**ABSTRACT**

Neural density estimators have proven remarkably powerful in performing efficient simulation-based Bayesian inference in various research domains. In particular, the BayesFlow framework uses a two-step approach to enable amortized parameter estimation in settings where the likelihood function is implicitly defined by a simulation program. But how faithful is such inference when simulations are poor representations of reality? In this paper, we conceptualize the types of model misspecification arising in simulation-based inference and systematically investigate the performance of the BayesFlow framework under these misspecifications. We propose an augmented optimization objective which imposes a probabilistic structure on the latent data space and utilize maximum mean discrepancy (MMD) to detect potentially catastrophic misspecifications during inference undermining the validity of the obtained results. We verify our detection criterion on a number of artificial and realistic misspecifications, ranging from toy conjugate models to complex models of decision making and disease outbreak dynamics applied to real data. Further, we show that posterior inference errors increase as a function of the distance between the true data-generating distribution and the typical set of simulations in the latent summary space. Thus, we demonstrate the dual utility of MMD as a method for detecting model misspecification and as a proxy for verifying the faithfulness of amortized Bayesian inference.

**Keywords** Deep Learning · Bayesian Inference · Model misspecification · Simulation Based Inference · Invertible Neural Networks

1 Introduction

Computer simulations play a fundamental role in many fields of science. However, finding simulation parameters that faithfully reproduce or predict real-world behavior is difficult and usually analytically intractable. Here, we consider *simulation-based inference* (SBI) as a general method to overcome this difficulty and conduct an in-depth analysis of the BayesFlow approach to SBI [29], which utilizes deep learning to infer Bayesian posterior probabilities for the simulation parameters of interest. Specifically, we study how BayesFlow performs under well-specified vs. misspecified simulations (i.e., simulations that do or do not precisely capture the mechanisms of the real system) and propose a new misspecification measure that reliably detects when the system behavior at test time deviates from the behavior

---

*The code for all reported experiments is available in the public repository at [https://github.com/marvinschmitt/ModelMisspecificationBF](https://github.com/marvinschmitt/ModelMisspecificationBF)*
during training. Detecting such deviations is of crucial importance because a mild, or even severe, mismatch between simulated and actual behavior is unavoidable in practice.

To this end, we develop a theoretical basis for misspecification detection, incorporate it into the BayesFlow framework, and investigate it on two representative scientific simulators: (i) Diffusion models are popular in psychology to decompose human decision processes into cognitive parameters, such as speed of information processing, cautiousness, and non-decisional factors [32]. These parameters must be estimated from response-time experiments, but diffusion models may not always fit human decision processes accurately. (ii) Compartmental models such as SIR are a mainstay of epidemiology [7]. They represent hidden disease dynamics by transitions between groups of people with different properties. The transition parameters of real diseases like Covid-19 are estimated from reported incidence data, but the details of the simulation design (e.g., the number and type of compartments or the handling of reporting errors and counter-measure effects) leave plenty of room for model mismatch.

Recently, deep generative neural networks (DGNNs) were introduced as promising SBI tools in various scientific domains (see [9] for a review). During training, DGNNs are presented with a large number of simulations to learn the relationship between simulation outcomes and the underlying parameters or parameter posteriors. When properly converged, DGNNs can generalize this knowledge to unseen simulated or real observations and infer the corresponding unknown parameters. The training effort thus amortizes quickly over multiple inference queries, in contrast to traditional sampling methods (e.g., MCMC), which cannot leverage experience and must re-run their expensive computations from scratch for each observed data set. Importantly, DGNNs are capable of fully Bayesian inference, that is, they can perform uncertainty quantification within a self-consistent probabilistic framework [13].

So far, work on deep SBI has mainly focused on finding network architectures and training algorithms that achieve the highest performance [11, 16, 22, 29]. In contrast, the consequences of model misspecification have received little to no attention. Model misspecification occurs when the model does not fully represent the actual behavior of the modeled system or when it does not completely account for measurement errors and contamination in the observations serving as inference inputs. In the context of SBI, this phenomenon is also referred to as a simulation gap, and we use the two terms interchangeably. Simulation gaps are especially critical for neural networks, since training data from misspecified models may translate into incorrect generalizations at test time.

This limitation does not exist for classical Bayesian inference methods, such as MCMC: Under certain regularity conditions, they enjoy the guarantee that the obtained samples represent the true parameter posterior even when the model is misspecified [20]. In contrast, amortized Bayesian methods require simulations to be faithful proxies of reality and might yield wrong posteriors when presented with observables which are atypical under the assumed model (cf. Figure 1). Our experiments clearly demonstrate this effect and show how amortized posterior inference gradually deteriorates as the simulation gap widens.

Consequently, amortized SBI methods must be able to detect simulation gaps and subsequent posterior errors, so that they can warn users about suspicious outputs (or even decline to make predictions under such circumstances) and guide model designers in their search for better simulations. Traditional Bayesian model checking methods [12] are not applicable here, because they may obscure deficiencies in the parameter posteriors (e.g., over- or underconfidence) and, more importantly, require correct posteriors to begin with. Reliable model misspecification detection is therefore a crucial prerequisite for trustworthy simulation-based inference with amortized deep learning methods.

The main purpose of this paper is twofold. First, we conceptualize model misspecification in applications of simulation-based Bayesian inference with neural networks. We then propose a simple and intuitive way to detect model misspecification and posterior inference errors (posterior errors, for short) arising due to finite training (see Figure 1 for a conceptual illustration). Our approach does not modify existing neural architectures and greatly benefits from the properties of amortized inference. Further, it builds upon our previous work with invertible neural networks (INNs, [1]) and the BayesFlow framework [29] towards a principled simulation-based Bayesian workflow with INNs. Indeed, the demand for a trustworthy workflow in amortized Bayesian inference increases heavily due to the growing number of applications relying on BayesFlow or related frameworks [5, 10, 15, 34]. The main contributions of our paper are:

1. A way to conceptualize different sources of model misspecification in amortized Bayesian inference with neural networks;
2. An augmented optimization objective and a criterion to detect model misspecification and posterior errors during inference, regardless of a model’s structure, input, or output;
3. A systematic investigation of our detection criterion and its relationship with posterior errors in a variety of models.
BayesFlow can reliably detect model misspecification and posterior errors

Figure 1: Conceptual overview of our framework as a step towards trustworthy simulation-based inference with neural networks. A summary network $h_{\psi}$ transforms the typical generative set $T(G)$ of a complex model $G$ into the typical set of a simple distribution (e.g., Gaussian). Discrepancies between the model-implied distribution of observables $p(x | M)$ and the one implied by reality $p^o(x)$ (i.e., simulation gaps) manifest themselves as detectable anomalies, causing posterior errors by the inference network $f_\phi$. These anomalies are flagged by $h_{\psi}$ via distribution matching (maximum mean discrepancy).

### 2 Related Work

Neural approaches to amortized simulation-based inference can be categorized as either targeting the posterior [16, 29], the likelihood [19, 26], or both [38]. We hypothesize that the estimation quality of these approaches will be, in general, unpredictable, when faced with atypical real-world data. When operating in a non-amortized manner, sequential neural posterior estimation (SNPE; e.g., [16]) methods might be less exposed to the effects of simulation gaps. SNPE methods iteratively transform a proposal distribution into an approximate posterior and thus progressively narrow down the scope of plausible simulations informed by an observed data set. However, the behavior and correctness of SNPE methods in the presence of a simulation gap remains an open future research endeavor.

On the other hand, model misspecification has been studied in the context of standard Bayesian inference and various ideas have been explored [18, 37]. For instance, likelihood tempering methods incorporate a modified likelihood (raised to a power $0 < t < 1$) in order to prevent potentially overconfident Bayesian updating [37]. However, the likelihood in SBI is assumed to be unknown and thus impossible to evaluate explicitly. Applying likelihood tempering methods to amortized neural likelihoods appears to be another avenue for future research.

From the perspective of Bayesian model comparison as classification, different outlier detection techniques appear to be viable options for uncovering simulation gaps. For instance, [27] propose to train regularized evidential networks which learn a higher-order distribution over posterior model probabilities. In this way, conclusions about the absolute misfit of all models in a set of candidate models can be drawn. However, this approach is not suitable for parameter estimation and requires a loss function which does not guarantee a correct approximation of posterior model probabilities.

### 3 Methods

#### 3.1 Defining Model Misspecification

For the purpose of simulation-based inference, we define a generative model as a triple $G = (g(\theta, \xi), p(\xi | \theta), p(\theta))$. Such a model generates data points $x \in \mathcal{X}$ according to the system

$$x = g(\theta, \xi) \quad \text{with} \quad \xi \sim p(\xi | \theta), \quad \theta \sim p(\theta),$$

(1)

where $g$ denotes a (randomized) simulation program, $\xi \in \Xi$ is a source of randomness (i.e., noise) with density function $p(\xi | \theta)$, and $p(\theta)$ encodes prior knowledge about plausible simulation parameters $\theta \in \Theta$. Intuitively, $x$ represents quantities that we can observe and measure in the real-world, $\theta$ consists of hidden properties whose role we explicitly understand and model in $g$, and $\xi$ takes care of nuisance effects that we only treat statistically. The abstract spaces $\mathcal{X}, \Xi, \Theta$ denote the domain of possible output data (possible worlds), the scope of noise, and the admissible region of model parameters, respectively. The distinction between hidden properties $\theta$ and contamination $\xi$ is not entirely clear-cut, but depends on our modeling goals and may vary across applications. Moreover, our understanding of the world is constantly evolving, and yesterday’s noise might become tomorrow’s parameters.
Whenever we employ simulations to investigate some real-world phenomenon, a close correspondence between model and reality is necessary. Unacceptably large discrepancies between the two realms are known as a simulation \textit{gap}, and the corresponding model is said to be \textit{misspecified}. Model misspecification can arise from any of the three model components in isolation or simultaneously. A few illustrative examples show what can go wrong in practice:

\textbf{Misspecified Simulator:} In a model for the hydraulic or electric conductivity of a medium, the spatial composition of the material is essential: A simulator \( g \) relying on the assumption of homogeneity will wrongfully predict the behavior of heterogeneous materials and will bias inference results in complex and essentially arbitrary ways [25, 33].

\textbf{Unexpected Contamination:} During an ongoing pandemic, data collection may be severely distorted, for example by noisy measurements, systematic underreporting, and delayed data transfer [10], to name just a few. An epidemiological model disregarding these factors in \( p(\xi | \theta) \) will produce erroneous inferences about key disease parameters, even if the underlying theory was otherwise a good approximation of the actual disease dynamics.

\textbf{Misspecified Prior:} When the admissible region of the prior \( p(\theta) \) is specified too large -- for example, allows for negative mass or intensities -- physically impossible or otherwise nonsensical simulation behavior may arise. On the other hand, when the prior is too narrow, the typical generative set of a model may leave out an important subset of observable outputs.

Our generative model formulation is equivalent to the standard probabilistic factorization of the Bayesian joint distribution into likelihood and prior, \( p(\theta, x | M) = p(x | \theta, M) p(\theta | M) \), where the symbol \( M \) abstractly expresses the prior knowledge and assumptions embodied in the design of \( G \). This likelihood is obtained by marginalizing the joint distribution \( p(\xi, x | \theta, M) \) over all possible values of the nuisance parameters \( \xi \), that is, over all possible execution paths of the simulation program, for a fixed parameter configuration \( \theta \):

\[ p(x | \theta, M) = \int_{\Xi} p(\xi, x | \theta, M) \, d\xi. \tag{2} \]

This integral is typically intractable [9], but we assume that it exists and is non-degenerate, that is, it defines a proper density over the constrained manifold \((\theta, \xi), \xi\), and this density can be learned.

Whenever we model a real-world complex system, we assume an opaque and unknown (true) generator \( x = G(\cdot) \), which yields an unknown (true) distribution \( x \sim p^o(x) \) and is available to the data analyst only via a finite realization (i.e., actually observed data). Then, using the Bayesian formulation, we say that a generative model \( G \) is strictly well-specified if

\[ p^o(x) = p(x | M) \equiv \int_{\Theta} p(x | \theta, M) \, d\theta \tag{3} \]

for every \( x \in \mathcal{X} \). Conversely, a generative model is misspecified if an observable \( x \in \mathcal{X} \) exists for which the above equality is violated. Since models necessarily simplify reality, the above criterion for well-specified models is often unattainable in practice. We therefore relax the requirement by quantifying a model’s degree of misspecification in terms of the information loss incurred by the simplification: For an acceptable upper bound \( \vartheta \) on the information loss, a model is well-specified if

\[ \mathbb{D}[p^o(x) \| p(x | M)] < \vartheta \tag{4} \]

and misspecified otherwise. The symbol \( \mathbb{D} \) denotes a divergence metric quantifying the “distance” between the data distribution implied by reality and the model-implied data distribution (the marginal likelihood). Notably, equality in Eq. 3 implies no information loss by modeling \( p^o(x) \) with \( p(x | M) \) and thus a divergence of zero. A natural choice for \( \mathbb{D} \) would be a metric from the family of \( F \)-divergences, such as the Kullback-Leibler (KL) divergence. However, since the practical computation of \( F \)-divergences requires closed-form densities, whereas \( p^o(x) \) is assumed unknown, we prefer a probability integral metric, such as the Maximum Mean Discrepancy (MMD; [17]). Using the kernel trick, the MMD can be expressed as

\[ \text{MMD}^2[p^o(x) || p(x | M)] = \mathbb{E}_{x,x' \sim p^o(x)} [\kappa(x,x')] + \mathbb{E}_{x,x' \sim p(x | M)} [\kappa(x,x')] - 2\mathbb{E}_{x,x' \sim p^o(x), x' \sim p(x | M)} [\kappa(x,x')] \tag{5} \]

Crucially, this metric is practically tractable because it can be efficiently estimated via finite samples from \( p^o(x) \) and \( p(x | M) \). In the above expression, \( \kappa(\cdot, \cdot) \) is any positive definite kernel and the metric equals zero if and only if the two densities are equal [17].

\subsection*{3.2 The BayesFlow Framework}

The BayesFlow architecture consists of a summary network \( h_{\psi} \) and an inference network \( f_{\phi} \) which jointly amortize a generative model \( G \). The input to BayesFlow is a tuple of parameters and simulated data \((\theta^{(n)}, x^{(n)})\) simulated from \( G \).
The summary network $h_{\psi}$ transforms input data $x^{(n)}$ of variable size to a fixed-length representation $h_{\psi}(x^{(n)})$ which can be interpreted as learned summary statistics. The architecture of the summary network depends on the probabilistic structure of the input data. For instance, a permutation-invariant network can deal with exchangeable data [6] but spatial or time-series data might necessitate convolutional [21] or recurrent blocks [14]. The inference network samples from an approximate posterior $q_\phi$ via a conditional invertible neural network (cINN) which implements a normalizing flow between $\theta$ and a normally distributed $z_\theta$ given the outputs of the summary network:

$$q_\phi(\theta | h_{\psi}(x), \mathcal{M}) = p(z_\theta = f_\phi(\theta; h_{\psi}(x))) \left| \det \left( \frac{\partial f_\phi(\theta; h_{\psi}(x))}{\partial \theta} \right) \right|$$

(6)

Together, the two networks minimize the expected KL divergence:

$$\psi^*, \phi^* = \arg\min_{\psi, \phi} \mathbb{E}_{p(x, \mathcal{M})} \left[ \int \log \frac{p(\theta | x, \mathcal{M})}{q_\phi(\theta | h_{\psi}(x), \mathcal{M})} d\theta \right]$$

(7)

which reduces to

$$\psi^*, \phi^* = \arg\min_{\psi, \phi} \mathbb{E}_{p(\theta | x, \mathcal{M})} \left[ -\log q_\phi(\theta | h_{\psi}(x), \mathcal{M}) \right]$$

(8)

since the true posterior $p(\theta | x, \mathcal{M})$ does not depend on the trainable neural network parameters. We approximate the above expectation via simulations from the generative model $G$ and repeat this process until the networks converge. This objective is self-consistent and recovers the true posterior $p(\theta | x, \mathcal{M})$ under optimal convergence [29]. However, note that simulation-based training takes the expectation with respect to the model $p(\theta, x | \mathcal{M})$, not the actual distribution $p^*(x)$. Thus, optimal convergence does not imply correct posterior inference or faithful prediction in the real world when there is a simulation gap, that is, when the model $G$ deviates critically from the unknown true generator $G^*$.

Importantly, we cannot in general detect a simulation gap by merely analyzing the resulting model posteriors. First, there is no obvious criterion that could reliably distinguish atypical posteriors from the infinite multitude of plausible ones arising from different observations. Second, we must detect simulation gaps before posterior inference, since posteriors under model misspecification may be invalid, which renders distribution analysis meaningless. Consequently, in order to apply Eq. 4, we need a misspecification criterion acting on the level of observables.

### 3.3 Structured Summary Networks

The summary network $h_{\psi}$ is BayesFlow’s main addition to a standard conditional normalizing flow. It acts as an interface between the data $x$ and the inference network $f_\phi$. Its role is to learn maximally informative summary vectors of fixed size $S$ from complex and structured observations (i.e., sets of i.i.d. measurements or multivariate time series). Therefore, the summary network’s data representation $h_{\psi}(x)$ is the most adequate target to detect simulation gaps.

However, the original BayesFlow objective in Eq. 8 does not constrain the outputs of the summary network in any way, so the resulting distribution over summary network outputs $p(h_{\psi}(x) | \mathcal{M})$ is arbitrary or might not even exist. Thus, we extend the original objective to impose a well-defined probabilistic structure on the summary space $h_{\psi}(x)$. Model misspecification can then be detected as a distribution discrepancy in summary space, before posterior inference.

Specifically, we propose to prescribe an $S$-dimensional multivariate unit Gaussian distribution to the summary space, $p(h_{\psi}(x) | \mathcal{M}) \approx \mathcal{N}(z_x | 0, I)$, by minimizing the MMD between summary network outputs and random draws from a unit Gaussian distribution. To ensure that the learned summary vectors comply with the support of the Gaussian density, we introduce a linear (bottleneck) output layer with $S$ units to the summary network. Thus, a random vector in summary space takes the form $h_{\psi}(x) := (s_1, \ldots, s_S) \in \mathbb{R}^S$. The number of summary statistics $S$ is a hyperparameter that depends on the structure and complexity of the data at hand. As a heuristic, we recommend setting $S$ to be at least double the number of generative model parameters $\theta$, in order to allow for enough degrees of freedom (see Experiment 1 for demonstration).

The extended optimization objective of BayesFlow then becomes

$$\psi^*, \phi^* = \arg\min_{\psi, \phi} \mathbb{E}_{p(\theta, x | \mathcal{M})} \left[ -\log q_\phi(\theta | h_{\psi}(x), \mathcal{M}) \right] + \gamma \cdot \text{MMD}^2 \left[ p(h_{\psi}(x) | \mathcal{M}) \parallel p(z_x) \right]$$

(9)

with a hyperparameter $\gamma$ to control the relative weight of the MMD term. The weight parameter $\gamma$ can either be fixed or gradually increased from zero up to a fixed value during training. Intuitively, this objective encourages the approximate posterior $q_\phi(\theta | h_{\psi}(x), \mathcal{M})$ to match the true posterior and the compressed data distribution $p(h_{\psi}(x) | \mathcal{M})$ to match a unit Gaussian distribution. Note, that the extended objective does not constitute a trade-off between the two terms, since the MMD term merely reshapes the summary distribution in an information preserving manner. Indeed, our experiments confirm that the extended objective does not impose restrictions on learnable posteriors or other limits on the networks’ estimation capabilities.
3.4 Theoretical Implications

Attaining the global minimum of Eq. 9 with an arbitrarily expressive BayesFlow architecture \( \{ h_{\psi^*}, f_{\phi^*}, \mathcal{M} \} \) implies that i) the inference and summary network jointly amortize the correct posterior \( p(\theta | x, \mathcal{M}) \) and ii) the summary network transforms \( p(x | \mathcal{M}) \) into a unit Gaussian \( p(z_x = h_{\psi^*}(x)) = \mathcal{N}(z_x | 0, 1) \). According to (i), the set of inference network parameters \( \phi^* \) is a minimizer of

\[
\phi^* = \arg\min_{\phi} \mathbb{E}_{p(z_x)} \mathbb{E}_{p(\theta | z_x)} \left[ -\log q_{\phi}(\theta | z_x) \right],
\]

while (ii) ensures that \( \text{MMD}^2 \left[ p^o(h_{\psi^*}(x)) || p(z_x) \right] > 0 \) implies \( \text{MMD}^2 \left[ p^o(x) || p(x | \mathcal{M}) \right] > 0 \), since a deviation of \( p^o(h_{\psi^*}(x)) \) from a unit Gaussian implies that the summary network is no longer transforming samples from \( p(x | \mathcal{M}) \). Accordingly, \( \text{MMD}^2 \left[ p^o(h_{\psi^*}(x)) || p(z_x) \right] > 0 \) no longer guarantees that the inference network parameters \( \phi^* \) are globally optimal. The preceding argumentation also motivates our augmented objective, since a divergence of \( p^o(h_{\psi^*}(x)) \) from a unit Gaussian signals a deficiency in the assumed generative model \( \mathcal{G} \) and a need to revise the generative model. We also hypothesize and show empirically that we can successfully detect simulation gaps in practice even when convergence of the summary network outputs to a unit Gaussian is not strictly optimal (e.g., in the presence of correlations, cf. Experiments 2 and 3).

However, the converse will not hold true in general, in other words, \( \text{MMD}^2 \left[ p^o(x) || p(x | \mathcal{M}) \right] > 0 \) does not generally imply \( \text{MMD}^2 \left[ p^o(h_{\psi^*}(x)) || p(z_x) \right] > 0 \). To show this via a counter-example, consider the generative model \( \mathcal{G} \) defined by \( x_1 \sim \mathcal{N}(\mu, \sigma^2 = 2), x_2 \sim \mathcal{N}(\mu, \sigma^2 = 2), \mu \sim \delta(0) \) for \( N = 2 \) observations and a single-output summary network, \( S = 1 \). Then, an optimal summary network outputs the minimal sufficient summary statistic \( h_{\psi^*}(x_1, x_2) = \bar{x} \equiv (x_1 + x_2)/2 \). Thus, \( p(x_1, x_2 | \mathcal{M}) = \mathcal{N}(x_1 | 0, 2) \times \mathcal{N}(x_2 | 0, 2) \) and \( p(\bar{x}) = \mathcal{N}(0, 1) \), since \( \text{Var}(\bar{x}) = \text{Var}(x_1 + x_2)/2 = (\text{Var}(x_1) + \text{Var}(x_2))/2 = 1 \), which implies \( \text{MMD}^2 \left[ p(h_{\psi^*}(x_1, x_2)) || \mathcal{N}(0, 1) \right] = 0 \).

Now, suppose that the real data are actually generated by a different process given by \( x_1 \sim \mathcal{N}(\mu, \sigma^2 = 1), x_2 \sim \mathcal{N}(\mu, \sigma^2 = 3), \mu \sim \delta(0) \). Clearly, \( p^o(x_1, x_2) = \mathcal{N}(x_1 | 0, 1) \times \mathcal{N}(x_2 | 0, 3) \neq p(x_1, x_2 | \mathcal{M}) \) and so \( \text{MMD}^2 \left[ p^o(x) || p(x | \mathcal{M}) \right] > 0 \). However, using the same calculations as above, we find that \( p^o(\bar{x}) = \mathcal{N}(0, 1) \) such that \( \text{MMD}^2 \left[ p^o(h_{\psi^*}(x_1, x_2)) || \mathcal{N}(0, 1) \right] = 0 \), despite the fact that the assumed generative model is misspecified.

The above example also shows that learning minimal sufficient summary statistics might not be optimal for detecting simulation gaps. On the other hand, increasing the output dimensions of the summary network \( S \) would enable the network to learn structurally richer (overcomplete) sufficient summary statistics. The latter would be invariant to fewer misspecifications and thus more useful for uncovering simulation gaps. In the above example, an overcomplete summary network with \( S = 2 \) which simply copies and scales the two variables by their corresponding variances is able to detect the misspecification. Next, we describe how to detect simulation gaps during inference using finite realizations from \( \mathcal{G} \) and \( p^o \).

3.5 Detecting Model Misspecification

Once the simulation-based training phase is completed, we can generate a validation sample \( \{ \theta^{(m)}, x^{(m)} \}^M_{m=1} \) from our generative model \( \mathcal{G} \) and pass it through the networks to obtain a sample of random latent vectors \( \{ z_\theta^{(m)}, z_x^{(m)} \}^M_{m=1} \), where \( z_\theta \) refers to the output of the inference network, and \( z_x \) denotes the output of the summary network. The properties of this sample contain important convergence information, which is a prerequisite for meaningful model comparison later on:

(i) If \( \bar{z}_\theta \) is approximately unit Gaussian, we can assume correct posterior inference given the generative model \( \mathcal{G} \);

(ii) If \( \bar{z}_x \) is approximately unit Gaussian, we can assume a structured summary space given \( \mathcal{G} \).

The former implies reasonable posterior performance of BayesFlow for well-specified models. The latter enables model misspecification diagnostics via distribution checking during inference on real data. Let \( \{ x^{(n)} \}^N_{n=1} \) be an observed sample, either simulated from a different generative model \( \mathcal{G}' \neq \mathcal{G} \), or arising from real-world observations with unknown generator \( \mathcal{G} \neq \mathcal{G} \). Before invoking the inference network, we pass this sample through the summary network to obtain \( \{ z_{x,obs}^{(n)} \}^N_{n=1} \). We then compare the validation summary distribution \( \{ z_x^{(m)} \}^M_{m=1} \) with the summary
representation at inference time \( \{ \tilde{z}^{(n)}_{\text{obs}} \}_{n=1}^N \) according to the MMD discrepancy [cf. 17]:

\[
\text{MMD}^2(\tilde{x}, \tilde{x}_{\text{obs}}) = \frac{1}{M^2} \sum_{m=1}^{M} \sum_{m' \neq m} \kappa(\tilde{x}^{(m)}_m, \tilde{x}^{(m')}_{m'}) + \frac{1}{N^2} \sum_{n=1}^{N} \sum_{n' \neq n} \kappa(\tilde{x}^{(n)}_{\text{obs}}, \tilde{x}^{(n')}_{\text{obs}})
- \frac{2}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} \kappa(\tilde{x}^{(m)}_m, \tilde{x}^{(n)}_{\text{obs}})
\]  

(11)

Importantly, we are not limited to pre-determined sizes of simulated or real-world datasets, since the MMD estimator is defined for arbitrary \( M \) and \( N \). Note that we will depict the square root of Eq. 11, \( \text{rMMD}(\cdot, \cdot) = \sqrt{\text{MMD}^2(\cdot, \cdot)} \) in the experimental section because it improves visibility in the graphs.

When the MMD is estimated from finite data, its estimates vary according to a sampling distribution and we can resort to a sampling-based (frequentist) hypothesis test to determine the probability of observed MMD values under well-specified models. We can estimate the MMD sampling distribution under the null hypothesis (i.e., no simulation gap) from multiple sets of simulations from the generative model, \( \{ \tilde{x}_m \}_{m=1}^M \) and \( \{ \tilde{x}^{(n)}_{\text{obs}} \}_{n=1}^N \), with \( M \) large and \( N \) equal to the number of real data sets. Based on the estimated sampling distribution, we can obtain a critical MMD value for a fixed Type I error probability \( (\alpha) \) and compare it to the one obtained via Eq. 11. In general, a larger \( \alpha \)-level corresponds to a more conservative modeling approach: A higher type I error probability implies that more tests reject the null hypothesis, which corresponds to more claimed model misspecification and a higher chance that a model will be revised. Note, that the Type II error probability \( (\beta) \) of this test will generally be high (i.e., the power of the test will be low) whenever the number of real data sets \( N \) is very small. However, we show in Experiment 3 that even as few as 5 data sets from a complex model suffice for obtaining \( \beta \approx 0 \).

If the number of independent summary dimensions \( S \) is small, we can also visually compare the distributions of \( \{ \tilde{z}^{(m)}_m \}_{m=1}^M \) and \( \{ \tilde{z}^{(n)}_{\text{obs}} \}_{n=1}^N \) by inspecting the resulting bivariate distribution plots. In case of correlations or dependencies between latent variables, we can further perform dimensionality reduction, such as principal component analysis (PCA), to ease visual inspection and even obtain simplified interpretable representations (cf. Experiment 2).

Importantly, combining the numeric and the visual approach is especially appealing in practical applications, since it provides an intuitive picture on i) whether the actually observed data lies in the out-of-distribution space of the summary network \( h_{\psi} \), and ii) which data points are flagged as anomalous by the summary network. We now discuss the practical implications of misspecified models for the inferential faithfulness of a BayesFlow architecture trained on a finite amount of simulations.

### 3.6 Posterior Inference Errors due to Misspecified Models

Assuming optimal convergence under a misspecified model \( \mathcal{M} \), the amortized posterior \( q_{\phi}(\theta \mid x = h_{\psi}(\mathcal{G}^o), \mathcal{M}) \) still corresponds to the correct posterior under the misspecified model \( p(\theta \mid x, \mathcal{M}) \) since any transformed \( x \) arising from \( p^o(x) \) has non-zero density in the latent Gaussian summary space\(^2\). Thus, the inference network should still be able to obtain the correct pushforward density under the original model assumptions \( \mathcal{M} \) for any query \( x \) as per Eq. 6. However, optimal convergence can never be achieved after finite training time, so we need to address its implications for the validity of BayesFlow’s posterior inference in practice.

Given finite training data, the summary and inference networks will mostly see simulations from the typical set \( \mathcal{T}(\mathcal{G}) \subset \mathcal{X} \) of the generative model \( \mathcal{G} \), that is, training instances whose self-information \( - \log p(x \mid \mathcal{M}) \) is close to the entropy \( \mathbb{E}[- \log p(x \mid \mathcal{M})] \). In high dimensional problems, the typical set may comprise a rather small subset of the possible outcomes, whose form is determined by a complex interaction between the components of \( \mathcal{G} \) [4]. Accordingly, good convergence in practice may mean that i) only observations from \( \mathcal{T}(\mathcal{G}) \) actually follow the approximate Gaussian in latent summary space and ii) the inference network has only seen enough training examples in \( \mathcal{T}(\mathcal{G}) \) to learn accurate posteriors for observables \( x \in \mathcal{T}(\mathcal{G}) \).

In contrast, atypical or improbable outcomes occur rarely during simulation-based training and have negligible effect on the loss in Eq. 9. Consequently, posterior approximation errors for observations outside of \( \mathcal{T}(\mathcal{G}) \) can be large, simply because the networks have not yet converged in these unusual regions, and the highly non-linear mapping of the inference network still deviates considerably from the true solution. While we cannot yet fix these errors by better

1Note that we do not use the unbiased MMD estimator from [17], because this definition does not allow for data sets with single instances (i.e., \( N = 1 \) or \( M = 1 \)). Estimating MMD for singleton data sets is a fundamental use case for our proposed method in practice. Potential advantages of unbiased estimators are not enough to justify exclusion of the singleton case.

2Assuming that we have no hard-limits in the prior or simulator in \( \mathcal{G} \) that make certain actual observables appear as “impossible”.

---
We then investigate the effect of an overcomplete summary space, namely \( p(x, \mathcal{M}) \). A sample from the noise model \( \mathcal{M}_N \) is simulated by randomly choosing a fraction \( \lambda \in [0, 1] \) of the Gaussian data \( x \) and replacing it with samples from \( \eta \sim \text{Beta}(2, 5) \) which is rescaled to \( \pm 3\sigma_x \) in order to match the empirical scale and make the detection task harder.

Moreover, we hypothesize and demonstrate empirically in the following experiments, that the difference between the true \( p(\theta | x, \mathcal{M}) \) and the approximate posterior \( q_\phi(\theta | h_\psi(x), \mathcal{M}) \) for misspecified models increases as a function of MMD and thus also measures the amount of misspecification. Therefore, our MMD criterion serves a dual purpose in practice: It can uncover potential simulation gaps and, at the same time, signal errors in posterior estimation.

### 4 Experiments

#### 4.1 Experiment 1: Multivariate Normal Distribution

**Aims.** Our first proof-of-concept experiment aims to i) demonstrate the utility of our new optimization objective in terms of accurate posterior estimation; ii) highlight the usefulness of MMD for detecting various simulation gaps during inference; iii) investigate the effects of a minimal vs. overcomplete summary space; and iv) elucidate the relationship between simulation gaps and posterior errors.

**Setup.** We set the stage by estimating the posterior mean of a \( D \)-dimensional conjugate toy multivariate normal (MVN) model with a known analytic posterior in order to best illustrate the behavior of our method. The toy generative model is defined as

\[
\begin{align*}
\mu &\sim \mathcal{N}(\mu_0, \Sigma_0) \\
x_k &\sim \mathcal{N}(x_k | \mu, \Sigma) \quad \text{for } k = 1, ..., K.
\end{align*}
\]

The analytic posterior of the mean \( p(\mu | \{x_k\}_{k=1}^K) \) given \( K \) observations and a known covariance matrix \( \Sigma \) can be analytically expressed as another normal distribution \( \mathcal{N}(\mu_K, \Sigma_K) \) with mean

\[
\mu_K = (\Sigma_0^{-1} + K \Sigma^{-1})^{-1} (\Sigma_0^{-1} \mu_0 + K \Sigma^{-1} \bar{x})
\]

and covariance matrix

\[
\Sigma_K = (\Sigma_0^{-1} + K \Sigma^{-1})^{-1}.
\]

For ease of visualization, we set \( D = 2 \) and simulate data sets consisting of \( K = 100 \) observations. We use a permutation invariant summary network [6] with \( S = 2 \) output dimensions, which equal the number of minimal sufficient statistics implied by the analytic posterior. We set the prior to a unit normal distribution and the likelihood covariance \( \Sigma \) to an identity matrix. The baseline model \( \mathcal{M}^* \) used for training the networks as well as the types of manipulated misspecifications are outlined in Table 1.

**Results.** The BayesFlow network trained to minimize the augmented objective (Eq. 9) exhibits excellent recovery of the analytic posterior means when no misspecification is present (see Figure C.1). All prior misspecifications manifest in anomalies in the resulting summary space which are directly detectable through visual inspection of the 2–dimensional summary space in Figure 2. Note, that the combined prior misspecification (location and scale) exhibits a summary space pattern that mirrors the location and scale of the respective location and scale misspecification. However, based on the 2–dimensional summary space, misspecifications in the fixed parameters of the likelihood (\( \sigma \)) and additive noise (\( \zeta \)) are neither detectable through visual inspection nor through an increased MMD.

We then investigate the effect of an overcomplete summary space, namely \( S = 4 \) summary outputs with an otherwise equal architecture. The overcomplete 4–dimensional summary space captures misspecifications in the noise and simulator through the MMD criterion (see Figure 3b). Furthermore, the induced misspecifications in the noise distribution and simulator are visually detectable in the summary space samples (see Figure A.1). Note, that the 2–dimensional summary space fails to capture these misspecifications (see Figure 3a).
BAYESFLOW can reliably detect Model Misspecification and Posterior Errors

Figure 2: Prior misspecification can be reliably detected with a minimal sufficient summary network $(S = D = 2)$. **Left:** Pairplot of 10,000 summary space samples. All prior misspecifications are distinguishable from the typical latent generative space (blue). **Right:** $\hat{r}_{\text{MMD}}$ as a function of $\mu_0$ (prior location) and $\tau_0$ (scale factor in the mean prior). The $\hat{r}_{\text{MMD}}$ estimate increases monotonically as the simulation gap gets more severe. The colored dots correspond to the respective misspecified model configuration in the pairplot.

Figure 3: $\hat{r}_{\text{MMD}}$ as a function of simulator and noise misspecification. While the minimal summary network yields essentially equal MMD estimates across the grid, the overcomplete summary network captures model misspecifications in both simulator and noise.
(a) Prior misspecification: Both summary networks (minimal and overcomplete) detect increasingly severe misspecification through an elevated rMMD and lead to a higher posterior error (RMSE) of the inference network.

(b) Noise and simulator misspecification: While the minimal network exhibits poor detection, its posterior recovery is not impaired either. On the other hand, the overcomplete network captures increasingly severe misspecification through the rMMD criterion but suffers from an increased posterior error (RMSE).

Figure 4: Posterior error (difference between analytic and approximate posterior means) as a function of model misspecification severity, as indexed by the rMMD criterion.

Finally, we compute the error in posterior recovery via RMSE(µp || µ̂θ) and MMD as a function of misspecification severity for both BayesFlow networks relying on a minimal (S = 2) or an overcomplete (S = 4) summary network. Figure 4 illustrates that a larger MMD estimate coincides with a larger error in posterior estimation across all model misspecifications for both summary networks. However, the minimal and overcomplete networks exhibit a drastically different behavior when processing data from misspecified noise and simulator (see Figure 4b). While the minimal summary network cannot detect simulation gaps due to misspecified noise or simulator, its posterior estimation performance is not heavily impaired either (see Figure 4b). On the other hand, the overcomplete summary network is able to capture misspecifications due to noise and simulator, but also incurs larger posterior inference error.

Note that this experiment used a minimalistic toy conjugate model as a working example in order to ease visualization and intuition. We further extended the toy model to higher dimensions (i.e., D > 2) and a more difficult estimation task (i.e., to recover information beyond first moments). Appendix D describes the results of applying our method to a 5-dimensional MVN with a fully estimated covariance matrix (i.e., having 18 additional free parameters).

4.2 Experiment 2: Drift Diffusion Model

Aims. The second experiment aims to i) apply the new optimization objective to a complex model of decision making; ii) illustrate the effect of dimensionality reduction (principal component analysis); iii) tackle strategies to determine the required number of learned summary statistics in more complex applications; iv) compare the posterior estimation of BayesFlow under a misspecified model with the estimation provided by the Stan implementation of HMC-MCMC [8, 35] as a current gold-standard for Bayesian inference.

Setup. We focus on the drift diffusion model (DDM) – a cognitive model describing reaction times (RTs) in binary decision tasks [31] which is well amenable to amortized inference [30]. The DDM assumes that perceptual information for a choice alternative accumulates continuously according to a Wiener diffusion process. Thus, the change in information \( dx_j \) in experimental condition \( j \) follows a random walk with drift and Gaussian noise:

\[
dx_j = v dt + \xi \sqrt{dt} \quad \text{with} \quad \xi \sim \mathcal{N}(0, 1)\]

Our model implementation assumes five free parameters \( \theta = (v_1, v_2, a_1, a_2, t_0) \) which produce 2-dimensional data stemming from two simulated conditions. The summary network is a permutation-invariant network which reduces \( i, i, d, \) RT data sets to 10-dimensional vectors. We realize a simulation gap by simulating typically observed contaminants: fast guesses (e.g., due to inattention), very slow responses (e.g., due to mind wandering), or a combination of the two. Accordingly, we first generate uncontaminated data \( x^* \) from the well-specified generative model. Second, we randomly choose a fraction \( \lambda \in [0, 1] \) of the data \( x^* \). Third, we replace this data fraction with contaminants \( \xi^o \) whose

\[\text{Notes: Other commonly used metrics (i.e., MSE and MAE) yield identical results. Similarly, error metrics over several posterior quantiles (i.e., } Q_{25} \text{ and } Q_{75}, \text{ instead of posterior means yield identical results.}\]
(a) rMMD estimate by degree of contamination. Dashed lines represent the respective parameter value in the well-specified model without contamination, namely $\lambda_{\text{slow}} = \lambda_{\text{fast}} = 0$.

(b) Correlation between model parameters $\theta$ and principal components (PCs).

Table 2: Posterior error as the estimated rMMD (median and 95% confidence interval) between samples from BayesFlow’s approximate posterior $q_\phi$ and samples from the Stan sampler. The bootstrapped rMMD values (median and 95% confidence interval) for the summary space representation of the 100 investigated data sets and 1000 samples from the uncontaminated model illustrate that posterior errors are mirrored by anomalies in the neural network’s summary space and thus detectable.

| Model (Contamination) | Posterior error rMMD | Summary space rMMD |
|-----------------------|----------------------|--------------------|
| Uncontaminated        | 0.25 [0.13, 0.56]    | 0.45 [0.42, 0.52]  |
| Fast contaminants     | 2.66 [1.44, 3.40]    | 2.68 [2.61, 2.74]  |
| Slow contaminants     | 0.55 [0.23, 1.01]    | 1.18 [1.13, 1.26]  |
| Fast and slow contaminants | 1.90 [0.83, 3.18] | 2.33 [2.19, 2.43] |

Figure 5: Results of Experiment 2 (Drift-Diffusion Model).

distribution depends on the quantiles of the uncontaminated data$^4$:

\[
\begin{align*}
\text{Fast guesses: } & \xi^* \sim \mathcal{U}(0.1, Q_{10}(x^*)) \\
\text{Slow responses: } & \xi^* \sim \mathcal{U}(Q_{75}(x^*), 10)
\end{align*}
\]  

For the comparison with Stan, we simulate 100 uncontaminated DDM data sets and three scenarios (fast guesses, slow responses, fast and slow combined) with a fraction of $\lambda = 0.1$ contaminants.

Results. During inference, our criterion reliably detects the induced misspecifications: Increasing fractions $\lambda$ of contaminants (fast, slow, and combined) manifest themselves in increasing MMD values (see Figure 5a). The results of applying PCA to the summary network outputs $\{\tilde{z}(n)_{\text{obs}}\}_{n=1}^N$ for the well-specified model (no contamination) are illustrated in Figure 5b. We observe that the first five principal components exhibit a large overlap with the true model parameters $\theta$ and jointly account for 85% of the variance in the summary output. Furthermore, the drift rates and decision thresholds within conditions are entangled (i.e., $v_1$, $a_1$ and $v_2$, $a_2$). This entanglement mimics the strong posterior correlations observed between these two parameters. In practical applications, dimensionality reduction might act as a guideline for determining the number of minimally sufficient summary statistics or parameter redundancies for a given problem.

For the comparison with Stan, we juxtapose 4000 samples from the neural network’s approximate posterior $q_\phi$ with 4000 samples obtained from the Stan sampler after ensuring MCMC convergence and sufficient sampling efficiency for each data set in each simulated scenario (see Figure 6 for an illustration). Because Stan is currently considered state-of-the-art for likelihood-based Bayesian inference, we assume the Stan samples are representative of the true posterior

\[Q_{k}(x^*)\] denotes the $k^{th}$ percentile of $x^*$. The asymmetry in percentiles between fast and slow responses arises from the inherent positive skewness of reaction time distributions. The fixed upper limit of slow response contamination is motivated by the maximum number of iterations of the utilized diffusion model simulator. The contamination procedure is executed separately for each condition and response type. If an experiment features both fast and slow contamination, the fraction $\lambda$ is equally split between fast and slow contamination. The uncontaminated data set is generated once and acts as a baseline for all analyses of an experiment, resulting in a baseline MMD of 0 since $x^*$ is unaltered if $\lambda = 0$. 

\[4\]}
BayesFlow can reliably detect model misspecification and posterior errors

Figure 6: Example comparison of the approximate posteriors from Stan and BayesFlow for an uncontaminated data set (left) as well as the contaminated version of the same data set (right; slow contamination).

and compute the average rMMD estimate between the BayesFlow and Stan posterior estimates. Under no model misspecification, the posterior samples from BayesFlow and Stan match almost perfectly (see Figure 6a). In contrast, the results in Figure 6b and Table 2 clearly indicate that the amortized BayesFlow posteriors deteriorate as a result of the induced misspecification. Moreover, these results closely mirror the overall detectability of misspecification obtained by matching the summary representations of 1000 data sets from the uncontaminated process with the representations of the 100 data sets for each of the above scenarios via rMMD (see Table 2).

4.3 Experiment 3: COVID-19 Modeling

Aims. The third experiment applies our new optimization objective to a dynamic model of the early COVID-19 outbreak in Germany [28]. It aims to i) fully replicate the results obtained in [28] using a large structured summary space of size $S = 192$; ii) demonstrate the utility of MMD to detect simulation gaps in the context of non-trivial complex models; iii) investigate the statistical power and characteristics of the sampling-based MMD hypothesis test in a realistic modeling scenario; and iv) showcase the use of MMD as a proxy for increasing model trustworthiness in socially relevant applications.

Setup. Compartmental models in epidemiology (CMs) have been especially popular for inferring relevant disease parameters, simulating possible outbreak scenarios, and projecting future outcomes [10]. Given the abundance of such models and their increasing complexity, the importance of detecting simulation gaps when relying on SBI is two-fold: First, since substantial conclusions are based on the posterior distributions of model parameters, it is important that these distributions are formally correct even when models do not capture all relevant real-world factors. Second, given the dynamic aspect of these models, it is important to detect if an initially well-specified model becomes misspecified at a later time, so the model and its predictions can be amended.

As a final real-world example, we thus focus on a CM representing the early months of the COVID-19 pandemic in Germany. The details of the particular CM used in this section have already been published elsewhere [28]. Here, we investigate the utility of our distribution matching method to detect gross as well as subtle simulation gaps. To achieve this, we train a BayesFlow setup identical to [28] but using our new optimization objective (Eq. 9) to encourage a structured summary space. We then simulate 1000 time-series from the original model $M^*$ and 1000 time-series from...
three misspecified models: i) a model $M_1$ without an intervention sub-model; ii) a model $M_2$ without an observation sub-model; iii) a model $M_3$ without a latent “carrier” compartment. Whereas the first two models are lacking essential components for representing the observed disease dynamics, some studies have demonstrated good results with a simpler model without a latent “carrier” compartment [10]. Thus, we expect that simulation gaps will be harder to detect when comparing the outputs of our original model $M^*$ and model $M_3$ in summary space.

**Results.** Table 3 summarizes the results obtained from this study. We compute the MMD between the summary representation of the 1000 time series from each model and the summaries of the 1000 time series from model $M^*$. We also calculate the power $(1 - \beta)$ of our hypothesis test for each misspecified model $M_j \in \{1, 2, 3\}$ under the sampling distribution estimated from 1000 samples of the 1000 data sets from $M^*$. We set the tolerable type I error probability to $\alpha = .05$. We observe that the power of the test rapidly increases with more data sets and the Type II error probability $(\beta)$ becomes essentially zero for as few as $N = 5$ time series. The detailed results of the power analyses are illustrated in Figure 7.

As a next step, we pass the actually reported data of infected, recovered, and deceased people between 1 March and 21 April 2020 through the summary network and compute the critical MMD value for a sampling-based hypothesis test with an $\alpha$-level of .05 (see Figure 7a). Since the observed MMD of the Germany data are well below the critical MMD value, we conclude that the model is not misspecified for the given time period. Finally, we perform linear dimensionality reduction (PCA) on the summary space and find that the first 40 principal components jointly explain 95% of the variance in the 192−dimensional summary space outputs (see Figure 7b). Thus, a 40-dimensional learned summary vector might provide a good approximation for the true (unknown) minimally sufficient statistics and render inference less fragile in the face of potential misspecifications.
5 Discussion

With this work, we aimed to approach a fundamental problem in simulation-based Bayesian inference, namely, capturing posterior errors due to model misspecification. We argued that misspecified models might cause so-called simulation gaps, resulting in large deviations between simulations and actual empirical data. We further argued that simulation gaps can be detrimental for the performance and faithfulness of simulation-based inference relying on (generative) neural networks. Thus, we proposed to increase the networks’ awareness of posterior errors by compressing the training data into a structured latent space induced by a modified optimization objective. We then applied the maximum mean discrepancy (MMD) estimator, equipped with a sampling-based hypothesis test, as a criterion to spotlight discrepancies between model-implied reality and actual observations in latent space.

Our experiments confirm the utility of the new optimization and detection methods. Furthermore, we observed an interesting connection between detectability, posterior errors, and latent space dimensionality. The results of Experiment 1 suggest a trade-off between model misspecification detection and posterior inference error for different numbers of learnable summary statistics (see Figure 4b). A minimally sufficient summary network failed to detect carefully designed noise and simulator misspecifications, whereas an overcomplete summary network was able to capture these differences. However, using the overcomplete summary network resulted in remarkably worse inference (as measured by deviations from the analytic posterior) under model misspecification.

This finding raises the question of how to determine the number of optimal learnable summary statistics in practical applications. While an intuitive heuristic might suggest “the more, the merrier”, our analyses in Experiment 1 beg to differ depending on the modeling goals. If the focus in a critical application lies in detecting potential simulation gaps, it would be advantageous to utilize a large (overcomplete) summary vector. However, modelers might also desire a network which is as robust as possible during inference. In this case, they might opt for a dimensionality reduction approach: In Experiment 2 and Experiment 3, we illustrate how PCA can be utilized to estimate the number of minimally sufficient summary statistics for an opaque model. Accordingly, using a vector of minimally sufficient statistics might render inference networks more robust against simulation gaps, but also hide potential errors.

Our methods can be extended and rendered more flexible in multiple ways. In this paper, we optimized the latent data space towards a spherical Gaussian structure. However, in principle, our method should work with arbitrary latent data distributions. For example, more heavy-tailed distributions (such as $\alpha$-stable distributions with tunable tail parameters) might be desirable to broaden the latent generative space and reduce the impact of outliers. In fact, initial experiments with the toy example indicate that variations of the summary network structure lowered the error in posterior recovery while still maintaining the ability to detect model misspecification through the proposed MMD criterion (cf. Figure 4).

In addition, considerations on information geometry and non-Euclidean spaces might guide future research into building flexible latent spaces equipped with more sensitive distance metrics [2].

To sum up, the current work presents an endeavor towards a systematic workflow to detect model misspecification and posterior inference errors in simulation-based inference with invertible neural networks. We delineated different forms of model misspecification and investigated new methods for reliable detection of simulation gaps and inference errors. The proposed methods are openly available5 and can be seamlessly integrated into the workflow for end-to-end simulation-based Bayesian parameter estimation with invertible neural networks.

Acknowledgments

This research was supported by the Cyber Valley Research Fund (grant number: CyVy-RF-2021-16), the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany’s Excellence Strategy – EXC-2075 - 390740016 (the Stuttgart Cluster of Excellence SimTech) and EXC-2181 - 390900948 (the Heidelberg Cluster of Excellence STRUCTURES), the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation; grant number GRK 2277 “Statistical Modeling in Psychology”), and the Informatics for Life initiative funded by the Klaus Tschira Foundation. The authors gratefully acknowledge the support and funding.

---

5The development version of the overarching BayesFlow project is available in the public repository at https://github.com/stefanradev93/BayesFlow. Integration to PyPI is planned.
References

[1] Lynton Ardizzone, Jakob Kruse, Sebastian Wirkert, Daniel Rahner, Eric W Pellegrini, Ralf S Klessen, Lena Maier-Hein, Carsten Rother, and Ullrich Köthe. Analyzing inverse problems with invertible neural networks. arXiv preprint arXiv:1808.04730, 2018.

[2] Georgios Arvanitidis, Miguel González-Duque, Alison Pouplin, Dimitris Kalatzis, and Søren Hauberg. Pulling back information geometry. arXiv preprint arXiv:2106.05367, 2021.

[3] John Barnard, Robert McCulloch, and Xiao-Li Meng. Modelling covariance matrices in terms of standard deviations and correlations, with application to shrinkage. Statistica Sinica, 10:1281–1311, 10 2000.

[4] Michael Betancourt. A conceptual introduction to hamiltonian monte carlo. arXiv preprint, 2017.

[5] Sebastian Bieringer, Anja Butter, Theo Heimel, Ullrich Köthe, Tilman Plehn, and Stefan T Radev. Measuring qcd splittings with invertible networks. SciPost Physics Proceedings, 10(6), 2021.

[6] Benjamin Bloem-Reddy and Yee Whye Teh. Probabilistic symmetries and invariant neural networks. J. Mach. Learn. Res., 21:90–1, 2020.

[7] Fred Brauer. Compartmental models in epidemiology. In Mathematical epidemiology, pages 19–79. Springer, 2008.

[8] 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.

[9] Kyle Cranmer, Johann Brehmer, and Gilles Louppe. The frontier of simulation-based inference. Proceedings of the National Academy of Sciences, 117(48):30055–30062, 2020.

[10] Jonas Dehning, Johannes Zierenberg, F Paul Spitzner, Michael Wilczek, and Viola Priesemann. Inferring change points in the spread of covid-19 reveals the effectiveness of interventions. Science, 369(6500), 2020.

[11] Conor Durkan, Iain Murray, and George Papamakarios. On contrastive learning for likelihood-free inference. In International Conference on Machine Learning, pages 2771–2781. PMLR, 2020.

[12] Jonah Gabry, Daniel Simpson, Aki Vehtari, Michael Betancourt, and Andrew Gelman. Visualization in bayesian workflow. Journal of the Royal Statistical Society: Series A (Statistics in Society), 182(2):389–402, 2019.

[13] Andrew Gelman, John B Carlin, Hal S Stern, and Donald B Rubin. Bayesian data analysis. Chapman and Hall/CRC, 1995.

[14] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. Neural computation, 12(10):2451–2471, 2000.

[15] Pedro J Gonçalves, Jan-Matthis Lueckmann, Michael Deistler, Marcel Nonnenmacher, Kaan Öcal, Giacomo Bassetto, Chaitanya Chintaluri, William F Podlaski, Sara A Haddad, Tim P Vogels, et al. Training deep neural density estimators to identify mechanistic models of neural dynamics. Elife, 9:e56261, 2020.

[16] David Greenberg, Marcel Nonnenmacher, and Jakob Macke. Automatic posterior transformation for likelihood-free inference. In International Conference on Machine Learning, pages 2404–2414. PMLR, 2019.

[17] A Gretton, K. Borgwardt, Malte Rasch, Bernhard Schölkopf, and AJ Smola. A kernel two-sample test. The Journal of Machine Learning Research, 13:723–773, 03 2012.

[18] Peter Grünwald, Thijs Van Ommen, et al. Inconsistency of bayesian inference for misspecified linear models, and a proposal for repairing it. Bayesian Analysis, 12(4):1069–1103, 2017.

[19] 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.

[20] Matthew D Hoffman, Andrew Gelman, et al. The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. J. Mach. Learn. Res., 15(1):1593–1623, 2014.

[21] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3431–3440, 2015.

[22] Jan-Matthis Lueckmann, Pedro J Goncalves, Giacomo Bassetto, Kaan Oc̆al, Marcel Nonnenmacher, and Jakob H Macke. Flexible statistical inference for mechanistic models of neural dynamics. Advances in Neural Information Processing Systems, 30, 2017.

[23] Kantilal Varichand Mardia, John T. Kent, and John M. Bibby. Multivariate analysis. Probability and mathematical statistics. Acad. Press, London, 1979.
BayesFlow can reliably detect model misspecification and posterior errors.

[24] Kevin Murphy. Conjugate bayesian analysis of the gaussian distribution, 11 2007.
[25] Wolfgang Nowak and Anneli Guthke. Entropy-based experimental design for optimal model discrimination in the geosciences. *Entropy*, 18(11):409–434, 2016.
[26] George Papamakarios, David Sterratt, and Iain Murray. Sequential neural likelihood: Fast likelihood-free inference with autoregressive flows. In The 22nd International Conference on Artificial Intelligence and Statistics, pages 837–848. PMLR, 2019.
[27] Stefan T Radev, Marco D’Alessandro, Ulf K Mertens, Andreas Voss, Ullrich Kölhe, and Paul-Christian Bürkner. Amortized bayesian model comparison with evidential deep learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.
[28] Stefan T Radev, Frederik Graw, Simiao Chen, Nico T Mutters, Vanessa M Eichel, Till Bärnighausen, and Ullrich Kölhe. Outbreakflow: Model-based bayesian inference of disease outbreak dynamics with invertible neural networks and its application to the covid-19 pandemics in Germany. *PLoS Computational Biology*, 17(10):e1009472, 2021.
[29] Stefan T Radev, Ulf K Mertens, Andreas Voss, Lynton Ardizzone, and Ullrich Kölhe. Bayesflow: Learning complex stochastic models with invertible neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 2020.
[30] Stefan T. Radev, Andreas Voss, Eva Marie Wieschen, and Paul-Christian Bürkner. Amortized Bayesian Inference for Models of Cognition, 2020.
[31] Roger Ratcliff and Gail McKoon. The diffusion decision model: Theory and data for two-choice decision tasks. *Neural Computation*, 20(4):873–922, April 2008.
[32] Roger Ratcliff, Trisha Van Zandt, and Gail McKoon. Connectionist and diffusion models of reaction time. *Psychological review*, 106(2):261, 1999.
[33] Anneli Schöniger, Walter A Illman, Thomas Wöhling, and Wolfgang Nowak. Finding the right balance between groundwater model complexity and experimental effort via bayesian model selection. *Journal of Hydrology*, 531:96–110, 2015.
[34] Takashi Shiono. Estimation of agent-based models using bayesian deep learning approach of bayesflow. *Journal of Economic Dynamics and Control*, 125:104082, 2021.
[35] Stan Development Team. The Stan Core Library, 2018. Version 2.18.0.
[36] Robert Stine. An introduction to bootstrap methods. *Sociological Methods & Research*, 18(2-3):243–291, November 1989.
[37] Owen Thomas and Jukka Corander. Diagnosing model misspecification and performing generalized bayes’ updates via probabilistic classifiers. *arXiv preprint arXiv:1912.05810*, 2019.
[38] Samuel Wiqvist, Jes Frellsen, and Umberto Picchini. Sequential neural posterior and likelihood approximation. *arXiv preprint arXiv:2102.06522*, 2021.
A  Multivariate Normal Distribution: Overcomplete summary statistics

Figure A.1 shows the latent summary space when overcomplete summary statistics ($S = 4$) are used in Experiment 1 to recover the means of a 2–dimensional normal distribution. Model misspecification with respect to both simulator and noise is detectable through anomalies in the latent summary space. Note that a network with $S = 2$ summary statistics and otherwise equivalent architecture could not capture these types of model misspecification.

Figure A.1: Pairplot of 10,000 latent summary space samples from the overcomplete summary network. Both noise (orange) and simulator (pink) misspecifications are distinguishable from the typical latent generative space (blue).
B Implementation details

Drift diffusion model

The diffusion model implementation in Experiment 2 assumes five free parameters which are estimated by BayesFlow, \( \theta = (v_1, v_2, a_1, a_2, t_0) \). For two conditions, the parameters describe individual drift rates \( v_1, v_2 \), individual decision thresholds \( a_1, a_2 \), and a shared non-decision time \( t_0 \). The starting point of the evidence accumulation process is unbiased, \( x_{t=0} = \frac{a_2}{2} \). During training, the parameter priors are:

\[
\begin{align*}
 v_1, v_2 &\sim \Gamma(5, 0.5) \\
 a_1, a_2 &\sim \Gamma(5, 0.5) \\
 t_0 &\sim \Gamma(5, 0.5)
\end{align*}
\] (15)

COVID-19 data: bootstrapping procedure

In Experiment 3, we estimate a sampling distribution of \( \hat{r}_{\text{MMD}} \) between samples from \( \mathcal{M}^* \) and an observational model \( \mathcal{M}_{\text{obs}} \). Since simulating time series from the compartmental models is time-consuming, we opt for bootstrapping [36] on 1 000 pre-simulated time series \( \{x^{\ast(i)}_i\}_{i=1}^{1000} \) from \( \mathcal{M}^* \) and 1 000 pre-simulated time series \( \{x^{(j)}_j\}_{j=1}^{1000} \) from \( \mathcal{M}_{\text{obs}} \).

In each bootstrapping iteration, we draw 1 000 samples (with replacement) from \( \{x^{\ast(i)}_i\}_{i=1}^{1000} \) as well as \( N \in \{1, 2, 5\} \) samples (with replacement) from \( \{x^{(j)}_j\}_{j=1}^{1000} \) and calculate \( \hat{r}_{\text{MMD}} \) between the sets of bootstrap samples.
C  Performance under Model Misspecification

(a) MVN: no misspecification

(b) MVN: Prior misspecification $\mu_0 = 5, \tau_0 = 2.5$

(c) MVN: Simulator misspecification $\tau = 10.0$

(d) MVN: Noise misspecification, $\lambda = 0.5$

Figure C.1: Multivariate Normal Distributions: Performance in recovering the means (Experiment 1)
D Multivariate normal distribution: Mean and full covariance recovery

As an extension to Experiment 1, we investigate the fully extended multivariate normal scenario where the mean and full covariance matrix need to be estimated. The mean vector and covariance matrix are drawn from a joint prior, namely a normal-inverse-Wishart distribution (N-$\mathcal{W}^{-1}$) [3]. The normal-inverse-Wishart prior $\text{N-$\mathcal{W}^{-1}$} (\mu, \Sigma \mid \mu_0, \lambda_0, \Psi_0, \nu_0)$ implies a hierarchical prior. Suppose the covariance matrix

\[ \Sigma \sim \mathcal{W}^{-1}(\Sigma \mid \Psi_0, \nu_0) \]  

has an inverse Wishart distribution and the mean vector

\[ \mu \sim \mathcal{N}(\mu \mid \mu_0, 1/\lambda_0) \]

has a multivariate normal distribution, then the tuple $(\mu, \Sigma)$ has a normal-inverse-Wishart distribution:

\[ (\mu, \Sigma) \sim \text{N-$\mathcal{W}^{-1}$} (\mu, \Sigma \mid \mu_0, \lambda_0, \Psi_0, \nu_0) \]  

Finally, the likelihood is Gaussian:

\[ x_k \sim \mathcal{N}(\mu, \Sigma) \text{ for } k = 1, \ldots, K \]

We apply the BayesFlow method to recover the posterior mean and covariance of a multivariate normal distribution (MVN). For a $D$-variate Gaussian with random means and random covariance, BayesFlow needs to learn $1 + \ldots + D$ covariances and $D$ means. We set the number of summary statistics to $K = 40$, equal to twice the sufficient size for the $5$-variate Gaussian. For a multivariate Gaussian with unknown mean and unknown covariance matrix, the analytic joint posterior $p(\mu_p, \Sigma_p \mid \{x_k\}_{k=1}^K)$ has a closed form, following a normal-inverse Wishart distribution again:

\[ (\mu_p, \Sigma_p \mid \{x_k\}_{k=1}^K) \sim \text{N-$\mathcal{W}^{-1}$} (\mu_p, \Sigma_p \mid \mu_K, \lambda_K, \Psi_K, \nu_K) \]

with

\[
\begin{align*}
\mu_K &= \frac{\lambda_0 \mu_0 + K \bar{x}}{\lambda_0 + K} \\
\lambda_K &= \frac{\lambda_0 + K}{K} \\
\nu_K &= \nu_0 + K \\
\Psi_K &= \Psi_0 + \sum_{k=1}^{K} (x_k - \bar{x})(x_k - \bar{x})^T + \frac{\lambda_0 K}{\lambda_0 + K} (\bar{x} - \mu_0)(\bar{x} - \mu_0)^T
\end{align*}
\]

The marginal posteriors for $\mu_p$ and $\Sigma_p$ then follow as [24]:

\[
\begin{align*}
\mu_p &\sim \mathcal{N}(\mu_p \mid \mu_K, \frac{\Psi_K^{-1}}{\lambda_K (\nu_K - D + 1)}) \\
\Sigma_p &\sim \mathcal{W}^{-1}(\Sigma_p \mid \Psi_K, \nu_K)
\end{align*}
\]

The model $M^*$ used for training the networks as well as the types of induced model misspecifications are outlined in Table 4.

| Model (MMS) | Prior | Likelihood |
|-------------|-------|------------|
| $M^*$ (No MMS) | $(\mu, \Sigma)$ \sim N-$\mathcal{W}^{-1}$($\mu_0 = 0, \lambda_0 = 5, \Psi = I, \nu = 10$) | $x_k \sim \mathcal{N}(\mu, \Sigma)$ |
| $M_P$ (Prior) | $(\mu, \Sigma)$ \sim N-$\mathcal{W}^{-1}$($\mu_0 \neq 0, \lambda_0 = 5, \Psi = \tau_0 I, \nu = 10$) | $x_k \sim \mathcal{N}(\mu, \Sigma)$ |
| $M_S$ (Simulator) | $(\mu, \Sigma)$ \sim N-$\mathcal{W}^{-1}$($\mu_0 = 0, \lambda_0 = 5, \Psi = I, \nu = 10$) | $x_k \sim \mathcal{t}_df(\mu, \Sigma)$, \hspace{1em} df \in \mathbb{N}_{>0}$ |
| $M_N$ (Noise) | $(\mu, \Sigma)$ \sim N-$\mathcal{W}^{-1}$($\mu_0 = 0, \lambda_0 = 5, \Psi = I, \nu = 10$) | $x_k \sim \lambda \cdot \mathcal{B}eta(2, 5) + (1 - \lambda) \cdot \mathcal{N}(\mu, \Sigma)$ |

Table 4: Investigated model misspecifications (MMS) for the 5-dimensional Gaussian with fully estimated covariance matrix. A sample from the noise model $M_N$ is simulated by randomly choosing a fraction $\lambda \in [0, 1]$ of the Gaussian data $x$ and replacing it with samples from $\eta \sim \mathcal{B}eta(2, 5)$ which is rescaled to $\pm 3 \sigma_x$ in order to match the data’s scale.

In the evaluation, we compare the means of BayesFlow’s predicted posterior samples with the first moment of the respective marginal analytic posterior from Equation 21. We evaluate correlation matrices with standard deviations on the diagonal. For the $t$ distributed posterior mean and inverse-Wishart distributed posterior covariance, we obtain [23]:

\[
\begin{align*}
\mathbb{E}(\mu_p) &= \mu_K \\
\mathbb{E}(\Sigma_p) &= \frac{\Psi_K}{\nu_K - D - 1}
\end{align*}
\]
BayesFlow can reliably detect model misspecification and posterior errors

The converged BayesFlow network can recover the analytic posterior means as well as standard deviations and the correlation structure when no model misspecification is present. The performance decreases when model misspecification occurs. Since the summary space comprises $S = 40$ dimensions, visual inspection is no longer feasible. In the following, we inspect the effect of different degrees of model misspecification on the proposed rMMD criterion (see Figure D.1). Both induced prior misspecifications are detectable through an increased rMMD. Model misspecifications through a heavy-tailed likelihood function (i.e., $df = 2$ in the context of a Student-t distribution) in the simulator are detectable as well. Increasing mixture weight $\lambda$ of the Beta noise variate $\eta$ leads to increased MMD values, rendering the investigated noise misspecifications detectable.

Figure D.1: rMMD as a function of model misspecification with respect to the prior (left) as well as simulator and noise (right). All induced model misspecifications are detectable through the proposed detection criterion.
Figure D.2: MVN full covariance: Performance in recovering the means and covariance matrix under model misspecification

(a) MVN full: No misspecification
(b) MVN full: Prior misspecification $\mu_0 = 2, \tau_0 = 1.5$
(c) MVN full: Simulator misspecification, $df = 2$
(d) MVN full: Noise misspecification, $\lambda = 0.5$
E  COVID: Detailed power analysis results

Figure E.1: Detailed illustration of the power analysis in Experiment 3.