CodeReport ST841 TAA Project

Andrew Hollis, Xinyu Zhang, and Qiang Heng

10/26/2020

```
knitr::opts_chunk$set(echo = TRUE)
library(knitr) # kable
library(readr) # read_delim
library(dplyr) # manipulate data
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(Amelia) # missing value imputation
```

```
## Loading required package: Rcpp

## ##
## ## Amelia II: Multiple Imputation
## ## (Version 1.7.6, built: 2019-11-24)
## ## Copyright (C) 2005-2020 James Honaker, Gary King and Matthew Blackwell
## ## Refer to http://gking.harvard.edu/amelia/ for more information
## ##
```

Introduction

This document is meant to explain and demonstrate the implementation of all analysis performed on the TAA dataset.

Data Transformations

We took the original data and performed several important transformations.

First we read in the initial data file:

```
init_TAA_data<-read.delim("initial_data.txt")
init_TAA_data$X3_firmSize <- as.numeric(init_TAA_data$Q26_num) # Insure Q26_num is actually numeric
init_TAA_data$Q29_num<-as.numeric(as.character(init_TAA_data$Q29_num)) #Insure Q29_num is actually nume
```

```
## Warning: Ç¿ÖÆ¸Ä±ä¹ý³ÌÖÐ²úÉúÁËNA
```

Preview the missing value

```r
missmap(init_TAA_data)
```

The above plot shows how the missing value is distributed among the data. The X-axis represent the variables that have missing value, and the y-axis shows the observation.

For some observation such as 1, 2, and 5, we could see there are too much missing values, thus it might need further consideration that if we should include those data with most variables missing.

For the variables from left to right, the missing proportion for each variable is decreasing, which need our further attention to select the proper variable for analysis.

We then remove the variables that are not as interesting for analysis, and these include the data from Q42, Q43, Q44, and Q45 which are demographic data about the respondent. We will also compute new summary score for question 6 from the survey, so we remove this as well.

```r
removal_list<-c("Q42","Q43","Q44","Q45","Q6_score")
init_TAA_data<-init_TAA_data[, !(names(init_TAA_data) %in%removal_list)]
```

Dealing with Likert Scale Items (Drew)

Questions 5, 10, 23, 30, 31, and 32 are composed of several 5-item Likert scale sub-questions. Each of these questions represents a single underlying conceptual variable. For instance, the 6 subquestions from question 5 together measure the extent to which a business considered various business objectives as they intially considered whether they would try TAA tools. We would like to create a one-dimensional measure that represents the level of this initial TAA consideration for a particular company.

Nonlinear principal components analysis is a methodology that allows us to transform several Likert scale items into the one-dimensional summary metric that captures the maximal amount of variability from the original data. (Linting et al. 2007)

Below, we use R to perform the nonlinear principal components analysis:

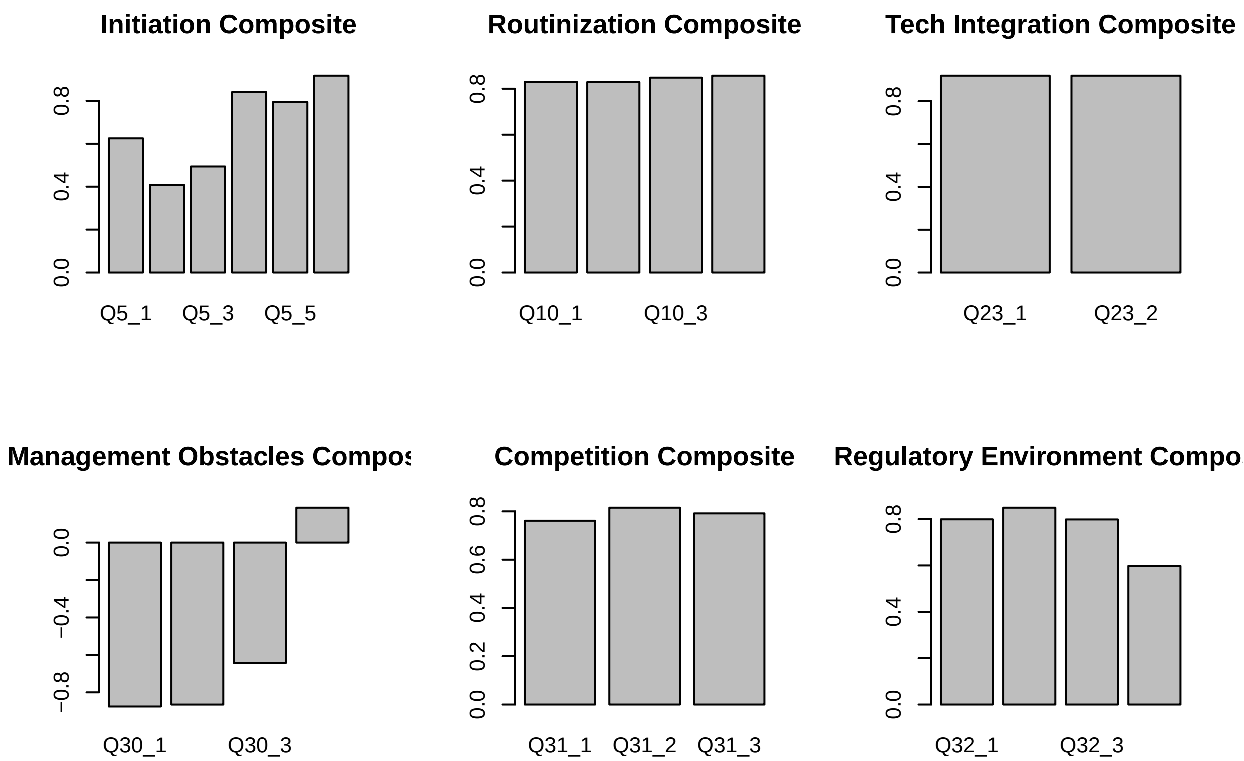
```
library(Gifi)
##Q5
#Set Aside Data for initiation
init_data_names<-c("Q5_1","Q5_2","Q5_3","Q5_4","Q5_5","Q5_6")
init_data<-init_TAA_data[init_data_names]

#Code Missing Values
init_data<-apply(init_data,2,function(x){ifelse(x==6,NA,x)})

#Impute, transform, do PCA
pca_init<-princals(init_data,ndim=1,missing = "a",degrees=2)
init_score<-pca_init$objectscores[,1]

##Q10
#Set Aside Data for routinization
rout_data_names<-c("Q10_1","Q10_2","Q10_3","Q10_4")
rout_data<-init_TAA_data[rout_data_names]

#Code Missing Values
rout_data<-apply(rout_data,2,function(x){ifelse(x==6,NA,x)})

#Impute, transform, do PCA
pca_rout<-princals(rout_data,ndim=1,missing = "a",degrees=2)
rout_score<-pca_rout$objectscores[,1]

##Q23
#Set Aside Data for routinization
integ_data_names<-c("Q23_1","Q23_2")
integ_data<-init_TAA_data[integ_data_names]

#Code Missing Values
integ_data<-apply(integ_data,2,function(x){ifelse(x==6,NA,x)})

#Impute, transform, do PCA
pca_integ<-princals(integ_data,ndim=1,missing = "a",degrees=2)
integ_score<-pca_integ$objectscores[,1]

##Q30
#Set Aside Data for management obstacles
manag_data_names<-c("Q30_1","Q30_2","Q30_3","Q30_4")
manag_data<-init_TAA_data[manag_data_names]

#Code Missing Values
manag_data<-apply(manag_data,2,function(x){ifelse(x==6,NA,x)})

#Impute, transform, do PCA
pca_manag<-princals(manag_data,ndim=1,missing = "a",degrees=2)
manag_score<-pca_manag$objectscores[,1]

##Q31
```

```r
#Set Aside Data for management obstacles
comp_data_names<-c("Q31_1","Q31_2","Q31_3")
comp_data<-init_TAA_data[comp_data_names]

#Code Missing Values
comp_data<-apply(comp_data,2,function(x){ifelse(x==6,NA,x)})

#Impute, transform, do PCA
pca_comp<-princals(comp_data,ndim=1,missing = "a",degrees=2)
comp_score<-pca_comp$objectscores[,1]

##Q32
#Set Aside Data for Government Regulation
gov_data_names<-c("Q32_1","Q32_2","Q32_3","Q32_4")
gov_data<-init_TAA_data[gov_data_names]

#Code Missing Values
gov_data<-apply(gov_data,2,function(x){ifelse(x==6,NA,x)})

#Impute, transform, do PCA
pca_gov<-princals(gov_data,ndim=1,missing = "a",degrees=2)
gov_score<-pca_gov$objectscores[,1]

#Add Nonlinear PCA Scores to data
init_TAA_data$init_composite<-init_score
init_TAA_data$rout_composite<-rout_score
init_TAA_data$integrate_composite<-integ_score
init_TAA_data$manag_composite<-manag_score
init_TAA_data$comp_composite<-comp_score
init_TAA_data$gov_composite<-gov_score
str(init_TAA_data)
```

```
## 'data.frame':    68 obs. of  61 variables:
##  $ Q5_1              : int  6 1 4 5 5 4 3 3 3 4 ...
##  $ Q5_2              : int  6 5 4 5 2 1 1 4 4 3 ...
##  $ Q5_3              : int  6 5 4 4 3 3 5 4 4 4 ...
##  $ Q5_4              : int  6 5 5 5 5 5 5 5 4 4 ...
##  $ Q5_5              : int  6 5 5 4 5 5 5 5 3 5 ...
##  $ Q5_6              : int  6 5 5 5 5 5 5 5 5 5 ...
##  $ Q6box_1           : int  NA NA 1 NA 1 1 1 NA 1 1 ...
##  $ Q6box_2           : int  NA NA NA 2 2 NA NA NA NA 2 ...
##  $ Q6box_3           : int  NA NA NA NA NA NA 3 NA NA NA ...
##  $ Q6box_4           : int  NA 4 4 4 NA 4 NA NA NA NA ...
##  $ Q10_1             : int  2 3 3 3 3 3 5 3 3 3 ...
##  $ Q10_2             : int  2 3 3 3 3 3 4 3 3 3 ...
##  $ Q10_3             : int  1 1 3 2 1 1 3 1 1 2 ...
##  $ Q10_4             : int  1 1 3 2 1 1 3 1 1 2 ...
##  $ Q35_ERP           : int  1 1 1 1 1 1 1 0 0 1 ...
##  $ Q40_TAA           : int  0 1 1 1 1 1 1 0 0 0 ...
##  $ Q36_ITcount       : int  3 2 3 3 NA 3 0 2 2 3 ...
##  $ Q37_ITassist      : int  1 0 3 0 1 1 2 1 1 3 ...
##  $ Q38_Taxcount      : int  23 23 40 12 100 200 55 19 15 45 ...
##  $ Q39_TAAexpert     : num  1 2.5 10 2 2 40 2 1 4 3 ...
##  $ Q22_box1          : int  1 1 1 1 1 1 1 1 1 1 ...
```

```
##  $ Q22_box2          : int  NA 2 2 NA 2 2 NA NA NA 2 ...
##  $ Q22_box3          : int  3 NA 3 NA 3 3 3 NA NA NA ...
##  $ Q22_box4          : int  4 4 4 NA NA NA NA NA NA NA ...
##  $ Q22_box5          : int  NA NA 5 5 5 5 NA NA NA 5 ...
##  $ Q22_box6          : int  NA NA NA 6 NA NA NA NA NA NA ...
##  $ Q22_box7          : int  NA NA 7 NA NA NA 7 NA 7 7 ...
##  $ Q22_box8          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ Q22_box9          : logi  NA NA NA NA NA NA ...
##  $ Q22_box10         : int  NA NA NA NA 10 NA NA NA NA NA ...
##  $ Q23_1             : int  4 3 3 1 2 3 4 2 3 2 ...
##  $ Q23_2             : int  4 3 3 1 1 2 4 1 1 2 ...
##  $ Q26               : int  6 6 6 6 6 6 6 6 6 6 ...
##  $ Q24_box1          : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ Q24_box2          : int  NA NA 2 2 2 2 NA NA 2 NA ...
##  $ Q24_box3          : int  NA NA 3 3 3 3 NA NA 3 NA ...
##  $ Q24_box4          : int  4 4 4 4 4 4 4 4 4 4 ...
##  $ Q27               : int  2 2 3 3 3 3 3 2 3 3 ...
##  $ Q29               : int  1 1 4 6 6 4 3 2 2 6 ...
##  $ Q41               : int  1 1 3 6 NA 4 1 2 2 NA ...
##  $ Q30_1             : int  5 1 3 4 4 3 1 4 4 3 ...
##  $ Q30_2             : int  4 1 2 2 3 3 1 5 3 3 ...
##  $ Q30_3             : int  4 3 2 4 5 4 1 3 5 3 ...
##  $ Q30_4             : int  NA NA NA NA NA 4 5 NA NA NA ...
##  $ Q31_1             : int  3 5 3 4 1 3 5 1 2 3 ...
##  $ Q31_2             : int  5 1 4 4 4 3 5 4 5 3 ...
##  $ Q31_3             : int  1 5 3 4 4 3 5 4 5 3 ...
##  $ Q32_1             : int  2 1 4 4 5 3 3 4 5 5 ...
##  $ Q32_2             : int  6 1 4 4 5 2 1 4 5 3 ...
##  $ Q32_3             : int  3 1 5 3 5 3 1 5 5 3 ...
##  $ Q32_4             : int  6 1 5 3 5 1 3 4 3 6 ...
##  $ Q26_num           : int  115000 9000 25000 28000 90000 60000 15000 5600 26000 70000 ...
##  $ Q29_num           : num  0 0.02 0.33 0.58 0.67 0.3 0.2 0.14 0.08 0.55 ...
##  $ Q41_num           : num  0 0 0.25 0.65 NA 0.3 0.01 0.1 0.08 NA ...
##  $ X3_firmSize       : num  115000 9000 25000 28000 90000 60000 15000 5600 26000 70000 ...
##  $ init_composite    : num  -1.5414 0.0756 0.5785 0.6462 0.3255 ...
##  $ rout_composite    : num  -0.9368 -0.9235 0.0819 -0.038 -0.9235 ...
##  $ integrate_composite: num  2.379 -0.182 -0.182 -0.758 -0.758 ...
##  $ manag_composite   : num  -0.864 2.7594 0.0495 -0.4805 -0.4805 ...
##  $ comp_composite    : num  -0.0581 0.9617 -0.9414 -0.9414 -0.9414 ...
##  $ gov_composite     : num  -0.197 -2.954 1.633 0.753 1.633 ...
```

```
#Create Loadings Plots
par(mfrow=c(2,3))
barplot(pca_init$loadings[,1],main="Initiation Composite")
barplot(pca_rout$loadings[,1],main="Routinization Composite")
barplot(pca_integ$loadings[,1],main="Tech Integration Composite")
barplot(pca_manag$loadings[,1],main="Management Obstacles Composite")
barplot(pca_comp$loadings[,1],main="Competition Composite")
barplot(pca_gov$loadings[,1],main="Regulatory Environment Composite")
```




```r
#Remove Original Likert Scale Data
removal_list<-c("Q5_1","Q5_2","Q5_3","Q5_4","Q5_5","Q5_6","Q10_1","Q10_2","Q10_3","Q10_4","Q23_1","Q23_
init_TAA_data<-init_TAA_data[, !(names(init_TAA_data) %in%removal_list)]
```

Combining Weighted Variables (Drew)

Q6 and Q22 have survey respondents check several boxes to indicate the level of TAA technology they have adopted and the specific types of TAA they adopted. For Q6, if respondents check the first box, they have adopted emerging TAA technology, if they check the second box they have adopted intermediate TAA technology, and if they check the third box they have adopted advanced TAA technology. We assign a weight of 1 to checkbox 1, 2 to checkbox 2, and 3 to checkbox 3. We create the single adoption score for each company by adding the weights and dividing by the maximum possible cumulative weight, 6.

Similarly, for Q22, if a weight of 0 is assigned to the first box, 1 to the second box, 2 for the 3rd, 4th, 5th, and 10th boxes, 3 for the 6th box, 4 for the 7th box, and 5 for the 8th and 9th boxes. As with the Q6 response, we form a single score by adding the weights and dividing by the maximum potential score, 26.

We standardize both new scores to have zero meana and standard deviation of 1.

We carry out this weighting in R below:

```r
#Transform 4's
init_TAA_data[2,9]=3
init_TAA_data[16,7]=1

#Form Adoption Score
adopt_data<-init_TAA_data[c("Q6box_1","Q6box_2","Q6box_3")]
adopt_comp<-apply(adopt_data,1,sum,na.rm=TRUE)/6
```

```
init_TAA_data$adoption_score<-adopt_comp
init_TAA_data$adoption_score<-scale(init_TAA_data$adoption_score)

#Technology Readiness/Capability Score
TAA_capability_data<-init_TAA_data[c("Q22_box1","Q22_box2","Q22_box3","Q22_box4","Q22_box5","Q22_box6",

TAA_capability_score<-rep(0,nrow(TAA_capability_data))
max_score=1+2*4+3+4+5*2
for(i in 1:length(TAA_capability_score)){
  score=0
  if(!is.na(TAA_capability_data[i,1])){
    score=score+0
  }
  if(!is.na(TAA_capability_data[i,2])){
    score=score+1
  }
  if(!is.na(TAA_capability_data[i,3])){
    score=score+2
  }
  if(!is.na(TAA_capability_data[i,4])){
    score=score+2
  }
  if(!is.na(TAA_capability_data[i,5])){
    score=score+2
  }
  if(!is.na(TAA_capability_data[i,6])){
    score=score+3
  }
  if(!is.na(TAA_capability_data[i,7])){
    score=score+4
  }
  if(!is.na(TAA_capability_data[i,8])){
    score=score+5
  }
  if(!is.na(TAA_capability_data[i,9])){
    score=score+5
  }
  if(!is.na(TAA_capability_data[i,10])){
    score=score+2
  }
  TAA_capability_score[i]=score/max_score
}
init_TAA_data$TAA_capability<-TAA_capability_score
init_TAA_data$TAA_capability<-scale(init_TAA_data$TAA_capability)
```

Creating Binary Variable for Q24

Q24 measures the global extent of a company. We transform this variable into a binary variable that is 0 if the company only has domestic offices and 1 if the company has foreign offices.

```
dom_int_data<-init_TAA_data[c("Q24_box1","Q24_box2","Q24_box3","Q24_box4")]
dom_int<-ifelse(is.na(dom_int_data[,4]),0,1) #Create binary variable
init_TAA_data$Domestic_International<-dom_int
NA_index=apply(is.na(init_TAA_data[c("Q24_box1","Q24_box2","Q24_box3","Q24_box4")]),1,all) #Determine w
```

```
init_TAA_data$Domestic_International[NA_index] = NA #Assign missingness pattern
```

Data Imputation (Xinyu)

Here, the data is renamed to prepare for the future analysis with a clearer variable

```
# names(init_TAA_data)
TAA_data <- init_TAA_data[-c(1, 2, 5), c(32:41)]
oldnames <- names(TAA_data)
newnames <- c("X3_firmSize" , "Y1_initialization", "Y3_routinization", "X2_integration", "X5_manag", "X6
names(TAA_data) <- newnames
rbind(oldnames, newnames)
```

```
##          [,1]          [,2]                  [,3]
## oldnames "X3_firmSize" "init_composite"    "rout_composite"
## newnames "X3_firmSize" "Y1_initialization" "Y3_routinization"
##          [,4]                   [,5]              [,6]
## oldnames "integrate_composite" "manag_composite" "comp_composite"
## newnames "X2_integration"      "X5_manag"        "X6_compete"
##          [,7]            [,8]              [,9]
## oldnames "gov_composite" "adoption_score" "TAA_capability"
## newnames "X7_gov"        "Y2_adoption"    "X1_readiness"
##          [,10]
## oldnames "Domestic_International"
## newnames "X4_global"
```

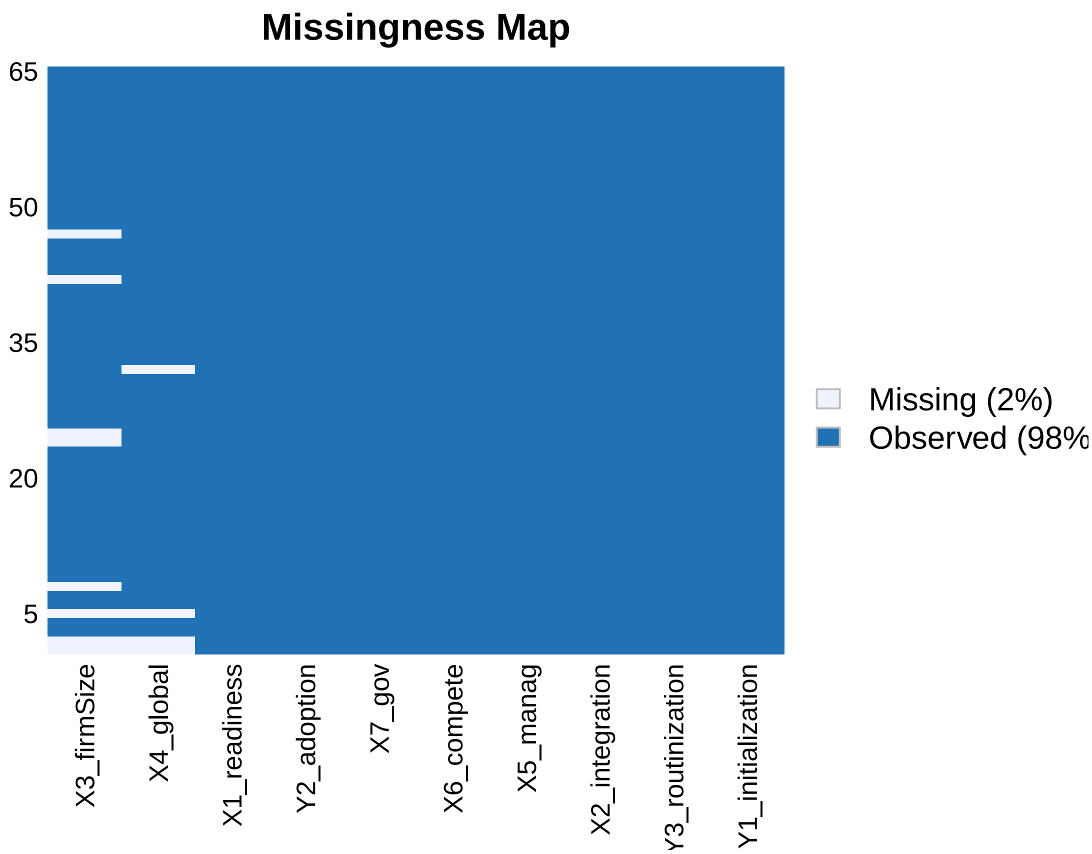
Using str(), we can see how many observations and how many variables are there in the data set.

```
TAA_data$X4_global <- as.factor(TAA_data$X4_global)
str(TAA_data)
```

```
## 'data.frame':    65 obs. of  10 variables:
##  $ X3_firmSize      : num  25000 28000 60000 15000 5600 26000 70000 16000 60000 20000 ...
##  $ Y1_initialization: num  0.578 0.646 0.251 0.185 0.513 ...
##  $ Y3_routinization : num  0.0819 -0.038 -0.9235 1.2726 -0.9235 ...
##  $ X2_integration   : num  -0.182 -0.758 -0.537 2.379 -0.758 ...
##  $ X5_manag         : num  0.0495 -0.4805 -0.1689 4.1609 -0.5452 ...
##  $ X6_compete       : num  -0.941 -0.941 -0.941 1.845 -0.941 ...
##  $ X7_gov           : num  1.633 0.753 -1.139 -1.43 1.563 ...
##  $ Y2_adoption      : num [1:65, 1] -0.492 0.205 -0.492 1.598 -1.188 ...
##  $ X1_readiness     : num [1:65, 1] 1.758 0.146 0.146 0.415 -1.197 ...
##  $ X4_global        : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
```

In this data set, we have three response variables (dependent variable), named as Y_1,Y_2, and Y_3, while other seven predictors (independent variable) named as $X_1, \cdots, X_7$. There are 65 observations since three of them have too much missing value and has been ignored for the following analysis.

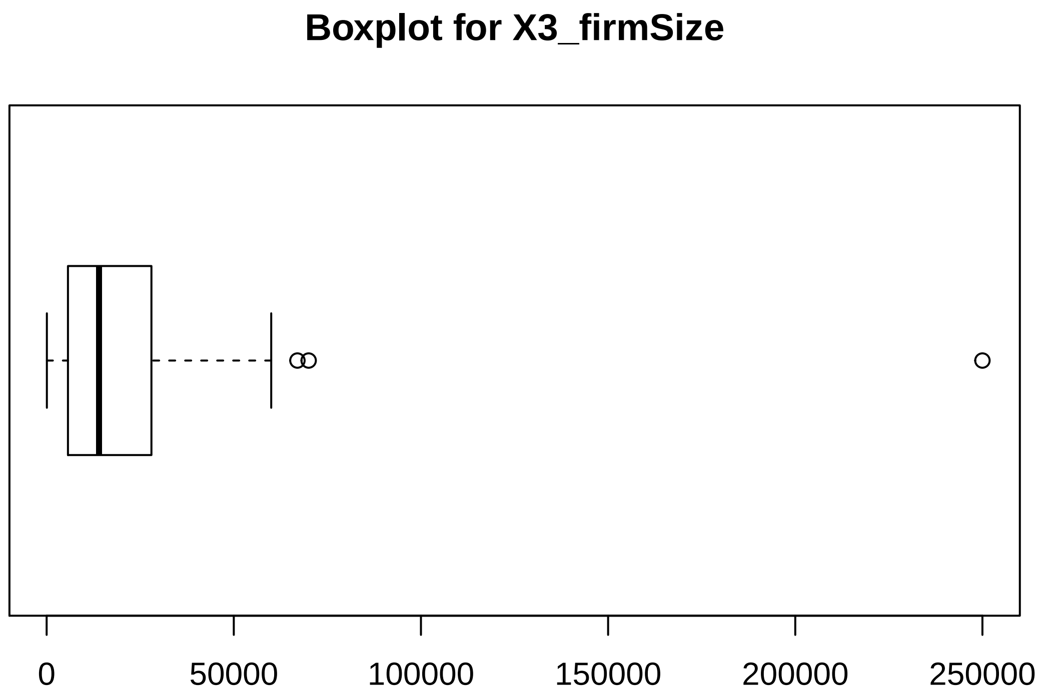
```
missmap(TAA_data)
```



Missing Value imputation in Amelia

Here the log transformation has been applied to the X3_firmSize since the range for the firm size is too large, and a log transformation would help to suppress the extreme large value of this variable.

```
boxplot(TAA_data$X3_firmSize, main="Boxplot for X3_firmSize", horizontal=T)
```



Besides, since the global size is a categorical variable (nominal variable), we want to specify this in the following function.

```
m = 5 # number of simulated datsets to create # See definition of m in ?amelia()
TAA_data_amelia <- amelia(x = TAA_data, logs="X3_firmSize", noms="X4_global", m = m) #
```

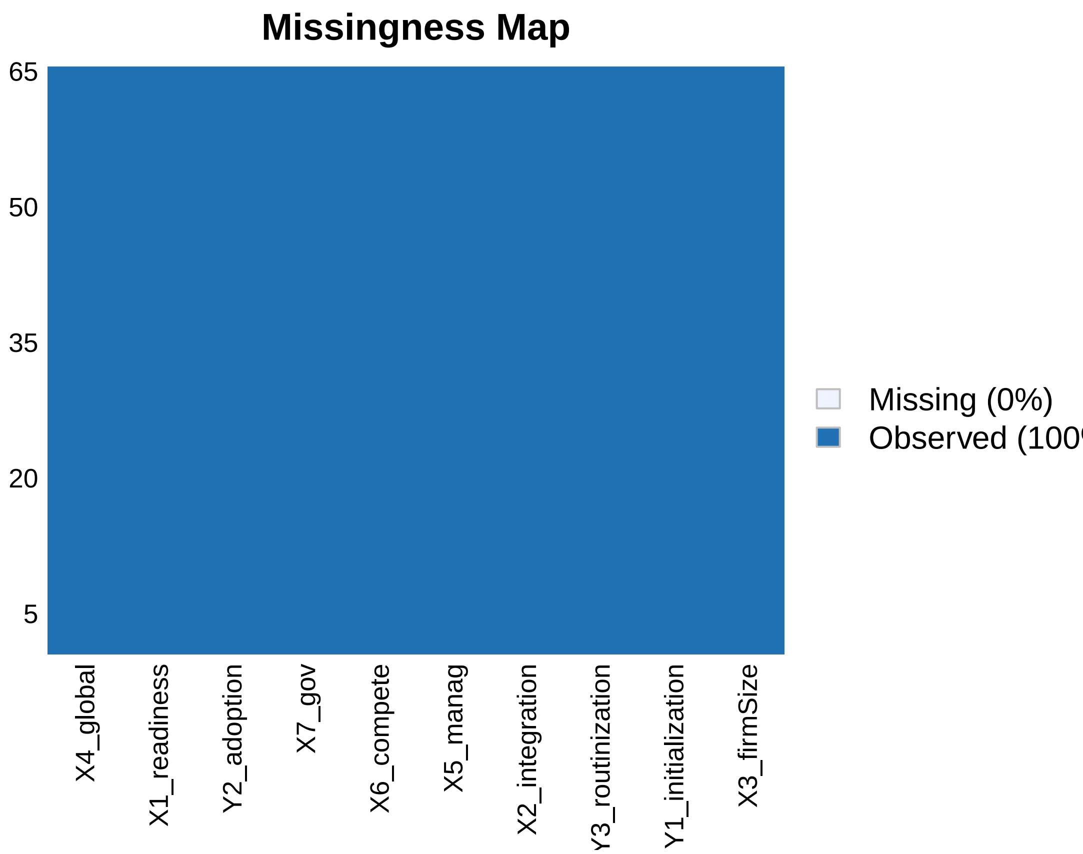
```
## -- Imputation 1 --
##
##   1  2  3  4  5  6  7  8  9 10 11
##
## -- Imputation 2 --
##
##   1  2  3  4  5  6  7  8
##
## -- Imputation 3 --
##
##   1  2  3  4  5  6
##
## -- Imputation 4 --
##
##   1  2  3  4  5  6  7
##
## -- Imputation 5 --
##
##   1  2  3  4  5
```

```r
str(TAA_data_amelia$imputations$imp1)
```

```
## 'data.frame':    65 obs. of  10 variables:
##  $ X3_firmSize      : num  25000 28000 60000 15000 5600 26000 70000 16000 60000 20000 ...
##  $ Y1_initialization: num  0.578 0.646 0.251 0.185 0.513 ...
##  $ Y3_routinization : num  0.0819 -0.038 -0.9235 1.2726 -0.9235 ...
##  $ X2_integration   : num  -0.182 -0.758 -0.537 2.379 -0.758 ...
##  $ X5_manag         : num  0.0495 -0.4805 -0.1689 4.1609 -0.5452 ...
##  $ X6_compete       : num  -0.941 -0.941 -0.941 1.845 -0.941 ...
##  $ X7_gov           : num  1.633 0.753 -1.139 -1.43 1.563 ...
##  $ Y2_adoption      : num [1:65, 1] -0.492 0.205 -0.492 1.598 -1.188 ...
##  $ X1_readiness     : num [1:65, 1] 1.758 0.146 0.146 0.415 -1.197 ...
##  $ X4_global        : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
```

```r
TAA_data_impute <- TAA_data
# Average the imputations between different simulated datasets
col_index = which(names(TAA_data_amelia$imputations$imp1) %in% c("X3_firmSize", "X4_global"))
for( col in col_index){
  temp=numeric()
  for (i in 1:m){
    temp = cbind(temp, TAA_data_amelia$imputations[[i]][,col])
  }
  TAA_data_impute[,col] = apply(temp, 1, mean)
}

TAA_data_impute$X4_global <- round((TAA_data_impute$X4_global))
```

```r
missmap(TAA_data_impute)
```



Now we can see there are no missing values in the data set, and the further regression analysis can be done. Let's look at the summary of the missing value imputations.

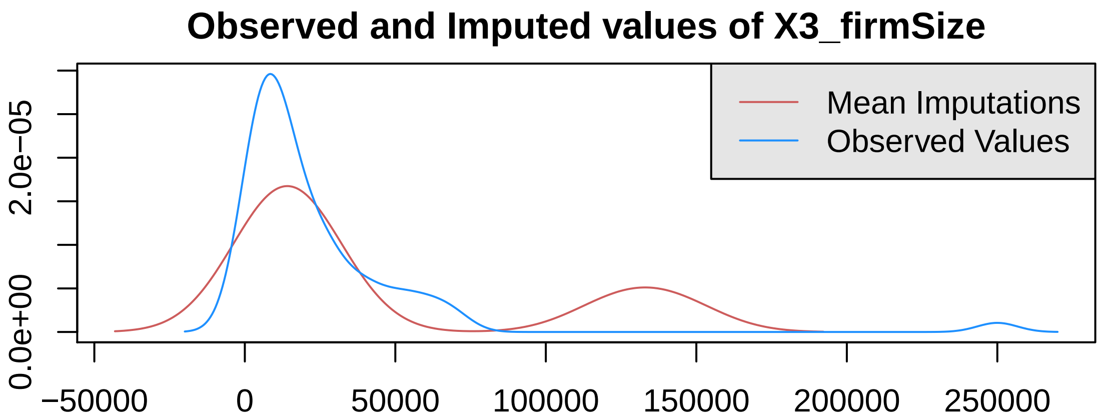
```
summary(TAA_data_impute)
```

```
##    X3_firmSize     Y1_initialization  Y3_routinization   X2_integration
##   Min.   :    65   Min.   :-5.42871   Min.   :-1.41257   Min.   :-0.75756
##   1st Qu.:  6586   1st Qu.:-0.05625   1st Qu.:-0.92351   1st Qu.:-0.75756
##   Median : 14000   Median : 0.32550   Median : 0.05474   Median :-0.53722
##   Mean   : 26253   Mean   : 0.01754   Mean   : 0.04283   Mean   :-0.02214
##   3rd Qu.: 28000   3rd Qu.: 0.57846   3rd Qu.: 0.47586   3rd Qu.: 0.32497
##   Max.   :250000   Max.   : 0.65275   Max.   : 4.12964   Max.   : 3.56297
##     X5_manag          X6_compete             X7_gov
##   Min.   :-0.86401   Min.   :-0.9413836   Min.   :-2.95425
##   1st Qu.:-0.54522   1st Qu.:-0.9413823   1st Qu.:-0.57265
##   Median :-0.26926   Median :-0.0580544   Median :-0.04600
##   Mean   :-0.02177   Mean   : 0.0005798   Mean   : 0.02334
##   3rd Qu.: 0.04952   3rd Qu.: 0.8776871   3rd Qu.: 0.75275
##   Max.   : 4.16092   Max.   : 1.8450747   Max.   : 1.63342
##     Y2_adoption.V1      X1_readiness.V1      X4_global
##   Min.   :-1.1881243   Min.   :-1.1967245   Min.   :1.000
##   1st Qu.:-0.4916377   1st Qu.:-0.9281527   1st Qu.:2.000
##   Median :-0.4916377   Median :-0.1224372   Median :2.000
##   Mean   : 0.0226910   Mean   :-0.0026128   Mean   :1.815
##   3rd Qu.: 0.9013357   3rd Qu.: 0.6832784   3rd Qu.:2.000
##   Max.   : 2.9907957   Max.   : 2.2947094   Max.   :2.000
```

```r
par(mfrow=c(2,1), mar=c(2, 3, 2, 3))
X4_orig <- table(as.character(TAA_data$X4_global), useNA = "ifany")
X4_impute <- c(table(TAA_data_impute$X4_global), 0)
rbind(X4_orig, X4_impute)
```

```
##            0  1 <NA>
## X4_orig   11 50    4
## X4_impute 12 53    0
```

```r
par(mfrow=c(2,1), mar=c(2, 3, 2, 3))
compare.density(TAA_data_amelia, var="X3_firmSize")
```



Hence, the TAA_data_impute or TAA_data_scale can be applied to the following analysis

```r
# 1. log transformation of firmSize
TAA_data_impute$X3_firmSize <- log(TAA_data_impute$X3_firmSize)
# 2. scale of data
TAA_data_scale = scale(TAA_data_impute)
str(TAA_data_scale)
```

```
##  num [1:65, 1:10] 0.49 0.575 1.146 0.107 -0.632 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:65] "3" "4" "6" "7" ...
##   ..$ : chr [1:10] "X3_firmSize" "Y1_initialization" "Y3_routinization" "X2_integration" ...
##  - attr(*, "scaled:center")= Named num [1:10] 9.4738 0.0175 0.0428 -0.0221 -0.0218 ...
##   ..- attr(*, "names")= chr [1:10] "X3_firmSize" "Y1_initialization" "Y3_routinization" "X2_integrati
##  - attr(*, "scaled:scale")= Named num [1:10] 1.333 1.012 1.01 0.982 0.963 ...
##   ..- attr(*, "names")= chr [1:10] "X3_firmSize" "Y1_initialization" "Y3_routinization" "X2_integrati
```

Multivariate Regression Analysis (Qiang)

Regression Trees (Drew)

Linting, Mariëlle, Jacqueline J Meulman, Patrick JF Groenen, and Anita J van der Koojj. 2007. "Nonlinear Principal Components Analysis: Introduction and Application." *Psychological Methods* 12 (3): 336.