Kernel Tests for Markov Chain Monte Carlo Methods

Andrew Wei Jiang[*][†]

Supervisors: Arthur Gretton, Peter Orbanz, Heishiro Kanagawa

September 2020

[*] **Disclaimer:** This report is submitted as part requirement for the MSc in Computational Statistics and Machine Learning at UCL. It is substantially the result of my own work except where explicitly indicated in the text. The report may be freely copied and distributed provided the source is explicitly acknowledged.

[†] **Acknowledgements:** I am greatly indebted to my supervisors, Arthur Gretton, Peter Orbanz, and Heishiro Kanagawa, for their patience and wisdom in challenging times.

Abstract

In Bayesian statistics, Markov chain Monte Carlo (MCMC) algorithms provide flexible ways to sample from analytically intractable posterior distributions. However, inference based on MCMC methods suffers from two sources of error: lack of convergence and mistakes in implementation. While there are a variety of methods to detect the former, less attention has been devoted to the latter. We propose two two-sample tests for diagnosing MCMC implementation errors based on Maximum Mean Discrepancy (MMD). The MMD tests exhibit lower Type I/II error rates than the commonly used Geweke test [Geweke, 2004] and perform competitively with recently proposed alternatives in several experiments. On the other hand, the computational costs of both MMD tests are quadratic in the number of observations compared to linear or log-linear for existing tests. Finally, we provide some evidence that, unlike existing alternatives, the MMD tests generalize well to domains other than $\mathbb{R}^d$. Source code is available at `https://github.com/awjiang/mcmc-kernel-tests-msc`.

Contents

Note on Notation

Unless otherwise noted, probability distributions and random variables are always denoted by uppercase unbolded letters (X). In $\mathbb{R}^d$, unbolded lowercase symbols (x) denote scalars, bolded lowercase symbols ($\mathbf{x}$) denote vectors, and bolded uppercase symbols ($\mathbf{X}$) denote matrices. In other domains, unbolded symbols (x) are used generically to refer to constituent elements.

1 Introduction

In Bayesian statistics, given the prior distribution of the parameters $P(\mathbf{\Theta})$ and the likelihood of the observed data $P(\mathbf{Y}|\mathbf{\Theta})$, we are interested in sampling from the posterior distribution

$$P(\theta|y) = \frac{P(y|\theta)P(\theta)}{P(y)} \propto P(y|\theta)P(\theta)$$

for inference. When the normalizer $P(y)$ is analytically intractable, directly sampling from $P(\theta|y)$ is impossible. Instead, Markov chain Monte Carlo (MCMC) algorithms draw dependent samples from a Markov chain with $P(\theta|y)$ as its stationary distribution, only requiring the density to be known up to a constant of proportionality. On the other hand, their stochastic nature makes MCMC algorithms potentially difficult to work with. For instance, valid inference requires the eponymous Markov chain to converge to its stationary distribution, but the user may stop the algorithm before it has 'burned in'. To avoid this pitfall, a variety of convergence diagnostics have been proposed in the literature. However, fewer studies have addressed the more fundamental issue of determining whether the implementation itself is correct; even if the algorithm has converged, it may not have the correct stationary distribution. Subtle errors can escape inspection by eye, particularly when the underlying algorithm is complex; worse still, they may subsequently propagate through the literature. For example [Del Negro and Primiceri, 2015] and [Karlsson, 2017] correct two highly cited MCMC algorithms in the econometrics literature — over a decade after their initial publications. Thus, the development of principled tests to verify MCMC correctness is of significant importance to researchers.

There are a handful of MCMC correctness tests in the literature; in fact, the errors in [Del Negro and Primiceri, 2015] and [Karlsson, 2017] were initially detected using the Geweke test [Geweke, 2004]. However, these tests rely on testing a finite family of hypotheses based on scalar quantities derived from samples. If none of the hypotheses are rejected, then the test concludes that the sampler is error-free. In many cases, testing the correctness of, for instance, all first and second moments of the parameters, may be sufficient for the user. However, these tests will fail to detect any error not captured by the specified hypotheses. In this study, we introduce two tests based on

Maximum Mean Discrepancy (MMD) in Reproducing Kernel Hilbert Spaces (RKHS) that can in theory detect any error in an MCMC algorithm, given the right kernel. The tests perform competitively with existing alternatives in a variety of settings, though they are more expensive to conduct, with a computational cost quadratic in the number of samples compared to linear or log-linear for existing tests. We show that kernel choice and the inclusion of hand-picked features relevant to MCMC can dramatically improve test power. In addition, unlike other tests, the MMD tests can be applied natively in domains other than $\mathbb{R}^d$.

2 Background

2.1 Markov Chain Monte Carlo (MCMC)

MCMC algorithms sample from a target distribution P by drawing dependent samples from a Markov chain with the same stationary distribution. They do not require the normalizer of the distribution to be known. Under certain conditions [Jones, 2004], a Markov chain central limit theorem holds. Given a realization of a Markov chain $\{x_1, x_2, \ldots, x_N\}$ with stationary distribution P and some real-valued function $g : \mathcal{X} \to \mathbb{R}$ with finite variance, let $\mu = \mathbb{E}_P[g(X)]$ and $\hat{\mu}^{(N)} = \frac{1}{N}\sum_{i=1}^{N} g(x_i)$. Then

$$\sqrt{N}(\hat{\mu}^{(N)} - \mu) \xrightarrow{D} \mathcal{N}(0, \sigma^2)$$

Other than in their target distributions, MCMC algorithms primarily differ in how they construct their Markov chains. We illustrate several methods below in $\mathbb{R}^d$.

2.1.1 The Metropolis-Hastings Algorithm

The Metropolis-Hastings algorithm [Metropolis et al., 1953, Hastings, 1970] constructs a Markov chain with a desired stationary distribution P. Let $T(\mathbf{X} \to \mathbf{X}')$ denote the transition probability from state $\mathbf{X}$ to $\mathbf{X}'$ for an ergodic Markov chain. If P satisfies the detailed balance condition

$$T(\mathbf{X} \to \mathbf{X}')P(\mathbf{X}) = T(\mathbf{X}' \to \mathbf{X})P(\mathbf{X}') \tag{1}$$

then the Markov chain is reversible and has unique stationary distribution P.

At every iteration, the algorithm draws a proposal $\mathbf{X}'$ from a proposal distribution q, which is required to have non-zero measure on the support of P. $\mathbf{X}'$ is accepted with probability $A(\mathbf{X}'|\mathbf{X})$. Let $q(\mathbf{X}'|\mathbf{X})$ denote the proposal probability. Each transition probability can be decomposed into the product of the probability of proposing the next state and the probability of accepting the proposal.

$$T(\mathbf{X} \to \mathbf{X}') = q(\mathbf{X}'|\mathbf{X})A(\mathbf{X}'|\mathbf{X})$$

Rearranging (1) and setting $A(\mathbf{X}|\mathbf{X}') = 1$, the acceptance probability is

$$A(\mathbf{X}'|\mathbf{X}) = \min\left(\frac{q(\mathbf{X}|\mathbf{X}')P(\mathbf{X}')}{q(\mathbf{X}'|\mathbf{X})P(\mathbf{X})}, 1\right)$$

It is easy to see that the normalizers of $P(\mathbf{X})$ and $P(\mathbf{X}')$ cancel.

2.1.2 The Gibbs Sampling Algorithm

In the Bayesian setting characterized by (11), the Gibbs sampling algorithm, a common Metropolis-Hastings variant, updates one parameter (or a block of parameters) Θ_i at a time using the proposal distribution

$$q(\mathbf{\Theta}'|\mathbf{\Theta}, \mathbf{Y}) = P(\Theta_i'|\mathbf{\Theta}_{\neg i}, \mathbf{Y})$$

Each proposal $\mathbf{\Theta}' = (\Theta_i', \mathbf{\Theta}_{\neg i})$ is always accepted, since

$$\begin{aligned} A(\mathbf{\Theta}'|\mathbf{\Theta}) &= \min\left(\frac{q(\mathbf{\Theta}|\mathbf{\Theta}', \mathbf{Y})P(\mathbf{\Theta}'|\mathbf{Y})}{q(\mathbf{\Theta}'|\mathbf{\Theta}, \mathbf{Y})P(\mathbf{\Theta}|\mathbf{Y})}, 1\right) \\ &= \min\left(\frac{P(\Theta_i|\mathbf{\Theta}_{\neg i}, \mathbf{Y})P(\mathbf{\Theta}'|\mathbf{Y})}{P(\Theta_i'|\mathbf{\Theta}_{\neg i}, \mathbf{Y})P(\mathbf{\Theta}|\mathbf{Y})}, 1\right) \\ &= \min\left(\frac{P(\mathbf{\Theta}, \mathbf{Y})P(\mathbf{\Theta}', \mathbf{Y})}{P(\mathbf{\Theta}', \mathbf{Y})P(\mathbf{\Theta}, \mathbf{Y})}, 1\right) = 1 \end{aligned}$$

A single iteration of the Gibbs sampler sweeps through all of the parameters. Note that while each individual update satisfies detailed balance, the entire sweep generally does not, unless the updates are conducted in palindromic or random order [Geyer, 1992]. An example of a Gibbs sampler can be found in experiment 1.

2.1.3 The Reversible-jump Algorithm

The reversible jump algorithm [Green, 1995] generalizes the Metropolis-Hastings algorithm to allow 'jumps' between states with different dimensions. This is particularly useful in a Bayesian model averaging context. An illustration can be found in [Brooks et al., 2011], section 1.2: suppose we propose a move from state $(j, \mathbf{\Theta}_j)$ in model $\mathcal{M}_j$ with dimension d_j to state $(j', \mathbf{\Theta}_{j'})$ in model $\mathcal{M}_{j'}$ with dimension $d_{j'} > d_j$. This is accomplished by augmenting the lower-dimensional state $(j, \mathbf{\Theta}_j)$ with a random vector $\mathbf{u} \sim \pi(\mathbf{u})$ of length $d_{j'} - d_j$ such that the dimensions of both states match. The augmented state is then mapped one-to-one to the proposal through a function $g_{j \to j'} : \mathbb{R}^{d_j} \times \mathbb{R}^{d_{j'} - d_j} \to \mathbb{R}^{d_{j'}}$. Let $q(j \to j')$ denote the probability of making the proposal. Then the acceptance probability is

$$A((j', \mathbf{\Theta}'_{j'})|(j, \mathbf{\Theta}_j)) = \min\left(\frac{P(j', \mathbf{\Theta}'_{j'} \mid \mathbf{Y}) q(j' \to j)}{P(j, \mathbf{\Theta}_j \mid \mathbf{Y}) q(j \to j') \pi(\mathbf{u})} \left|\frac{\partial g_{j \to j'}(\mathbf{\Theta}_j, \mathbf{u})}{\partial(\mathbf{\Theta}_j, \mathbf{u})}\right|, 1\right)$$

$\left|\frac{\partial g_{j \to j'}(\mathbf{\Theta}_j, \mathbf{u})}{\partial(\mathbf{\Theta}_j, \mathbf{u})}\right|$ is called the Jacobian in the literature. In practice, g is often just the identity function, with the Jacobian equal to one. The reverse jump $(j', \mathbf{\Theta}'_{j'}) \to (j, \mathbf{\Theta}_j)$ is similar, but uses g^{-1} rather than g. When the dimensionality of the proposal matches the dimensionality of the current state, we have Metropolis-Hastings. See experiment 3 for an example.

2.2 Statistical Hypothesis Testing

This section follows chapter 8 of [Casella and Berger, 1990] and chapter 9 of [Lehmann and Romano, 2005].

DEFINITION 1 (Two-sample Test) *Given metric space (M, d), let P and Q be two Borel probability measures defined on M, and let random variables $X \sim P$ and $Y \sim Q$. Given samples $\{x_i\}_{i=1}^{N_x}$, $\{y_i\}_{i=1}^{N_y}$, a statistical test $\mathcal{T}(\{x_i\}_{i=1}^{N_x}, \{y_i\}_{i=1}^{N_y}) : M^{N_X} \times M^{N_Y} \to \{0, 1\}$ distinguishes between the null hypothesis $H_0 : P = Q$ and the alternative hypothesis $H_1 : P \neq Q$. We call testing H_0 against H_1 the two-sample problem.*

$\mathcal{T}$ compares a test statistic s of the samples to a threshold c_α, which is determined by design parameter (significance level) α. If the statistic exceeds the threshold, then $\mathcal{T}$ rejects the null hypothesis; otherwise, it fails to reject

H_0. Thus, the rejection region is $S_\alpha = \{s|s > c_\alpha\}$. Put another way, the test rejects H_0 when its p-value $p(s) = \inf\{\alpha|s \in S_\alpha\}$ is less than α.

DEFINITION 2 (Type I and Type II Errors) *A Type I error is made when H_0 is true, but rejected by test $\mathcal{T}$. A Type II error is made when H_1 is true, but $\mathcal{T}$ fails to reject H_0.*

If $\mathcal{T}$ is consistent, in the large-sample limit, its Type I error rate is upper bounded by α and its Type II error rate is zero. Consistency is essential for valid inference. Another commonly used metric is test power, defined as $1 - \beta$, where β is the Type II error rate. Tests that require fewer samples to achieve a given Type II error rate are more efficient than others and thus, all else equal, preferable.

2.2.1 Multiple Testing Corrections

DEFINITION 3 (Family-wise Error Rate) *Given a family of N tests $\{\mathcal{T}_i\}_{i=1}^N$, the Family-wise Error Rate (FWER) is the probability that at least one test commits a Type I error.*

When testing multiple hypotheses, the FWER is analogous to the Type I error rate of a single test. When comparing families of tests to single tests later on, we will refer to both as Type I error rates.

For a fixed significance level α, the FWER grows with the size of the associated family of tests — however, this behavior can result in the inconsistency of the family. There are numerous procedures to address this. The simplest, known as the Bonferroni procedure [Bonferroni, 1935], divides the significance level used for each individual test by the number of tests in the family.

Direct control of the FWER tends to reduce test power. Alternative procedures sacrifice some control of the FWER to combat this; these methods instead target the False Discovery Rate (FDR), defined as the proportion of rejected null hypotheses that are incorrectly rejected. One such approach is the Benjamini-Hochberg procedure [Benjamini and Hochberg, 1995]

1. Calculate p-values $\{p_i\}_{i=1}^N$, sorted in ascending order, for family $\{\mathcal{T}_i\}_{i=1}^N$
2. Calculate $k = \sup\{i|\frac{i}{N}p_k \leq \alpha\}$

3. Reject $\mathcal{H}_0^i$ for $i \leq k$

In this study, we will use the Benjamini-Hochberg procedure for all multiple testing corrections.

2.3 Reproducing Kernel Hilbert Space (RKHS)

This section follows chapter 4 of [Steinwart and Christmann, 2008].

DEFINITION 4 (Hilbert Space) *A vector space $\mathcal{H}$ over $\mathbb{R}$ is called a Hilbert Space if there exists a function $\langle \cdot, \cdot \rangle : \mathcal{H} \times \mathcal{H} \to \mathbb{R}$ satisfying $\forall f, f_1, f_2, g \in \mathcal{H}$*

1. $\langle \alpha_1 f_1 + \alpha_2 f_2, g \rangle_{\mathcal{H}} = \alpha_1 \langle f_1, g \rangle_{\mathcal{H}} + \alpha_2 \langle f_2, g \rangle_{\mathcal{H}}$
2. $\langle f, g \rangle_{\mathcal{H}} = \langle g, f \rangle_{\mathcal{H}}$
3. $\langle f, f \rangle_{\mathcal{H}} \geq 0$ *and* $\langle f, f \rangle_{\mathcal{H}} = 0$ *iff* $f = 0$

$\langle \cdot, \cdot \rangle$ *is called an inner product.*

DEFINITION 5 (Kernel) *Given a be a non-empty set $\mathcal{X}$, a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a kernel if there exists a Hilbert space $\mathcal{H}$ of real-valued functions on $\mathcal{X}$ and a feature map $\phi : \mathcal{X} \to \mathcal{H}$ such that*

$$k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}} \quad \forall x, x' \in \mathcal{X}$$

Since $\phi(x) \in \mathcal{H}$ is a function, we may write $\phi(x) = k(\cdot, x)$. $k(\cdot, x)$ is called the canonical feature map.

Sums and products of kernels are also kernels. Two examples of commonly used kernels in $\mathbb{R}^d$ are the linear kernel

$$k_{\text{linear}}(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}' \rangle$$

and the Gaussian kernel, also called the RBF (radial basis function) kernel

$$k_{\text{RBF}}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{\sigma^2}\right), \quad \sigma > 0 \tag{2}$$

where σ is called the bandwidth. The behavior of k_{RBF} is highly dependent on σ. $k_{\text{RBF}}(\mathbf{x}, \mathbf{x})$ is always equal to 1. As $\sigma \to \infty$, $k_{\text{RBF}}(\mathbf{x}, \mathbf{x}') \to 1$ for all $\mathbf{x}, \mathbf{x}'$; informally, the kernel becomes insensitive to differences between $\mathbf{x}$ and $\mathbf{x}'$. Conversely, as $\sigma \to 0$, $k_{\text{RBF}}(\mathbf{x}, \mathbf{x}') \to 0$ for $\mathbf{x} \neq \mathbf{x}'$; the kernel becomes very sensitive. Bandwidth selection is an area of ongoing research; for example, see [Sutherland et al., 2019].

DEFINITION 6 (Reproducing Kernel Hilbert Space) *A Hilbert space $\mathcal{H}$ is a Reproducing Kernel Hilbert Space (RKHS) if, for an associated kernel k*

1. $k(\cdot, x) \in \mathcal{H} \quad \forall x \in \mathcal{X}$
2. $\langle f, k(\cdot, x)\rangle_{\mathcal{H}} = f(x) \quad \forall f \in \mathcal{H}, x \in \mathcal{X}$ *(reproducing property)*

$k(\cdot, \cdot)$ *is called a reproducing kernel.*

DEFINITION 7 (Universal Kernel) *Let $\mathcal{Z}$ be a compact subset of $\mathcal{X}$, and let $C(\mathcal{Z})$ denote the space of all continuous functions from $\mathcal{Z}$ to $\mathbb{R}$. A continuous*[1] *reproducing kernel k on $\mathcal{Z}$ is universal if $\forall f \in C(\mathcal{Z}), \epsilon > 0$, there exists a function in the associated RKHS $g \in \mathcal{H}$ such that*

$$\|f - g\|_\infty \leq \epsilon$$

The associated RKHS of a universal kernel is called a universal RKHS [Steinwart, 2001].

This notion of universality can be generalized to non-compact spaces [Sriperumbudur et al., 2011]. In particular, the Gaussian kernel is universal on $\mathbb{R}^d$.

DEFINITION 8 (Mean embedding) *The mean embedding μ_P in RKHS $\mathcal{H}$ of a distribution P is defined as*

$$\mu_P(t) = \mathbb{E}_P[k(X, t)]$$

In particular $\mu_P \in \mathcal{H}$ satisfies

$$\mathbb{E}_P[f(X)] = \langle \mu_P, f\rangle_{\mathcal{H}}$$

The empirical estimate of the mean embedding is

$$\hat{\mu}_P(t) = \frac{1}{N}\sum_{i=1}^{N} \phi(x_i)$$

[1]By [Steinwart, 2001] Lemma 3, a kernel is continuous iff its feature map is continuous

2.4 Maximum Mean Discrepancy (MMD)

This section follows [Gretton et al., 2012]. The workhorse of the tests we will present later is

DEFINITION 9 (Maximum Mean Discrepancy (MMD)) *Let $\mathcal{F}$ be a space of functions $f : \mathcal{X} \to \mathbb{R}$, where $\mathcal{X}$ is a non-empty set. Then the Maximum Mean Discrepancy (MMD) is defined as*

$$\mathrm{MMD}[\mathcal{F}, P, Q] = \sup_{f \in \mathcal{F}} \left(\mathbf{E}_X[f(X)] - \mathbf{E}_Y[f(Y)] \right) \tag{3}$$

A biased estimate of the MMD using samples $\{x_i\}_{i=1}^{N_x}$, $\{y_i\}_{i=1}^{N_y}$ is

$$\widehat{\mathrm{MMD}}_V[\mathcal{F}, \{x_i\}_{i=1}^{N_x}, \{y_i\}_{i=1}^{N_y}] := \sup_{f \in \mathcal{F}} \left(\frac{1}{N_x} \sum_{i=1}^{N_x} f(x_i) - \frac{1}{N_y} \sum_{i=1}^{N_y} f(y_i) \right)$$

When $\mathcal{F}$ is a unit ball on an RKHS $\mathcal{H}$ with associated kernel $k(\cdot, \cdot)$, the MMD admits the closed form

$$\mathrm{MMD}[\mathcal{F}, P, Q] = \|\mu_P - \mu_Q\|_{\mathcal{F}}$$

and the biased empirical MMD can be computed via

$$\widehat{\mathrm{MMD}}_V^2 = \|\hat{\mu}_P - \hat{\mu_Q}\|_{\mathcal{F}} = \frac{1}{N_x^2} \sum_{i=1}^{N_x} \sum_{j=1}^{N_x} k(x_i, x_j) + \frac{1}{N_y^2} \sum_{i=1}^{N_y} \sum_{j=1}^{N_y} k(y_i, y_j) - \frac{2}{N_x N_y} \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} k(x_i, y_j) \tag{4}$$

which is the sum of two V-statistics and a sample average. An unbiased estimate can be obtained by exchanging the V-statistics for U-statistics

$$\widehat{\mathrm{MMD}}_U^2 = \frac{1}{\binom{N_x}{2}} \sum_{i=1}^{N_x} \sum_{i \neq i'}^{N_x} k(x_i, x_{i'}) + \frac{1}{\binom{N_y}{2}} \sum_{j=1}^{N_y} \sum_{j \neq j'}^{N_y} k(y_j, y_{j'}) - \frac{2}{N_x N_y} \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} k(x_i, y_j) \tag{5}$$

When $N_x = N_y = N$, (4) and (5) can be written as one-sample V- and U-statistics, respectively

$$\widehat{\mathrm{MMD}}_V^2 = \frac{1}{N^2} \sum_{i,j} h(z_i, z_j) \tag{6}$$

$$\widehat{\mathrm{MMD}}_U^2 = \frac{1}{N(N-1)} \sum_{i \neq j} h(z_i, z_j)$$

where $Z = (X, Y) \sim P \times Q$ and $h(z_i, z_j) = k(x_i, x_j) + k(y_i, y_j) - k(x_i, y_j) - k(x_j, y_i)$. These estimates can be computed in quadratic time, and we will make use of them here. It should be noted that linear time versions also exist; however, their speed comes at the cost of higher variance and reduced power of related tests [Gretton et al., 2012].

For a certain set of reproducing kernels, (3) is a metric on the space of probability distributions M, i.e. $\mathrm{MMD}[\mathcal{H}, P, Q]$ iff $P = Q$. Kernels that satisfy this property are called characteristic, and all universal kernels are also characteristic [Sriperumbudur et al., 2010].

2.4.1 Asymptotics

Assuming a bounded kernel, the limiting distributions of both test statistics as $N \to \infty$ under H_0 and H_1 can be obtained by applying the asymptotic theory of V- and U-statistics [Serfling, 2002]. In particular, tests based on MMD exploit the limiting distributions under the null hypothesis [Gretton et al., 2012, ?]. Let $\rho_x = \frac{N_x}{N_x + N_y}$ and $\rho_y = \frac{N_y}{N_x + N_y}$. Under H_0, the biased and unbiased test statistics converge to infinite sums of χ^2 random variables

$$\rho_x \rho_y \, \mathrm{MMD}_V^2[\mathcal{F}, X, Y] \xrightarrow{D} \sum_{l=1}^{\infty} \lambda_l z_r^2 \tag{7}$$

where $z_r = \left[\left(\rho_x^{1/2} \tilde{a}_l - \rho_y^{1/2} \tilde{b}_l \right)^2 \right]$, $\{\tilde{a}_r\}$ is a sequence of standard normal random variables with dependence structure based on $\{x_r\}$ and $\{\tilde{b}_r\}$ is defined analogously, swapping $\{x_r\}$ for $\{y_r\}$.

$$(N_x + N_y) \, \mathrm{MMD}_U^2[\mathcal{F}, X, Y] \xrightarrow{D} \sum_{l=1}^{\infty} \lambda_l \left[\left(\rho_x^{-1/2} a_l - \rho_y^{-1/2} b_l \right)^2 - (\rho_x \rho_y)^{-1} \right] \tag{8}$$

where $\{\lambda_l\}$ are the eigenvalues of the centered kernel $\tilde{k}(x, x') = \langle \phi(x) - \mu_p, \phi(x') - \mu_p \rangle_{\mathcal{H}}$, $a_l \sim \mathcal{N}(0, 1)$ i.i.d., and $b_l \sim \mathcal{N}(0, 1)$ i.i.d. for all $l \in \mathbb{N}$.

2.4.2 Simulating the Null Distribution

In tests based on the limiting distributions under the null in (7) and (8), we calculate test thresholds for a given significance level α by bootstrap.

When the samples are i.i.d., we can apply the permutation bootstrap. We draw N_x observations from the pooled data $\{x_1, x_2, \ldots, x_{N_x}, y_1, y_2, \ldots, y_{N_y}\}$ and call the sample $\tilde{\mathbf{x}}$; the remaining N_y observations we call $\tilde{\mathbf{y}}$. Since $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{y}}$ are drawn from the same distribution, when the number of samples is large, the resulting bootstrapped MMD is approximately distributed according to the limiting distribution under H_0. The unbiased bootstrapped MMD we will use is

$$\widehat{\mathrm{MMD}}_{U,b}^2 = \frac{1}{\binom{N_x}{2}} \sum_{i=1}^{N_x} \sum_{i \neq i'}^{N_x} k\left(\tilde{x}_i, \tilde{x}_{i'}\right) + \frac{1}{\binom{N_y}{2}} \sum_{j=1}^{N_y} \sum_{j \neq j'}^{N_y} k\left(\tilde{y}_j, \tilde{y}_{j'}\right) - \frac{2}{N_x N_y} \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} k\left(\tilde{x}_i, \tilde{y}_j\right)$$

Repeating this process many times, we can estimate the test threshold using the $1-\alpha$ empirical quantile of the bootstrapped statistics.

When the samples are not i.i.d., as in MCMC, the permutation bootstrap is inappropriate because it breaks the dependence between observations. [**?**] provide a general procedure for simulating the distribution in (7) based on the dependent wild bootstrap [Shao, 2010]. The procedure multiplies the evaluations of the kernel (not the data themselves) in (6) by randomly generated time series, implicitly generating new samples with the same dependence structure as the original data.

The main assumption is that the samples to be tested are τ-dependent

DEFINITION 10 (τ-dependence) *Let $\{Z_t, \mathcal{F}_t\}_{t \in \mathbb{N}}$ be a stationary sequence of integrable random variables defined on probability space $(\Omega, \mathcal{A}, P)$ with natural filtration $\mathcal{F}_t$. $\{Z_t, \mathcal{F}_t\}_{t \in \mathbb{N}}$ is τ-dependent if*

$$\tau(r) = \sup_{l \in \mathbb{N}} \frac{1}{l} \sup_{r \leq i_1 \leq \ldots \leq i_l} \tau\left(\mathcal{F}_0, (Z_{i_1}, \ldots, Z_{i_l})\right) \overset{r \to \infty}{\longrightarrow} 0$$

where $\tau(\mathcal{M}, X) = E\left(\sup_{g \in \Lambda} \left| \int g(t) P_{X|\mathcal{M}}(dt) - \int g(t) P_X(dt) \right|\right)$ and Λ is the set of all one-Lipschitz continuous real-valued functions on the domain of X.

In particular, Markov chains are τ-mixing under certain assumptions (see [**?**] appendix B and [Dedecker and Prieur, 2005]).

The type of time series required is called a wild bootstrap process

DEFINITION 11 (Wild bootstrap process) *A wild bootstrap process* $\mathbf{W} \in \mathbb{R}^{N\times N}$ *or* $\{W_{t,N}\}_{1\leq t\leq N}$ *is a row-wise strictly stationary, triangular array such that* $\mathbb{E}[W_{t,N}] = 0$ *and* $\sup_N \mathbb{E}[|W_{t,N}^{2+\sigma}|] < \infty$ *for some* $\sigma > 0$*. The autocovariance of the process is given by* $\mathbb{E}[W_{s,N}, W_{t,N}] = \rho(\frac{|s-t|}{l_B})$ *for some function* ρ *satisfying*

- $\lim_{u\to\infty} \rho(u) = 1$
- $\sum_{r=1}^{N-1} \rho(\frac{|r|}{l_N}) = O(l_N)$

The sequence $\{l_N\}$ *satisfies* $l_N = O(N)$ *and* $\lim_{N\to\infty} l_N = \infty$*.* $\{W_{t,N}\}_{1\leq t\leq N}$ *are* τ*-dependent with coefficients* $\tau(r) \leq C\zeta^{\frac{r}{l_N}}$ *for* $r \in \{1,\ldots,N\}, \zeta \in (0,1), C \in \mathbb{R}$*.*

Specifically, we will use the wild bootstrap process from [Leucht and Neumann, 2013, **?**]

$$W_{t,N} = \exp\left(-\frac{1}{l_N}\right)W_{t-1,N} + \sqrt{1-\exp\left(-\frac{2}{l_N}\right)}\epsilon_t \tag{9}$$

with $W_{0,N}, \epsilon_t \sim \mathcal{N}(0,1)$. For convenience, we write $W_{i,N}$ as W_i. The wild-bootstrapped MMD is

$$\widehat{\mathrm{MMD}}_{V,b}^2 = \frac{1}{N_x^2}\sum_{i=1}^{N_x}\sum_{j=1}^{N_x} W_i^{(x)}W_j^{(x)}k\left(X_i,X_j\right) + \frac{1}{N_y^2}\sum_{i=1}^{N_y}\sum_{j=1}^{N_y} W_i^{(y)}W_j^{(y)}k\left(Y_i,Y_j\right) - \frac{2}{N_xN_y}\sum_{i=1}^{n_s}\sum_{j=1}^{N_y} W_i^{(x)}W_j^{(y)}k\left(X_i,Y_j\right) \tag{10}$$

We could also replace $W_t^{(x)}, W_t^{(y)}$ with their centered versions $\tilde{W}_t^{(x)} = W_t^{(x)} - \frac{1}{N_x}\sum_{i=1}^{N_x} W_i^{(x)}, \tilde{W}_t^{(y)} = W_t^{(y)} - \frac{1}{N_y}\sum_{j=1}^{N_y} W_j^{(y)}$; however, this tends to result in weaker control of Type I errors [**?**].

Under the null hypothesis

$$\varphi\left(\rho_x\rho_y n\widehat{MMD}_V^2, \rho_x\rho_y n\widehat{MMD}_{V,b}^2\right) \xrightarrow{p} 0, \quad N\to\infty$$

where $\rho_x = \frac{N_x}{N_x+N_y}$, $\rho_y = \frac{N_y}{N_x+N_y}$.

2.5 Related Literature

2.5.1 The Geweke Test

The first and most well-known test for detecting MCMC errors (in $\mathbb{R}^d$) was proposed in [Geweke, 2004]. We will use the Geweke test as a benchmark in this study. Given the model

$$P(\mathbf{Y}, \mathbf{\Theta}) = P(\mathbf{Y}|\mathbf{\Theta})P(\mathbf{\Theta}) \tag{11}$$

The Geweke test is a two-sample test that exploits the fact that there are multiple ways to generate samples from the joint distribution $P(\mathbf{\Theta}, \mathbf{Y})$. We can draw from the prior $P(\mathbf{\Theta})$ and then the likelihood $P(\mathbf{Y}|\mathbf{\Theta})$; this is described in Algorithm 1. Alternatively, we can alternate between drawing from the likelihood $P(\mathbf{Y}|\mathbf{\Theta})$ and the posterior $P(\mathbf{\Theta}|\mathbf{Y})$; see Algorithm 2. If both algorithms are correctly implemented, the resulting empirical distributions should be indistinguishable. In this study, we will restrict ourselves to examining errors within the posterior sampler we wish to check, rather than other parts of Algorithms 1 and 2. To (indirectly) test the two-sample null hypothesis, we first define some test functions of the samples.

DEFINITION 12 (Test Function) *Let Θ denote the space of parameters and $\mathcal{Y}$ the space of the data in* (11). *A test function satisfies $g : \Theta \times \mathcal{Y} \to \mathbb{R}$ such that* $\mathrm{Var}(g(\mathbf{\Theta}, \mathbf{Y})) < \infty$.

Algorithm 1 Marginal-conditional (MC) joint simulator

```
1: Initialize $\mathbf{g}_{MC} \in \mathbb{R}_{N\times|\mathbf{g}|}$
2: for $n = 1, \dots, N$ do
3:     $\mathbf{\Theta}_n \sim P(\mathbf{\Theta})$
4:     $\mathbf{Y}_n \sim P(\mathbf{Y}|\mathbf{\Theta}_n)$
5:     $\mathbf{g}_{MC}[n, :] = \mathbf{g}(\mathbf{\Theta}_n, \mathbf{Y}_n)$
6: end for
7: return $\mathbf{g}_{MC}$
```

Algorithm 2 Successive-conditional (SC) joint simulator

```
1: Initialize $\mathbf{g}_{SC} \in \mathbb{R}_{N\times|\mathbf{g}|}$
2: $\mathbf{\Theta}_0 \sim P(\mathbf{\Theta})$
3: for $n = 1, \dots, N$ do
4:     $\mathbf{Y}_n \sim P(\mathbf{Y}|\mathbf{\Theta}_{n-1})$
5:     $\mathbf{\Theta}_n \sim$ PosteriorSampler($\mathbf{\Theta}_{n-1}, \mathbf{Y}_n$)
6:     $\mathbf{g}_{SC}[n, :] = \mathbf{g}(\mathbf{\Theta}_n, \mathbf{Y}_n)$
7: end for
8: return $\mathbf{g}_{SC}$
```

Let $\mathbf{g}$ denote a vector of test functions. For each element g of $\mathbf{g}$, the Geweke test compares two estimates of $\bar{g} = \mathbb{E}[g(\mathbf{\Theta}, \mathbf{Y})]$ using samples from

Algorithms 1 and 2.

$$\frac{\hat{\bar{g}}_{MC} - \hat{\bar{g}}_{SC}}{\sqrt{\frac{\hat{\sigma}^2_{MC}}{N_{MC}} + \frac{\hat{\sigma}^2_{SC}}{N_{SC}}}} \xrightarrow{d} \mathcal{N}(0,1) \tag{12}$$

with the mean estimates

$$\hat{\bar{g}}_{MC} = \frac{1}{N_{MC}} \sum_{n=1}^{N_{MC}} g_{MC}^{(n)}, \qquad \hat{\bar{g}}_{SC} = \frac{1}{N_{SC}} \sum_{n=1}^{N_{SC}} g_{SC}^{(n)}$$

The variance estimate for the marginal-conditional samples is straightforward

$$\hat{\sigma}^2_{MC} = \frac{1}{N_{MC}} \sum_{n=1}^{N_{MC}} (g_{MC}^{(n)} - \hat{\bar{g}}_{MC})^2$$

However, the successive-conditional variance estimate is not so simple, since the samples are dependent. Following [Geweke, 1999], we use the triangular window estimator

$$\begin{aligned}
\hat{\sigma}^2_{SC} &= \frac{1}{N_{SC}} \sum_{t=-\infty}^{\infty} w(t)\hat{\gamma}(t) \\
\hat{\gamma}(t) = \hat{\gamma}(-t) &= \frac{1}{N_{SC}} \sum_{i=1}^{N_{SC}-t} (g_{SC}^{i} - \hat{\bar{g}}_{SC})(g_{SC}^{i+t} - \hat{\bar{g}}_{SC}) \\
w(t) &= \max\left(\frac{L-|t|}{L}, 0\right), \quad L \in \{0.04, 0.08, 0.15\} \times N
\end{aligned}$$

See [Priestley, 1981] for more on window functions.

Each test function thus receives a z-score; if at least one z-score deviates too much from zero, then we reject the null hypothesis. For a given significance level α, the testing procedure is

1. Define test functions $\mathbf{g}$
2. Draw $\mathbf{g}_{MC}, \mathbf{g}_{SC}$
3. Calculate the left side z_j of (12) for each test function element of $\mathbf{g}_{MC}, \mathbf{g}_{SC}$
4. If any $|z_j| \geq \Phi^{-1}(1 - \frac{\alpha^*}{2})$, reject the null hypothesis that the distributions are the same

where α^* is determined by the multiple testing correction. The computational cost of the Geweke test is linear in the number of observations, $O(N)$.

A reasonable selection of test functions is the set of all first and second empirical moments of the parameters; this is the approach taken by the original paper. The successive nature of Algorithm 2 ensures that if the marginal distribution of the sampled parameters is correct, then the marginal distribution of the sampled data is as well.

2.5.2 Alternative Tests

There are a handful of two-sample and parametric alternatives to the Geweke test.

The two-sample approach found in [Grosse and Duvenaud, 2014], like the Geweke test, uses Algorithms 1 and 2 to generate $\mathbf{g}(\mathbf{\Theta}_n, \mathbf{Y}_n)$. Rather than conducting a formal test, though, they advocate diagnosing errors visually. They generate PP plots of the corresponding test functions from each simulator. If for any of the plots, the points are far from the unit line, then they reject the null hypothesis that the posterior sampler is error-free.

[Cook et al., 2006] take a parametric approach, under the assumption that the posterior is absolutely continuous. Each iteration of their algorithm draws $\mathbf{\Theta}_0$ from the prior $P(\mathbf{\Theta})$ and $\mathbf{Y}$ from the likelihood $P(\mathbf{Y}|\mathbf{\Theta}_0)$, and then draws $\{\mathbf{\Theta}_\ell\}_{\ell=1}^L$ from the posterior $P(\mathbf{\Theta}_0)$ using the to-be-checked posterior sampler. The algorithm then calculates the empirical quantile q of $\mathbf{\Theta}_0$ among $\{\mathbf{\Theta}_\ell\}_{\ell=1}^L$

$$q = \frac{1}{L}\sum_{\ell=1}^{L} \mathbb{I}_{\mathbf{\Theta}_\ell > \mathbf{\Theta}_0}$$

The test relies on the claim that, if the posterior sampler is correct, q should be uniformly distributed, and $\Phi^{-1}(q) \sim \mathcal{N}(0,1)$, where Φ^{-1} is the inverse standard normal CDF. Unfortunately, this claim is incorrect in general, due to discretization error and dependence among $\{\theta_\ell\}_{\ell=1}^L$ [Gelman, 2017].

[Talts et al., 2018] modify the approach in [Cook et al., 2006], computing rank statistics on test functions of the parameters rather than empirical quantiles. The Markov chain from the posterior sampler is thinned to yield L (approximately) independent samples. If the posterior sampler is correct and the posterior is continuous, then the resulting rank statistics should be uniformly distributed on $\{1, \ldots, L\}$. The authors suggest examining histograms of the rank statistics to better understand errors rather than conducting a formal test.

A recent study [Gandy and Scott, 2020] proposes two tests — one two-sample and one parametric — that we will use as benchmarks. The authors also introduce a wrapper function that, if the null hypothesis is not rejected, repeatedly runs a given test with reduced critical values, improving power while keeping false positive rates low. Though the sequential wrapper could in principle be applied in this study, it makes fair comparisons difficult, as the number of samples drawn for each test is then stochastic. We will thus focus on the tests themselves. The two-sample test is independently based on Algorithm 3, shown later. Then, the equality of the marginal distributions of each of the test functions is tested. The authors avoid prescribing a specific two-sample test, but for example, the Kolmogorov-Smirnov test may be used for continuous parameters, and a likelihood ratio test may be used in the discrete case. The parametric test in [Gandy and Scott, 2020] is similar to [Talts et al., 2018], but does not require independent samples or a continuous posterior. Instead, it requires only reversibility. Each iteration of the algorithm draws $M \sim \mathrm{Unif}(\{1, \ldots, L\})$, and then draws $\mathbf{\Theta}_M$ from the prior $P(\mathbf{\Theta})$ and and $\mathbf{Y}$ from the likelihood $P(\mathbf{Y}|\mathbf{\Theta}_0)$. The posterior sampler is then run forward to generate $\{\mathbf{\Theta}_\ell\}_{\ell=M+1}^{L}$ and backward to generate $\{\mathbf{\Theta}_\ell\}_{\ell=1}^{M-1}$. If the posterior is correct, then the rank statistic of $\mathbf{g}(\mathbf{Y}, \mathbf{\Theta}_M)$ among $\{\mathbf{g}(\mathbf{Y}, \mathbf{\Theta}_\ell)\}_{\ell=1}^{\ell=L}$ should be uniformly distributed. This is verified using a χ^2 test. The two-sample test using Kolmogorov-Smirnov and the Rank test both involve sorting; the former compares the empirical CDFs of the test functions and the latter computes rank statistics. This means that their computational complexities are $O(N \log N)$.

3 MMD Tests

Existing tests suffer from several major limitations. First, they are unable to directly test the null hypothesis of interest, i.e., the joint distributions $P(\mathbf{Y}, \mathbf{\Theta})$ and $Q(\mathbf{Y}, \mathbf{\Theta})$ are the same. Instead, they require us to first anticipate dimensions in which the distributions may differ, and then test hypotheses concerning those dimensions only. If we select too few test functions, the number of errors we cannot detect will be too high. On the other hand, if we select too many test functions, correcting for multiple testing may hamstring test power. Second, the tests are limited to probability distributions on $\mathbb{R}^d$. Posterior samplers of distributions on other domains such as graphs cannot be checked without significant feature engineering. Third, drawing enough samples from Algorithm 2 to offset their serial dependence may require significant computation that is difficult to parallelize beyond low-level matrix operations, though this dependence may also benefit test power by magnifying subtle errors in the sampler [Grosse and Duvenaud, 2014].

We propose two two-sample MMD-based tests that address the first two limitations. Instead of manually specifying many test functions, we specify a kernel, and test functions (features) are implicit. Rather than conducting many indirect tests to approximately determine if two joint distributions are equal, we can test the null hypothesis $P = Q$ directly by using a characteristic kernel. In addition, as long as we can define a valid kernel on the domain and run Algorithm 1, we can conveniently check any MCMC sampler, even on, for example, graphs or strings. The test based on Algorithm 3 addresses the third limitation and can be parallelized.

3.1 Successive-conditional MMD Test

The successive-conditional MMD (MMD-SC) test is similar to the Geweke test in that it uses Algorithms 1 and 2 to generate samples. The idea behind the MMD-SC test was briefly mentioned in [Lloyd and Ghahramani, 2015], and a similar test was applied in [Liu et al., 2016] as a benchmark for model goodness-of-fit tests. However, our MMD-SC test differs in several ways. Most importantly, [Liu et al., 2016] incorrectly uses the unbiased MMD test statistic (5) with permutation bootstrapped null distribution. Because the MCMC samples are dependent, the permutation bootstrap assumption that the samples are i.i.d. is violated. Instead, we use the biased MMD test statistic (4) with wild bootstrapped null distribution, which allows for dependent samples [**?**]. The second difference is that the other test's MCMC algorithm requires a burn-in period to reach the stationary distribution, while Algorithm 2 is initialized in the stationary distribution and no burn-in is required.

In this study, we set $l_N = 20$ for the wild bootstrap process (9). For significance level α and B bootstrap samples, the testing procedure is

1. Define test functions $\mathbf{g}(\mathbf{\Theta}, \mathbf{Y})$
2. Draw $\mathbf{g}_{MC}, \mathbf{g}_{SC}$ using Algorithms 1 and 2 and normalize scale

3. Calculate $\widehat{\mathrm{MMD}}_V^2$ on $\mathbf{g}_{MC}, \mathbf{g}_{SC}$

4. Simulate $\{\rho_1\rho_2 n\widehat{\mathrm{MMD}}_{V,b}^2\}_{b=1}^B$ using (10) and calculate c_α, their $1-\alpha$ empirical quantile

5. If $\rho_1\rho_2 n\widehat{MMD}_V^2 \geq c_\alpha$, reject the null hypothesis that the distributions are the same

Note that if the MCMC sampler is reversible, we can run two instances of Algorithm 2 at once. In order to draw N samples, we can initialize a starting sample, index it by $\frac{N+1}{2}$, then run the chain backward on one thread to generate samples $\mathbf{g}^{(\frac{N-1}{2})}, \mathbf{g}^{(\frac{N-3}{2})}, \ldots, \mathbf{g}^{(1)}$, and run it forward on another thread for samples $\mathbf{g}^{(\frac{N+3}{2})}, \mathbf{g}^{(\frac{N+5}{2})}, \ldots, \mathbf{g}^{N}$.

3.2 Backward-conditional MMD Test

A major disadvantage of the successive-conditional sampler is that it cannot be parallelized because of the autocorrelation in the samples. The backward-conditional MMD (MMD-BC) test sidesteps this by drawing independent samples from the joint distribution $P(\mathbf{\Theta}, \mathbf{Y})$. To draw a single sample, we first draw $\mathbf{Y}_i$ from its marginal distribution, then run the posterior simulator to get $\mathbf{\Theta}_i$ after some burn-in. The entire process is displayed in Algorithm 3.

Algorithm 3 Backward-conditional (BC) joint simulator

```
1: Initialize $\mathbf{g}_{BC} \in \mathbb{R}_{N\times|\mathbf{g}|}$
2: for $n = 1, \ldots, N_{BC}$ do
3:     $\mathbf{\Theta}_0 \sim P(\mathbf{\Theta})$
4:     $\mathbf{Y}_n \sim P(\mathbf{Y}|\mathbf{\Theta}_0)$
5:     for $m = 1, \ldots, M$ do
6:         $\mathbf{\Theta}_n \sim \mathrm{PosteriorSampler}(\mathbf{\Theta}_n, \mathbf{Y}_n)$
7:     end for
8:     $\mathbf{g}_{BC}[n, :] = \mathbf{g}(\mathbf{\Theta}_n, \mathbf{Y}_n)$
9: end for
10: return $\mathbf{g}_{BC}$
```

Since the samples are independent, we can then apply a test based on the unbiased MMD, using permutation bootstrapping to generate the null distribution as described in [Gretton et al., 2012]. For a significance level α and B bootstrap samples, the testing procedure is

1. Define test functions $\mathbf{g}(\mathbf{\Theta}, \mathbf{Y})$
2. Draw $\mathbf{g}_{MC}, \mathbf{g}_{BC}$ via Algorithms 1 and 3 and normalize scale
3. Calculate $\widehat{\mathrm{MMD}}_U^2$ on $\mathbf{g}_{MC}, \mathbf{g}_{BC}$
4. Simulate the null distribution of $\widehat{\mathrm{MMD}}_U^2$ via permutation and calculate the $1-\alpha$ empirical quantile c_α
5. If $\widehat{\mathrm{MMD}}_U^2 \geq c_\alpha$, reject the null hypothesis that the distributions are the same

3.3 Kernel Selection

In this study, since all our experiments can be cast as tests in $\mathbb{R}^d$, we will use the Gaussian kernel, which is characteristic on that domain. There are a variety of methods in the literature for selecting the Gaussian kernel bandwidth. Though optimizing the t-statistic in [Sutherland et al., 2019] generally outperforms other methods for kernel parameter selection in terms of test power, this approach is relatively data-intensive, requiring a training set on which the kernel parameters are learned. Because we assume drawing samples is expensive, we use the median heuristic, a common choice in the literature, choosing the median pairwise L2 norm in the pooled samples as the bandwidth.

If the feature scales of the sample differ by too much, the median heuristic may be inappropriate. Thus, before applying the heuristic, we first normalize the feature scales by dividing each sample feature by the corresponding pooled standard deviation.

3.4 Test Functions

The MMD tests benefit from the inclusion of MCMC-specific test functions. In our experiments, following [Gandy and Scott, 2020], we augment the samples with two additional test functions — the evaluations of the likelihood $P(\mathbf{Y}|\mathbf{\Theta})$ and the prior $P(\mathbf{\Theta})$ on the sample. These features make intuitive sense in the MCMC setting. Under an incorrect posterior distribution, certain parameter values will be more or less likely to be observed than they would be under the correct posterior; since $P(\mathbf{\Theta}|\mathbf{Y}) \propto P(\mathbf{Y}|\mathbf{\Theta})P(\mathbf{\Theta})$, these discrepancies should manifest in the likelihood and prior.

We use the test functions

$$\mathbf{g}(\boldsymbol{\Theta}, \mathbf{Y}) = \begin{bmatrix} \boldsymbol{\Theta} & P(\mathbf{Y}|\boldsymbol{\Theta}) & P(\boldsymbol{\Theta}) \end{bmatrix}^\top \tag{13}$$

which mirror the test functions used in [Gandy and Scott, 2020], but exclude higher moments of the parameters. As we will see in experiment 1, the additional features can have a dramatic effect on test power.

Strictly speaking, for $\mathbf{x} = (\boldsymbol{\Theta}, \mathbf{Y})$ and $\mathbf{x}'$ similarly defined, we are using the kernel

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{g}(\mathbf{x}) - \mathbf{g}(\mathbf{x}')\|^2}{\sigma^2}\right)$$

rather than the true Gaussian kernel (2). This kernel is characteristic for distributions of test functions, but not guaranteed to be characteristic on the space of joint distributions under (13). If the user is concerned about this, then they may set $\mathbf{g}(\boldsymbol{\Theta}, \mathbf{Y}) = \begin{bmatrix} \mathbf{Y} & \boldsymbol{\Theta} & P(\mathbf{Y}|\boldsymbol{\Theta}) & P(\boldsymbol{\Theta}) \end{bmatrix}^\top$ for a consistent joint test. In practice, however, (13) scales better with the dimensionality of the data, avoiding consequent power loss [Reddi et al., 2014], while being almost as good at error detection. Due to the structure of Algorithms 2 and 3, if the marginal distribution of the sampled parameters is correct, then the marginal distribution of the sampled data is as well. In addition, non-pathological errors in the dependence between $\mathbf{Y}$ and $\boldsymbol{\Theta}$ will manifest in the test function $P(\mathbf{Y}|\boldsymbol{\Theta})$.

4 Experiments

We will conduct three experiments to compare the performance of the successive-conditional and backward-conditional MMD tests (MMD-SC test and MMD-BC test, respectively) to the Geweke test as well as other benchmark tests from [Gandy and Scott, 2020], where appropriate.

When comparing tests, it is important to consider their varying computational costs as well as their Type I/II error rates. These costs can be decomposed into two components: the cost of drawing samples and the cost of running the test. While we would ideally calculate the performance of each test keeping total computational cost fixed, this can be quite difficult. It is tedious to enumerate all elementary operations involved, and even if we complete this task, our implementations are likely suboptimal; if we try to sidestep this issue by using CPU wall time as a proxy as in [Gandy and Scott, 2020], artificial discrepancies may arise from thermal throttling, specific libraries used, etc.

We focus on keeping sampling cost approximately fixed across tests by counting the total number of samples they require from Algorithms 1, 2, or 3. To illustrate, assume we allocate a sample budget of 1800 to a test that draws samples using Algorithms 1 and 2. Assume we thin the samples from Algorithm 2 such that 1 out of every 5 samples are kept. We want the test to receive an equal number of samples from each Algorithm after thinning. Thus, we draw 300 samples from Algorithm 1 and 1500 samples from Algorithm 2 for a total of 1800, thin the latter, and finally conduct the test on 300 samples from each algorithm. This corresponds to a 'Test Sample Size' of 300 in the figures below. We perform an analogous procedure for tests using Algorithm 3, replacing thinning with burn-in.

4.1 Gibbs Sampling

We first evaluate test performance on the Gibbs sampler from [Gandy and Scott, 2020]. The data generating process is

$$\begin{aligned} \theta_i &\sim \mathcal{N}(0, \sigma^2), \quad i = 1, 2 \\ y &= \theta_1 + \theta_2 + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2) \end{aligned}$$

By conjugacy, the Gibbs updates for θ are

$$\theta_i \sim \mathcal{N}\left(\frac{\sigma^2}{\sigma_\epsilon^2 + \sigma^2}(y - \theta_j), \frac{1}{\frac{1}{\sigma_\epsilon^2} + \frac{1}{\sigma^2}}\right), \quad i = 1, 2$$

The hyperparameters are set to $\sigma^2 = 100, \sigma_\epsilon^2 = 0.1$. Each iteration of the Gibbs sampler updates θ_1 and θ_2 in random order to preserve reversibility, which is required for the Rank test.

Since the three errors introduced in [Gandy and Scott, 2020] are all either too easy to detect or unclear, we introduce two errors into the posterior sampler illustrating the importance of the auxiliary features. In the first error, the posterior distribution means are swapped. This is a plausible but subtle error; because the parameters are symmetric, the marginal distribution of the parameters is unchanged, and two-sample tests based solely on this marginal will fail to detect any error. In the second error, we replace the normal distributions in the Gibbs steps with Laplace distributions preserving the correct means and variance. These errors are summarized in Table 1.

Error	Components	Incorrect	Correct
Mean Swap	Means	$\frac{\sigma^2}{\sigma_\epsilon^2+\sigma^2}(y-\theta_i)$	$\frac{\sigma^2}{\sigma_\epsilon^2+\sigma^2}(y-\theta_j)$
Laplace	Distributions	Laplace	Gaussian

Table 1: Experiment 1 intentional errors in posterior sampler

First, we compare the performance of the MMD-BC test with the median-heuristic Gaussian kernel against the KS and Rank tests from [Gandy and Scott, 2020]. Each Type I/II error rate is calculated over 100 trials with $\alpha = 0.05$ and all tests use the same sets of independent samples from Algorithms 1 and 3, with $M = 5$ for Algorithm 3. The test functions used in each test are shown in Table 2. Note that the MMD tests do not use higher moments of the parameters. The results are plotted in Figure 1.

Test	θ_1	θ_2	θ_1^2	$\theta_1\theta_2$	$P(\mathbf{y}\|\theta)$	$P(\theta)$
MMD-SC	Y	Y			Y	Y
MMD-BC	Y	Y			Y	Y
Geweke	Y	Y	Y	Y	Y	Y
KS	Y	Y	Y	Y	Y	Y
Rank	Y	Y	Y	Y	Y	Y

Table 2: Experiment 1 test functions

While the Type I error rates hover around the design parameter, without the auxiliary test functions, the Type II error rates for the MMD-BC and KS tests, shown in green, are stuck just below 1. Under the Mean Swap error, the marginal distributions of θ_1 and θ_2 are unchanged, so the two-sample tests relying solely on θ test functions ($\{\theta_1, \theta_2\}$ for the MMD-BC test and $\{\theta_1, \theta_2, \theta_1^2, \theta_1\theta_2\}$ for the KS test) cannot detect this error. In the Laplace error, the posterior mean and variance are correct; because the θ-KS test only tests up to the second moments, it cannot detect this error either. Surprisingly, while the θ-MMD-BC test should be able to detect the Laplace error due the the characteristic nature of the Gaussian kernel, it seems to require a very high number of samples to do so.

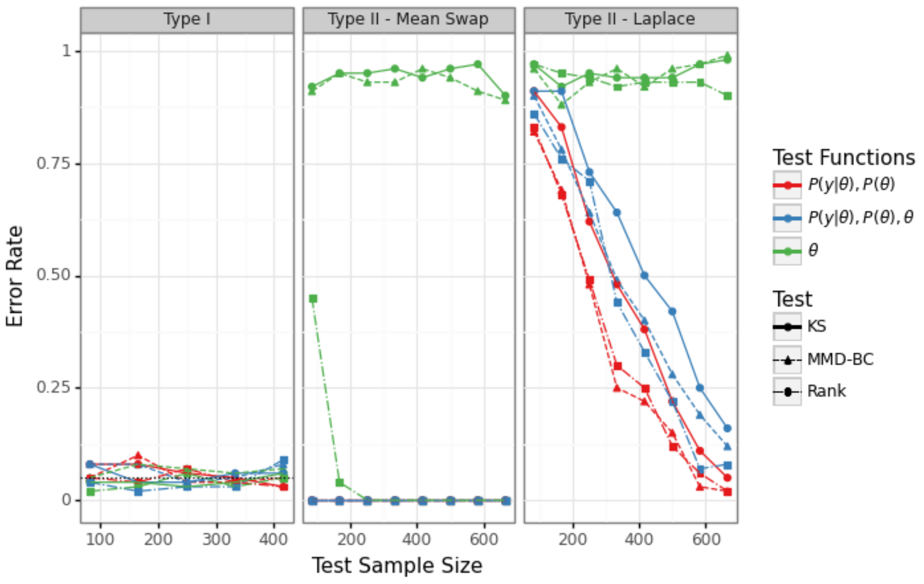


Figure 1: Experiment 1 Type I/II error rates of BC tests with varying test functions, calculated over 100 trials with $\alpha = 0.05$. In Algorithm 3, we set $M = 5$. For the MMD-BC test, θ test functions refer to $\{\theta_1, \theta_2\}$; for non-MMD tests, θ test functions refer to $\{\theta_1, \theta_2, \theta_1^2, \theta_1\theta_2\}$. The MMD-BC test used the Gaussian kernel with bandwidth set using the median heuristic, with scale-normalized data. For the Rank test, we ordered test functions directly. For the MMD-BC and KS tests, 'Test Sample Size' refers to the size of each sample seen by the tests. Though the Rank test is not a two-sample test, at a given 'Test Sample Size', it runs roughly the same number of sampler iterations as the KS and MMD-BC tests.

These errors were chosen to be difficult to detect in θ-space. Consequently, the auxiliary test functions appear to capture the greatest discrepancies between samples. In the Laplace Error, the inclusion of the θ test functions on top of the auxiliary test functions visibly increases the Type II error across all tests. This illustrates the trade-off inherent in enumerating test functions. Loosely speaking, each additional test function drastically improves detection rates for errors along that dimension, but reduces detection rates for discrepancies along orthogonal or sufficiently 'distant' dimensions. In the Geweke and KS tests, this reduction in detection rates is driven by the multiple testing correction applied. For the MMD tests, this reduction occurs because the similarity between observations increases when irrelevant features are added.

The likelihood captures underlying structure in the model in a way that is otherwise difficult to characterize. We also attempted directly testing the equality of the joint distributions $P(\mathbf{Y}, \mathbf{\Theta})$ using the MMD tests using the Gaussian kernel with median heuristic (on scale-normalized samples). To be explicit, the test functions we used for this exercise (not shown) were $\mathbf{g}(\mathbf{\Theta}, \mathbf{Y}) = \begin{bmatrix}\mathbf{Y} & \mathbf{\Theta}\end{bmatrix}^\top$, rather than (13). This approach was unable to detect the errors given the same computational budgets as in Figure 1. Computing separate Gaussian kernels on the data and parameters and using their sum or product as the test kernel was also ineffective.

To the extent that the benchmark tests rely on a finite set of test functions, they are only able to examine discrepancies in a limited number of nonlinear features. In contrast, MMD-based tests are able to account for an infinite number of nonlinear features when using specific nonlinear kernels. Figure 2 illustrates the resulting benefits to power for the MMD-BC test.

The autocorrelation in the samples from Algorithm 2 is extreme; though the Type II error rates are close to zero for the given hyperparameters, the SC tests require far more samples than the BC tests to push the Type I error rate below $\alpha = 0.05$. Comparing efficiency across test types is not meaningful for the default parameters. However, we can reduce the autocorrelation by increasing σ_ϵ to obtain a fairer comparison. Turning this knob will allow us to analyze how the tests behave given samplers with varying mixing speeds. In Figure 3, we plot the Type I/II error rates for the correct sampler and the Laplace error sampler, holding both the total sampler iterations used by each test and the test sample sizes fixed. As we might expect, when the sample

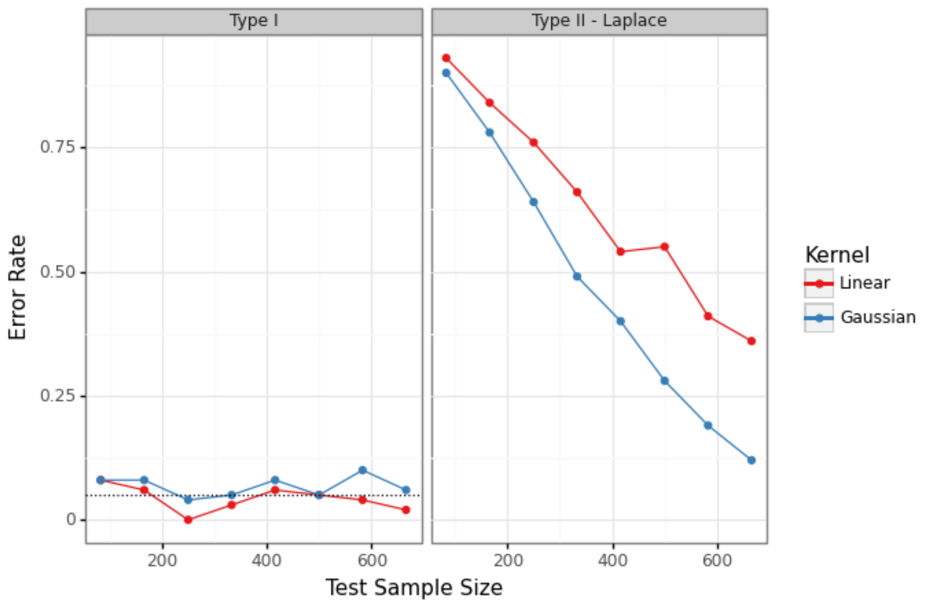


Figure 2: Experiment 1 MMD-BC Type I/II error rates using $\boldsymbol{\Theta}$ and the likelihood and prior auxiliary variables as test functions, calculated over 100 trials with $\alpha = 0.05$. Algorithm 3 set $M = 5$. We used Gaussian kernels with median heuristic bandwidths on scale-normalized data.

autocorrelation is very high, the SC tests always reject the null hypothesis. As σ_ϵ increases and the autocorrelation falls, the SC Type I error rates fall. In this plot, the SC Type I error rates generally exceed the BC test Type I rates; this is caused by high autocorrelation in the successive-conditional chains. However, note that for a given σ_ϵ, we can reduce the MMD-SC and Geweke Type I error rates by thinning the chains.

As the autocorrelation decreases and mixing improves, the Type II error rates tend to decrease across the board. The exception to this trend is the initial spike experienced by the SC tests. As we move right from the leftmost extreme ($\sigma_\epsilon = 0.1$), the autocorrelation weakens enough that the SC tests sometimes fail to reject the null. Notably, the MMD-SC test exhibits better control of Type I error rates than the Geweke test here, despite the fact that both tests use Algorithm 2. On the other hand, the MMD-SC test is also less likely to correctly reject the null hypothesis. For $\sigma_\epsilon \geq 2.5$, the MMD-SC test has similar Type II error rates to the MMD-BC test. The KS test's Type II error rates lag behind the other tests in this setting.

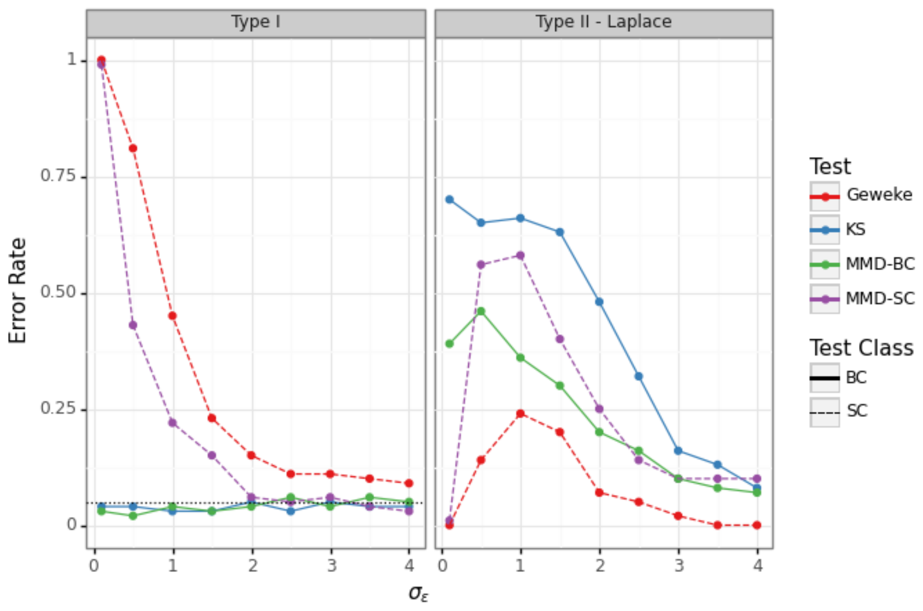


Figure 3: Experiment 1 Type I/II error rates of BC and SC tests against σ_ϵ, calculated over 100 trials with $\alpha = 0.05$. As σ_ϵ increases, autocorrelation in the successive-conditional samples used in the Geweke and MMD-SC tests decreases. We allocated a budget of 1800 sampling iterations, keeping every 5th sample; each test received 300 samples from each sampler.

4.2 Reversible-jump Bayesian Lasso

This example is based on [Chen et al., 2011], but avoids using improper (hyper)priors in order to allow us to run Algorithm 1. We are interested in learning the parsimonious linear regression model

$$\mathbf{y} = \mathbf{X}\beta + \epsilon, \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$$

in a Bayesian setting. Rather than introducing an L1 penalty to the objective function with a corresponding regularization parameter as in frequentist Lasso, we instead promote sparsity by placing a truncated Poisson prior on the number ℓ of nonzero coefficients β_j, drawing ℓ coefficients from a Laplace distribution, and setting the remaining $p - \ell$ coefficients to zero. In addition, we place an inverse-gamma prior on σ^2. Let $\mathbf{\Theta} = \{\lambda, \sigma, \ell, \gamma, \beta\}$ denote the parameters, $\mathbf{X} \in \mathbb{R}^{n \times p}$ denote the (fixed) design matrix, and

$\mathbf{y} \in \mathbb{R}^{n\times 1}$ denote the data. The prior is

$$P(\boldsymbol{\Theta}|\tau, a, b) = P(\ell|\lambda)P(\gamma|\ell)P(\beta_j|\tau,\gamma)P(\sigma^2|a,b)$$

$$P(\ell|\lambda) = \frac{\exp(-\lambda)\lambda^\ell}{C\ell!}, \quad \ell \in \{1,\dots,p\}$$

$$P(\gamma|\ell) = \binom{p}{\ell}^{-1}$$

$$P(\beta_j|\tau,\gamma) = \begin{cases} (2\tau)^{-1}\exp(-\frac{|\beta_j|}{\tau}) & j \in \gamma \\ \delta(\beta_j) & \text{otherwise} \end{cases}$$

$$P(\sigma^2|a,b) = \frac{b^a}{\Gamma(a)}(\sigma^2)^{-a-1}\exp\left(-\frac{b}{\sigma^2}\right)$$

where C is a normalization constant, δ is the Dirac delta function, and γ is a vector of the nonzero indices of β. The likelihood is given by

$$P(\mathbf{y}|\sigma,\beta,\mathbf{X}) = (2\pi)^{-\frac{n}{2}}\sigma^{-n}\exp\left(-\frac{\|\mathbf{y}-\mathbf{X}\beta\|_2^2}{2\sigma^2}\right)$$

The joint probability is then

$$P(\mathbf{y},\boldsymbol{\Theta}|\mathbf{X}) \propto \sigma^{-n}\exp\left(-\frac{\|\mathbf{y}-\mathbf{X}\beta\|_2^2}{2\sigma^2}\right)\times$$
$$\frac{\exp(-\lambda)\lambda^\ell}{\ell!}\binom{p}{\ell}^{-1}\prod_{j\in\gamma}(2\tau)^{-1}\exp\left(-\frac{|\beta_j|}{\tau}\right)\prod_{j'\notin\gamma}\delta(\beta_{j'})\frac{b^a}{\Gamma(a)}(\sigma^2)^{-a-1}\exp\left(-\frac{b}{\sigma^2}\right)$$

Each iteration of the reversible-jump MCMC posterior sampler takes two steps in random order. The first is a Gibbs step, which is straightforward due to the conjugacy of the Inverse Gamma prior on σ^2.

$$\sigma^2|\mathbf{y},\mathbf{X},\beta \sim \mathcal{IG}\left(a+\frac{n}{2}, b+\frac{\sum_{i=1}^n(y_i-\mathbf{x}_i\beta)^2}{2}\right)$$

where $\mathbf{x}_i$ denotes row i of $\mathbf{X}$.

The second is a more involved reversible jump step. We start by proposing $\ell' \in \{\ell-1,\ell,\ell+1\}$ uniformly at random, disallowing $\ell < 1$ and $\ell > p$. Thus, when $\ell \in \{1,p\}$, there are only two valid proposals, not three. Then, depending on the ℓ' proposed, we complete the proposal $\boldsymbol{\Theta}'$ via one of the following

- Update: $\ell' = \ell$
    1. Choose $j \in \{1, \ldots, \ell\}$ uniformly at random
    2. Propose $\gamma' = \gamma, \beta'_j = \beta_j + \mathcal{N}(0, \epsilon_{\text{update}}), \beta'_{i \neq j} = \beta_i$
    3. $P(\mathbf{\Theta}' \to \mathbf{\Theta}) = P(\mathbf{\Theta} \to \mathbf{\Theta}') = \mathcal{N}(\beta'_j; \beta_j, \epsilon_{\text{update}}) = \mathcal{N}(\beta_j; \beta'_j, \epsilon_{\text{update}})$
- Birth: $\ell' = \ell + 1$
    1. Choose $j \in \{\ell + 1, \ldots, p\}$ uniformly at random
    2. Propose $\gamma' = \gamma \cup j$
    3. Propose $\beta'_j = \mathcal{N}(0, \epsilon_{\text{birth}}), \beta'_{i \neq j} = \beta_i$
    4. $P(\mathbf{\Theta}' \to \mathbf{\Theta}) = \begin{cases} \frac{1}{2}\frac{1}{\ell'} & \ell' = p \\ \frac{1}{3}\frac{1}{\ell'} & 1 < \ell < p \end{cases}$
    5. $P(\mathbf{\Theta} \to \mathbf{\Theta}') = \begin{cases} \frac{1}{2}\frac{1}{p-\ell}\mathcal{N}(\beta'_j; 0, \epsilon_{\text{birth}}) & \ell = 1 \\ \frac{1}{3}\frac{1}{p-\ell}\mathcal{N}(\beta'_j; 0, \epsilon_{\text{birth}}) & 1 < \ell < p \end{cases}$
- Death: $\ell' = \ell - 1$
    1. Choose $j \in \{1, \ldots, \ell\}$ uniformly at random
    2. Propose $\gamma' = \gamma \setminus j$
    3. Propose $\beta'_j = 0, \beta'_{i \neq j} = \beta_i$
    4. $P(\mathbf{\Theta}' \to \mathbf{\Theta}) = \begin{cases} \frac{1}{2}\frac{1}{p-\ell'}\mathcal{N}(\beta_j; 0, \epsilon_{\text{birth}}) & k' = 1 \\ \frac{1}{3}\frac{1}{p-\ell'}\mathcal{N}(\beta_j; 0, \epsilon_{\text{birth}}) & 1 < \ell' < p \end{cases}$
    5. $P(\mathbf{\Theta} \to \mathbf{\Theta}') = \begin{cases} \frac{1}{2}\frac{1}{\ell} & \ell = p \\ \frac{1}{3}\frac{1}{\ell} & 1 < \ell < p \end{cases}$

where $\epsilon_{\text{update}}, \epsilon_{\text{birth}}$ are random walk sizes. We accept proposal $\mathbf{\Theta}'$ with probability

$$A(\mathbf{\Theta}'|\mathbf{\Theta}) = \min\left(\frac{P(\mathbf{y}, \mathbf{\Theta}'|\mathbf{X})}{P(\mathbf{y}, \mathbf{\Theta}|\mathbf{X})}\frac{P(\mathbf{\Theta}' \to \mathbf{\Theta})}{P(\mathbf{\Theta} \to \mathbf{\Theta}')}, 1\right)$$

For this experiment, we fix the design matrix $\mathbf{X} \in \mathbb{R}^{1\times 3}$, with $\lambda = 1$ and hyperparameters $\tau = 1, a = 3, b = 1$. The random walk sizes are set to $\epsilon_{\text{update}} = \epsilon_{\text{birth}} = 1$.

Error	Components	Incorrect	Correct
Transition	$A_{\text{birth}}(\mathbf{\Theta}'\|\mathbf{\Theta}),\ A_{\text{death}}(\mathbf{\Theta}'\|\mathbf{\Theta})$	$\min\left(\frac{P(\mathbf{y},\mathbf{\Theta}'\|\mathbf{X})}{P(\mathbf{y},\mathbf{\Theta}\|\mathbf{X})}, 1\right)$	$\min\left(\frac{P(\mathbf{y},\mathbf{\Theta}'\|\mathbf{X})}{P(\mathbf{y},\mathbf{\Theta}\|\mathbf{X})}\frac{P(\mathbf{\Theta}'\to\mathbf{\Theta})}{P(\mathbf{\Theta}\to\mathbf{\Theta}')}, 1\right)$
Poisson	$P(\mathbf{y},\mathbf{\Theta}'\|\mathbf{X}),\ P(\mathbf{y},\mathbf{\Theta}\|\mathbf{X})$	$\ldots\frac{\exp(-\lambda)\lambda^{\ell}}{(\ell-1)!}\ldots$	$\ldots\frac{\exp(-\lambda)\lambda^{\ell}}{\ell!}\ldots$

Table 3: Experiment 2 intentional errors in posterior sampler

We introduce several intentional errors into the the posterior sampler summarized in Table 3. The first error affects the Metropolis-Hastings acceptance probabilities used in the birth/death moves; it omits the jump probabilities $P(\mathbf{\Theta}' \to \mathbf{\Theta}), P(\mathbf{\Theta} \to \mathbf{\Theta}')$ from the calculation. The second error affects all of the Metropolis-Hastings acceptance probabilities. The $P(\ell|\lambda)$ factors of the joint probabilities have incorrect denominator $(\ell - 1)!$ rather than $\ell!$. The results are shown in Figure 4, with test functions shown in Table 4.

Test	β, σ	$\{\beta, \sigma\} \times \{\beta, \sigma\}$	$P(\mathbf{y}\|\mathbf{\Theta}, \mathbf{X})$	$P(\mathbf{\Theta})$
MMD-SC	Y		Y	Y
MMD-BC	Y		Y	Y
Geweke	Y	Y	Y	Y
KS	Y	Y	Y	Y
Rank	Y	Y	Y	Y

Table 4: Experiment 2 test functions

The Type I error rates of the BC tests hover around the design parameter of $\alpha = 0.05$, while the Geweke Type I error rates are elevated, at 19-25%. The MMD-SC test has similar Type I error rates to the MMD-BC test. Note that these error rates, calculated using 100 trials, are relatively noisy.

For the two-sample tests, the Transition error is much easier to detect than the Poisson error, while the opposite is true for the Rank test. Notably, while the Geweke test detects the Transition error quite easily, it essentially is unable to detect the Poisson error at the computation levels shown. In contrast, the other tests perform reasonably well, with Type II error rates that fall relatively quickly with sample size.

Across both errors, we see that the MMD-SC test has lower Type II error rates than the MMD-BC test — further evidence of the error magnification properties of Algorithm 2. The MMD-SC test most efficiently detects the Transition error, but lags behind the KS and Rank tests in detecting the

Poisson error. The performance of the KS and Rank tests are quite consistent across both errors, with the KS test detecting the errors especially efficiently. This implies that the test functions accurately capture some of the discrepancies between samples.

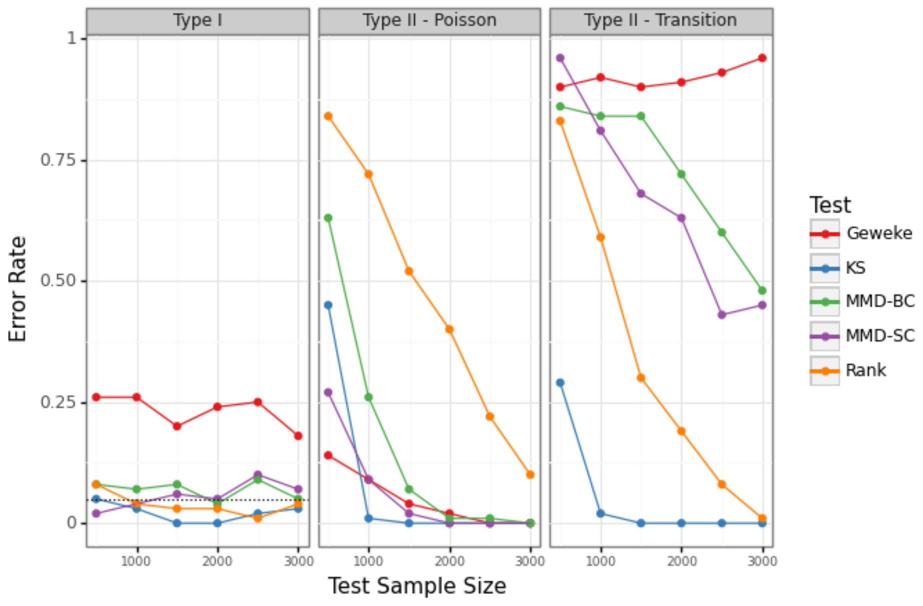
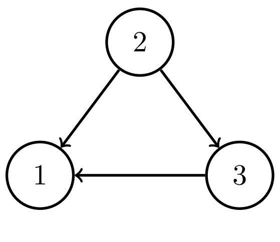


Figure 4: Experiment 2 Type I/II error rates calculated over 100 trials with $\alpha = 0.05$. Every 5th sample from Algorithm 2 was kept, and Algorithm 3 set $M = 5$. The MMD tests used Gaussian kernels with median heuristic bandwidths on scale-normalized data.

4.3 Metropolis-Hastings for Learning DAG Structure

In this experiment, we are interested in learning the structure of a directed acyclic graph (DAG) given observed data. See Figure 5 for an example of a DAG.



$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

Figure 5: Directed acyclic graph (DAG) with adjacency matrix representation. In the adjacency matrix, entry $(i, j) = 1$ indicates the presence of a directed edge from node i to node j; node i is called a parent of j and node node j is called a child of node i. A node with no parents is called a root, and a node with no children is called a leaf.

Let each observation on a root node x_r be drawn from a normal distribution with standard deviation $\epsilon = 1$, and let each child node be drawn from a normal distribution centered on the sum of its parents, also with standard deviation ϵ. In other words, given graph structure $\mathcal{G}$ and data $\mathbf{X}$

$$x_r \sim \mathcal{N}(0, \epsilon^2)$$

$$x_j|\mathbf{pa}(x_j) \sim \mathcal{N}(\sum_{z \in \mathbf{pa}(x_j)} z, \epsilon^2)$$

where $\mathbf{pa}(x)$ denotes the set of parent nodes of node x. Since the number of possible DAG structures is exponential in the number of nodes, for computational convenience, we consider all 3-node DAG structures.

Placing a uniform prior on $\mathcal{G}$, the posterior is proportional to the likelihood

$$P(\mathcal{G}|\mathbf{X}) \propto P(\mathbf{X}|\mathcal{G}) = \prod_{i=1}^{n} \prod_{j=1}^{p} P(x_{ij}|\mathbf{pa}(x_{ij})) = \prod_{i=1}^{n} \prod_{j=1}^{p} \mathcal{N}(\sum_{z_i \in \mathbf{pa}(x_{ij})} z_i, \epsilon^2)$$

The sampling algorithm we will consider is a modified version of the MCMC scheme from [Madigan et al., 1995] and is detailed in section 2 of [Grzegorczyk and Husmeier, 2008]. Given a graph structure $\mathcal{G}$, the proposal structure is sampled uniformly from the neighborhood of $\mathcal{G}$

$$P(\mathcal{G}'|\mathcal{G}) = \frac{1}{|\mathbf{Ne}(\mathcal{G})|}$$

where the neighborhood $\mathbf{Ne}(\mathcal{G})$ is defined as the union of $\mathcal{G}$ and the set of

Error	Components	Incorrect	Correct
Cyclic Check	$\lvert\mathbf{Ne}(\mathcal{G})\rvert, \lvert\mathbf{Ne}(\mathcal{G}')\rvert$	Count all graphs	Count all DAGs
Rev Count	$\lvert\mathbf{Ne}(\mathcal{G})\rvert, \lvert\mathbf{Ne}(\mathcal{G}')\rvert$	Count edge reversals twice	Count reversals once

Table 5: Experiment 3 intentional errors in posterior sampler

all DAGs that can be reached by adding, deleting, or reversing an edge.

The Metropolis-Hastings acceptance probability is thus

$$A(\mathcal{G}'|\mathcal{G}) = \min\left(1, \frac{P(\mathbf{X}|\mathcal{G}')|\mathbf{Ne}(\mathcal{G})|}{P(\mathbf{X}|\mathcal{G})|\mathbf{Ne}(\mathcal{G}')|}\right)$$

The sampler errors considered all affect the Metropolis-Hastings acceptance probability; specifically, they alter the calculation of $|\mathbf{Ne}(\mathcal{G})|$. In the first error, we count all *graphs* that can be reached by modifying a single edge, regardless of whether the modification induces a cycle. In the second error, we double-count the DAGs that can be reached by reversing an edge. These errors are summarized in Table 5.

We compare the MMD tests against the benchmark Geweke and Rank tests, with each test using the same set of manually engineered test functions (features) in a subset of $\mathbb{R}^d$. The MMD tests again use the Gaussian kernel with median heuristic bandwidth calculated on scale-normalized inputs. As a final benchmark, we include a χ^2 test of the sample DAG frequencies.

Most of the features are derived from the adjacency matrix representation of the sampled graph structure $\mathcal{G}$, shown in Figure 5. As an initial set of features, for all i, j, i', j', we take take each entry (i, j), each $\text{AND}((i, j), (i', j'))$, and each $\text{XOR}((i, j), (i', j'))$. We then prune features that are exactly equal to other features or constant by construction. First, we remove features derived from duplicate edge pairs; for example, we keep only one of $\{\text{AND}((i, j), (i', j')), \text{AND}((i', j'), (i, j))\}$. Next, diagonal entries $(i, i) = 0$ are excluded (if they are nonzero, the graph is cyclic), along with all features derived from pairs with at least one diagonal entry. We also exclude $\text{AND}((i, j), (j, i)) = 0$ and $\text{XOR}((i, j), (i, j)) = 0$. Finally, we remove $\text{AND}((i, j), (i, j)) = (i, j)$. As in the previous experiments, we also include the evaluation of the likelihood $P(\mathbf{Y}|\mathcal{G})$; the evaluation of the uniform prior $P(\mathcal{G})$ is omitted because it is constant.

The results are shown in Figure 6. Type I error rates are approximately what we would expect from the design parameter $\alpha = 0.05$.

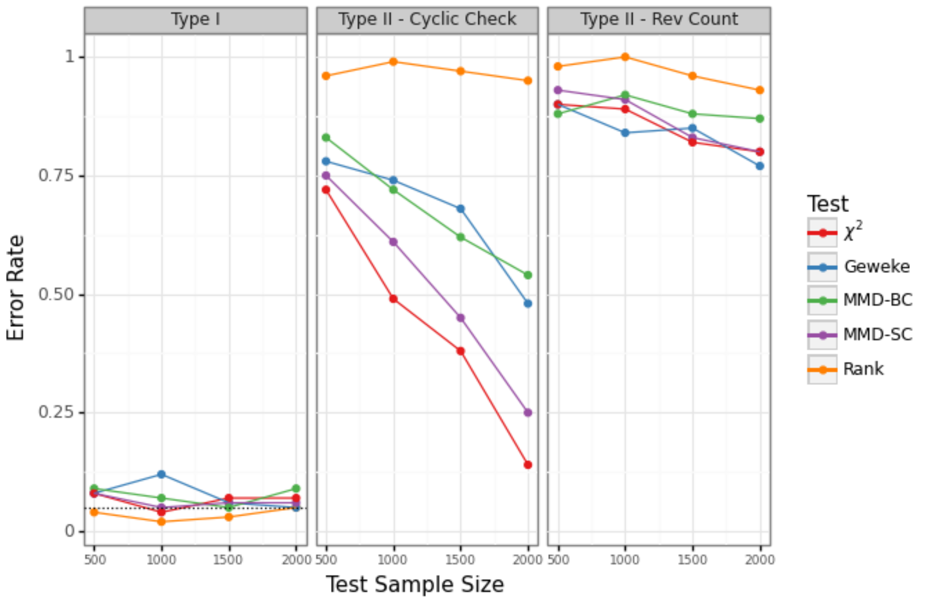


Figure 6: Experiment 3 Type I/II error rates calculated over 100 trials with $\alpha = 0.05$. Every 5th sample from Algorithm 2 was kept, and Algorithm 3 set $M = 5$. The MMD tests used Gaussian kernels with median heuristic bandwidths on scale-normalized data. For the Rank test, we ordered test functions directly using a random tiebreak.

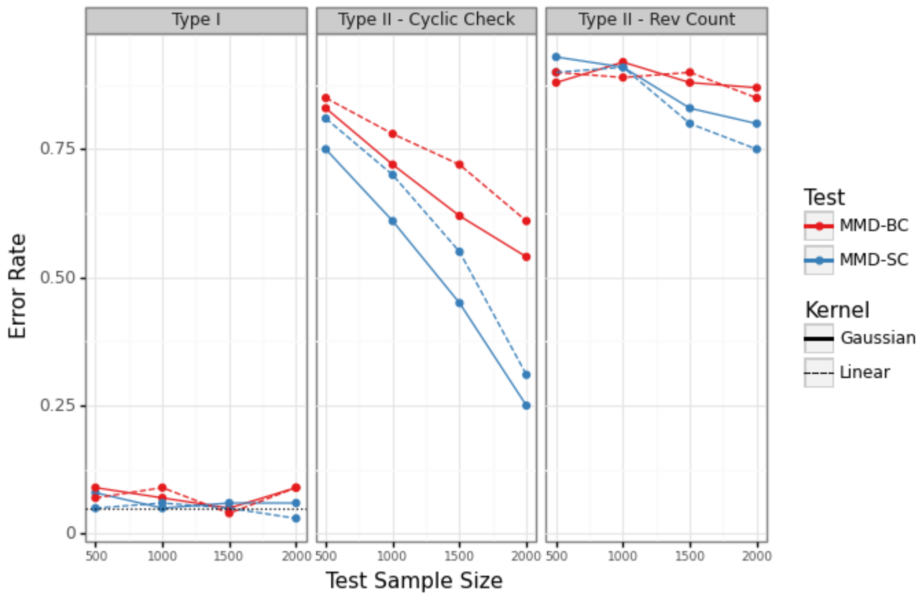


Figure 7: Experiment 3 Type I/II error rates calculated over 100 trials with $\alpha = 0.05$ for MMD tests using linear/nonlinear kernels. Every 5^{th} sample from Algorithm 2 was kept, and Algorithm 3 set $M = 5$.

It is clear that the Rank test is inappropriate for this setting; it is unable to detect either error. For the other tests, the Cyclic Check error is much easier to detect than the Rev Count error. The χ^2 test generally has the best detection rates, likely due to the fact that it operates directly on the space of DAGs rather than an incomplete representation in $\mathbb{R}^d$. Surprisingly, however, the MMD-SC test comes quite close to matching its ability to detect the Cyclic Check error. In the Rev Count error, the SC tests slightly lead the MMD-BC test in performance and match the χ^2 test. Finally, we see once more that the MMD-SC test has lower Type II error rates than the MMD-BC test.

Another question we may ask is how much the use of a nonlinear kernel in the MMD tests buys us here, given that all but one of the features used are binary. Figure 7 suggests that the gains can be substantial. In the case of the Cyclic Check error, using the Gaussian kernel improves Type II by around 8% across both tests. The effect on the Rev Count Type II error is unclear.

It is important to emphasize that the tests as formulated are generally too expensive to conduct in richer DAG spaces; see the section 5 for more details.

It is possible, however, to adapt the MMD tests to these settings by using graph kernels; see [Kriege et al., 2020] for a survey. As a proof of concept we ran this experiment using random walk kernels [Gärtner et al., 2003], [Vishwanathan et al., 2006] in Figure 8. The kernels did not include any of our auxiliary test functions; combining data from different domains is another issue entirely. Using the random walk kernel, the MMD tests are able to detect the Cyclic Check error better than the χ^2 test; however, the Rev Count Type II error degrades. In fact, random walk kernels cannot distinguish between isomorphic graphs. This illustrates a central issue with MMD; if a characteristic kernel is unavailable, MMD is not a metric on the space of probability distributions, and the test, however useful, will be inconsistent.

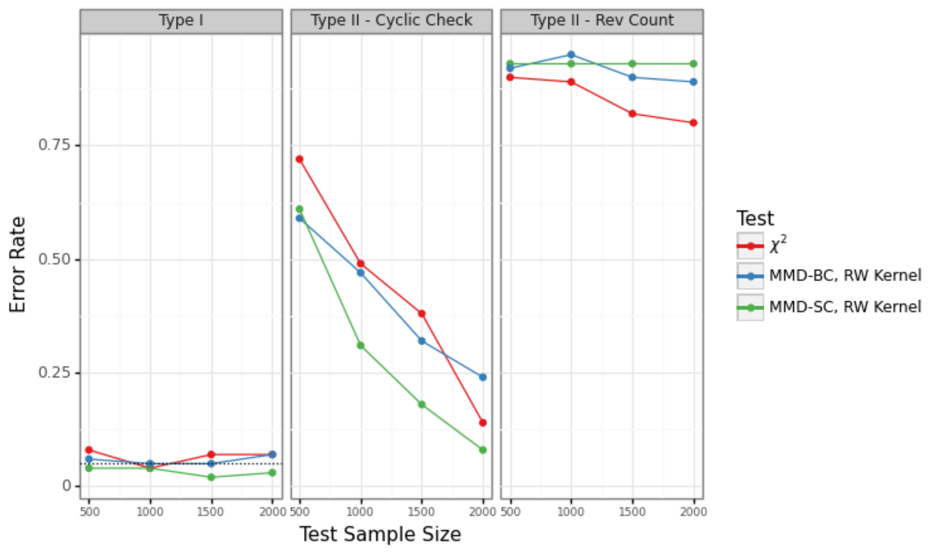


Figure 8: Experiment 3 Type I/II error rates for MMD tests using the GraKeL [Siglidis et al., 2020] random walk kernel implementation, with $\lambda = 0.1$ and 'geometric' summation. Calculated over 100 trials with $\alpha = 0.05$. Every 5th sample from Algorithm 2 was kept, and Algorithm 3 set $M = 5$.

5 Discussion

The MMD tests are preferable to existing tests from a theoretical perspective because they are able to directly test the null hypothesis that two distributions are the same, rather than approximating the null hypothesis via a family of sub-hypotheses. As we saw in experiment 1, Figure 1, for existing tests, the choice of test functions determines what errors can be detected. However, the MMD tests in a sense choose a collection of implicit test functions automatically based on their kernel. For example, a polynomial kernel of degree d encodes all first to d^{th} moments as 'test functions' (features). In addition, the mean embeddings μ_P and μ_Q in an RKHS associated with a Gaussian kernel consist of the expectations of infinitely many features under distributions P and Q, respectively. If any 'test function' differs in expectation across P and Q, it will contribute positively to the MMD. As discussed in 3.4, applying the Gaussian kernel to our particular choice of test functions (13) results in a test that is not quite consistent, but scales better with the dimensionality of the data. The tests can easily be made consistent by ensuring that the test functions contain both the parameters and the data.

The infinite dimension of the mean embeddings is a double-edged sword in the sense that testing all features at once reduces the interpretability of the test. If the null hypothesis is rejected, we do not know which features drove the rejection and we are left none the wiser about where the error might be. In contrast, testing a family of user-specified test functions non-MMD tests, may allow the user to pinpoint the block of parameters in which the error lies. For example, in experiment 2, the rejected test functions all concern β rather than σ, telling us correctly that the errors are not in the σ Gibbs step.

Naturally, the choice of kernel k has a significant impact on the performance of the MMD tests. In experiments 1 and 3, we saw that using a nonlinear kernel generally improved Type II error rates over a linear kernel for both tests; the exception was in the Rev Count error in experiment 3, where the effect was ambiguous. It is reasonable that Type II error rates would benefit from the use of the Gaussian kernel in particular; the linear kernel only takes into account features we provide to it, while the Gaussian kernel takes an infinite number of features into account. Further, unlike the linear kernel, the Gaussian kernel is universal and thus characteristic on $\mathbb{R}^d$, and any difference between two probability distributions will manifest in the associated MMD test.

We exploited the universality of the Gaussian kernel in experiment 3 by mapping graph structures to sets of binary features (test functions) in $\mathbb{R}^d$. By [Christmann and Steinwart, 2010], Theorem 2.2, as long as this mapping is one-to-one, the Gaussian kernel is universal on the embedding. Many graph kernels take a similar approach, inducing an explicit feature vector and then applying a linear kernel. The fact that applying the Gaussian kernel improves test power in experiment 3 compared to the linear kernel, as shown in Figure 7, implies that our features do not fully capture the underlying graph structures. This evokes similar results in the literature showing that using nonlinear decision boundaries on explicit feature vectors induced by graph kernels can improve performance [Kriege et al., 2020].

Further, the feature vectors used in experiment 3, however lacking, are still challenging to deal with from both computational and statistical perspectives. In experiment 3 specifically, the number of features is exponential in the number of nodes. In our case, with just over 30 features characterizing 3-node graphs, this is manageable. However, for even moderately-sized structures, just computing the feature vector for one sample becomes difficult. For example, 30-node graphs yield over 750,000 features. Even if it were feasible to compute the feature vectors for each sample, multiple testing corrections would significantly reduce test power. The χ^2 test also does not scale well; the number of possible DAGs is also exponential in the number of nodes. In contrast, the MMD tests easily scale in this setting given the right choice of graph kernel. The trade-off is instead that it is difficult to prove that kernels are characteristic on graphs. As we saw in Figure 8, the random walk kernel in particular is unable to detect the Rev Count error, while the Gaussian kernel can.

In $\mathbb{R}^d$, the MMD tests perform competitively compared to existing tests of the same class. However, they are much more expensive to conduct, with computational costs quadratic in the number of observations rather than linear or log-linear. In particular, the MMD-SC test exhibits better control of Type I error rates than the Geweke test, which also uses Algorithm 2, in experiments 1 and 2. However, as we can see in experiment 2, it is able to detect errors that the Geweke test cannot. On the other hand, in our experiments, the MMD-BC test does not have a clear performance advantage over other tests using Algorithm 3. As we might expect, the independent sample tests' Type I error rates are quite similar. However, the Type II error rates differ quite dramatically by experiment and by error. The MMD test outperforms the KS test in experiment 1, but not in experiment 2; it is outperformed by the Rank test in experiment 1 and in the Poisson error of experiment 2, but not in the transition error of experiment 2 and experiment 3.

The inconsistency in the MMD-BC test's Type II error rates relative to other tests based on Algorithm 3 may be driven by differences in the effects of increasing dimensionality. This is illustrated in the rightmost facet of Figure 1. Under the alternative hypothesis, assume a subset of test functions sufficiently captures some of the discrepancies between samples such that the corresponding null hypotheses are rejected. In the Laplace error of experiment 1, this subset of test functions is $\{P(\mathbf{y}|\theta), P(\theta)\}$. Using only this subset of test functions for hypothesis testing results in the greatest power for each test; for example, see the red lines in Figure 1. However, in practice, we do not know where the error will manifest ahead of time. The non-MMD tests lose power from testing features outside of the subset due to multiple testing corrections. As the number of features increases, MMD tests also lose power [Reddi et al., 2014] — the curse of dimensionality. In the Laplace error in particular, comparing the red and blue lines in Figure 1, we see that the MMD-BC test loses power more quickly from increasing dimensionality than the KS and Rank tests. This suggests that if the number of rejecting test functions is small relative to the total, then the MMD tests will perform poorly relative to existing tests.

In addition to the implicit features encoded in the chosen kernel, as we saw in experiment 1, including hand-picked features in the MMD test can significantly improve detection rates. The features we chose were the likelihood and prior evaluated on the sample. In particular, these capture the relationship between the data and parameters much better than including the data as test functions directly in experiment 1. The likelihood and prior are especially good choices for test functions because their dimensionality is fixed, and do not significantly reduce the power of the tests if they prove irrelevant, even when the data are very high-dimensional.

If we are given a choice between the MMD-SC and MMD-BC tests, which one should we pick? The answer depends on how much we value accuracy versus speed. The MMD-SC test is preferable for accuracy. In experiments 2 and 3, we saw that it was significantly better at detecting errors than the MMD-BC test, while keeping the Type I error rates similar. On the other hand, this improved performance comes at a potentially steep cost. The results of experiment 1, Figure 3, showed us that high autocorrelation in the samples (slow mixing in the posterior sampler) drawn by Algorithm 2 hamstrings the MMD-SC test. In order to control the Type I error rate under this regime, we must draw many more, heavily thinned samples. relative to the MMD-BC test.

In terms of speed, the MMD-BC test is the clear winner. The MMD-SC and MMD-BC tests are quite similar in raw operation count. Assume we have an equal number of samples N from Algorithms 1, 2, and 3. Let K denote the cost of computing the kernel on one pair of observations, let B denote the number of bootstrapped statistics generated. In the experiments we conducted, we kept the number of samples drawn constant. Computing the biased statistic (4) used in the MMD-SC test is $O(3KN^2)$, while computing the unbiased statistic (5) is $O(3KN^2 - 2KN)$. Computing the wild and permutation bootstrapped statistics both cost $O(3N^2)$. The main computational difference between the MMD-SC and MMD-BC tests is in their ability to be parallelized. The samples from Algorithm 2 must be drawn in sequence if they are to amplify errors; we cannot take advantage of more than two threads. In contrast, since each sample from Algorithm 3 is independent, we can use up to N threads.

6 Conclusion

In this study, we have introduced two new MMD-based tests for checking the correctness of MCMC algorithms. These tests offer several advantages over existing tests. First, the tests are correctly specified, directly testing the two-sample null hypothesis. As a result, the tests generally exhibit better sample efficiency than the commonly-used Geweke test and perform on par with newer tests, but in some cases lag behind the latter. Second, the tests are generalizable to more abstract domains such as graphs, while all current alternatives are not (in nontrivial cases). On the other hand, the MMD tests are generally more computationally intensive than competitors.

We have demonstrated the efficacy of the MMD tests in three significantly different settings — a toy Gibbs sampler, a reversible-jump Bayesian Lasso, and structure-MCMC via Metropolis-Hastings. In particular, we showed the advantages of encoding pre-existing knowledge specific to MCMC into the kernel, using test functions corresponding to the likelihood and prior. We also showed that these advantages could be further amplified by using nonlinear instead of linear kernels. Finally, we showed that the MMD tests have different properties. The user should keep in mind that the MMD test based on the successive-conditional simulator exhibits lower Type II error rates than the backwards-conditional MMD test, at the cost of potentially elevated Type I error rates when samples are highly autocorrelated.

6.1 Future Research

We see four directions for future research. The first is the construction of better comparisons across tests through tighter control of computational costs. Commonly used approximations based on CPU wall time have several disadvantages. In this study, we only controlled for the number of samples drawn to conduct each test; however, this inherently advantages the MMD tests, which are more expensive to conduct given the same sample size. More accurate comparisons between tests would allow researchers to better determine which to choose.

The second direction is finer optimization of test parameters to maximize test power. Here, we have used the median heuristic for the Gaussian kernel due to its sample efficiency. However, we may be able to do better using other methods, including those which learn the kernel parameters on a training set perform the test on a held-out test set. For instance, we might maximize the t-statistic from [Sutherland et al., 2019].

The third direction is further extension of the MCMC MMD tests to domains other than $\mathbb{R}^d$. One challenge in this area is figuring out how to combine test functions from different domains. In particular, the likelihood and prior test functions are in $\mathbb{R}^d$, while the parameters are not. While we could just ignore the likelihood and prior test functions entirely, we ideally would like to exploit the information they contain. A potential solution would be to simply compute separate kernels for the auxiliary test functions and the parameters, then use their product as the test kernel.

The final direction is generalizing our MCMC test frameworks to other methodologies. A straightforward extension is assessing approximate MCMC algorithms; for a review, see [Bardenet et al., 2015] section 6. Approximate MCMC algorithms, when the underlying dataset is large, achieve substantial speed gains over their exact counterparts by estimating acceptance ratio likelihood terms, reducing variance at the cost of introducing asymptotic bias. By simply swapping out the posterior sampler in Algorithms 1, 2, and 3 for an approximate version, our tests can be used directly to evaluate the validity of the schemes. We may also be able to apply the two-sample approach in approximate Bayesian computation (ABC) methods, which simulate the posterior via likelihood-free rejection sampling; see [Grazian and Fan, 2020] for a review.

References

[Bardenet et al., 2015] Bardenet, R., Doucet, A., and Holmes, C. (2015). On Markov chain Monte Carlo methods for tall data. *arXiv:1505.02827 [stat]*. arXiv: 1505.02827.

[Benjamini and Hochberg, 1995] Benjamini, Y. and Hochberg, Y. (1995). Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B (Methodological)*, 57(1):289–300.

[Bonferroni, 1935] Bonferroni, C. (1935). Il calcolo delle assicurazioni su gruppi di teste. pages 13–60, Rome, Italy.

[Brooks et al., 2011] Brooks, S., Gelman, A., Jones, G., and Meng, X.-L., editors (2011). *Handbook of Markov Chain Monte Carlo*. Chapman and Hall/CRC, 0 edition.

[Casella and Berger, 1990] Casella, G. and Berger, R. L. (1990). *Statistical inference*. The Wadsworth & Brooks/Cole Statistics/Probability series. Brooks/Cole Pub. Co, Pacific Grove, Calif.

[Chen et al., 2011] Chen, X., Jane Wang, Z., and McKeown, M. J. (2011). A Bayesian Lasso via reversible-jump MCMC. *Signal Processing*, 91(8):1920–1932.

[Christmann and Steinwart, 2010] Christmann, A. and Steinwart, I. (2010). Universal kernels on non-standard input spaces. In *in Advances in Neural Information Processing Systems*, pages 406–414.

[Cook et al., 2006] Cook, S. R., Gelman, A., and Rubin, D. B. (2006). Validation of Software for Bayesian Models Using Posterior Quantiles. *Journal of Computational and Graphical Statistics*, 15(3):675–692.

[Dedecker and Prieur, 2005] Dedecker, J. and Prieur, C. (2005). New dependence coefficients. Examples and applications to statistics. *Probability Theory and Related Fields*, 132:203–236.

[Del Negro and Primiceri, 2015] Del Negro, M. and Primiceri, G. E. (2015). Time varying structural vector autoregressions and monetary policy: A corrigendum. *The Review of Economic Studies*, 82(4):1342–1345.

[Gandy and Scott, 2020] Gandy, A. and Scott, J. (2020). Unit testing for MCMC and other Monte Carlo methods. *arXiv:2001.06465 [stat]*. arXiv: 2001.06465.

[Gelman, 2017] Gelman, A. (2017). Correction to Cook, Gelman, and Rubin (2006). *Journal of Computational and Graphical Statistics*, 26(4):940–940.

[Geweke, 1999] Geweke, J. (1999). Using simulation methods for bayesian econometric models: inference, development,and communication. *Econometric Reviews*, 18(1):1–73.

[Geweke, 2004] Geweke, J. (2004). Getting it right: Joint distribution tests of posterior simulators. *Journal of the American Statistical Association*, 99(467):799–804. Publisher: [American Statistical Association, Taylor & Francis, Ltd.].

[Geyer, 1992] Geyer, C. J. (1992). Practical Markov Chain Monte Carlo. *Statistical Science*, 7(4):473–483.

[Grazian and Fan, 2020] Grazian, C. and Fan, Y. (2020). A review of approximate Bayesian computation methods via density estimation: Inference for simulator-models. *WIREs Computational Statistics*, 12(4).

[Green, 1995] Green, P. J. (1995). Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82(4):711–732.

[Gretton et al., 2012] Gretton, A., Borgwardt, K. M., Rasch, M. J., Schölkopf, B., and Smola, A. (2012). A kernel two-sample test. *Journal of Machine Learning Research*, 13(25):723–773.

[Grosse and Duvenaud, 2014] Grosse, R. B. and Duvenaud, D. K. (2014). Testing MCMC code. *arXiv:1412.5218 [cs, stat]*. arXiv: 1412.5218.

[Grzegorczyk and Husmeier, 2008] Grzegorczyk, M. and Husmeier, D. (2008). Improving the structure MCMC sampler for Bayesian networks by introducing a new edge reversal move. *Machine Learning*, 71(2-3):265–305.

[Gärtner et al., 2003] Gärtner, T., Flach, P. A., and Wrobel, S. (2003). On graph kernels: Hardness results and efficient alternatives. In *COLT*.

[Hastings, 1970] Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109.

[Jones, 2004] Jones, G. L. (2004). On the Markov chain central limit theorem. *Probability Surveys*, 1(0):299–320.

[Karlsson, 2017] Karlsson, S. (2017). Corrigendum to "Bayesian reduced rank regression in econometrics" [J. Econometrics 75 (1996) 121–146]. *Journal of Econometrics*, 201(1):170–171.

[Kriege et al., 2020] Kriege, N. M., Johansson, F. D., and Morris, C. (2020). A Survey on Graph Kernels. *Applied Network Science*, 5(1):6. arXiv: 1903.11835.

[Lehmann and Romano, 2005] Lehmann, E. L. and Romano, J. P. (2005). *Testing statistical hypotheses.* Springer texts in statistics. Springer, New York, 3rd ed edition.

[Leucht and Neumann, 2013] Leucht, A. and Neumann, M. H. (2013). Dependent wild bootstrap for degenerate U - and V -statistics. *Journal of Multivariate Analysis*, 117:257–280.

[Liu et al., 2016] Liu, Q., Lee, J. D., and Jordan, M. (2016). A kernelized stein discrepancy for goodness-of-fit Tests. page 9.

[Lloyd and Ghahramani, 2015] Lloyd, J. R. and Ghahramani, Z. (2015). Statistical Model Criticism using Kernel Two Sample Tests. page 9.

[Madigan et al., 1995] Madigan, D., York, J., and Allard, D. (1995). Bayesian Graphical Models for Discrete Data. *International Statistical Review / Revue Internationale de Statistique*, 63(2):215.

[Metropolis et al., 1953] Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087–1092.

[Priestley, 1981] Priestley, M. B. (1981). *Spectral analysis and time series.* Probability and mathematical statistics. Academic Press, London ; New York.

[Reddi et al., 2014] Reddi, S. J., Ramdas, A., Póczos, B., Singh, A., and Wasserman, L. (2014). On the decreasing power of kernel and distance based nonparametric hypothesis tests in high dimensions. *arXiv:1406.2083 [cs, math, stat].* arXiv: 1406.2083.

[Serfling, 2002] Serfling, R. J. (2002). *Approximation theorems of mathematical statistics.* Wiley series in probability and statistics. Wiley, New York, NY, paperback ed edition. OCLC: 49527610.

[Shao, 2010] Shao, X. (2010). The Dependent Wild Bootstrap. *Journal of the American Statistical Association*, 105(489):218–235.

[Siglidis et al., 2020] Siglidis, G., Nikolentzos, G., Limnios, S., Giatsidis, C., Skianis, K., and Vazirgiannis, M. (2020). GraKeL: A graph kernel library in Python. *Journal of Machine Learning Research*, 21(54):1–5.

[Sriperumbudur et al., 2011] Sriperumbudur, B. K., Fukumizu, K., and Lanckriet, G. R. G. (2011). Universality, characteristic kernels and RKHS embedding of measures. *J. Mach. Learn. Res.*, 12(null):2389–2410. Publisher: JMLR.org.

[Sriperumbudur et al., 2010] Sriperumbudur, B. K., Gretton, A., Fukumizu, K., Scholkopf, B., and Lanckriet, G. R. G. (2010). Hilbert Space Embeddings and Metrics on Probability Measures. page 45.

[Steinwart, 2001] Steinwart, I. (2001). On the Inuence of the Kernel on the Consistency of Support Vector Machines. page 27.

[Steinwart and Christmann, 2008] Steinwart, I. and Christmann, A. (2008). *Support Vector Machines.* Information Science and Statistics. Springer New York, New York, NY. ISSN: 1613-9011.

[Sutherland et al., 2019] Sutherland, D. J., Tung, H.-Y., Strathmann, H., De, S., Ramdas, A., Smola, A., and Gretton, A. (2019). Generative

Models and Model Criticism via Optimized Maximum Mean Discrepancy. *arXiv:1611.04488 [cs, stat]*. arXiv: 1611.04488.

[Talts et al., 2018] Talts, S., Betancourt, M., Simpson, D., Vehtari, A., and Gelman, A. (2018). Validating bayesian inference algorithms with simulation-based calibration. *arXiv:1804.06788 [stat]*. arXiv: 1804.06788.

[Vishwanathan et al., 2006] Vishwanathan, S. V. N., Borgwardt, K. M., and Schraudolph, N. N. (2006). Fast computation of graph kernels. In *Proceedings of the 19th International Conference on Neural Information Processing Systems*, NIPS'06, pages 1449–1456, Cambridge, MA, USA. MIT Press. event-place: Canada.