

Methods and Algorithms to Test the Simplex and Hausdorff Dispersion Orders and an R Package: software

G. Ayala¹, M.C. López-Díaz², M. López-Díaz³ *, L. Martínez-Costa⁴

¹Universidad de Valencia. Dpto. de Estadística e Investigación Operativa. Av. Vicent Andrés Estellés, 1. E-46100 Burjasot, Valencia. Spain. Guillermo.Ayala@uv.es

²Universidad de Oviedo. Dpto. de Matemáticas. C/ Calvo Sotelo, s/n. E-33007 Oviedo. Spain. cld@uniovi.es

³Universidad de Oviedo. Dpto. de Estadística e I.O. y D.M. C/ Calvo Sotelo, s/n. E-33007 Oviedo. Spain. mld@uniovi.es

⁴Hospital Dr. Peset, Servicio de Oftalmología, Av. Gaspar Aguilar 90, E-46017 Valencia, Spain. martinez_luc@gva.es

Abstract

This document contains the software needed to reproduce the results given in the paper Ayala et al. [2012]. We will refer to the paper. The basic package needed is Ayala [2012].

1 The introduction

First, we load the packages needed later Ayala [2012], Barber et al. [2012], Genz et al. [2011] and Jr [2012].

```
> library(disp2D)
> library(geometry)
> library(mvtnorm)
> library(Hmisc)
```

We use some additional functions non included in the package because they are mainly useful in the applications. Load these additional functions.

*corresponding author E-mail: mld@uniovi.es Phone: (34) 985 10 33 62 Fax: (34) 985 10 33 54

```

> #####
> ## Different tests of the usual dispersion orderings
> ## Wilcoxon and Kolmogorov-Smirnov tests
> #####
> uso.test = function(dx,dy,whichPlot=0){
+
+   if(whichPlot == 1){
+     plot(density(dx,bw=.8))
+     lines(density(dy,bw=.8),lty=2)
+   }
+   if(whichPlot == 2){
+     boxplot(dx,dy)
+   }
+
+   wil = wilcox.test(dx,dy,alternative="less")
+   p.wil = wil$p.value
+   stat.wil = wil$statistic
+
+   ks = ks.test(dx,dy,alternative="less")
+   stat.ks = ks$statistic
+   p.ks = ks$p.value
+
+   result = list(p.wil = p.wil,stat.wil = stat.wil,p.ks = p.ks,stat.ks=stat.ks)
+   result
+ }
> #####
> ### testSD0 = test for simplex dispersion ordering
> ### x = data set
> ### y = experimental factor
> #####
>
> testSD0 = function(y,x,bootstrap=TRUE,nresamples=10,dib=FALSE,nameplot ="testSD0.eps"){
+   ## y is the nx2 matrix with the original data
+   ## x is the categorical covariable
+   ny = nrow(y)
+   x = factor(x)
+   levels.x = levels(x)
+   z1 = NULL
+   z2 = NULL
+   for(i in levels.x){
+     y1 = y[x == i,]
+     y1 = na.omit(y1)
+     z0 = simplex(y1,bootstrap,nresamples)
+     z1 = c(z1,z0)
+     z2 = c(z2,rep(i,length(z0)))
+   }
+ }

```

```

+
+   if(dib){
+     postscript(nameplot,horizontal=F)
+     Ecdf(z1,group=z2,lty=1:nlevels(factor(z2)))
+     dev.off()
+   }
+
+   pvaluesKS = matrix(0,nrow=nlevels(x),ncol=nlevels(x))
+   for(i in 1:nlevels(x)){
+     for(j in 1:nlevels(x)){
+       pvaluesKS[i,j] = ks.test(z1[z2 == levels.x[i]],
+                                z1[z2 == levels.x[j]],alternative="greater")$p.value
+     }
+   }
+   pvaluesKS
+
+ }
+
> #####
> ### testHDO = test for Hausdorff dispersion ordering
> ### x = data set
> ### y = experimental factor
> #####
>
> testHDO = function(y,x,r,dib=FALSE,nameplot="testHDO.eps"){
+   ## y is the nx2 matrix with the original data
+   ## x is the categorical covariable
+   ny = nrow(y)
+   x = factor(x)
+   levels.x = levels(x)
+   z1 = NULL
+   z2 = NULL
+   for(i in levels.x){
+     y1 = y[x == i,]
+     y1 = na.omit(y1)
+     n1 = nrow(y1)
+     prob1 = rep(1/n1,n1)
+     z0 = exactHausdorff(y1, prob1, r)
+     z1 = c(z1,z0$alldistances)
+     z2 = c(z2,rep(i,length(z0$alldistances)))
+   }
+
+   if(dib){
+     postscript(nameplot,horizontal=F)
+     Ecdf(z1,group=z2,lty=1:nlevels(factor(z2)))
+     dev.off()
+   }
+ }

```

```

+
+   pvaluesKS = matrix(0,nrow=nlevels(x),ncol=nlevels(x))
+   for(i in 1:nlevels(x)){
+     for(j in 1:nlevels(x)){
+       pvaluesKS[i,j] = ks.test(z1[z2 == levels.x[i]],
+                                z1[z2 == levels.x[j]],alternative="greater")$p.value
+     }
+   }
+ pvaluesKS
+ }
>

```

2 A simulation study

This section is devoted to a simulation study of the methods developed to test the Hausdorff and simplex dispersion orderings.

The model used in this simulation study is as follows. Let us consider the \mathbb{R}^2 -valued random vectors \mathbf{X} and \mathbf{Y} with normal distributions, $\mathbf{X} \sim_{st} N(\mu_{\mathbf{X}}, \Sigma_{\mathbf{X}})$ and $\mathbf{Y} \sim_{st} N(\mu_{\mathbf{Y}}, \Sigma_{\mathbf{Y}})$, where $\Sigma_{\mathbf{X}} = AA^T$, A being a 2×2 real matrix whose values are randomly chosen with uniform distribution in the interval $(0, 1)$, the superindex T denoting the transpose matrix, and $\Sigma_{\mathbf{Y}} = \Sigma_{\mathbf{X}} + \lambda I_2$, with $\lambda \geq 0$ and I_2 being the 2×2 identity matrix. It is well-known that the eigenvalues of $\Sigma_{\mathbf{Y}}$ are those of $\Sigma_{\mathbf{X}}$ plus the value λ .

In the rest of the manuscript we will talk about p -values obtained under different conditions. These p -values are all related to the test $H_0 : \mathbf{X} \preceq_{sx} \mathbf{Y}$ versus $H_1 : H_0$ is false, or to $H_0 : \mathbf{X} \preceq_H^r \mathbf{Y}$ versus $H_1 : H_0$ is false.

Roughly speaking, larger values of λ will produce larger dispersion for the random vector \mathbf{Y} .

In the rest of the manuscript we will talk about p -values obtained under different conditions. These p -values are all related to the test $H_0 : \mathbf{X} \preceq_{sx} \mathbf{Y}$ versus $H_1 : H_0$ is false, or to $H_0 : \mathbf{X} \preceq_H^r \mathbf{Y}$ versus $H_1 : H_0$ is false.

Roughly speaking, larger values of λ will produce larger dispersion for the random vector \mathbf{Y} .

Let us generate two point sets from the model just considered. The multivariate normal data are generated using the package *mvtnorm* Genz et al. [2011].

```

> d = 2
> n = 100
> mu1 = rep(0,d)
> mu2 = mu1
> lambda = .5
> n1=n2=n= 100
> sigma1 = matrix(runif(d*d),nrow=d,ncol=d)
> sigma1 = t(sigma1) %*% sigma1

```

```

> sigma2 = sigma1 + lambda * diag(1,d)
> A = rmvnorm(n1,mean=mu1,sigma=sigma1)
> B = rmvnorm(n2,mean=mu2,sigma=sigma2)

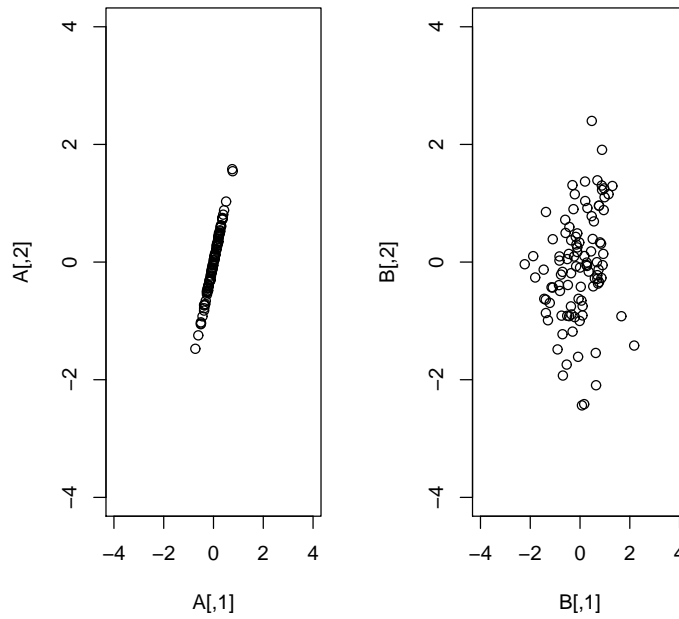
```

Let us see the datasets

```

> par(mfrow=c(1,2))
> plot(A,xlim=c(-4,4),ylim=c(-4,4))
> plot(B,xlim=c(-4,4),ylim=c(-4,4))
> par(mfrow=c(1,1))

```



In particular, in our study λ ranges from 0 to 1 and samples from \mathbf{X} and \mathbf{Y} will have the same size equal to 100. We have simulated 100 sample replications for each value of λ and for each generated data set we have tested the simplex and Hausdorff dispersion ordering. Therefore we have obtained 100 p -values corresponding to the sample replications for each value of λ .

This is the code.

```

> numreplicas = 100
> n1 = n2 = n = 100
> lambdavalues = seq(0,1,length.out=14)
> nresamplesvalues = seq(100,300,50)
> rvalues = seq(.01,.5,length.out=20)

```

The following code (is given within a verbatim environment because it needs a lot of computation) calculates the p -values for different λ -values and r -values (for the Hausdorff dispersion ordering). The results are saved at the files `resultH.RData` and `resultS.RData`.

```

resultH = NULL
resultS = NULL
for(lambda in lambdavalues){
  cat("lambda ",lambda,"\n")
  pH = NULL
  pS = NULL
  for(i in 1:numreplicas){
    cat(date(),"Replication ",i,"\n")
    date()
    sigma1 = matrix(runif(d*d),nrow=d,ncol=d)
    sigma1 = t(sigma1) %*% sigma1
    sigma2 = sigma1 + lambda * diag(1,d)
    A = rmvnorm(n1,mean=mu1,sigma=sigma1)
    B = rmvnorm(n2,mean=mu1,sigma=sigma2)
    ## For Hausdorff
    pHtemp = NULL
    for(r in rvalues){
      u1 = exactHausdorff(A,prob=rep(1/nrow(A),nrow(A)),r)$alldistances
      u2 = exactHausdorff(B,prob=rep(1/nrow(B),nrow(B)),r)$alldistances
      pHtemp = c(pHtemp,uso.test(u1,u2)$p.ks)
    }
    pH = rbind(pH,pHtemp)
    ## For simplex
    pStemp = NULL
    for(nresamples in nresamplesvalues){
      v1 = simplex(A,bootstrap=TRUE,nresamples)
      v2 = simplex(B,bootstrap=TRUE,nresamples)
      pStemp = c(pStemp,uso.test(v1,v2)$p.ks)
    }
    pS = rbind(pS,pStemp)
  }
  resultH = rbind(resultH,c(lambda,apply(pH,2,mean)))
  resultS = rbind(resultS,c(lambda,apply(pS,2,mean)))
}

save(resultH,file="resultH.RData")
save(resultS,file="resultS.RData")

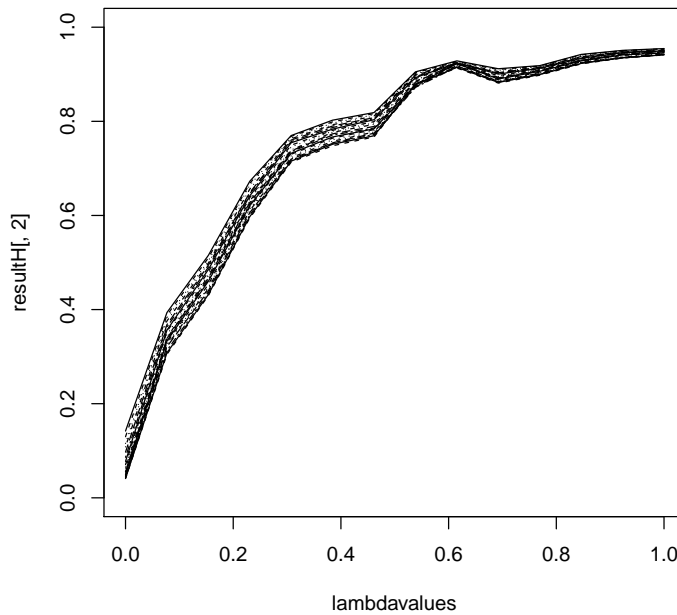
```

For the Hausdorff dispersion order, the mean p -values appear in the following figure (not included in the paper). They correspond to the following r values.

```
> rvalues
```

```
[1] 0.01000000 0.03578947 0.06157895 0.08736842 0.11315789 0.13894737
[7] 0.16473684 0.19052632 0.21631579 0.24210526 0.26789474 0.29368421
[13] 0.31947368 0.34526316 0.37105263 0.39684211 0.42263158 0.44842105
[19] 0.47421053 0.50000000
```

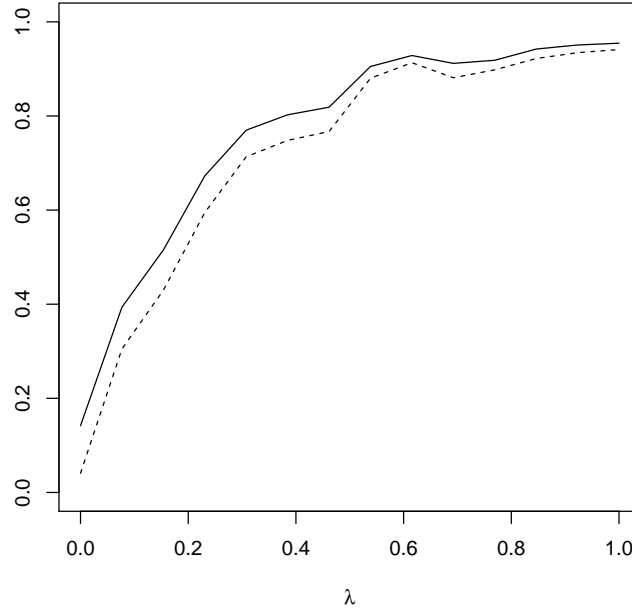
```
> load("resultH.RData")
> plot(lambdavalues,resultH[,2],type="l",ylim=c(0,1))
> for(i in 3:ncol(resultH))
+   lines(lambdavalues,resultH[,i],lty=i-1)
```



In the paper (figure 7) there appear a simplified plot using For the Hausdorff dispersion order, the mean p -values appear in Figure ???. They correspond to the indices $r = 0.01$ (solid line) and $r = 0.5$ (dashed line).

The code use to produce this figure 7. Hausdorff dispersion order: Mean p -values for the different λ 's using $r = 0.01$ (solid line) and $r = 0.5$ (dashed line).

```
> plot(lambdavalues,resultH[,2],type="l",ylim=c(0,1),
+       xlab=expression(lambda),ylab="")
> lines(lambdavalues,resultH[,ncol(resultH)],lty=2)
```

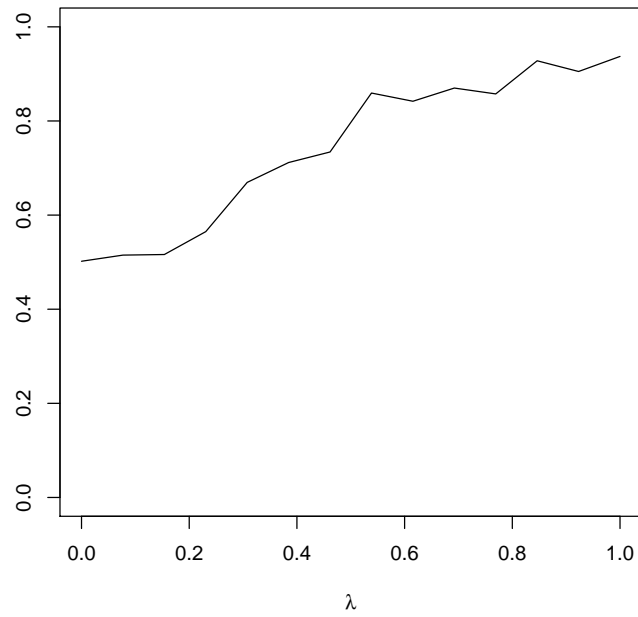


Note that greater p -values correspond in this case with a better performance of the test. We should indicate that the choice of r should depend on the particular problem at hand. Note that in this case the whole set of Hausdorff distances is calculated.

For the simplex dispersion ordering we have used the same data sets. We should note that in this case it is not possible to obtain the whole set of distances by computational reasons. Therefore we have used bootstrap resamples from the whole set. An experimental factor to be evaluated is the number of bootstrap resamples. Note that the sample sizes used with simplex dispersion order are clearly smaller. The number of resamples goes from 100 to 300 with a step of 50. For each resample size, we have calculated the mean p -values for all the replications. The different curves in Figure ?? correspond with the different resample sizes: 100 (solid), 150 (dashed), 200 (dotted), 250 (dotdash) and 300 (longdash). We think that the number of resamples is not so critic in the final performance.

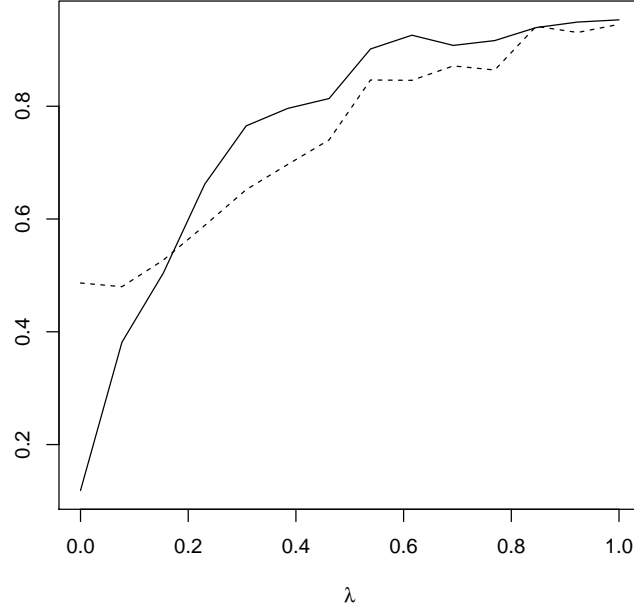
Now the first plot display one simulation.

```
> load("resultS.RData")
> plot(lambdavalues,resultS[,2],type="l",ylim=c(0,1),
+      xlab=expression(lambda),ylab="")
```

The second plot can be found at figure 8 in the paper.

```
> plot(lambdavalues, apply(resultH[, 2:ncol(resultS)], 1, mean), type="l",
+       xlab=expression(lambda), ylab="")
> lines(lambdavalues, apply(resultS[, 2:ncol(resultS)], 1, mean), lty=2)
```



Simplex dispersion order: Mean p -values for the different λ 's. Each curve corresponds with a different number of bootstrap samples. This number of resamples goes from 100 to 300 with a step of 50: 100 (solid), 150 (dashed), 200 (dotted), 250 (dotdash) and 300 (longdash).

3 An ophthalmological application

The 2D Hausdorff and simplex dispersion orderings will be applied to the study of the effects produced by hypertension in the retinal vessels. It is a well known fact in ophthalmology that hypertension alters retinal vessel diameters. For this research we have selected fundus images belonging to normal persons and also to hypertensive patients with different severity levels. Additionally, the effect of other clinically important covariables will be considered. The possible difference in dispersion that could exist between the groups defined using these categorical covariables will be tested. Any software tool that pretends a reliable follow up of the disease and its consequences over the retinal vessel tree, should take into account those covariables producing a clearly higher dispersion. Observe that the comparison between homogeneous groups will produce accurate results in medical research.

Note that for an experimental factor with J levels we are performing hypothesis tests per each circumference and there are three circumferences. Thus the total number of related test is $N = 3 \times J \times (J - 1)$. If a global significance

level α is specified then the significance level for each individual test will be α/N because a Bonferroni correction will be used.

The number of levels will be two for the diabetes and three for the evolution times of diabetes. As we will use the usual significance level $\alpha = 0.05$ then the corresponding significance levels will be 0.008 and 0.003 respectively. The tables will show the p -values observed, the interpretation of them is given within the text.

First, we give some previous considerations about the application of both dispersion orders. Note that, given a sample of size n then we can calculate the whole set of Hausdorff distances. However, this is not possible for the simplex dispersion ordering, from the original data set of size n we have the different convex hulls from each three points, i.e., with no multiple points (as usual for continuous data) the total number is $\binom{n}{3}$. Therefore we have chosen a random sample from this huge number of possible distances, namely one thousand distances. When the Hausdorff dispersion ordering is used, the value chosen for r has been 7 pixels.

Lo primero hemos de leer los datos.

```
> data(hypertension)
> attach(hypertension)
```

We can see the two first rows.

```
> hypertension[1:2,]

  dm tevoldm mveinC1 mveinC2 mveinC3 martC1 martC2 martC3
1  2         2 19.62457 21.35279 24.83126 21.00412 18.20606 20.04386
2  2         2 20.20696 20.07059 20.99320 17.76969 22.75071 18.06508
```

These are the data that we will use later in this paper.

The whole explanation appears in section 6.1 of Ayala et al. [2012].

6.1.1. Diabetes

There are three different categories for the factor diabetes: non diabetic (coded as 0), diabetic type 1 (coded as 1) and diabetic type 2 (coded as 2). In the paper, we consider the recoded variable where we distinguish only between diabetic and non diabetic.

```
> levels(dm) = c("0", "1", "1")
>
```

Set the value of r (see details in the paper).

```
> r = 7
```

The following code give us the p -values used in Table 1.

```

> y1 = cbind(mveinC1,martC1)
> y2 = cbind(mveinC2,martC2)
> y3 = cbind(mveinC3,martC3)
> duplicados = !duplicated(y1)
> y1 = y1[duplicados,]
> y2 = y2[duplicados,]
> y3 = y3[duplicados,]
> x0 = dm[duplicados]
> hc1 = testHDO(y1,x0,r,dib=T,nameplot = "dm_hc1.eps")
> hc2 = testHDO(y2,x0,r,dib=T,nameplot = "dm_hc2.eps")
> hc3 = testHDO(y3,x0,r,dib=T,nameplot = "dm_hc3.eps")

```

The obtained p-values are

```

> hc1

      [,1]      [,2]
[1,] 1.0000000 7.980641e-10
[2,] 0.5752035 1.000000e+00

> hc2

      [,1]      [,2]
[1,] 1.0000000 3.560855e-19
[2,] 0.6867962 1.000000e+00

> hc3

      [,1]      [,2]
[1,] 1.0000000 3.560855e-19
[2,] 0.6867962 1.000000e+00

```

Similarly, the values in table 2 are obtained using the following code.

```

> bootstrap=TRUE
> nresamples= 10
> sc1 = testSDO(y1,x0,bootstrap,nresamples,dib=TRUE,nameplot = "dm_sc1.eps")
> sc2 = testSDO(y2,x0,bootstrap,nresamples,dib=TRUE,nameplot = "dm_sc2.eps")
> sc3 = testSDO(y3,x0,bootstrap,nresamples,dib=TRUE,nameplot = "dm_sc3.eps")

```

The obtained p-values are

```

> sc1

      [,1]      [,2]
[1,] 1 0.2018965
[2,] 1 1.0000000

> sc2

```

```

      [,1]      [,2]
[1,] 1.0000000 0.4065697
[2,] 0.9048374 1.0000000

```

```
> sc3
```

```

      [,1]      [,2]
[1,] 1.00000 0.67032
[2,] 0.67032 1.00000

```

6.1.2. Evolution time of diabetes

Now, we reproduce the analysis for the evolution time of the diabetes. Now the factor is `tevoldm` in the data frame `hypertension`.

```

> x0 = tevoldm[duplicados]
> hc1 = testHDO(y1,x0,r,dib=T,nameplot = "tevoldm_hc1.eps")
> hc2 = testHDO(y2,x0,r,dib=T,nameplot = "tevoldm_hc2.eps")
> hc3 = testHDO(y3,x0,r,dib=T,nameplot = "tevoldm_hc3.eps")
> bootstrap=TRUE
> nresamples= 10
> sc1 = testSDO(y1,x0,bootstrap=TRUE,nresamples,dib=T,
+   nameplot = "tevoldm_sc1.eps")
> sc2 = testSDO(y2,x0,bootstrap=TRUE,nresamples,dib=T,
+   nameplot = "tevoldm_sc2.eps")
> sc3 = testSDO(y3,x0,bootstrap=TRUE,nresamples,dib=T,
+   nameplot = "tevoldm_sc3.eps")

```

The obtained p-values are

```
> hc1
```

```

      [,1]      [,2]      [,3]
[1,] 1.0000000 0.3267524 1.577646e-06
[2,] 0.3202877 1.0000000 6.124532e-03
[3,] 0.4056814 1.0000000 1.000000e+00

```

```
> hc2
```

```

      [,1]      [,2]      [,3]
[1,] 1.0000000 5.410832e-05 7.22265e-08
[2,] 0.4178555 1.000000e+00 5.74510e-01
[3,] 0.2127720 5.044714e-01 1.00000e+00

```

```
> hc3
```

```

      [,1]      [,2]      [,3]
[1,] 1.0000000 5.410832e-05 7.22265e-08
[2,] 0.4178555 1.000000e+00 5.74510e-01
[3,] 0.2127720 5.044714e-01 1.00000e+00

```

```

> sc1

      [,1]      [,2]      [,3]
[1,] 1.0000000 0.9048374 0.4065697
[2,] 0.4065697 1.0000000 0.2018965
[3,] 0.6703200 0.9048374 1.0000000

> sc2

      [,1]      [,2]      [,3]
[1,] 1.0000000 0.6703200 0.4065697
[2,] 0.6703200 1.0000000 0.4065697
[3,] 0.9048374 0.9048374 1.0000000

> sc3

      [,1]      [,2]      [,3]
[1,] 1.0000000 0.4065697 0.4065697
[2,] 0.6703200 1.0000000 0.6703200
[3,] 0.9048374 0.6703200 1.0000000

>

```

References

- G. Ayala. *disp2D: 2D Hausdorff and Simplex Dispersion Orderings*, 2012. URL <http://CRAN.R-project.org/package=disp2D>. R package version 1.0.
- G. Ayala, M. López-Díaz, M. López-Díaz, and L. Martínez-Costa. Methods and algorithms to test the simplex and hausdorff dispersion orders with a simulation study and an ophthalmological application. Technical Report, 2012.
- C. B. Barber, K. Habel, R. Grasman, R. B. Gramacy, A. Stahel, and D. C. Sterratt. *geometry: Mesh generation and surface tessellation*, 2012. URL <http://CRAN.R-project.org/package=geometry>. R package version 0.3-2.
- A. Genz, F. Bretz, T. Miwa, X. Mi, F. Leisch, F. Scheipl, and T. Hothorn. *mvtnorm: Multivariate Normal and t Distributions*, 2011. URL <http://CRAN.R-project.org/package=mvtnorm>. R package version 0.9-9991.
- F. Harrell. *Hmisc: Harrell Miscellaneous*, 2012. URL <http://CRAN.R-project.org/package=Hmisc>. R package version 3.9-3.