Contents

1 Introduction

In recent years, kernel machine-based hypothesis tests (e.g. SKAT) for high-dimensional, nonlinear effects has seen widespread application in GWAS and gene-environment interaction studies. However, constructing a test for the interaction between groups of continuous features (for example, interaction between groups of air pollutants and multicategory nutrition intake) remains difficult in practice. The main challenges root from (1) constructing an appropriate main-effect kernel that induce unbiased estimator for the null model, and (2) constructing an appropriate interaction-effect kernel describing only the effect of between-groups interaction, which is necessary for building a valid test statistic. Recently, [1] addressed the first challenge by proposing cross-validated ensemble of kernels (CVEK), an ensemble-based estimator that adaptively learn the form of the main-effect kernel from data, and constructed an companion variance component test. While interesting, the null distribution of CVEK is constructed using asymptotic approximation, and requires the interaction-kernel to be fixed a priori, therefore calling into question the validity of the test in limited sample, and prevents practitioners from deploying flexible methods to learn the interaction kernel from data. In this work, we seek to address these shortcomings by proposing a bootstrap test for CVEK. We conduct comprehensive simulation study to evaluate the validity (i.e. Type I error) and power of the proposed test using diverse choices of modeling strategy and under a wide range of data-generation mechanisms. Our simulation results revealed valuable insight on the impact of choice of estimation strategy (i.e. choices of tuning-parameter selection criteria and ensemble strategy) on the performance of the resulting hypothesis test.

2 Method

2.1 Cross-validated Ensemble of Kernerls

2.1.1 Overview

Cross-validated Ensemble of Kernerls (CVEK) [1] is a ensemble-based method for an unknown data-generating function $h : \mathbb{R}^p \to \mathbb{R}$. Traditional practices for estimating h, for example Gaussian Process (GP) regression using a single kernel function, tend to impose *a priori* assumption on the mathematical property of h by specifying the reproducing kernel function k for $h \in \mathcal{H}$, thereby risking inducing biased estimation and incorrect inference. To circumvent this issue, CVEK propose estimating h using the ensemble of GP predictions generated from a library of (fixed) base kernel functions $\{k_d\}_{d=1}^D$:

$$\hat{h}(\mathbf{x}) = \sum_{d=1}^{D} u_d \hat{h}_d(\mathbf{x}), \quad \mathbf{u} \in \Delta = \{\mathbf{u} \mid \mathbf{u} \geqslant 0, \| \mathbf{u} \|_1 = 1\} \tag{2.1.1}$$

where $\hat{h}_d$ is the kernel predictor generated by d^{th} base kernel k_d.
To be more specific, for each given basis kernel $\{k_d\}_{d=1}^D$, CVEK first estimate $\hat{\mathbf{h}}_d = \mathbf{K}_d(\mathbf{K}_d + \hat{\lambda}_d \mathbf{I})^{-1}\mathbf{y}$, the prediction based on d^{th} kernel, where the tuning parameter $\hat{\lambda}_d$ is selected by minimizing certain criteria (see section 1.2). After which, CVEK combines the the individual base kernel predictors $\{\hat{h}_d\}_{d=1}^D$ according to some ensemble strategy (see section 1.3). In addition to producing ensemble prediction $\hat{h}$, CVEK also produces an ensemble kernel matrix $\hat{\mathbf{K}}$ which describes the mathematical property of the ensemble RKHS. $\hat{\mathbf{K}}$ is estimated by solving:

$$\hat{\mathbf{K}}(\hat{\mathbf{K}} + \lambda \mathbf{I})^{-1} = \hat{\mathbf{A}}$$

In fact, $\hat{\mathbf{K}}$ can be computed in closed form. If we denote $\mathbf{U}_A$ and $\{\delta_{A,k}\}_{k=1}^n$ the eigenvector and eigenvalues of $\hat{\mathbf{A}}$, then:

$$\hat{\mathbf{K}} = \mathbf{U}_A \, diag\left(\frac{\delta_{A,k}}{1 - \delta_{A,k}}\right) \mathbf{U}_A^T$$

A complete summary of the proposed procedure (using LooCV for tuning-parameter selection and empirical risk minimization for ensemble strategy) is available in Algorithm 1.

2.1.2 Tuning Parameter Selection

Models may provide a good fit to the training data, but it will not fit sufficiently well to the test data. Tuning parameter could be chosen to address this problem. Here we define four objective functions in terms of tuning parameter $\lambda \in \Lambda$ to be minimized. Denote

$$\mathbf{A}_\lambda = \mathbf{K}(\mathbf{X}, \mathbf{X})[\mathbf{K}(\mathbf{X}, \mathbf{X}) + \lambda \mathbf{I}]^{-1} \tag{2.1.2}$$

In this way, $tr(\mathbf{A}_\lambda)$ is the effective number of model parameters, excluding μ and σ^2. It decreases monotonically with $\lambda > 0$. Additionally, we denote $\mathbf{y}^\star$ as the centered $\mathbf{y}$: $\mathbf{y}^\star = \mathbf{y} - \hat{\boldsymbol{\mu}}$, where $\hat{\mu} = \frac{1}{n}\sum_{i=1}^n y_i$.

K-Fold and leave-one-out Cross Validation

Cross validation is probably the simplest and most widely used method for estimating prediction error. Suppose we do a K-fold cross-validation, which partitions observations into K groups, $\kappa(1), ..., \kappa(K)$, and calculates $\mathbf{A}_\lambda$ K times, each time leaving out group $\kappa(i)$, to get $\mathbf{A}_\lambda^{-\kappa(1)}, \mathbf{A}_\lambda^{-\kappa(2)}$, etc. For $\mathbf{A}_\lambda^{-\kappa(i)}$, cross-validated residuals are calculated on the observations in $\kappa(i)$, which did not contribute to estimating $\mathbf{A}$. The objective function estimated prediction error and is the sum of the squared cross-validated residuals:

$$\lambda_{K-CV} = \underset{\lambda \in \Lambda}{\operatorname{argmin}} \left\{ \log \sum_{i=1}^{K} [\mathbf{y}^\star_{\kappa(i)} - \mathbf{A}_\lambda^{-\kappa(i)} \mathbf{y}^\star_{\kappa(i)}]^\top [\mathbf{y}^\star_{\kappa(i)} - \mathbf{A}_\lambda^{-\kappa(i)} \mathbf{y}^\star_{\kappa(i)}] \right\} \tag{2.1.3}$$

LooCV is the situation when $K = n$. In this case, we can write our objective function as [2]:

$$\lambda_{n-CV} = \underset{\lambda \in \Lambda}{\operatorname{argmin}} \left\{ \log \mathbf{y}^{\star\top} [\mathbf{I} - diag(\mathbf{A}_\lambda) - \frac{1}{n}\mathbf{I}]^{-1} (\mathbf{I} - \mathbf{A}_\lambda)^2 [\mathbf{I} - diag(\mathbf{A}_\lambda) - \frac{1}{n}\mathbf{I}]^{-1} \mathbf{y}^\star \right\} \tag{2.1.4}$$

The value K influences bias and variance of cross-validation. With $K = n$, the cross-validation estimator is approximately unbiased for the true (expected) prediction error because it almost use all the data in each training set. Therefore, it can have high variance because n training sets are so similar to one another. Additionally, the computational burden is also considerable, requiring n applications of the learning method. On the other hand, with larger K such as 5 [3] or 10, cross-validation will have lower variance, but making bias a problem.

Akaike Information Criteria

Based on the idea of "model fit + model complexity", Akaike's Information Criterion (AIC) [4] chooses λ by minimizing,

$$\begin{aligned} AIC =& 2(p+2) - 2\log(\hat{L}) \\ =& 2(p+2) - 2[-\frac{n}{2}\log(\hat{\sigma}^2) - \frac{n}{2}\log(2\pi) - \frac{1}{2\hat{\sigma}^2}\mathbf{y}^{\star\top}(\mathbf{I} - \mathbf{A}_\lambda)^2\mathbf{y}^\star] \\ =& 2(p+2) + n\log[\frac{1}{n}\mathbf{y}^{\star\top}(\mathbf{I} - \mathbf{A}_\lambda)^2\mathbf{y}^\star] + n + n\log(2\pi) \end{aligned}$$

Drop the constant n and divide it by n, we obtain our objective function:

$$\lambda_{AIC} = \underset{\lambda \in \Lambda}{\operatorname{argmin}} \left\{ \log \mathbf{y}^{\star\top}(\mathbf{I} - \mathbf{A}_\lambda)^2\mathbf{y}^\star + \frac{2[tr(\mathbf{A}_\lambda) + 2]}{n} \right\} \tag{2.1.5}$$

When n is small, extreme overfitting is possible, giving small bias/ large variance estimates. The small-sample correction of AIC [5, 6] is derived by minimizing minus 2 times expected log likelihood, where we plug in $\mathbf{A}_\lambda$ and $\hat{\sigma}^2$. In this case, we obtain our small-sample size objective function AICc:

$$\lambda_{AICc} = \underset{\lambda \in \Lambda}{\operatorname{argmin}} \left\{ \log \mathbf{y}^{\star\top}(\mathbf{I} - \mathbf{A}_\lambda)^2\mathbf{y}^\star + \frac{2[tr(\mathbf{A}_\lambda) + 2]}{n - tr(\mathbf{A}_\lambda) - 3} \right\} \tag{2.1.6}$$

Compare (2.1.5) and (2.1.6) , it is easy to tell that AICc considers more of the model complexity since $\frac{n}{n - tr(\mathbf{A}_\lambda) - 3} > 1$. It makes sense intuitively because it need to shrink more to prevent small bias/ large variance estimates.

Generalized Cross Validation
In (2.1.4), if we approximate each $A_{\lambda[ii]}$ with their mean $\frac{tr(\mathbf{A}_\lambda)}{n}$, in a sense that we give equal weight to all observations. We get the Generalized Cross Validation (GCV) objective function:

$$\lambda_{GCV} = \underset{\lambda\in\Lambda}{\operatorname{argmin}}\Big\{\log \mathbf{y}^{\star\top}(\mathbf{I}-\mathbf{A}_\lambda)^2\mathbf{y}^\star - 2\log[1-\frac{tr(\mathbf{A}_\lambda)}{n}-\frac{1}{n}]\Big\} \tag{2.1.7}$$

The "$-\frac{1}{n}$" terms in (2.1.7) is because GCV counts μ as part of model complexity, but not σ^2. This motivates the proposed small-sample correction to GCV [7], which does count σ^2 as a parameter:

$$\lambda_{GCVc} = \underset{\lambda\in\Lambda}{\operatorname{argmin}}\Big\{\log \mathbf{y}^{\star\top}(\mathbf{I}-\mathbf{A}_\lambda)^2\mathbf{y}^\star - 2\log[1-\frac{tr(\mathbf{A}_\lambda)}{n}-\frac{2}{n}]_+\Big\} \tag{2.1.8}$$

Under this situation, perfect fit of the observations to the predictions, given by $\lambda = 0$, cannot occur.

Generalized Maximum Profile Marginal Likelihood
Suppose we relate Y and $\mathbf{X}$ by a linear model, $Y = \mu + \phi(\mathbf{X})^\top\boldsymbol{\beta} + \epsilon$, with $\epsilon \sim N(0,\sigma^2)$, where $\phi(\mathbf{x})$ is a function mapping a D-dimensional input vector $\mathbf{x}$ into an p-dimensional feature space. If we assume $\boldsymbol{\beta}$ are jointly and independently normal with mean zero and variance σ^2/λ, the penalty term matches the negative normal log-density, up to a normalizing constant not depending on $\boldsymbol{\beta}$:

$$p_\lambda(\boldsymbol{\beta},\sigma^2) = \frac{\lambda}{2\sigma^2}\boldsymbol{\beta}^\top\boldsymbol{\beta} - \frac{p}{2}\log(\lambda) + \frac{p}{2}\log(\sigma^2)$$

One can consider a marginal likelihood, where λ is interpreted as the variance component of a mixed-effects model:

$$\begin{aligned} m(\lambda,\sigma^2) =& \log\int_{\boldsymbol{\beta}} exp\{l(\boldsymbol{\beta},\sigma^2) - p_\lambda(\boldsymbol{\beta},\sigma^2)\}d\boldsymbol{\beta} \\ =& -\frac{1}{2\sigma^2}\mathbf{y}^{\star\top}(\mathbf{I}-\mathbf{A}_\lambda)\mathbf{y}^\star - \frac{n}{2}\log(\sigma^2) + \frac{1}{2}\log|\,\mathbf{I}-\mathbf{A}_\lambda\,| \end{aligned}$$

From this, $\mathbf{y}^\star \mid \lambda,\sigma^2$ is multivariate normal with mean $\mathbf{0}_n$ and covariance $\sigma^2(\mathbf{I}-\mathbf{A}_\lambda)^{-1}$. The maximum profile marginal likelihood (MPML) estimate, originally proposed for smoothing spline [8], profiles $m(\lambda,\sigma^2)$ over σ^2, replacing each instance with $\hat{\sigma}^2_\lambda = \mathbf{y}^{\star\top}(\mathbf{I}-\mathbf{A}_\lambda)\mathbf{y}^\star/n$, and maximized the "concentrated" log-likelihood, $m(\lambda,\hat{\sigma}^2_\lambda)$:

$$\lambda_{MPML} = \underset{\lambda\in\Lambda}{\operatorname{argmin}}\Big\{\log \mathbf{y}^{\star\top}(\mathbf{I}-\mathbf{A}_\lambda)\mathbf{y}^\star - \frac{1}{n}\log|\,\mathbf{I}-\mathbf{A}_\lambda\,|\Big\}$$

Closely related is the generalized/ restricted MPML [9, 10], which adjusts the penalty to account for estimation of regression parameter β_0 that is not marginalized, resulting in one degree of freedom:

$$\lambda_{GMPML} = \underset{\lambda\in\Lambda}{\operatorname{argmin}}\Big\{\log \mathbf{y}^{\star\top}(\mathbf{I}-\mathbf{A}_\lambda)\mathbf{y}^\star - \frac{1}{n-1}\log|\,\mathbf{I}-\mathbf{A}_\lambda\,|\Big\} \tag{2.1.9}$$

Relation between AIC, GCV and GMPML
It's interesting to compare the λ's selected from AIC, GCV, or GMPML, since AIC comes from Kullback-Leibler divergence, GCV from cross-validation and GMPML from likelihood. To do

this, first we need to introduce some notations.
If we denote $\mathbf{U}_K$ and $\{\eta_{K,j}\}_{j=1}^n$ the eigenvector and eigenvalues of $\mathbf{K}$, then $\mathbf{A}_\lambda$ adopts the form:

$$\mathbf{A}_\lambda = \mathbf{U}_K diag(\frac{\eta_{K,j}}{\eta_{K,j}+\lambda})\mathbf{U}_K^T = \mathbf{U}_K \mathbf{D}_{K,\lambda}\mathbf{U}_K^T$$

Denote the objective functions in (2.1.5), (2.1.7) and (2.1.9) as f_{AIC}, f_{GCV} and f_{GMPML}, and calculate the derivatives of them with respect to λ respectively,

$$\frac{\partial f_{AIC}}{\partial \lambda} = \frac{2tr\Big[\mathbf{U}_K^T\mathbf{y}^\star\mathbf{y}^{\star T}\mathbf{U}_K(\mathbf{D}_{K,\lambda}-1)\frac{\partial \mathbf{D}_{K,\lambda}}{\partial\lambda}\Big]}{tr[\mathbf{U}_K^T\mathbf{y}^\star\mathbf{y}^{\star T}\mathbf{U}_K(\mathbf{I}-\mathbf{D}_{K,\lambda})^2]} + \frac{2tr\Big(\frac{\partial \mathbf{D}_{K,\lambda}}{\partial\lambda}\Big)}{n} \tag{2.1.10}$$

$$\frac{\partial f_{GCV}}{\partial \lambda} = \frac{2tr\Big[\mathbf{U}_K^T\mathbf{y}^\star\mathbf{y}^{\star T}\mathbf{U}_K(\mathbf{D}_{K,\lambda}-1)\frac{\partial \mathbf{D}_{K,\lambda}}{\partial\lambda}\Big]}{tr[\mathbf{U}_K^T\mathbf{y}^\star\mathbf{y}^{\star T}\mathbf{U}_K(\mathbf{I}-\mathbf{D}_{K,\lambda})^2]} + \frac{2tr\Big(\frac{\partial \mathbf{D}_{K,\lambda}}{\partial\lambda}\Big)}{n-tr(\mathbf{A}_\lambda)-1} \tag{2.1.11}$$

$$\frac{\partial f_{GMPML}}{\partial \lambda} = \frac{-tr\Big[\mathbf{U}_K^T\mathbf{y}^\star\mathbf{y}^{\star T}\mathbf{U}_K\frac{\partial \mathbf{D}_{K,\lambda}}{\partial\lambda}\Big]}{tr[\mathbf{U}_K^T\mathbf{y}^\star\mathbf{y}^{\star T}\mathbf{U}_K(\mathbf{I}-\mathbf{D}_{K,\lambda})]} + \frac{tr\Big[(\mathbf{I}-\mathbf{D}_{K,\lambda})^{-1}\frac{\partial \mathbf{D}_{K,\lambda}}{\partial\lambda}\Big]}{n-1} \tag{2.1.12}$$

Notice for j^{th} element of diagonal vector of $\mathbf{D}_{K,\lambda}$, its derivative with respect to λ is negative,

$$\frac{\partial}{\partial\lambda}\Big[\frac{\eta_{K,j}}{\eta_{K,j}+\lambda}\Big] = -\frac{\eta_{K,j}}{(\eta_{K,j}+\lambda)^2} < 0, \quad \text{for } j = 1,\ 2, ...,\ n$$

Further notice that the difference between (2.1.10) and (2.1.11) focuses on the second terms, both of which are increasing function of λ.

$$\frac{2tr\Big(\frac{\partial \mathbf{D}_{K,\lambda}}{\partial\lambda}\Big)}{n-tr(\mathbf{A}_\lambda)-1} < \frac{2tr\Big(\frac{\partial \mathbf{D}_{K,\lambda}}{\partial\lambda}\Big)}{n}$$

Therefore, when $\frac{\partial f_{AIC}}{\partial\lambda} = 0$, $\frac{\partial f_{GCV}}{\partial\lambda} < 0$, which means $\lambda_{AIC} < \lambda_{GCV}$.

The relationship between GMPML and GCV, however, are more complicated, as their relative magnitude depends on the relation between model and data [11]. See Appendix for more detailed discussion.

2.1.3 Ensemble Strategy

Empirical Risk Minimization
After obtaining the estimated errors $\{\hat{\mathbf{e}}_d\}_{d=1}^D$, we estimate the ensemble weights $\mathbf{u} = \{u_d\}_{d=1}^D$ such that it minimizes the overall error [1]:

$$\hat{\mathbf{u}} = \underset{\mathbf{u}\in\Delta}{argmin} \parallel \sum_{d=1}^{D} u_d\hat{\mathbf{e}}_d \parallel^2 \quad where\ \Delta = \{\mathbf{u} \mid \mathbf{u} \geqslant 0, \parallel \mathbf{u} \parallel_1 = 1\}$$

Then produce the final ensemble prediction:

$$\hat{\mathbf{h}} = \sum_{d=1}^{D} \hat{u}_d\mathbf{h}_d = \sum_{d=1}^{D} \hat{u}_d\mathbf{A}_{d,\hat{\lambda}_d}\mathbf{y} = \hat{\mathbf{A}}\mathbf{y}$$

where $\hat{\mathbf{A}} = \sum_{d=1}^{D} \hat{u}_d\mathbf{A}_{d,\hat{\lambda}_d}$ is the ensemble matrix.

Simple Averaging
Motivated by existing literature in omnibus kernel [12], we propose another way to obtain the ensemble matrix by simply choosing unsupervised weights $u_d = 1/D$ for $d = 1, 2, ...D$.

Exponential Weighting
Additionally, [13] gives a new strategy to calculate weights based on the estimated errors $\{\hat{\mathbf{e}}_d\}_{d=1}^{D}$.

$$u_d(\beta) = \frac{exp(- \parallel \hat{\mathbf{e}}_d \parallel_2^2 / \beta)}{\sum_{d=1}^{D} exp(- \parallel \hat{\mathbf{e}}_d \parallel_2^2 / \beta)}$$

2.1.4 Kernel Choice

In this section [14] we give introductions to some commonly-used kernel functions, including two stationary covariance functions (gaussian rbf, matérn and rational quadratic), as well as non-stationary covariance functions (polynomial and neural network). For convenience, we define $r = \mid \mathbf{x} - \mathbf{x}' \mid$.

Gaussian RBF Kernels
The Gaussian radial basis function (RBF), also known as *squared exponential* (SE) kernel function has the form

$$k_{SE}(r) = exp\Big(-\frac{r^2}{2l^2}\Big)$$

with parameter l defining the *characteristic length-scale*. This covariance function is infinitely differentiable, which means that the GP with this covariance function has mean square derivatives of all orders, and is thus very smooth. The spectral density of SE covariance function is $S(s) = (2\pi l^2)^{D/2} exp(-2\pi^2 l^2 s^2)$. [15] argues that such strong smoothness assumptions are unrealistic for modeling many physical processes, and recommends the Matérn class. However, the squared exponential is probably the most widely-used kernel within the kernel machines field. The SE kernel is *infinitely divisible* in that $(k(r))^t$ is a valid kernel for all $t > 0$; the effect of raising k to the power of t is simply to rescale l. The eigenvalues are subject to exponential decay: $\lambda_i = O(e^{-\beta i})$ for $\beta > 0$.

Matérn Kernels
The *Matérn class* of covariance functions is given by

$$k_{Matern}(r) = \frac{2^{1-\nu}}{\Gamma(\nu)}\Big(\frac{\sqrt{2\nu r}}{l}\Big)^{\nu} K_{\nu}\Big(\frac{\sqrt{2\nu r}}{l}\Big)$$

with positive parameters ν and l, where K_ν is a modified Bessel function [16]. This covariance function has a spectral density

$$S(s) = \frac{2^D \pi^{D/2} \Gamma(\nu + D/2)(2\nu)^{\nu}}{\Gamma(\nu) l^{2\nu}}\Big(\frac{2\nu}{l^2} + 4\pi^2 s^2\Big)^{-(\nu + D/2)}$$

in D dimensions. For the Matérn class the process $f(\mathbf{x})$ is k-times MS differentiable is and only if $\nu > k$. The Matérn covariance functions become especially simple when ν is half-integer: $\nu =$

$p + 1/2$, where p is a non-negative integer. In this case the covariance function is a product of an exponential and a polynomial of order p, the general expression can be derived from [16], giving

$$k_{\nu=p+1/2}(r) = exp\Big(-\frac{\sqrt{2\nu r}}{l}\Big)\frac{\Gamma(p+1)}{\Gamma(2p+1)}\sum_{i=0}^{p}\frac{(p+i)!}{i!(p-i)!}\Big(\frac{\sqrt{8\nu r}}{l}\Big)^{p-i}$$

It is possible that the most interesting cases for machine learning are $\nu = 3/2$ and $\nu = 5/2$, since for $\nu = 1/2$ the process becomes very rough and for $\nu \geqslant 7/2$, in the absence of explicit prior knowledge about the existence of higher order derivatives, it is probably very hard from finite noisy training examples to distinguish between $\nu \geqslant 7/2$. The eigenvalues are subject to polynomial decay: $\lambda_i = O(i^{-\alpha})$ for $\alpha > 1$.

Rational Quadratic Kernels
The *rational quadratic* (RQ) covariance function

$$k_{RQ}(r) = \Big(1 + \frac{r^2}{2\alpha l^2}\Big)^{-\alpha}$$

with $\alpha, l > 0$ can be seen as a *scale mixture* (an infinite sum) of squared exponential (SE) covariance functions with different characteristic length-scales (sum of covariance functions are also a valid covariance). Parameterizing now in terms of inverse squared length scales, $\tau = l^{-2}$, and putting a gamma distribution on $p(\tau \mid \alpha, \beta) \propto \tau^{\alpha-1}exp(-\alpha\tau/\beta)$, we can add up the contributions through the following integral

$$k_{RQ}(r) = \int p(\tau \mid \alpha, \beta)k_{SE}(r \mid \tau)d\tau \propto \int \tau^{\alpha-1}exp\Big(-\frac{\alpha\tau}{\beta}\Big)exp\Big(-\frac{\tau r^2}{2}\Big)d\tau \propto \Big(1 + \frac{r^2}{2\alpha l^2}\Big)^{-\alpha}$$

where we have set $\beta^{-1} = l^2$. The limit of the RQ covariance for $\alpha \to \infty$ is the SE covariance function with characteristic length-scale l.

Polynomial Kernels
It is also interesting to show an explicit feature space construction for the *polynomial covariance function*. We consider the homogeneous polynomial case as the inhomogeneous case can simply be obtained by considering $\mathbf{x}$ to be extended by concatenating a constant. We write

$$\begin{aligned} k(\mathbf{x}, \mathbf{x}') =& (\mathbf{x} \cdot \mathbf{x}')^p = (\sum_{d=1}^{D} x_d x'_d)^p = (\sum_{d_1=1}^{D} x_{d_1} x'_{d_1})...(\sum_{d_p=1}^{D} x_{d_p} x'_{d_p}) \\ &= \sum_{d_1=1}^{D} ... \sum_{d_p=1}^{D} (x_{d_1}...x_{d_p})(x'_{d_1}...x'_{d_p}) \stackrel{\Delta}{=} \phi(\mathbf{x}) \cdot \phi(\mathbf{x}') \end{aligned}$$

Notice that this sum apparently contains D^p terms but in fact it is less than this as the order of the indices in the monomial $x_{d_1}...x_{d_p}$ is unimportant, e.g. for $p = 2$, x_1x_2 and x_2x_1 are the same monomial. We can remove the redundancy by defining a vector $\mathbf{m}$ whose entry m_d specifies the number of times index d appears in the monomial, under the constraint that $\sum_{i=1}^{D} m_i = p$. Thus $\phi_{\mathbf{m}}(\mathbf{x})$, the feature corresponding to vector $\mathbf{m}$ is proportional to the monomial $x_1^{m_1}...x_D^{m_D}$. The degeneracy of $\phi_{\mathbf{m}}(\mathbf{x})$ is $\frac{p!}{m_1!...m_D!}$, giving the feature map

$$\phi_{\mathbf{m}}(\mathbf{x}) = \sqrt{\frac{p!}{m_1!...m_D!}}x_1^{m_1}...x_D^{m_D}$$

For example, for $p = 2$ in $D = 2$, we have $\phi(\mathbf{x}) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)^T$.
Additionally, we have intercept kernel when $p = 0$, and linear kernel when $p = 1$.

Neural Network Kernels
Consider a network which takes an input $\mathbf{x}$, has one hidden layer with N_H units and then linearly combines the outputs of the hidden units with a bias b to obtain $f(\mathbf{x})$. The mapping can be written

$$f(\mathbf{x}) = b + \sum_{j=1}^{N_H} v_j h(\mathbf{x}; \mathbf{u}_j)$$

where the v_j's are the hidden-to-output weights and $h(\mathbf{x}; \mathbf{u})$ is the hidden unit transfer function (which we shall assume is bounded) which depends on the input-to-hidden weights $\mathbf{u}$. Let b and the v's have independent zero-mean distributions of variance σ_b^2 and σ_v^2, respectively, and let the weights $\mathbf{u}_j$ for each hidden unit be independently and identically distributed. Denoting all weights by $\mathbf{w}$, we obtain

$$\begin{aligned} \mathbb{E}_{\mathbf{w}}[f(\mathbf{x})] =& 0 \\ \mathbb{E}_{\mathbf{w}}[f(\mathbf{x})f(\mathbf{x}')] =& \sigma_b^2 + \sum_j \sigma_v^2 \mathbb{E}_{\mathbf{u}}[h(\mathbf{x}; \mathbf{u}_j)h(\mathbf{x}'; \mathbf{u}_j)] \qquad (2.1.13) \\ =& \sigma_b^2 + N_H \sigma_v^2 \mathbb{E}_{\mathbf{u}}[h(\mathbf{x}; \mathbf{u})h(\mathbf{x}'; \mathbf{u})] \qquad (2.1.14) \end{aligned}$$

where (2.1.14) follows because all of the hidden units are identically distributed. The sum in (2.1.13) is over N_H identically and independently distributed random variables. As the transfer function is bounded, all moments of the distribution will be bounded and hence the central limit theorem can be applied, showing that the stochastic process will converge to a Gaussian process in the limit as $N_H \to \infty$.
By evaluating $\mathbb{E}_{\mathbf{u}}[h(\mathbf{x}; \mathbf{u})h(\mathbf{x}'; \mathbf{u})]$ we can obtain the covariance function of the neutral network. For example if we choose the error function $h(z) = erf(z) = 2/\sqrt{\pi}\int_0^z e^{-t^2}dt$ as the transfer function, let $h(\mathbf{x}; \mathbf{u}) = erf\left(u_0 + \sum_{j=1}^D u_j x_j\right)$ and choose $\mathbf{u} \sim N(0, \Sigma)$ then we obtain

$$k_{NN}(\mathbf{x}, \mathbf{x}') = \frac{2}{\pi} sin^{-1}\left(\frac{2\tilde{\mathbf{x}}^T \Sigma \tilde{\mathbf{x}}'}{\sqrt{(1 + 2\tilde{\mathbf{x}}^T \Sigma \tilde{\mathbf{x}})(1 + 2\tilde{\mathbf{x}}'^T \Sigma \tilde{\mathbf{x}}')}}\right)$$

where $\tilde{\mathbf{x}} = (1, x_1, ..., x_d)^T$ is an augmented input vector.

2.2 Hypothesis Test

2.2.1 Asymptotic Test

We use the classical variance component test [17] to construct a testing procedure for the hypothesis about Gaussian process function:

$$H_0 : h \in \mathcal{H}_0. \qquad (2.2.1)$$

We first translate above hypothesis into a hypothesis in terms of model parameters. The key of our approach is to assume that h lies in a RKHS generated by a *garrote kernel function* $k_\delta(\mathbf{z}, \mathbf{z}')$ [18], which is constructed by including an extra *garrote parameter* δ to a given kernel function. When $\delta = 0$, the garrote kernel function $k_0(\mathbf{x}, \mathbf{x}') = k_\delta(\mathbf{x}, \mathbf{x}') \mid_{\delta=0}$ generates exactly $\mathcal{H}_0$, the space of

functions under the null hypothesis. In order to adapt this general hypothesis to their hypothesis of interest, practitioners need only to specify the form of the garrote kernel so that $\mathcal{H}_0$ corresponds to the null hypothesis. For example, if $k_\delta(\mathbf{x}) = k(\delta * x_1, x_2, ..., x_p), \delta = 0$ corresponds to the null hypothesis $H_0 : h(\mathbf{x}) = h(x_2, ..., x_p)$, i.e. the function $h(\mathbf{x})$ does not depend on x_1. As a result, the general hypothesis is equivalent to:

$$H_0 : \delta = 0. \tag{2.2.2}$$

We now construct a test statistic $\hat{T}_0$ for (2.2.2) by noticing that the garrote parameter δ can be treated as a variance component parameter in the linear mixed model. This is because the Gaussian process under garrote kernel can be formulated into below LMM:

$$\mathbf{y} = \boldsymbol{\mu} + \mathbf{h} + \boldsymbol{\epsilon} \quad where \quad \mathbf{h} \sim N(\mathbf{0}, \tau \mathbf{K}_\delta) \quad \boldsymbol{\epsilon} \sim N(\mathbf{0}, \sigma^2 \mathbf{I}) \tag{2.2.3}$$

where $\mathbf{K}_\delta$ is the kernel matrix generated by $k_\delta(\mathbf{z}, \mathbf{z}')$. Consequently, we can derive a variance component test for H_0 by calculating the square derivative of L_{REML} with respect to δ under H_0 [17]:

$$\hat{T}_0 = \hat{\tau} * (\mathbf{y} - \hat{\boldsymbol{\mu}})^T \mathbf{V}_0^{-1} [\partial \mathbf{K}_0] \mathbf{V}_0^{-1} (\mathbf{y} - \hat{\boldsymbol{\mu}}) \tag{2.2.4}$$

where $\mathbf{V}_0 = \hat{\sigma}^2 \mathbf{I} + \hat{\tau} \mathbf{K}_0$. In this expression, $\mathbf{K}_0 = \mathbf{K}_\delta \mid_{\delta=0}$, and $\partial \mathbf{K}_0$ is the null derivative kernel matrix whose $(i, j)^{th}$ entry is $\frac{\partial}{\partial \delta} k_\delta(\mathbf{x}, \mathbf{x}') \mid_{\delta=0}$.
The null distribution of $\hat{T}$ can be approximated using a scaled chi-square distribution $\kappa \chi^2_\nu$ using Satterthwaite method by matching the first two moments of T:

$$\kappa * \nu = E(T) = \hat{\tau} * tr(\mathbf{V}_0 \partial \mathbf{K}_0) \quad 2 * \kappa^2 * \nu = Var(T) = \hat{\mathbf{I}}_{\delta\delta}$$

with solution:

$$\hat{\kappa} = \hat{\mathbf{I}}_{\delta\delta} / [\hat{\tau} * tr(\mathbf{V}_0^{-1} \partial \mathbf{K}_0)] \quad \hat{\nu} = [\hat{\tau} * tr(\mathbf{V}_0^{-1} \partial \mathbf{K}_0)]^2 / (2 * \hat{\mathbf{I}}_{\delta\theta})$$

where $\hat{\mathbf{I}}_{\delta\delta} = \mathbf{I}_{n,\delta\delta} - \mathbf{I}_{\delta\theta}^T \mathbf{I}_{\theta\theta}^{-1} \mathbf{I}_{\delta\theta}$ is the efficient information of δ under REML. $\mathbf{I}_{\delta\delta}$, $\mathbf{I}_{\theta\theta}$ and $\mathbf{I}_{\delta\theta}$ are submatrices of the REML information matrix. Numerically more accurate, but computationally less efficient approximation methods are also available.
Finally, the p-value of this test is calculated by examining the tail probability of $\hat{\kappa}\chi^2_{\hat{\nu}}$:

$$p = P(\hat{\kappa}\chi^2_{\hat{\nu}} > \hat{T}) = P(\chi^2_{\hat{\nu}} > \hat{T}/\hat{\kappa})$$

A complete summary of the proposed testing procedure is available in Algorithm 2.

2.2.2 Bootstrap Test

In practice, the sample size of the collected data is always small. To make valid inferences about a population from the sample, we need to perform resampling. Commonly used methods in small sample size are permutation tests and bootstrap.

Permutation Tests
Permutation tests are very popular in genomic research. They are simple to compute where analytic approaches may be intractable, and can be exact where analytic results may be only approximate. However, the main limitation of permutation tests is that they are only applicable when the null hypothesis being tested specifies a suitable group of permutations under which the distribution of the data would be unaffected. To illustrate this idea, consider a test for interaction between the effects of the genetic polymorphism G and a environmental exposure E on an outcome Y. The null hypothesis is that the interaction term is zero [19].

$$E[Y] = \beta_0 + \beta_G G + \beta_E E \tag{2.2.5}$$

An alternative hypothesis of interest may be that,

$$E[Y] = \beta_0 + \beta_G G + \beta_E E + \delta G * E$$

For a valid permutation test of the hypothesis of no interaction, we would require a group of permutations that exactly preserves both β_G and β_E in (2.2.5), but also ensures $\delta = 0$. In general it is impossible to construct such a group of permutations, as demonstrated by Edgington (1987, Chap. 6). If the permutations fix G and E they will not give $\delta = 0$, and if they do not fix G and E they will not preserve the relationship between G and E and so will not preserve β_G and β_E.

Bootstrap Test
Instead we can use parametric bootstrap, which can give valid tests with moderate sample sizes and requires similar computational effort to a permutation test.
Testing in a regression model framework requires computing the distribution of the test statistic under sampling from the null-hypothesis model. A good approximation to the distribution of the test statistic under sampling from the true-hypothesis model is the distribution of the test statistic under sampling from the fitted null-hypothesis model. For instance, when testing (2.2.2), we first fit the model under the null,

$$E(\mathbf{y}^\star) = \mathbf{K}_0(\mathbf{K}_0 + \lambda \mathbf{I})^{-1}\mathbf{y} = \mathbf{A}_0\mathbf{y} \tag{2.2.6}$$

and generate $\mathbf{Y}^\star$ for each individuals with a random noise, whose variance is also estimated. We then compute the test statistic for this simulated sample, and repeat this process B times. The empirical distribution these provide is an estimate of the test statistic's distribution under the null. Correspondingly, p-values are calculated as the proportion of simulated test statistics that are most extreme than the observed value.
If the distribution of the test statistic depends smoothly on the regression parameter values, which is true in all standard examples, this 'parametric bootstrap' approach gives an asymptotically valid test(Davison & Hinkley 1997, 4.2.3). Like the classical bootstrap, it samples from a distribution based on the observed data, but the simulations are from a fitted parametric model rather than the empirical distribution. To obtain a valid test, the fitted parametric model is chosen so that the null hypothesis is satisfied. A complete summary of the proposed testing procedure is available in Algorithm 3.

3 Simulation

We evaluated the finite-sample performance of the proposed interaction test in a simulation study that is analogous to a real nutrition-environment interaction study. We generate two groups of

input features $(\mathbf{x}_{i,1}, \mathbf{x}_{i,2}) \in \mathbb{R}^{p_1} \times \mathbb{R}^{p_2}$ independently from standard Gaussian distribution, representing normalized data representing subject's level of exposure to p_1 environmental pollutants and the levels of a subject's intake of p_2 nutrients during the study. Throughout the simulation scenarios, we keep $n = 100$, and $p_1 = p_2 = 2$. We generate the outcome y_i as:

$$y_i = h_1(\mathbf{x}_{i,1}) + h_2(\mathbf{x}_{i,2}) + \delta * h_1(\mathbf{x}_{i,1}) * h_2(\mathbf{x}_{i,2}) + \epsilon_i$$

where h_1, h_2 are sampled from RKHSs $\mathcal{H}_1$, $\mathcal{H}_2$, generated using a ground-truth kernel k_{true}. We standardize all sampled functions to have unit form, so that δ represents the strength of interaction relative to the main effect.
For each simulation scenario, we first generated data using δ and k_{true} as above, then selected a k_{model} to estimate the null model and obtain p-value using Algorithm 2 and 3 repsectively. We repeated each scenario 200 times, and evaluate the test performance using the empirical probability $\hat{P}(p \leqslant 0.05)$ estimates the test's Type I error, and should be smaller or equal to the significance level 0.05. Under alternative hypothesis $H_a : \delta > 0$, $\hat{P}(p \leqslant 0.05)$ estimates the test's power, and should ideally approach 1 quickly as the strength of interaction δ increases.
In this study, we varied k_{true} to produce data-generating functions $h_\delta(\mathbf{x}_{i,1}, \mathbf{x}_{i,2})$ with different smoothness and complexity properties, and varied k_{model} to reflect different common modeling strategies for the null model in addition to using CVEK. We then evaluated how these two aspects impact the hypothesis test's Type I error and power.

In terms of data-generating mechanism, we consider 12 combinations: 3 polynomial kernels ($p = 1, 2, 3$) representing finite-dimensional, parametric functions of different degree of nonlinearity. 3 Gaussian RBF kernels ($l = 0.5, 1, 1.5$) representing smooth functions of different frequency, and also 6 Matern kernels, with $l \in \{0.5, 1, 1.5\}$ and $\nu \in \{\frac{3}{2}, \frac{5}{2}\}$, representing functions with different frequency and differentiability. For modeling strategies, we consider 4 types of kernel libraries: (1) 3 polynomial kernels with degree ($p = 1, 2, 3$, Figure 2 & 7); (2) 3 RBF kernels with wavelength ($l = 0.6, 1, 2$, Figure 3 & 8); (3) 3 polynomial kernels ($p = 1, 2, 3$) and 3 RBF kernels ($l = 0.6, 1, 2$, Figure 4 & 9), and (4) 3 $l = 1$ Matern kernels ($\nu = 1/2, 3/2, 5/2$) and 3 RBF kernels ($l = 0.6, 1, 2$, Figure 5 & 10). For each combination of kernel library and data-generating mechanism, we estimate the null model using four methods for tuning-parameter selection (LooCV, AICc, GCVc, GMPML) and the three methods for ensemble strategy (ERM, simple averaging, exponential weighting), resulting in 12 null model estimates for each of the $12 \times 4 \times 2 = 96$ combinations.

4 Conclusion

The simulation results are presented graphically from Figure 1 to 10 (the first five are asymptotic tests and the last five bootstrap tests). They show the estimated $\hat{P}(p < 0.05)$ (y-axis) as a function of Interaction Strength $\delta \in \{0, 0.1, 0.2, 0.3\}$ (x-axis). For each figure, the combination of top margin and right margin indicate the data-generating mechanism: Polynomial, RBF, Matern $\nu = 3/2$, and Matern $\nu = 5/2$; $l = 0.5, 1, 1.5$ (represent $p = 1, 2, 3$ for Polynomial). And the lines refer to the different combination of tuning parameter selection (colors) and ensemble strategy (line types).

Generally, the power of asymptotic test is greater than our bootstrap test, but the bootstrap is capable of achieving satisfying performance (i.e. correct Type I error and decent power compared to the true kernel), when tuning parameter and ensemble method are chosen carefully. However, the exact performance of the test depends on the choice of tuning parameter selection, ensemble

strategy, and the relation between model and data.

For asymptotic test, we observed clear differences in test performance between different tuning-parameter criteria and different ensemble strategies. For tuning-parameter selection, LooCV and GMPML generally guarantee correct Type I error except when the base kernels are strictly simpler than the truths (Figure 2), but GMPML potentially leads to low power under the alternative(Figure 3/ 4). Moreover, AICc can guarantee correct Type I error only when the kernels in the library are smoother than the data-generating kernel (Figure 3, column 2-4). For ensemble strategies, ERM leads to better power (when Type I error is guaranteed) if base kernels are flexible and infinite-dimensional (Figure 3/ 5). In the more complex scenario when the library consists of a mixture of finite- and infinite-dimensional kernels (Figure 4), simple averaging performs better if data-generating kernels are of low frequency (row 3), otherwise ERM performs better (row 2).

In terms of bootstrap test, we also observed differences based on different tuning-parameter criteria and different ensemble strategies. For tuning-parameter selection, LooCV generally guarantees correct Type I error, but potentially leads to low power under the alternative (Figure 10). On the other hand, AICc is more powerful if kernels in the library are as or more complex (in terms of smoothness) than the true kernel (Figure 8, column 2; Figure 10, column 2, row 3 & column 3-4). Moreover, GMPML is more powerful if kernels in the library are smoother than the data-generating kernel (Figure 7; Figure 8/ 9, column 3-4). For ensemble strategies, simple averaging provides better Type I error and power if base kernels are simple and finite-dimensional (Figure 7). However, ERM leads to better power (when Type I error is guaranteed) if base kernels are flexible and infinite-dimensional (Figure 8/ 10). In the more complex scenario when the library consists of a mixture of finite- and infinite-dimensional kernels (Figure 9), simple averaging performs better if data-generating kernels are of low frequency (row 3), otherwise ERM performs better (row 2). However, the case is different for GMPML as GMPML-ERM strategy is always (and sometimes substantially) more powerful than GMPML-AVG.

Additionally, for our bootstrap test, there are two interesting observations that we'd like to point out: if the kernel library contains only very flexible kernels, model estimates will overfit the data and produce test with very low power, even if the kernels in the library are exactly the data-generation kernels (Figure 6, row 1 & column 2-4). On the other hand, if the kernel library contains kernels that are strictly simpler than the truths, even the proposed LooCV-ERM will inevitably produce slight-to-moderate level of Type I error inflation (e.g. using polynomial kernels to fit RBF data (Figure 7, column 2), or using too few RBF kernels to fit cubic polynomial (Figure 8/ 10, row 3 & column 1). In the latter case, the range of RBF length-scale parameters is not sufficient to cover the spectral density of the cubic function). Finally, in some cases, we observed powers decrease (LooCV in Figure 6/ 7, row 3 & column 1; LooCV with ERM in Figure 8/ 9, row 3 & column 1). We will investigate the mechanism behind such phenomenon in future work.

As for exponential weighting in our case, it shares similar power with simple averaging in both asymptotic and bootstrap tests. Moreover, Figure 2 & 7 (column 2-4) see the apparent advantages of our proposed bootstrap-LooCV: when the base kernels are strictly simpler than the truths, only bootstrap-LooCV can guarantee correct Type I error. Also, under the circumstance that base kernels are strictly more flexible than the truths (Figure 5/ 10, column 1), ERM performs better in asymptotic test while simple averaging better in bootstrap test.

Algorithm 1 Cross Validated Ensemble of Kernels (CVEK)

procedure CVEK
Input: A library of kernels $\{k_d\}_{d=1}^{D}$, Data $(\mathbf{y}, \mathbf{x})$
Output: Ensemble Kernel Matrix $\widehat{\mathbf{K}}$
# Stage 1: Estimate λ and CV error for each kernel
  for $d = 1$ to D **do**
    $\mathbf{K}_d = \mathbf{K}_d / tr(\mathbf{K}_d)$
    $\widehat{\lambda}_d = \textit{argmin}\ \texttt{LOOCV}\Big(\lambda \mid \mathbf{K}_d\Big)$
    $\widehat{\epsilon}_d = \texttt{CV}\Big(\widehat{\lambda}_d \mid \mathbf{K}_d\Big)$
  end for
# Stage 2: Estimate ensemble weights $\mathbf{u}_{D\times 1} = \{u_1, \ldots, u_D\}$
  $\widehat{\mathbf{u}} = \underset{\mathbf{u}\in\Delta}{\operatorname{argmin}} \ \| \sum_{d=1}^{D} u_d \widehat{\epsilon}_d \|^2 \qquad \text{where} \quad \Delta = \{\mathbf{u} \mid \mathbf{u} \geqslant 0, \| \mathbf{u} \|_1 = 1\}$
Stage 3: Assemble the ensemble kernel matrix $\widehat{\mathbf{K}}_{ens}$
  $\widehat{\mathbf{A}} = \sum_{d=1}^{D} \widehat{\mu}_d \mathbf{A}_{\widehat{\lambda}_d, k_d}$
  $\mathbf{U}_A, \delta_A = \texttt{spectral\_decomp}(\widehat{\mathbf{A}})$
  $\lambda_{\mathbf{K}} = min\Big(1, (\sum_{k=1}^{n} \frac{\delta_{A,k}}{1-\delta_{A,k}})^{-1}, min\big(\{\widehat{\lambda}_d\}_{d=1}^{D}\big)\Big)$
  $\widehat{\mathbf{K}} = \lambda_{\mathbf{K}} * \widehat{\mathbf{U}}_A \ diag\Big(\frac{\delta_{A,k}}{1-\delta_{A,k}}\Big) \ \widehat{\mathbf{U}}_A^T$
end procedure

Algorithm 2 Variance Component Test for $h \in \mathcal{H}_0$

procedure VCT FOR INTERACTION
Input: Null Kernel Matrix $\mathbf{K}_0$, Derivative Kernel Matrix $\partial\mathbf{K}_0$, Data $(y, \mathbf{x})$
Output: Hypothesis Test p-value p
Stage 1: Estimate Null Model using REML
  $(\hat{\mu}, \hat{\tau}, \hat{\sigma}^2) = argmin L_{REML}(\mu, \tau, \sigma^2 \mid \mathbf{K}_0)$
Stage 2: Compute Test Statistic and Null Distribution Parameters
  $\hat{T}_0 = \hat{\tau} * (\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}})^T \mathbf{V}_0^{-1} \partial\mathbf{K}_0 \mathbf{V}_0^{-1} (\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}})$
  $\hat{\kappa} = \hat{\mathbf{I}}_{\delta\delta} / [\hat{\tau} * tr(\mathbf{V}_0^{-1}\partial\mathbf{K}_0)], \quad \hat{\nu} = [\hat{\tau} * tr(\mathbf{V}_0^{-1}\partial\mathbf{K}_0)]^2 / (2 * \hat{\mathbf{I}}_{\delta\theta})$
Stage 3: Compute p-value and reach conclusion
  $p = P(\hat{\kappa}\chi^2_{\hat{\nu}} > \hat{T}) = P(\chi^2_{\hat{\nu}} > \hat{T}/\hat{\kappa})$
end procedure

Algorithm 3 Parametric Bootstrap Test

procedure PARAMETRIC BOOTSTRAP TEST
Input: Null Kernel Matrix $\mathbf{K}_0$, Derivative Kernel Matrix $\partial\mathbf{K}_0$, Data $(\mathbf{y}, \mathbf{x})$
Output: Hypothesis Test p-value p
`# Stage 1: Estimate Null Model using Gaussian Process Regression`
$\hat{\boldsymbol{\mu}} = \mathbf{A}_0\mathbf{y}, \quad \hat{\sigma}^2 = \frac{\mathbf{y}^T(\mathbf{I}-\mathbf{A}_0)\mathbf{y}}{n-tr(\mathbf{A}_0)}, \quad \hat{\tau}$
`# Stage 2: Sample response from the fitted model obtain in Step 1`
`# and compute the test statistic based on fitting the alternative`
`# model, repeat for B times`
    for $b = 1$ to B **do**
        $\mathbf{y}^\star = \hat{\boldsymbol{\mu}} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim N(0, \hat{\sigma}^2)$
        $\hat{T}_{0b} = \hat{\tau} * (\mathbf{y}^\star - \hat{\boldsymbol{\mu}})^T\mathbf{V}_0^{-1}\partial\mathbf{K}_0\mathbf{V}_0^{-1}(\mathbf{y}^\star - \hat{\boldsymbol{\mu}})$
    end for
`# Stage 3: Compute the test statistic for the original data, based`
`# on fitting the alternative hypothesis model`
    $\hat{T}_0 = \hat{\tau} * (\mathbf{y} - \hat{\boldsymbol{\mu}})^T\mathbf{V}_0^{-1}\partial\mathbf{K}_0\mathbf{V}_0^{-1}(\mathbf{y} - \hat{\boldsymbol{\mu}})$
`# Stage 4: Compute p-value and reach conclusion`
    $p = \frac{1}{B}\sum_{b=1}^{B} I(\hat{T}_{0b} > \hat{T}_0)$
end procedure

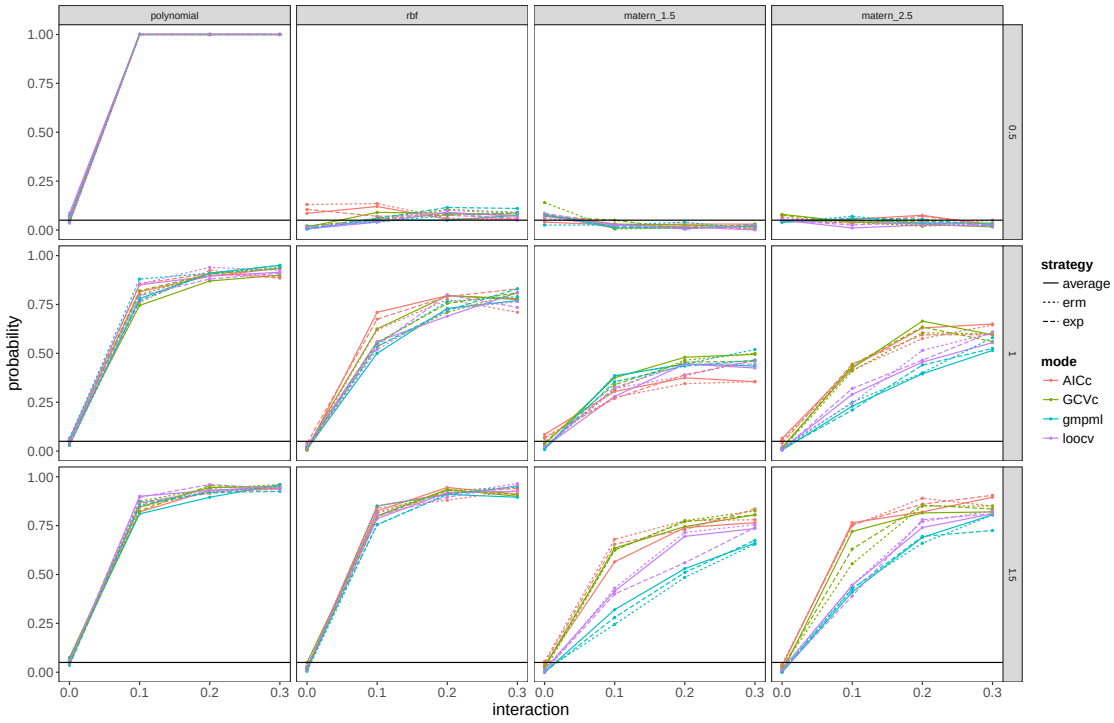

Figure 1: Asym, True kernel only

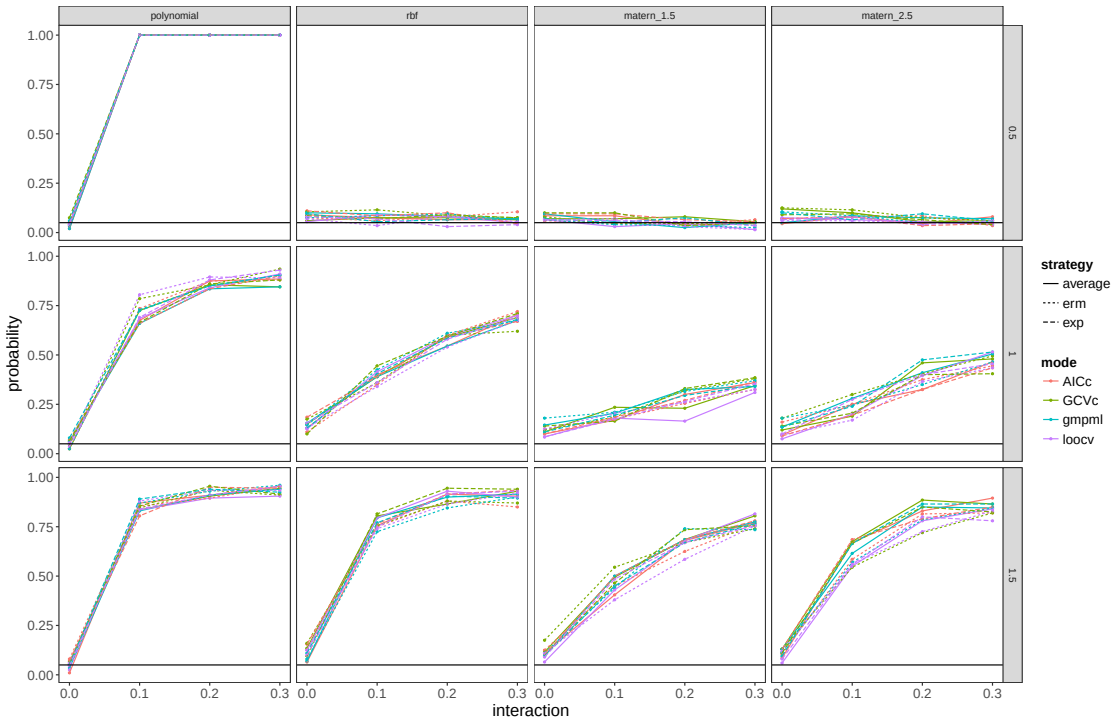

Figure 2: Asym, 3 Polynomial kernels

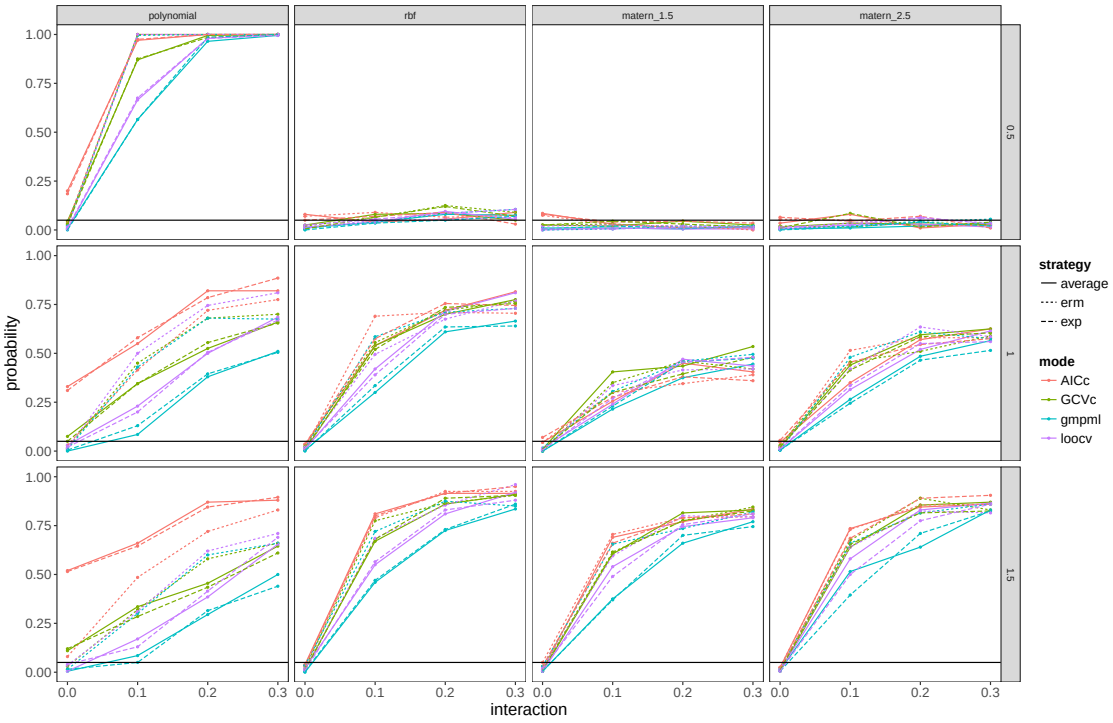

Figure 3: Asym, 3 RBF kernels

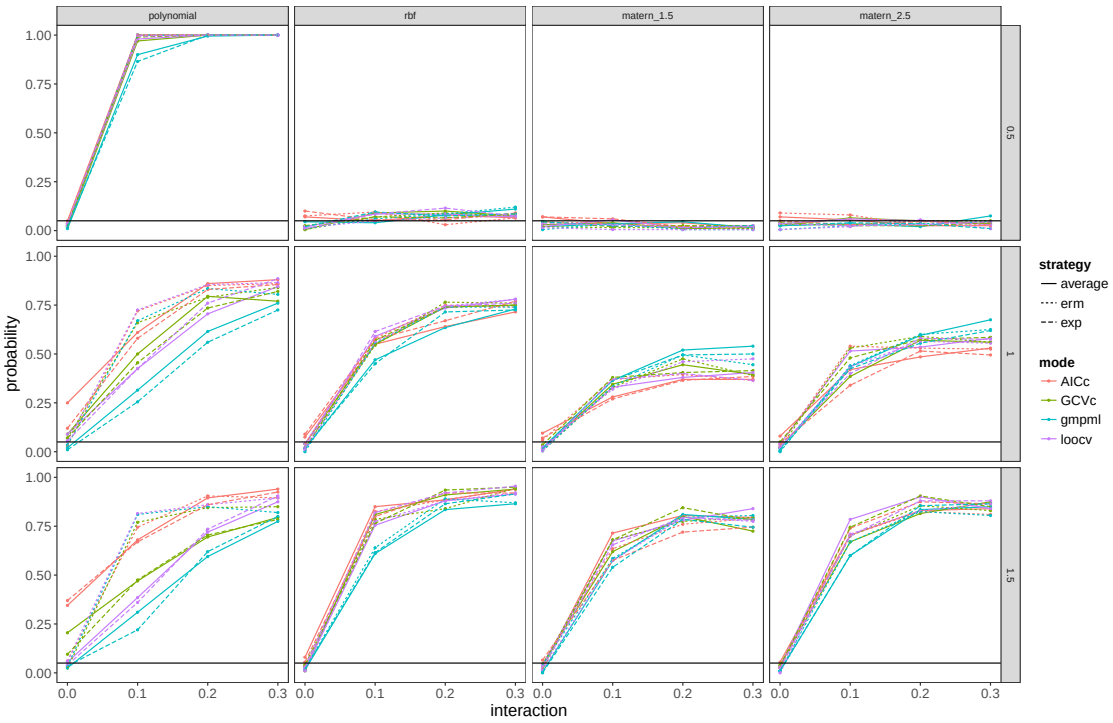

Figure 4: Asym, 3 Polynomial kernels and 3 RBF kernels

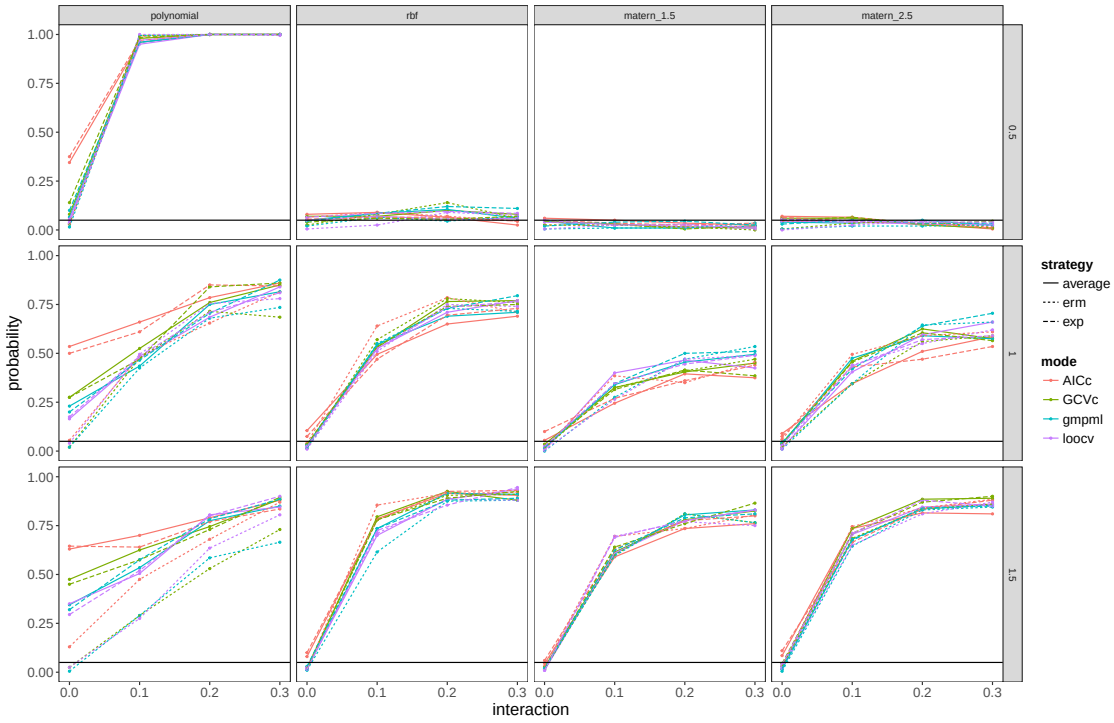

Figure 5: Asym, 3 Matern kernels and 3 RBF kernels

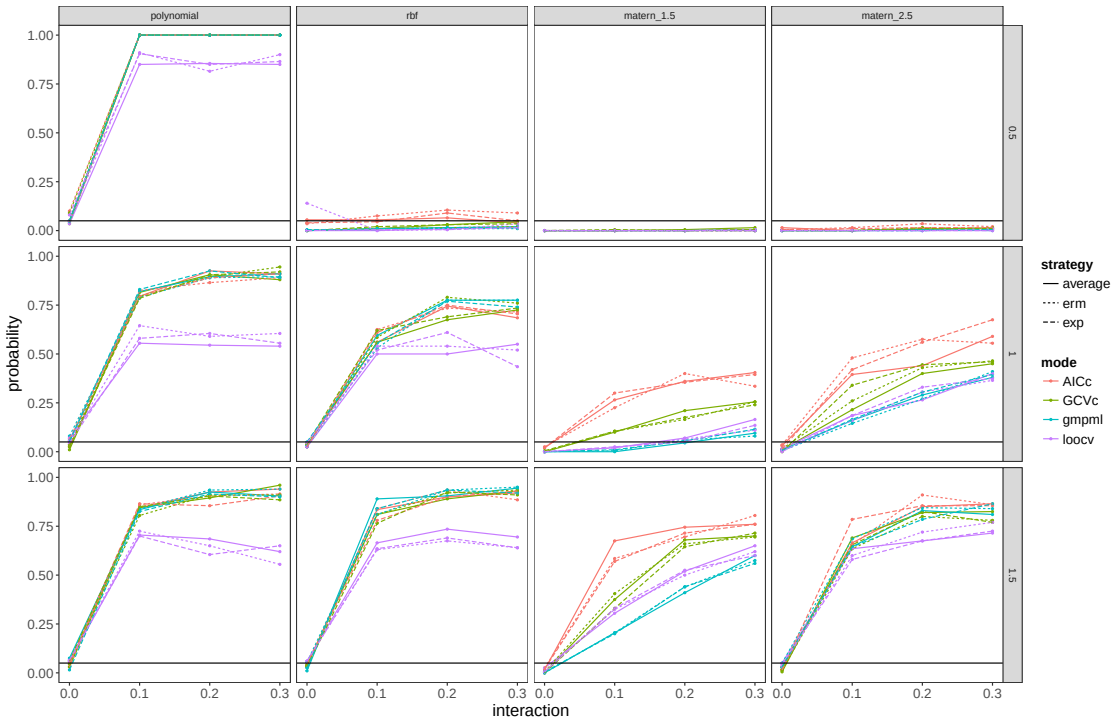

Figure 6: Boot, True kernel only

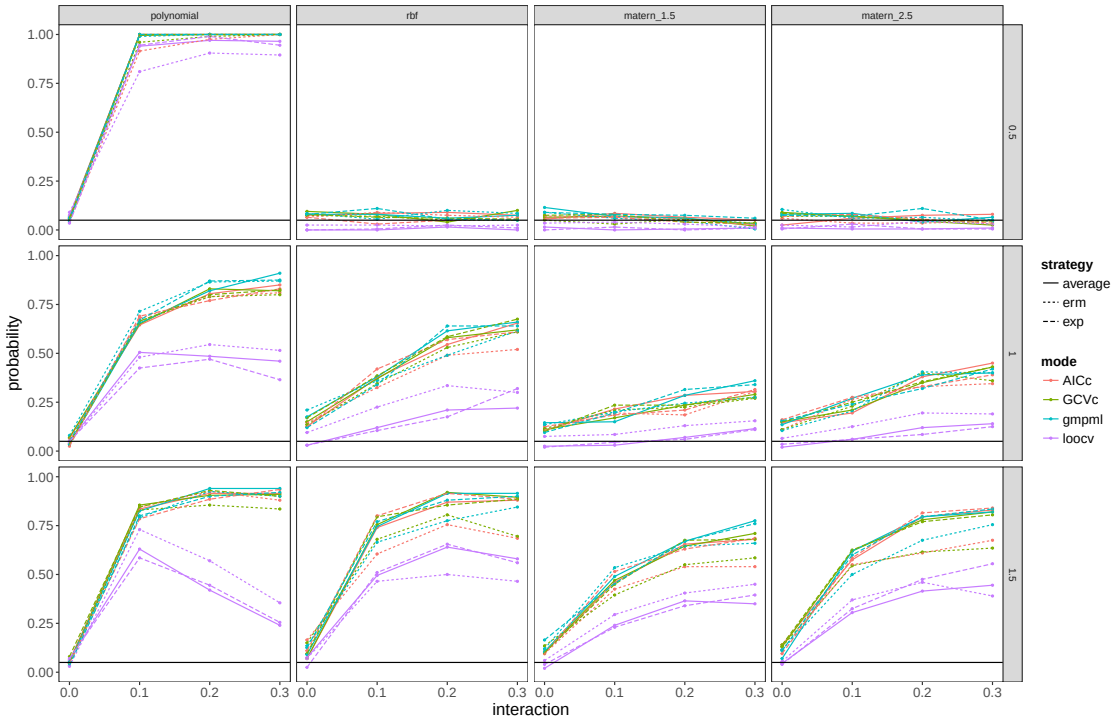

Figure 7: Boot, 3 Polynomial kernels

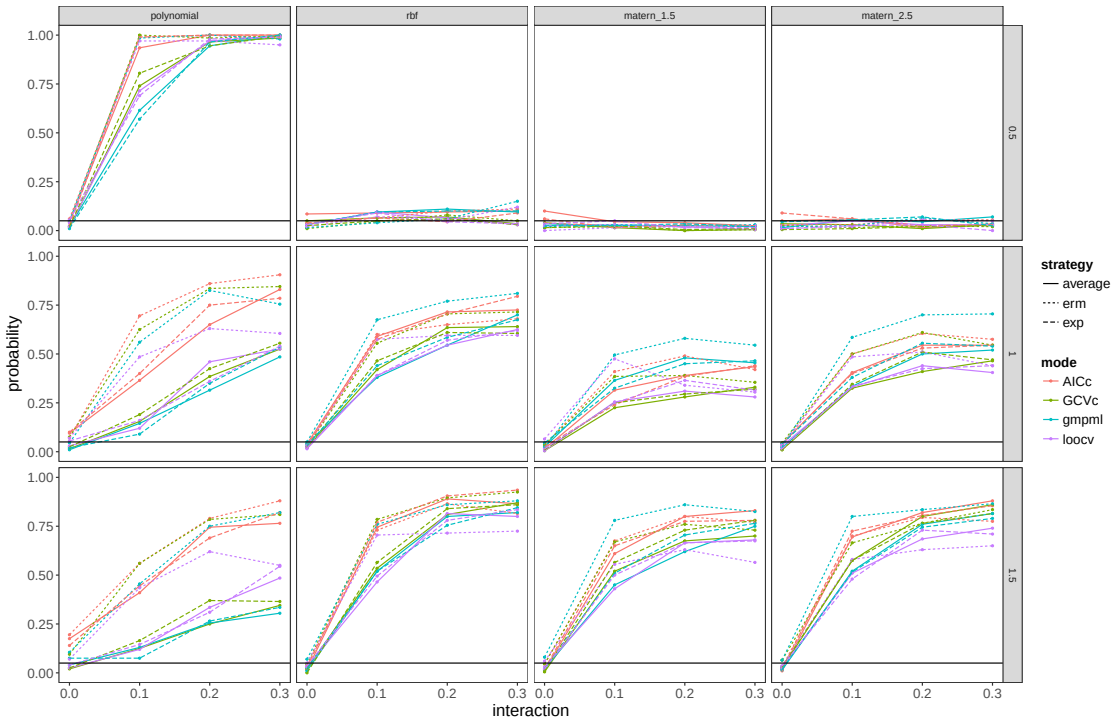

Figure 8: Boot, 3 RBF kernels

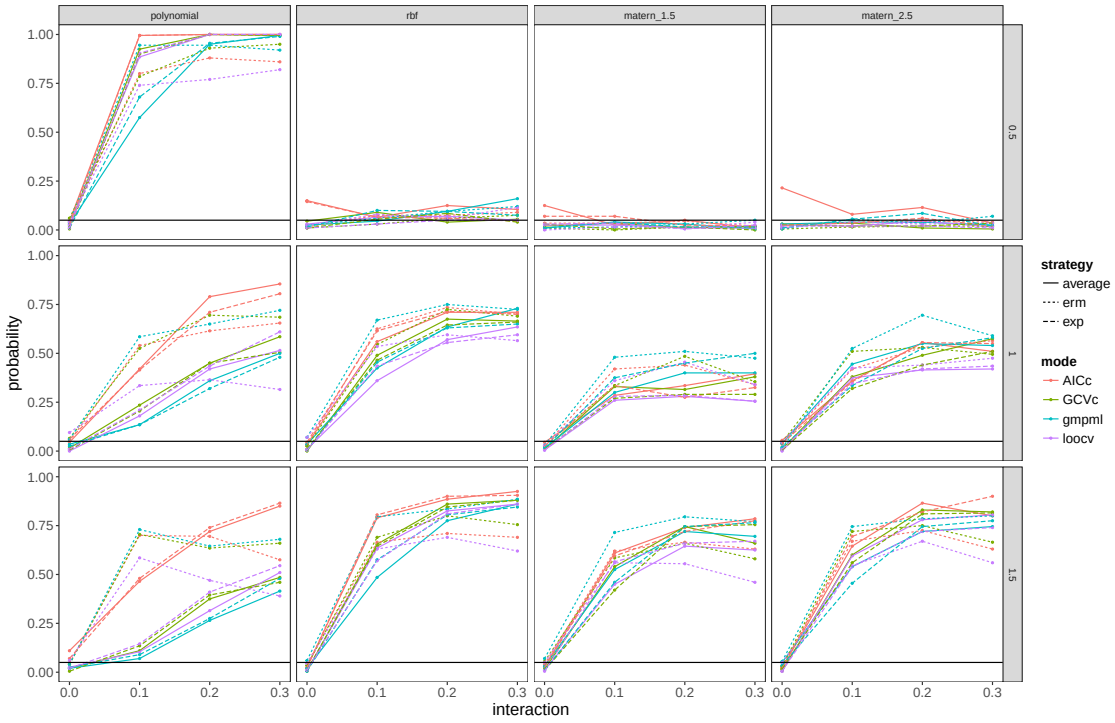

Figure 9: Boot, 3 Polynomial kernels and 3 RBF kernels

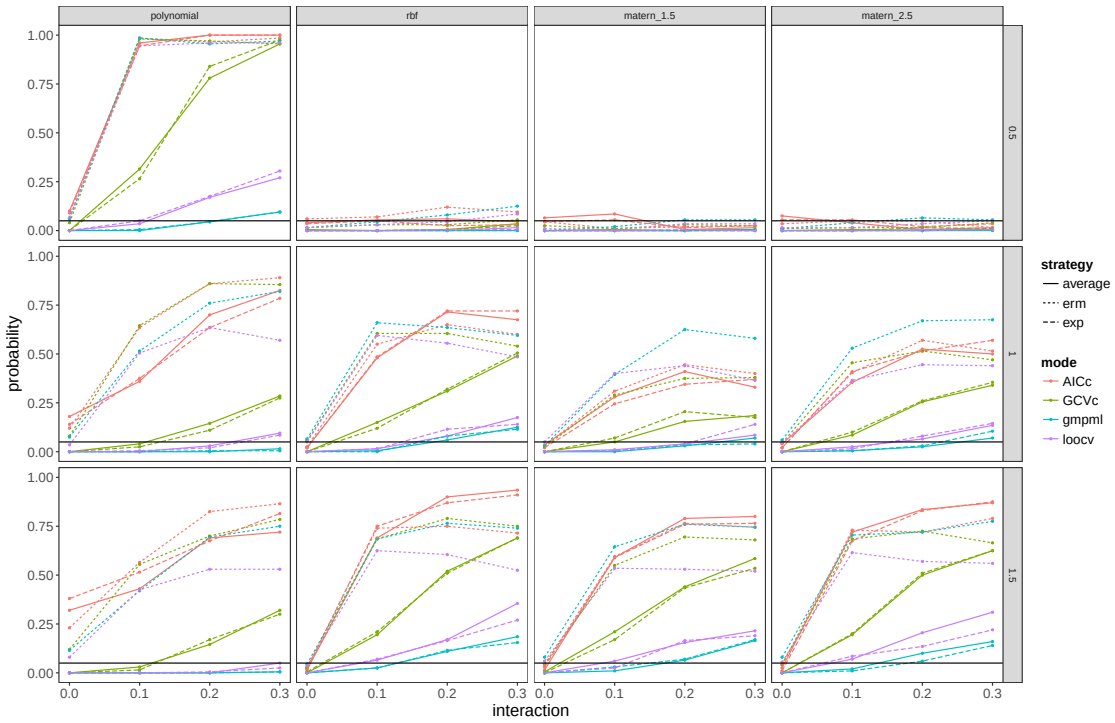

Figure 10: Boot, 3 Matern kernels and 3 RBF kernels

References

[1] Jeremiah Zhe Liu and Brent Coull. Robust Hypothesis Test for Nonlinear Effect with Gaussian Processes. *arXiv:1710.01406 [stat]*, October 2017.

[2] Gene H. Golub, Michael Heath, and Grace Wahba. Generalized Cross-Validation as a Method for Choosing a Good Ridge Parameter. *Technometrics*, 21(2):215–223, May 1979.

[3] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition*. Springer Series in Statistics. Springer-Verlag, New York, 2 edition, 2009.

[4] Hirotogu Akaike. Information Theory and an Extension of the Maximum Likelihood Principle. In *Selected Papers of Hirotugu Akaike*, Springer Series in Statistics, pages 199–213. Springer, New York, NY, 1998.

[5] Clifford M. Hurvich and Chih-Ling Tsai. Regression and time series model selection in small samples. *Biometrika*, 76(2):297–307, June 1989.

[6] Hurvich Clifford M., Simonoff Jeffrey S., and Tsai Chih-Ling. Smoothing parameter selection in nonparametric regression using an improved Akaike information criterion. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 60(2):271–293, January 2002.

[7] Philip S. Boonstra, Bhramar Mukherjee, and Jeremy M. G. Taylor. A Small-Sample Choice of the Tuning Parameter in Ridge Regression. *Statistica Sinica*, 25(3):1185–1206, July 2015.

[8] William E. Wecker and Craig F. Ansley. The Signal Extraction Approach to Nonlinear Regression and Spline Smoothing. *Journal of the American Statistical Association*, 78(381):81–89, March 1983.

[9] David A. Harville. Maximum Likelihood Approaches to Variance Component Estimation and to Related Problems. *Journal of the American Statistical Association*, 72(358):320–338, 1977.

[10] Grace Wahba. A Comparison of GCV and GML for Choosing the Smoothing Parameter in the Generalized Spline Smoothing Problem. *The Annals of Statistics*, 13(4):1378–1402, December 1985.

[11] Philip T. Reiss and R. Todd Ogden. Smoothing Parameter Selection for a Class of Semiparametric Linear Models. *JRSS-B*, 71(2), 2009.

[12] Xiang Zhan, Anna Plantinga, Ni Zhao, and Michael C. Wu. A fast small-sample kernel independence test for microbiome community-level association analysis. *Biometrics*, 73(4):1453–1463, December 2017.

[13] Arnak S. Dalalyan and Alexandre B. Tsybakov. Aggregation by Exponential Weighting and Sharp Oracle Inequalities. In *Learning Theory*, Lecture Notes in Computer Science, pages 97–111. Springer, Berlin, Heidelberg, June 2007.

[14] The MIT Press. Gaussian Processes for Machine Learning, 2006.

[15] Michael L. Stein. *Interpolation of Spatial Data: Some Theory for Kriging*. Springer Series in Statistics. Springer-Verlag, New York, 1999.

[16] Milton Abramowitz. *Handbook of Mathematical Functions, With Formulas, Graphs, and Mathematical Tables,*. Dover Publications, Inc., New York, NY, USA, 1974.

[17] Xihong Lin. Variance component testing in generalised linear models with random effects. *Biometrika*, 84(2):309–326, June 1997.

[18] Arnab Maity and Xihong Lin. Powerful tests for detecting a gene effect in the presence of possible gene-gene interactions using garrote kernel machines. *Biometrics*, 67(4):1271–1284, December 2011.

[19] Petra Bůžková, Thomas Lumley, and Kenneth Rice. Permutation and parametric bootstrap tests for gene-gene and gene-environment interactions. *Annals of Human Genetics*, 75(1):36–45, January 2011.