

Supplementary Materials 1 to “Linear hypothesis testing with functional data”

Łukasz Smaga ^{*1} and Jin-Ting Zhang²

¹Faculty of Mathematics and Computer Science, Adam Mickiewicz University
Umultowska 87, 61-614 Poznań, Poland, ls@amu.edu.pl

²Department of Statistics and Applied Probability, National University of Singapore
3 Science Drive 2, Lower Kent Ridge Road, Singapore, stazjt@nus.edu.sg

Journal: Technometrics

Asymptotic properties of L^2 -norm-based and F -type tests

In this section, we consider the theoretical properties of the L^2 -norm-based and F -type tests proposed in Section 5.3.3 of Zhang (2013). Their test statistics T_n and F_n are given by (3) of the paper.

In Zhang (2013), the null distribution of T_n is approximated by the Welch-Satterthwaite χ^2 -approximation. We have $T_n \sim \hat{\beta} \chi_{q\hat{\kappa}}^2$ approximately, where $\hat{\beta} = \text{tr}(\hat{\gamma}^{\otimes 2}) / \text{tr}(\hat{\gamma})$ and $\hat{\kappa} = \text{tr}^2(\hat{\gamma}) / \text{tr}(\hat{\gamma}^{\otimes 2})$ are the estimators of the naive method, or $\hat{\beta} = \widehat{\text{tr}(\gamma^{\otimes 2})} / \text{tr}(\hat{\gamma})$ and $\hat{\kappa} = \widehat{\text{tr}^2(\gamma)} / \widehat{\text{tr}(\gamma^{\otimes 2})}$ are the estimators of the bias-reduced method, where

$$\begin{aligned}\widehat{\text{tr}^2(\gamma)} &= \frac{(n-k)(n-k+1)}{(n-k-1)(n-k+2)} \left(\text{tr}^2(\hat{\gamma}) - \frac{2\text{tr}(\hat{\gamma}^{\otimes 2})}{n-k+1} \right), \\ \widehat{\text{tr}(\gamma^{\otimes 2})} &= \frac{(n-k)^2}{(n-k-1)(n-k+2)} \left(\text{tr}(\hat{\gamma}^{\otimes 2}) - \frac{\text{tr}^2(\hat{\gamma})}{n-k} \right).\end{aligned}$$

Thus, for each method of estimation of the parameters $\beta = \text{tr}(\gamma^{\otimes 2}) / \text{tr}(\gamma)$ and $\kappa = \text{tr}^2(\gamma) / \text{tr}(\gamma^{\otimes 2})$, we have the L^2 -norm-based test with critical region $\{T_n > T_n(\alpha) = \hat{\beta} \chi_{q\hat{\kappa}}^2(\alpha)\}$. These two tests are referred to as the L²N and L²B tests for naive and bias-reduced method of estimation, respectively.

For approximating the null distribution of F_n , Zhang (2013) used the two-cumulant matched F -approximation method (see Section 4.4 in Zhang, 2013). Namely, $F_n \sim F_{q\hat{\kappa}, (n-k)\hat{\kappa}}$ approximately, where $\hat{\kappa}$ is given above. Hence, we also have two F -type tests which differ in the method of estimation of the parameter κ , referring to the FN and FB

*Corresponding author (ls@amu.edu.pl)

tests. The critical region is of the form $\{F_n > F_n(\alpha) = F_{q\hat{\kappa},(n-k)\hat{\kappa}}(\alpha)\}$, where $F_{d_1,d_2}(\alpha)$ denotes the upper 100α percentile of the F_{d_1,d_2} -distribution.

The above approximations of the null distributions of T_n and F_n are derived under Gaussian or large samples. For other samples, Zhang (2013) proposed the nonparametric bootstrap procedure described in Subsection 2.2 of the paper. The nonparametric bootstrap L^2 -norm-based and F -type tests will be referred to as the L^2b and Fb tests, respectively. Similarly as the GPF and Fmaxb tests, this procedure is valid for the L^2 -norm-based and F -type statistics in the sense of the following result (In Subsection 2.3 of the paper, some remarks about such validity of the bootstrap methods are given.).

Theorem 1. *Let T_n^b (resp. F_n^b) denote the nonparametric bootstrap statistic obtained by computing T_n (resp. F_n) based on the bootstrap samples given by (9) in the paper taking $\mathbf{c} \equiv \mathbf{0}$. Under Assumptions A1-A3 (resp. A1-A5 and A7) in the paper, the unconditional null distribution of T_n (resp. F_n) given in Theorem 5.12 in Zhang (2013) (resp. in (1)) is the same as the conditional distribution of T_n^b (resp. F_n^b), as $n \rightarrow \infty$.*

Proof. First, we establish the asymptotic null distribution of F_n . By Theorem 5.12 in Zhang (2013) and Lemma 1 in Zhang and Liang (2014), under the null hypothesis, we have

$$F_n = \frac{\int_T \text{SSH}_n(t) dt/q}{\int_T \hat{\gamma}(t,t) dt} \xrightarrow{d} \frac{\sum_{r=1}^m \lambda_r A_r}{\text{qtr}(\gamma)}, \quad (1)$$

as $n \rightarrow \infty$, where A_r are the independent χ_q^2 random variables, λ_r are the decreasing-ordered eigenvalues of $\gamma(s,t)$ (If $\lambda_r > 0$ for all r , $m = \infty$). As we noticed in the proof of Theorem 2.2 in the paper, the bootstrap samples $\hat{v}_{ij}^b(t)$, $j = 1, \dots, n_i$, $i = 1, \dots, k$ are i.i.d. $SP(0, \hat{\gamma})$, where $\hat{\gamma}$ is as in (1) in the paper, and thus they satisfy the null hypothesis $H_0 : \mathbf{C}\boldsymbol{\eta}(t) = \mathbf{0}$ for all $t \in T$. Moreover, $\hat{\gamma}(s,t) \xrightarrow{P} \gamma(s,t)$ uniformly for all $(s,t) \in T^2$, as $n \rightarrow \infty$. So, by Theorem 5.12 in Zhang (2013) and (1), we finish the proof. \square

The consistency of the L^2 -norm-based and F -type tests can be considered under two different types of local alternatives. First, we take into account local alternatives considered in Subsection 2.3 of the paper, and obtain the following result for these testing procedures, which is analogous to that given in Theorem 2.3 in the paper for the GPF tests.

Theorem 2. *Let $\mathbf{D}_\tau = \text{diag}(1/\tau_1, \dots, 1/\tau_k)$ and $\delta^2 = \int_T \mathbf{d}(t)^\top (\mathbf{C}\mathbf{D}_\tau \mathbf{C}^\top)^{-1} \mathbf{d}(t) dt$. Under the local alternatives $H_{1n} : \mathbf{C}\boldsymbol{\eta}(t) - \mathbf{c}(t) = n^{-1/2} \mathbf{d}(t)$, $\|d_i\| \in (0, \infty)$, $i = 1, \dots, q$ and Assumptions A1-A5 and A7 in the paper, as $n \rightarrow \infty$, the asymptotic power of the L^2N , L^2B , FN and FB tests as well as the power of the nonparametric bootstrap L^2 -norm-based and F -type tests tend to 1 as $\delta \rightarrow \infty$.*

Proof. In the proof of Theorem 2.3 in the paper, we established that under H_{1n} , $\mathbf{z}_n(t) \xrightarrow{d} \mathbf{z}_{alt}(t) \sim GP_q(\boldsymbol{\mu}_{alt}, \gamma \mathbf{I}_q)$, as $n \rightarrow \infty$, where $\boldsymbol{\mu}_{alt}(t) = (\mathbf{C}\mathbf{D}_\tau \mathbf{C}^\top)^{-1/2} \mathbf{d}(t)$. Hence, under H_{1n}

$$T_n \xrightarrow{d} T_{alt} = \int_T \mathbf{z}_{alt}(t)^\top \mathbf{z}_{alt}(t) dt, \quad F_n \xrightarrow{d} F_{alt} = \frac{T_{alt}}{\text{qtr}(\gamma)}.$$

From Theorem 4.10 in Zhang (2013), it follows that $T_{alt} \stackrel{d}{=} \sum_{r=1}^m \lambda_r A_r^{alt} + \sum_{r=m+1}^\infty \delta_r^2$ and $F_{alt} \stackrel{d}{=} (\sum_{r=1}^m \lambda_r A_r^{alt} + \sum_{r=m+1}^\infty \delta_r^2) / [\text{qtr}(\gamma)]$, where A_r^{alt} are independent $\chi_q^2(\delta_r^2/\lambda_r)$ random variables, $\delta_r^2 = \|\int_T \boldsymbol{\mu}_{alt}(t) \varphi_r(t) dt\|^2$, λ_r are the decreasing-ordered eigenvalues

of $\gamma(s, t)$ with only the first m eigenvalues being positive, and $\varphi_r(t)$ are the associated eigenfunctions, $r = 1, 2, \dots$ (When $\lambda_r > 0$ for all r , we take $m = \infty$). The rest of the proof for L^2 -norm-based (resp. F -type) tests runs in an analogous way as in the proof of Theorem 2.3 in the paper by using Theorems 5.10 and 5.12 in Zhang (2013) (resp. (1)) and Theorem 1 instead of Proposition 2 in Zhang and Liang (2014) and Theorems 2.1 and 2.2 of the paper, respectively. \square

Now, the power of the L^2 -norm-based and F -type tests is considered under the local alternatives $H_{1n} : \mathbf{C}\boldsymbol{\eta}(t) - \mathbf{c}(t) = n^{-\omega/2}\mathbf{d}(t)$, where $\omega \in [0, 1)$ and $\mathbf{d}(t) = (d_1(t), \dots, d_q(t))^\top$ is any fixed vector of real functions, independent of n , and $\|d_i\| \in (0, \infty)$, $i = 1, \dots, q$ (see, for example, Zhang, 2011). In contrast to local alternatives considered in Subsection 2.3 of the paper, these alternatives tend to the null hypothesis at a rate slightly slower than root- n . Under the Gaussianity assumption, we show that the L^2 -norm-based and F -type testing procedures can detect departures from the null hypothesis determined by these alternatives with probability tending to one as $n \rightarrow \infty$.

Theorem 3. *Let $x_{i1}(t), \dots, x_{in_i}(t)$ defined over T be i.i.d. $GP(\eta_i, \gamma)$, $i = 1, \dots, k$. Under the local alternatives $H_{1n} : \mathbf{C}\boldsymbol{\eta}(t) - \mathbf{c}(t) = n^{-\omega/2}\mathbf{d}(t)$, $\omega \in [0, 1)$, $\|d_i\| \in (0, \infty)$, $i = 1, \dots, q$ and Assumptions A1–A5 and A7 in the paper, the asymptotic power of the L^2N , L^2B , FN and FB tests as well as the power of the nonparametric bootstrap L^2 -norm-based and F -type tests tend to 1 as $n \rightarrow \infty$.*

Proof. Under the Gaussianity of the k samples, we have $\mathbf{z}_n(t) \sim GP_q(\boldsymbol{\eta}_z, \gamma\mathbf{I}_q)$, where $\boldsymbol{\eta}_z(t) = (\mathbf{C}\mathbf{D}\mathbf{C}^\top)^{-1/2}[\mathbf{C}\boldsymbol{\eta}(t) - \mathbf{c}(t)]$ (see Subsection 2.2 of the paper). First, we consider the L^2 -norm-based tests. Theorem 4.10 in Zhang (2013) implies $T_n \stackrel{d}{=} \sum_{r=1}^m \lambda_r A_r^{alt} + \sum_{r=m+1}^\infty \delta_r^2$, where $\delta_r^2 = \|\int_T \boldsymbol{\eta}_z(t) \varphi_r(t) dt\|^2$ and A_r^{alt} , λ_r , φ_r and m the same as in the proof of Theorem 2, $r = 1, 2, \dots$. Under H_{1n} , we have

$$\begin{aligned} \delta_r^2 &= \left\| \int_T (\mathbf{C}\mathbf{D}\mathbf{C}^\top)^{-1/2} n^{-\omega/2} \mathbf{d}(t) \varphi_r(t) dt \right\|^2 \\ &= \left\| \int_T (\mathbf{C}n\mathbf{D}\mathbf{C}^\top)^{-1/2} n^{(1-\omega)/2} \mathbf{d}(t) \varphi_r(t) dt \right\|^2 \\ &= n^{1-\omega} \left(\left\| \int_T (\mathbf{C}\mathbf{D}_\tau \mathbf{C}^\top)^{-1/2} \mathbf{d}(t) \varphi_r(t) dt \right\|^2 (1 + o(1)) \right), \end{aligned}$$

and hence $T_n \stackrel{d}{=} \sum_{r=1}^m \lambda_r A_r^{alt} + n^{1-\omega} \sum_{r=m+1}^\infty \delta_{*r}^2$, where $A_r^{alt} \sim \chi_q^2(n^{1-\omega} \delta_{*r}^2 / \lambda_r)$ and $\delta_{*r}^2 = \|\int_T (\mathbf{C}\mathbf{D}_\tau \mathbf{C}^\top)^{-1/2} \mathbf{d}(t) \varphi_r(t) dt\|^2 (1 + o(1))$. Using Theorems 5.10 and 5.12 in Zhang (2013) and Theorem 1, the rest of the proof runs in analogous way as in Section 2.4 of Zhang et al. (2010) by taking “ $\|u\| = \delta_*$ ”, where

$$\delta_*^2 = \sum_{r=1}^\infty \delta_{*r}^2 = \int_T \mathbf{d}(t)^\top (\mathbf{C}\mathbf{D}_\tau \mathbf{C}^\top)^{-1} \mathbf{d}(t) dt > 0.$$

Now, we take into account the F -type testing procedures. By the above considerations and Theorem 5.11 in Zhang (2013), we have

$$F_n \stackrel{d}{=} \frac{\sum_{r=1}^m \lambda_r A_r^{alt} + n^{1-\omega} \sum_{r=m+1}^\infty \delta_{*r}^2}{q \sum_{r=1}^m \lambda_r E_r / (n - k)} \stackrel{d}{=} \frac{\sum_{r=1}^m \lambda_r A_r^{alt} + n^{1-\omega} \sum_{r=m+1}^\infty \delta_{*r}^2}{q \text{tr}(\gamma)} + o_p(1), \quad (2)$$

as $n \rightarrow \infty$, where E_r are independent χ_{n-k}^2 random variables. By (2), Theorem 5.10 in Zhang (2013) and Theorem 1, we can continue the proof for F -type tests using similar arguments and techniques as in Section 2.2 of Zhang (2011). \square

Implementation of the tests in the R program

In this section, we present the R function `tests.linear.hypotheses()`, which calculates the values of test statistics and p -values of the tests under consideration.

As in the paper, assume that we consider k groups of functions and q linear hypotheses. The arguments of the function `tests.linear.hypotheses()` are described as follows:

- **x** - a $n \times J$ data frame or matrix of data, whose each row is a discretized version of a function in J design time points (see Subsection 2.4 of the paper for more details),
- **group.label** - a vector containing group labels. The labels of groups in **group.label** must be the numbers from 1 to the number of groups (first group – label 1, second group – label 2, ..., last group – label k).
- **CC** - a $q \times k$ coefficient matrix **C** of full row rank. Remember to specify it correctly taking into account the group labels in **group.label**.
- **cc** - a vector or matrix representing a known $q \times 1$ vector $\mathbf{c}(t)$ of fixed functions in the hypotheses. When $\mathbf{c} = \mathbf{0}$, we can simply use **cc** = 0. In case of constant functions in $\mathbf{c}(t)$ with at least one non-zero, **cc** is a vector of length q containing numbers representing those functions. When at least one function in $\mathbf{c}(t)$ is not constant, **cc** is a $q \times J$ matrix, whose each row is a discretized version of a function in $\mathbf{c}(t)$ in J design time points, in which the observations in **x** are given.
- **nboot** - a number of bootstrap samples,
- **seed** - a seed given by `set.seed(seed)`,
- **parallel** - a logical indicating whether to use parallelization of execution of bootstrap loop,
- **ncores** - if **parallel** = TRUE, a number of logical processes used for parallel computing. Its default value means that it is chosen automatically based on the computer used.

To reduce the computational cost of bootstrap loop, its execution may be parallelized using process forking, based on package `doParallel` (Revolution Analytics and Steve Weston, 2015). To do this, we choose **parallel** = TRUE.

The function `tests.linear.hypotheses()` returns a data frame whose first row contains the values of test statistics and the second one – p -values. The columns of outputted data frame correspond to the L^2N , L^2B , L^2b , FN, FB, Fb, GPF, GPFb and Fmaxb tests, respectively.

```
tests.linear.hypotheses = function(x, group.label, CC, cc, nboot = 10000, seed = 1234,
                                   parallel = FALSE, ncores = NULL){
  if(nrow(x) != length(group.label)){
    stop("number of observations (number of rows in x) and number of elements
         in vector of group labels (group.label) must be the same")
  }
  if(any(is.na(group.label))){ stop("argument group.label can not contain NA values") }
  if(length(unique(group.label)) != ncol(CC)){
    stop("number of group labels in group.label and number of columns in
```

```

    coefficient matrix C must be equal")
}
if(any(sort(unique(group.label)) != 1:length(unique(group.label)))){
  stop("group labels must be the numbers from 1 to the number of groups")
}
if(qr(CC)$rank != nrow(CC)){ stop("coefficient matrix C must be of full row rank") }
if(! is.logical(parallel)){ stop("argument parallel is not logical") }
if(nboot < 1){ stop("invalid number of bootstrap samples (nboot)") }
if(is.null(seed)){ warning("argument seed equals NULL") }

n = nrow(x); p = ncol(x); x = as.matrix(x); qq = nrow(CC)
k = max(group.label); n.i = numeric(k)
for(i in 1:k) n.i[i] = sum(group.label==i)
vmu = matrix(0, nrow = k, ncol = p)
z = matrix(0, nrow = n, ncol = p)
for(i in 1:k){
  xi = x[group.label==i,]
  mui = colMeans(xi); vmu[i,] = mui
  zi = xi - as.matrix(rep(1, n.i[i])) %*% mui
  if(i==1){ z[1:n.i[i],] = zi }else{ z[(cumsum(n.i)[i-1]+1):cumsum(n.i)[i],] = zi }
}
if(n > p){ gamma.z = t(z) %*% z/(n-k) }else{ gamma.z = z %*% t(z)/(n-k) }
A = sum(diag(gamma.z)); B = sum(diag(gamma.z %*% gamma.z))
A2N = A^2; B2N = B
A2B = (n-k)*(n-k+1)/(n-k-1)/(n-k+2)*(A^2-2*B/(n-k+1))
B2B = (n-k)^2/(n-k-1)/(n-k+2)*(B-A^2/(n-k))
SSE = diag(t(z) %*% z)
weight.matrix = solve(CC %*% diag(1/n.i) %*% t(CC))
SSH = diag(t(CC) %*% vmu - cc) %*% weight.matrix %*% (CC %*% vmu - cc)

statL2 = sum(SSH)
betaL2N = B2N/A; kappaL2N = A2N/B2N
betaL2B = B2B/A; kappaL2B = A2B/B2B
pvalueL2N = 1-pchisq(statL2/betaL2N, qq*kappaL2N)
pvalueL2B = 1-pchisq(statL2/betaL2B, qq*kappaL2B)

statF = sum(SSH)/sum(SSE)*(n-k)/qq
kappaFN = A2N/B2N; kappaFB = A2B/B2B
pvalueFN = 1-pf(statF, qq*kappaFN, (n-k)*kappaFN)
pvalueFB = 1-pf(statF, qq*kappaFB, (n-k)*kappaFB)

statGPF = mean(SSH/SSE*(n-k)/qq)
z = z/(as.matrix(rep(1, n)) %*% sqrt(colSums(z^2)))
if(n >= p){ z = t(z) %*% z }else{ z = z %*% t(z) }
betaGPF = (sum(diag(z %*% z))/p^2/qq)/((n-k)/(n-k-2))
dGPF = ((n-k)/(n-k-2))^2/(sum(diag(z %*% z))/p^2/qq)
pvalueGPF = 1-pchisq(statGPF/betaGPF, dGPF)

Fmax = max(SSH/SSE*(n-k)/qq)

set.seed(seed)
bootstrap.samples = matrix(0, nrow = nboot, ncol = n)
for(ii in 1:nboot){ bootstrap.samples[ii,] = sample(1:n, replace = T) }
xs = matrix(0, nrow = n, ncol = p)
for(i in 1:k){ xs[group.label==i,] = x[group.label==i,] -

```

```

as.matrix(rep(1, n.i[i])) %*% vmu[i,] }

if(parallel == FALSE){
  L2boot = numeric(nboot); Fboot = numeric(nboot)
  GPFboot = numeric(nboot); Fmaxboot = numeric(nboot)
  for(ii in 1:nboot){
    vmuboot = matrix(0, nrow = k, ncol = p)
    zboot = matrix(0, nrow = n, ncol = p)
    xboot = xs[bootstrap.samples[ii,],]
    for(i in 1:k){
      xiboot = xboot[group.label==i,]
      muiboot = colMeans(xiboot)
      vmuboot[i,] = muiboot
      ziboot = xiboot - as.matrix(rep(1, n.i[i])) %*% muiboot
      if(i==1){ zboot[1:n.i[i],] = ziboot }else{
        zboot[(cumsum(n.i)[i-1]+1):cumsum(n.i)[i],] = ziboot }
    }
    SSHboot = diag(t(CC %*% vmuboot) %*% weight.matrix %*% (CC %*% vmuboot))
    SSEboot = diag(t(zboot) %*% zboot)
    L2boot[ii] = sum(SSHboot)
    Fboot[ii] = sum(SSHboot)/sum(SSEboot)*(n-k)/qq
    GPFboot[ii] = mean(SSHboot/SSEboot*(n-k)/qq)
    Fmaxboot[ii] = max(SSHboot/SSEboot*(n-k)/qq)
  }
}else{
  nlp = detectCores()
  if(is.null(ncores)){
    if(nlp >= 2){
      ncores = nlp; parallel.method = "parallel.method1"
    }else{
      parallel.method = "parallel.method0"
    }
  }else{
    if(nlp >= 2){
      if(ncores >= 2){
        ncores = ncores
      }else{
        ncores = nlp
      }
      parallel.method = "parallel.method1"
    }else{
      parallel.method = "parallel.method0"
    }
  }
}
if(parallel.method != "parallel.method0"){
  cl = makePSOCKcluster(ncores)
  registerDoParallel(cl)
}
if(parallel.method == "parallel.method0"){
  L2boot = numeric(nboot); Fboot = numeric(nboot)
  GPFboot = numeric(nboot); Fmaxboot = numeric(nboot)
  for(ii in 1:nboot){
    vmuboot = matrix(0, nrow = k, ncol = p)
    zboot = matrix(0, nrow = n, ncol = p)
    xboot = xs[bootstrap.samples[ii,],]

```

```

    for(i in 1:k){
      xiboot = xboot[group.label==i,]
      muiboot = colMeans(xiboot)
      vmuboot[i,] = muiboot
      ziboot = xiboot - as.matrix(rep(1, n.i[i])) %*% muiboot
      if(i==1){ zboot[1:n.i[i],] = ziboot }else{
        zboot[(cumsum(n.i)[i-1]+1):cumsum(n.i)[i],] = ziboot }
    }
    SSHboot = diag(t(CC %*% vmuboot) %*% weight.matrix %*% (CC %*% vmuboot))
    SSEboot = diag(t(zboot) %*% zboot)
    L2boot[ii] = sum(SSHboot)
    Fboot[ii] = sum(SSHboot)/sum(SSEboot)*(n-k)/qq
    GPFboot[ii] = mean(SSHboot/SSEboot*(n-k)/qq)
    Fmaxboot[ii] = max(SSHboot/SSEboot*(n-k)/qq)
  }
}
if(parallel.method == "parallel.method1"){
  rs = foreach(ii = 1:nboot, .combine = rbind) %dopar%
  {
    vmuboot = matrix(0, nrow = k, ncol = p)
    zboot = matrix(0, nrow = n, ncol = p)
    xboot = xs[bootstrap.samples[ii,],]
    for(i in 1:k){
      xiboot = xboot[group.label==i,]
      muiboot = colMeans(xiboot)
      vmuboot[i,] = muiboot
      ziboot = xiboot - as.matrix(rep(1, n.i[i])) %*% muiboot
      if(i==1){ zboot[1:n.i[i],] = ziboot }else{
        zboot[(cumsum(n.i)[i-1]+1):cumsum(n.i)[i],] = ziboot }
    }
    SSHboot = diag(t(CC %*% vmuboot) %*% weight.matrix %*% (CC %*% vmuboot))
    SSEboot = diag(t(zboot) %*% zboot)
    c(sum(SSHboot), sum(SSHboot)/sum(SSEboot)*(n-k)/qq,
      mean(SSHboot/SSEboot*(n-k)/qq), max(SSHboot/SSEboot*(n-k)/qq))
  }
  L2boot = rs[, 1]; Fboot = rs[, 2]; GPFboot = rs[, 3]; Fmaxboot = rs[, 4]
}
if(parallel.method != "parallel.method0"){
  stopCluster(cl)
}
}
result = data.frame(L2N = c(statL2, pvalueL2N), L2B = c(statL2, pvalueL2B),
  L2b = c(statL2, mean(L2boot >= statL2)),
  FN = c(statF, pvalueFN), FB = c(statF, pvalueFB),
  Fb = c(statF, mean(Fboot >= statF)),
  GPF = c(statGPF, pvalueGPF),
  GPFb = c(statGPF, mean(GPFboot >= statGPF)),
  Fmaxb = c(Fmax, mean(Fmaxboot >= Fmax)))
rownames(result) = c("test statistic", "p-value")
return(result)
}

```

To illustrate the use of the function `tests.linear.hypotheses()`, we present the following code, which performs the real data example given in Section 4 of the paper. First, we read the corneal surface data from the file `CornealDataSet.txt`. Next, we remove the

outliers in the unilateral suspect cornea group and define the vector of group labels. Finally, for given coefficient matrices, we apply the function `tests.linear.hypotheses()` to the data. The resulting p -values are given in Table 1 in the paper (rounded to five decimal places). Since the bootstrap tests applied to this data set are time-consuming, we use parallelization.

```
data1 <- read.table("CornealDataSet.txt")
dim(data1)
## 150 2000
data2 <- data1[-c(49, 54),]
dim(data2)
## 148 2000
group.label <- rep(1:4, c(43, 12, 21, 72))
CC1 <- cbind(diag(rep(1, 3)), matrix(-1, nrow = 3, ncol = 1))
CC2 <- matrix(c(1, -1, 0, 0), nrow = 1)
CC3 <- matrix(c(1, 0, -1, 0), nrow = 1)
CC4 <- matrix(c(1, 0, 0, -1), nrow = 1)
CC5 <- matrix(c(0, 1, -1, 0), nrow = 1)
CC6 <- matrix(c(0, 1, 0, -1), nrow = 1)
CC7 <- matrix(c(0, 0, 1, -1), nrow = 1)
library(doParallel)
tests.linear.hypotheses(x = data2, group.label = group.label, CC = CC1, cc = 0,
                        nboot = 10000, seed = 1234, parallel = TRUE)
tests.linear.hypotheses(x = data2, group.label = group.label, CC = CC2, cc = 0,
                        nboot = 10000, seed = 1234, parallel = TRUE)
tests.linear.hypotheses(x = data2, group.label = group.label, CC = CC3, cc = 0,
                        nboot = 10000, seed = 1234, parallel = TRUE)
tests.linear.hypotheses(x = data2, group.label = group.label, CC = CC4, cc = 0,
                        nboot = 10000, seed = 1234, parallel = TRUE)
tests.linear.hypotheses(x = data2, group.label = group.label, CC = CC5, cc = 0,
                        nboot = 10000, seed = 1234, parallel = TRUE)
tests.linear.hypotheses(x = data2, group.label = group.label, CC = CC6, cc = 0,
                        nboot = 10000, seed = 1234, parallel = TRUE)
tests.linear.hypotheses(x = data2, group.label = group.label, CC = CC7, cc = 0,
                        nboot = 10000, seed = 1234, parallel = TRUE)
```

References

- [1] Revolution Analytics and Steve Weston (2015). doParallel: Foreach parallel adaptor for the 'parallel' package. R package version 1.0.10. <http://CRAN.R-project.org/package=doParallel>
- [2] Zhang, J.-T. (2011). Statistical inferences for linear models with functional responses. *Statistica Sinica*, **21**, 1431–1451.
- [3] Zhang, J.-T. (2013). *Analysis of variance for functional data*. Chapman & Hall, London.
- [4] Zhang, J.-T. and Liang, X. (2014). One-way ANOVA for functional data via globalizing the pointwise F -test. *Scandinavian Journal of Statistics*, **41**, 51–71.
- [5] Zhang, J.-T., Liang, X., and Xiao, S. (2010). On the two-sample Behrens-Fisher problem for functional data. *Journal of Statistical Theory and Practice*, **4**, 571–587.