The Gifi Package for Categorical Multivariate Analysis in R

Patrick Mair
Harvard University

Jan de Leeuw
University of California, Los Angeles

Patrick Groenen
Erasmus University Rotterdam

Abstract

This package vignette is an update and extension of the paper by published in the Journal of Statistical Software. Homogeneity analysis combines the idea of maximizing the correlations between variables of a multivariate data set with that of optimal scaling. In this article we present methodological and practical issues of the R package **homals** which performs homogeneity analysis and various extensions. By setting rank constraints nonlinear principal component analysis can be performed. The variables can be partitioned into sets such that homogeneity analysis is extended to nonlinear canonical correlation analysis or to predictive models which emulate discriminant analysis and regression models. For each model the scale level of the variables can be taken into account by setting level constraints. All algorithms allow for missing values.

Keywords: Gifi methods, optimal scaling, homogeneity analysis, correspondence analysis, nonlinear principal component analysis, nonlinear canonical correlation analysis, homals, R.

1. Introduction

2. Gifi Methods for Categorical Multivariate Analysis

2.1. The Gifi Loss Function

In this section we give the very general definition of the Gifi loss function. We restrict our elaborations to the very basic expressions. For simplicity, we are not incorporating the formulation for missing data even though the package allows for this. A more detailed and slightly more technical formulation can be found in De Leeuw and Mair (2009).

Homogeneity analysis is based on the criterion of minimizing the departure from homogeneity. This departure is measured by a loss function. To write the corresponding basic equations the following definitions are needed. For $i = 1, \ldots, n$ objects, data on m (categorical) variables are collected where each of the $j = 1, \ldots, m$ variable takes on k_j different values (their *levels* or *categories*). We code them using $n \times k_j$ binary *indicator matrices* G_j, i.e., a matrix of dummy variables for each variable. The whole set of indicator matrices can be collected in a block matrix

$$G \triangleq \left[G_1 \;\vdots\; G_2 \;\vdots\; \cdots \;\vdots\; G_m \right]. \tag{1}$$

For convenience we introduce $D_j = G_j' G_j$ as the $k_j \times k_j$ diagonal matrix with the (marginal) frequencies of variable j in its main diagonal.

Let X be the unknown $n \times p$ matrix containing the coordinates (*object scores*) of the object projections into $\mathbb{R}^p$. Furthermore, let Y_j be the unknown $k_j \times p$ matrix containing the coordinates of the category projections into the same p-dimensional space (*category quantifications*). The problem of finding these solutions can be formulated by means of the following loss function to be minimized:

$$\sigma(X; Y_1, \ldots, Y_m) = \frac{1}{m} \sum_{j=1}^{m} \mathbf{tr}(X - G_j Y_j)'(X - G_j Y_j) \tag{2}$$

We use the normalization $X'X = I$ in order to avoid the trivial solution $X = 0$ and $Y_j = 0$. Note that from an analytical point of view the loss function represents the sum-of-squares of $(X - G_j Y_j)$ which obviously involves the object scores and the category quantifications. Thus, we minimize simultaneously over X and Y_j.

This minimization problem is solved by the iterative *alternating least squares algorithm* (ALS; sometimes quoted as *reciprocal averaging algorithm*). At iteration $t = 0$ we start with arbitrary object scores $X^{(0)}$. Each iteration t consists of three steps:

1. Update category quantifications: $Y_j^{(t+1)} = D_j^{-1} G_j' X^{(t)}$ for $j = 1, \ldots, m$
2. Update object scores: $\tilde{X}^{(t+1)} = \sum_{j=1}^{m} G_j Y_j^{(t+1)}$
3. Normalization: $X^{(t+1)} = \sqrt{n}\tilde{X}^{(t+1)} \left(\tilde{X}^{(t+1)\prime} \tilde{X}^{(t+1)} \right)^{-\frac{1}{2}}$

The algorithm stops when the decrease in the loss function gets below a prespecified level ε.

These are the ingredients we need in order to formulate important variants of the general Gifi model. In the following section we start with the most important one: PRINCALS.

3. Categorical Principal Component Analysis

Standard PCA was developed for metric variables and assumes that the relationships between the variables are linear. PCA aims to explain variance. It can be solved by an eigenvalue decomposition of the corresponding correlation (or covariance) matrix: the eigenvalues represent the variances of the principal component scores and the eigenvectors give the vectors of weights (loadings). PCA is targeted on explaining variance and typically we are aiming to find a satisfactory solution (in terms of the explained amount of variance) in a low-dimensional space.

From a homogeneity analysis perspective, the PCA problem can be stated as follows:

In many fields researchers often have to deal with categorical variables such as Likert items in the Social Sciences. Running a PCA on a set of items implies treating them as metric, i.e. the distances between categories (i.e. 1–2, 2–3, 3–4, etc.) have to be constant across categories. Together with the linear relationship assumption, these requirements are often not fulfilled in practice. In addition, the concept of variance only applied to metric variables.

If we want to treat the variables on an ordinal scale, we have two options: run an ordinal factor analysis (FA) based on polychoric correlations as implemented in the `fa.poly()` function in the **psych** package, or run a categorical (ordinal) PCA as presented here. Apart from the conceptual differences between FA and PCA, the advantage of PRINCALS over polychoric FA is that we do not have to pose any underlying distribution assumption on our data, whereas a polychoric (or tetrachoric in the binary case) correlation assumes that the categories are realizations of an underlying latent normal distribution.

PRINCALS does not only allow for ordinal input data, it can be also applied to mixed scale levels: some variables can be metric, some variables ordinal, some variables nominal.

In **homals** there is the `princals()` function which performs NLPCA. Since NLPCA is just a rank-1 restricted version of general homogeneity analysis, internally `princals()` uses `homals()` as engine. An effort was made to make the PRINCALS output as PCA-like (i.e. `princomp()`-like) as possible in terms of comparable eigenvalues, explaned amount of variance on each dimension, and loadings.

Standard PCA is solved by an eigenvalue decomposition of the input correlation matrix R based on the original data which gives us eigenvalue vector λ of length m. Subsequently, the amount of explained variance for each dimension can be computed by dividing each eigenvalue by the sum of the m eigenvalues. One of the cores outputs of any Gifi model is that it provides a "new" data matrix where the original categories are optimally scaled for each dimension. Now can now compute the correlation matrix R^* on the new data matrix (it does not matter which one we use since the p matrices are linearly dependent) and perform an eigenvalue decomposition. This gives us the eigenvalue vector λ^* of length m. As above, we can compute the amount of explained variance for each of the m dimensions. In addition, in order to evaluate the amount of "improvement" of NLPCA over PCA we can compute the eigenvalue ratio, e.g. for the first dimension we have λ_1^*/λ_1. This gives us a measure for the violations of equidistance and linearity in our original data.

Regarding the loadings, in standard PCA they are normed to $\|w\|^2 = 1$. In order to make the loadings w^* from NLPCA comparable to standard PCA, they need to be normalized the same way. The `princals()` function performs all these computations internally and returns the p eigenvalues based on the R^*, the amount of variance explained for each of the p dimensions, and the standardized loadings.

Through the `level` argument the user can specify the scale levels of the variables (`"ordinal" as default`. If all variables are set to `"numerical"`, PRINCALS mimics standard PCA. In terms of plotting possibilities, a generic plot function allows for a loadings plot (default), a scree plot, transformation plots, and a biplot by specifying the `plot.type` argument accordingly.

3.1. Rotation and Goodness-Of-Fit

The package also offers options for rotating a PRINCALS solution. Note that "rotation" is a concept developed within factor analysis (FA) by transforming the factor loadings through a rotation matrix and computing the corresponding rotated factor scores afterwards. Strictly speaking, PCA eigenvectors of the correlation/covariance matrix are not loadings (even though we just called them “loadings”) above. Loadings within an FA context are eigenvectors scaled by the square roots of the corresponding eigenvalues. For unrotated PCA/PRINCALS solutions this difference is not so important, but if we want to rotate a solution the PCA/PRINCALS “loadings” need to be re-scaled such that they are loadings in the FA sense. Thus, we can apply standard `varimax()` and `promax()` rotations as provided in **stats**. In order to get the rotated principal scores, we multiply unrotated matrix of the PRINCALS PC-scores with the generalized inverse of the rotated loadings matrix.

3.2. Example: ABC Company

Now we show an ordinal PCA example on the ABC dataset which reproduces the analysis in Ferrari and A. (2012). ABC is a ficticious company which launched a customer satisfaction survey. In this analysis we use six items, each of them on a 5-point Likert scale, covering certain aspects of customer satisfaction: equipment, sales support, technical support, training, purchase, and pricing.

First, we start with a full-dimensional PRINCALS solution and examine the scree plot.

```
ABC6 <- ABC[, 6:11]
fitfull <- princals(ABC6, ndim = 6)
fitfull

## Call: princals(data = ABC6, ndim = 6)
##
## Loss: 0.000667735
## Number of iterations: 19
##
## Eigenvalues:
##  Comp.1  Comp.2  Comp.3  Comp.4  Comp.5  Comp.6
## 2.31083 1.00248 0.76242 0.73749 0.67682 0.50996

summary(fitfull)

##
## Loadings:
##           Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6
## equipment  0.275 -0.744 -0.371 -0.411  0.243  0.003
## sales      0.458 -0.091  0.504 -0.252 -0.438  0.517
## technical  0.409  0.035 -0.611  0.455 -0.508  0.025
## training   0.443  0.273 -0.043  0.266  0.695  0.422
## purchase   0.361  0.567 -0.190 -0.623 -0.022 -0.349
## pricing    0.469 -0.199  0.445  0.318  0.089 -0.658
##
## Variance accounted for:
##                     Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6
## VAF (%)              38.51  16.71  12.71  12.29  11.28    8.5
## Cumulative VAF (%)   38.51  55.22  67.93  80.22  91.50  100.0
```

The scree plot is given in in the left panel of Figure **??**.

Second, we fit a two-dimensional ordinal solution. The loadings plot is given in Figure 1 (right panel).

```
fit2d <- princals(ABC6, ndim = 2)
fit2d

## Call: princals(data = ABC6, ndim = 2)
##
## Loss: 0.0005475663
## Number of iterations: 38
##
## Eigenvalues:
##  Comp.1  Comp.2
## 2.94546 0.85430

summary(fit2d)

##
## Loadings:
##           Comp.1 Comp.2
## equipment  0.456 -0.032
## sales      0.374 -0.652
## technical  0.392  0.326
## training   0.393  0.491
## purchase   0.397  0.270
## pricing    0.432 -0.393
##
## Variance accounted for:
##                     Comp.1 Comp.2
## VAF (%)              49.09  14.24
## Cumulative VAF (%)   49.09  63.33
```

We see that we explain around 63% of the variance.

```
op <- par(mfrow = c(1,2))
plot(fitfull, plot.type = "screeplot")
plot(fit2d)
par(op)
```

Now we fit a one-dimensional ordinal PCA.

```
fit1d <- princals(ABC6, ndim = 1)
fit1d

## Call: princals(data = ABC6, ndim = 1)
```

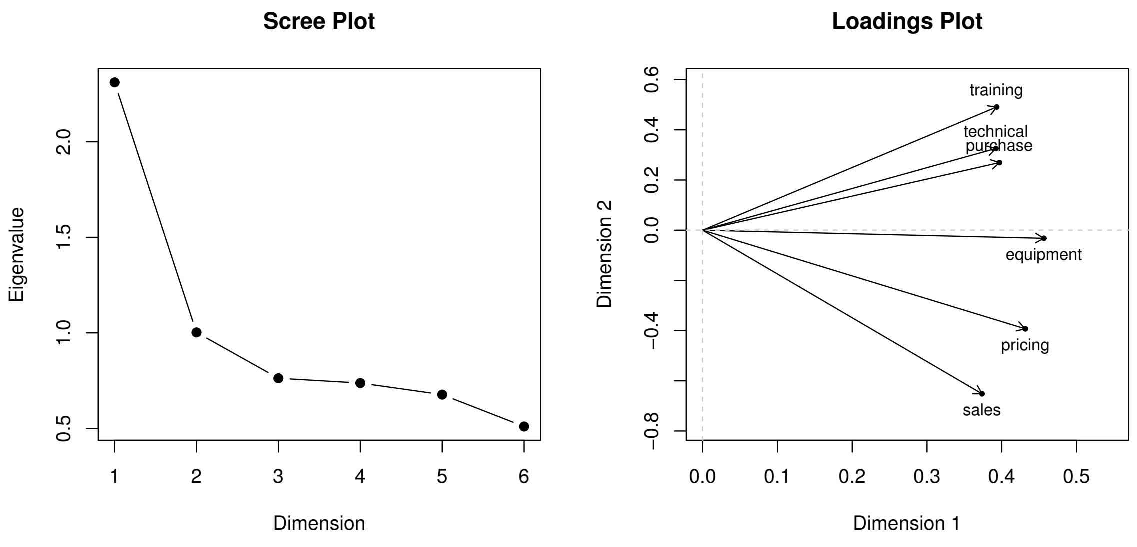

Figure 1: Left panel: Scree plot for full-dimensional PRINCALS. Right panel: Loadings plot for two-dimensional solution.

```
##
## Loss: 0.0004027625
## Number of iterations: 10
##
## Eigenvalues:
##  Comp.1
## 2.98414

summary(fit1d)

##
## Loadings:
##           Comp.1
## equipment  0.448
## sales      0.387
## technical  0.390
## training   0.395
## purchase   0.396
## pricing    0.428
##
## Variance accounted for:
##                     Comp.1
## VAF (%)              49.74
## Cumulative VAF (%)  49.74
```

Let us compare it with the outcome of a standard PCA solution using `princomp()` and compute the ratio of the first eigenvalues.

```
ABC6m <- sapply(ABC6, function(x) as.numeric(levels(x))[x])
fitpc <- princomp(ABC6m)
fitpc

## Call:
## princomp(x = ABC6m)
##
## Standard deviations:
##    Comp.1    Comp.2    Comp.3    Comp.4    Comp.5    Comp.6
## 1.6732946 0.9832548 0.8268262 0.7708195 0.6209486 0.5968660
##
##  6  variables and  208 observations.

fit1d$eigenvalues/(fitpc$sdev^2)[1]       ## eigenvalue ratio NLPCA/PCA

##   Comp.1
## 1.065798
```

The eigenvalue ratio of suggests 1.07 a slight improvement of NLPCA over standard PCA. This implies that the response categories are approximately equidistant and the relationship between the variables is not far from linear. If we want to mimic standard PCA with PRINCALS, we declare all the variables as numeric.

```
fit1dm <- princals(ABC6, ndim = 1, level = "numerical")
fit1dm

## Call: princals(data = ABC6, ndim = 1, level = "numerical")
##
## Loss: 0.0004178631
## Number of iterations: 6
##
## Eigenvalues:
##  Comp.1
## 2.87104

fit1dm$eigenvalues/(fitpc$sdev^2)[1]       ## eigenvalue ratio NLPCA/PCA

##   Comp.1
## 1.025404
```

The size of the eigenvalue ratio decreased since we are essentially doing the same thing. Finally, we can also abandon the order assumption in the response categories and treat all variables as nominal. This is the least restrictive of our one-dimensional PRINCALS models.

```
fit1dc <- princals(ABC6, ndim = 1, level = "nominal")
fit1dc

## Call: princals(data = ABC6, ndim = 1, level = "nominal")
##
## Loss: 0.0004026197
## Number of iterations: 11
##
## Eigenvalues:
##  Comp.1
## 2.98521
```

We see that the nominal PCA leads pretty much to the same fit as the ordinal version which suggests that the ordinal scale level for the variables holds.

4. Discussion

In this paper theoretical foundations of the methodology used in the **homals** package are elaborated and package application and visualization issues are presented. Basically, **homals** covers the techniques described in Gifi (1990): Homogeneity analysis, NLCCA, predictive models, and NLPCA. It can handle missing data and the scale level of the variables can be taken into account. The package offers a broad variety of real-life datasets and furthermore provides numerous methods of visualization, either in a two-dimensional or in a three-dimensional way. Future enhancements will be to replace indicator matrices by more general B-spline bases and to incorporate weights for observations. To conclude, **homals** provides flexible, easy-to-use routines which allow researchers from different areas to compute, interpret, and visualize methods belonging to the Gifi family.

References

De Leeuw J, Mair P (2009). "Homogeneity Analysis in R: The Package homals." *Journal of Statistical Software*, **31(4)**, 1–21. URL http://www.jstatsoft.org/v31/i04/.

Ferrari PA, A B (2012). "Nonlinear principal component analysis." In RS Kenett, S Salini (eds.), *Modern Analysis of Customer Surveys with Applications in R*, pp. 333–356. Wiley, New York.

Gifi A (1990). *Nonlinear Multivariate Analysis*. Wiley, Chichester, England.