

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Method</b>	<b>2</b>
2.1	CVEK . . . . .	2
2.1.1	Main Algorithm . . . . .	2
2.1.2	Tuning Parameter Selection . . . . .	3
2.1.3	Ensemble Strategy . . . . .	7
2.1.4	Kernel Choice . . . . .	7
2.2	Hypothesis Test . . . . .	9
2.2.1	Asymptotic Test . . . . .	9
2.2.2	Bootstrap Test . . . . .	11
<b>3</b>	<b>Simulation</b>	<b>13</b>
<b>4</b>	<b>Conclusion</b>	<b>19</b>

# 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 multcategory nutrition intake) remains difficult in practice. The main challenges roots 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, [Liu and Coull, 2017] addressed challenge (1) by proposing 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 learning scheme for the interaction kernel. In this work, we seek to address these shortcomings by proposing a bootstrap test for CVEK. As we will illustrate through simulation, comparing to other popular nonlinear interaction tests, our test has robust Type I error and good power in small sample size and under a wide range of data-generation mechanisms.

## 2 Method

### 2.1 CVEK

#### 2.1.1 Main Algorithm

We study the problem of constructing a hypothesis test for an unknown data-generating function  $h : \mathbb{R}^p \rightarrow \mathbb{R}$ , when  $h$  is estimated with a black-box algorithm (specifically, Gaussian Process regression) from  $n$  observations  $\{y_i, \mathbf{x}_i\}_{i=1}^n$ . We need a kernel estimation strategy [Liu and Coull, 2017] that is *flexible* so that it does not underfit under the null, yet *stable* so that it does not overfit under the alternative. To this end, we propose estimating  $h$  using the ensemble of a library of fixed base kernels  $\{k_d\}_{d=1}^D$ :

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

where  $\hat{h}_d$  is the kernel predictor generated by  $d^{\text{th}}$  base kernel  $k_d$ .

To be more specific, for each given basis kernel  $\{k_d\}_{d=1}^D$ , we first estimate  $\hat{\mathbf{h}}_d = \mathbf{K}_d(\mathbf{K}_d + \hat{\lambda}_d \mathbf{I})^{-1} \mathbf{y}$ , the prediction based on  $d^{\text{th}}$  kernel, where the tuning parameter  $\hat{\lambda}_d$  is selected by minimizing one of the four objective functions given in section 1.2. Denote the estimated error for  $d^{\text{th}}$  kernel as  $\hat{e}_d$  and  $\mathbf{A}_{d,\lambda} = \mathbf{K}_d(\mathbf{K}_d + \lambda \mathbf{I})^{-1}$ . After that, we introduce two ways to ensemble these  $D$  kernel predictors  $\{\hat{h}_d\}_{d=1}^D$  in section 1.3. Then we get the ensemble matrix  $\hat{\mathbf{A}}$  and can estimate the ensemble kernel matrix  $\hat{\mathbf{K}}$  by solving:

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

Specifically, if we denote  $\mathbf{U}_A$  and  $\{\delta_{A,k}\}_{k=1}^n$  the eigenvector and eigenvalues of  $\hat{\mathbf{A}}$ , then  $\hat{\mathbf{K}}$  adopts the form:

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

For example, a complete summary of the proposed procedure using LooCV to choose tuning parameter and cross-validation to ensemble kernel predictors is available in Algorithm 1.

---

**Algorithm 1** Cross Validated Ensemble Kernel (CVEK)

---

```

1: procedure CVEK
   Input: A library of kernels  $\{\mathbf{k}_d\}_{d=1}^D$ , Data  $(\mathbf{y}, \mathbf{x})$ 
   Output: Ensemble Kernel Matrix  $\hat{\mathbf{K}}$ 
   # Stage 1: Estimate  $\lambda$  and CV error for each kernel
2:   for  $d = 1$  to  $D$  do
3:      $\mathbf{K}_d = \mathbf{K}_d / \text{tr}(\mathbf{K}_d)$ 
4:      $\hat{\lambda}_d = \text{argmin}_{\lambda} \text{LOOCV}(\lambda | \mathbf{K}_d)$ 
5:      $\hat{\epsilon}_d = \text{CV}(\hat{\lambda}_d | \mathbf{K}_d)$ 
6:   end for
   # Stage 2: Estimate ensemble weights  $\mathbf{u}_{D \times 1} = \{\mathbf{u}_1, \dots, \mathbf{u}_D\}$ 
7:    $\hat{\mathbf{u}} = \text{argmin}_{\mathbf{u} \in \Delta} \|\sum_{d=1}^D \mathbf{u}_d \hat{\epsilon}_d\|^2$  where  $\Delta = \{\mathbf{u} | \mathbf{u} \geq 0, \|\mathbf{u}\|_1 = 1\}$ 
   # Stage 3: Assemble the ensemble kernel matrix  $\hat{\mathbf{K}}_{\text{ens}}$ 
8:    $\hat{\mathbf{A}} = \sum_{d=1}^D \hat{\mathbf{u}}_d \mathbf{A}_{\hat{\lambda}_d, \mathbf{k}_d}$ 
9:    $\mathbf{U}_A, \delta_A = \text{spectral\_decomp}(\hat{\mathbf{A}})$ 
10:   $\lambda_K = \min\left(1, \left(\sum_{k=1}^n \frac{\delta_{A,k}}{1 - \delta_{A,k}}\right)^{-1}, \min(\{\hat{\lambda}_d\}_{d=1}^D)\right)$ 
11:   $\hat{\mathbf{K}} = \lambda_K * \hat{\mathbf{U}}_A \text{diag}\left(\frac{\delta_{A,k}}{1 - \delta_{A,k}}\right) \hat{\mathbf{U}}_A^T$ 
12: end procedure

```

---

### 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

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

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

#### Cross-Validation: K Fold and Loo

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,  $\kappa(1), \dots, \kappa(K)$ , 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:

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

LooCV is the situation when  $K = n$ . In this case, we can write our objective function as [Golub et al., 1979]:

$$\lambda_{n-CV} = \operatorname{argmin}_{\lambda \in \Delta} \left\{ \log \mathbf{y}^{*T} [\mathbf{I}_n - \operatorname{diag}(\mathbf{A}_{\lambda}) - \frac{1}{n} \mathbf{I}_n]^{-1} (\mathbf{I}_n - \mathbf{A}_{\lambda})^2 [\mathbf{I}_n - \operatorname{diag}(\mathbf{A}_{\lambda}) - \frac{1}{n} \mathbf{I}_n]^{-1} \mathbf{y}^* \right\} \quad (2.1.4)$$

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 [Hastie et al., 2009] or 10, cross-validation will have lower variance, but making bias a problem.

### AIC and Small Sample Correction

Based on the idea of "model fit + model complexity", Akaike's Information Criterion (AIC) [Akaike, 1998] choose  $\lambda$  by minimizing,

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

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

$$\lambda_{\text{AIC}} = \operatorname{argmin}_{\lambda \in \Lambda} \left\{ \log \mathbf{y}^{*T} (\mathbf{I}_n - \mathbf{A}_{\lambda})^2 \mathbf{y}^* + \frac{2[\operatorname{tr}(\mathbf{A}_{\lambda}) + 2]}{n} \right\} \quad (2.1.5)$$

When  $n$  is small, extreme overfitting is possible, giving small bias/ large variance estimates. The small-sample correction of AIC [Hurvich and Tsai, 1989, Hurvich Clifford M. et al., 2002] 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:

$$\lambda_{\text{AICc}} = \operatorname{argmin}_{\lambda \in \Lambda} \left\{ \log \mathbf{y}^{*T} (\mathbf{I}_n - \mathbf{A}_{\lambda})^2 \mathbf{y}^* + \frac{2[\operatorname{tr}(\mathbf{A}_{\lambda}) + 2]}{n - \operatorname{tr}(\mathbf{A}_{\lambda}) - 3} \right\} \quad (2.1.6)$$

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

### GCV and Small Sample Correction

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

$$\lambda_{\text{GCV}} = \underset{\lambda \in \Lambda}{\text{argmin}} \left\{ \log \mathbf{y}^{\star\top} (\mathbf{I}_n - \mathbf{A}_\lambda)^2 \mathbf{y}^* - 2 \log \left[ 1 - \frac{\text{tr}(\mathbf{A}_\lambda)}{n} - \frac{1}{n} \right] \right\} \quad (2.1.7)$$

The “ $-\frac{1}{n}$ ” terms in (2.1.7) is because GCV counts  $\mu$  as part of model complexity, but not  $\sigma^2$ . This motivates the proposed small-sample correction to GCV [Boonstra et al., 2015], which does count  $\sigma^2$  as a parameter:

$$\lambda_{\text{GCVc}} = \underset{\lambda \in \Lambda}{\text{argmin}} \left\{ \log \mathbf{y}^{\star\top} (\mathbf{I}_n - \mathbf{A}_\lambda)^2 \mathbf{y}^* - 2 \log \left[ 1 - \frac{\text{tr}(\mathbf{A}_\lambda)}{n} - \frac{2}{n} \right]_+ \right\} \quad (2.1.8)$$

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

### GMPML-based Selection

Suppose we relate  $Y$  and  $\mathbf{X}$  by a linear model,  $Y = \mu + \phi(\mathbf{X})^\top \boldsymbol{\beta} + \sigma \epsilon$ , with  $\epsilon \sim N(0, 1)$ , 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  $\sigma^2/\lambda$ , 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  $\lambda$  is interpreted as the variance component of a mixed-effects model:

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

From this,  $\mathbf{y}^*|\lambda, \sigma^2$  is multivariate normal with mean  $\mathbf{0}_n$  and covariance  $\sigma^2(\mathbf{I}_n - \mathbf{A}_\lambda)^{-1}$ . The maximum profile marginal likelihood (MPML) estimate, originally proposed for smoothing spline [Wecker and Ansley, 1983], profiles  $m(\lambda, \sigma^2)$  over  $\sigma^2$ , replacing each instance with  $\hat{\sigma}_\lambda^2 = \mathbf{y}^{\star\top} (\mathbf{I}_n - \mathbf{A}_\lambda) \mathbf{y}^* / n$ , and maximized the “concentrated” log-likelihood,  $m(\lambda, \hat{\sigma}_\lambda^2)$ :

$$\lambda_{\text{MPML}} = \underset{\lambda \in \Lambda}{\text{argmin}} \left\{ \log \mathbf{y}^{\star\top} (\mathbf{I}_n - \mathbf{A}_\lambda) \mathbf{y}^* - \frac{1}{n} \log |\mathbf{I}_n - \mathbf{A}_\lambda| \right\}$$

Closely related is the generalized/ restricted MPML [Harville, 1977, Wahba, 1985], which adjusts the penalty to account for estimation of regression parameter  $\beta_0$  that is not marginalized, resulting in one degree of freedom:

$$\lambda_{\text{GMPML}} = \underset{\lambda \in \Lambda}{\text{argmin}} \left\{ \log \mathbf{y}^{\star\top} (\mathbf{I}_n - \mathbf{A}_\lambda) \mathbf{y}^* - \frac{1}{n-1} \log |\mathbf{I}_n - \mathbf{A}_\lambda| \right\} \quad (2.1.9)$$

### Discussion on AIC, GCV and GMPML

It's interesting to compare the  $\lambda$ '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:

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

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

$$\frac{\partial f_{\text{AIC}}}{\partial \lambda} = \frac{2\text{tr}\left[\mathbf{U}_K^\top \mathbf{y}^* \mathbf{y}^{*\top} \mathbf{U}_K (\mathbf{D}_{K,\lambda} - \mathbf{I}) \frac{\partial \mathbf{D}_{K,\lambda}}{\partial \lambda}\right]}{\text{tr}[\mathbf{U}_K^\top \mathbf{y}^* \mathbf{y}^{*\top} \mathbf{U}_K (\mathbf{I}_n - \mathbf{D}_{K,\lambda})^2]} + \frac{2\text{tr}\left(\frac{\partial \mathbf{D}_{K,\lambda}}{\partial \lambda}\right)}{n} \quad (2.1.10)$$

$$\frac{\partial f_{\text{GCV}}}{\partial \lambda} = \frac{2\text{tr}\left[\mathbf{U}_K^\top \mathbf{y}^* \mathbf{y}^{*\top} \mathbf{U}_K (\mathbf{D}_{K,\lambda} - \mathbf{I}) \frac{\partial \mathbf{D}_{K,\lambda}}{\partial \lambda}\right]}{\text{tr}[\mathbf{U}_K^\top \mathbf{y}^* \mathbf{y}^{*\top} \mathbf{U}_K (\mathbf{I}_n - \mathbf{D}_{K,\lambda})^2]} + \frac{2\text{tr}\left(\frac{\partial \mathbf{D}_{K,\lambda}}{\partial \lambda}\right)}{n - \text{tr}(\mathbf{A}_\lambda) - 1} \quad (2.1.11)$$

$$\frac{\partial f_{\text{GMPML}}}{\partial \lambda} = \frac{-\text{tr}\left[\mathbf{U}_K^\top \mathbf{y}^* \mathbf{y}^{*\top} \mathbf{U}_K \frac{\partial \mathbf{D}_{K,\lambda}}{\partial \lambda}\right]}{\text{tr}[\mathbf{U}_K^\top \mathbf{y}^* \mathbf{y}^{*\top} \mathbf{U}_K (\mathbf{I}_n - \mathbf{D}_{K,\lambda})]} + \frac{\text{tr}\left[(\mathbf{I}_n - \mathbf{D}_{K,\lambda})^{-1} \frac{\partial \mathbf{D}_{K,\lambda}}{\partial \lambda}\right]}{n - 1} \quad (2.1.12)$$

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

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

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  $\lambda$ .

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

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

In terms of GCV and GMPML, probably we need the relative size of  $\eta_{K,j}$ 's and  $\lambda$  to make a deterministic conclusion. To be more specific, we need to prove,

$$\frac{\partial}{\partial \lambda} \left[ \log \mathbf{y}^{*\top} (\mathbf{I}_n - \mathbf{A}_\lambda)^2 \mathbf{y}^* \right] > \frac{\partial}{\partial \lambda} \left[ \log \mathbf{y}^{*\top} (\mathbf{I}_n - \mathbf{A}_\lambda) \mathbf{y}^* \right]$$

And then, up to a constant not depending on  $\lambda$ , the difference boils down to,

$$\text{GCV} : \log[\text{tr}(\mathbf{I}_n - \mathbf{A}_\lambda)] = \log\left[\sum_{j=1}^n \frac{\lambda}{\eta_{K,j} + \lambda}\right] \quad (2.1.13)$$

$$\text{GMPML} : \log |\mathbf{I}_n - \mathbf{A}_\lambda| = \log\left[\prod_{j=1}^n \frac{\lambda}{\eta_{K,j} + \lambda}\right] \quad (2.1.14)$$

Take derivative of these two equations with respect to  $\lambda$ ,

$$\begin{aligned}\frac{\partial}{\partial \lambda} \log[\text{tr}(\mathbf{I}_n - \mathbf{A}_\lambda)] &= \frac{\sum_{j=1}^n \frac{\eta_{\kappa,j}}{(\eta_{\kappa,j} + \lambda)^2}}{\sum_{j=1}^n \frac{\lambda}{\eta_{\kappa,j} + \lambda}} \\ \frac{\partial}{\partial \lambda} \log |\mathbf{I}_n - \mathbf{A}_\lambda| &= \sum_{j=1}^n \frac{\eta_{\kappa,j}}{\lambda(\eta_{\kappa,j} + \lambda)}\end{aligned}$$

Since,

$$\sum_{j=1}^n \frac{\frac{\eta_{\kappa,j}}{(\eta_{\kappa,j} + \lambda)^2}}{\sum_{j=1}^n \frac{\lambda}{\eta_{\kappa,j} + \lambda}} \leq \sum_{j=1}^n \frac{\frac{\eta_{\kappa,j}}{(\eta_{\kappa,j} + \lambda)^2}}{\frac{\lambda}{\eta_{\kappa,j} + \lambda}} = \sum_{j=1}^n \frac{\eta_{\kappa,j}}{\lambda(\eta_{\kappa,j} + \lambda)}$$

which gives us,

$$-\frac{2}{n} \log |\mathbf{I}_n - \mathbf{A}_\lambda| \geq -\frac{1}{n} \log[\text{tr}(\mathbf{I}_n - \mathbf{A}_\lambda)]$$

Therefore, we obtain  $\frac{\partial f_{\text{GMPML}}}{\partial \lambda} < \frac{\partial f_{\text{GCV}}}{\partial \lambda}$ . When  $\frac{\partial f_{\text{GCV}}}{\partial \lambda} = 0$ ,  $\frac{\partial f_{\text{GMPML}}}{\partial \lambda} < 0$ , which means  $\lambda_{\text{GCV}} < \lambda_{\text{GMPML}}$ .

### 2.1.3 Ensemble Strategy

#### Empirical Risk Minimization

After obtaining the estimated errors  $\{\hat{\epsilon}_d\}_{d=1}^D$ , we estimate the ensemble weights  $\mathbf{u} = \{u_d\}_{d=1}^D$  such that it minimizes the overall error [Liu and Coull, 2017]:

$$\hat{\mathbf{u}} = \underset{\mathbf{u} \in \Delta}{\text{argmin}} \left\| \sum_{d=1}^D u_d \hat{\epsilon}_d \right\|^2 \quad \text{where } \Delta = \{\mathbf{u} | \mathbf{u} \geq 0, \|\mathbf{u}\|_1 = 1\}$$

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 [Zhan et al., 2017], we propose another way to obtain the ensemble matrix by simply choosing unsupervised weights  $u_d = 1/D$  for  $d = 1, 2, \dots, D$ .

### 2.1.4 Kernel Choice

In this section we give introductions to some commonly-used covariance functions, including two stationary covariance functions, Squared Exponential Covariance Function and The Matérn Class of Covariance Functions, as well as non-stationary covariance functions, Polynomial Covariance Functions and Neural Network Covariance Function.

### Squared Exponential Covariance Function

The *squared exponential* (SE) covariance function has the form

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

with  $r = |\mathbf{x} - \mathbf{x}'|$  and 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  $S(s) = (2\pi l^2)^{D/2} \exp(-2\pi^2 l^2 s^2)$ . [Stein, 1999] 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  $\beta > 0$ .

### The Matérn Class of Covariance Functions

The Matérn class of covariance functions is given by

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

with positive parameters  $\nu$  and  $l$ , where  $K_\nu$  is a modified Bessel function [Abramowitz, 1974]. This covariance function has a spectral density

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

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  $\nu$  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 [Abramowitz, 1974], giving

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

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  $\nu \geq 7/2$ , 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  $\nu \geq 7/2$ . The eigenvalues are subject to polynomial decay:  $\lambda_i = O(i^{-\alpha})$  for  $\alpha > 1$ .

### Polynomial Covariance Functions

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

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

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  $x_{d_1} \dots x_{d_p}$  is unimportant, e.g. for  $p = 2$ ,  $x_1 x_2$  and  $x_2 x_1$  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  $x_1^{m_1} \dots x_D^{m_D}$ . The degeneracy of  $\phi_{\mathbf{m}}(\mathbf{x})$  is  $\frac{p!}{m_1! \dots m_D!}$ , giving the feature map

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

For example, for  $p = 2$  in  $D = 2$ , we have  $\phi(\mathbf{x}) = (x_1^2, x_2^2, \sqrt{2}x_1 x_2)^T$ .

### Neural Network Covariance Function

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  $\sigma_b^2$  and  $\sigma_v^2$ , 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

$$\mathbb{E}_{\mathbf{w}}[f(\mathbf{x})] = 0$$

$$\mathbb{E}_{\mathbf{w}}[f(\mathbf{x})f(\mathbf{x}')] = \sigma_b^2 + \sum_j \sigma_v^2 \mathbb{E}_{\mathbf{u}}[h(\mathbf{x}; \mathbf{u}_j)h(\mathbf{x}'; \mathbf{u}_j)] \quad (2.1.15)$$

$$= \sigma_b^2 + N_H \sigma_v^2 \mathbb{E}_{\mathbf{u}}[h(\mathbf{x}; \mathbf{u})h(\mathbf{x}'; \mathbf{u})] \quad (2.1.16)$$

where (2.1.16) follows because all of the hidden units are identically distributed. The sum in (2.1.15) 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  $N_H \rightarrow \infty$ .

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  $h(z) = \text{erf}(z) = 2/\sqrt{\pi} \int_0^z e^{-t^2} dt$  as the transfer function,

let  $h(\mathbf{x}; \mathbf{u}) = \text{erf}\left(u_0 + \sum_{j=1}^D u_j x_j\right)$  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  $\tilde{\mathbf{x}} = (1, x_1, \dots, x_d)^T$  is an augmented input vector.

## 2.2 Hypothesis Test

### 2.2.1 Asymptotic Test

We use the classical variance component test [Lin, 1997] to construct a testing procedure for the hypothesis about Gaussian process function:

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

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*  $k_\delta(\mathbf{z}, \mathbf{z}')$  [Maity and Lin, 2011], which is constructed by including an extra *garrote parameter*  $\delta$  to a given kernel function. When  $\delta = 0$ , the garrote kernel function  $k_0(\mathbf{x}, \mathbf{x}') = k_\delta(\mathbf{x}, \mathbf{x}')|_{\delta=0}$  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  $k_\delta(\mathbf{x}) = k(\delta * x_1, x_2, \dots, x_p)$ ,  $\delta = 0$  corresponds to the null hypothesis  $H_0 : h(\mathbf{x}) = h(x_2, \dots, x_p)$ , i.e. the function  $h(\mathbf{x})$  does not depend on  $x_1$ . As a result, the general hypothesis is equivalent to:

$$H_0 : \delta = 0. \quad (2.2.2)$$

We now construct a test statistic  $\hat{T}_0$  for (2.2.2) by noticing that the garrote parameter  $\delta$  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:

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

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  $L_{\text{REML}}$  with respect to  $\delta$  under  $H_0$  [Lin, 1997]:

$$\hat{T}_0 = \hat{\tau} * (\mathbf{y} - \hat{\boldsymbol{\mu}})^T \mathbf{V}_0^{-1} [\partial \mathbf{K}_0] \mathbf{V}_0^{-1} (\mathbf{y} - \hat{\boldsymbol{\mu}}) \quad (2.2.3)$$

where  $\mathbf{V}_0 = \hat{\sigma}^2 \mathbf{I} + \hat{\tau} \mathbf{K}_0$ . In this expression,  $\mathbf{K}_0 = \mathbf{K}_\delta|_{\delta=0}$ , and  $\partial \mathbf{K}_0$  is the null derivative kernel matrix whose  $(i, j)^{\text{th}}$  entry is  $\frac{\partial}{\partial \delta} k_\delta(\mathbf{x}, \mathbf{x}')|_{\delta=0}$ . As discussed previously, misspecifying the null kernel function  $k_0$  negatively impacts the performance of the resulting hypothesis test. To better understand the mechanism at play, we express the test statistic  $\hat{T}_0$  from (2.2.3) in terms of the model residual  $\hat{\boldsymbol{\epsilon}} = \mathbf{y} - \hat{\boldsymbol{\mu}} - \hat{\mathbf{h}}$ :

$$\hat{T}_0 = \left( \frac{\hat{\tau}}{\hat{\sigma}^4} \right) * \hat{\boldsymbol{\epsilon}}^T [\partial \mathbf{K}_0] \hat{\boldsymbol{\epsilon}}, \quad (2.2.4)$$

where we have used the fact  $\mathbf{V}_0^{-1}(\mathbf{y} - \hat{\boldsymbol{\mu}}) = (\hat{\sigma}^2)^{-1}(\hat{\boldsymbol{\epsilon}})$  [Harville, 1977]. As shown, the test statistic  $\hat{T}_0$  is a scaled quadratic-form statistic that is a function of the model residual. If  $k_0$  is too restrictive,

model estimates will **underfit** the data even under the null hypothesis, introducing extraneous correlation among the  $\hat{\epsilon}_i$ 's, therefore leading to overestimated  $\hat{T}_0$  and eventually underestimated p-value under the null. In this case, the test procedure will frequently reject the null hypothesis (i.e. suggest the existence of nonlinear interaction) even when there is in fact no interaction, yielding an invalid test due to **inflated Type I error**. On the other hand, if  $k_0$  is too flexible, model estimates will likely **overfit** the data in small samples, producing underestimated residuals, an underestimated test statistic, and overestimated p-values. In this case, the test procedure will too frequently fail to reject the null hypothesis (i.e. suggesting there is no interaction) when there is in fact interaction, yielding an insensitive test with **diminished power**.

The null distribution of  $\hat{T}$  can be approximated using a scaled chi-square distribution  $\kappa\chi_v^2$  using Satterthwaite method by matching the first two moments of  $T$ :

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

with solution:

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

where  $\hat{\mathbf{I}}_{\delta\delta} = \mathbf{I}_{\delta\delta} - \mathbf{I}_{\delta\theta}^T \mathbf{I}_{\theta\theta}^{-1} \mathbf{I}_{\delta\theta}$  is the efficient information of  $\delta$  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  $\hat{\kappa}\chi_{\hat{v}}^2$ :

$$p = P(\hat{\kappa}\chi_{\hat{v}}^2 > \hat{T}) = P(\chi_{\hat{v}}^2 > \hat{T}/\hat{\kappa})$$

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

In light of the discussion about model misspecification in Introduction section, we highlight the fact that our proposed test (2.2.3) is robust against model misspecification under the alternative [Lin, 1997], since the calculation of test statistics do not require detailed parametric assumption about  $k_\delta$ . However, the test is NOT robust against model misspecification under the null, since the expression of both test statistic  $\hat{T}_0$  and the null distribution parameters  $(\hat{\kappa}, \hat{v})$  still involve the kernel matrices generated by  $k_0$  (see Algorithm 2). We address this problem by proposing a robust estimation procedure for the kernel matrices under the null.

---

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

---

- 1: **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
  - 2:  $(\hat{\mu}, \hat{\tau}, \hat{\sigma}^2) = \text{argmin}_{\text{REML}}(\mu, \tau, \sigma^2 | \mathbf{K}_0)$   
# Stage 2: Compute Test Statistic and Null Distribution Parameters
  - 3:  $\hat{T}_0 = \hat{\tau} * (\mathbf{y} - \mathbf{X}\hat{\beta})^T \mathbf{V}_0^{-1} \partial \mathbf{K}_0 \mathbf{V}_0^{-1} (\mathbf{y} - \mathbf{X}\hat{\beta})$
  - 4:  $\hat{\kappa} = \hat{\mathbf{I}}_{\delta\delta} / [\hat{\tau} * \text{tr}(\mathbf{V}_0^{-1} \partial \mathbf{K}_0)], \quad \hat{v} = [\hat{\tau} * \text{tr}(\mathbf{V}_0^{-1} \partial \mathbf{K}_0)]^2 / (2 * \hat{\mathbf{I}}_{\delta\delta})$   
# Stage 3: Compute p-value and reach conclusion
  - 5:  $p = P(\hat{\kappa}\chi_{\hat{v}}^2 > \hat{T}) = P(\chi_{\hat{v}}^2 > \hat{T}/\hat{\kappa})$
  - 6: **end procedure**
-

### 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 an environmental exposure  $E$  on an outcome  $Y$ . The null hypothesis is that the interaction term is zero [Bůžková et al., 2011].

$$E[Y] = \beta_0 + \beta_G G + \beta_E E \quad (2.2.5)$$

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  $\beta_G$  and  $\beta_E$  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  $\beta_G$  and  $\beta_E$ .

#### 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,

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

and generate  $\mathbf{Y}^*$  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.

---

**Algorithm 3** Parametric Bootstrap Test

---

```

1: 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
2:    $\hat{\boldsymbol{\mu}} = \mathbf{A}_0 \mathbf{y}$ ,  $\hat{\sigma}^2 = \frac{\mathbf{y}^\top (\mathbf{I}_n - \mathbf{A}_0) \mathbf{y}}{n - \text{tr}(\mathbf{A}_0)}$ ,  $\hat{\tau}$ 
   # 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
3:   for  $b = 1$  to B do
4:      $\mathbf{y}^* = \hat{\boldsymbol{\mu}} + \boldsymbol{\epsilon}$ ,  $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \hat{\sigma}^2)$ 
5:      $\hat{\tau}_{0b} = \hat{\tau} * (\mathbf{y}^* - \hat{\boldsymbol{\mu}})^\top \mathbf{V}_0^{-1} \partial\mathbf{K}_0 \mathbf{V}_0^{-1} (\mathbf{y}^* - \hat{\boldsymbol{\mu}})$ 
6:   end for
   # Stage 3: Compute the test statistic for the original data, based
   # on fitting the alternative hypothesis model
7:    $\hat{\tau}_0 = \hat{\tau} * (\mathbf{y} - \hat{\boldsymbol{\mu}})^\top \mathbf{V}_0^{-1} \partial\mathbf{K}_0 \mathbf{V}_0^{-1} (\mathbf{y} - \hat{\boldsymbol{\mu}})$ 
   # Stage 4: Compute p-value and reach conclusion
8:    $p = \frac{1}{B} \sum_{b=1}^B \mathbb{I}(\hat{\tau}_{0b} > \hat{\tau}_0)$ 
9: end procedure

```

---

### 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  $k_{\text{true}}$ . We standardize all sampled functions to have unit form, so that  $\delta$  represents the strength of interaction relative to the main effect.

For each simulation scenario, we first generated data using  $\delta$  and  $k_{\text{true}}$  as above, then selected a  $k_{\text{model}}$  to estimate the null model and obtain p-value using Algorithm 3. We repeated each scenario 200 times, and evaluate the test performance using the empirical probability  $\hat{P}(p \leq 0.05)$  estimates the test's Type I error, and should be smaller or equal to the significance level 0.05. Under alternative hypothesis  $H_a : \delta > 0$ ,  $\hat{P}(p \leq 0.05)$  estimates the test's power, and should ideally approach 1 quickly as the strength of interaction  $\delta$  increases.

In this study, we varied  $k_{\text{true}}$  to produce data-generating functions  $h_\delta(\mathbf{x}_{i,1}, \mathbf{x}_{i,2})$  with different smoothness and complexity properties, and varied  $k_{\text{model}}$  to reflect different common modeling

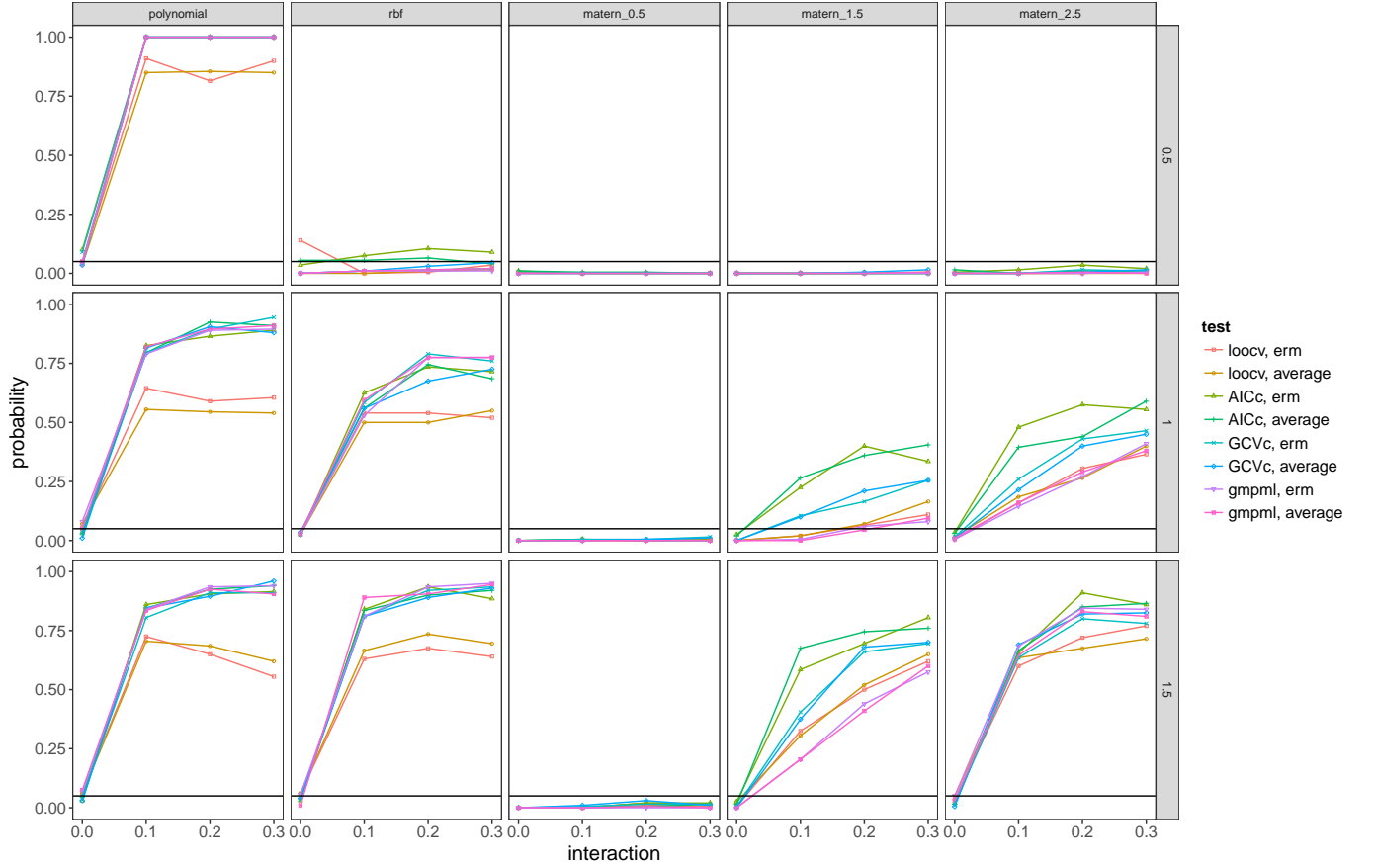


Figure 1: True kernel only

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.

### Data-generating Mechanism

We consider 15 data-generating mechanisms: Matern  $\nu = 1/2$  ( $l = 0.5, 1, 1.5$ ); Matern  $\nu = 3/2$  ( $l = 0.5, 1, 1.5$ ); Matern  $\nu = 5/2$  ( $l = 0.5, 1, 1.5$ ); Polynomial ( $p = 1, 2, 3$ ); RBF ( $l = 0.5, 1, 1.5$ ).

### Tuning Parameter

Loop over the four methods: LooCV, AICc, GCVc and GMPML respectively.

### Modeling Strategies

We consider 5 kinds of kernel libraries: True kernel only; 3 Polynomial kernels ( $p = 1, 2, 3$ ); 3 RBF kernels ( $l = 0.6, 1, 2$ ); 3 Polynomial kernels ( $p = 1, 2, 3$ ) and 3 RBF kernels ( $l = 0.6, 1, 2$ ); 3  $l = 1$  Matern kernels ( $\nu = 1/2, 3/2, 5/2$ ) and 3 RBF kernels ( $l = 0.6, 1, 2$ ).

### Ensemble Strategies

Loop over the two methods: erm ensemble and simple averaging respectively.

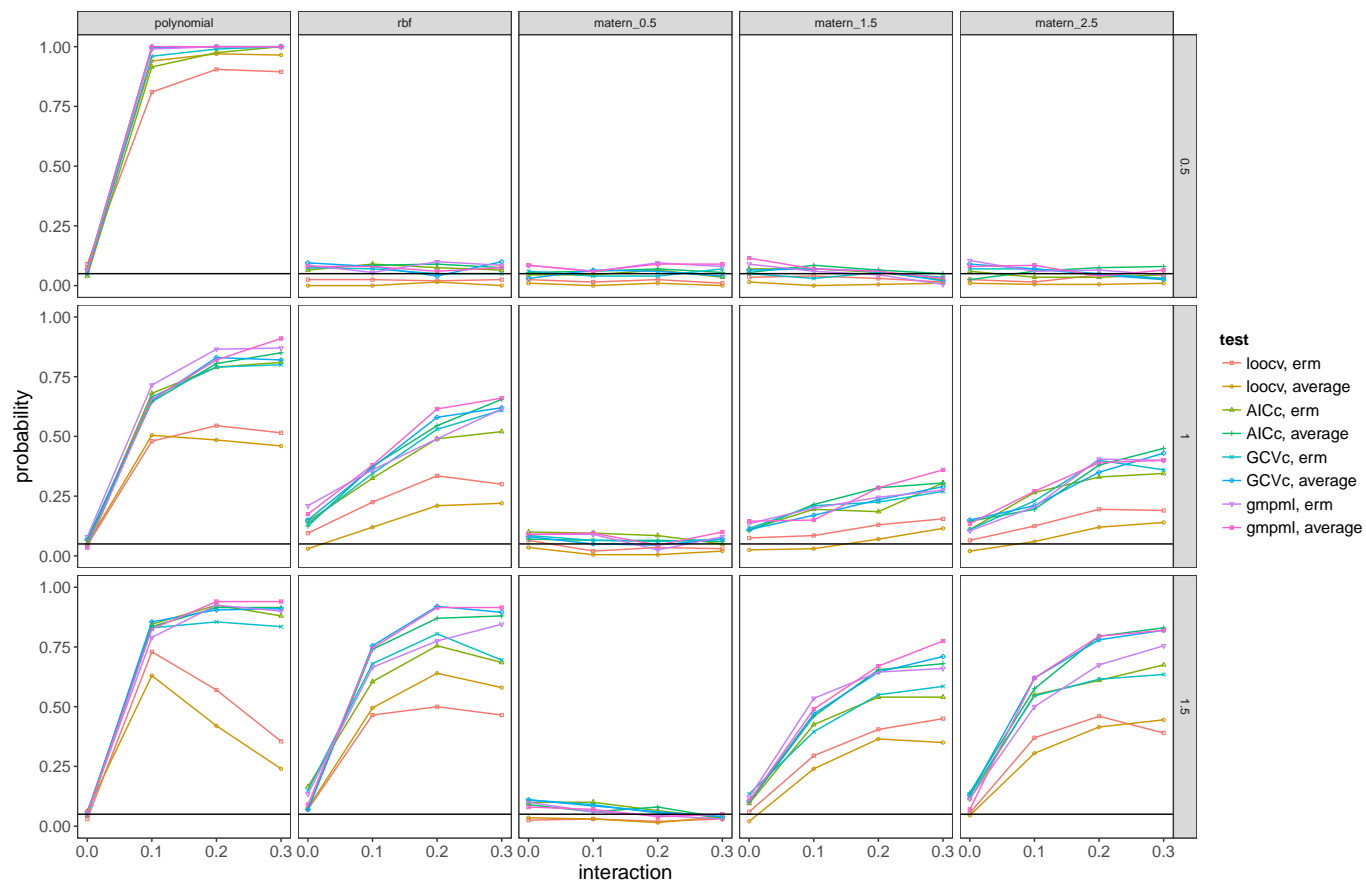


Figure 2: 3 Polynomial kernels

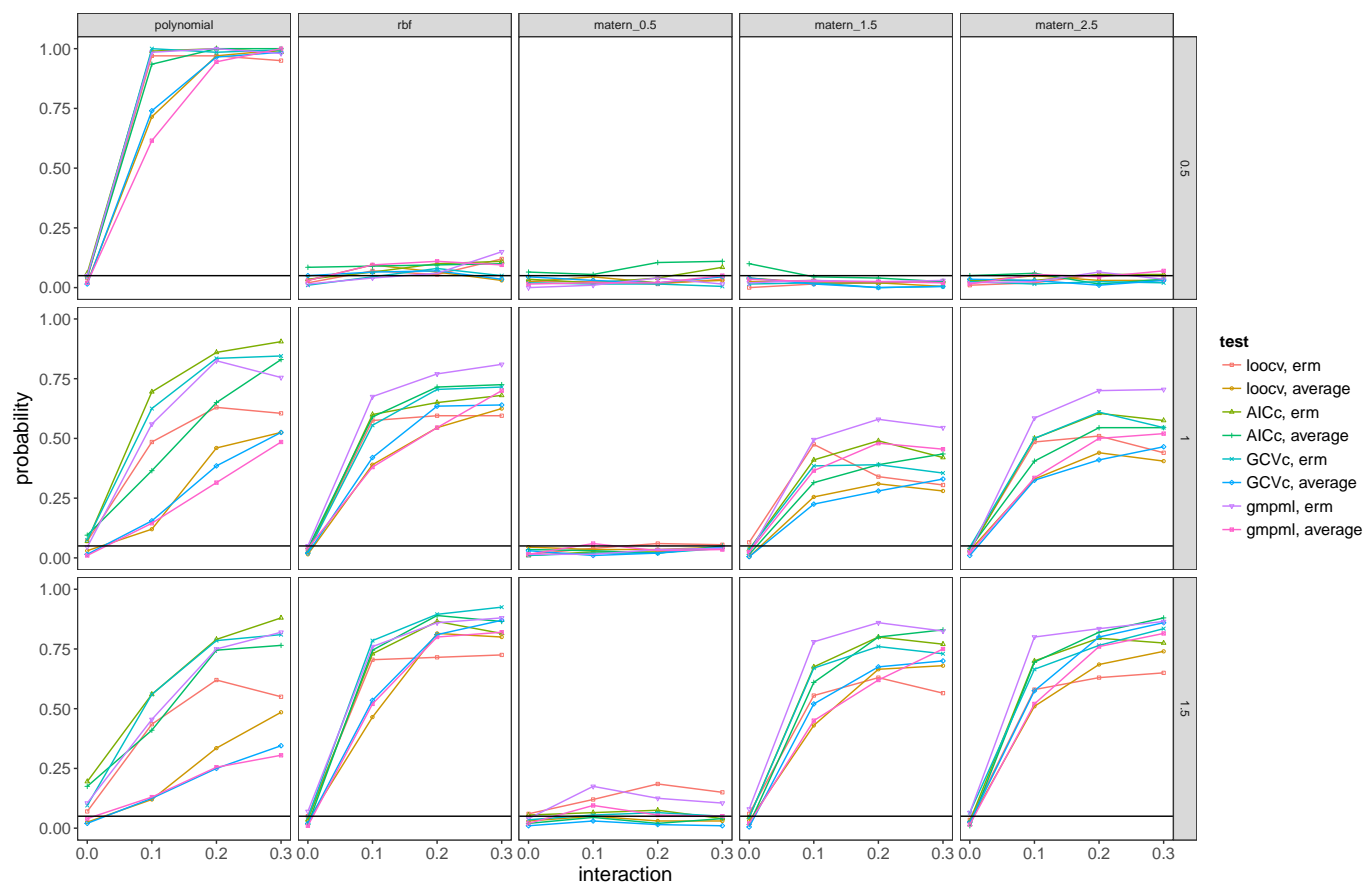


Figure 3: 3 RBF kernels



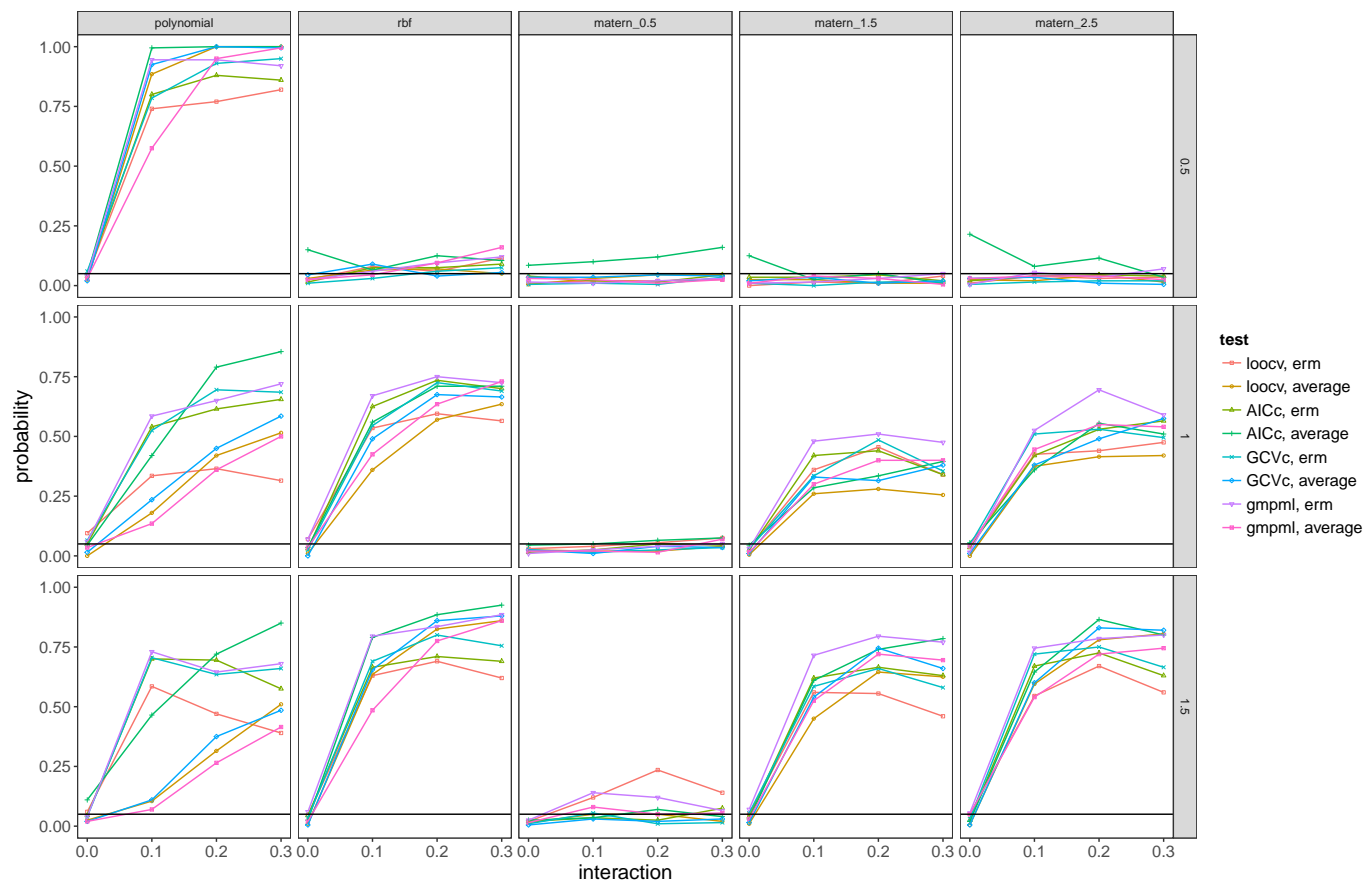


Figure 4: 3 Polynomial kernels and 3 RBF kernels

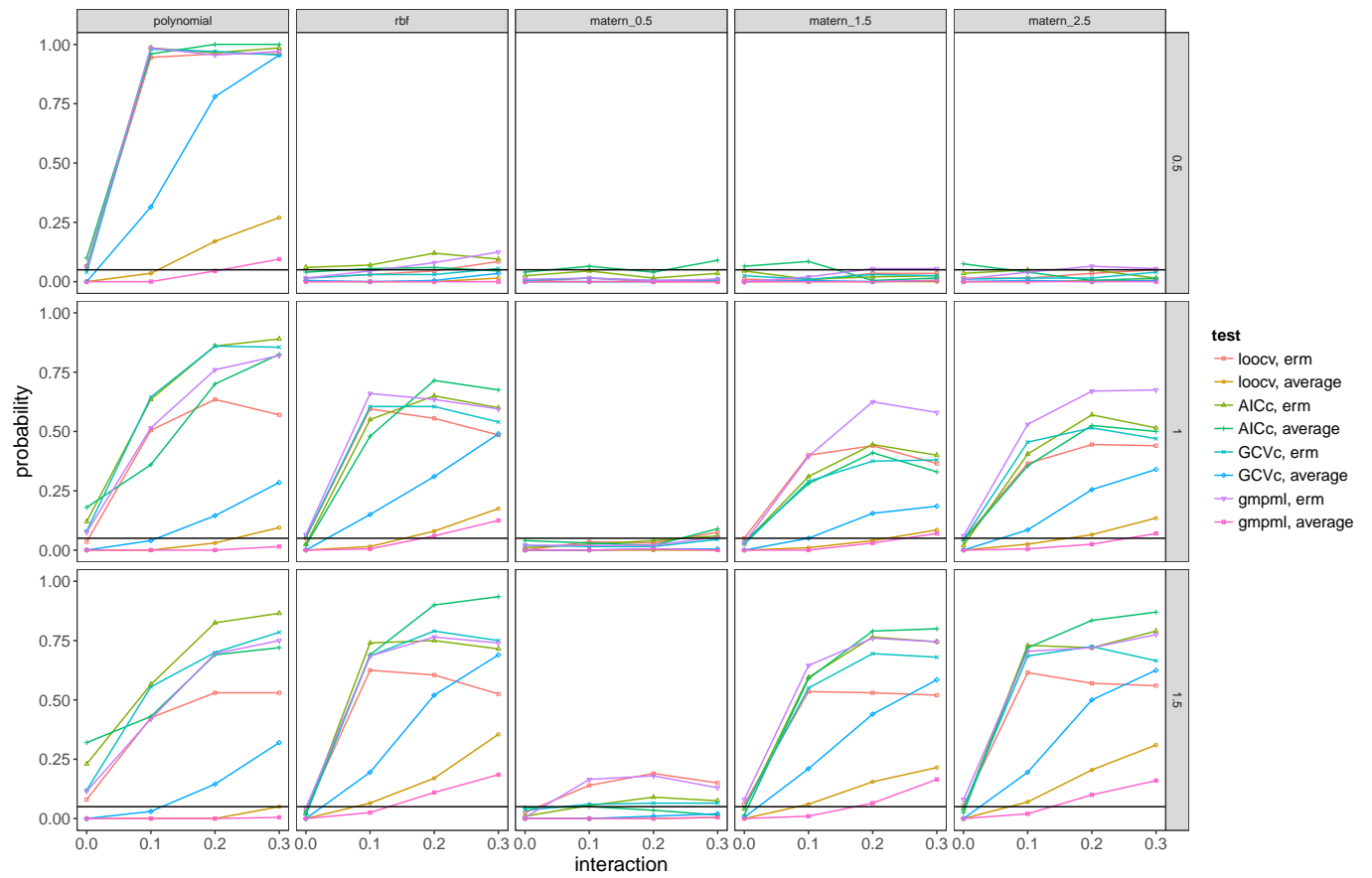


Figure 5: 3 Matern kernels and 3 RBF kernels

## 4 Conclusion

### true kernel only

- 1) If the data-generating function space  $\mathcal{H}$  is too flexible, model estimates will likely **overfit** the data even in the true kernel.
- 2) Empirically, loocv leads to **overfitting**.
- 3) Validated theory:  $\lambda_{\text{AIC}} < \lambda_{\text{GCV}}$ .

### library of polynomials

- 1) Models estimates are capable of achieving satisfying performance (**correct Type I error** and **decent power** compared to true kernel), but only when tuning parameter and ensemble method are chosen carefully.
- 2) loocv leads to **less power** but **better Type I error**.
- 3) In terms of ensemble strategy, average is better than erm.

### summary

- 1) When library includes more flexible kernels, ERM ensemble is **more powerful** than average ensemble.
- 2) When library does not include true kernels, loocv **encourages Type I error**.
- 3) Mix a few simple kernels with reasonably flexible kernels gives good test performance in both linear and nonlinear case.

## References

- Jeremiah Zhe Liu and Brent Coull. Robust Hypothesis Test for Nonlinear Effect with Gaussian Processes. *arXiv:1710.01406 [stat]*, October 2017. URL <http://arxiv.org/abs/1710.01406>. arXiv: 1710.01406.
- 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. ISSN 0040-1706. doi: 10.1080/00401706.1979.10489751. URL <https://www.tandfonline.com/doi/abs/10.1080/00401706.1979.10489751>.
- 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. ISBN 978-0-387-84857-0. URL <http://www.springer.com/fr/book/9780387848570>.
- Hirotougu Akaike. Information Theory and an Extension of the Maximum Likelihood Principle. In *Selected Papers of Hirotougu Akaike*, Springer Series in Statistics, pages 199–213. Springer, New York, NY, 1998. ISBN 978-1-4612-7248-9 978-1-4612-1694-0.
- Clifford M. Hurvich and Chih-Ling Tsai. Regression and time series model selection in small samples. *Biometrika*, 76(2):297–307, June 1989. ISSN 0006-3444. doi: 10.1093/biomet/76.2.297. URL <https://academic.oup.com/biomet/article/76/2/297/265326>.
- 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. ISSN 1369-7412. doi: 10.1111/1467-9868.00125. URL <https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/1467-9868.00125>.
- 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. ISSN 1017-0405. doi: 10.5705/ss.2013.284.
- 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. ISSN 0162-1459. doi: 10.1080/01621459.1983.10477935. URL <https://amstat.tandfonline.com/doi/abs/10.1080/01621459.1983.10477935>.
- 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. ISSN 0162-1459. doi: 10.2307/2286796. URL <http://www.jstor.org/stable/2286796>.
- 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. ISSN 0090-5364, 2168-8966. doi: 10.1214/aos/1176349743. URL <https://projecteuclid.org/euclid.aos/1176349743>.
- 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. ISSN 0006-341X. doi: 10.1111/biom.12684. URL <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5592124/>.

- Michael L. Stein. *Interpolation of Spatial Data: Some Theory for Kriging*. Springer Series in Statistics. Springer-Verlag, New York, 1999. ISBN 978-0-387-98629-6. URL [//www.springer.com/us/book/9780387986296](http://www.springer.com/us/book/9780387986296).
- Milton Abramowitz. *Handbook of Mathematical Functions, With Formulas, Graphs, and Mathematical Tables*. Dover Publications, Inc., New York, NY, USA, 1974. ISBN 978-0-486-61272-0.
- Xihong Lin. Variance component testing in generalised linear models with random effects. *Biometrika*, 84(2):309–326, June 1997. ISSN 0006-3444. doi: 10.1093/biomet/84.2.309. URL <https://academic.oup.com/biomet/article/84/2/309/233889>.
- 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. ISSN 1541-0420. doi: 10.1111/j.1541-0420.2011.01598.x.
- 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. ISSN 1469-1809. doi: 10.1111/j.1469-1809.2010.00572.x.