Supplementary Materials: Learning Robust Statistics for Simulation-based Inference under Model Misspecification

The appendix is organized as follows:

- In Appendix A, we present the results on detecting model misspecification.
- In Appendix B, we provide further implementation details and results for the numerical experiment in Section 4.
  - Appendix B.1: Implementation details
  - Appendix B.2: Additional posterior plots
  - Appendix B.3: Results for $\mathcal{D}$ being the Euclidean distance
  - Appendix B.4: Computational cost analysis
- In Appendix C, we provide details of the radio propagation experiment of Section 5.

A Detecting misspecification of simulators

Considering that existing SBI methods can yield unreliable results under misspecification and that real-world simulators are probably not able to fully replicate observed data in most cases, detecting whether the simulator is misspecified becomes necessary for generating confidence in the results given by these methods. As misspecification can lead to observed statistics or features falling outside the distribution of training statistics, detecting for it essentially boils down to a class of out-of-distribution detection problems known as *novelty detection*, where the aim is to detect if the test sample $\mathbf{s}_{\text{obs}}$ come from the training distribution induced by $\{s_i\}_{i=1}^m$. This two-label classification problem can potentially be solved by adapting any of the numerous novelty detection methods from the literature. We propose the following two simple novelty detection techniques for detecting misspecification:

Distance-based approach. We assign a score to the observed statistic based on the value of the margin upper bound, as introduced in the main text. We use the MMD as the choice of distance $\mathcal{D}$, and estimate the MMD between the set of simulated statistics $\{s_i\}_{i=1}^m$ and the observed statistic $\mathbf{s}_{\text{obs}}$. This MMD-based score can be used in a classification method to detect misspecification.

Density-based approach. In this method, the training samples $\{s_i\}_{i=1}^m$ are used to fit a generative model q, and the log-likelihood of the observed statistics under q are used as the classification score. We use a Gaussian mixture model (GMM) with k components as q, having the distribution

$$q(s) = \sum_{i=1}^{k} \nu_i \varphi(s|\mu_i, \Sigma_i), \tag{1}$$

where ν_i, μ_i, and Σ_i are the weight, the mean and the covariance matrix associated with the i^{th} component, and φ denotes the Gaussian pdf. The score $\ln q(\mathbf{s}_{\text{obs}})$ can then be used to classify it as either being from in or out of the training distribution.

Experimental set-up. We test the performance of the proposed detection methods on the Ricker model and the OUP with the same contamination model as given in the main text. For each of these simulators, we first train the NPE method on $m = 1000$ training data points, and fit a GMM with $k = 2$ components to them. We then generate 1000 test data-sets or points, half of them from the well-specified model and the other half from the misspecified model, and compute their score. The area under the receiver operating characteristic (AUROC) is used as the performance metric.

Baseline. We construct a baseline for comparing performance of the proposed detection methods. The baseline is based on the insight that under model misspecification, the NPE posterior moves away from the true parameter value (even going outside the prior range). Therefore, we take the root mean squared error (RMSE), defined as $(1/N \sum_{i=1}^{N} (\theta_i - \theta_{\text{true}})^2)^{\frac{1}{2}}$ where $\{\theta_i\}_{i=1}^N$ are posterior samples, as the classification score.

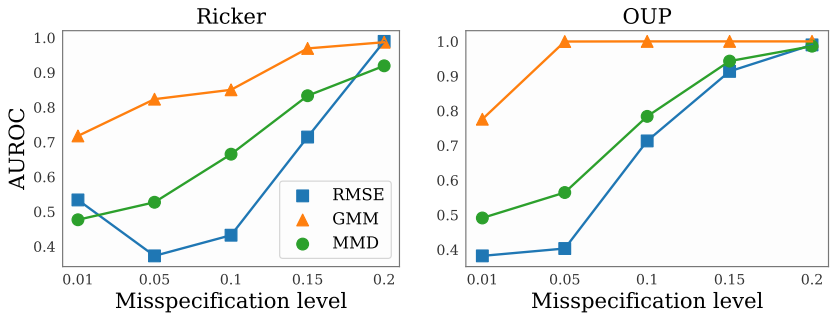


Figure 1: **Misspecification detection experiment**. AUROC of the proposed detection methods (GMM and MMD) versus misspecification level for the Ricker model and the OUP. The RMSE-based baseline is shown in blue.

Results. The AUROC of the classifiers for different levels of misspecification (ϵ in the main text) is shown in Fig. 1 for both the models. The proposed GMM-based detection method performs the best, followed by the MMD-based method. The RMSE-based baseline performs the worst at the classification task. We conclude that it is possible to detect model misspecification in the space of summary statistics using simple to use novelty detection methods.

B Additional details and results of the numerical experiments

B.1 Implementation details

We implement our NPE-RS models based on publicly available implementations from `https://github.com/mackelab/sbi`. We use the NPE-C model [1] with Masked Autoregressive Flow (MAF) [3] as the backbone inference network, and adopt the default configuration with 50 hidden units and 5 transforms for MAF. The batch size is set to 50, and we maintain a fixed learning rate of 5×10^{-4}. The implementation for RNPE is sourced directly from the original repository at `https://github.com/danielward27/rnpe`.

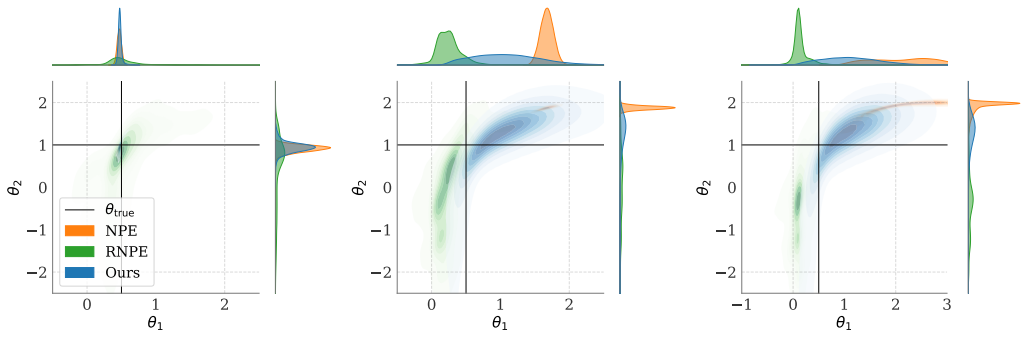
Regarding the summary network in NPE tasks, for the Ricker model, we employ three 1D convolutional layers with 4 hidden channels, and we set the kernel size to 3. For the OUP model, we combine three 1D convolutional layers with one bidirectional LSTM layer. The convolutional layers have 8 hidden channels and a kernel size equal to 3, while the LSTM layer has 2 hidden dimensions. We pass the data separately through the convolutional layers and the LSTM layer and then concatenate the resulting representations to obtain our summary statistics. For the Turin model in Section 5, we utilize five 1D convolutional layers with hidden units set to [8, 16, 32, 64, 8], and the kernel size is set to 3. Across all three summary networks, we employ the `mean` operation as our aggregator to ensure permutation invariance among realizations.

In ABC tasks, we incorporate autoencoders as our summary network. For the Ricker model, the encoder consists of three 1D convolutional layers with 4 hidden channels, where the kernel size is set to 3. The decoder comprises of three 1D transposed convolutional layers with the same settings as the encoder's convolutional layers, allowing for data reconstruction. For the OUP model, we adopt a similar summary network as the one used for the Ricker model but with a smaller stride.

In NPE tasks, we use 1000 samples for the training data, along with 100 realizations of both observed and simulated data for each value of θ. We also use 1000 samples for training the autoencoders. For ABC, we use 4000 samples from the prior and accept $n_\delta = 200$ samples giving a tolerance rate of 5%. We take ρ to be Euclidean distance in the rejection ABC and normalize the statistics by the median absolute deviation before computing the distance to account for the difference in their magnitude.

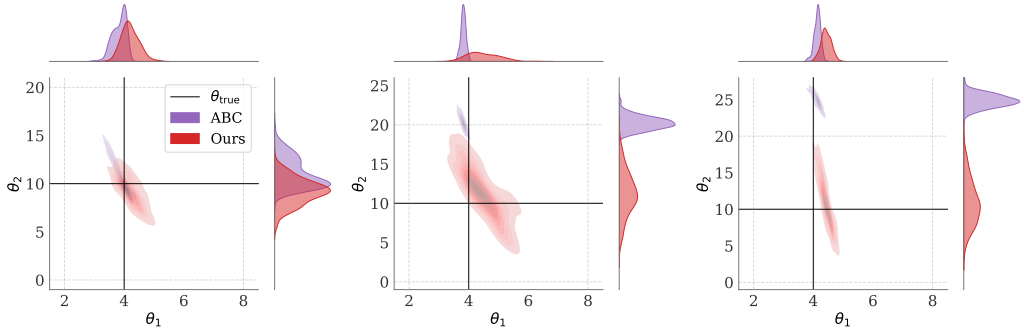
B.2 Additional posterior plots

We now present examples of the remaining posterior plots, apart from the one shown in the main text. The posterior plots for OUP using the NPE-based methods is shown in Figure 2. The observations



(a) Well-specified ($\epsilon = 0$) (b) Misspecified ($\epsilon = 10\%$) (c) Misspecified ($\epsilon = 20\%$)

Figure 2: **Ornstein-Uhlenbeck process.** Posteriors obtained from our method (NPE-RS), RNPE, and NPE for different degrees of model misspecification.



(a) Well-specified ($\epsilon = 0$) (b) Misspecified ($\epsilon = 10\%$) (c) Misspecified ($\epsilon = 20\%$)

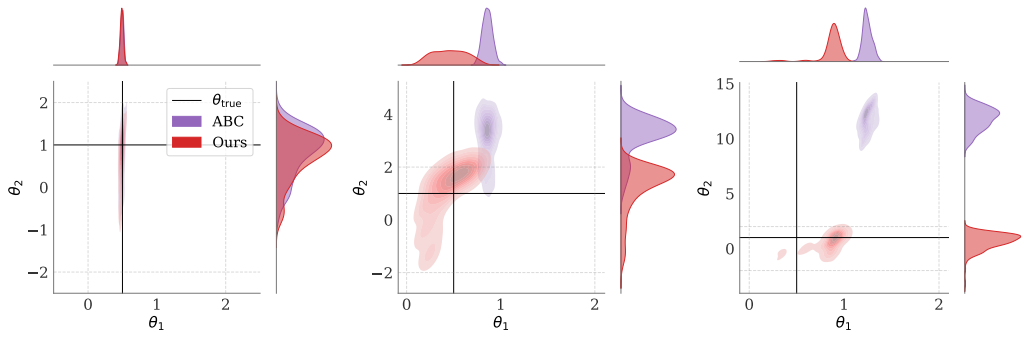
Figure 3: **Ricker model.** Posteriors obtained from our method (ABC-RS) and ABC for different degrees of model misspecification.

are similar to the Ricker model example in the main text: we see that our NPE-RS method yields similar posterior as NPE in the well-specified case, whereas RNPE posteriors are underconfident. When the model is misspecified, NPE posterior goes far from the true parameter value. The NPE-Rs posteriors, however, are still around θ_{true}, demonstrating robustness to misspecification.

Similar behavior is observed in the ABC case for both the Ricker model and OUP in Figure 3 and Figure 4, respectively. The ABC posteriors go outside the prior range under misspecification, while ABC with our robust statistics yields posteriors closer to θ_{true}. In Table 1, we report the sample mean and standard deviations for the results shown in Figure 2 of the main text.

B.3 Results for $\mathcal{D}$ being the Euclidean distance

We present results for $\mathcal{D}$ being the Euclidean distance in the well-specified case of the Ricker model in Figure 5(a). As mentioned in Section 3 of the main text, this choice leads to very underconfident posteriors. This is because the Euclidean distance is not a robust distance: it becomes large even if a few points are far from the observed statistic. As a result, using this as the regularization term penalises most choices of summarizer η, and we learn statistics that are very concentrated around the observed statistic (orange dot). Although a good choice for being robust, Euclidean distance leads to statistics that are not informative about the model parameters, yielding posterior that is similar to the uniform prior. Hence, we used the MMD as the distance in the margin upper bound, which provides better a trade-off between robustness and efficiency (in terms of learning about model parameters).



(a) Well-specified ($\epsilon = 0$) (b) Misspecified ($\epsilon = 10\%$) (c) Misspecified ($\epsilon = 20\%$)

Figure 4: **Ornstein-Uhlenbeck process.** Posteriors obtained from our method (ABC-RS) and ABC for different degrees of model misspecification.

Table 1: Performance of the SBI methods in terms of RMSE and MMD for both Ricker and OUP. We report the average (± 1 std. deviation) values across 100 runs for varying levels of misspecification.

		RMSE (↓)			MMD (↓)		
		$\epsilon = 0\%$	$\epsilon = 10\%$	$\epsilon = 20\%$	$\epsilon = 0\%$	$\epsilon = 10\%$	$\epsilon = 20\%$
Ricker	NPE	**2.16** (3.07)	7.86 (1.57)	11.2 (1.70)	**0.04** (0.07)	0.74 (0.09)	1.06 (0.17)
	RNPE	3.27 (0.35)	5.51 (0.58)	7.14 (1.15)	0.06 (0.05)	0.51 (0.19)	0.79 (0.25)
	NPE-RS (ours)	2.18 (2.66)	**2.19** (1.01)	**4.66** (4.15)	0.09 (0.14)	**0.21** (0.16)	**0.42** (0.37)
	ABC	1.46 (0.44)	6.95 (0.25)	9.79 (0.96)	0.01 (0.01)	0.85 (0.02)	1.18 (0.04)
	ABC-RS (ours)	**1.20** (0.51)	**3.16** (1.08)	**2.99** (1.28)	0.01 (0.02)	**0.17** (0.15)	**0.18** (0.16)
OUP	NPE	0.79 (0.62)	1.26 (1.18)	2.59 (2.75)	0.01 (0.01)	0.34 (0.15)	0.63 (0.29)
	RNPE	0.78 (0.09)	0.87 (0.10)	0.98 (0.15)	0.01 (0.01)	0.22 (0.13)	0.49 (0.26)
	NPE-RS (ours)	**0.74** (0.70)	**0.62** (0.33)	**0.63** (0.36)	0.02 (0.05)	**0.09** (0.09)	**0.21** (0.17)
	ABC	0.50 (0.07)	1.20 (0.40)	5.16 (2.39)	0.05 (0.03)	0.88 (0.21)	0.92 (0.23)
	ABC-RS (ours)	**0.44** (0.06)	**0.62** (0.23)	**0.88** (0.48)	**0.02** (0.02)	**0.26** (0.17)	**0.50** (0.38)

B.4 Computational cost analysis

We now present a quantitative analysis of the computational cost of training with and without our MMD regularization term. The results, presented in Table 2, are calculated on an Apple M1 Pro CPU. As expected, we observe a higher runtime for our method due to the computational cost of estimating the MMD from 200 samples of simulated data. The total runtime also depends on the number of batchsize N_{batch}, hence, as N_{batch} increases, the proportion of runtime used for estimating MMD reduces. As a result, we see that for large N_{batch}, the increase in the computational cost of our method with robust statistics is not significant.

C Details of the radio propagation experiment

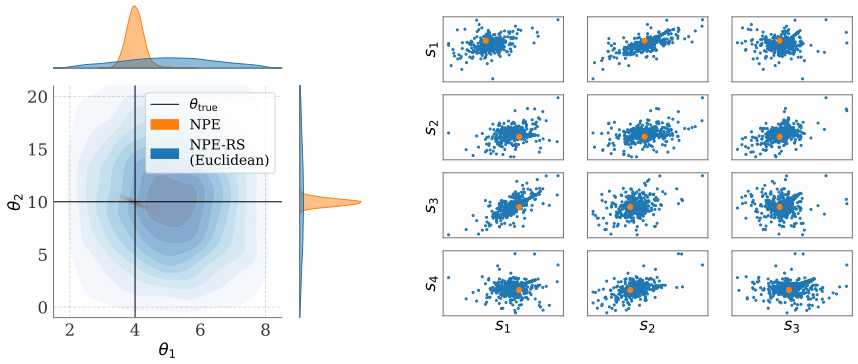
In this section, we describe the data and the Turin model used in Section 5 of the main text.

Data and model description. Let B be the frequency bandwidth used to measure radio channel data at K equidistant points, leading to a frequency separation of $\Delta f = B/(K-1)$. The measured transfer function at kth point, Y_k, is modelled as

$$Y_k = H_k + W_k, \quad k = 0, 1, \ldots, K-1,$$

where H_k is the transfer function at the kth frequency, and W_k is additive zero-mean complex circular symmetric Gaussian noise with variance σ_W^2. Taking the inverse Fourier transform, the time-domain signal $y(t)$ can be obtained as

$$y(t) = \frac{1}{K} \sum_{k=0}^{K-1} Y_i \exp(j2\pi k \Delta f t).$$



(a) Posteriors (b) Summary statistics

Figure 5: **Ricker model.** Posteriors and summary statistics for $\mathcal{D}$ being the Euclidean distance.

Table 2: Comparison of computational costs across different models on Ricker model. We report the mean value (standard deviation) derived from 20 updates. We use different batch size N_{batch} and generate 100 realizations for each θ.

	Runtime (seconds)		
	$N_{\text{batch}} = 50$	$N_{\text{batch}} = 100$	$N_{\text{batch}} = 200$
NPE	0.22 (0.03)	0.46 (0.04)	0.87 (0.03)
NPE-RS (ours)	1.26 (0.05)	1.53 (0.14)	1.92 (0.10)
ABC	0.68 (0.04)	1.41 (0.04)	3.29 (0.27)
ABC-RS (ours)	1.79 (0.04)	2.71 (0.25)	4.25 (0.46)

The Turin model defines the transfer function as $H_k = \sum_l \alpha_l \exp(-j2\pi\Delta f k \tau_l)$, where τ_l is the time-delay and α_l is the complex gain of the l^{th} component. The arrival time of the delays is modelled as one-dimensional homogeneous Poisson point processes, i.e., $\tau_l \sim \text{PPP}(\mathbb{R}_+, \Lambda)$, with $\Lambda > 0$. The gains conditioned on the delays are modelled as iid zero-mean complex Gaussian random variables with conditional variance $\mathbb{E}[|\alpha_l|^2|\tau_l] = G_0 \exp(-\tau_l/T)/\Lambda$. The parameters of the model are $\theta = [G_0, T, \nu, \sigma_W^2]^\top$. The prior ranges used for the parameters are given in Table 3.

Table 3: Prior distributions for the parameters of the Turin model.

	G_0	T	Λ	σ_W^2
Prior	$\mathcal{U}(10^{-9}, 10^{-8})$	$\mathcal{U}(10^{-9}, 10^{-8})$	$\mathcal{U}(10^7, 5 \times 10^9)$	$\mathcal{U}(10^{-10}, 10^{-9})$

The radio channel data from [2] is collected in a small conference room of dimensions $3 \times 4 \times 3$ m^3, using a vector network analyzer. The measurement was performed with a bandwidth of $B = 4$ GHz, and $K = 801$. Denote each complex-valued time-series by $\tilde{\mathbf{y}} \in \mathbb{R}^K$, and the whole data-set by $\tilde{\mathbf{y}}_{1:n}$, where $n = 100$ realizations. We take the input to the summary network to be $\mathbf{y}_{1:n} = 10 \log_{10}(|\tilde{\mathbf{y}}_{1:n}|^2)$.

Scatter-plot of learned statistics. In Figure 6 and Figure 7, we show the scatter-plots of the learned statistics using the NPE and our NPE-RS method, respectively. We observe that the observed statistics (shown in orange) is often outside the set of simulated statistics (shown in blue) for the NPE method. Hence, the inference network is forced to generalize outside its training distribution, which leads to poor fit of the model, as shown in Section 5 of the main text. On the other hand, the observed statistic is always inside the set of simulated statistics (or the training distribution) for our method in Figure 7, which leads to robustness against model misspecification.

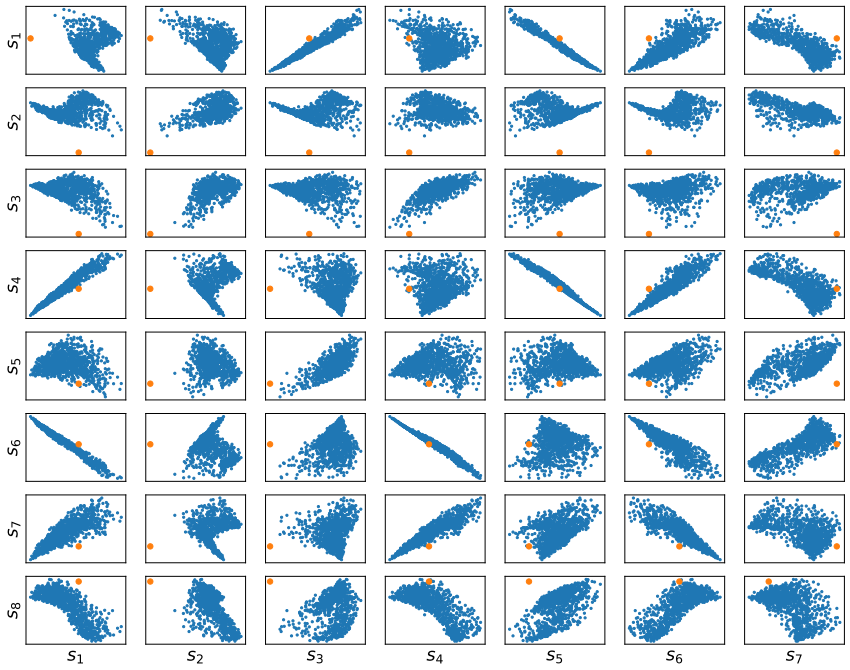


Figure 6: Pairwise scatter-plots of summary statistics learned using NPE method for the Turin model. Each blue dot corresponds to simulated statistic obtained from a parameter value sampled from the prior. The orange dot represents the observed statistic.

References

[1] D. Greenberg, M. Nonnenmacher, and J. Macke. Automatic posterior transformation for likelihood-free inference. In *International Conference on Machine Learning*, volume 97, pages 2404–2414, 2019.

[2] C. Gustafson, D. Bolin, and F. Tufvesson. Modeling the polarimetric mm-wave propagation channel using censored measurements. In *2016 Global Communications Conference*. IEEE, 2016.

[3] G. Papamakarios, T. Pavlakou, and I. Murray. Masked autoregressive flow for density estimation. *Advances in neural information processing systems*, 30, 2017.

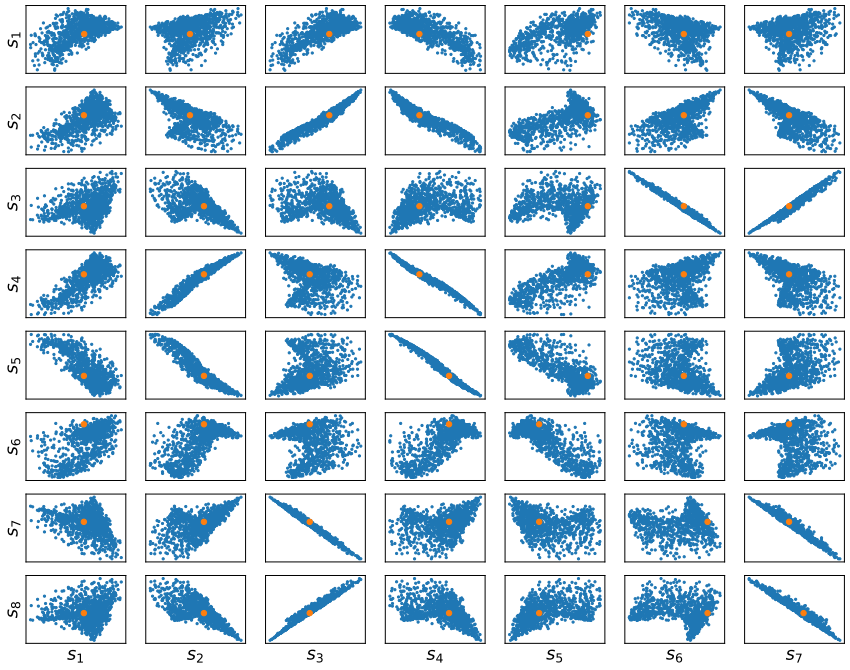


Figure 7: Pairwise scatter-plots of summary statistics learned using our NPE-RS method for the Turin model. Each blue dot corresponds to simulated statistic obtained from a parameter value sampled from the prior. The orange dot represents the observed statistic.