, we use a U-NET architecture [Olaf et al., 2015], which is an encoder-decoder structure, where each subsequent layer of the encoder outputs a downscaled latent representation of the input variables. The final output of the encoder is passed to a bottleneck layer, which performs no up/down scaling. The output of this bottleneck layer is then passed to the decoder. Conversely to the encoder, each subsequent layer of the decoder outputs an upscaled latent representation of the bottleneck layer output. Additionally, skip connections allow information to pass directly between layers of the encoder and decoder at the same scale; in this way, both large scale structures and high-frequency information contributes to the output. The latent variable \mathbf{Z} is summed to the latent representation in the bottleneck layer. Figure 5 gives a graphical representation of the UNet. For the deterministic setting trained with the regression loss, the architecture is analogous, the only difference being that no latent variable \mathbf{Z} is summed to the latent representation.

In the adversarial setups, we use the PatchGAN critic suggested in Isola et al. [2017]. Specifically, this is a convolutional network which considers separate patches of the input image and outputs a

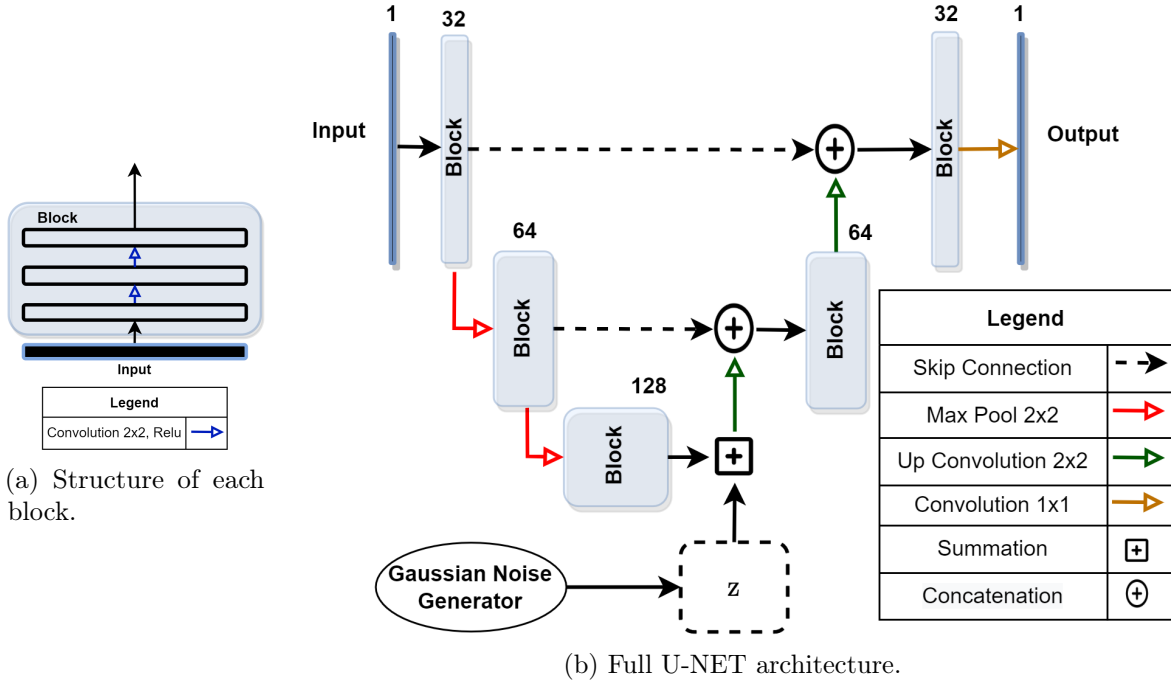


Figure 5: U-NET architecture.

numerical value for each patch, corresponding, in the original GAN setting of Goodfellow et al. [2014], to the confidence with which the critic believes that patch is real, in contrast to generated from the generative network. The GAN or WGAN loss is then computed for each of the output values and averaged.

The PatchGAN critic employs some Batch Normalization layers; however, these cannot be used when the gradient penalization strategy of WGAN-GP is used [Gulrajani et al., 2017]. Therefore, as suggested in Gulrajani et al. [2017], we replace the Batch Normalization layers with Layer Normalization.

As before, in the GAN case, the critic outputs a value between 0 and 1 indicating how confident the critic believes that is a fake sample. In the WGAN-GP case, the critic output is a real number.

E.4.4 Training hyperparameters

For the SR approaches for the WeatherBench dataset, we considered the batch size to be 128 for all experiments, except for those on the Energy-Variogram and Kernel-Variogram score, which resulted in GPU memory overflow with that batch size (in fact, computing the Variogram Score is an operation requiring quadratic memory with respect to data size); for these two, we fixed therefore the batch size to be 48. We used Adam optimizer and tested the following learning rate values 10^{-i} for $i = 1, \dots, 6$. We report then the performance achieved with the learning rate yielding lower loss on the validation set in Table 8.

For the deterministic network trained via regression, we test learning rule values 10^{-i-1} for $i = 1, \dots, 4$; additionally, we use an exponential learning rate scheduler which reduces the learning rate by multiplying it by a factor γ every 10 training epochs. We also use a ℓ_2 weight regularization with weight λ . We try different values of these parameters in conjunction with the learning rate values; the ones with which best validation loss is obtained are $\gamma = 0.8$ and $\lambda = 0.001$. The best learning rate value is reported in Table 8. Notice that the same learning rate value was optimal for the full (non-patched) regression loss and for the patched loss in both configurations.

For the GAN and WGAN-GP approach, we used Adam optimizer and we tested the following learning rate values for both critic and generative network: 10^{-i} , $i = 1, \dots, 7$. In total, those are 7 learning rate values, which result in $7^2 = 49$ experiments. Notice additionally that the adversarial approaches does not allow to select hyperparameters according to loss on a validation set, as the generator loss depends on the current state of the discriminator (i.e., there is no absolute loss scale). Additionally, the adversarial approaches do not allow to select hyperparameters according to loss on a

Table 8: Optimal learning rate values for the SR and regression (deterministic) approaches for Weather-Bench.

	Regression	Energy	kernel	Energy-Kernel	Energy-Variogram
Learning rate	0.01	0.0001	0.0001	0.0001	10^{-5}
	Kernel-Variogram	Patched Energy (8)	Patched Energy (16)		
Learning rate	10^{-5}	10^{-5}	10^{-5}		

validation set, as the generator loss depends on the current state of the discriminator (i.e., there is no absolute loss scale). Therefore, we report results for 3 different configurations for GAN, maximizing either deterministic performance (1) or calibration (2), or striking the best balance between these two (3). For WGAN-GP, a single configuration maximized both calibration and deterministic performance, so that we report that one. The resulting learning rates are in Table 9 below.

Table 9: Optimal hyperparameter values for adversarial approaches for WeatherBench.

	GAN (1)	GAN (2)	GAN (3)	WGAN-GP
Generator learning rate	0.001	10^{-6}	10^{-5}	10^{-5}
Critic learning rate	0.0001	0.0001	10^{-5}	0.01

F Additional experimental results

F.1 Lorenz63 model

F.1.1 Additional results with RNN

We report here additional results with the RNN used in the main text of the paper. Figure 6 contains separate plots for all methods showing forecasts and realization for a portion of the test set (the same used in Section 5.1 in the main text).

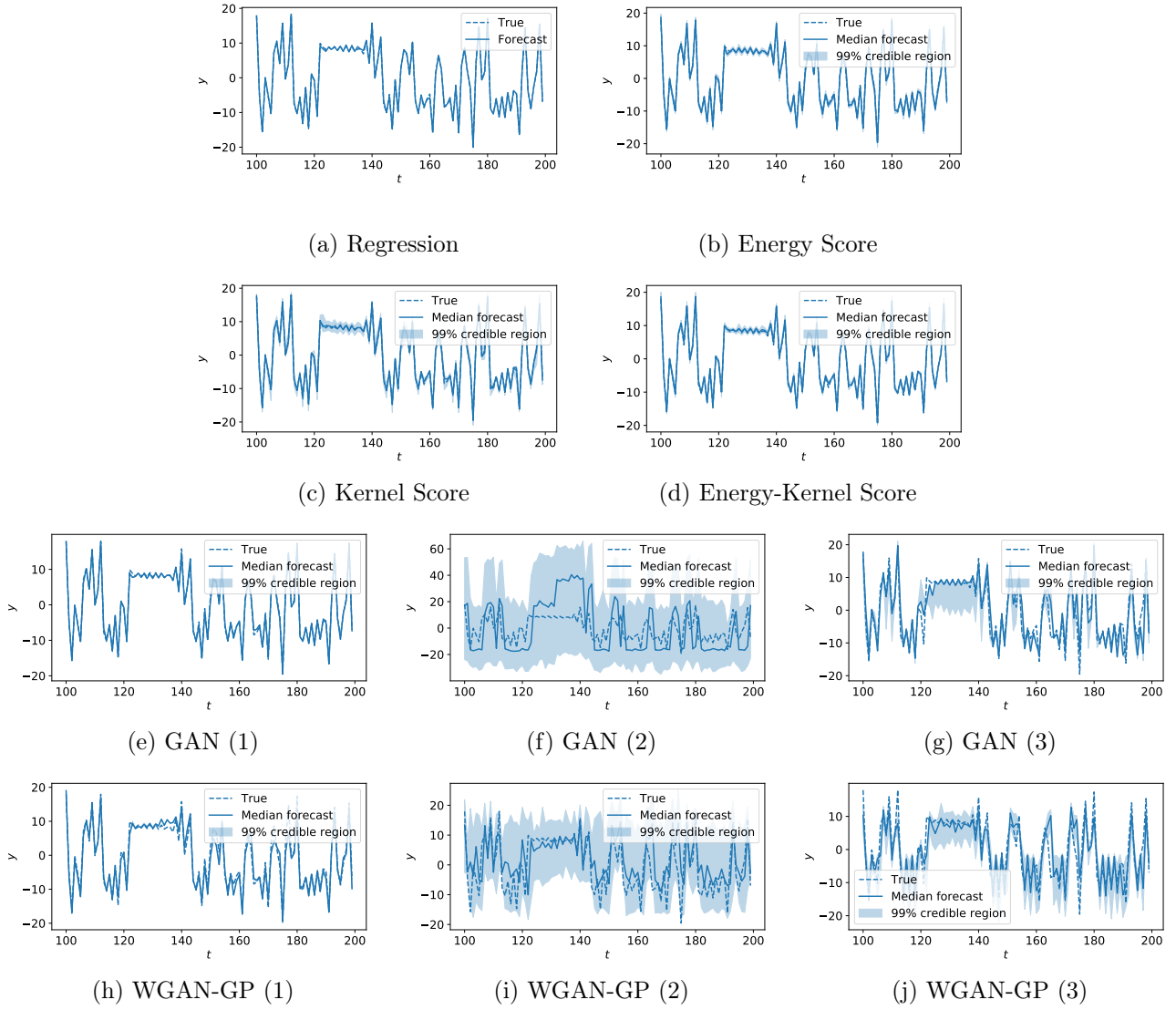


Figure 6: Results for the Lorenz63 model with all considered methods, for RNN. The figures show observations, median forecast and 99% credible interval for a portion of the test set. For each time-step, forecasts are obtained using the previous observation window.

F.1.2 Results with FCNN

We report here results with the Fully Connected NN (FCNN). Table 10 contains performance metrics for the different methods. Figure 7 contains separate plots for all methods showing forecasts and realization for a portion of the test set. Overall, performances are worse than with RNN.

Table 10: Performance measures for forecasts obtained with the different methods, on the test set for the Lorenz63 dataset, with FCNN.

	Cal. error ↓	NRMSE ↓	R ² ↑
Regression	-	0.0297	0.9682
Energy	0.0510	0.0293	0.9692
Kernel	0.1220	0.0155	0.9913
Energy-Kernel	0.0800	0.0188	0.9873
GAN (1)	0.4930	0.0651	0.8475
GAN (2)	0.3710	0.1890	-0.2857
GAN (3)	0.4580	0.1201	0.4805
WGAN-GP (1)	0.4410	0.1018	0.6269
WGAN-GP (2)	0.3260	0.1164	0.5120
WGAN-GP (3)	0.4330	0.1083	0.5776

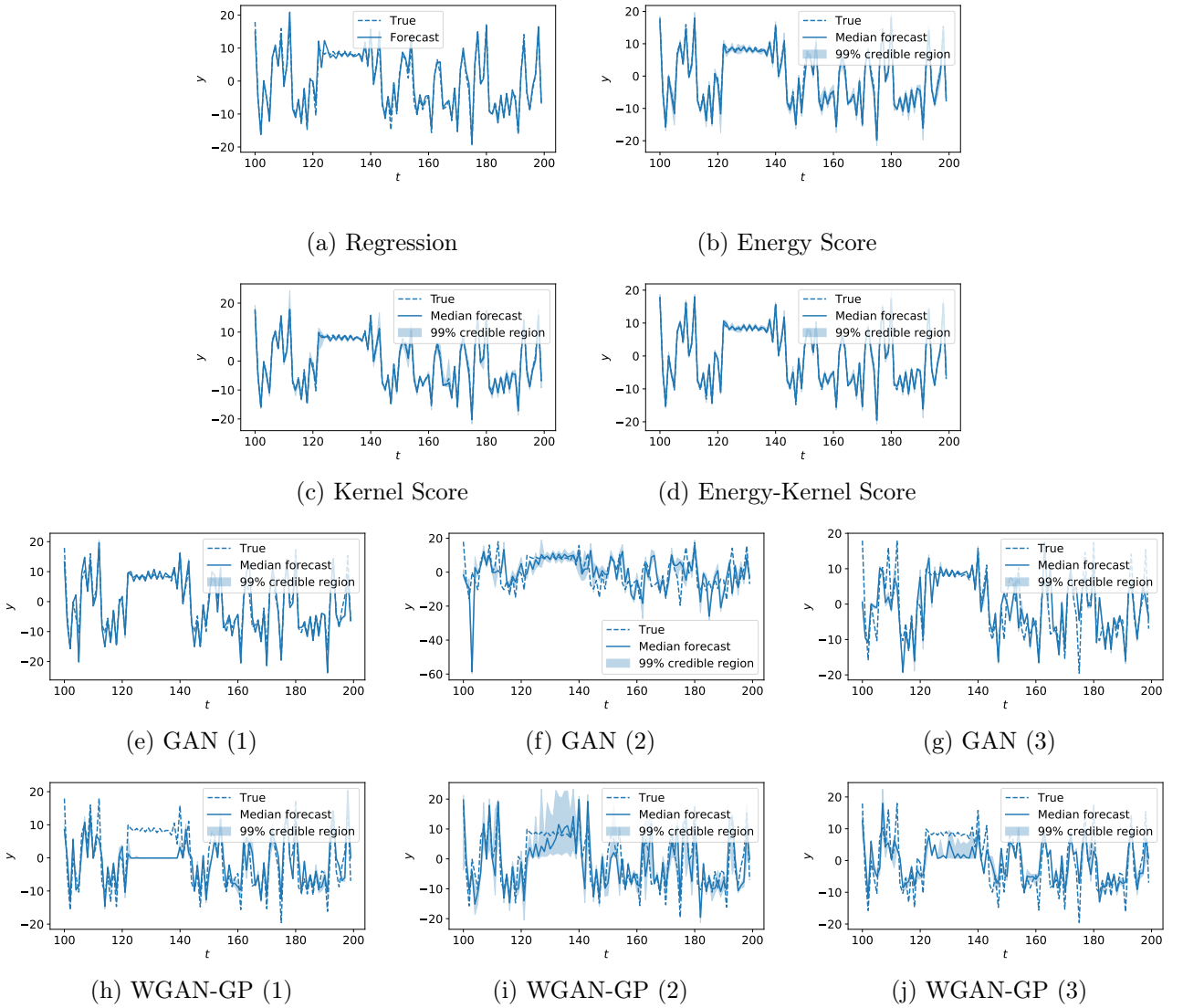


Figure 7: Results for the Lorenz63 model with all considered methods, for FCNN. The figures show observations, median forecast and 99% credible interval for a portion of the test set. For each time-step, forecasts are obtained using the previous observation window.

F.2 Lorenz96 model

F.2.1 Additional results with RNN

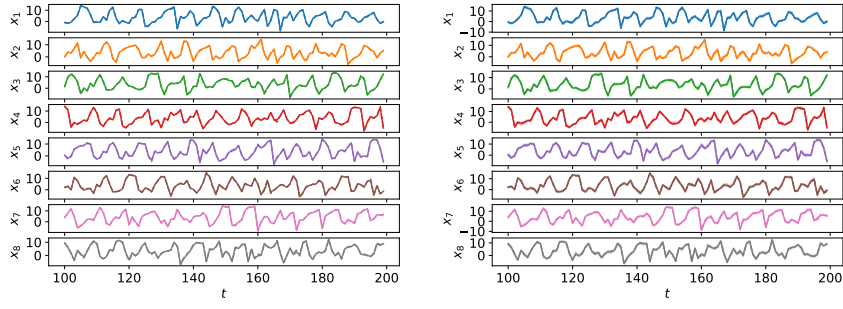
We report here additional results with the RNN used in the main text of the paper.

Table 11 reports the average and standard deviation of the different performance measures computed across the different data components. It contains the same results as Table 1 in the main text, where however the standard deviation was not reported.

Figure 8 contains separate plots for all methods showing forecasts and realization for a portion of the test set (the same used in Section 5.1 in the main text).

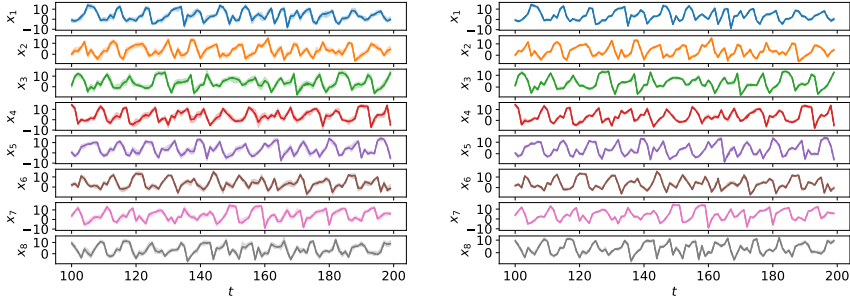
Table 11: Average and standard deviation of performance measures for forecasts obtained with the different methods, on the test set for the Lorenz96 dataset, with the RNN. Metrics are computed on each data component individually; then, the average and standard deviation is computed.

	Cal. error ↓	NRMSE ↓	R ² ↑
Regression	-	0.0198 ± 0.0006	0.9905 ± 0.0006
Energy	0.0205 ± 0.0176	0.0166 ± 0.0014	0.9933 ± 0.0012
Kernel	0.2196 ± 0.0123	0.0164 ± 0.0003	0.9935 ± 0.0003
Energy-Kernel	0.0104 ± 0.0060	0.0173 ± 0.0004	0.9928 ± 0.0004
GAN (1)	0.4644 ± 0.0062	0.0354 ± 0.0026	0.9696 ± 0.0044
GAN (2)	0.2671 ± 0.0559	0.1500 ± 0.0090	0.4537 ± 0.0619
GAN (3)	0.3700 ± 0.0369	0.0763 ± 0.0030	0.8590 ± 0.0099
WGAN-GP (1)	0.4134 ± 0.0051	0.0330 ± 0.0007	0.9736 ± 0.0009
WGAN-GP (2)	0.0565 ± 0.0339	0.1081 ± 0.0037	0.7165 ± 0.0200
WGAN-GP (3)	0.1648 ± 0.0444	0.0786 ± 0.0041	0.8502 ± 0.0149



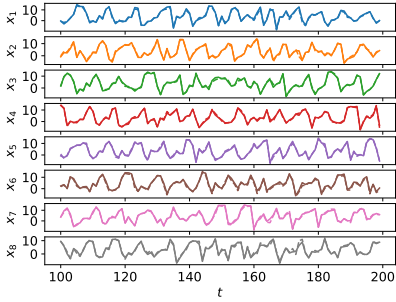
(a) Regression

(b) Energy Score.

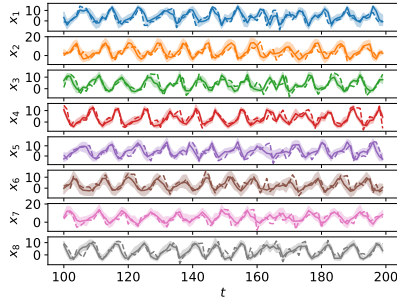


(c) Kernel Score.

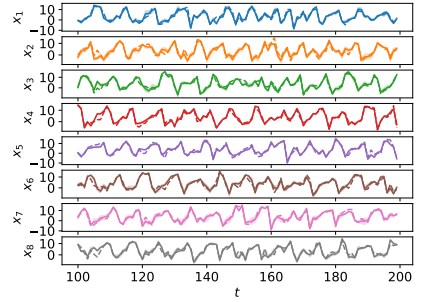
(d) Energy-Kernel Score



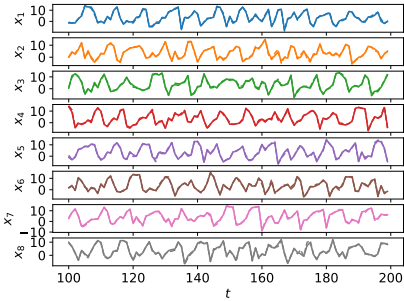
(e) GAN (1)



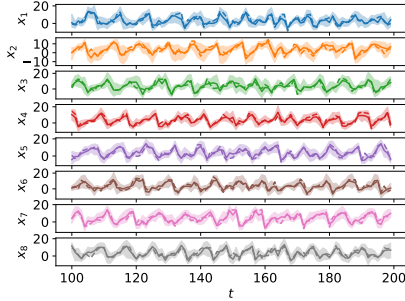
(f) GAN (2)



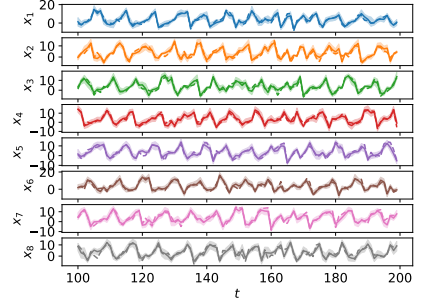
(g) GAN (3)



(h) WGAN-GP (1)



(i) WGAN-GP (2)



(j) WGAN-GP (3)

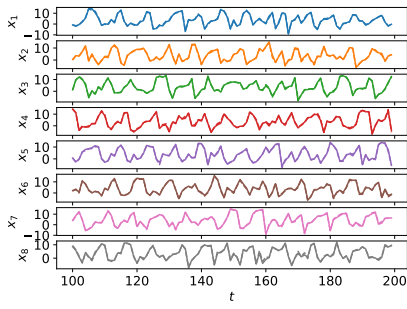
Figure 8: Results for the Lorenz96 model with all considered methods for RNN. Panels show observations (dashed line), median forecast (solid line) and 99% credible interval (shaded region) for a portion of the test set. That is done for all 8 components of \mathbf{x} . For each time-step, forecasts are obtained using the previous observation window.

F.2.2 Results with FCNN

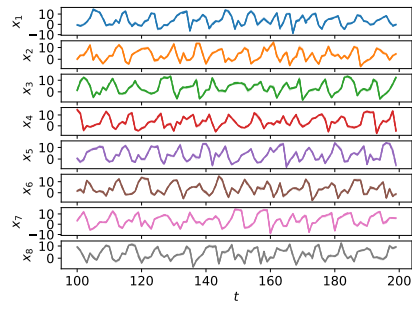
We report here results with the Fully Connected NN (FCNN). In Table 12 below, the average and standard deviation of the different performance measures are computed across the different data components, for the various methods. Figure 7 contains separate plots for all methods showing forecasts and realization for a portion of the test set, as done before for FCNN. Overall, performances are worse than with RNN.

Table 12: Average and standard deviation of performance measures for forecasts obtained with the different methods, on the test set for the Lorenz96 dataset, with FCNN. Metrics are computed on each data component individually; then, the average and standard deviation is computed.

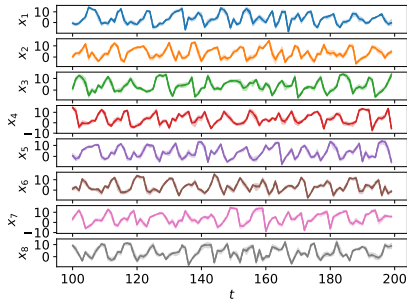
	Cal. error ↓	NRMSE ↓	R ² ↑
Regression	-	0.0243 ± 0.0007	0.9857 ± 0.0008
Energy	0.1230 ± 0.0366	0.0176 ± 0.0015	0.9925 ± 0.0012
Kernel	0.1179 ± 0.0244	0.0175 ± 0.0009	0.9926 ± 0.0008
Energy-Kernel	0.1560 ± 0.0172	0.0145 ± 0.0005	0.9949 ± 0.0003
GAN (1)	0.4875 ± 0.0025	0.0873 ± 0.0039	0.8151 ± 0.0167
GAN (2)	0.3775 ± 0.0125	0.1113 ± 0.0079	0.6994 ± 0.0372
WGAN-GP (1)	0.2054 ± 0.0218	0.0762 ± 0.0017	0.8593 ± 0.0040
WGAN-GP (2)	0.1678 ± 0.0181	0.0928 ± 0.0025	0.7913 ± 0.0091
WGAN-GP (3)	0.2016 ± 0.0208	0.0823 ± 0.0022	0.8361 ± 0.0075



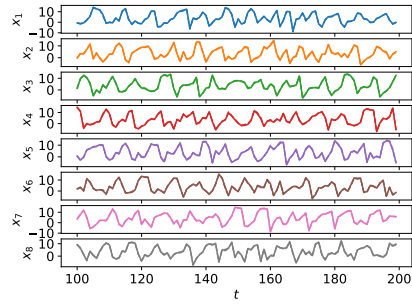
(a) Regression



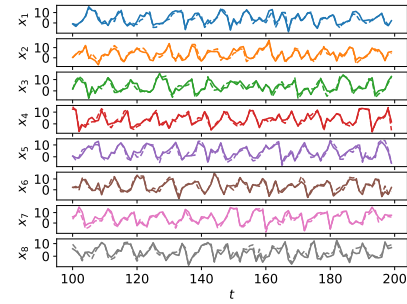
(b) Energy Score.



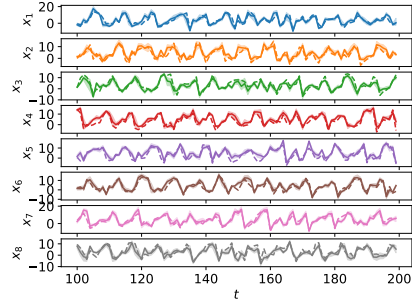
(c) Kernel Score.



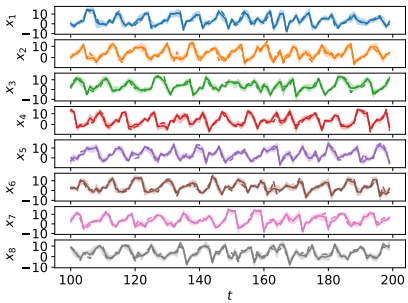
(d) Energy-Kernel Score



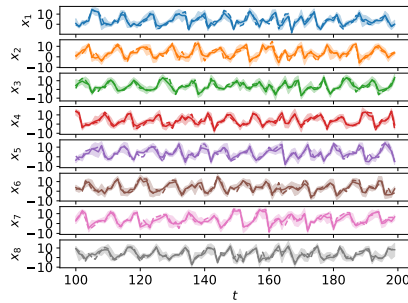
(e) GAN (1)



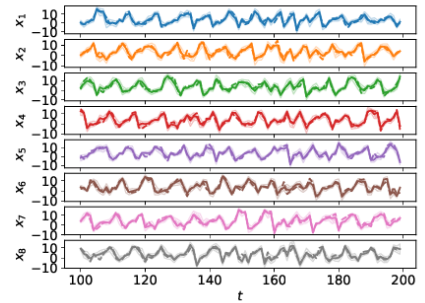
(f) GAN (2)



(g) WGAN-GP (1)



(h) WGAN-GP (2)



(i) WGAN-GP (3)

Figure 9: Results for the Lorenz96 model with all considered methods for FCNN. Panels show observations (dashed line), median forecast (solid line) and 99% credible interval (shaded region) for a portion of the test set. That is done for all 8 components of \mathbf{x} . For each time-step, forecasts are obtained using the previous observation window.

F.3 WeatherBench dataset

F.3.1 Standard deviation of performance measures

In Table 13 below, the average and standard deviation of the different performance measures are computed across the different data components.

Table 13: Average and standard deviation of performance measures for forecasts obtained with the different methods, on the test section of the WeatherBench dataset. Metrics are computed on each data component individually; then, the average and standard deviation is computed.

	Cal. error ↓	NRMSE ↓	R ² ↑
Regression	-	0.1162 ± 0.0256	0.5300 ± 0.2559
Patched Regression, 8	-	0.1147 ± 0.0238	0.5459 ± 0.2297
Patched Regression, 16	-	0.1144 ± 0.0227	0.5509 ± 0.2188
Energy	0.0863 ± 0.0407	0.1208 ± 0.0256	0.4968 ± 0.2596
Kernel	0.0797 ± 0.0455	0.1200 ± 0.0226	0.5097 ± 0.2226
Energy-Kernel	0.0794 ± 0.0433	0.1194 ± 0.0226	0.5150 ± 0.2225
Energy-Variogram	0.0899 ± 0.0541	0.1192 ± 0.0220	0.5177 ± 0.2180
Kernel-Variogram	0.1704 ± 0.0607	0.1203 ± 0.0238	0.5050 ± 0.2399
Patched Energy, 8	0.0550 ± 0.0348	0.1189 ± 0.0209	0.5217 ± 0.2064
Patched Energy, 16	0.0690 ± 0.0478	0.1186 ± 0.0208	0.5248 ± 0.2034
GAN (1)	0.4845 ± 0.0089	0.1573 ± 0.0391	0.1418 ± 0.5267
GAN (2)	0.3130 ± 0.1143	0.2487 ± 0.2248	-2.7970 ± 17.1346
GAN (3)	0.3625 ± 0.0545	0.1693 ± 0.0494	-0.0117 ± 0.8348
WGAN-GP	0.1009 ± 0.0679	0.1302 ± 0.0214	0.4340 ± 0.2271

F.3.2 Number of generator simulations for the SR methods

We study here the effect of using different numbers of simulations from the generative network for each input (i.e., how many forecasts the generative network provides) during training. Recall in fact how the Energy and Kernel Score need multiple samples to be estimated (see Sec. 1.2.2 in the introduction of the present thesis).

Specifically, we consider the WeatherBench dataset and the Energy Score, with learning rate 0.0001, which was found to be the optimal value when using 10 generator simulations (Appendix E.4.4). We report the measures used in the main text in Table 14. Notice how good performance is achieved when using as little as 2 or 3 simulations.

Table 14: Performance on test set of probabilistic forecasts obtained by training with the Energy Score, with different numbers of generator simulations, for the WeatherBench dataset.

	Cal. error ↓	NRMSE ↓	R ² ↑
2	0.0625 ± 0.0340	0.1211 ± 0.0258	0.4935 ± 0.2656
3	0.0701 ± 0.0342	0.1176 ± 0.0208	0.5338 ± 0.1961
5	0.0727 ± 0.0348	0.1164 ± 0.0198	0.5446 ± 0.1842
10	0.0863 ± 0.0407	0.1208 ± 0.0256	0.4968 ± 0.2596
20	0.0738 ± 0.0336	0.1179 ± 0.0206	0.5329 ± 0.1925
30	0.0738 ± 0.0350	0.1169 ± 0.0202	0.5407 ± 0.1864
50	0.0749 ± 0.0356	0.1172 ± 0.0203	0.5379 ± 0.1889

F.3.3 Computational cost and early stopping

In Table 15, we report the computational cost and the early stopping achieved by the methods presented in the main text. All experiments are run on a Tesla v100 GPU, and methods are run for a maximum of 1000 epochs. We use early stopping for the SR methods, but not for GAN and WGAN-GP, for which early stopping is not possible. Recall that the methods with the Variogram Score used training batch size 48, while all others used 128; this fact contributes to the larger computational time for both the Energy-Variogram and Kernel-Variogram Scores.

Table 15: Per-epoch and total computational cost, in seconds, for the different methods reported in the main text. We also report epoch at which early stopping occurred.

	Per-epoch Computational cost	Early stopping at epoch	Total computational cost
Regression	8.45	250	2112
Patched Regression, 8	8.65	200	1729
Patched Regression, 16	8.5	250	2122
Energy	54.2	100	5417
Kernel	53.3	100	5329
Energy-Kernel	55.4	100	5542
Energy-Variogram	97.38	250	24346
Kernel-Variogram	95.52	250	24393
Patched Energy, 8	56.71	400	22682
Patched Energy, 16	54.93	450	24717
GAN (1)	8.36	-	8357
GAN (2)	8.37	-	8373
GAN (3)	8.33	-	8326
WGAN-GP	7.00	-	7000

Additionally, recall that, in order to achieve the performance reported in the main text, we tried 49 learning rate values for GAN and WGAN-GP, but only 6 for the SR methods. Therefore, the total computing time for GAN and WGAN-GP is the one below multiplied by 49, with respect to 6 for the SR methods. Under that perspective, even the total computing time for Energy-Variogram and Kernel-Variogram Scores is smaller than the one for the adversarial methods. For instance, if we consider Energy-Variogram, do not use early stopping and run for 1000 epochs 6 times, we get a total of $97.38 \times 6000 = 584280$ seconds. For WGAN-GP, we obtain instead $7.00 \times 49 \times 1000 = 343000$ seconds, which is only slightly smaller than the grand total for Energy-Variogram. For the latter, this number does not take into account early stopping which, as can be seen from Table 15, reduces largely the total number of epochs required for training.

Additionally, we highlight how, in the results used for Table 15, the SR methods were trained using 10 simulations from the generator for each observation window (i.e., 10 forecasts). In Appendix F.3.2, we studied the effect of the number of simulations used on training, highlighted how the performance is good with as little as 2 or 3 simulations. This greatly reduces the computational cost; we report that in Table 16; for this study, the Energy Score was used.

Table 16: Per-epoch and total computational cost, in seconds, for the Energy Score for different numbers of generator simulations. We also report epoch at which early stopping occurred.

	Per-epoch Computational cost	Early stopping at epoch	Total computational cost
2	13.7	100	1371
3	19.1	100	1913
5	29.6	100	2967
10	54.2	100	5417
20	107.0	100	10700
30	159.2	100	15916
50	258.7	100	25865

F.3.4 Maps for a chosen date

Figure 10 reports realization and prediction with the deterministic regression methods; instead, in Figures 11, 12, and 13 we report realization and 5 different forecasts obtained with all probabilistic methods discussed in the main text (Section 5.2). In Figures 14, 15, and 16, we show the deviation of the same realizations and forecasts from the forecast mean (obtained empirically from 100 forecasts). If the forecast distribution is calibrated, the realization should look similar to the forecasts themselves. You can see how this is roughly the case for the best performing SRs (as for instance the Patched Energy Score in Figure 15) as well as for WGAN-GP, but it clearly not the case for GAN (Figure 16).

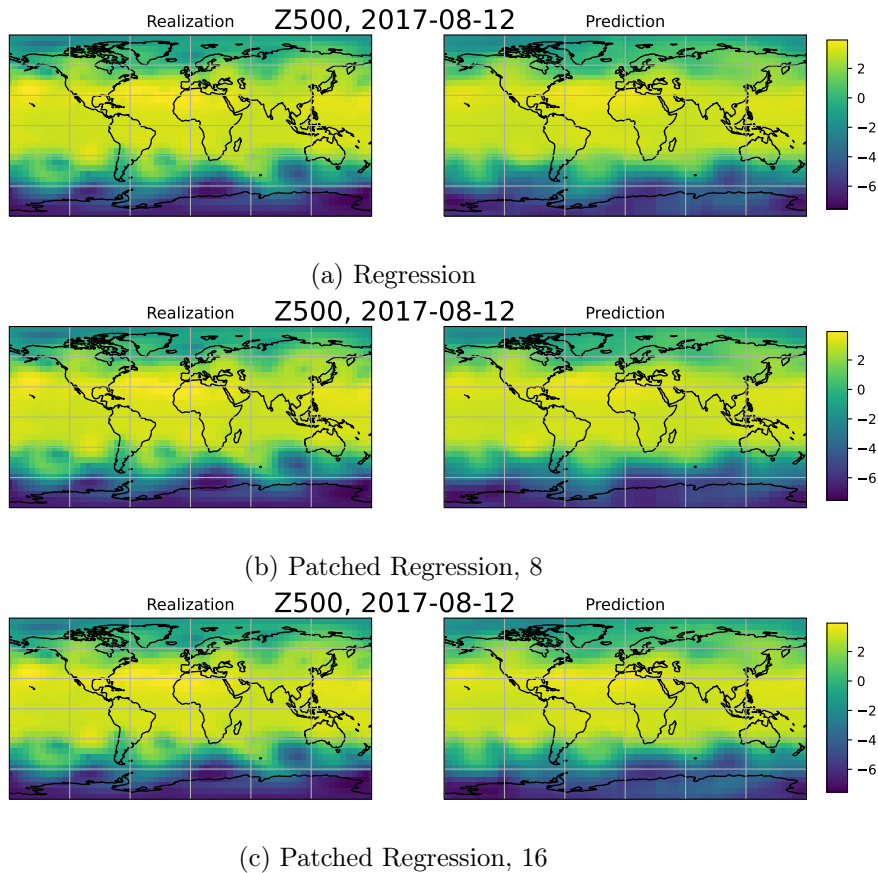
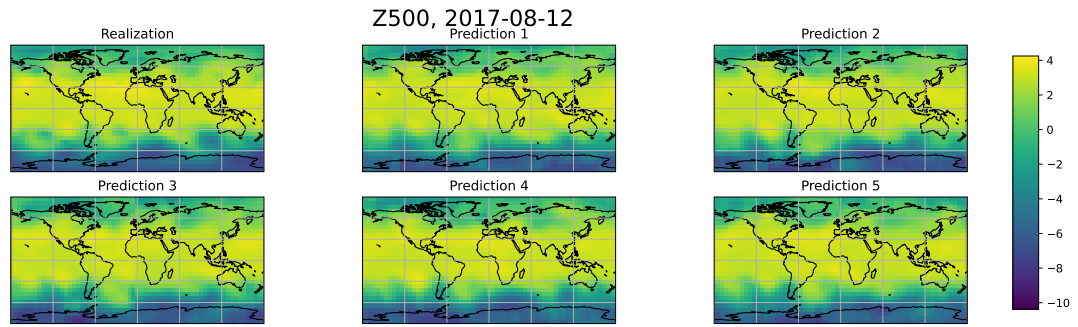
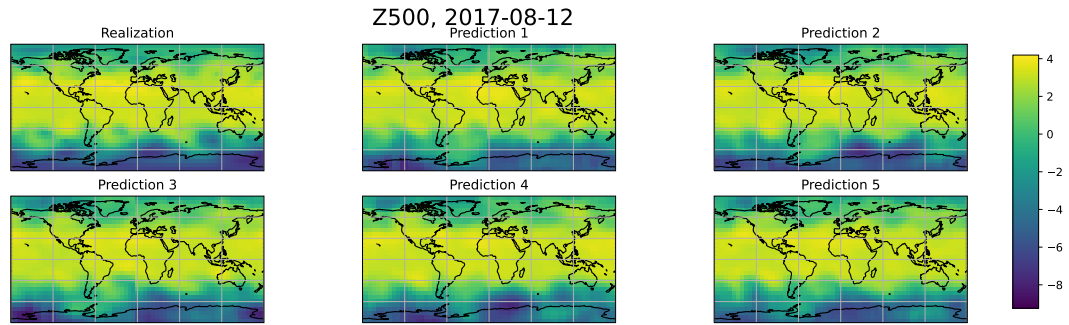


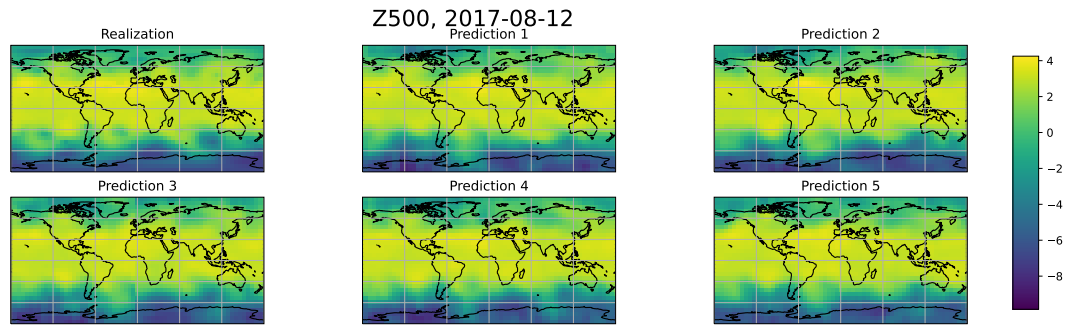
Figure 10: Realization and prediction obtained with the Regression and Patched Regressions for a specific date in the test set for the WeatherBench dataset.



(a) Energy Score

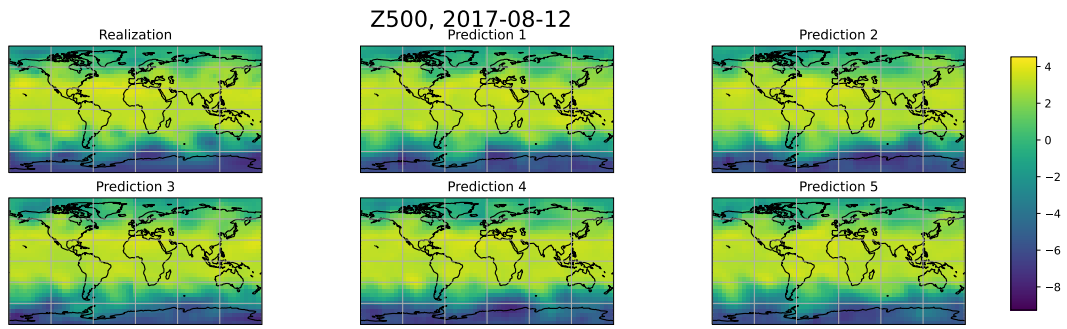


(b) Kernel Score

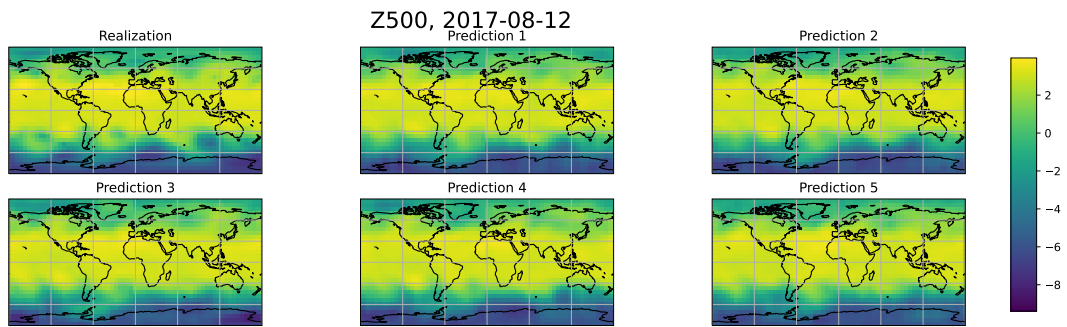


(c) Energy-Kernel Score

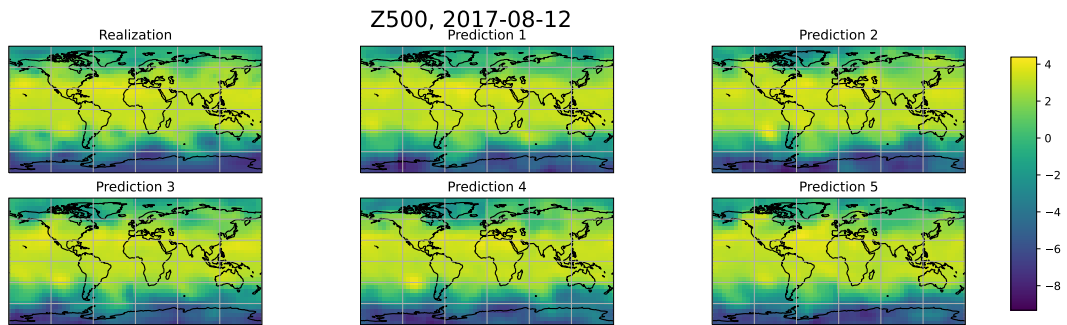
Figure 11: Realization and example of predictions obtained with the Energy, Kernel and Energy-Kernel Scores for a specific date in the test set for the WeatherBench dataset.



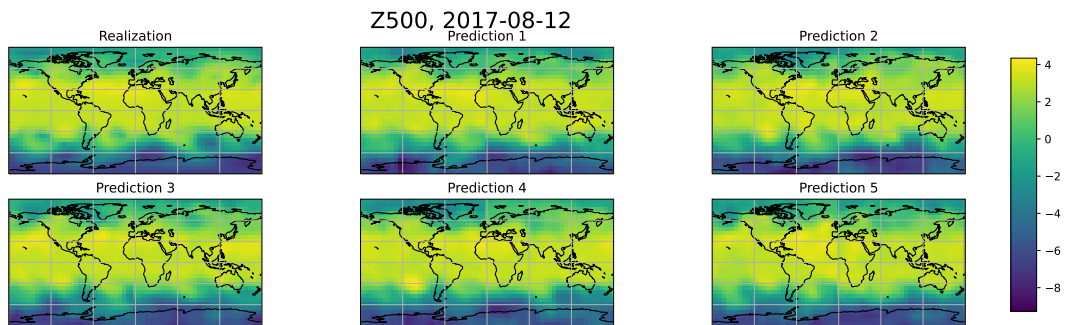
(a) Energy-Variogram Score



(b) Kernel-Variogram Score

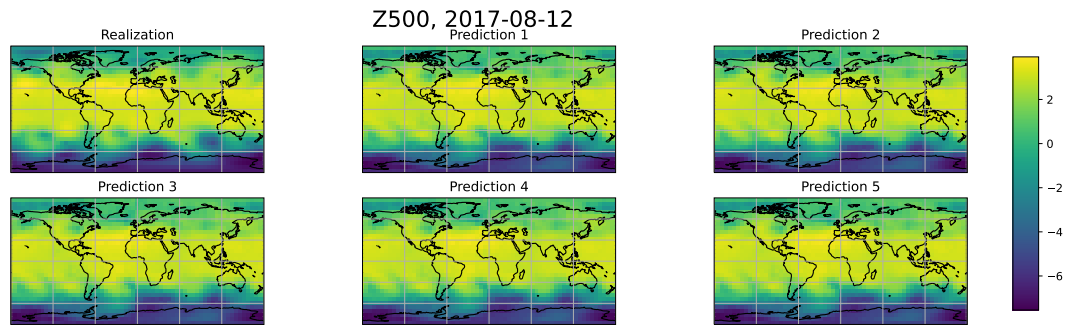


(c) Patched Energy Score (8)

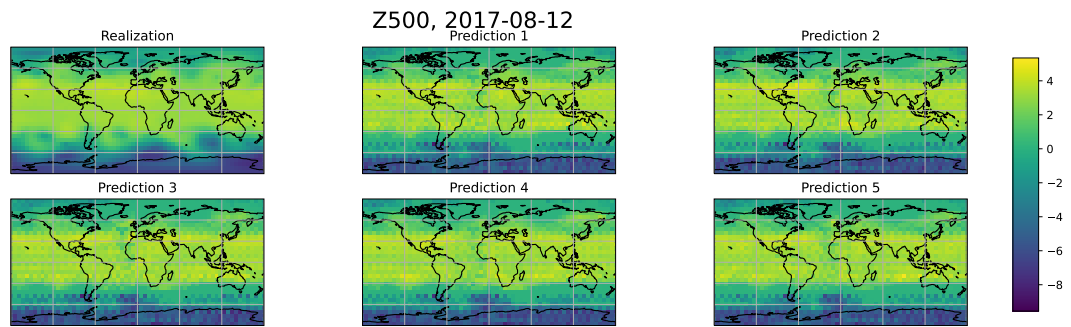


(d) Patched Energy Score (16)

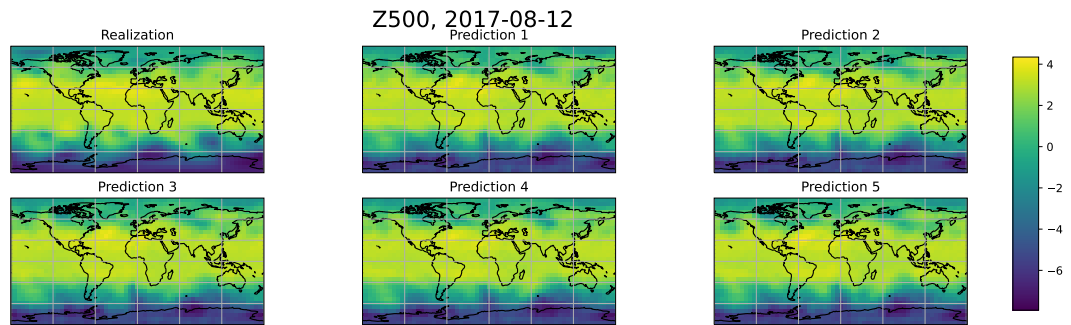
Figure 12: Realization and example of predictions obtained with the Energy-Variogram, Kernel-Variogram and Patched Energy Score (with patch size 8 and 16) for a specific date in the test set for the WeatherBench dataset.



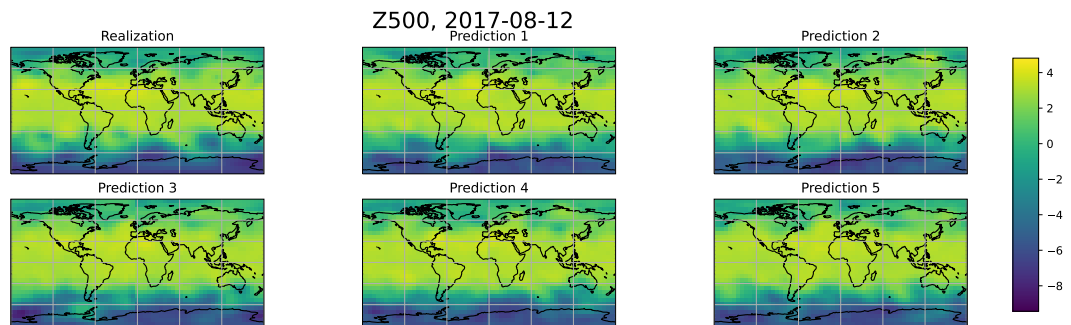
(a) GAN (1)



(b) GAN (2)



(c) GAN (3)



(d) WGAN-GP

Figure 13: Realization and example of predictions obtained with the three considered GAN setups and WGAN-GP for a specific date in the test set for the WeatherBench dataset. Notice how the second GAN setup leads to unphysical features.

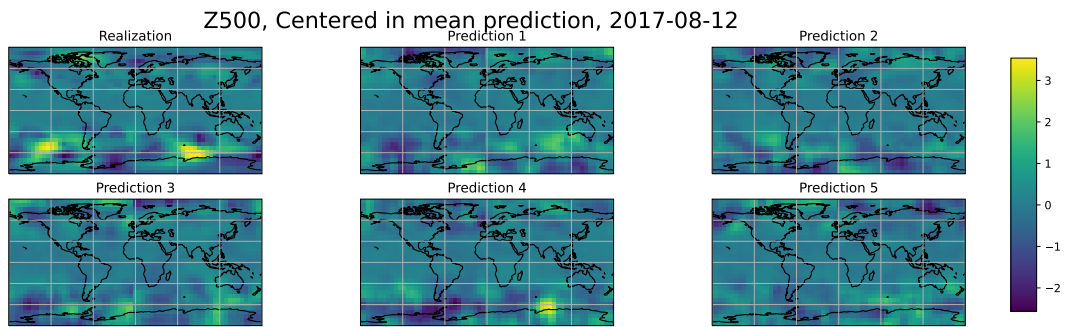
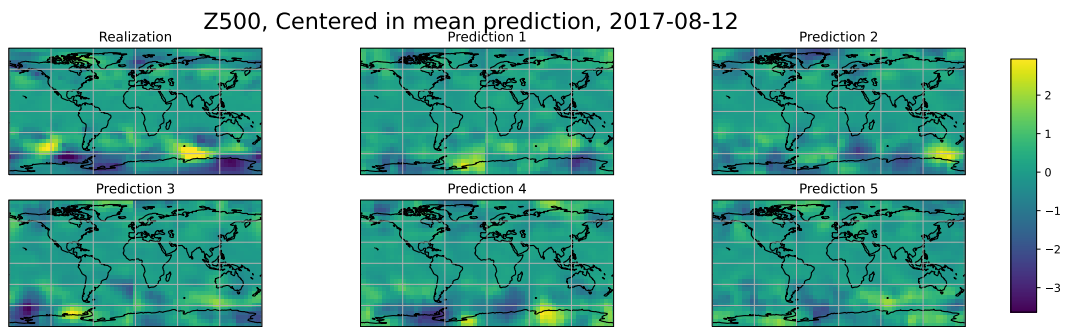
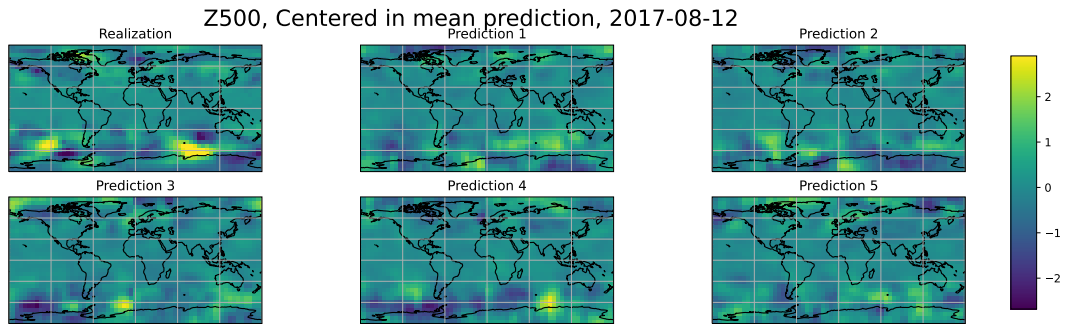


Figure 14: Deviations of the realization and forecasts from the forecast mean (obtained empirically from 100 forecasts) for the Energy, Kernel and Energy-Kernel Scores for a specific date in the test set for the WeatherBench dataset. The absolute values of the forecasts used here are shown in Figure 11.

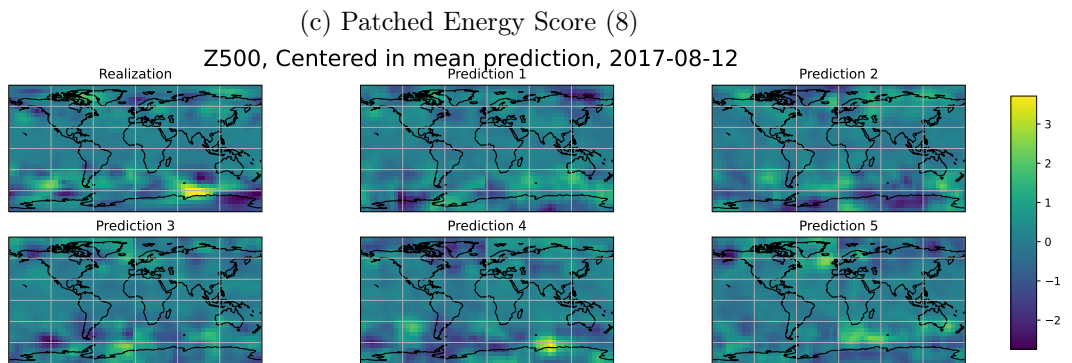
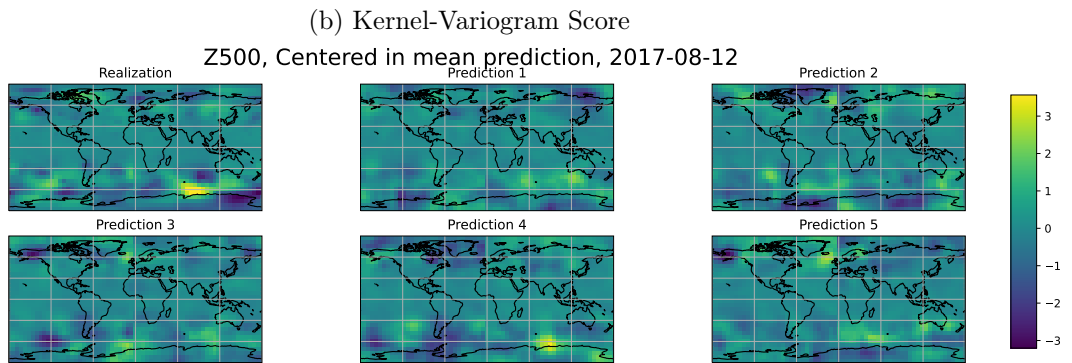
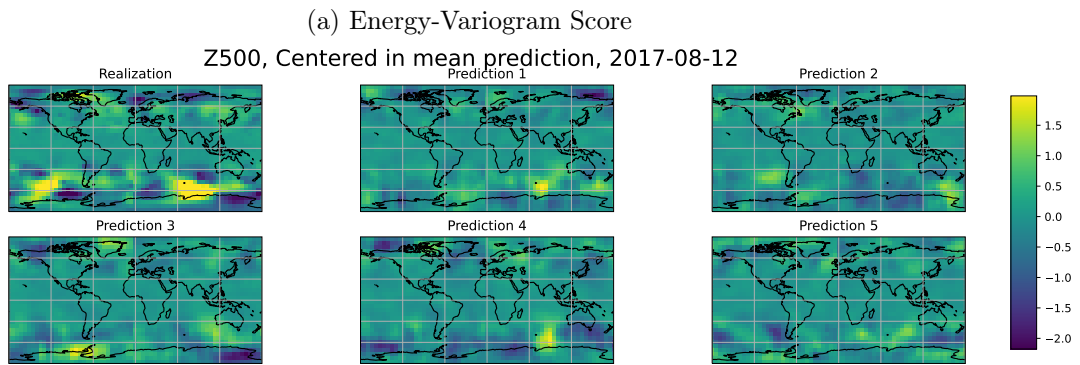
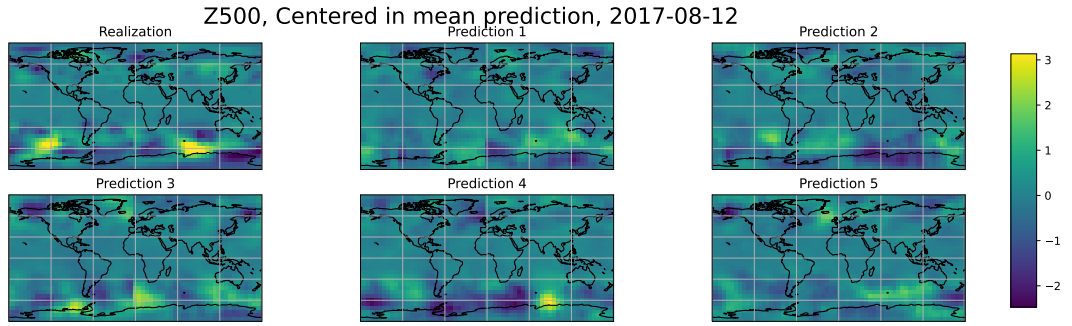
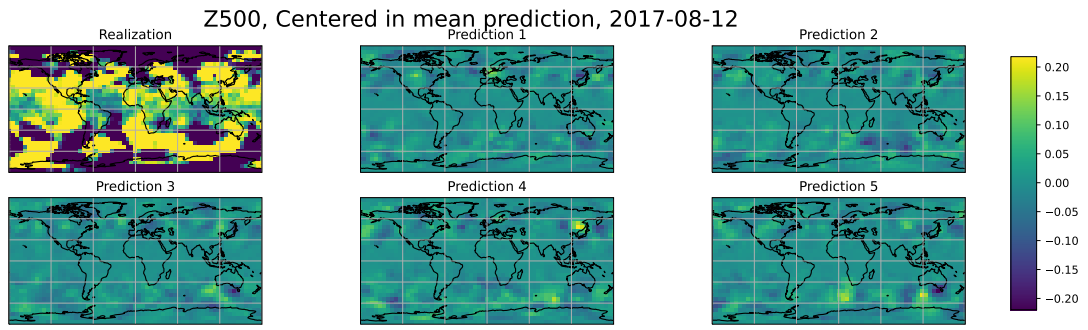
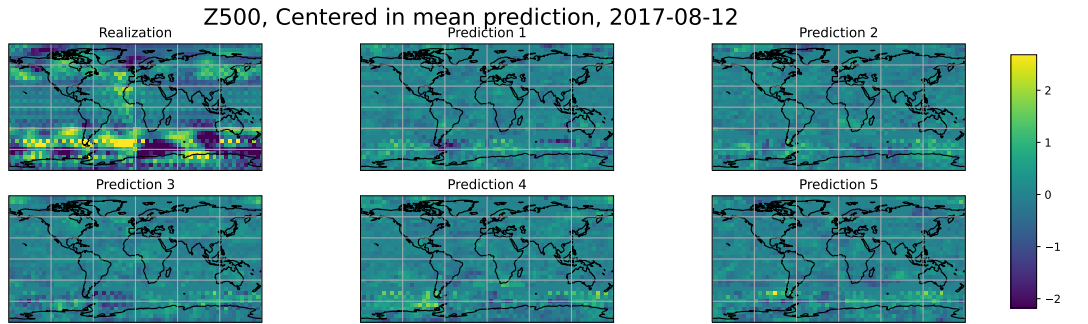


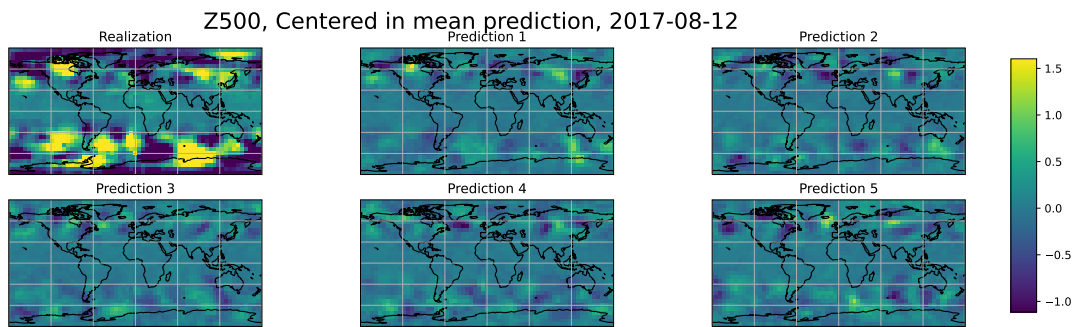
Figure 15: Deviations of the realization and forecasts from the forecast mean (obtained empirically from 100 forecasts) for the Energy-Variogram, Kernel-Variogram and Patched Energy Score (with patch size 8 and 16) for a specific date in the test set for the WeatherBench dataset. The absolute values of the forecasts used here are shown in Figure 12.



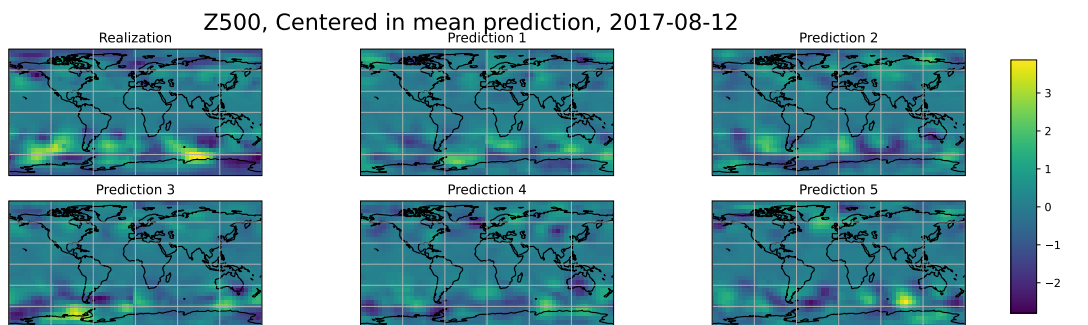
(a) GAN (1)



(b) GAN (2)



(c) GAN (3)



(d) WGAN-GP

Figure 16: Deviations of the realization and forecasts from the forecast mean (obtained empirically from 100 forecasts) for the three considered GAN setups and WGAN-GP for a specific date in the test set for the WeatherBench dataset. Notice how, for the first and third GAN setups, the scale of variations with respect to the predictive mean to the realization is much larger for the realization than for the predictions. Instead, the second GAN setup leads to unphysical features. The absolute values of the forecasts used here are shown in Figure 13.

F.3.5 Time-series plots for selected variables on the grid

In Figures 17, 18, 19 and 20, and show the time series evolution, for a portion of the test period, for 8 randomly selected locations on the WeatherBench grid, for all considered methods (the same locations are shown for all methods). The dashed line represents the true evolution, the solid one the forecast mean, while the shaded region represents 99% credible intervals.

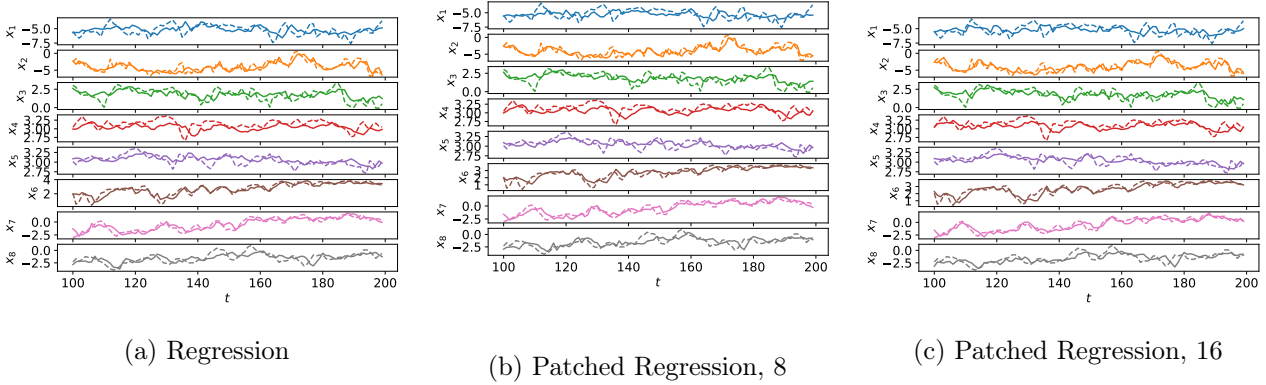


Figure 17: Results with the Regression and patched regression losses for 8 locations on the WeatherBench grid. The panels show observations (dashed line) and median forecast (solid line)

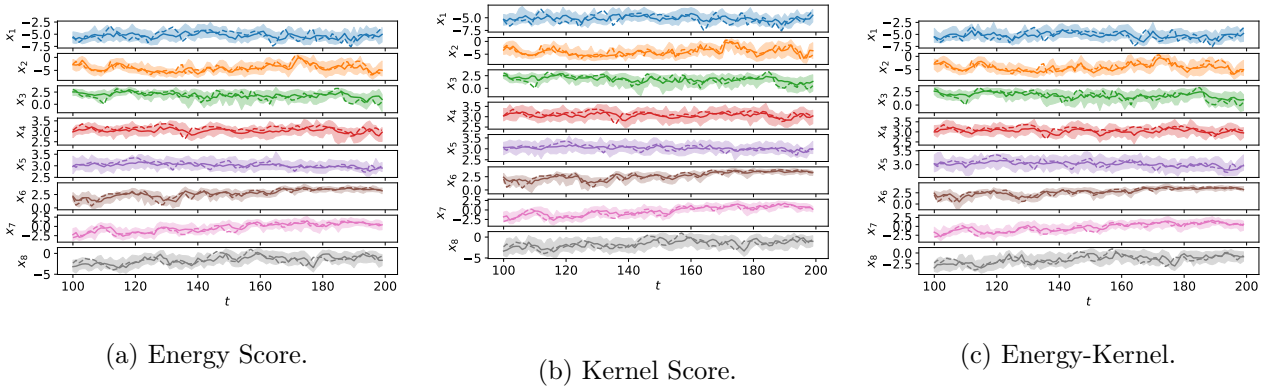


Figure 18: Results with the the Energy, Kernel and Energy-Kernel Scores for 8 locations on the WeatherBench grid. The panels show observations (dashed line), median forecast (solid line) and 99% credible interval (shaded region) for a portion of the test set.

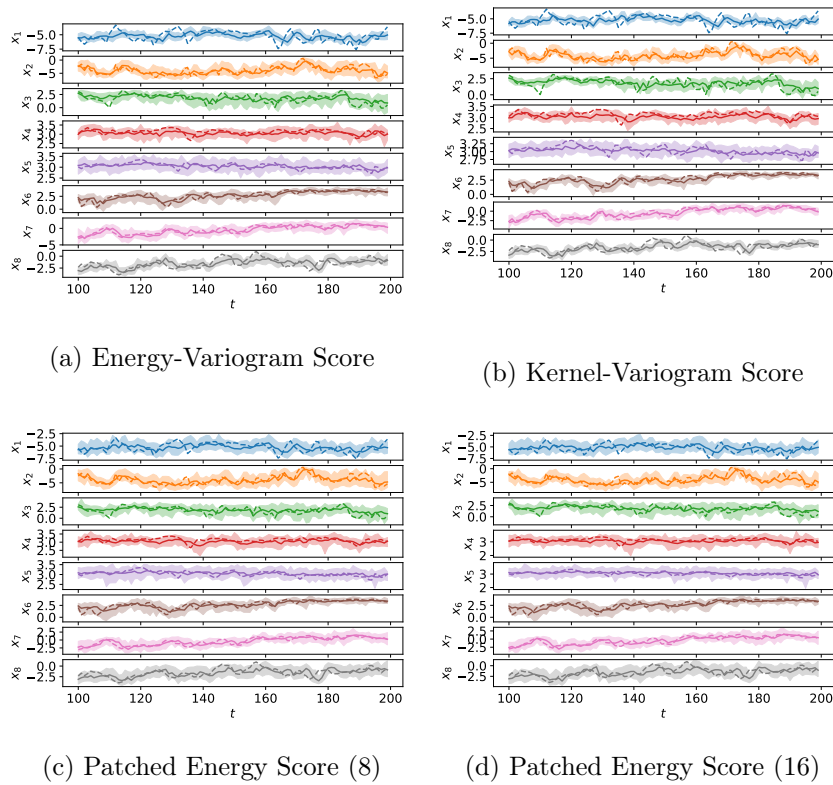


Figure 19: Results with the Energy-Variogram, Kernel-Variogram and Patched Energy Score (with patch size both 8 and 16) Scores for 8 locations on the WeatherBench grid. The panels show observations (dashed line), median forecast (solid line) and 99% credible interval (shaded region) for a portion of the test set.

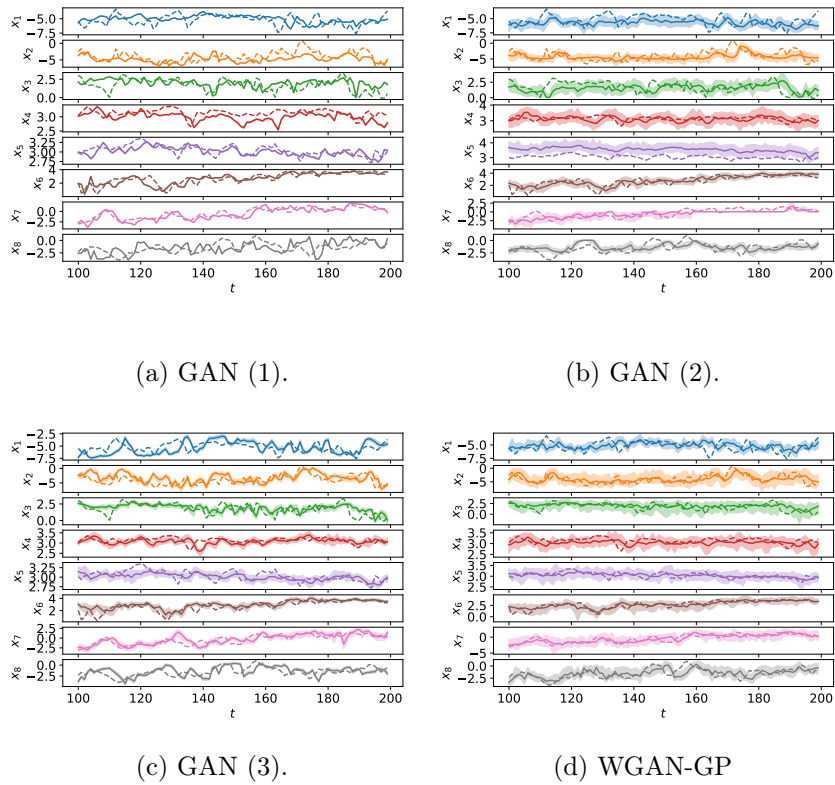


Figure 20: Results with the three considered GAN setups and WGAN-GP Scores for 8 locations on the WeatherBench grid. The panels show observations (dashed line), median forecast (solid line) and 99% credible interval (shaded region) for a portion of the test set. Notice how the first GAN setup severely underestimates the uncertainty region, while the second one forecasts unphysical evolution for some time intervals.

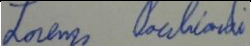
Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor (**only required where there isn't already a statement of contribution within the paper itself**).


Title of Paper	Probabilistic Forecasting with Generative Networks via Scoring Rule Minimization
Publication Status	<input type="checkbox"/> Published <input type="checkbox"/> Accepted for Publication <input type="checkbox"/> Submitted for Publication <input checked="" type="checkbox"/> Unpublished and unsubmitted work written in a manuscript style
Publication Details	Joint work with Prof. Ritabrata Dutta (University of Warwick), Rilwan A. Adewoyin (University of Warwick) and Dr. Peter Dueben (European Center for Medium-Range Weather Forecast).

Student Confirmation

Student Name:	Lorenzo Pacchiardi		
Contribution to the Paper	I am the first author of this paper. I originally thought of using scoring rules to train generative networks for probabilistic forecasting. To reach this idea, the discussions with Dr Dueben and Prof. Dutta about the use of scoring rules in meteorology and the possibility of weather forecasting based on deep learning were essential. I then formalized the methodology, proved the theoretical results, implemented most of the code, run the experiments and wrote the paper. Adewoyin helped with the neural network implementation. Dr Dutta advised along the way, especially in connecting our approach with related works, and corrected the paper draft. Dr Dueben further advised on relevant datasets and models on which the method could be tested.		
Signature		Date	1 st September 2022

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Professor Geoff Nicholls			
Supervisor comments			
Signature		Date	15-09-22

This completed form should be included in the thesis, at the end of the relevant chapter.

Chapter 6

Conclusions and discussion

In this final chapter, I summarise the results presented in the thesis and outline possible directions for future research.

6.1 Summary

In this thesis, I considered generative networks and simulator models. These (*implicit*) *generative models* enable high flexibility through the generation of samples from a distribution whose density is inaccessible, so they require specialised inference methods that only rely on sample generation. However, methods for the two classes of generative models have different desiderata: as simulator models typically have relatively few physically meaningful parameters, a distribution on parameter values is convenient; instead, the many parameters (or weights) of generative networks are physically meaningless and this, jointly with the availability of gradients via automatic differentiation, makes giving a point estimate more practical.

Although much research effort has focussed on inference for generative models (Sec. 1.1), open problems remain: methods for simulator models often require specifying summary statistics and are sensitive to outliers in the data; I address these issues respectively in Chapters 2 and 3. On the other hand, standard adversarial training for generative networks can lead to overconfident distributions; in Chapters 4 and 5, I discuss an alternative training framework avoiding this danger.

More in detail, in Chapter 2 I used the sufficient statistics of the best exponential family approximation to the simulator model as summary statistics for Approximate Bayesian Computation (ABC, Sec. 1.1.1.1). The exponential family is fitted to simulated data using score matching (Hyvärinen, 2005) and can be also employed to directly sample from an approximate posterior using MCMC for doubly intractable distributions, without requiring additional model simulations.

Next, in Chapter 3, I designed a new posterior for simulator models based on a generalised Bayesian inference formulation (Sec. 1.2.1). In contrast, common Bayesian Likelihood-Free Inference methods (Sec. 1.1.1) are motivated as approximations of the standard posterior. I proved concentration and outlier robustness properties for this new posterior and extended the results by Drovandi et al. (2015) to justify the use of pseudomarginal MCMC. In empirical studies, I showed improved performance over related methods.

Finally, in Chapters 4 and 5, I trained generative networks via Scoring Rule Minimization. This strategy is simpler than the standard adversarial one (Sec. 1.1.2.1) and yields better calibrated generative distributions. Therefore, I applied this approach to tasks where uncertainty quantification is paramount: in Chapter 4 I addressed Bayesian Likelihood-Free Inference for simulator models, thus falling back to the theme of Chapters 2 and 3; instead, in Chapter 5 I tackled probabilistic forecasting by extending the training objective to dependent data, for which I established consistency.

6.2 Extensions

Here, I outline some possible extensions for my work.

Gradient-based inference for the posterior in Chapter 3 In Chapter 3, I relied on correlated pseudo-marginal MCMC (Picchini et al., 2022) to sample from the Scoring Rule posterior. Unfortunately, the chains produced by this method are “sticky” (albeit less so than those obtained with the standard pseudomarginal MCMC of Andrieu et al., 2009) and require a large number of simulations from the simulator model at each MCMC step. Furthermore, the target is biased, although the bias vanishes as the number of simulations increases.

Another possible inference strategy is to fit a variational approximation to the target through black-box variational inference (Ranganath et al., 2014), which relies on unstable gradient estimates and, as such, typically requires control variates (Tran et al., 2017; Ong et al., 2018b,a). Variational inference methods based on the reparametrisation trick (Kingma and Welling, 2014) are generally better, but applying them to the Scoring Rule posterior requires computing unbiased estimates of the gradients of the log target, which involves propagating gradients through the simulator model.

Modern automatic differentiation packages (Bradbury et al., 2018; Paszke et al., 2019) enable effortless gradient computation; if the simulator model is implemented

with such packages, reparametrisation-trick variational inference can be applied to the Scoring Rule posterior (this was already done for other targets for simulator models in Moreno et al., 2016 and Chérif-Abdellatif and Alquier, 2020) Doing so would also allow sampling from the Scoring Rule posterior by Stochastic-Gradient Markov-Chain Monte Carlo (SGMCMC, Nemeth and Fearnhead, 2021); these class of methods are approximate but only require unbiased gradient estimates of the log target and do not involve rejection steps, and are thus likely to perform better than pseudomarginal MCMC for the Scoring Rule posterior.

Training Neural Stochastic Differential Equations with the Signature Kernel Score

Stochastic Differential Equations (SDEs) are an extension of Ordinary Differential Equations including a deterministic and a diffusion term. Neural SDEs (Tzen and Raginsky, 2019; Li et al., 2020; Hodgkinson et al., 2021) parametrise these two terms using neural networks. From these models, it is possible to sample a path (by integrating the SDE with a numerical solver), but the density of the SDE solutions cannot be evaluated. Kidger et al. (2021) thus proposed to consider Neural SDEs as continuous-time generative networks and designed an adversarial training approach, using another type of Neural Differential Equation as the discriminator.

Similarly to what was done for traditional generative networks in Chapters 4 and 5, the adversarial formulation of Kidger et al. (2021) could be replaced with Scoring Rule minimization. As Neural SDEs produce paths, defining a Scoring Rule over paths is required. To this end, recall that the *signature* of a path $\gamma = (\gamma_1, \dots, \gamma_d) : [0, T] \rightarrow \mathbb{R}^d$ is the infinite collection of iterated integrals (Lyons, 2014; Kidger et al., 2019):

$$\text{Sig}(\gamma) = \left(\left(\int \dots \int_{0 < t_1 < \dots < t_k < 1} \prod_{j=1}^k \frac{d\gamma_{i_j}}{dt}(t_j) dt_1 \dots dt_k \right)_{1 \leq i_1, \dots, i_k \leq d} \right)_{k \geq 0} .$$

The signature characterises univocally γ up to a negligible equivalence class (Kidger et al., 2019). Therefore, two paths γ and ξ can be compared through the dot product $\langle \text{Sig}(\gamma), \text{Sig}(\xi) \rangle$ of their signatures, whose direct computation can, however, only be approximated by truncation. Luckily, Király and Oberhauser (2019) showed that there exists a kernel k_{Sig} such that $k_{\text{Sig}}(\gamma, \xi) = \langle \text{Sig}(\gamma), \text{Sig}(\xi) \rangle$. The kernel Score (Sec. 1.2.2.2) built using k_{Sig} was proven to be proper in Bonnier and Oberhauser (2021); it is thus possible to use it to train neural SDEs by Scoring Rule Minimization, taking advantage of the method in Salvi et al. (2021) to obtain gradients of k_{Sig} .

Combining Scoring Rule Minimization and adversarial training In Chapters 4 and 5, I presented Scoring Rule Minimization and adversarial training as alternative approaches but, in some cases, it may be beneficial to combine them: targeting Scoring Rules would allow to better represent the uncertainty, while the discriminator trained adversarially may lead to more realistic samples by the generative network. Many ways to combine the two approaches are possible: for instance, one can alternate a few steps of adversarial training and Scoring Rule Minimization, or train via Scoring Rule Minimization up to convergence and then perform adversarial training. In future work, I plan to investigate the best combination approach.

Bibliography

- Akesson, M., Singh, P., Wrede, F., and Hellander, A. (2021). Convolutional neural networks as summary statistics for approximate Bayesian computation. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*.
- Alsing, J., Charnock, T., Feeney, S., and Wandelt, B. (2019). Fast likelihood-free cosmology with neural density estimators and active learning. *Monthly Notices of the Royal Astronomical Society*, 488(3):4440–4458.
- An, Z., Nott, D. J., and Drovandi, C. (2020). Robust Bayesian synthetic likelihood via a semi-parametric approach. *Statistics and Computing*, 30(3):543–557.
- An, Z., South, L. F., Nott, D. J., and Drovandi, C. C. (2019). Accelerating Bayesian synthetic likelihood with the graphical lasso. *Journal of Computational and Graphical Statistics*, 28(2):471–475.
- Andrieu, C., Roberts, G. O., et al. (2009). The pseudo-marginal approach for efficient Monte Carlo computations. *The Annals of Statistics*, 37(2):697–725.
- Arjovsky, M. and Bottou, L. (2017). Towards principled methods for training generative adversarial networks. In *International Conference on Learning Representations*.
- Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR.
- Arora, S., Ge, R., Liang, Y., Ma, T., and Zhang, Y. (2017). Generalization and equilibrium in generative adversarial nets (GANs). In *International Conference on Machine Learning*, pages 224–232. PMLR.
- Arora, S., Risteski, A., and Zhang, Y. (2018). Do GANs learn the distribution? some theory and empirics. In *International Conference on Learning Representations*.
- Beaumont, M. A. (2010). Approximate Bayesian computation in evolution and ecology. *Annual review of ecology, evolution, and systematics*, 41:379–406.

- Beaumont, M. A. (2019). Approximate bayesian computation. *Annual review of statistics and its application*, 6:379–403.
- Bellemare, M. G., Danihelka, I., Dabney, W., Mohamed, S., Lakshminarayanan, B., Hoyer, S., and Munos, R. (2017). The Cramer distance as a solution to biased Wasserstein gradients. *arXiv preprint arXiv:1705.10743*.
- Bernton, E., Jacob, P. E., Gerber, M., and Robert, C. P. (2019). Approximate Bayesian computation with the Wasserstein distance. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 81(2):235–269.
- Bharti, A., Filstroff, L., and Kaski, S. (2022). Approximate Bayesian computation with domain expert in the loop. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S., editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 1893–1905. PMLR.
- Bińkowski, M., Sutherland, D. J., Arbel, M., and Gretton, A. (2018). Demystifying MMD GANs. In *International Conference on Learning Representations*.
- Bissiri, P. G., Holmes, C. C., and Walker, S. G. (2016). A general framework for updating belief distributions. *Journal of the Royal Statistical Society. Series B, Statistical methodology*, 78(5):1103.
- Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877.
- Bonnier, P. and Oberhauser, H. (2021). Proper scoring rules, gradients, divergences, and entropies for paths and time series. *arXiv preprint arXiv:2111.06314*.
- Bouchacourt, D., Mudigonda, P. K., and Nowozin, S. (2016). DISCO nets: DISsimilarity COefficient networks. *Advances in Neural Information Processing Systems*, 29:352–360.
- Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. (2018). JAX: composable transformations of Python+NumPy programs.

- Bröcker, J. (2009). Reliability, sufficiency, and the decomposition of proper scores. *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, 135(643):1512–1519.
- Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.
- Candille, G. and Talagrand, O. (2005). Evaluation of probabilistic prediction systems for a scalar variable. *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, 131(609):2131–2150.
- Cannon, P., Ward, D., and Schmon, S. M. (2022). Investigating the impact of model misspecification in neural simulation-based inference. *arXiv preprint arXiv:2209.01845*.
- Chen, Y., Zhang, D., Gutmann, M. U., Courville, A., and Zhu, Z. (2021). Neural approximate sufficient statistics for implicit models. In *Ninth International Conference on Learning Representations 2021*.
- Chérif-Abdellatif, B.-E. and Alquier, P. (2020). MMD-Bayes: Robust Bayesian estimation via maximum mean discrepancy. In *Symposium on Advances in Approximate Bayesian Inference*, pages 1–21. PMLR.
- Clark, A., Donahue, J., and Simonyan, K. (2019). Adversarial video generation on complex datasets. *arXiv preprint arXiv:1907.06571*.
- Cockayne, J., Graham, M. M., Oates, C. J., Sullivan, T. J., and Teymur, O. (2022). Testing whether a learning procedure is calibrated. *Journal of Machine Learning Research*, 23(203):1–36.
- Cranmer, K., Pavez, J., and Louppe, G. (2015). Approximating likelihood ratios with calibrated discriminative classifiers. *arXiv preprint arXiv:1506.02169*.
- Dawid, A. P. (1984). Present position and potential developments: Some personal views statistical theory the prequential approach. *Journal of the Royal Statistical Society: Series A (General)*, 147(2):278–290.
- Dawid, A. P. (2006). Probability forecasting. *Encyclopedia of statistical sciences*.

- Dawid, A. P. and Musio, M. (2014). Theory and applications of proper scoring rules. *Metron*, 72(2):169–183.
- Dawid, A. P., Musio, M., and Ventura, L. (2016). Minimum scoring rule inference. *Scandinavian Journal of Statistics*, 43(1):123–138.
- Del Moral, P., Doucet, A., and Jasra, A. (2012). An adaptive sequential Monte Carlo method for approximate Bayesian computation. *Statistics and Computing*, 22(5):1009–1020.
- Drovandi, C. and Frazier, D. T. (2022). A comparison of likelihood-free methods with and without summary statistics. *Statistics and Computing*, 32(3):1–23.
- Drovandi, C. C., Grazian, C., Mengersen, K., and Robert, C. (2018). Approximating the likelihood in ABC. *Handbook of Approximate Bayesian Computation*, pages 321–368.
- Drovandi, C. C., Pettitt, A. N., and Lee, A. (2015). Bayesian indirect inference using a parametric auxiliary model. *Statistical Science*, 30(1):72–95.
- Durkan, C., Murray, I., and Papamakarios, G. (2020). On contrastive learning for likelihood-free inference. In *International Conference on Machine Learning*, pages 2771–2781. PMLR.
- Dziugaite, G. K., Roy, D. M., and Ghahramani, Z. (2015). Training generative neural networks via maximum mean discrepancy optimization. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, pages 258–267.
- Fasiolo, M., Wood, S. N., Hartig, F., and Bravington, M. V. (2018). An extended empirical saddlepoint approximation for intractable likelihoods. *Electronic Journal of Statistics*, 12(1):1544–1578.
- Fearnhead, P. and Prangle, D. (2012). Constructing summary statistics for approximate Bayesian computation: semi-automatic approximate Bayesian computation [with Discussion]. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 74(3):419–474.
- Forbes, F., Nguyen, H., Nguyen, T., and Arbel, J. (2021). Approximate bayesian computation with surrogate posteriors.

- Frazier, D. T. (2020). Robust and efficient approximate Bayesian computation: A minimum distance approach. *arXiv preprint arXiv:2006.14126*.
- Frazier, D. T. and Drovandi, C. (2021). Robust approximate bayesian inference with synthetic likelihood. *Journal of Computational and Graphical Statistics*, pages 1–39.
- Frazier, D. T., Drovandi, C., and Nott, D. J. (2021). Synthetic likelihood in misspecified models: Consequences and corrections. *arXiv preprint arXiv:2104.03436*.
- Frazier, D. T., Nott, D. J., Drovandi, C., and Kohn, R. (2022). Bayesian inference using synthetic likelihood: Asymptotics and adjustments. *Journal of the American Statistical Association*, 0(0):1–12.
- Fujisawa, M., Teshima, T., Sato, I., and Sugiyama, M. (2021). γ -ABC: Outlier-robust approximate Bayesian computation based on a robust divergence estimator. In *International Conference on Artificial Intelligence and Statistics*, pages 1783–1791. PMLR.
- Ghosh, J. K., Delampady, M., and Samanta, T. (2006). *An introduction to Bayesian analysis: theory and methods*, volume 725. Springer.
- Glaser, P., Arbel, M., Doucet, A., and Gretton, A. (2022). Maximum likelihood learning of energy-based models for simulation-based inference. *arXiv preprint arXiv:2210.14756*.
- Glöckler, M., Deistler, M., and Macke, J. H. (2022). Variational methods for simulation-based inference. In *International Conference on Learning Representations*.
- Gneiting, T., Balabdaoui, F., and Raftery, A. E. (2007). Probabilistic forecasts, calibration and sharpness. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(2):243–268.
- Gneiting, T. and Katzfuss, M. (2014). Probabilistic forecasting. *Annual Review of Statistics and Its Application*, 1:125–151.
- Gneiting, T. and Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378.

- Goodfellow, I. (2016). Nips 2016 tutorial: Generative adversarial networks. *arXiv preprint arXiv:1701.00160*.
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. *Advances in neural information processing systems*, 27.
- Greenberg, D., Nonnenmacher, M., and Macke, J. (2019). Automatic posterior transformation for likelihood-free inference. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2404–2414. PMLR.
- Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. (2012). A kernel two-sample test. *The Journal of Machine Learning Research*, 13(1):723–773.
- Gritsenko, A., Salimans, T., van den Berg, R., Snoek, J., and Kalchbrenner, N. (2020). A spectral energy distance for parallel speech synthesis. *Advances in Neural Information Processing Systems*, 33:13062–13072.
- Grover, A., Dhar, M., and Ermon, S. (2018). Flow-gan: Combining maximum likelihood and adversarial learning in generative models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.
- Grünwald, P. and Van Ommen, T. (2017). Inconsistency of Bayesian inference for misspecified linear models, and a proposal for repairing it. *Bayesian Analysis*, 12(4):1069–1103.
- Gui, J., Sun, Z., Wen, Y., Tao, D., and Ye, J. (2021). A review on generative adversarial networks: Algorithms, theory, and applications. *IEEE Transactions on Knowledge and Data Engineering*.
- Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. (2017). Improved training of Wasserstein GANs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 5769–5779.
- Gutmann, M. U., Dutta, R., Kaski, S., and Corander, J. (2018). Likelihood-free inference via classification. *Statistics and Computing*, 28(2):411–425.
- Gutmann, M. U. and Hyvärinen, A. (2012). Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13(Feb):307–361.

- Harakeh, A. and Waslander, S. L. (2021). Estimating and evaluating regression predictive uncertainty in deep object detectors. In *International Conference on Learning Representations*.
- Hermans, J., Begy, V., and Louppe, G. (2020). Likelihood-free MCMC with amortized approximate ratio estimators. In *International Conference on Machine Learning*, pages 4239–4248. PMLR.
- Hersbach, H. (2000). Decomposition of the continuous ranked probability score for ensemble prediction systems. *Weather and Forecasting*, 15(5):559–570.
- Hodgkinson, L., van der Heide, C., Roosta, F., and Mahoney, M. W. (2021). Stochastic continuous normalizing flows: training SDEs as ODEs. In de Campos, C. and Maathuis, M. H., editors, *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, volume 161 of *Proceedings of Machine Learning Research*, pages 1130–1140. PMLR.
- Holmes, C. and Walker, S. (2017). Assigning a value to a power likelihood in a general Bayesian model. *Biometrika*, 104(2):497–503.
- Hyvärinen, A. (2005). Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(Apr):695–709.
- Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134.
- Jewson, J., Smith, J. Q., and Holmes, C. (2018). Principles of Bayesian inference using general divergence criteria. *Entropy*, 20(6):442.
- Jiang, B., Wu, T.-y., Zheng, C., and Wong, W. H. (2017). Learning summary statistic for approximate Bayesian computation via deep neural network. *Statistica Sinica*, pages 1595–1618.
- Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., et al. (2021). Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589.
- Kajihara, T., Kanagawa, M., Yamazaki, K., and Fukumizu, K. (2018). Kernel recursive ABC: Point estimation with intractable likelihood. In *International Conference on Machine Learning*, pages 2400–2409. PMLR.

- Kidger, P., Bonnier, P., Perez Arribas, I., Salvi, C., and Lyons, T. (2019). Deep signature transforms. *Advances in Neural Information Processing Systems*, 32.
- Kidger, P., Foster, J., Li, X., and Lyons, T. J. (2021). Neural SDEs as infinite-dimensional GANs. In *International Conference on Machine Learning*, pages 5453–5463. PMLR.
- Kingma, D. P. and Welling, M. (2014). Auto-encoding variational Bayes. In Bengio, Y. and LeCun, Y., editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.
- Király, F. J. and Oberhauser, H. (2019). Kernels for sequentially ordered data. *Journal of Machine Learning Research*, 20.
- Kleijn, B. J. and van der Vaart, A. W. (2012). The bernstein-von-mises theorem under misspecification. *Electronic Journal of Statistics*, 6:354–381.
- Koochali, A., Dengel, A., and Ahmed, S. (2021). If you like it, GAN it—probabilistic multivariate times series forecast with GAN. *Engineering Proceedings*, 5(1):40.
- Li, C.-L., Chang, W.-C., Cheng, Y., Yang, Y., and Póczos, B. (2017). MMD GAN: Towards deeper understanding of moment matching network. In *NIPS*.
- Li, X., Wong, T.-K. L., Chen, R. T., and Duvenaud, D. K. (2020). Scalable gradients and variational inference for stochastic differential equations. In *Symposium on Advances in Approximate Bayesian Inference*, pages 1–28. PMLR.
- Li, Y., Swersky, K., and Zemel, R. (2015). Generative moment matching networks. In *International Conference on Machine Learning*, pages 1718–1727. PMLR.
- Lintusaari, J., Gutmann, M. U., Dutta, R., Kaski, S., and Corander, J. (2017). Fundamentals and recent developments in approximate Bayesian computation. *Systematic Biology*, 66(1):e66–e82.
- Lueckmann, J.-M., Bassetto, G., Karaletsos, T., and Macke, J. H. (2019). Likelihood-free inference with emulator networks. In *Symposium on Advances in Approximate Bayesian Inference*, pages 32–53. PMLR.
- Lueckmann, J.-M., Boelts, J., Greenberg, D., Goncalves, P., and Macke, J. (2021). Benchmarking simulation-based inference. In Banerjee, A. and Fukumizu, K., editors, *Proceedings of The 24th International Conference on Artificial Intelligence and*

- Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pages 343–351. PMLR.
- Lueckmann, J.-M., Goncalves, P. J., Bassetto, G., Öcal, K., Nonnenmacher, M., and Macke, J. H. (2017). Flexible statistical inference for mechanistic models of neural dynamics. In *Advances in Neural Information Processing Systems*, pages 1289–1299.
- Lyddon, S., Holmes, C., and Walker, S. (2019). General Bayesian updating and the loss-likelihood bootstrap. *Biometrika*, 106(2):465–478.
- Lyons, T. (2014). Rough paths, signatures and the modelling of functions on streams. *arXiv preprint arXiv:1405.4537*.
- Marjoram, P., Molitor, J., Plagnol, V., and Tavaré, S. (2003). Markov chain Monte Carlo without likelihoods. *Proceedings of the National Academy of Sciences*, 100(26):15324–15328.
- Matsubara, T., Knoblauch, J., Briol, F.-X., Oates, C., et al. (2022a). Generalised bayesian inference for discrete intractable likelihood. *arXiv preprint arXiv:2206.08420*.
- Matsubara, T., Knoblauch, J., Briol, F.-X., and Oates, C. J. (2022b). Robust generalised Bayesian inference for intractable likelihoods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 84(3):997–1022.
- Miller, J. W. (2021). Asymptotic normality, concentration, and coverage of generalized posteriors. *Journal of Machine Learning Research*, 22(168):1–53.
- Mohamed, S. and Lakshminarayanan, B. (2017). Learning in implicit generative models. *arXiv preprint arXiv:1610.03483*.
- Montahaei, E., Alihosseini, D., and Baghshah, M. S. (2021). Dgsan: discrete generative self-adversarial network. *Neurocomputing*, 448:364–379.
- Moreno, A., Adel, T., Meeds, E., Rehg, J. M., and Welling, M. (2016). Automatic variational ABC. *arXiv preprint arXiv:1606.08549*.
- Murray, I., Ghahramani, Z., and MacKay, D. (2012). MCMC for doubly-intractable distributions. *arXiv preprint arXiv:1206.6848*.

- Nadjahi, K., De Bortoli, V., Durmus, A., Badeau, R., and Şimşekli, U. (2020). Approximate Bayesian computation with the sliced-Wasserstein distance. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5470–5474. IEEE.
- Nemeth, C. and Fearnhead, P. (2021). Stochastic gradient markov chain monte carlo. *Journal of the American Statistical Association*, 116(533):433–450.
- Nguyen, H. D., Arbel, J., Lü, H., and Forbes, F. (2020). Approximate Bayesian computation via the energy statistic. *IEEE Access*, 8:131683–131698.
- Nowozin, S., Cseke, B., and Tomioka, R. (2016). f-GAN: Training generative neural samplers using variational divergence minimization. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 271–279.
- Ong, V. M., Nott, D. J., Tran, M.-N., Sisson, S. A., and Drovandi, C. C. (2018a). Variational Bayes with synthetic likelihood. *Statistics and Computing*, 28(4):971–988.
- Ong, V. M.-H., Nott, D. J., Tran, M.-N., Sisson, S. A., and Drovandi, C. C. (2018b). Likelihood-free inference in high dimensions with synthetic likelihood. *Computational Statistics & Data Analysis*, 128:271–291.
- Pacchiardi, L., Künzli, P., Schöngens, M., Chopard, B., and Dutta, R. (2020). Distance-learning for approximate Bayesian computation to model a volcanic eruption. *Sankhya B*.
- Papamakarios, G. and Murray, I. (2016). Fast ε -free inference of simulation models with Bayesian conditional density estimation. In *Advances in Neural Information Processing Systems*, pages 1028–1036.
- Papamakarios, G., Nalisnick, E., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. (2021). Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64.
- Papamakarios, G., Sterratt, D., and Murray, I. (2019). Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. In Chaudhuri, K. and Sugiyama, M., editors, *Proceedings of Machine Learning Research*, volume 89 of *Proceedings of Machine Learning Research*, pages 837–848. PMLR.

- Park, M., Jitkrittum, W., and Sejdinovic, D. (2016). K2-ABC: Approximate Bayesian computation with kernel embeddings. In *Artificial Intelligence and Statistics*.
- Parry, M., Dawid, A. P., Lauritzen, S., et al. (2012). Proper local scoring rules. *The Annals of Statistics*, 40(1):561–592.
- Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d’Alché Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.
- Picchini, U., Simola, U., and Corander, J. (2022). Sequentially Guided MCMC Proposals for Synthetic Likelihoods and Correlated Synthetic Likelihoods. *Bayesian Analysis*, pages 1 – 31.
- Prangle, D. (2018). Summary statistics. In *Handbook of approximate Bayesian computation*, pages 125–152. Chapman and Hall/CRC.
- Price, L. F., Drovandi, C. C., Lee, A., and Nott, D. J. (2018). Bayesian synthetic likelihood. *Journal of Computational and Graphical Statistics*, 27(1):1–11.
- Radev, S. T., Mertens, U. K., Voss, A., Ardizzone, L., and Köthe, U. (2020). Bayes-Flow: Learning complex stochastic models with invertible neural networks. *IEEE Transactions on Neural Networks and Learning Systems*.
- Ramesh, P., Lueckmann, J.-M., Boelts, J., Tejero-Cantero, Á., Greenberg, D. S., Goncalves, P. J., and Macke, J. H. (2022). GATSBI: Generative adversarial training for simulation-based inference. In *International Conference on Learning Representations*.
- Ranganath, R., Gerrish, S., and Blei, D. (2014). Black Box Variational Inference. In Kaski, S. and Corander, J., editors, *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics*, volume 33 of *Proceedings of Machine Learning Research*, pages 814–822, Reykjavik, Iceland. PMLR.
- Ravuri, S., Lenc, K., Willson, M., Kangin, D., Lam, R., Mirowski, P., Fitzsimons, M., Athanassiadou, M., Kashem, S., Madge, S., et al. (2021). Skilful precipitation nowcasting using deep generative models of radar. *Nature*, 597(7878):672–677.

- Richardson, E. and Weiss, Y. (2018). On GANs and GMMs. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 5852–5863.
- Rizzo, M. L. and Székely, G. J. (2016). Energy distance. *Wiley interdisciplinary reviews: Computational statistics*, 8(1):27–38.
- Robert, C. P. and Casella, G. (2005). *Monte Carlo Statistical Methods (Springer Texts in Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.
- Rozet, F. and Louppe, G. (2021). Arbitrary marginal neural ratio estimation for simulation-based inference. In *Fourth Workshop on Machine Learning and the Physical Sciences (NeurIPS 2021)*.
- Saatci, Y. and Wilson, A. G. (2017). Bayesian GAN. *Advances in neural information processing systems*, 30.
- Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training GANs. *Advances in neural information processing systems*, 29.
- Salvi, C., Cass, T., Foster, J., Lyons, T., and Yang, W. (2021). The signature kernel is the solution of a Goursat PDE. *SIAM Journal on Mathematics of Data Science*, 3(3):873–899.
- Schmitt, M., Bürkner, P.-C., Köthe, U., and Radev, S. T. (2022). Detecting model misspecification in amortized bayesian inference with neural networks. *arXiv preprint arXiv:2112.08866*.
- Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016). Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489.
- Sisson, S. A., Fan, Y., and Beaumont, M. (2018). *Handbook of approximate Bayesian computation*. CRC Press.
- Song, Y., Garg, S., Shi, J., and Ermon, S. (2020). Sliced score matching: A scalable approach to density and score estimation. In Adams, R. P. and Gogate, V., editors, *Proceedings of The 35th Uncertainty in Artificial Intelligence Conference*, volume

- 115 of *Proceedings of Machine Learning Research*, pages 574–584, Tel Aviv, Israel. PMLR.
- Stephenson, D. B. (2012). Glossary. In Jolliffe, I. T. and Stephenson, D. B., editors, *Forecast verification: a practitioner’s guide in atmospheric science*, pages 241–249. John Wiley & Sons, Second edition.
- Sutherland, D. J., Tung, H.-Y., Strathmann, H., De, S., Ramdas, A., Smola, A. J., and Gretton, A. (2017). Generative models and model criticism via optimized maximum mean discrepancy. In *ICLR (Poster)*.
- Syring, N. and Martin, R. (2019). Calibrating general posterior credible regions. *Biometrika*, 106(2):479–486.
- Székel, G. J. and Rizzo, M. L. (2005). A new test for multivariate normality. *Journal of Multivariate Analysis*, 93(1):58–80.
- Thomas, O., Dutta, R., Corander, J., Kaski, S., Gutmann, M. U., et al. (2020). Likelihood-free inference by ratio estimation. *Bayesian Analysis*.
- Tran, M.-N., Nott, D. J., and Kohn, R. (2017). Variational Bayes with intractable likelihood. *Journal of Computational and Graphical Statistics*, 26(4):873–882.
- Tzen, B. and Raginsky, M. (2019). Theoretical guarantees for sampling and inference in generative models with latent diffusions. In *Conference on Learning Theory*, pages 3084–3114. PMLR.
- Wang, Y., Kaji, T., and Ročková, V. (2021). Approximate bayesian computation via classification. *arXiv preprint arXiv:2111.11507*.
- Wilks, D. S. (2019). Chapter 9 - forecast verification. In Wilks, D. S., editor, *Statistical Methods in the Atmospheric Sciences*, pages 369–483. Elsevier, Fourth edition.
- Winkler, R. L. (1977). Rewarding expertise in probability assessment. In *Decision making and change in human affairs*, pages 127–140. Springer.
- Wiqvist, S., Frellsen, J., and Picchini, U. (2021). Sequential neural posterior and likelihood approximation.

- Wiqvist, S., Mattei, P.-A., Picchini, U., and Frelsen, J. (2019). Partially exchangeable networks and architectures for learning summary statistics in approximate Bayesian computation. In Chaudhuri, K. and Salakhutdinov, R., editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6798–6807. PMLR.
- Wu, P.-S. and Martin, R. (2020). A comparison of learning rate selection methods in generalized bayesian inference. *arXiv preprint arXiv:2012.11349*.
- Xing, H. (2021). Improving bridge estimators via f -gan. *arXiv preprint arXiv:2106.07462*.
- Yoon, J., Jarrett, D., and van der Schaar, M. (2019). Time-series generative adversarial networks. In Wallach, H., Larochelle, H., Beygelzimer, A., d’Alché Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.
- Zellner, A. (1988). Optimal information processing and Bayes’s theorem. *The American Statistician*, 42(4):278–280.
- Zhang, D., Malkin, N., Liu, Z., Volokhova, A., Courville, A., and Bengio, Y. (2022). Generative flow networks for discrete probabilistic modeling. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S., editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 26412–26428. PMLR.