

Supplementary Materials

Title: “Thurstonian model for the difference-from-control test” by Bi and Kuesten

Appendix A: R codes used in the paper

R codes

No	Code	Description
1	DFCindex(d)	The DFC indices δ_f and A_{dfc} value(s) for a given δ value(s)
2	DFCmle(x)	Maximum likelihood estimations of δ , δ_f , and A_{dfc} for given DFC data
3	DFCnoe(x)	Nonparametric estimations of A_{dfc} and δ for given DFC data
4	dptest(d,v)	Difference testing for one d-prime
5	dpstest(d,v, slim)	Similarity testing for one d-prime
6	dtest (d, v)	Difference testing for multiple d-prime values
7	s2dptest(d,v,d0)	Similarity testing for two d-prime values
8	DFCpower(d,samp,alpha)	Difference testing power
9	DFCsamp(d,pow,alpha)	Difference testing sample size
10	dfc6<-DFC6ps()	Producing 6-point scale DFC data
11	dfc3<-DFC3ps()	Producing 3-point scale DFC data
12	dfc2<-DFC2ps()	Producing 2-point scale DFC data
13	alldp<-DFCdps()	Producing d-prime data for testing in the paper
14	DFCmle0(x)	Maximum likelihood estimations of fitted number of ratings for given observed DFC data

Data file: “dfc6<-DFC6ps()”, “dfc3<-DFC3ps()”, “dfc2<-DFC2ps()”

```
> library(numDeriv)
> library(MASS)
> library(multcomp)
```

Example:

```
> DFCindex(d=seq(0,3,0.1))
      d'    df Adfc
1  0.0  1.13  0.50
2  0.1  1.13  0.50
3  0.2  1.14  0.50
...

30 2.9  2.92  0.86
31 3.0  3.02  0.88
> DFCmle(x=dfc6[,c(1,2)])
      value variance
d'      2.42    0.0203
df      2.47    0.0169
Adfc    0.80    0.0004
> DFCmle(x=dfc6[,c(1,3)])
      value variance
```

```

d'      0.88    0.0390
df      1.34    0.0085
Adfc    0.56    0.0006
> DFCmle(x=dfc6[,c(1,4)])
      value variance
d'      2.33    0.0199
df      2.39    0.0161
Adfc    0.79    0.0005

>dfc6<-DFC6ps()
> dfc6
      Blind Control vs Identified Control Test 1 vs Identified Control Test 2 vs
Identified Control Test 3 vs Identified Control
5              2              10
2              15
4              5              18
6              5
3              15              34
23             46
2              17              30
22             19
1              20              2
18             10
0              41              6
29             5

> DFCnoe(x=dfc6[,c(1,2)])
Ratings for DFC
      estimate variance
Adfc      0.79    0.0010
d'        2.38    0.0447
df        2.43    0.0367
> DFCnoe(x=dfc6[,c(1,3)])
Ratings for DFC
      estimate variance
Adfc      0.58    0.0016
d'        1.03    0.0856
df        1.42    0.0245
> DFCnoe(x=dfc6[,c(1,4)])
Ratings for DFC
      estimate variance
Adfc      0.78    0.0011
d'        2.31    0.0446
df        2.37    0.0359

> alldp<-DFCdps()
> alldp
      d'  v(d')
T1 2.42 0.0203
T2 0.88 0.0390
T3 2.33 0.0199

> dpdtest(d = alldp [2,1],v = alldp [2,2])
      d' var(d')      z      p-v
[1,] 0.88    0.039 4.456053 4.174116e-06
> dpstest(d = alldp [2,1],v = alldp [2,2],slim=1.5)

```

```

      d' var(d') lim      z      p-v
[1,] 0.88  0.039 1.5 -3.139492 0.0008462051
> dstest(d=alldp[,1],v=alldp[,2])
p-value: 0
  Weighted mean: 2.0684
  Variance of Wm: 0.008
[1] 0.0000 2.0684 0.0080
> s2dptest(d = alldp[c(1,3),1],v =alldp[c(1,3),2],d0 = 0.5)
Z1,Z2,pv1,pv2:
[1] 2.9427 -2.0449 0.0016 0.0204
> dp1<-alldp[,1]
> dv1<-matrix(0,3,3)
> diag(dv1)<-alldp[,2]
> dp1
      T1      T2      T3
2.4168 0.8847 2.3322
> dv1
      [,1] [,2] [,3]
[1,] 0.0203 0.000 0.0000
[2,] 0.0000 0.039 0.0000
[3,] 0.0000 0.000 0.0199
In S-PLus
> multcomp(x=dp1,vmat=dv1,alpha = 0.2)
80 % simultaneous confidence intervals for specified
linear combinations, by the Tukey method
critical point: 1.7151
response variable:
intervals excluding 0 are flagged by '****'
      Estimate Std.Error Lower Bound Upper Bound
T1-T2      1.54      0.244      1.120      1.960 ****
T1-T3      0.09      0.200     -0.254      0.434
T2-T3     -1.45      0.243     -1.870     -1.030 ****
In R
> library(multcomp)
> confint(glht(model=parm(dp1,dv1),linfct=c("T1-T2=0","T1-T3=0","T2-
T3=0")),level=0.8)
      Simultaneous Confidence Intervals
Fit: NULL
Quantile = 1.7099
80% family-wise confidence level
Linear Hypotheses:
      Estimate lwr      upr
T1 - T2 == 0  1.5400  1.1236  1.9564
T1 - T3 == 0  0.0900 -0.2528  0.4328
T2 - T3 == 0 -1.4500 -1.8650 -1.0350

> DFCpower(d=1.25,samp=100,alpha=0.1)

```

```

A1: 0.61
[1] 0.8242243
> DFCpower(d=1.2,samp=100,alpha=0.1)
A1: 0.6
[1] 0.7802828
> DFCsamp(d=1.25, pow=0.8, alpha=0.1)
A1: 0.61
[1] 93
dfc3<-DFC3ps()
dfc2<-DFC2ps()
> dfc3
  Blind Control vs Identified Control Test 1 vs Identified Control
Test 2 vs Identified Control Test 3 vs Identified Control
3                                7                                28
8                                20                                64
2                                32                                8
45                               65                                8
1                                61                                8
47                               15
> dfc2
  Blind Control vs Identified Control Test 1 vs Identified Control
Test 2 vs Identified Control Test 3 vs Identified Control
2                                22                                62
31                               66                                38
1                                78                                38
69                               34

> library(sensR)
> dod(same=rev(dfc6[,1]),diff=rev(dfc6[,2]))

Results for the Thurstonian model for the Degree-of-Difference method

Confidence level for 2-sided profile likelihood interval: 95%

      Estimates Std. Error Lower Upper
d.prime      2.391      0.2211 1.946 2.818
...

> 0.2211^2
[1] 0.04888521

> DFCmle0(dfc6[,c(1,2)])
  C1 vs C T vs C
5  0.0043 0.1263
4  0.0290 0.2113
3  0.1582 0.3201
2  0.2647 0.1817
1  0.1660 0.0624
0  0.3777 0.0981
> dfc6[,c(1,2)]/100
  Blind Control vs Identified Control Test 1 vs Identified Control
5                                0.02                                0.10
4                                0.05                                0.18

```

3	0.15	0.34
2	0.17	0.30
1	0.20	0.02
0	0.41	0.06

```
> DFCmle0(df6[,c(1,3)])
```

C1 vs C T vs C

5	0.0118	0.0299
4	0.0412	0.0705
3	0.1688	0.2075
2	0.1945	0.1930
1	0.2013	0.1797
0	0.3824	0.3194

```
> df6[,c(1,3)]/100
```

Blind Control vs Identified Control Test 2 vs Identified Control

5	0.02	0.02
4	0.05	0.06
3	0.15	0.23
2	0.17	0.22
1	0.20	0.18
0	0.41	0.29

```
> DFCmle0(df6[,c(1,4)])
```

C1 vs C T vs C

5	0.0090	0.1680
4	0.0152	0.1048
3	0.1933	0.3905
2	0.2058	0.1458
1	0.2119	0.0877
0	0.3648	0.1032

```
> df6[,c(1,4)]/100
```

Blind Control vs Identified Control Test 3 vs Identified Control

5	0.02	0.15
4	0.05	0.05
3	0.15	0.46
2	0.17	0.19
1	0.20	0.10
0	0.41	0.05

R Codes

#1

```
DFCindex<-function(d) {
#DFC values of parameters
#####
DFCf0<-function(d) {
#Difference-from-control, Bradley (1963)
#for  $E(|x-y|)/se$ 
k<-length(d)
df<-function(d) {
dfc<-d+(2/sqrt(pi))*exp(-d^2/4)-2*d*(1-pnorm(d/sqrt(2)))
dfc<-round(dfc,4)
dfc}
dfcs<-rep(0,k)
for (i in 1:k) {dfcs[i]<-df(d[i])}
dfcs
}
#####
Adfc0<-function(d) {
#Area of AUC for the DFC
area<-pnorm(d/2)^2+pnorm(-d/2)^2
area<-round(area,4)
area
}
#####
df<-DFCf0(d)
are<-Adfc0(d)
tab<-round(cbind(d,df,are),2)
dimnames(tab)<-list(seq(1,length(d)),c("d'", "df", "Adfc"))
#write.table(tab,"c:\\tem\\tab",sep=",")
tab
}
```

#2

```
DFCmle<-function(x) {
DFCdp0<-function(x) {
#ML estimation of d and df in DFC tes
#Ratings of CDF
x0 <- x
k <- length(x0[, 1])
#ml estimation
#####
sdml0<-function(pr, y)
{
x <- y
k <- length(x[, 1])
x1 <- as.vector(cumsum(rev(x[, 1])))
x2 <- as.vector(cumsum(rev(x[, 2])))
ns <- x1[k]
nd <- x2[k]
p <- k - 1
L <- 0
for(i in 1:p) {
```

```

ps <- 2 * pnorm(pr[i]/sqrt(2)) - 1
pd <- pnorm(pr[i]/sqrt(2) - pr[p + 1]/sqrt(2)) - pnorm(- pr[i]/sqrt(2) -
pr[p + 1]/sqrt(2))
L <- L + x1[i] * log(ps) + (ns - x1[i]) * log(1 - ps) + x2[i] * log(pd) + (nd
- x2[i]) * log(1 - pd)
}
- L
}
#####
k <- length(x[, 1])
a0 <- cumsum(rev(x[, 1]))/sum(x[, 1])
a <- qnorm((a0 + 1)/2)
a[k] <- 1
xx <- nlminb(start = a, y = x, objective = sdml0)
dd <- xx$par[k]
library(numDeriv)
vv<-solve(hessian(sdml0,xx$par,y=x))[k,k]
all<-c(dd,vv)
#cat("d-prime and its variance for DFC test","\n")
all}
#####
DFCdf0<-function(dvd){
###dvd is a vector of d' and its variance from DFC
#####
dfcdf0<-function(d,vd){
#For DF
DFCf0<-function(d){
#Difference-from-control, Bradley (1963)
#for  $E(|x-y|)/se$ 
k<-length(d)
df<-function(d){
dfc<-d+(2/sqrt(pi))*exp(-d^2/4)-2*d*(1-pnorm(d/sqrt(2)))
dfc<-round(dfc,4)
dfc}
dfcs<-rep(0,k)
for (i in 1:k){dfcs[i]<-df(d[i])}
dfcs
}
#####
DF<-DFCf0(d)
df<--1-(d/sqrt(pi))*exp(-d^2/4)+d*sqrt(2)*dnorm(d/sqrt(2))+2*pnorm(d/sqrt(2))
VD<-df^2*vd
all<-round(c(DF,VD),4)
#cat("df and its variance","\n")
all
}
#####
df<-dfcdf0(d=dvd[1],vd=dvd[2])
df
}
#####
DFCA0<-function(dd){
#from d to A
d<-dd[1]
vd<-dd[2]
a<-pnorm(d/2)^2+pnorm(-d/2)^2
ap<-dnorm(d/2)*(2*pnorm(d/2)-1)

```

```

va<-vd*ap^2
all<-c(a,va)
all
}
#####
dd<-DFCdp0(x)
df<-DFCdf0(dd)
aa<-DFCA0(dd)
all<-matrix(0,3,2)
dimnames(all)<-list(c("d'", "df", "Adfc"), c("value", "variance"))
all[1,]<-dd
all[2,]<-df
all[3,]<-aa
all[,1]<-round(all[,1],2)
all[,2]<-round(all[,2],4)
all
}

#3
DFCnoe<-function(x)
{
#non-parameter estimate of d' and delta method for variance of d'
av <- function(a, n, m)
{
q1 <- a/(2 - a)
q2 <- (2 * a * a)/(1 + a)
va <- a * (1 - a) + (n - 1) * (q1 - a * a) + (m - 1) * (q2 - a * a)
va <- va/(n * m)
va
}
#####
aest0 <- function(x)
{
k <- length(x[, 1])
y1 <- rep(seq(1, k), x[, 1])
y2 <- rep(seq(1, k), x[, 2])
k1 <- sum(x[, 1])
k2 <- sum(x[, 2])
ee <- matrix(0, k1, k2)
for(i in 1:k1)
for(j in 1:k2) {
ee[i, j] <- sign(y1[i] - y2[j]) + 1
}
ee <- ee/2
su <- sum(ee)/(k1 * k2)
su
}
#####
k <- length(x[, 1])
m <- sum(x[, 1])
n <- sum(x[, 2])
a <- aest0(x)
a <- as.numeric(a)
if(a < 0.5) {
a <- 1 - a
}
else {

```

```

a <- a
}
arv <- av(a, n, m)
arv
m <- c("Ratings for DFC")
d <- 2 * qnorm(0.5 + 0.5 * sqrt(2 * a - 1))
dv <- arv/(dnorm(d/2)^2 * (2 * pnorm(d/2) - 1)^2)
df<-d+(2/sqrt(pi))*exp(-d^2/4)-2*d*(1-pnorm(d/sqrt(2)))
dp<--1-(d/sqrt(pi))*exp(-d^2/4)+d*sqrt(2)*dnorm(d/sqrt(2))+2*pnorm(d/sqrt(2))
dfv<-dp^2*dv
#all <- round(c(a, arv, d, dv, df, dfv), 4)
all<-matrix(0,3,2)
all[,1]<-c(a,d,df)
all[,2]<-c(arv,dv,dfv)
dimnames(all)<-list(c("Adfc","d',"df"),c("estimate","variance"))
cat(m, "\n")
all[,1]<-round(all[,1],2)
all[,2]<-round(all[,2],4)
all
}

```

#4

```

dpdtest<-function(d,v){
#difference test for d'
t2<-matrix(0,1,4)
t2[,1:2]<-c(d,v)
dimnames(t2)[[2]]<-c("d'", "var(d')", "z", "p-v")
t2[,3]<-d/sqrt(v)
t2[,4]<-1-pnorm(t2[,3])
t2
}

```

#5

```

dpstest<-function(d,v,slim){
#similarity test for d'
#slim is similarity limit in terms of d'
t2<-matrix(0,1,5)
dimnames(t2)[[2]]<-c("d'", "var(d')", "lim", "z", "p-v")
t2[,1:2]<-c(d,v)
t2[,3]<-slim
t2[,4]<-(d-slim)/sqrt(v)
t2[,5]<-pnorm(t2[,4])
t2
}

```

#6

```

dstest<-function(d, v)
{
#test for multiple d's
k <- length(d)
for(i in 1:k) {
if(v[i] == 0) {
v[i] <- 0.0001
}
}
w0 <- sum(1/v)
d0 <- sum(d/v)/w0
}

```

```

u <- sum((d - d0)^2/v)
p <- 1 - pchisq(u, k - 1)
all <- c(p, d0, 1/w0)
all <- round(all, 4)
cat("p-value:", all[1], "\n", "Weighted mean:", all[2], "\n", "Variance of
Wm:", all[3], "\n")
all
}

#7
s2dptest<-function(d,v, d0)
{#simi test for two d-primes
#d and v are vectors
#Schuirmann TOST, JPB (1987), 15,657
d<-as.vector(d)
v<-as.vector(v)
s <- sqrt(v[1] + v[2])
z2 <- (d[1] - d[2] - d0)/s
z1 <- (d[1] - d[2] + d0)/s
pv2 <- pnorm(z2)
pv1 <- 1 - pnorm(z1)
res <- c(z1,z2,pv1, pv2)
res <- round(res, 4)
cat("Z1,Z2,pv1,pv2:", "\n")
res
}

#8
DFCpower<-function(d, samp,alpha)
{#Difference testing power for DFC
rocsdpow2 <- function(d, samp,alpha)
{
#power of SD with ratings
rocs <- function(d)
{
#area under same-different roc
a <- pnorm(d/2)^2 + pnorm(- d/2)^2
a
}
d1 <- rocs(d)
cat("A1:", round(d1, 2), "\n")
pow <- 1 - pnorm((qnorm(1-alpha) * 0.5 - (d1 - 0.5) * sqrt(samp))/sqrt(d1 *
(1 - d1)))
pow
}
powf <- rocsdpow2
pow <- powf(d, samp,alpha)
pow
}

#9
DFCsamp<-function(d, pow,alpha)
{#Sample size for DFC difference testing
rocsdsam2 <- function(d, pow, alpha)
{
#sample of SD with ratings
rocs <- function(d)

```

```

{
#area under same-different roc
a <- pnorm(d/2)^2 + pnorm( - d/2)^2
a
}
d1 <- rocs(d)
cat("A1:", round(d1, 2), "\n")
sam <- ((qnorm(1-alpha) * 0.5 + sqrt(d1 * (1 - d1)) * qnorm(pow))/(d1 -
0.5))^2
#sam <- sam + 2/(d1 - 0.5)
sam <- ceiling(sam)
sam
}
samf <- rocSDsam2
sam <- samf(d, pow, alpha)
sam
}

#10
DFC6ps<-function(){
#DFC data (6-point scale)
dat<-matrix(0,6,4)
dimnames(dat)<-list(seq(5,0),c("Blind Control vs Identified Control", "Test 1
vs Identified Control","Test 2 vs Identified Control","Test 3 vs Identified
Control"))
dat[,1]<-c(2,5,15,17,20,41)
dat[,2]<-c(10,18,34,30,2,6)
dat[,3]<-c(2,6,23,22,18,29)
dat[,4]<-c(15,5,46,19,10,5)
dat
}

#11
DFC3ps<-function(){
#DFC data (3-point scale)
dat<-matrix(0,3,4)
dimnames(dat)<-list(seq(3,1),c("Blind Control vs Identified Control", "Test 1
vs Identified Control","Test 2 vs Identified Control","Test 3 vs Identified
Control"))
dat[,1]<-c(7,32,61)
dat[,2]<-c(28,64,8)
dat[,3]<-c(8,45,47)
dat[,4]<-c(20,65,15)
dat
}

#12
DFC2ps<-function(){
#DFC data (2-point scale)
dat<-matrix(0,2,4)
dimnames(dat)<-list(seq(2,1),c("Blind Control vs Identified Control", "Test 1
vs Identified Control","Test 2 vs Identified Control","Test 3 vs Identified
Control"))
dat[,1]<-c(22,78)
dat[,2]<-c(62,38)
dat[,3]<-c(31,69)
dat[,4]<-c(66,34)

```

```
dat
}
```

#13

```
DFCdps<-function(){
#DFC all dps
all<-matrix(0,3,2)
dimnames(all)<-list(c("T1","T2","T3"),c("d'", "v(d')"))
all[,1]<-c(2.42,0.88,2.33)
all[,2]<-c(0.0203,0.0390,0.0199)
all
}
```

#14

```
DFCmle0<-function(x){
#Predicted probabilities of categories of the DFC ratings
#Ratings of DFC
x0 <- x
k <- length(x0[, 1])
#mle estimation
#####
sdml0<-function(pr, y)
{
x <- y
k <- length(x[, 1])
x1 <- as.vector(cumsum(rev(x[, 1])))
x2 <- as.vector(cumsum(rev(x[, 2])))
ns <- x1[k]
nd <- x2[k]
p <- k - 1
L <- 0
for(i in 1:p) {
ps <- 2 * pnorm(pr[i]/sqrt(2)) - 1
pd <- pnorm(pr[i]/sqrt(2) - pr[p + 1]/sqrt(2)) - pnorm(- pr[i]/sqrt(2) -
pr[p + 1]/sqrt(2))
L <- L + x1[i] * log(ps) + (ns - x1[i]) * log(1 - ps) + x2[i] * log(pd) + (nd
- x2[i]) * log(1 - pd)
}
- L
}
#####
k <- length(x[, 1])
a0 <- cumsum(rev(x[, 1]))/sum(x[, 1])
a <- qnorm((a0 + 1)/2)
a[k] <- 1
xx <- nlminb(start = a, y = x, objective = sdml0)
dd <- xx$par[k]
library(numDeriv)
vv<-solve(hessian(sdml0,xx$par,y=x))[k,k]
all<-c(dd,vv)
#cat("d-prime and its variance for DFC test","\n")
est<-xx$par
est
p0<-dim(x)[1]
ps0<-rep(0,p0-1)
pd0<-ps0
ps0[1]<-2*pnorm(est[1]/sqrt(2))-1
```

```

pd0[1]<-pnorm(est[1]/sqrt(2)-est[p0]/sqrt(2))-pnorm(-est[1]/sqrt(2)-
est[p0]/sqrt(2))
for(i in 2:(p0-1)){ps0[i]<-2*pnorm(est[i]/sqrt(2))-1
pd0[i]<-pnorm(est[i]/sqrt(2)-est[p0]/sqrt(2))-pnorm(-est[i]/sqrt(2)-
est[p0]/sqrt(2))
}
ps1<-c(ps0,1)
pd1<-c(pd0,1)
for(i in p0:2){ps1[i]<-ps1[i]-ps1[i-1]
pd1[i]<-pd1[i]-pd1[i-1]}
all<-cbind(rev(ps1),rev(pd1))
dimnames(all)<-list(seq(p0-1,0),c("C1 vs C","T vs C"))
all<-round(all,4)
all
}

```