Compositional Data Analysis Supplement

Introduction

This Supplement contains a brief guide to current software dedicated to compositional analysis, followed by the annotated R code and selected output for all of the examples in the paper. The data-sets for these examples are freely available from the corresponding author.

Most of the software packages dedicated to compositional data are in the R environment. An exception is `CoDaPack`, a standalone add-on to Excel written in Visual Basic (Comas Cufí & Thió i Fernández de Henestrosa, 2011). Its functions include log-ratio transformations, descriptive statistics, a variety of graphs and plots, and operations within the simplex: Centering, perturbation, power transformation, amalgamation, subcomposition (closure) or rounded zero replacement.

Turning now to packages in R, the `compositions` package (Van den Boogaart & Tolosana-Delgado, 2008) supports four different multivariate scales, two of which implement the "relative geometry" required for compositional data. Its functions include log-ratio transformations, ternary plots, descriptive statistics for compositions, and compositional principal components analysis.

The `compositional` package (Tsagris & Athineou, 2016) has a considerable variety of functions, including those mentioned thus far, plus numerous helper functions to assist various multivariate analysis techniques such as regression, classification, and Dirichlet general linear models.

The `ToolsForCoDa` package (Graffelman, Pawlowsky-Glahn, Egozcue & Buccianti, 2018) is more specialized, focusing on canonical correlation analysis with compositional data. A function for log-ratio principal components analysis also is included in this package.

Likewise, the `robcompositions` package (Templ, Hron & Filzmoser, 2011) is specialized, focusing on robust methods for principal components and factor analysis, discriminant analysis, and multiple regression. It also incorporates popular log-ratio transforms and imputation methods for dealing with missing data.

Finally, the `zcompositions` package (Palarea-Albaladejo & Martín-Fernández, 2015) is devoted to the zeros problem and missing data imputation. It provides several methods for imputing zeros, left-censored, and missing data, including Bayesian imputation methods tailored for compositional data.

Emergency Department Example

The first code chunk creates the necessary variables and reshapes the data-set.

```
# Load libraries
library(gamlss)
# Read the data
yoon <- read.table("Patients894.txt", header = TRUE)
attach(yoon)
#
# Combine the first two stages (registration and triage).
# We need to combine Triage 1 and 2 levels:
Triage12 <- Triage1 + Triage2
pregtriage <- preg + ptriage
pregtriage0 <- preg0*ptriage0; table(pregtriage0)
# First, create the log-ratio variables.
lrat1 <- c(rep(NA,894)); lrat2 <- c(rep(NA,894))
lrat1[pregtriage0==0 & pnurse0==0] <-
log(pregtriage[pregtriage0==0 & pnurse0==0]/
pphysdecis[pregtriage0==0 & pnurse0==0])
lrat2[pregtriage0==0 & pnurse0==0] <-
log(pnurse[pregtriage0==0 & pnurse0==0]/
pphysdecis[pregtriage0==0 & pnurse0==0])
# Also create the probability-ratios:
prat1 <- c(rep(NA,894)); prat2 <- c(rep(NA,894))
prat1[pregtriage0==0 & pnurse0==0] <-
```

```
pregtriage[pregtriage0==0 & pnurse0==0]/
(pregtriage[pregtriage0==0 & pnurse0==0]
+pphysdecis[pregtriage0==0 & pnurse0==0])
prat2[pregtriage0==0 & pnurse0==0] <-
pnurse[pregtriage0==0 & pnurse0==0]/
(pnurse[pregtriage0==0 & pnurse0==0]
+pphysdecis[pregtriage0==0 & pnurse0==0])
# Put all of the relevant variables together and get rid of the NA's:
newdat <- as.data.frame(na.omit(cbind(id,Day,Ambulance,Triage12,
Triage3,Triage4,Triage5,Lab,Xray,Other,LOS,LOSh,pregtriage,
pnurse,pphysdecis,pregtriage0,pnurse0,lrat1,lrat2,prat1,prat2)))
#
#
# Create a long version of the same data:
matsize <- 40*length(newdat[,1])
longdat <- matrix(c(rep(0,matsize)), nrow = 2*length(newdat[,1]))
for (i in 1:790) {
  longdat[2*i-1,] <-
  c(newdat$id[i],newdat$Day[i],newdat$Ambulance[i],newdat$Triage12[i],
  newdat$Triage3[i],newdat$Triage4[i],newdat$Triage5[i],newdat$Lab[i],
  newdat$Xray[i],newdat$Other[i],newdat$LOS[i],newdat$LOSh[i],
  newdat$pregtriage[i],newdat$pnurse[i],newdat$pphysdecis[i],
  newdat$pregtriage0[i],newdat$pnurse0[i],newdat$lrat1[i],
  newdat$prat1[i],0);
  longdat[2*i,] <- c(newdat$id[i],newdat$Day[i],newdat$Ambulance[i],
  newdat$Triage12[i],newdat$Triage3[i],newdat$Triage4[i],
  newdat$Triage5[i],newdat$Lab[i],newdat$Xray[i],newdat$Other[i],
  newdat$LOS[i],newdat$LOSh[i],newdat$pregtriage[i],newdat$pnurse[i],
  newdat$pphysdecis[i],newdat$pregtriage0[i],newdat$pnurse0[i],
```

```
  newdat$lrat2[i],newdat$prat2[i],1);
}
colnames(longdat) <- c("id","Day","Ambulance","Triage12","Triage3",
"Triage4","Triage5","Lab","Xray","Other","LOS","LOSh","pregtriage",
"pnurse","pphysdecis","pregtriage0","pnurse0","lrat","prat","tlevel")
longdat <- as.data.frame(longdat)
#
```

The next code chunk runs homoscedastic and heteroscedastic log-ratio and probability-ratio regression models. The outputs for the heteroscedastic models also are displayed, verifying that the log-ratio and probability-ratio models give the same results. Note, however, that they have different global deviance and AIC values, due to the fact that the log-ratio and probability-ratio models are on different scales.

```
# Homoscedastic log-ratio model:
longmod1b <- lm(longdat$lrat ~ longdat$tlevel + longdat$LOSh
+ longdat$Triage12+longdat$Triage3+longdat$tlevel*longdat$Ambulance
+longdat$tlevel*pmax(longdat$Xray,longdat$Lab,longdat$Other))
summary(longmod1b)
#
# Homoscedastic probability-ratio model:
gmmod0 <- gamlss(formula = longdat$prat ~ longdat$tlevel + longdat$LOSh
+ longdat$Triage12+longdat$Triage3+longdat$tlevel*longdat$Ambulance
+longdat$tlevel*pmax(longdat$Xray,longdat$Lab,longdat$Other),
sigma.formula = ~ 1,  family = LOGITNO(mu.link = "logit",
sigma.link = "log"))
summary(gmmod0)
#
# Now we add predictors to the dispersion submodel.
# First, the probability-ratio model:
gmmod2 <- gamlss(formula = longdat$prat ~ longdat$tlevel + longdat$LOSh
```

```
+ longdat$Triage12+longdat$Triage3+longdat$tlevel*longdat$Ambulance
+longdat$tlevel*pmax(longdat$Xray,longdat$Lab,longdat$Other),
sigma.formula = ~ longdat$tlevel
+ pmax(longdat$Xray,longdat$Lab,longdat$Other) + longdat$Triage3,
family = LOGITNO(mu.link = "logit", sigma.link = "log"))
summary(gmmod2)
## Family:  c("LOGITNO", "Logit Normal")
##
## Call:  gamlss(formula = longdat$prat ~ longdat$tlevel + longdat$LOSh +
##     longdat$Triage12 + longdat$Triage3 + longdat$tlevel *
##     longdat$Ambulance + longdat$tlevel * pmax(longdat$Xray,
##     longdat$Lab, longdat$Other), sigma.formula = ~longdat$tlevel +
##     pmax(longdat$Xray, longdat$Lab, longdat$Other) +
##     longdat$Triage3, family = LOGITNO(mu.link = "logit",
##     sigma.link = "log"))
##
## Fitting method: RS()
##
## ------------------------------------------------------------------
## Mu link function:  logit
## Mu Coefficients:
##                                                      Estimate
## (Intercept)                                        -0.061707
## longdat$tlevel                                     -0.110086
## longdat$LOSh                                       -0.139530
## longdat$Triage12                                   -1.607249
## longdat$Triage3                                    -0.526839
## longdat$Ambulance                                  -0.831328
## pmax(longdat$Xray, longdat$Lab, longdat$Other)     -1.001119
```

```
## longdat$tlevel:longdat$Ambulance                                 0.459645
## longdat$tlevel:pmax(longdat$Xray, longdat$Lab, longdat$Other)  0.309571
##                                                                Std. Error
## (Intercept)                                                      0.078794
## longdat$tlevel                                                   0.109881
## longdat$LOSh                                                     0.008258
## longdat$Triage12                                                 0.165267
## longdat$Triage3                                                  0.065436
## longdat$Ambulance                                                0.099829
## pmax(longdat$Xray, longdat$Lab, longdat$Other)                   0.096904
## longdat$tlevel:longdat$Ambulance                                 0.139823
## longdat$tlevel:pmax(longdat$Xray, longdat$Lab, longdat$Other)   0.131481
##                                                              t value Pr(>|t|)
## (Intercept)                                                   -0.783  0.43366
## longdat$tlevel                                                -1.002  0.31656
## longdat$LOSh                                                 -16.896  < 2e-16
## longdat$Triage12                                              -9.725  < 2e-16
## longdat$Triage3                                               -8.051 1.61e-15
## longdat$Ambulance                                             -8.327  < 2e-16
## pmax(longdat$Xray, longdat$Lab, longdat$Other)               -10.331  < 2e-16
## longdat$tlevel:longdat$Ambulance                               3.287  0.00103
## longdat$tlevel:pmax(XXX)                                       2.354  0.01867
##
## (Intercept)
## longdat$tlevel
## longdat$LOSh                                                        ***
## longdat$Triage12                                                    ***
## longdat$Triage3                                                     ***
## longdat$Ambulance                                                   ***
```

```
## pmax(longdat$Xray, longdat$Lab, longdat$Other)                ***
## longdat$tlevel:longdat$Ambulance                              **
## longdat$tlevel:pmax(longdat$Xray, longdat$Lab, longdat$Other) *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## ----------------------------------------------------------------
## Sigma link function:  log
## Sigma Coefficients:
##                                              Estimate Std. Error t value
## (Intercept)                                   0.27873    0.03427   8.134
## longdat$tlevel                                0.03404    0.03580   0.951
## pmax(longdat$Xray, longdat$Lab, longdat$Other) -0.16184    0.03817  -4.240
## longdat$Triage3                              -0.12983    0.03935  -3.299
##                                              Pr(>|t|)
## (Intercept)                                  8.37e-16 ***
## longdat$tlevel                               0.341821
## pmax(longdat$Xray, longdat$Lab, longdat$Other) 2.37e-05 ***
## longdat$Triage3                              0.000991 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## ----------------------------------------------------------------
## No. of observations in the fit:  1580
## Degrees of Freedom for the fit:  13
##       Residual Deg. of Freedom:  1567
##                       at cycle:  3
##
## Global Deviance:     -2464.938
```

```
##             AIC:     -2438.938
##             SBC:     -2369.19
#
# Last, we reproduce it for the log-ratios:
lmmod2 <- gamlss(formula = longdat$lrat ~ longdat$tlevel + longdat$LOSh
+ longdat$Triage12+longdat$Triage3+longdat$tlevel*longdat$Ambulance
+longdat$tlevel*pmax(longdat$Xray,longdat$Lab,longdat$Other),
sigma.formula = ~ longdat$tlevel
+ pmax(longdat$Xray,longdat$Lab,longdat$Other) + longdat$Triage3,
family = NO(mu.link = "identity", sigma.link = "log"))
summary(lmmod2)
## Family:  c("NO", "Normal")
##
## Call:  gamlss(formula = longdat$lrat ~ longdat$tlevel + longdat$LOSh +
##     longdat$Triage12 + longdat$Triage3 + longdat$tlevel *
##     longdat$Ambulance + longdat$tlevel * pmax(longdat$Xray,
##     longdat$Lab, longdat$Other), sigma.formula = ~longdat$tlevel +
##     pmax(longdat$Xray, longdat$Lab, longdat$Other) +
##     longdat$Triage3, family = NO(mu.link = "identity",
##     sigma.link = "log"))
##
## Fitting method: RS()
##
## ------------------------------------------------------------------
## Mu link function:  identity
## Mu Coefficients:
##                                                          Estimate
## (Intercept)                                            -0.061707
## longdat$tlevel                                         -0.110086
```

```
## longdat$LOSh                                                    -0.139530
## longdat$Triage12                                                -1.607249
## longdat$Triage3                                                 -0.526839
## longdat$Ambulance                                               -0.831328
## pmax(longdat$Xray, longdat$Lab, longdat$Other)                  -1.001119
## longdat$tlevel:longdat$Ambulance                                 0.459645
## longdat$tlevel:pmax(longdat$Xray, longdat$Lab, longdat$Other)    0.309571
##                                                                 Std. Error
## (Intercept)                                                       0.078794
## longdat$tlevel                                                    0.109881
## longdat$LOSh                                                      0.008258
## longdat$Triage12                                                  0.165267
## longdat$Triage3                                                   0.065436
## longdat$Ambulance                                                 0.099829
## pmax(longdat$Xray, longdat$Lab, longdat$Other)                    0.096904
## longdat$tlevel:longdat$Ambulance                                  0.139823
## longdat$tlevel:pmax(longdat$Xray, longdat$Lab, longdat$Other)     0.131481
##                                                                  t value Pr(>|t|)
## (Intercept)                                                       -0.783  0.43366
## longdat$tlevel                                                    -1.002  0.31656
## longdat$LOSh                                                     -16.896  < 2e-16
## longdat$Triage12                                                  -9.725  < 2e-16
## longdat$Triage3                                                   -8.051 1.61e-15
## longdat$Ambulance                                                 -8.327  < 2e-16
## pmax(longdat$Xray, longdat$Lab, longdat$Other)                   -10.331  < 2e-16
## longdat$tlevel:longdat$Ambulance                                   3.287  0.00103
## longdat$tlevel:pmax(XXX)                                           2.354  0.01867
##
## (Intercept)
```

```
## longdat$tlevel
## longdat$LOSh                                                    ***
## longdat$Triage12                                                ***
## longdat$Triage3                                                 ***
## longdat$Ambulance                                               ***
## pmax(longdat$Xray, longdat$Lab, longdat$Other)                  ***
## longdat$tlevel:longdat$Ambulance                                **
## longdat$tlevel:pmax(longdat$Xray, longdat$Lab, longdat$Other) *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## ----------------------------------------------------------------
## Sigma link function:  log
## Sigma Coefficients:
##                                                Estimate Std. Error t value
## (Intercept)                                     0.27873    0.03427   8.134
## longdat$tlevel                                  0.03404    0.03580   0.951
## pmax(longdat$Xray, longdat$Lab, longdat$Other) -0.16184    0.03817  -4.240
## longdat$Triage3                                -0.12983    0.03935  -3.299
##                                                Pr(>|t|)
## (Intercept)                                    8.37e-16 ***
## longdat$tlevel                                 0.341821
## pmax(longdat$Xray, longdat$Lab, longdat$Other) 2.37e-05 ***
## longdat$Triage3                                0.000991 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## ----------------------------------------------------------------
## No. of observations in the fit:  1580
```

```
## Degrees of Freedom for the fit:  13
##       Residual Deg. of Freedom:  1567
##                       at cycle:  3
##
## Global Deviance:     4960.467
##             AIC:     4986.467
##             SBC:     5056.214
#
```

Emergency Department Example Continued: Dealing with Zeros

The code chunks in this section contain the logistic regression model for the zeros and the probability-ratio (logit-normal) model for the data-set with the zeros replaced by 0.01 and the nonzero values multiplied by the appropriate value to reconstitute the composition.

The first chunk reshapes the data-set in the same way as before, but this time the dependent variable is an indicator variable that is 1 when there is a zero and 0 when not. The logistic regression model code is shown at the end of this chunk.

```
# Form a new long version of the data
# that includes the zero cases.
# Make a vector of 0's and a vector of 1's for the Stage
# dummy variable:
xeros <- c(rep(0,894)); wons <- c(rep(1,894))
# Assemble the relevant variables:
zerodat1 <- cbind(id,Day,Ambulance,Triage12,Triage3,Triage4,Triage5,
pmax(Xray,Lab,Other),LOS,LOSh,pregtriage,pnurse,pphysdecis,pregtriage0,xeros)
zerodat2 <- cbind(id,Day,Ambulance,Triage12,Triage3,Triage4,Triage5,
pmax(Xray,Lab,Other),LOS,LOSh,pregtriage,pnurse,pphysdecis,pnurse0,wons)
colnames(zerodat1) = c("id","Day","Ambulance","Triage12","Triage3","Triage4",
"Triage5","treatmt","LOS","LOSh","pregtriage","pnurse",
```

```
"pphysdecis","zeros","stage")
colnames(zerodat2) = c("id","Day","Ambulance","Triage12","Triage3","Triage4",
"Triage5","treatmt","LOS","LOSh","pregtriage","pnurse",
"pphysdecis","zeros","stage")
zerodat1 <- as.data.frame(zerodat1)
zerodat2 <- as.data.frame(zerodat2)
# Concatenate:
zerodat <- rbind(zerodat1,zerodat2)
#
# Logistic regression model:
lmodn4 <- glm(zerodat$zeros ~ zerodat$Triage12 +
zerodat$stage*(zerodat$Triage3 + zerodat$Ambulance),
family = binomial(link = "logit")); summary(lmodn4)
#
```

The next chunk reshapes the data as before, but this time conducting "multiplicative" replacement of the zeros. The probability-ratio model code is at the end of this chunk.

```
# Substitute 0.01 for the 0's and rescale the other parts:
pregtriage[pphysdecis==1] <- 0.01
pnurse[pphysdecis==1] <- 0.01
pphysdecis[pphysdecis==1] <- 0.98
pregtriage[pnurse==0] <- 0.99*pregtriage[pnurse==0]
pphysdecis[pnurse==0] <- 0.99*pphysdecis[pnurse==0]
pnurse[pnurse==0] <- 0.01
pnurse[pregtriage==0] <- 0.99*pnurse[pregtriage==0]
pphysdecis[pregtriage==0] <- 0.99*pphysdecis[pregtriage==0]
pregtriage[pregtriage==0] <- 0.01
#
# Now, create the log-ratio variables.
```

```
lrat1 <- c(rep(NA,894)); lrat2 <- c(rep(NA,894))
lrat1 <- log(pregtriage/pphysdecis)
lrat2 <- log(pnurse/pphysdecis)
# Also create the probability-ratios:
prat1 <- c(rep(NA,894)); prat2 <- c(rep(NA,894))
prat1 <- pregtriage/(pregtriage+pphysdecis)
prat2 <- pnurse/(pnurse+pphysdecis)
# Put all of the relevant variables together and get rid of the NA's:
newdat <- as.data.frame(na.omit(cbind(id,Day,Ambulance,
Triage12,Triage3,Triage4,Triage5,Lab,Xray,Other,LOS,
LOSh,pregtriage,pnurse,pphysdecis,pregtriage0,pnurse0,
lrat1,lrat2,prat1,prat2)))
#
#
# Create a long version of the same data:
matsize <- 40*length(newdat[,1])
longdat <- matrix(c(rep(0,matsize)),
nrow = 2*length(newdat[,1]))
for (i in 1:894) {
  longdat[2*i-1,] <- c(newdat$id[i],newdat$Day[i],newdat$Ambulance[i],
  newdat$Triage12[i],newdat$Triage3[i],newdat$Triage4[i],
  newdat$Triage5[i],newdat$Lab[i],newdat$Xray[i],
  newdat$Other[i],newdat$LOS[i],newdat$LOSh[i],
  newdat$pregtriage[i],newdat$pnurse[i],newdat$pphysdecis[i],
  newdat$pregtriage0[i],newdat$pnurse0[i],newdat$lrat1[i],
  newdat$prat1[i],0);
  longdat[2*i,] <- c(newdat$id[i],newdat$Day[i],newdat$Ambulance[i],
  newdat$Triage12[i],newdat$Triage3[i],newdat$Triage4[i],
  newdat$Triage5[i],newdat$Lab[i],newdat$Xray[i],
```

```
  newdat$Other[i],newdat$LOS[i],newdat$LOSh[i],
  newdat$pregtriage[i],newdat$pnurse[i],newdat$pphysdecis[i],
  newdat$pregtriage0[i],newdat$pnurse0[i],newdat$lrat2[i],
  newdat$prat2[i],1);
}
colnames(longdat) <- c("id","Day","Ambulance","Triage12","Triage3","Triage4",
"Triage5","Lab","Xray","Other","LOS","LOSh","pregtriage",
"pnurse","pphysdecis","pregtriage0","pnurse0","lrat",
"prat","tlevel")
longdat <- as.data.frame(longdat)
#
#
# This is our final model:
gmmod2 <- gamlss(formula = longdat$prat ~ longdat$tlevel +
longdat$LOSh + longdat$Triage12+longdat$Triage3+
longdat$tlevel*longdat$Ambulance+
longdat$tlevel*pmax(longdat$Xray,longdat$Lab,longdat$Other),
sigma.formula = ~ longdat$tlevel +
pmax(longdat$Xray,longdat$Lab,longdat$Other) + longdat$Triage3,
family = LOGITNO(mu.link = "logit", sigma.link = "log"))
```

Employment Data Example

This example contains a copula model, so we provide a brief explanation of its role and nature here. The copula model's role in this analysis is to handle the dependencies among the probability-ratios. Because the primary focus in our analysis is on the marginal distribution models, the copula parameters are being treated as nuisance parameters.

As mentioned in the main text, we use a two-stage estimation procedure commonly employed in copula modeling. The marginal distributions are modeled first, and their estimates are then fed to the next step, which is applying their estimated

quantile functions to the data to create pseudo-observations whose marginals are uniformly distributed. A copula is then fitted to these pseudo-observations. This two-stage estimation procedure does not permit statistical inference about the copula parameters because the sampling error in the marginal models is not accounted for in the standard errors of the copula parameter estimates. However, because we are treating the copula parameters as nuisance parameters, this is sufficient for our purposes and it also is computationally efficient. If we wanted to test hypotheses about the dependency structure then we would need to employ the more computationally burdensome one-stage estimation procedure whereby all of the marginal distribution and copula parameters are estimated simultaneously.

There are several families of copulas in the literature, and we have chosen the t-copula because of its flexibility. A t-copula imposes a multivariate t-distribution structure on the uniform-marginal pseudo-observations created by the first step in generating a copula model, and then fits that to the resulting covariance structure while also estimating the degrees of freedom for the t-distribution.

This first code chunk loads the relevant libraries (including the `copula` package), reads the data-set, and prepares the data.

```
# Load the libraries:
library("betareg")
library("copula")
library("gamlss")
library(lmtest)
#
# Read the data file
fulldat <- read.csv("worldbankemploydata2015.csv", header = T)
# Create a version of the data with no NA's
pidat <- na.omit(fulldat)
attach(pidat)
#
```

```
# Extract the probability-ratios:
simout <- matrix(data = c(rep(0, 5*length(agpctfem))), ncol = 5)
simout[,1] <- agpctfem/(agpctfem + agpctmal)
simout[,2] <- indpctfem/(indpctfem + indpctmal)
simout[,3] <- serpctfem/(serpctfem + serpctmal)
simout[,4] <- (agpctfem + agpctmal)/(agpctfem + agpctmal
+ serpctfem + serpctmal)
simout[,5] <- (indpctfem + indpctmal)/(indpctfem + indpctmal
+ serpctfem + serpctmal)
#
# Extract the log-ratios:
loddout <- matrix(data = c(rep(0, 5*length(agpctfem))), ncol = 5)
loddout[,1] <- log(agpctfem/agpctmal)
loddout[,2] <- log(indpctfem/indpctmal)
loddout[,3] <- log(serpctfem/serpctmal)
loddout[,4] <- log((agpctfem + agpctmal)/(serpctfem + serpctmal))
loddout[,5] <- log((indpctfem + indpctmal)/(serpctfem
+ serpctmal))
#
```

Now we run the logit-normal models, starting with the conditional marginal distribution models and then combining that with a copula model. This next chunk runs the logit normal models for the five marginal distributions of the probability-ratios.

```
# Fit the marginal distributions.
# The logistic-normal models:
# gamma_{afm}
logitnoB1 <- gamlss(simout[,1] ~ log(GDPcap),
sigma.formula = ~ log(GDPcap),
family = LOGITNO(mu.link = "logit", sigma.link = "log"))
summary(logitnoB1)
```

```
# gamma_{ifm}
logitnoB2 <- gamlss(simout[,2] ~ log(GDPcap),
sigma.formula = ~ log(GDPcap),
family = LOGITNO(mu.link = "logit", sigma.link = "log"))
summary(logitnoB2)
# gamma_{sfm}
logitnoB3 <- gamlss(simout[,3] ~ log(GDPcap),
sigma.formula = ~ log(GDPcap),
family = LOGITNO(mu.link = "logit", sigma.link = "log"))
summary(logitnoB3)
# gamma_{as}
logitnoB4 <- gamlss(simout[,4] ~ log(GDPcap),
sigma.formula = ~ log(GDPcap),
family = LOGITNO(mu.link = "logit", sigma.link = "log"))
summary(logitnoB4)
# gamma_{is}
logitnoB5 <- gamlss(simout[,5] ~ log(GDPcap),
sigma.formula = ~ log(GDPcap),
family = LOGITNO(mu.link = "logit", sigma.link = "log"))
summary(logitnoB5)
#
```

The next chunk estimates a t-copula for the data, after generating the uniform-marginal pseudo-observations. The output from the copula model is displayed at the end of the chunk.

```
# Now set up the copula model:
## Form a quintivariate t-copula.
tCop1 <-
tCopula(c(0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1),
dim = 5, dispstr = "un", df.fixed = FALSE)
```

```
# Generate the uniform-marginal pseudo-observations for simout.
# First, get the conditional means from the logit normal parameters:
nmean1 <- exp(logitnoB1$mu.coefficients[1] +
logitnoB1$mu.coefficients[2]*log(GDPcap))/(1 +
exp(logitnoB1$mu.coefficients[1] +
logitnoB1$mu.coefficients[2]*log(GDPcap)))
nmean2 <- exp(logitnoB2$mu.coefficients[1] +
logitnoB2$mu.coefficients[2]*log(GDPcap))/(1 +
exp(logitnoB2$mu.coefficients[1] +
logitnoB2$mu.coefficients[2]*log(GDPcap)))
nmean3 <- exp(logitnoB3$mu.coefficients[1] +
logitnoB3$mu.coefficients[2]*log(GDPcap))/(1 +
exp(logitnoB3$mu.coefficients[1] +
logitnoB3$mu.coefficients[2]*log(GDPcap)))
nmean4 <- exp(logitnoB4$mu.coefficients[1] +
logitnoB4$mu.coefficients[2]*log(GDPcap))/(1 +
exp(logitnoB4$mu.coefficients[1] +
logitnoB4$mu.coefficients[2]*log(GDPcap)))
nmean5 <- exp(logitnoB5$mu.coefficients[1] +
logitnoB5$mu.coefficients[2]*log(GDPcap))/(1 +
exp(logitnoB5$mu.coefficients[1] +
logitnoB5$mu.coefficients[2]*log(GDPcap)))
# Get the conditional sigmas:
nsigma1 <- exp(logitnoB1$sigma.coefficients[1] +
logitnoB1$sigma.coefficients[2]*log(GDPcap))
nsigma2 <- exp(logitnoB2$sigma.coefficients[1] +
logitnoB2$sigma.coefficients[2]*log(GDPcap))
nsigma3 <- exp(logitnoB3$sigma.coefficients[1] +
logitnoB3$sigma.coefficients[2]*log(GDPcap))
```

```
nsigma4 <- exp(logitnoB4$sigma.coefficients[1] +
logitnoB4$sigma.coefficients[2]*log(GDPcap))
nsigma5 <- exp(logitnoB5$sigma.coefficients[1] +
logitnoB5$sigma.coefficients[2]*log(GDPcap))
# Then produce the uniform-marginals:
udatLN2 <- cbind(pLOGITNO(simout[, 1], nmean1, nsigma1),
pLOGITNO(simout[, 2], nmean2, nsigma2),
pLOGITNO(simout[, 3], nmean3, nsigma3),
pLOGITNO(simout[, 4], nmean4, nsigma4),
pLOGITNO(simout[, 5], nmean5, nsigma5))
## Now to fit the copula:
copfitudatLN2 <- fitCopula(tCop1, udatLN2,
start = c(0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 5))
#
# Examine the output:
summary(copfitudatLN2)
## Call: fitCopula(copula, data = data, start = ..1)
## Fit based on "maximum pseudo-likelihood"
## and 178 5-dimensional observations.
## t-copula, dim. d = 5
##        Estimate Std. Error
## rho.1   0.30155      0.166
## rho.2   0.38731      0.205
## rho.3   0.40560      0.164
## rho.4  -0.02264      0.456
## rho.5   0.56839      0.206
## rho.6   0.14718      0.494
## rho.7   0.18280      0.878
## rho.8  -0.03952      0.676
```

```
## rho.9  -0.22842      0.444
## rho.10  0.18920      0.323
## df     13.39894         NA
## The maximized loglikelihood is 106.1
## Optimization converged
#
#
```

Now we run beta regression models, starting with the conditional marginal distribution models and then combining that with a copula model. This next chunk runs the beta regression models for the five marginal distributions of the probability-ratios.

```
# Here are the beta-regression models.
# gamma_{afm}
fitbeta1 <- gamlss(simout[,1] ~ log(GDPcap),
sigma.formula = ~ log(GDPcap),
family = BE(mu.link = "logit", sigma.link = "log"))
summary(fitbeta1)
# gamma_{ifm}
fitbeta2 <- gamlss(simout[,2] ~ log(GDPcap),
sigma.formula = ~ log(GDPcap),
family = BE(mu.link = "logit", sigma.link = "log"))
summary(fitbeta2)
# gamma_{sfm}
fitbeta3 <- gamlss(simout[,3] ~ log(GDPcap),
sigma.formula = ~ log(GDPcap),
family = BE(mu.link = "logit", sigma.link = "log"))
summary(fitbeta3)
# gamma_{as}
fitbeta4 <- gamlss(simout[,4] ~ log(GDPcap),
sigma.formula = ~ log(GDPcap),
```

```
family = BE(mu.link = "logit", sigma.link = "log"))
summary(fitbeta4)
# gamma_{is}
fitbeta5 <- gamlss(simout[,5] ~ log(GDPcap),
sigma.formula = ~ log(GDPcap),
family = BE(mu.link = "logit", sigma.link = "log"))
summary(fitbeta5)
#
```

The next chunk estimates a t-copula for the data, after generating the uniform-marginal pseudo-observations from the beta regression parameters. The output from the copula model is displayed at the end of the chunk.

```
## Form a quintivariate t-copula.
tCop1 <-
tCopula(c(0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1),
dim = 5, dispstr = "un", df.fixed = FALSE)
# Generate the uniform-marginal pseudo-observations for simout.
# Get the conditional means from the beta regression parameters:
cmean1 <- exp(fitbeta1$mu.coefficients[1] +
fitbeta1$mu.coefficients[2]*log(GDPcap))/(1 +
exp(fitbeta1$mu.coefficients[1] +
fitbeta1$mu.coefficients[2]*log(GDPcap)))
cmean2 <- exp(fitbeta2$mu.coefficients[1] +
fitbeta2$mu.coefficients[2]*log(GDPcap))/(1 +
exp(fitbeta2$mu.coefficients[1] +
fitbeta2$mu.coefficients[2]*log(GDPcap)))
cmean3 <- exp(fitbeta3$mu.coefficients[1] +
fitbeta3$mu.coefficients[2]*log(GDPcap))/(1 +
exp(fitbeta3$mu.coefficients[1] +
fitbeta3$mu.coefficients[2]*log(GDPcap)))
```

```
cmean4 <- exp(fitbeta4$mu.coefficients[1] +
fitbeta4$mu.coefficients[2]*log(GDPcap))/(1 +
exp(fitbeta4$mu.coefficients[1] +
fitbeta4$mu.coefficients[2]*log(GDPcap)))
cmean5 <- exp(fitbeta5$mu.coefficients[1] +
fitbeta5$mu.coefficients[2]*log(GDPcap))/(1 +
exp(fitbeta5$mu.coefficients[1] +
fitbeta5$mu.coefficients[2]*log(GDPcap)))
# Get the conditional sigmas:
csigma1 <- exp(fitbeta1$sigma.coefficients[1] +
fitbeta1$sigma.coefficients[2]*log(GDPcap))
csigma2 <- exp(fitbeta2$sigma.coefficients[1] +
fitbeta2$sigma.coefficients[2]*log(GDPcap))
csigma3 <- exp(fitbeta3$sigma.coefficients[1] +
fitbeta3$sigma.coefficients[2]*log(GDPcap))
csigma4 <- exp(fitbeta4$sigma.coefficients[1] +
fitbeta4$sigma.coefficients[2]*log(GDPcap))
csigma5 <- exp(fitbeta5$sigma.coefficients[1] +
fitbeta5$sigma.coefficients[2]*log(GDPcap))
# Then produce the uniform-marginals:
udatBE2 <- cbind(pBE(simout[, 1], cmean1, csigma1),
pBE(simout[, 2], cmean2, csigma2),
pBE(simout[, 3], cmean3, csigma3),
pBE(simout[, 4], cmean4, csigma4),
pBE(simout[, 5], cmean5, csigma5))
## Now to fit the copula:
copfitudatBE2 <- fitCopula(tCop1, udatBE2,
start = c(0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 5))
# Examine the output:
```

```
summary(copfitudatBE2)
## Call: fitCopula(copula, data = data, start = ..1)
## Fit based on "maximum pseudo-likelihood"
## and 178 5-dimensional observations.
## t-copula, dim. d = 5
##        Estimate Std. Error
## rho.1   0.28710      0.137
## rho.2   0.35634      0.243
## rho.3   0.42376      0.505
## rho.4  -0.01575      0.284
## rho.5   0.54709      0.169
## rho.6   0.11448      0.454
## rho.7   0.19462      0.452
## rho.8  -0.08864      0.281
## rho.9  -0.24688      0.234
## rho.10  0.15437      0.781
## df     12.90622         NA
## The maximized loglikelihood is 104.8
## Optimization converged
#
```

Finally, the next code chunk runs the "hybrid" copula model and displays its output, also comparing its estimates with the sample correlations for the pseudo-observations.

```
# We already have a quintivariate t-copula,
# so we don't need to recreate tCop1.
# Get the hybrid copula.
# First generate the uniform-marginals.
# We use the beta models for the first three and the
# logit-normal models for the last two.
```

```
udatH <- cbind(pBE(simout[, 1], cmean1, csigma1),
pBE(simout[, 2], cmean2, csigma2),
pBE(simout[, 3], cmean3, csigma3),
pLOGITNO(simout[, 4], nmean4, nsigma4),
pLOGITNO(simout[, 5], nmean5, nsigma5))
## Now to fit the copula:
copfitudatH <- fitCopula(tCop1, udatH,
start = c(0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 5))
# Examine the output:
summary(copfitudatH)
## Call: fitCopula(copula, data = data, start = ..1)
## Fit based on "maximum pseudo-likelihood"
## and 178 5-dimensional observations.
## t-copula, dim. d = 5
##          Estimate Std. Error
## rho.1   0.2776999      0.153
## rho.2   0.3475900      0.166
## rho.3   0.4079034      0.091
## rho.4  -0.0004034      0.248
## rho.5   0.5449609      0.161
## rho.6   0.1358443      0.198
## rho.7   0.2132017      0.447
## rho.8  -0.0612346      0.298
## rho.9  -0.2257242      0.244
## rho.10  0.1916581      0.168
## df     11.1692551         NA
## The maximized loglikelihood is 102.5
## Optimization converged
#
```

```
# Compare with the sample correlations:
cordat <- cor(udatH); cordat
##             [,1]      [,2]       [,3]       [,4]        [,5]
## [1,]  1.00000000 0.2170443  0.2782280  0.4028022 -0.05722231
## [2,]  0.21704430 1.0000000  0.4379587  0.1336812  0.26818290
## [3,]  0.27822804 0.4379587  1.0000000 -0.1359744 -0.23279972
## [4,]  0.40280222 0.1336812 -0.1359744  1.0000000  0.18745278
## [5,] -0.05722231 0.2681829 -0.2327997  0.1874528  1.00000000
# Compute absolute differences:
absdif <- 1- cordat
absdif[2,1] <- abs(summary(copfitudatH)$coef[1] - cordat[2,1])
absdif[3,1] <- abs(summary(copfitudatH)$coef[2] - cordat[3,1])
absdif[4,1] <- abs(summary(copfitudatH)$coef[3] - cordat[4,1])
absdif[5,1] <- abs(summary(copfitudatH)$coef[4] - cordat[5,1])
absdif[3,2] <- abs(summary(copfitudatH)$coef[5] - cordat[3,2])
absdif[4,2] <- abs(summary(copfitudatH)$coef[6] - cordat[4,2])
absdif[5,2] <- abs(summary(copfitudatH)$coef[7] - cordat[5,2])
absdif[4,3] <- abs(summary(copfitudatH)$coef[8] - cordat[4,3])
absdif[5,3] <- abs(summary(copfitudatH)$coef[9] - cordat[5,3])
absdif[5,4] <- abs(summary(copfitudatH)$coef[10] - cordat[5,4])
absdif
##           [,1]        [,2]       [,3]        [,4]      [,5]
## [1,] 0.000000000 0.782955702 0.72177196 0.597197782 1.0572223
## [2,] 0.060655573 0.000000000 0.56204129 0.866318837 0.7318171
## [3,] 0.069361976 0.107002214 0.00000000 1.135974439 1.2327997
## [4,] 0.005101201 0.002163091 0.07473985 0.000000000 0.8125472
## [5,] 0.056818869 0.054981156 0.00707556 0.004205272 0.0000000
```

Probability Judgment Example

ILR Model

This first code chunk reads the original data-file and generates a data-frame with the ILR compositional variates. The purpose of including this chunk is simply to illustrate how that is done in the R compositions package.

```
Long <- read.csv("long_IPCC.csv", header=TRUE, sep=",")
library(compositions)
library(cdfquantreg)
library(lme4)
#
# Cheat the 0's and 1's away from the boundaries
Long$b3 <- scaleTR(Long$b3)
Long$b2 <- scaleTR(Long$b2)
Long$b1 <- scaleTR(Long$b1)
#
#Form Composition
#Make Proportions Composition Long data
long_prop = (Long[,c("b1","b2","b3")])
#Turn proportions to comp class
long_comp = acomp(long_prop, parts=c("b1","b3","b2"))
#
Y_ilr <- ilr(long_comp)
VPE <- factor(Long$VPE)
Val <- factor(Long$Valence)
Cond <- factor(Long$Cond)
ID <- Long$ID
# Get the first variate
cID <- 1
```

```
Y1_ilr <- Y_ilr[,1]
long_longilr1 <- data.frame(Y1_ilr,VPE,Val,Cond,ID,cID)
# Get the ssecond variate
cID <- 2
Y1_ilr <- Y_ilr[,2]
long_longilr2 <- data.frame(Y1_ilr,VPE,Val,Cond,ID,cID)
# Cmbine them
long_longilr <- rbind(long_longilr1,long_longilr2)
# Create the reverse indicator variable
long_longilr$dID <- long_longilr$cID - 1
long_longilr$dIDr <- 1 - long_longilr$dID
#
```

The next code-chunk runs the mixed regressions for the ILR model, displaying the effects and coefficients for the fixed effects, and plots the predicted versus sample ILR variates. Output also is displayed here.

```
# Read the compositions data (saved from the preceding chunk
# but with dummy variables appended to it):
long_tot <- read.csv("ILR_long_tot.csv", header = TRUE)
# Run the mixed regression model:
#
mod1 <- lmer(Y1_ilr ~ (narrow + treat + wide + ivpe + ival)*dIDr
+ (1 + dIDr|ID), data=long_tot)
summary(mod1)
## Linear mixed model fit by REML ['lmerMod']
## Formula: Y1_ilr ~ (narrow + treat + wide + ivpe + ival) * dIDr + (1 +
##     dIDr | ID)
##    Data: long_tot
##
## REML criterion at convergence: 5683.4
```

```
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -5.1468 -0.4304 -0.0271  0.4407  4.6387
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  ID       (Intercept) 0.5268   0.7258
##           dIDr        2.1950   1.4816   -0.69
##  Residual             0.9739   0.9869
## Number of obs: 1784, groups:  ID, 223
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)  0.70625    0.11847   5.962
## narrow      -0.70391    0.17252  -4.080
## treat       -0.18181    0.15280  -1.190
## wide        -0.07732    0.16703  -0.463
## ivpe        -0.23209    0.06608  -3.512
## ival        -0.26547    0.06608  -4.017
## dIDr        -2.58360    0.21361 -12.095
## narrow:dIDr  0.28145    0.32192   0.874
## treat:dIDr  -0.03911    0.28512  -0.137
## wide:dIDr   -0.35470    0.31167  -1.138
## ivpe:dIDr    1.16984    0.09346  12.517
## ival:dIDr    0.75360    0.09346   8.064
##
# The correlations of the fixed-effects coefficients
# are not shown here.
```

```
#
# Check the correlation between fitted values and data:
cor(long_tot$Y1_ilr,fitted.values(mod1))
# By variate:
# log(b2/(b1 + b3))
cor(long_tot$Y1_ilr[long_tot$dID==1],
fitted.values(mod1)[long_tot$dID==1])
plot(long_tot$Y1_ilr[long_tot$dID==1],
fitted.values(mod1)[long_tot$dID==1], pch = 16,
xlab = "log(b2/(b1 + b3))", ylab = "fitted values")
# log(b3/b1)
cor(long_tot$Y1_ilr[long_tot$dID==0],
fitted.values(mod1)[long_tot$dID==0])
plot(long_tot$Y1_ilr[long_tot$dID==0],
fitted.values(mod1)[long_tot$dID==0], pch = 16,
xlab = "log(b3/b1)", ylab = "fitted values")
#
# Gather the coefficients:
coefs <- fixef(mod1)
coefsd0 <- rbind(coefs[1],coefs[2],coefs[3],coefs[4],coefs[5],coefs[6])
coefsd1 <- rbind(coefs[7],coefs[8],coefs[9],coefs[10],coefs[11],coefs[12])
coefsd01 <- cbind(coefsd0,coefsd0+coefsd1); coefsd01
#
# Run the reverse model to get the standard errors
# and significance-tests for the
# coefficients pertaining to the b3/b1 contrast
mod1r <- lmer(Y1_ilr ~ (narrow + treat + wide + ivpe + ival)*dID
+ (1 + dID|ID), data=long_tot)
summary(mod1r)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: Y1_ilr ~ (narrow + treat + wide + ivpe + ival) * dID + (1 + dID |
##     ID)
##    Data: long_tot
##
## REML criterion at convergence: 5683.4
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -5.1468 -0.4304 -0.0271  0.4407  4.6387
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  ID       (Intercept) 1.2286   1.1084
##           dID         2.1949   1.4815   -0.88
##  Residual             0.9739   0.9869
## Number of obs: 1784, groups:  ID, 223
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept) -1.87735    0.15758 -11.914
## narrow      -0.42245    0.23850  -1.771
## treat       -0.22092    0.21123  -1.046
## wide        -0.43202    0.23090  -1.871
## ivpe         0.93775    0.06608  14.190
## ival         0.48813    0.06608   7.386
## dID          2.58360    0.21360  12.095
## narrow:dID  -0.28145    0.32191  -0.874
## treat:dID    0.03911    0.28511   0.137
```

```
## wide:dID     0.35470    0.31166   1.138
## ivpe:dID    -1.16984    0.09346 -12.517
## ival:dID    -0.75360    0.09346  -8.064
##
# The correlations of the fixed-effects coefficients
# are not shown here.
#
```

The final code chunk in this subsection illustrates the ILR model effects, both for the ILR variates and the composition parts. It does this by displaying marginal means that are relevant to the model.

```
# Now, get the main effects for each of the conditions
# and the inverse-transforms
# Condition:
logcond <- matrix(c(
mean(long_tot$Y1_ilr[long_tot$Cond=="CONTROL" &
long_tot$dID==0]),
mean(long_tot$Y1_ilr[long_tot$Cond=="NARROW" &
long_tot$dID==0]),
mean(long_tot$Y1_ilr[long_tot$Cond=="TREATMENT" &
long_tot$dID==0]),
mean(long_tot$Y1_ilr[long_tot$Cond=="WIDE" &
long_tot$dID==0]),
mean(long_tot$Y1_ilr[long_tot$Cond=="CONTROL" &
long_tot$dID==1]),
mean(long_tot$Y1_ilr[long_tot$Cond=="NARROW" &
long_tot$dID==1]),
mean(long_tot$Y1_ilr[long_tot$Cond=="TREATMENT" &
long_tot$dID==1]),
mean(long_tot$Y1_ilr[long_tot$Cond=="WIDE" &
```

```
long_tot$dID==1])),nrow = 4)
colnames(logcond) <- c("ILR2","ILR1")
rownames(logcond) <-c("control","narrow","treatment","wide")
logcond
# Apply the inverse transform:
probcond <- ilrInv(logcond)
colnames(probcond) <- c("b1","b3","b2")
rownames(probcond) <-c("control","narrow","treatment","wide")
probcond
#
# Valence
logval <- matrix(c(mean(long_tot$Y1_ilr[long_tot$Val=="neg" &
long_tot$dID==0]),mean(long_tot$Y1_ilr[long_tot$Val=="pos" &
long_tot$dID==0]),mean(long_tot$Y1_ilr[long_tot$Val=="neg" &
long_tot$dID==1]),mean(long_tot$Y1_ilr[long_tot$Val=="pos" &
long_tot$dID==1])), ncol = 2)
colnames(logval) <- c("ILR2","ILR1")
rownames(logval) <-c("negative","positive")
logval
# Apply the inverse transform:
probval <- ilrInv(logval)
colnames(probval) <- c("b1","b3","b2")
rownames(probval) <-c("negative","positive")
probval
#
# Extremity
logext <- matrix(c(mean(long_tot$Y1_ilr[long_tot$VPE=="vl" &
long_tot$dID==0]),mean(long_tot$Y1_ilr[long_tot$VPE=="l" &
long_tot$dID==0]),mean(long_tot$Y1_ilr[long_tot$VPE=="vl" &
```

```
long_tot$dID==1]),mean(long_tot$Y1_ilr[long_tot$VPE=="l" &
long_tot$dID==1])), ncol = 2)
colnames(logext) <- c("ILR2","ILR1")
rownames(logext) <-c("very","no very")
logext
# Apply the inverse transform:
probext <- ilrInv(logext)
colnames(probext) <- c("b1","b3","b2")
rownames(probext) <-c("very","no very")
probext
#
```

Stick-Breaking Model

We now turn to the stick-breaking model. Using a prefabricated compositional data file, this next chunk runs the stick-breaking regression models, displaying the effects and coefficients for the fixed effects, and examines the correlations between the predicted versus sample variates.

```
# Load libraries and read data
library(lme4)
long_tot <- read.csv("SB_long_tot.csv", header = TRUE)
# Run the mixed regression model:
#
mod1 <- lmer(Y1_sb ~ (narrow + treat + wide + ivpe + ival)*dID + (1 + dID|ID),
data=long_tot)
summary(mod1)
## Linear mixed model fit by REML ['lmerMod']
## Formula: Y1_sb ~ (narrow + treat + wide + ivpe + ival) * dID + (1 + dID |
##     ID)
##    Data: long_tot
```

```
##
## REML criterion at convergence: 6628.4
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.6903 -0.3985 -0.0043  0.4231  5.0133
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  ID       (Intercept) 0.6137   0.7834
##           dID         1.9401   1.3929   -0.02
##  Residual             1.7025   1.3048
## Number of obs: 1784, groups:  ID, 223
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept) -0.78324    0.14073  -5.565
## narrow      -1.00147    0.20040  -4.997
## treat       -0.34838    0.17748  -1.963
## wide        -0.27294    0.19401  -1.407
## ivpe         0.30191    0.08738   3.455
## ival        -0.17569    0.08738  -2.011
## dID         -1.87173    0.22490  -8.323
## narrow:dID   0.40403    0.32842   1.230
## treat:dID    0.03594    0.29087   0.124
## wide:dID    -0.33803    0.31796  -1.063
## ivpe:dID     1.02427    0.12357   8.289
## ival:dID     0.86602    0.12357   7.008
##
```

```
# The correlations of the fixed-effects coefficients
# are not shown here.
#
# Check the correlation between fitted values and data:
cor(long_tot$Y1_sb,fitted.values(mod1))
# By variate:
# log(b2/(b1 + b3))
cor(long_tot$Y1_sb[long_tot$dID==0],
fitted.values(mod1)[long_tot$dID==0])
# log(b3/b1)
cor(long_tot$Y1_sb[long_tot$dID==1],
fitted.values(mod1)[long_tot$dID==1])
#
# Gather the coefficients
coefs <- fixef(mod1)
coefsd0 <- rbind(coefs[1],coefs[2],coefs[3],coefs[4],coefs[5],coefs[6])
coefsd1 <- rbind(coefs[7],coefs[8],coefs[9],coefs[10],coefs[11],coefs[12])
coefsd01 <- cbind(coefsd0,coefsd0+coefsd1); coefsd01
#
# Running the reverse model to get the
# standard errors and significance-tests for the
# coefficients pertaining to the b3/b1 contrast:
mod1r <- lmer(Y1_sb ~ (narrow + treat + wide + ivpe + ival)*dIDr + (1 + dIDr|ID),
data=long_tot)
summary(mod1r)
## Linear mixed model fit by REML ['lmerMod']
## Formula: Y1_sb ~ (narrow + treat + wide + ivpe + ival) * dIDr + (1 + dIDr |
##     ID)
##    Data: long_tot
```

```
##
## REML criterion at convergence: 6628.4
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.6903 -0.3985 -0.0043  0.4231  5.0133
##
## Random effects:
##  Groups   Name        Variance Std.Dev. Corr
##  ID       (Intercept) 2.518    1.587
##           dIDr        1.940    1.393    -0.87
##  Residual             1.703    1.305
## Number of obs: 1784, groups:  ID, 223
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept) -2.65497    0.22161 -11.980
## narrow      -0.59744    0.33729  -1.771
## treat       -0.31243    0.29872  -1.046
## wide        -0.61097    0.32654  -1.871
## ivpe         1.32618    0.08738  15.178
## ival         0.69033    0.08738   7.901
## dIDr         1.87173    0.22489   8.323
## narrow:dIDr -0.40403    0.32841  -1.230
## treat:dIDr  -0.03594    0.29087  -0.124
## wide:dIDr    0.33803    0.31795   1.063
## ivpe:dIDr   -1.02427    0.12357  -8.289
## ival:dIDr   -0.86602    0.12357  -7.008
##
```

```
# The correlations of the fixed-effects coefficients
# are not shown here.
#
```

Finally, we observe the effects via conditional means.

```
# Now, get the main effects for each of the conditions
# and the inverse-transforms
# First, define the inverse transform:
invsb <- function(c1,c2){
  b1 <- 1/((1+exp(c1))*(1+exp(c2)))
  b2 <- exp(c2)/(1+exp(c2))
  b3 <- exp(c1)/((1+exp(c1))*(1+exp(c2)))
  return (c(b1,b2,b3))
}
#
# Condition:
logcond <- matrix(c(mean(long_tot$Y1_sb[long_tot$Cond=="CONTROL" &
long_tot$dID==0]),mean(long_tot$Y1_sb[long_tot$Cond=="NARROW" &
long_tot$dID==0]),mean(long_tot$Y1_sb[long_tot$Cond=="TREATMENT" &
long_tot$dID==0]),mean(long_tot$Y1_sb[long_tot$Cond=="WIDE" &
long_tot$dID==0]),mean(long_tot$Y1_sb[long_tot$Cond=="CONTROL" &
long_tot$dID==1]),mean(long_tot$Y1_sb[long_tot$Cond=="NARROW" &
long_tot$dID==1]),mean(long_tot$Y1_sb[long_tot$Cond=="TREATMENT" &
long_tot$dID==1]),mean(long_tot$Y1_sb[long_tot$Cond=="WIDE" &
long_tot$dID==1])),nrow = 4)
colnames(logcond) <- c("SB2","SB1")
rownames(logcond) <-c("control","narrow","treatment","wide")
logcond
# Apply the inverse transform:
probcond <- matrix(c(rep(0,12)),ncol = 3)
```

```
for (i in 1:4) {
  probcond[i,] <- invsb(logcond[i,2],logcond[i,1])
}
colnames(probcond) <- c("b1","b2","b3")
rownames(probcond) <-c("control","narrow","treatment","wide")
probcond
#
# Valence
logval <- matrix(c(mean(long_tot$Y1_sb[long_tot$Val=="neg" &
long_tot$dID==0]),mean(long_tot$Y1_sb[long_tot$Val=="pos" &
long_tot$dID==0]),mean(long_tot$Y1_sb[long_tot$Val=="neg" &
long_tot$dID==1]),mean(long_tot$Y1_sb[long_tot$Val=="pos" &
long_tot$dID==1])), ncol = 2)
colnames(logval) <- c("SB2","SB1")
rownames(logval) <-c("negative","positive")
logval
# Apply the inverse transform:
probval <- matrix(c(rep(0,6)),ncol = 3)
for (i in 1:2) {
  probval[i,] <- invsb(logval[i,2],logval[i,1])
}
colnames(probval) <- c("b1","b2","b3")
rownames(probval) <-c("negative","positive")
probval
#
# Extremity
logext <- matrix(c(mean(long_tot$Y1_sb[long_tot$VPE=="vl" &
long_tot$dID==0]),mean(long_tot$Y1_sb[long_tot$VPE=="l" &
long_tot$dID==0]),mean(long_tot$Y1_sb[long_tot$VPE=="vl" &
```

```
long_tot$dID==1]),mean(long_tot$Y1_sb[long_tot$VPE=="l" &
long_tot$dID==1])), ncol = 2)
colnames(logext) <- c("SB2","SB1")
rownames(logext) <-c("very","no very")
logext
# Apply the inverse transform:
probext <- matrix(c(rep(0,6)),ncol = 3)
for (i in 1:2) {
  probext[i,] <- invsb(logext[i,2],logext[i,1])
}
colnames(probext) <- c("b1","b2","b3")
rownames(probext) <-c("very","no very")
probext
#
```

References

Comas Cufí, M. & Thió i Fernández de Henestrosa, S. (2011). Codapack 2.0: a stand-alone, multi-platform compositional software [Computer software manual]. Universitat de Girona. Departament d'Informàtica i Matemàtica Aplicada.

Graffelman, J., Pawlowsky-Glahn, V., Egozcue, J. J. & Buccianti, A. (2018). Exploration of geochemical data with compositional canonical biplots. *Journal of Geochemical Exploration*, *194*, 120–133. Retrieved from https://doi.org/10.1016/j.gexplo.2018.07.014 doi: 10.1016/j.gexplo.2018.07.014

Palarea-Albaladejo, J. & Martín-Fernández, J. A. (2015). zcompositions—r package for multivariate imputation of left-censored data under a compositional approach. *Chemometrics and Intelligent Laboratory Systems*, *143*, 85–96.

Templ, M., Hron, K. & Filzmoser, P. (2011). robcompositions: an r-package for robust statistical analysis of compositional data [Computer software manual]. CRAN.

Tsagris, M. & Athineou, G. (2016). Compositional-package 3. *Compositional Data Analysis*, 3.

Van den Boogaart, K. G. & Tolosana-Delgado, R. (2008). "compositions": a unified r package to analyze compositional data. *Computers & Geosciences*, *34*(4), 320–338.