

How to Calculate Empirically Derived Composite or Indicator Scores.

Dr. Jon Starkweather, Research and Statistical Support Consultant

This month's article was motivated by the frequent need to calculate composite scores from multiple variables. Often we have Likert response survey data and want to combine several questions' or items' responses into a composite score; and then use the composite score(s) as a variable of interest in some traditional analysis. For those unfamiliar, Likert response survey questions have response choices such as; strongly disagree, disagree, agree, strongly agree. The response choices are typically considered ordinal; meaning, they represent sequentially ordered categories (e.g. strongly disagree = 1, disagree = 2, agree = 3, strongly agree = 4). In this context, we typically refer to the words as *labels* (e.g. "strongly disagree") and the numbers as *values* (e.g. "1"). Occasionally, we are confronted with a client who wants to simply average each participant's response values on several questions to arrive at a composite score for the domain which the questions are believed to be assessing. This is generally a bad idea because, it treats each question as contributing to the composite score equally – which is often not the case when one considers the latent variable structure of what one is attempting to measure or assess.

Essentially, we will be using factor analysis to generate the composite scores. In a very real sense, the best composite scores are factor scores when there is a known, or strongly supported belief in, structure of the data. There are a few ways to go about generating the composite, or factor, scores based on what type of structure you believe the data contains (e.g. single factor, multiple correlated factors, multiple uncorrelated factors, bifactor model, hierarchical factor model, etc.); and how the variables are measured or scaled (e.g., nominal scaled, ordinal or Likert scaled, interval/ratio scaled, etc.).

The general procedure for generating composite / indicator scores includes the following steps: (1) convert, or recode, nominal or ordinal (Likert) responses to numeric responses, (2) apply a factor analysis model which reflects the known structure, or calculated correlation structure, of the variables, (3) save the factor scores and factor loadings, (4) rescale the factor scores using the factor loadings, the weighted mean, and the weighted standard deviation of the original data so that the composite scores reflect (as nearly as possible) the original semantic (i.e., word) meaning of the original data. In this process, the factor loadings serve as weights for the weighted mean and weighted standard deviation calculations. The last step of rescaling the composite scores is necessary because it allows us to retain the meaning of the responses which went into creating the composites. For instance, if we have a composite score of 3.6 and the four questions' responses which were used to create that composite were all 4-point Likert style with the labels and values; strongly disagree = 1, disagree = 2, agree = 3, strongly agree = 4, then we can say the 3.6 means that the person associated with that score responded more with strongly agree than they did with agree, disagree, or strongly disagree. The primary benefits of using the rescaled factor scores as composite scores are that they are considered interval/ratio scaled and they reflect more closely a *true score* on the latent construct we were attempting to measure.

Examples

First, import some example data (the data file used below has been simulated).

```

R Console
File Edit Misc Packages Windows Help

> data.df <- read.table(
+ "http://www.unt.edu/rss/class/Jon/R_SC/Module4/CompositeIndicators_001.txt",
+ header=TRUE, sep=";", na.strings="NA", dec=".", strip.white=TRUE)
> summary(data.df); nrow(data.df)
      id          city.names      gender          age          education
Min.   : 5      Bolder       :346   Female:879   Min.   : 15.00   Min.   : 2.00
1st Qu.: 2538   Burkley       :351   Male  :852   1st Qu.: 30.00   1st Qu.: 7.00
Median : 5028   Ditroit      :365                    Median : 40.00   Median :10.00
Mean   : 5005   El Pasio     :333                    Mean   : 42.82   Mean   :10.08
3rd Qu.: 7480   South Beech:336                    3rd Qu.: 53.00   3rd Qu.:13.00
Max.   :10000                    Max.   :104.00   Max.   :21.00

      income          q1          q2
Min.   : 27923   agree       :415   agree       :426
1st Qu.: 70125   disagree    :425   disagree    :429
Median : 89214   strongly agree :451   strongly agree :437
Mean   : 88256   strongly disagree:440   strongly disagree:439
3rd Qu.:107016
Max.   :164972

      q3          q4          q5
agree       :449   agree       :436   hyperactive :339
disagree    :417   disagree    :426   lethargic   :341
strongly agree :417   strongly agree :445   more active  :359
strongly disagree:448   strongly disagree:424   no difference :345
not very active:347

      q6          q7          q8
hyperactive :375   hyperactive :348   hyperactive :335
lethargic   :346   lethargic   :361   lethargic   :344
more active  :324   more active  :355   more active  :334
no difference :345   no difference :321   no difference :365
not very active:341   not very active:346   not very active:353

      q9          q10          q11
hyperactive :330   agree       :419   agree       :424
lethargic   :356   disagree    :428   disagree    :433
more active  :346   strongly agree :436   strongly agree :437
no difference :352   strongly disagree:448   strongly disagree:437
not very active:347

      q12          q13          q14
agree       :423   agree       :408   agree       :396
disagree    :423   disagree    :430   disagree    :430
strongly agree :442   strongly agree :447   strongly agree :447
strongly disagree:443   strongly disagree:446   strongly disagree:458

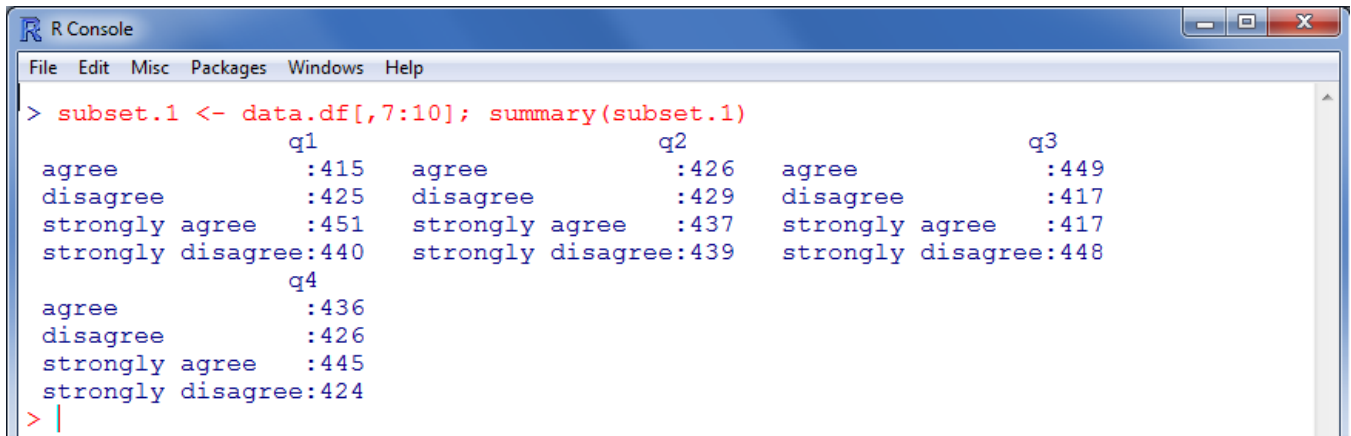
[1] 1731
>

```

In this example, we have 3 groups of questions: q1 - q4, q5 - q9, and q10 - q14. The response choices for q1 - q4 and q10 - q14 were the same: 1 = strongly disagree, 2 = disagree, 3 = agree, 4 = strongly agree. The response choices for q5 - q9 were: 1 = lethargic, 2 = not very active, 3 = no difference, 4 = more active, 5 = hyperactive. In this example, we have three groups of questions; each

group measures a particular latent construct (i.e. indirect measurement) and the three latent constructs are mildly correlated to one another. The above statements represent the known (or strongly supported hypothesis of) the data's (factor) structure; in this example a model with three mildly correlated factors.

Consider the situation where you have a set of Likert scaled items which you believe are the result of one continuously scaled latent factor which is not related to any other questions or factors in the analysis, then you would need to recode the ordinal responses as numeric, then simply run a one factor model, collect the factor scores, and rescale the factor scores as composite scores which reflect the original metric. As an example, consider q1, q2, q3, q4 which we believe reflect a single latent construct.



```
R Console
File Edit Misc Packages Windows Help
> subset.1 <- data.df[,7:10]; summary(subset.1)
      q1      q2      q3
agree   :415  agree   :426  agree   :449
disagree :425  disagree :429  disagree :417
strongly agree :451  strongly agree :437  strongly agree :417
strongly disagree:440  strongly disagree:439  strongly disagree:448
      q4
agree   :436
disagree :426
strongly agree :445
strongly disagree:424
> |
```

First, recode the responses into numbers which reflect the ordinality of the original responses (the words). This can be tricky sometimes because R cannot tell if "agree" should be a 1, 2, 3, etc. So, it's best to impose the values on specific labels using a fairly simple function which returns a subset of variables containing the recoded data. Here, extract the four columns of the original data and assign them to an object called 'subset.1', then we submit that object to the recoding function and re-assign the result to that same name (subset.1).

```

R Console
File Edit Misc Packages Windows Help
> recoding.4.point <- function(data){
+ new.data <- data.frame(matrix(rep(0, nrow(data)*ncol(data)), ncol = ncol(data)))
+ for (j in 1:ncol(data)){
+   for (i in 1:nrow(data)){
+     if(data[i,j] == "strongly disagree"){new.data[i,j] <- 1}
+     if(data[i,j] == "disagree"){new.data[i,j] <- 2}
+     if(data[i,j] == "agree"){new.data[i,j] <- 3}
+     if(data[i,j] == "strongly agree"){new.data[i,j] <- 4}
+   }
+ }
+ names(new.data) <- names(data)
+ return(new.data)
+ }
>
> head(subset.1)
      q1      q2      q3      q4
1 disagree strongly agree strongly agree disagree
2 strongly agree strongly disagree agree strongly agree
3 strongly agree strongly disagree agree strongly agree
4 disagree strongly disagree agree agree
5 strongly agree agree disagree strongly agree
6 disagree disagree strongly agree strongly agree
> subset.1 <- recoding.4.point(subset.1); head(subset.1)
  q1 q2 q3 q4
1  2  4  4  2
2  4  1  3  4
3  4  1  3  4
4  2  1  3  3
5  4  3  2  4
6  2  2  4  4
> |

```

Since we now have a properly re-coded (and numeric) version of the subset/data, we can apply a one-factor model. Of course, factor analysis assumes a linear relationship between each of the variables included in the factor model, so it is suggested that linearity be checked among each pair of variables included.

```
R Console
File Edit Misc Packages Windows Help
> fa.subset.1 <- factanal(x = subset.1, factors = 1, scores = "regression")
> fa.subset.1

Call:
factanal(x = subset.1, factors = 1, scores = "regression")

Uniquenesses:
  q1    q2    q3    q4
0.634 0.634 0.670 0.646

Loadings:
  Factor1
q1 0.605
q2 0.605
q3 0.574
q4 0.595

          Factor1
SS loadings      1.415
Proportion Var   0.354

Test of the hypothesis that 1 factor is sufficient.
The chi square statistic is 5.94 on 2 degrees of freedom.
The p-value is 0.0513
> |
```

Simply use the "\$scores" operator on the factor analysis object to extract the factor scores.

```
R Console
File Edit Misc Packages Windows Help
> subset.1.scores <- as.vector(fa.subset.1$scores)
> summary(subset.1.scores)
  Min.   1st Qu.   Median     Mean   3rd Qu.   Max.
-1.543000 -0.754000 -0.003227  0.000000  0.742600  1.538000
> |
```

You'll notice the factor scores have a mean of zero, so in order for them to have (semantic) meaning, we must convert them back into the scale of the original 1 - 4 responses. To do this, we will need three things; the new factor scores, the raw data, and the factor loadings. The factor loadings augment the meaning of the composite scores by providing insight into how each question contributed to the composite scores. Notice in this example each of the four questions contributed approximately equally to the composite scores (i.e. the loadings are roughly equal). However, if you have loadings which are not (roughly) equal, then you must communicate the loadings (and why they are important) to anyone interpreting or using the composite scores.

```

R Console
File Edit Misc Packages Windows Help
> fa1.loadings <- fa.subset.1$loadings[,1]
> fa1.loadings
      q1      q2      q3      q4
0.6049592 0.6049515 0.5742426 0.5946235
> |

```

Unfortunately, there is no base level function for calculating the weighted standard deviation (as there is for the weighted mean). Therefore, we create a small function for calculating the weighted standard deviation; needed below during the rescaling of the factor scores. The function takes the vector of values (x) and the weights (w), which are the loadings here, and returns the weighted standard deviation of the vector of values.

```

R Console
File Edit Misc Packages Windows Help
> weighted.sd <- function(x, w){
+   sum.w <- sum(w)
+   sum.w2 <- sum(w^2)
+   mean.w <- sum(x * w) / sum(w)
+   x.sd.w <- sqrt((sum.w / (sum.w^2 - sum.w2)) * sum(w * (x - mean.w)^2))
+   return(x.sd.w)
+ }
> |

```

The rescaling function below simply puts the scores back into the metric of the original questions. Keep in mind, some of the final scores may be slightly below '1' and some slightly above '4'; this is because we modeled the latent 'true scores'. Now, we have one set of scores or one variable, which contains each participant's score on the latent variable (subset.1 = ss1: q1 - q4).

```

R Console
File Edit Misc Packages Windows Help
> re.scale <- function(f.scores, raw.data, loadings){
+   fz.scores <- (f.scores + mean(f.scores)) / (sd(f.scores))
+   means <- apply(raw.data, 1, weighted.mean, w = loadings)
+   sds <- apply(raw.data, 1, weighted.sd, w = loadings)
+   grand.mean <- mean(means)
+   grand.sd <- mean(sds)
+   final.scores <- ((fz.scores * grand.sd) + grand.mean)
+   return(final.scores)
+ }
> final.scores.ss1 <- re.scale(subset.1.scores, subset.1, fa1.loadings)
> summary(final.scores.ss1)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.9813 1.7590  2.4990  2.5020 3.2340  4.0180
> |

```

We can go ahead and apply this same general procedure to the other two sets of questions (q5 - q9 and q10 - q14). However, because q5 - q9 have a 5 point Likert response format, we need a second recoding function to put those responses into numeric format.

```

R Console
File Edit Misc Packages Windows Help

> recoding.5.point <- function(data){
+ new.data <- data.frame(matrix(rep(0, nrow(data)*ncol(data)), ncol = ncol(data)))
+ for (j in 1:ncol(data)){
+   for (i in 1:nrow(data)){
+     if(data[i,j] == "lethargic"){new.data[i,j] <- 1}
+     if(data[i,j] == "not very active"){new.data[i,j] <- 2}
+     if(data[i,j] == "no difference"){new.data[i,j] <- 3}
+     if(data[i,j] == "more active"){new.data[i,j] <- 4}
+     if(data[i,j] == "hyperactive"){new.data[i,j] <- 5}
+   }
+ }
+ names(new.data) <- names(data)
+ return(new.data)
+ }
> subset.2 <- recoding.5.point(data.df[,11:15]); summary(subset.2)
      q5      q6      q7      q8      q9
Min.   :1.000  Min.   :1.000  Min.   :1.00  Min.   :1.000  Min.   :1.000
1st Qu.:2.000  1st Qu.:2.000  1st Qu.:2.00  1st Qu.:2.000  1st Qu.:2.000
Median :3.000  Median :3.000  Median :3.00  Median :3.000  Median :3.000
Mean   :3.005  Mean   :3.024  Mean   :2.99  Mean   :2.979  Mean   :2.969
3rd Qu.:4.000  3rd Qu.:4.000  3rd Qu.:4.00  3rd Qu.:4.000  3rd Qu.:4.000
Max.   :5.000  Max.   :5.000  Max.   :5.00  Max.   :5.000  Max.   :5.000
> subset.3 <- recoding.4.point(data.df[,16:20]); summary(subset.3)
      q10      q11      q12      q13      q14
Min.   :1.000  Min.   :1.000  Min.   :1.000  Min.   :1.000  Min.   :1.000
1st Qu.:1.000  1st Qu.:1.000  1st Qu.:1.000  1st Qu.:1.000  1st Qu.:1.000
Median :2.000  Median :2.000  Median :2.000  Median :2.000  Median :2.000
Mean   :2.487  Mean   :2.497  Mean   :2.499  Mean   :2.495  Mean   :2.481
3rd Qu.:4.000  3rd Qu.:4.000  3rd Qu.:4.000  3rd Qu.:4.000  3rd Qu.:4.000
Max.   :4.000  Max.   :4.000  Max.   :4.000  Max.   :4.000  Max.   :4.000
> |

```

Next, we create a single function which will take the numeric data and apply the 1 factor model, extract the factor scores, extract the factor loadings, and apply the re-scaling function. This function returns a list object which includes two elements: the rescaled scores and the factor loadings.

```

R Console
File Edit Misc Packages Windows Help

> get.scores.fun <- function(data){
+ fact <- factanal(data, factors = 1, scores = "regression")
+ f.scores <- fact$scores[,1]
+ f.loads <- fact$loadings[,1]
+ rescaled.scores <- re.scale(f.scores, data, f.loads)
+ output.list <- list(rescaled.scores, f.loads)
+ names(output.list) <- c("rescaled.scores", "factor.loadings")
+ return(output.list)
+ }
> |

```

Now, we can apply the above function to subset.2 and subset.3.


```
R Console
File Edit Misc Packages Windows Help
> scores.and.loadings.2 <- get.scores.fun(subset.2)
> scores.and.loadings.2$factor.loadings
      q5      q6      q7      q8      q9
0.6280197 0.6445726 0.6321581 0.6234082 0.6199569
> scores.and.loadings.3 <- get.scores.fun(subset.3)
> scores.and.loadings.3$factor.loadings
      q10      q11      q12      q13      q14
0.6160466 0.6226077 0.6152795 0.6227423 0.5866833
> |
```

Notice the factor loadings from above are all roughly equal. Next, we can extract the rescaled factor scores.

```
R Console
File Edit Misc Packages Windows Help
> final.scores.ss2 <- scores.and.loadings.2$rescaled.scores
> summary(final.scores.ss2)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 1.007  2.200   3.001   2.994   3.797   4.992
> final.scores.ss3 <- scores.and.loadings.3$rescaled.scores
> summary(final.scores.ss3)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.9663 1.8020  2.4270  2.4920  3.1920  4.0340
> |
```

Now, we can create a data frame which contains just the composite scores for each subset or section of the questionnaire.

```
R Console
File Edit Misc Packages Windows Help
> composite.data <- data.frame(final.scores.ss1, final.scores.ss2, final.scores.ss3)
> names(composite.data) <- c("composite.1", "composite.2", "composite.3")
> summary(composite.data)
  composite.1      composite.2      composite.3
  Min.   :0.9813  Min.   :1.007    Min.   :0.9663
 1st Qu.:1.7589  1st Qu.:2.200    1st Qu.:1.8023
  Median :2.4989  Median :3.001    Median :2.4269
  Mean   :2.5020  Mean   :2.994    Mean   :2.4919
 3rd Qu.:3.2340  3rd Qu.:3.797    3rd Qu.:3.1919
  Max.   :4.0184  Max.   :4.992    Max.   :4.0337
> |
```

Keep in mind, because the three composite score variables are likely to be related, we could have chosen to run a single factor analysis specifying three latent factors (rather than doing three separate factor analyses). However, if a single model was applied, each question would have a loading for each latent factor and those loadings might be substantial. If those cross-loadings were substantial, then they might call into question the factor structure (i.e. question 2 was *supposed* to load on factor 1, but instead loaded most on factor 3....). Furthermore, if we know these three latent variables, represented by the three composite score vectors, supported a global or general factor in a hierarchical fashion; then we

would use these three composite score vectors in another one factor model to calculate the composite scores for that general factor.

Conclusions

Generating composite scores using weighted factor scores is an extremely useful skill to have in one's repertoire. The composite scores can be used as independent or dependent variables in more traditional analysis (e.g. linear regression). However, the example above provides only an introduction to calculating these composite scores. When data does not display the necessary linear relationship required of factor analysis, one might explore the use of correspondence analysis, optimal scaling, or data transformations. The best defense against violations of assumptions, such as linearity, are a sound design and careful planning which can often ensure the data one collects is capable of providing the information one is seeking.

References and Resources

Organization for Economic Co-operation and Development (OECD). (2008). *Handbook on Constructing Composite Indicators*. <http://composite-indicators.jrc.ec.europa.eu/Handbook.htm>

Statistics Canada. (2010). *Survey Methods and Practices*. Ottawa, Canada: Minister of Industry. <http://www.statcan.gc.ca/bsolc/olc-cel/olc-cel?lang=eng&catno=12-587-X>