Building Severity Index with R

Edouard Legoupil

2019-11-28

Contents

In this tutorial, We will see how to build a **severity index** using a key informant interview dataset. Building such index, also called **composite indicator**, is among the regular and expected task of any humanitarian data analyst.The obejctive is synthetise information into a simple indicator that can reduce information overflow. **Severity index** informs the sectoral and inter-sectoral discussions taking place as part of the Humanitarian Needs Overview (HNO) process, in particular facilitating a comparison of needs across geographic areas

An approach to follow for this is described in the *Annex1: Technical Guidance* of the 2014 GUIDANCE: Humanitarian Needs Comparison Tool. Those guidance were provided with a complex and heavy excel tool that includes many macros script and 13 pages of narrative explanation to explain how it works.

Rather than using script through complicated *Macros*, using R is a good and easier alternative. Build those severity index though a documented and linear process brings more credibility in the severity index generation as the analysis becomes de facto entirely reproducible (as a difference with complex excel spreadsheet with macro formula spread around multiple worksheet). Existing packages such as R Compind package for composite indicators have already been developped and relased as open source and therefore ready to be leveraged without any software procurement.

As the analysis workflow shall remain the same, anyone can take and study this tutorial source code, then change the dataset behind with their own in order to re-use it.

0.1 Why do you need statistical analysis for indicator composition?

When developping severity index, each steps comes with methodological questions:

- How to select and organise indicators?
- How to calculate and reshape them?
- How to assemble them together (aggregation and weighting).

The 2014 guidance advise to determine the weight of each sub-indicator through (often unreachable. . .) expert consensus. Such consensus, when obtained, usually takes (very long. . .) consultation and consideration with concerned stakeholders that are most of the time not statisticians themselves and will provide mostly guesstimates for each weight. This approach in the academic world is known as a budget allocation process (BAP), where set of chosen decision makers (e.g. a panel of experts) is given 'n' points to distribute to the indicators, or groups of indicators (e.g. dimensions), and then an average of the experts' choices is used. An alternative to BAP, called "analytic hierarchy process - AHP", is collect each expert opinion in a structured way through pairwise comparison. A rule of thumb for any expert judgement process is to have fewer than

10 indicators so that the approach is optimally executed cognitively, as such most of the time, they are not appropriate for the development of humanitarian severity index.

Arithmetic and geometric aggregation with equal weighting are originally also suggested in the guidance may also comes with assumption of non-compensability or subsituability that are often not verified. Such points have been largely discussed in litterarture (see Benini, Aldo (2016). Severity measures in humanitarian needs assessments - Purpose,measurement, integration. Technical note - 8 August 2016 -. Geneva, Assessment CapacitiesProject (ACAPS)).

The OECD handbook for composite Indicators calculation also thaught through the European Joint Research Training on Composite Indicators present a series of alternatives that can inform / help confirming or triangulate "expert-judgement" . As we will see below, some statistical method are easily accessible for R users in order to get indicators weights.

0.2 Installing packages

To get started, if you don't have them already, the following packages are necessary.

```
## This function will retrieve the packae if they are not yet installed.
using <- function(...) {
    libs <- unlist(list(...))
    req <- unlist(lapply(libs,require,character.only = TRUE))
    need <- libs[req == FALSE]
    if (length(need) > 0) {
        install.packages(need)
        lapply(need,require,character.only = TRUE)
    }
}

## Getting all necessary package
using("readr","readxl","dplyr","ggplot2","corrplot","psych","bbplot","scales","ggiraph","ggrepel","Comp

rm(using)
# This small function is used to have nicely left align text within charts produced with ggplot2
left_align <- function(plot_name, pieces){
  grob <- ggplot2::ggplotGrob(plot_name)
  n <- length(pieces)
  grob$layout$l[grob$layout$name %in% pieces] <- 2
  return(grob)
}
```

0.3 Dataset - Key Informant Intervew for Huricane Dorian, Bahamas

This dataset was collected at the begining of November by IOM in the Bahamas. It covers Abaco Island which has been severly hit by Hurriacan Dorian at the begining of September.

The dataset has been publicly release and is available through HDX here. An initial report has been produced.

,

```
data <- read_excel("DTM R3 DB Great-Little Abaco MSLA V4.xlsx", sheet = "BD")
```

The data includes {r} ncol(data) variables and {r} nrow(data) records, each records corresponding to a location in Abaco.

0.4 Building the theoretical framework.

The original dataset already includes a data dictionnary. The initial step is add more column to this dictionnary in order to record the Severity theoretical framework.

A simple approach here is to regroup indicators by topics / composite dimensions. A composite dimension shall be apprehended as a *construct* i.e. a defined idea of sectoral severity that can only be measured through a number of simpler elements. A good excercise then is to try to give a definition of the type of sectoral severity that is looked at: The definition of the concept should give a clear sense of what is being measured by the composite index. It should be noted that not all indicators shall be blindly included: common sense shall be used to confirm that selected sub-indicators are indeed constructive or reflective of the actual dimension 'construct'.

Looking at the spreadsheet from HDX, we can see that the data structure has already been reshaped / 'hot coded' as what is called a dummy variable so that each variable that includes modalities (categorical: i.e select_one with more than 2 modalities or select_multiple) is split into as many variable as there are modalities.

Single questions could then be extracted and had to be renamed both in terms of shortcode qid shortened Label for good apperance on charts qlabel. as a rule of thumb, a good variable name shall be self-explanatory (avoid q1, q2 , etc.) and very short - not more than 10 characters - A label should not be more than 80 characters and even less if possible.

Then we need to mention how the variable shall be used for the composite calculation, ie. either "binary", "value" or "scored" as we will see later.

```
### Need to implemet manually the analysis framework in the data dictionnary in order to get the calcul
dico <- read_excel("DTM R3 DB Great-Little Abaco MSLA V4.xlsx", sheet = "CatPregs")


## Checking the dico matche with dataframe --
dico1 <- as.data.frame(names(data))
names(dico1)[1] <- "cod_pregunta"
dico <- plyr::join(x = dico1 , y = dico , by = "cod_pregunta", type = "left" )
# Adding variable label in data frame
for (i in 1:nrow(data)) { attributes(data)$variable.labels[ i] <- as.character(dico[ i, c("label")]) }


rm(i, dico1)

## Now creating score variable and applying same direction to all of them!

#levels(as.factor(dico$Calculation))
#labels(dico)
subindicator <- dico[ dico$Calculation %in% c("binary", "value", "scored"), ]
subindicator.unique <- as.data.frame( unique(subindicator[ ,c("qid",  "question",  "qtype" ,
                                                              "Calculation", "Dimension",
                                                              "qlabel","justification",  "Polarity"   )]


## Display the table
subindicator.unique2 <- as.data.frame( unique(subindicator[ ,c( "Dimension","qlabel",  "qtype" ,
                                                              "Calculation",  "Polarity"   )]))
row.names(subindicator.unique2) <- NULL

kable(subindicator.unique2, caption = "Theoretical Framework") %>%
          kable_styling(bootstrap_options = c("striped", "bordered", "condensed", "responsive"), font_s
```

Table 1: Theoretical Framework

Dimension	qlabel	qtype	Calculation	Polarity
Shelter	Shelter Type Impact	select_multiple	scored	Negative (the higher score
Shelter	Mostly living outside	select_one	binary	Negative (the higher score
Shelter	Mostly living in Tent	select_one	binary	Negative (the higher score
Shelter	Mostly living in makeshift	select_one	binary	Negative (the higher score
Shelter	Mostly living indoor	select_one	binary	Negative (the higher score
Shelter	Mostly having eletricity	select_one	binary	Positive (the higher score
Shelter	Mostly having Cooking Facility	select_one	binary	Positive (the higher score
Shelter	Mostly having Private Living	select_one	binary	Positive (the higher score
Shelter	Mostly having Moskito Net	select_one	binary	Positive (the higher score
Protection	Needs hierachy	select_multiple	scored	Negative (the higher score
Shelter	Core Relief Item	select_one	binary	Positive (the higher score
Shelter	Shelter Material Need	select_one	binary	Negative (the higher score
Wash_Waste	Water Source Safety	select_multiple	scored	Negative (the higher score
Wash_Waste	Water Availibility	integer	value	Positive (the higher score
Wash_Waste	Water Quality	select_one	binary	Negative (the higher score
Wash_Waste	Water Smell	select_one	binary	Negative (the higher score
Wash_Waste	Garbage Safety	select_multiple	scored	Negative (the higher score
Wash_Waste	Garbage Pickup	integer	value	Positive (the higher score
Wash_Waste	Debris Removal Safety	select_multiple	scored	Negative (the higher score
Wash_Waste	Waste Problem	select_one	binary	Negative (the higher score
Wash_Waste	Open Defecation	select_one	binary	Negative (the higher score
Food	Food Proximity	select_multiple	scored	Positive (the higher score
Food	Market Access	select_one	binary	Positive (the higher score
Food	Food Distrib Frequency	select_one	scored	Negative (the higher score
Food	Food source dignity	select_multiple	scored	Positive (the higher score
Public_Services_Health_Education	Increased Health Issue	select_one	binary	Negative (the higher score
Public_Services_Health_Education	access to health on side	select_one	binary	Positive (the higher score
Public_Services_Health_Education	Access to medicine	select_one	binary	Positive (the higher score
Public_Services_Health_Education	Health Provider Sustainability	select_one	scored	Positive (the higher score
Public_Services_Health_Education	Health Facility Proximity	select_one	scored	Negative (the higher score
Public_Services_Health_Education	Formal Education Proximity	select_one	scored	Negative (the higher score
Public_Services_Health_Education	Informal Education Proximity	select_one	scored	Negative (the higher score
Public_Services_Health_Education	School Proximity	select_one	scored	Negative (the higher score
Public_Services_Health_Education	School Attendance	select_one	scored	Negative (the higher score
Public_Services_Health_Education	School Fixability	select_multiple	scored	Negative (the higher score
Social_Assistance	Livelihood Opportunity	select_one	binary	Positive (the higher score
Social_Assistance	Current Social Assistance	select_one	binary	Positive (the higher score
Social_Assistance	Current Benefit Diversity	select_multiple	scored	Positive (the higher score
Social_Assistance	Social Assistance before	select_one	binary	Positive (the higher score
Social_Assistance	Previous Benefit Diversity	select_multiple	scored	Positive (the higher score
Protection	Community Relation Level	select_one	scored	Negative (the higher score
Protection	Relation with Host Community	select_one	scored	Negative (the higher score
Protection	Women safety	select_one	scored	Negative (the higher score
Protection	Men safety	select_one	scored	Negative (the higher score
Protection	Child safety	select_one	scored	Negative (the higher score
Protection	Reported Security Incident	select_one	binary	Negative (the higher score
Protection	Reported Immigration Visit	select_one	binary	Negative (the higher score
Protection	Referral Mechanisms	select_one	binary	Positive (the higher score
Protection	Safe/Recreational Places	select_one	binary	Positive (the higher score
Protection	Diversity of Information source	select_multiple	scored	Positive (the higher score

0.5 Calculating each sub-indicator value

The survey has mostly responses of types: 'select_one' and 'select_multiple'. Indeed, collecting reliable numeric value is a well-knwon limitation of key-informant based survey.

Different calculations were used depending on the type of questions:

- For binary questions, negative questions receive a score of 1 when answered positively, such as answering "no", and the score was lowered towards 0 with each negative response given. This will require as we will see below a normalisation based on ranking in order to ensure that geometric means can be done.
- Some'select_one' response choices is ordinal data with an imputed ordered response, where the max score is given either to the best or worst possible choice.
- "select_multiple" questions might have many dummy variables corresponded to the question and the sum of these scores represented the highest score possible. Other 'select_multiple' response choices are discrete choice data with nominal responses; each answer in the 'select_multiple' are weighted according to importance or criticallity.

```
## Creating scores for each of those indicator ###########
## Num col where indic will start to be appended
numcol1 <- ncol(data) + 1
numcol <- ncol(data)

## looping around each new sub indicator to create
for (j in 1:nrow(subindicator.unique)) {

    # j <- 2
    this.indicator.comp <- as.character(subindicator.unique[ j, c("Calculation")])
    this.indicator.type <- as.character(subindicator.unique[ j, c("qtype")])
    this.indicator.name <- as.character(subindicator.unique[ j, c("qid")])
    this.indicator.label <- as.character(subindicator.unique[ j, c("qlabel")])
    this.indicator.question <- as.character(subindicator.unique[ j, c("question")])
    this.indicator.Polarity <- as.character(subindicator.unique[ j, c("Polarity")])
    this.indicator.Dimension <- as.character(subindicator.unique[ j, c("Dimension")])

    ## let's add a variable in the data frame and give it the indic name
    numcol <- numcol + 1
    data[ , numcol] <- ""
    names(data)[numcol] <- this.indicator.name
    attributes(data)$variable.labels[numcol] <- this.indicator.label

    ## Display where we are in the console - always usefull to debug!
    ## Take the # when playing with the script
    #cat(paste0("\n\n===  Indicator: ", j, "-",this.indicator.Dimension ,
    #           "-",this.indicator.label ,"\n",this.indicator.comp ,
    #           "-",this.indicator.Polarity ,"\n"))

    ## Now accounting for 2 distinct cases to get my calculation
    ## "binary" / "value" or "scored"

      ## If it's a score, we need to sum up all scores for that questions  #####
      # - independently of wether it's a select_one or select_multiple -
      if (this.indicator.comp == "scored" ) {
      ## get the correponding value var
        this.value.var <- as.character(subindicator[ subindicator$question == this.indicator.question,
```

```
        this.subset <- data[ , this.value.var]

        ## Now get an apply the coeffcient
        this.subsetscore <- t(as.data.frame(subindicator[ subindicator$cod_pregunta %in% this.value.var

      ## multiply data frame by a vector
      this.subset3 <- data.frame(mapply(`*`,this.subset, this.subsetscore, SIMPLIFY = FALSE))
      #str(this.subset3)
      # this.subset4 <- cbind(this.subset3,  rowSums(this.subset3, na.rm = TRUE))

      #cat(paste0("Calculation of Indicator: ", j, "\n"))
      ## Get the sum of the row
      data[ , numcol] <- rowSums(this.subset3, na.rm = TRUE)
    }


    ## If it's a  value, we just take the value of that binary  #####-
    if (this.indicator.comp %in% c("value")) {
      ## get the correponding value var
      this.value.var <- as.character(dico[ dico$question == this.indicator.question, c("cod_pregunta") ]
      ## Apply the value
      #cat(paste0("Calculation of Indicator: ", j, "\n"))
      data[ , numcol] <- data[ , this.value.var]
    }

    ## If it's a binary , we add 1 to avoid zero value #####-
    if (this.indicator.comp %in% c("binary")) {
      ## get the correponding value var
      this.value.var <- as.character(dico[ dico$question == this.indicator.question, c("cod_pregunta") ]
      ## Apply the value
      #cat(paste0("Calculation of Indicator: ", j, "\n"))
      data[ , numcol] <- data[ , this.value.var] + 1
    }

    ## clean
    rm(this.indicator.comp, this.indicator.type, this.indicator.name,
       this.indicator.label, this.indicator.question, this.indicator.Polarity ,
       this.indicator.Dimension, this.subset, this.subset3, this.subsetscore, this.value.var )
}
```

0.5.1 Sub-Indicator Polarity & Data Normalisation,

We can now isolate our new numeric and scaled indicators within a matrix as this object type will be required for the rest of the calculations.

First, we need to eliminate indicators with poor discrimination capacity, i.e. when all values are the sames.

```
## Double Checking results
#View(data[ , numcol1:ncol(data)])
indic <- data[ , numcol1:ncol(data)]

### Remove var when standard deviation is 0
indic2 <- indic[, sapply(indic, function(x) { sd(x) != 0} )]
```

```r
## Refresh my dictionnary of indic
subindicator.unique2 <- subindicator.unique[ subindicator.unique$qid %in% names(indic2), ]

## Transform this object as a matrix and inject location name as row.names
indic.matrix <- as.matrix(indic2)
row.names(indic.matrix) <- data$C_101_name
```

The **polarity** of a sub-indicator is the sign of the relationship between the indicator and the phenomenon to be measured (e.g., in a well-being index, "GDP per capita" has 'positive' polarity and "Unemployment rate" has 'negative' polarity). This componnent is accounted for during the normalisation process below.

Due to the structure of the indicators, distinct approaches of normalisation shall be considered in order to avoid having zero value that would create issues for geometric means aggregation:

- If the variable is scored, a z-score approach `method = 1`
- If the variable is value, a min-max approach `method = 2`
- If the variable is binary, ranking method `method = 3`

```r
## Retrieve polarity from dictionnary
for (i in 1:nrow(subindicator.unique2)) {
  if (subindicator.unique2[ i, c("Polarity")] == "Negative (the higher score, the worst)")
    {subindicator.unique2[ i, c("polarity")]  <- "NEG"} else
    {subindicator.unique2[ i, c("polarity")]  <- "POS"}
 }
subindicator.unique2$dir <- 1

## Case 1
subindicator.scored <- subindicator.unique2[ subindicator.unique2$Calculation == "scored", ]
var.scored <- as.character(subindicator.scored[ , c("qid") ])
indic.matrix.scored <- indic.matrix[ , var.scored ]
indic.matrix.scored.obj <- normalise_ci(indic.matrix.scored,
                                        c(1:ncol(indic.matrix.scored)),
                                        polarity =  as.character(subindicator.scored$polarity ),
                                        method = 2)

## Case 2
subindicator.value <- subindicator.unique2[ subindicator.unique2$Calculation == "value", ]
var.value <- as.character(subindicator.value[ , c("qid") ])
indic.matrix.value <- indic.matrix[ , var.value ]
indic.matrix.value.obj <- normalise_ci(indic.matrix.value,
                                        c(1:ncol(indic.matrix.value)),
                                        polarity =  as.character(subindicator.value$polarity ),
                                        method = 2)

## Case 3
subindicator.binary <- subindicator.unique2[ subindicator.unique2$Calculation == "binary", ]
var.binary <- as.character(subindicator.binary[ , c("qid") ])
indic.matrix.binary <- indic.matrix[ , var.binary ]
indic.matrix.binary.obj <- normalise_ci(indic.matrix.binary,
                                        c(1:ncol(indic.matrix.binary)),
                                        polarity =  as.character(subindicator.binary$polarity ),
                                        method = 2)

## Binding this together so that we have the full normalised matrix
```

```r
indic.matrix.norm <- cbind(indic.matrix.scored.obj$ci_norm,
                           indic.matrix.value.obj$ci_norm,
                           indic.matrix.binary.obj$ci_norm)

## Clean the work environment
rm(indic2, subindicator.unique,
   subindicator.value, var.value , indic.matrix.value, indic.matrix.value.obj,
   subindicator.binary, var.binary , indic.matrix.binary, indic.matrix.binary.obj,
   subindicator.scored, var.scored , indic.matrix.scored, indic.matrix.scored.obj   )
```

0.5.2 Correlation analysis

The investigation of the structure of simple indicators can be done by means of correlation analysis.

We will check such correlation first within each dimension, using the ggcorrplot package. ’

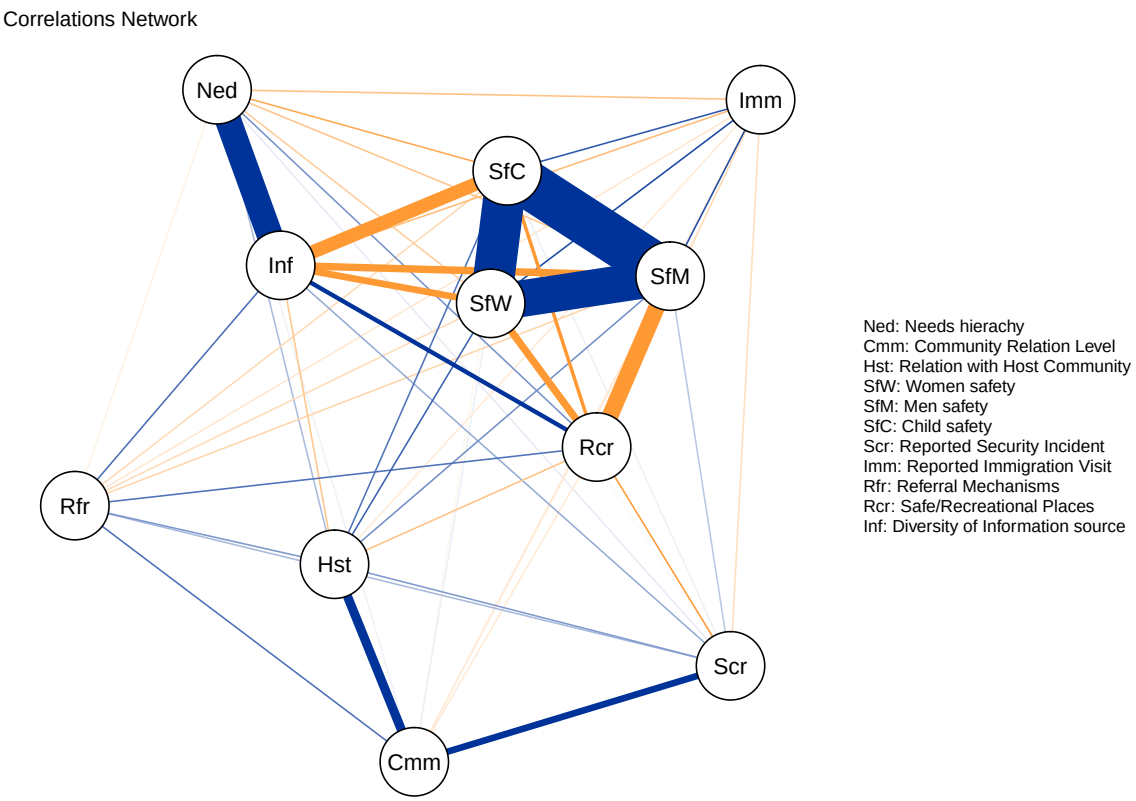
```r
##  Frame with all dimensions
dimensions <- as.data.frame( unique(subindicator[ ,c( "Dimension" )]))
names(dimensions)[1] <- "Dimension"

## Creating severity subindice on each dimensions with Data Envelopment analysis #####

#for (i in 1:nrow(dimensions)) {
#for (i in 1:2) {
  i <- 2
  ## looping around dimensions
  this.dimension <- as.character(dimensions[i,1])
  ## subset related indicator names
  this.indicators <- as.character(subindicator.unique2[ subindicator.unique2$Dimension == this.dimensio
                                                        c("qid") ])
  this.indicators.label <- as.character(subindicator.unique2[ subindicator.unique2$Dimension == this.di
                                                        c("qlabel") ])
  ##subset matrix
  this.indic.matrix.norm <- indic.matrix[ , this.indicators]

### Check correlation
corr.matrix <- cor(this.indic.matrix.norm, method = "pearson",  use = "pairwise.complete.obs")


## replace with Label inside the matrix
corr.matrix1 <- corr.matrix
rownames(corr.matrix1) <- as.character(subindicator.unique2[subindicator.unique2$qid %in% rownames(corr
colnames(corr.matrix1) <- as.character(subindicator.unique2[subindicator.unique2$qid %in% colnames(corr

plot1 <- ggcorrplot(corr.matrix1 ,
                    method = "circle",
                    hc.order = TRUE,
                    type = "upper") +
  labs(title = paste0( "Selected Indicators for ",this.dimension ),
       subtitle = "Identified Correlation between indicators",
       caption = "Correlation level = dot size, Positive Correlation  = Red - Negative = Blue",
       x = NULL, y = NULL) +
  bbc_style() +
```

```
  theme( plot.title = element_text(size = 13),
         plot.subtitle = element_text(size = 11),
         plot.caption = element_text(size = 7, hjust = 1),
         axis.text = element_text(size = 7),
         strip.text.x = element_text(size = 7),
         axis.text.x = element_text(angle = 45, hjust = 1),
         legend.position = "top",
         legend.box = "horizontal",
         legend.text = element_text(size = 9),
         panel.grid.major.x = element_line(color = "#cbcbcb"),
         panel.grid.major.y = element_line(color = "#cbcbcb"))
ggpubr::ggarrange(left_align(plot1, c("subtitle", "title")), ncol = 1, nrow = 1)
```

Another approach to better visualise correlation between indicators is to represent them through a network

with the ggpraph package.

```
qgraph(cor(this.indic.matrix.norm),
     # shape = "circle",
     # posCol = "darkgreen",
     # negCol = "darkred",
     # threshold = "bonferroni", #The threshold argument can be used to remove edges that are not signi
     # sampleSize = nrow(scores.this.norm),
     # graph = "glasso",
       esize = 35, ## Size of node
       vsize = 6,
       vTrans = 600,
       posCol = "#003399", ## Color positive correlation Dark powder blue
       negCol = "#FF9933", ## Color negative correlation Deep Saffron
       alpha = 0.05,
       cut = 0.4, ## cut off value for correlation
       maximum = 1, ## cut off value for correlation
       palette = 'pastel', # adjusting colors
       borders = TRUE,
       details = FALSE,
       layout = "spring",
       nodeNames = this.indicators.label ,
       legend.cex = 0.4,
       title = "Correlations Network",
       line = -2,
       cex.main = 2)
```



0.5.3 Consistency between indicators

Cronbach's alpha, (or coefficient alpha), developed by Lee Cronbach in 1951, measures reliability (i.e. how well a test measures what it should: measure of the stability of test scores), or internal consistency.

As a rule of thumbs, a score of more than 0.7 indicates an acceptable level of consistency. This should still be used with caution. A high level for alpha may mean that the items in the test are highly correlated. However, alpha is also sensitive to the number of items in a test. A larger number of items can result in a larger alpha, and a smaller number of items in a smaller alpha. If alpha is high, this may mean redundant questions (i.e. they're asking the same thing). A low value for alpha may mean that there aren't enough questions on the test. Adding more relevant items to the test can increase alpha. Poor interrelatedness between test questions can also cause low values, so can measuring more than one latent variable.

,

```
  Cronbach.this <- psych::alpha(this.indic.matrix.norm, check.keys = TRUE)

  cat(paste0("The Cronbach Alpha measure of consistency for this combination of indicators is  ", round
```

```
The Cronbach Alpha measure of consistency for this combination of indicators is  0.75
```

.

0.6 Aggregation & Weighting

For weighting, the main issue to adress is related to the concept of **compensability**. Namely the question is to know to what extent can we accept that the high score of an indicator go to compensate the low score of another indicator? This problem of compensability is intertwined with the issue of attribution of weights for each sub-indicator in order to calculate the final aggregation.

We can foresee that using *"equal weight"* (all indicators account for the same in the final index) and *"arithmetic aggregation"* (all indicators are subsituable) is unlikely to depict the complex isssue of Humanitarian Severity and is likely to comes with the risk of misrepresenting the reality.

Various methods are avaialble within the Compind package are described below. This R package is also available through a ShinyApp. We will then share the code to use them based on our example.

0.6.1 Benefit of the Doubt approach (BoD)

This method is the application of Data Envelopment Analysis (DEA) to the field of composite indicators. It was originally proposed by Melyn and Moesen (1991) to evaluate macroeconomic performance. ACAPS has prepared an excellent note on The use of data envelopment analysis to calculate priority scores in needs assessments.

BoD approach offers several advantages:

- Weights are endogenously determined by the observed performances and benchmark is not based on theoretical bounds, but it's a linear combination of the observed best performances.
- Principle is easy to communicate: since we are not sure about the right weights, we look for "benefit of the doubt" weights (such that your overall relative performance index is as high as possible).

```
CI_BoD_estimated = ci_bod(this.indic.matrix.norm,
                          indic_col = (1:ncol(this.indic.matrix.norm)))

ci_bod_est <- as.data.frame( CI_BoD_estimated$ci_bod_est)
names(ci_bod_est) <- "Benef_Doubt"
```

0.6.2 Directional Benefit of the Doubt (D-BoD)

Directional Benefit of the Doubt (D-BoD) model enhance non-compensatory property by introducing directional penalties in a standard BoD model in order to consider the preference structure among simple indicators. This method is described in the article Enhancing non compensatory composite indicators: a directional proposal.

```
## Endogenous weight - no zero weight --
CI_Direct_BoD_estimated <-  ci_bod_dir(this.indic.matrix.norm,
                                       indic_col = (1:ncol(this.indic.matrix.norm)),
                                       dir = as.numeric(subindicator.unique2[subindicator.unique2$Dimens

ci_bod_dir_est <- data.frame(CI_Direct_BoD_estimated$ci_bod_dir_est)
names(ci_bod_dir_est) <- "Benef_Doubt_Dir"
```

0.6.3 Robust Benefit of the Doubt approach (RBoD)

ci_rbod_est is the robust version of the BoD method. It is based on the concept of the expected minimum input function of order-m so "in place of looking for the lower boundary of the support of F, as was typically the case for the full-frontier (DEA or FDH), the order-m efficiency score can be viewed as the expectation of the maximal score, when compared to m units randomly drawn from the population of units presenting a greater level of simple indicators", Daraio and Simar (2005). This method is described with more detail in the article Robust weighted composite indicators by means of frontier methods with an application to European infrastructure endowment.

```
CI_RBoD_estimated <-  ci_rbod(this.indic.matrix.norm,
                              indic_col = (1:ncol(this.indic.matrix.norm)),
                              M = 20,  #The number of elements in each sample.
                              B = 200) #The number of bootstap replicates.


ci_rbod_est <- data.frame(CI_RBoD_estimated$ci_rbod_est)
names(ci_rbod_est) <- "Benef_Doubt_Rob"
```

0.6.4 Benefit of the Doubt approach (BoD) index with constraints on weights

ci_bod_constr_est_mpi allows for constraints (so that there is none not valued) and with a penalty as proposed by Mazziotta - Pareto (also adopted by the Italian National Institute of Statistics). This method is described in the article Geometric mean quantity index numbers with Benefit-of-the-Doubt weights

```
CI__BoD_MPI_estimated = ci_bod_constr_mpi(this.indic.matrix.norm,
                                          indic_col = (1:ncol(this.indic.matrix.norm)),
                                          up_w = 1,
                                          low_w = 0.1,
                                          penalty = "POS")

ci_bod_constr_est_mpi <- data.frame(CI__BoD_MPI_estimated$ci_bod_constr_est_mpi)
names(ci_bod_constr_est_mpi) <- "Benef_Doubt_Cons"
```

0.6.5 Factor analysis

ci_factor_est groups together collinear simple indicators to estimate a composite indicator that captures as much as possible of the information common to individual indicators.

```
##  Doing PCA with ci_factor.R
# If method = "ONE" (default) the composite indicator estimated values are equal to first component sco
# if method = "ALL" the composite indicator estimated values are equal to component score multiplied by
# if method = "CH" it can be choose the number of the component to take into account.
dimfactor <- ifelse(ncol(this.indic.matrix.norm) > 2, 3, ncol(this.indic.matrix.norm))
CI_Factor_estimated <-  ci_factor(this.indic.matrix.norm,
                                  indic_col = (1:ncol(this.indic.matrix.norm)),
                                  method = "CH",  # if method = "CH" it can be choose the number of the
                                  dim = dimfactor)
ci_factor_est <- data.frame( CI_Factor_estimated$ci_factor_est)
names(ci_factor_est) <- "Factor"
```

0.6.6 Mean-Min Function (MMF)

ci_mean_min_est is an intermediate case between arithmetic mean, according to which no unbalance is penalized, and min function, according to which the penalization is maximum. It depends on two parameters that are respectively related to the intensity of penalization of unbalance (alpha) and intensity of complementarity (beta) among indicators. "An unbalance adjustment method for development indicators"

```
CI_mean_min_estimated <- ci_mean_min(this.indic.matrix.norm,
                                     indic_col = (1:ncol(this.indic.matrix.norm)),
                                     alpha = 0.5,  #intensity of penalization of unbalance  (alpha)
                                     beta = 1) # intensity of complementarity (beta) among indicators

ci_mean_min_est <- data.frame( CI_mean_min_estimated$ci_mean_min_est)
names(ci_mean_min_est) <- "Mean_Min"
```

0.7 Geometric aggregation

This method uses the geometric mean to aggregate the single indicators and therefore allows to bypass the full compensability hypothesis using geometric mean. Two weighting criteria are possible: EQUAL: equal weighting and BOD: Benefit-of-the-Doubt weights following the Puyenbroeck and Rogge (2017) approach.

```
CI_Geom_estimated = ci_geom_gen(this.indic.matrix.norm,
                                indic_col = (1:ncol(this.indic.matrix.norm)),
                                meth = "EQUAL",
                                ## "EQUAL" = Equal weighting set, "BOD" = Benefit-of-the-Doubt weightin
                                up_w = 1,
                                low_w = 0.1,
                                bench = 1)
# Row number of the benchmark unit used to normalize the data.frame x.

ci_mean_geom_est <- data.frame( CI_Geom_estimated$ci_mean_geom_est)
names(ci_mean_geom_est) <- "Mean_Geom"
```

0.7.1 Mazziotta-Pareto Index (MPI)

This method is is a non-linear composite index method which transforms a set of individual indicators in standardized variables and summarizes them using an arithmetic mean adjusted by a "penalty" coefficient related to the variability of each unit (method of the coefficient of variation penalty).

```
CI_MPI_estimated <- ci_mpi(this.indic.matrix.norm,
                           indic_col = (1:ncol(this.indic.matrix.norm)),
                           penalty = "NEG")  # Penalty direction; "POS" (default) in case of increasing
#  or "positive" composite index (e.g., well-being index),
#  "NEG" in case of decreasing or "negative" composite
#  index (e.g., poverty index).

ci_mpi_est <- data.frame( CI_MPI_estimated$ci_mpi_est)
names(ci_mpi_est) <- "Mazziotta_Pareto"
```

0.7.2 Wroclaw taxonomy method

This last method (also known as the dendric method), originally developed at the University of Wroclaw, is based on the distance from a theoretical unit characterized by the best performance for all indicators considered; the composite indicator is therefore based on the sum of euclidean distances from the ideal unit and normalized by a measure of variability of these distance (mean + 2*std).

```
CI_wroclaw_estimated <-  ci_wroclaw(this.indic.matrix.norm,
                                    indic_col = (1:ncol(this.indic.matrix.norm)))

ci_wroclaw_est <- data.frame( CI_wroclaw_estimated$ci_wroclaw_est)
names(ci_wroclaw_est) <- "Wroclaw"
```

0.8 Visualise output

0.8.1 In a table

```
this.indic.matrix.norm2 <- cbind( #row.names(scores.this),
  ci_bod_est, # Benefit of the Doubt approach
  ci_rbod_est, # Robust Benefit of the Doubt approach
  ci_bod_dir_est, # Directional Robust Benefit of the Doubt approach
  ci_bod_constr_est_mpi, # Robust Benefit of the Doubt approach with constraint
  ci_factor_est, # Factor analysis  componnents
  ci_mean_geom_est, # Geometric aggregation
  ci_mean_min_est, # Mean-Min Function
  ci_mpi_est, # Mazziotta-Pareto Index
  ci_wroclaw_est) # Wroclaw taxonomy method

this.indic.matrix.norm22 <- this.indic.matrix.norm2[, colSums(this.indic.matrix.norm2 != 0, na.rm = TRU
this.indic.matrix.norm22 <- this.indic.matrix.norm22[, colSums(this.indic.matrix.norm22 != 0, na.rm = TR


kable(this.indic.matrix.norm22, caption = "Composite with different algorithm") %>%
          kable_styling(bootstrap_options = c("striped", "bordered", "condensed", "responsive"), font_s
```

0.8.2 Differences between algorithm

Indicators can be normalised again on a 0 to 1 scale in order to be compared.

```
this.indic.matrix.norm22 <- cbind( #row.names(scores.this),
  ci_bod_constr_est_mpi, # Robust Benefit of the Doubt approach with constraint
```

Table 2: Composite with different algorithm

	Benef_Doubt	Benef_Doubt_Dir	Factor	Mean_Min	Mazziotta_Pareto	Wroclaw
Bahama Palm Shores	1.0000000	1.0000000	0.1263115	95.68749	102.74861	0.7283793
Bahamas Coral Island	1.0000000	1.0000000	0.1667973	17615.55029	419799.13986	0.0011632
Blackwood	1.0000000	1.0000000	-0.4191904	105.47782	146.03615	0.7283068
Casuarina Point	0.8289113	0.7465266	0.1077680	102.84590	133.41678	0.7283794
Cedar Harbour	1.0000000	1.0000000	0.1655291	99.40646	112.97202	0.7283796
Central Pines	1.0000000	1.0000000	0.3038825	93.97824	115.02301	0.7283800
Cherokee Sound	1.0000000	1.0000000	-0.3494143	96.30149	108.62189	0.7283796
Cooper's Town	1.0000000	1.0000000	-0.1243346	98.15548	113.48195	0.7283798
Crossing Rocks	0.6666722	0.6666722	-0.4137166	96.13554	115.99038	0.7283798
Crown Heaven	1.0000000	1.0000000	0.1879958	107.06983	179.38669	0.7283796
Dundas Town	1.0000000	1.0000000	0.2474314	99.91206	110.52062	0.7283798
Fire Road	0.8571429	0.8000000	-0.0305124	93.68844	97.78775	0.7283800
Fox Town	0.8148148	0.7142872	-0.3126482	96.22623	112.93238	0.7283797
Great Cistern	1.0000000	1.0000000	0.3501212	106.48115	141.64900	0.7283796
Leisure Lee	0.8333333	0.8000000	-0.3532438	95.86313	106.03791	0.7283798
Little Harbour	1.0000000	1.0000000	-0.5568539	92.84687	98.86473	0.7283072
Marsh Harbour	1.0000000	1.0000000	0.3753384	102.76031	119.34111	0.7283066
Mount Hope	1.0000000	1.0000000	-0.2465344	98.32946	118.88988	0.7283795
Murphy Town	1.0000000	1.0000000	0.4581897	98.28368	115.58783	0.7283069
Sandy Point	1.0000000	1.0000000	0.4454088	116.71618	277.00729	0.7283064
Spring City	0.8571429	0.8000000	-0.0305124	90.29095	115.78289	0.7283800
Treasure Cay	1.0000000	1.0000000	-0.0680113	96.83595	118.10885	0.7283070
Wood Cay	1.0000000	1.0000000	-0.0298015	95.42621	100.11525	0.7283069

```
  ci_bod_est, # Benefit of the Doubt approach
  ci_bod_dir_est, # Directional Robust Benefit of the Doubt approach
  ci_rbod_est, # Robust Benefit of the Doubt approach
  ci_factor_est, # Factor analysis  componnents
  ci_mean_min_est, # Mean-Min Function
  ci_mpi_est, # Mazziotta-Pareto Index
  ci_mean_geom_est, # Geometric aggregation
  ci_wroclaw_est) # Wroclaw taxonomy method

rm( ci_bod_constr_est_mpi, # Robust Benefit of the Doubt approach with constraint
  ci_bod_est, # Benefit of the Doubt approach
  ci_bod_dir_est, # Directional Robust Benefit of the Doubt approach
  ci_rbod_est, # Robust Benefit of the Doubt approach
  ci_factor_est, # Factor analysis  componnents
  ci_mean_min_est, # Mean-Min Function
  ci_mpi_est, # Mazziotta-Pareto Index
  ci_mean_geom_est, # Geometric aggregation
  ci_wroclaw_est,# Wroclaw taxonomy method
  CI_wroclaw_estimated)



this.indic.matrix.norm22$Factor <- NULL
## Polarity are all the same except for worclaw
polarity2 <-  c(
  "POS", # Robust Benefit of the Doubt approach with constraint
  "POS", # Benefit of the Doubt approach
```

Table 3: Ranking with different algorithm

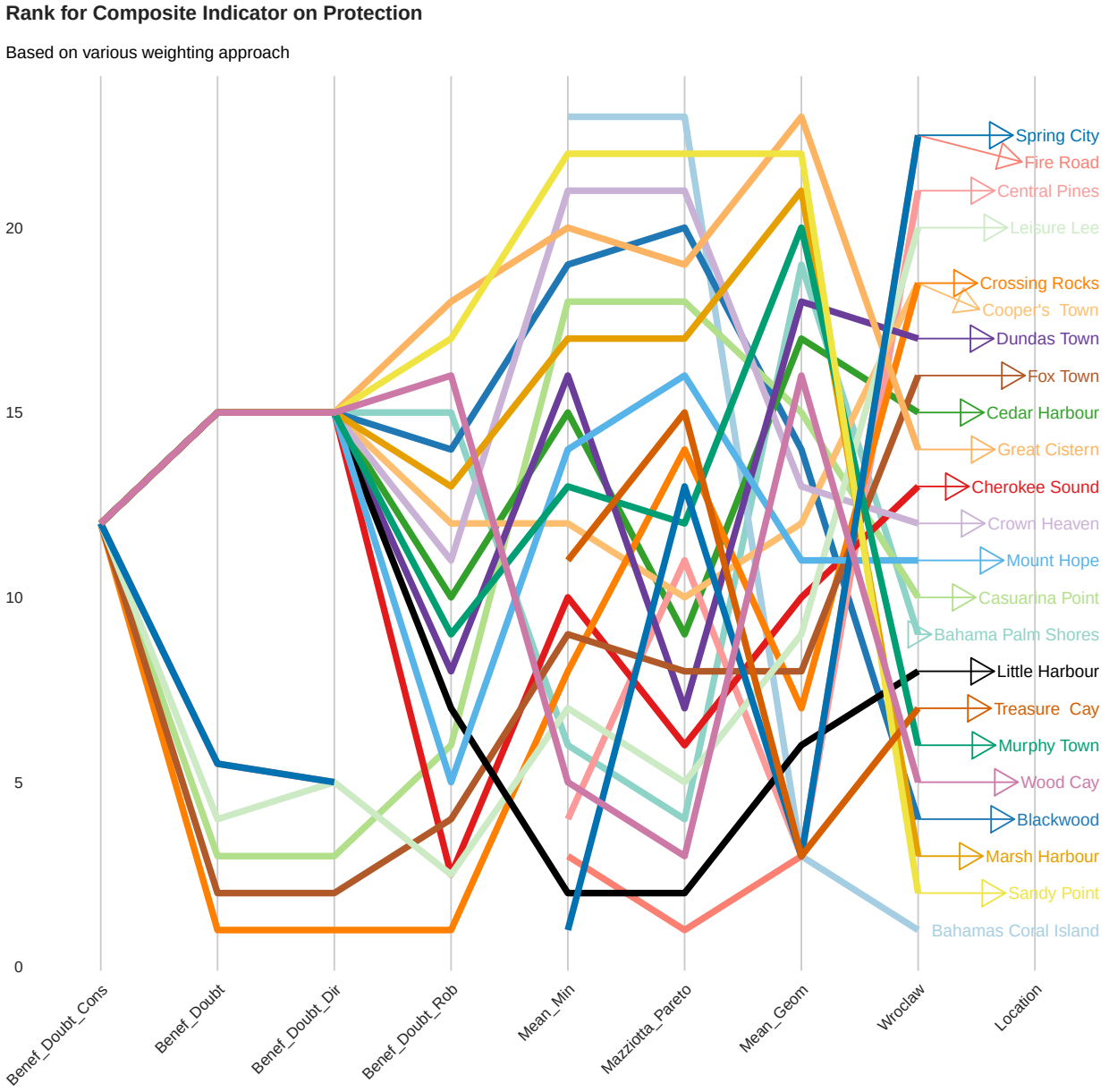
	Benef_Doubt_Cons	Benef_Doubt	Benef_Doubt_Dir	Benef_Doubt_Rob	Mean_Min	Mazziott
Bahama Palm Shores	12	15.0	15	15.0	6	
Bahamas Coral Island	12	15.0	15	NA	23	
Blackwood	12	15.0	15	14.0	19	
Casuarina Point	12	3.0	3	6.0	18	
Cedar Harbour	12	15.0	15	10.0	15	
Central Pines	12	15.0	15	NA	4	
Cherokee Sound	12	15.0	15	2.5	10	
Cooper's Town	12	15.0	15	12.0	12	
Crossing Rocks	12	1.0	1	1.0	8	
Crown Heaven	12	15.0	15	11.0	21	
Dundas Town	12	15.0	15	8.0	16	
Fire Road	12	5.5	5	NA	3	
Fox Town	12	2.0	2	4.0	9	
Great Cistern	12	15.0	15	18.0	20	
Leisure Lee	12	4.0	5	2.5	7	
Little Harbour	12	15.0	15	7.0	2	
Marsh Harbour	12	15.0	15	13.0	17	
Mount Hope	12	15.0	15	5.0	14	
Murphy Town	12	15.0	15	9.0	13	
Sandy Point	12	15.0	15	17.0	22	
Spring City	12	5.5	5	NA	1	
Treasure Cay	12	15.0	15	NA	11	
Wood Cay	12	15.0	15	16.0	5	

```
  "POS", # Directional Robust Benefit of the Doubt approach
  "POS",# Robust Benefit of the Doubt approach
  "POS", # Factor analysis  componnents
  "POS", # Mean-Min Function
  "POS", # Mazziotta-Pareto Index
  "POS", # Geometric aggregation
  "POS") # Wroclaw taxonomy method



## Normalisation with Min Max -- with Compind packages
this.indic.matrix.norm3 <- normalise_ci(this.indic.matrix.norm22,
                                        c(1:ncol(this.indic.matrix.norm22)),
                                        polarity =  polarity2,
                                        method = 3)

this.indic.matrix.norm3 <- this.indic.matrix.norm3$ci_norm


kable(this.indic.matrix.norm3, caption = "Ranking with different algorithm") %>%
           kable_styling(bootstrap_options = c("striped", "bordered", "condensed", "responsive"), font_s
```

We can present that table in a chart as well

```
## Remove NaN
this.indic.matrix.norm3 <-  this.indic.matrix.norm3[,colSums(this.indic.matrix.norm3 != 0, na.rm = TRUE)
```

```r
## keep that frame for later on for the viz
assign(  paste("scores.", this.dimension, sep = ""), this.indic.matrix.norm3 )

## Add blank variable for nice chart display
this.indic.matrix.norm3$Location <- NA

this.indic.matrix.norm3.melt <- melt(as.matrix(this.indic.matrix.norm3))


#Make plot
line <- ggplot(this.indic.matrix.norm3.melt, aes(x = Var2,
                                                 y = value,
                                                 color = Var1,
                                                 group = Var1)) +
  geom_line(size = 2) +
  scale_colour_manual(values = c("#8dd3c7","#A6CEE3", "#1F78B4", "#B2DF8A", "#33A02C",
                                 "#FB9A99", "#E31A1C", "#FDBF6F", "#FF7F00", "#CAB2D6",
                                 "#6A3D9A", "#fb8072", "#B15928", "#fdb462","#ccebc5",
                                 "#000000", "#E69F00", "#56B4E9", "#009E73", "#F0E442",
                                 "#0072B2", "#D55E00", "#CC79A7")) +
  geom_text_repel(
    data = this.indic.matrix.norm3.melt[ this.indic.matrix.norm3.melt$Var2 == "Wroclaw", ],
    aes(label = Var1),
    arrow = arrow(length = unit(0.03, "npc"), type = "closed", ends = "first"),
    direction = "y",
    size = 4,
    nudge_x = 45 ) +
  labs(title = paste0("Rank for Composite Indicator on ",  this.dimension ),
       subtitle = "Based on various weighting approach") +
  bbc_style() +
  theme( plot.title = element_text(size = 13),
         plot.subtitle = element_text(size = 11),
         plot.caption = element_text(size = 7, hjust = 1),
         axis.text = element_text(size = 10),
         strip.text.x = element_text(size = 11),
         panel.grid.major.x = element_line(color = "#cbcbcb"),
         panel.grid.major.y = element_blank(),
         axis.text.x = element_text(angle = 45, hjust = 1),
         legend.position = "none")

print(ggpubr::ggarrange(left_align(line, c("subtitle", "title")), ncol = 1, nrow = 1))
```



0.8.3 Severity Index on a map

```
## Remove NaN
#map <- cbind( as.data.frame(this.indic.matrix.norm22, data[ ,c("C_102_lat","C_103_lon")])
```