Response Surface Analysis: an Tutorial for Examining Linear and Curvilinear Effects

Readme: R Code Walkthrough to Response Surface Analysis

Purpose: In this readme file, we will consider practical procedures on how to conduct the RSA in R and demonstrate the usage with a concrete example. For this, we divided the file into seven steps.

Step 1 - Installing and loading packages

You need to download and install both R and RStudio (Desktop version) on your computer to use this tutorial. It is important that you install R first and then install RStudio. Go to: http://cloud.r-project.org/, click the download link for R according to your operating system (Windows, macOS or Linux) to install R. To install RStudio, go to: https://rstudio.com/products/rstudio/download/ and click on the download link for the appropriate version.

After it got R and RStudio, we can install the packages. A way to install a package is by typing `install.packages("tidyverse")` in the console pane of RStudio and pressing Return/Enter on your keyboard. The # symbol is used when you want to put comments. Note you must include the quotation marks around the name of the package, as in the example of packages used in our analysis below:

```
install.packages("tidyverse") #install tidyverse package
install.packages("RSA") #install RSA package
```

Recall that after you've installed a package, you need to "load it." In other words, you need to "open it." We do this by using the `library()`command

```
library(tidyverse) #Load tidyverse package
library(RSA) #Load RSA package
```

Step 2 – Load dataset from desktop

You can imported the CSV files interactively using combination functions read_csv () and file.choose (). For this tutorial we provide the dataset.csv file. The data was extracted from https://osf.io/meprk/ e we renamed the dataset for use it in this tutorial[1].

```
df <- read.csv(file.choose()) #read "dataset.csv" from desktop
```

Step 3 – Centering predictor variables

Centering is the rescaling of predictors by subtracting the mean. This function is used to group-mean center the predictors. Grand mean centering subtracts the grand mean of the

[1] X = female predictor variable, Y = male predictor variable, Z_f = female outcome variable have been renamed in this tutorial Y = supplier dependence predictor variable, X = buyer dependence predictor variable, Z_b = buyer satisfaction outcome variable

predictor using the mean from the full sample. In this step you center *x* and *y* on grand mean across both variables.

```
grand.M <- mean(c(df$X, df$Y)) # center x and y on grand mean across both var
iables
df$Y.c <- df$Y-grand.M
df$X.c <- df$X-grand.M
```

Step 4 – Run RSA function

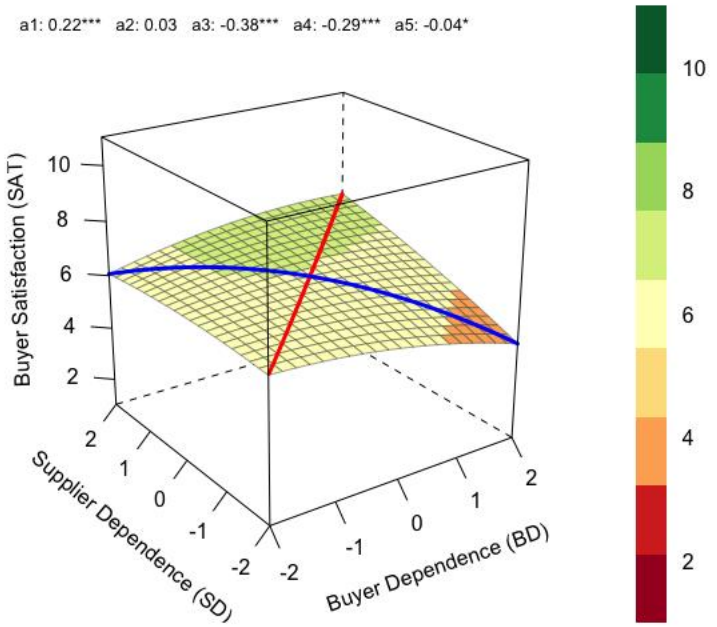
Run the `RSA()` function to obtain: (1) percent of matches and mismatches, (2) the polynomial model and, (3) four RSA coefficients (a_1 - a_4) the four a coefficients.

```
rsa.model <-  RSA(formula = Z ~ X.c*Y.c, #Specify which variable is outcome a
nd which are the two predictors
                  data    = df,   #Specify data
                  center  = ,   #DON'T FOGET TO ASSIGN THIS ABOVE!
                  scale   = TRUE,      #Keep scales in original units
                  na.rm   = TRUE,       #Remove missing variables
                  out.rm  = TRUE,       #Default: remove outliers
                  models  = c("full"),  #Use the full polynomial equation (X
+ Y + x^2 + XY + Y^2)
                  missing = "listwise") #listwise deletion of missing values
to be comparable to traditional methods

## [1] "Computing polynomial model (full) ..."

summary(rsa.model)  #This displays:

## RSA output (package version 0.10.1)
## ============================================
##
## Are there discrepancies in the predictors (with respect to numerical congr
uence)?

## (A cutpoint of |Δz| > 0.5 is used)
##
## ----------------------------
## Congruence
##      X < Y Congruence      X > Y
##       "40%"       "12%"       "48%"
##
## Is the full polynomial model significant?
## ----------------------------
## Test on model significance: R^2 = 0.273, p <.001
##
##
## Number of observations: n = 300
## ----------------------------
##
##
## Regression coefficients for model <full>
## ----------------------------
##           label    est    se ci.lower ci.upper   beta   pvalue sig
## Z~1          b0  6.488 0.120    6.253    6.722  3.706  p <.001 ***
## Z~X.c        b1 -0.078 0.042   -0.160    0.005 -0.086 p = .065   †
## Z~Y.c        b2  0.298 0.046    0.209    0.388  0.329  p <.001 ***
## Z~X.c2       b3 -0.088 0.014   -0.115   -0.062 -0.301  p <.001 ***
```

```
## Z~X.c_Y.c     b4  0.162 0.024    0.114    0.209  0.331  p <.001 ***
## Z~Y.c2        b5 -0.044 0.017   -0.078   -0.011 -0.127 p = .009  **
##
##
## Surface tests (a1 to a5) for model <full>
## ---------------------------
##   label    est    se ci.lower ci.upper   pvalue sig
## 1    a1  0.221 0.061    0.101    0.340  p <.001 ***
## 2    a2  0.029 0.031   -0.031    0.090 p = .345
## 3    a3 -0.376 0.063   -0.500   -0.252  p <.001 ***
## 4    a4 -0.295 0.035   -0.363   -0.226  p <.001 ***
## 5    a5 -0.044 0.021   -0.086   -0.002 p = .038   *
##
## a1: Linear additive effect on line of congruence? YES
## a2: Is there curvature on the line of congruence? NO
## a3: Is the ridge shifted away from the LOC? YES
## a4: Is there a general effect of incongruence? YES
##
##
## Location of stationary point for model <full>
## ---------------------------
## X.c = -3.927; Y.c = -3.812; predicted Z = 6.072
##
##
## Principal axes for model <full>
## ---------------------------
##                    label    est    se ci.lower ci.upper   pvalue sig
## Intercept of 1. PA   p10  1.331 0.288    0.766    1.896  p <.001 ***
## Slope of 1. PA       p11  1.310 0.168    0.981    1.639  p <.001 ***
## Intercept of 2. PA   p20 -6.811 6.151  -18.866    5.245 p = .268
## Slope of 2. PA       p21 -0.763 0.098   -0.955   -0.572  p <.001 ***
  --> Lateral shift of first PA from LOC at point (0; 0): C1 =  -0.576
  --> Lateral shift of second PA from LOC at point (0; 0): C2 =  28.796
# 1) Percent of matches & mismatches
# 2) The polynomial model (i.e., unstandardized slopes and R^2)
# 3) Four RSA coefficients (a1 - a4)
```

Step 5 – Run RSA function

To create the 3D graph, we use the `plot()` function package. The parameters generated by the polynomial regression are displayed graphically. Then, the researcher can view relationships between combinations of the two predictor variables and the outcome variable in a three-dimensional space.

```
plot(rsa.model,           #Fitted model
     type = "3d",         #For those who have difficulty interpreting 3-d fig
ures try "interactive" which will let you rotate the figure
     xlab = "Buyer Dependence (BD)",
     ylab = "Supplier Dependence (SD)",
     zlab = "Buyer Satisfaction (SAT)",
     surface = "predict", #Response surface is based on predicted values of o
utcome
     rotation = list(x = -63, y = 32, z = 15),
     legend  = TRUE,     #TRUE displays color legend
     param   = TRUE,      #Display RSA parameters
     coefs   = FALSE,      #Display polynomial coefficients
     axesStyles = list(LOC = list(lty="dotted",  lwd=3, col= "red"), LOIC= li
```

```
st(lty="solid",  lwd=3, col="blue")),
     points = FALSE, #Display raw scatter points
     axes    = c("LOC", "LOIC"), #Display line of congruence and line of inco
ngruence
     project = NULL,      #TRUE displays projections onto the bottom of the p
lot
     hull    = FALSE,     #TRUE displays a bag plot on the surface
     xlim    = c(-2,2),   #Set range for x axis to fit the range of value for
predictor 1
     ylim    = c(-2,2)   #set range for y axis to fit the range of value for
predictor 2
)
```