Visualization of a correlation matrix with the Correlplot package

Jan Graffelman - Universitat Politècnica de Catalunya
Jan de Leeuw - University of California Los Angeles

2023-02-01

Introduction

This documents gives some instructions on how to create graphical representations of a correlation matrix in the statistical environment R with package `Correlplot`, using a variety of different statistical methods (Graffelman and De Leeuw (2022)). We use principal component analysis (PCA), multidimensional scaling (MDS), principal factor analysis (PFA), weighted alternating least squares (WALS), correlograms (CRG) and corrgrams to produce displays of correlation structure. The next section shows how to use the functions of the package in order to create the different graphical representations. The computation of goodness-of-fit statistics is also addressed. All methods are illustrated on a single data set, the wheat kernel data introduced below.

Graphical representations of a correlation matrix

We first load some packages we will use:

```
library(calibrate)
library(corrplot)
library(Correlplot)
```

Throughout this vignette, we will use the wheat kernel data set taken from the UCI Machine Learning Repository (https://archive.ics.uci.edu/ml/datasets/seeds) in order to illustrate the different plots. The wheat kernel data (Charytanowicz et al. (2010)) consists of 210 wheat kernels, of which the variables *area* (A), *perimeter* (P), *compactness* ($C = 4 * \pi * A/P^2$), *length*, *width*, *asymmetry coefficient* and *groove* (length of kernel groove) were registered. There are 70 kernels of each of three varieties *Kama*, *Rosa* and *Canadian*; here we will only use the kernels of variety *Kama*. The data is made available with:

```
data("Kernels")
X <- Kernels[Kernels$variety==1,]
X <- X[,-8]
head(X)
#>    area perimeter compactness length width asymmetry groove
#> 1 15.26     14.84      0.8710  5.763 3.312     2.221  5.220
#> 2 14.88     14.57      0.8811  5.554 3.333     1.018  4.956
#> 3 14.29     14.09      0.9050  5.291 3.337     2.699  4.825
#> 4 13.84     13.94      0.8955  5.324 3.379     2.259  4.805
#> 5 16.14     14.99      0.9034  5.658 3.562     1.355  5.175
#> 6 14.38     14.21      0.8951  5.386 3.312     2.462  4.956
```

The correlation matrix of the variables is given by:

```
R <- cor(X)
round(R,digits=3)
#>               area perimeter compactness length  width asymmetry groove
#> area         1.000     0.976       0.371  0.835  0.900    -0.050  0.721
```

```
#> perimeter     0.976      1.000       0.165  0.921  0.802    -0.054  0.794
#> compactness   0.371      0.165       1.000 -0.146  0.667     0.037 -0.131
#> length        0.835      0.921      -0.146  1.000  0.551    -0.037  0.866
#> width         0.900      0.802       0.667  0.551  1.000    -0.027  0.447
#> asymmetry    -0.050     -0.054       0.037 -0.037 -0.027     1.000 -0.011
#> groove        0.721      0.794      -0.131  0.866  0.447    -0.011  1.000
```

1. The corrgram

The corrgram (Friendly (2002)) is a tabular display of the entries of a correlation matrix that uses colour and shading to represent correlations. Corrgrams can be made with the fuction `corrplot`

```
corrplot(R, method="circle",type="lower")
```

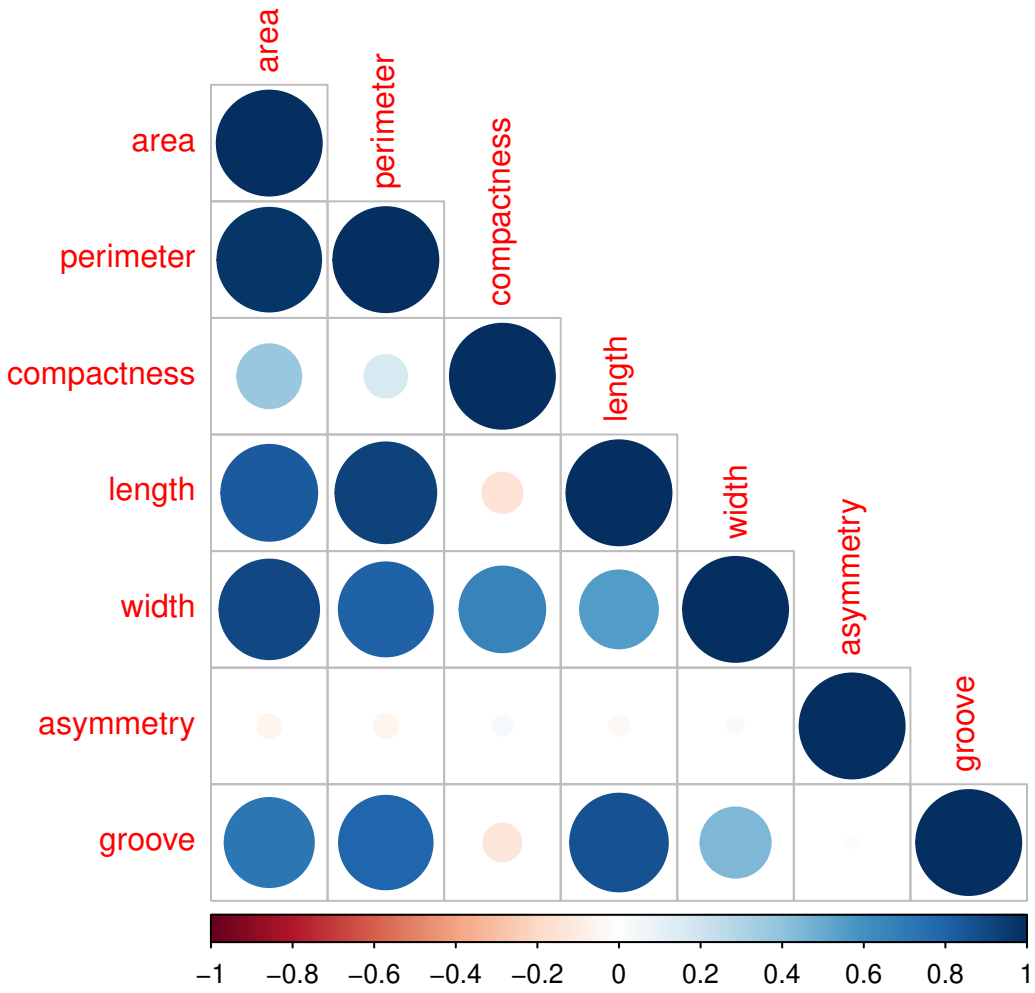

Figure 1: A corrgram of the wheat kernel data.

This shows most correlations are positive, and correlations with *asymmetry* are weak.

2. The correlogram

The correlogram (Trosset (2005)) represents correlations by the cosines between vectors.

```
theta.cos <- correlogram(R,xlim=c(-1.1,1.1),ylim=c(-1.1,1.1),main="CRG")
```

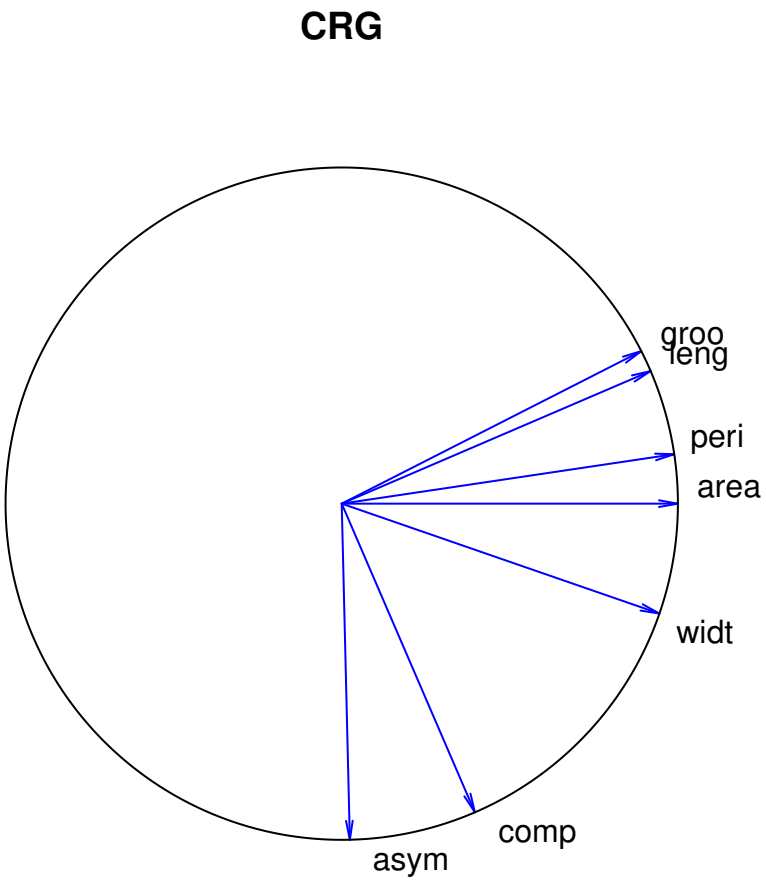

Figure 2: The correlogram of the wheat kernel data.

The vector `theta.cos` contains the angles (in radians) of each variable with respect to the positive x-axis. The approximation provided by these angles to the correlation matrix is calculated by

```
Rhat.cor <- angleToR(theta.cos)
```

The correlogram always perfectly represents the correlations of the variables with themselves, and these have a structural contribution of zero to the loss function. We calculate the root mean squared error (RMSE) of the approximation by using function `rmse`. We include the diagonal in the RMSE calculation by setting `omit.diagonal` to `FALSE`.

```
rmse.crg <- rmse(R,Rhat.cor,omit.diagonal=FALSE)
rmse.crg
#> [1] 0.2437535
```

This gives and RMSE of 0.2438, which shows this representation has a large amount of error. The correlogram can be modified by using a linear interpretation rule, rendering correlations linear in the angle (Graffelman (2013)). This representation is obtained by:

```
theta.lin <- correlogram(R,ifun="lincos",labs=colnames(R),xlim=c(-1.1,1.1),
                         ylim=c(-1.1,1.1),main="CRG")
```

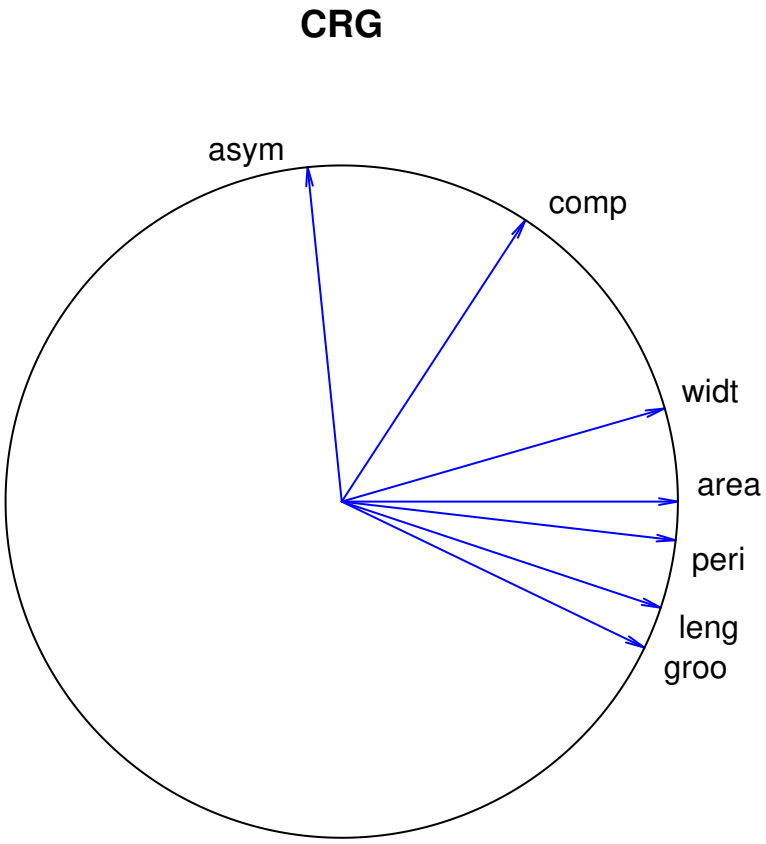

Figure 3: Linear correlogram of the wheat kernel data.

The approximation to the correlation matrix by using this linear interpretation function is calculated by

```
Rhat.corlin <- angleToR(theta.lin,ifun="lincos")
rmse.lin <- rmse(R,Rhat.corlin,omit.diagonal=FALSE)
rmse.lin
#> [1] 0.1667556
```

and this gives a RMSE of 0.1668. In this case, the linear representation is seen to improve the approximation.

3. The PCA biplot of the correlation matrix

We create a PCA biplot of the correlation matrix, doing the calculations for a PCA by hand, using the singular value decomposition of the (scaled) standardized data. Alternatively, standard R function `princomp` may be used to obtain the coordinates needed for the correlation biplot. We use function `bplot` from package `calibrate` to make the biplot:

```
n <- nrow(X)
Xt <- scale(X)/sqrt(n-1)
res.svd <- svd(Xt)
Fs <- sqrt(n)*res.svd$u # standardized principal components
Gp <- res.svd$v%*%diag(res.svd$d) # biplot coordinates for variables
bplot(Fs,Gp,colch=NA,collab=colnames(X),xlab="PC1",ylab="PC2",main="PCA")
```

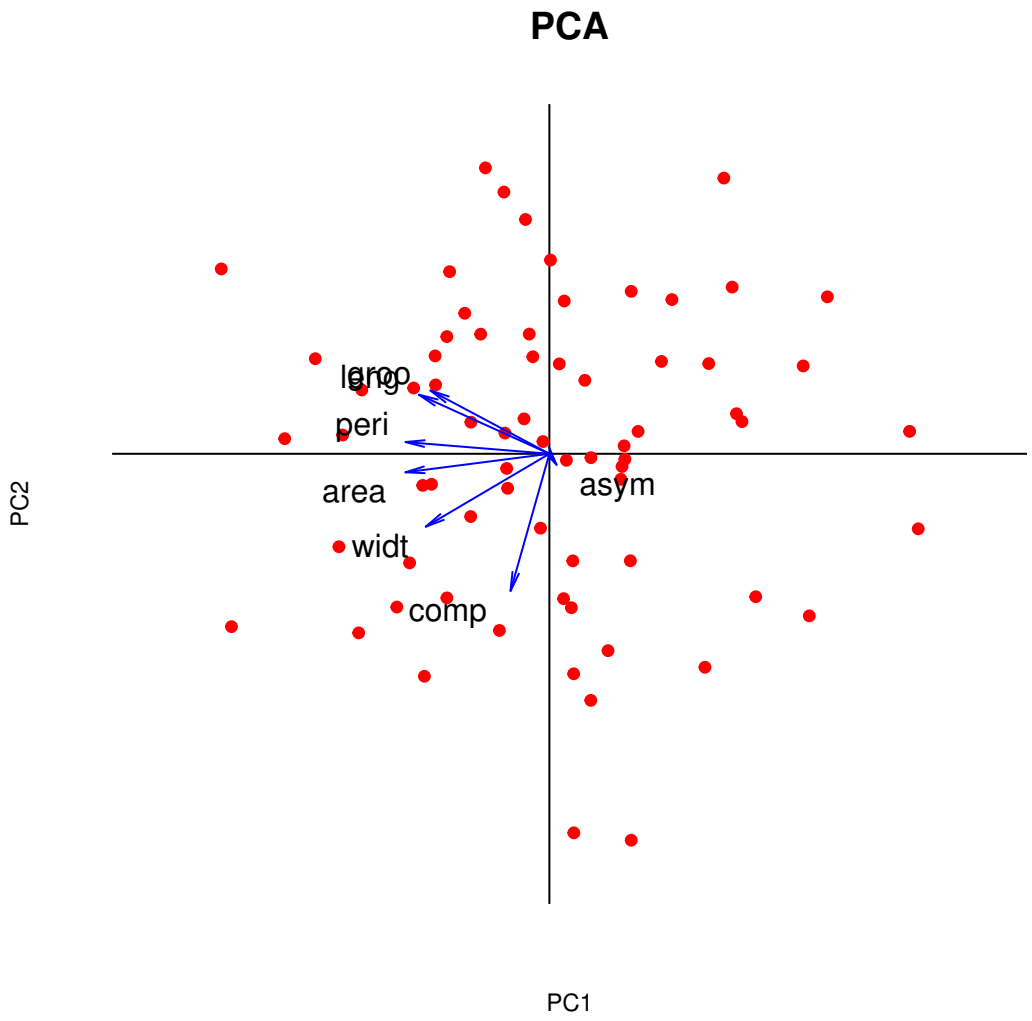

Figure 4: PCA biplot of the wheat kernel data.

The joint representation of kernels and variables emphasizes this is a biplot of the (standardized) data matrix. However, this plot is a *double biplot* because scalar products between variable vectors approximate the correlation matrix. We stress this by plotting the variable vectors only, and adding a unit circle:

```
bplot(Gp,Gp,colch=NA,rowch=NA,collab=colnames(X),xl=c(-1,1),
      yl=c(-1,1),main="PCA")
circle()
```

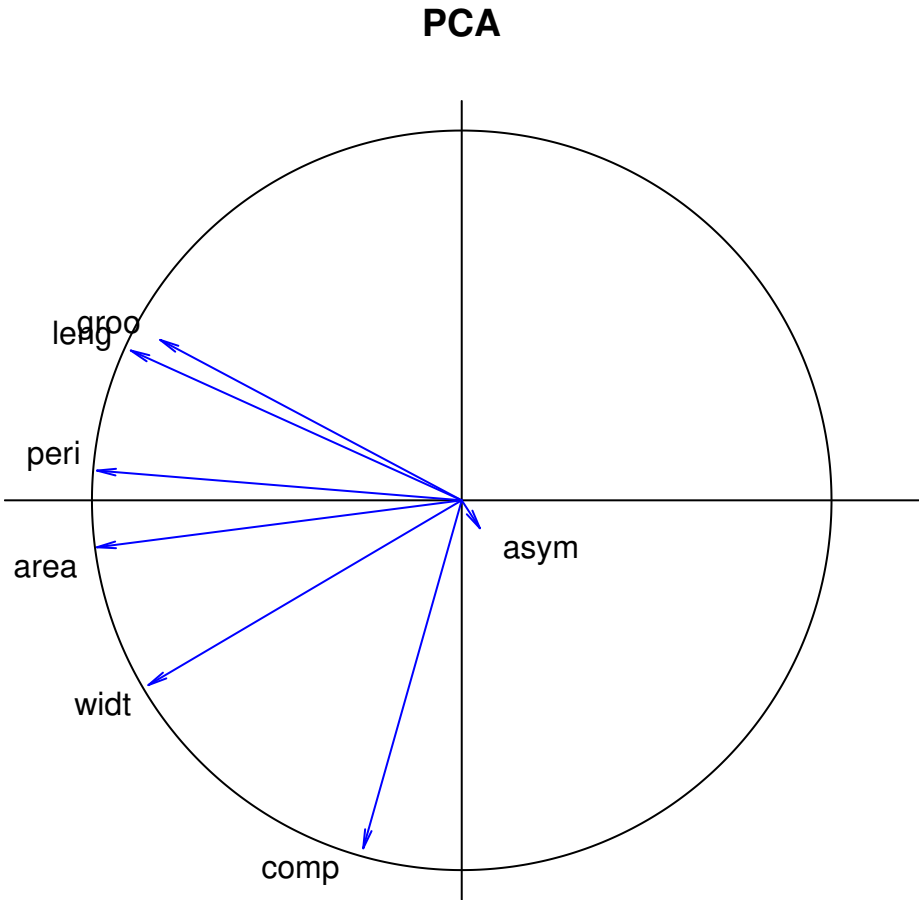

Figure 5: PCA biplot of the correlation matrix.

The PCA biplot of the correlation matrix can be obtained from a correlation-based PCA or also directly from the spectral decomposition of the correlation matrix. The rank two approximation, obtained by means of scalar products between vectors, is calculated by:

```
Rhat.pca <- Gp[,1:2]%*%t(Gp[,1:2])
```

In principle, PCA also tries to approximate the ones on the diagonal of the correlation matrix. These are included in the RMSE calculation by setting `omit.diagonal` to `FALSE`.

```
rmse.pca <- rmse(R,Rhat.pca,omit.diagonal=FALSE)
rmse.pca
#> [1] 0.145959
```

This gives a RMSE of 0.1460, which is an improvement over the previous correlograms. Function `rmse` can also be used to calculate the RMSE of each variable separately:

```
rmse(R,Rhat.pca,omit.diagonal=FALSE,per.variable=TRUE)
#>       area       peri       comp       leng       widt       asym       groo
#> 0.01403868 0.02158021 0.03255115 0.02480187 0.02151041 0.37610214 0.06982317
```

This shows that *asymmetry* has the worst fit.

4. The MDS map of a correlation matrix

We transform correlations to distances with the $\sqrt{2(1-r)}$ transformation, and use the `cmdscale` function from the `stats` package to perform metric multidimensional scaling. We mark negative correlations with a dashed red line.

```
Di <- sqrt(2*(1-R))
out.mds <- cmdscale(Di,eig = TRUE)
Fp <- out.mds$points
opar <- par(bty = "l")
plot(Fp[,1],Fp[,2],asp=1,xlab="First principal axis",
     ylab="Second principal axis",main="MDS")
textxy(Fp[,1],Fp[,2],colnames(R),cex=0.75)
par(opar)

ii <- which(R < 0,arr.ind = TRUE)

for(i in 1:nrow(ii)) {
  segments(Fp[ii[i,1],1],Fp[ii[i,1],2],
           Fp[ii[i,2],1],Fp[ii[i,2],2],col="red",lty="dashed")
}
```

We calculate distances in the map, convert these back to correlations:

```
Dest <- as.matrix(dist(Fp[,1:2]))
Rhat.mds <- 1-0.5*Dest*Dest
```

Again, correlations of the variables with themselves are perfectly approximated (zero distance), and we include these in the RMSE calculations:

```
rmse.mds <- rmse(R,Rhat.mds,omit.diagonal=FALSE)
rmse.mds
#> [1] 0.06837469
```

5. The PFA biplot of a correlation matrix

Principal factor analysis can be performed by the function `pfa` of package `Correlplot`. We extract the factor loadings.

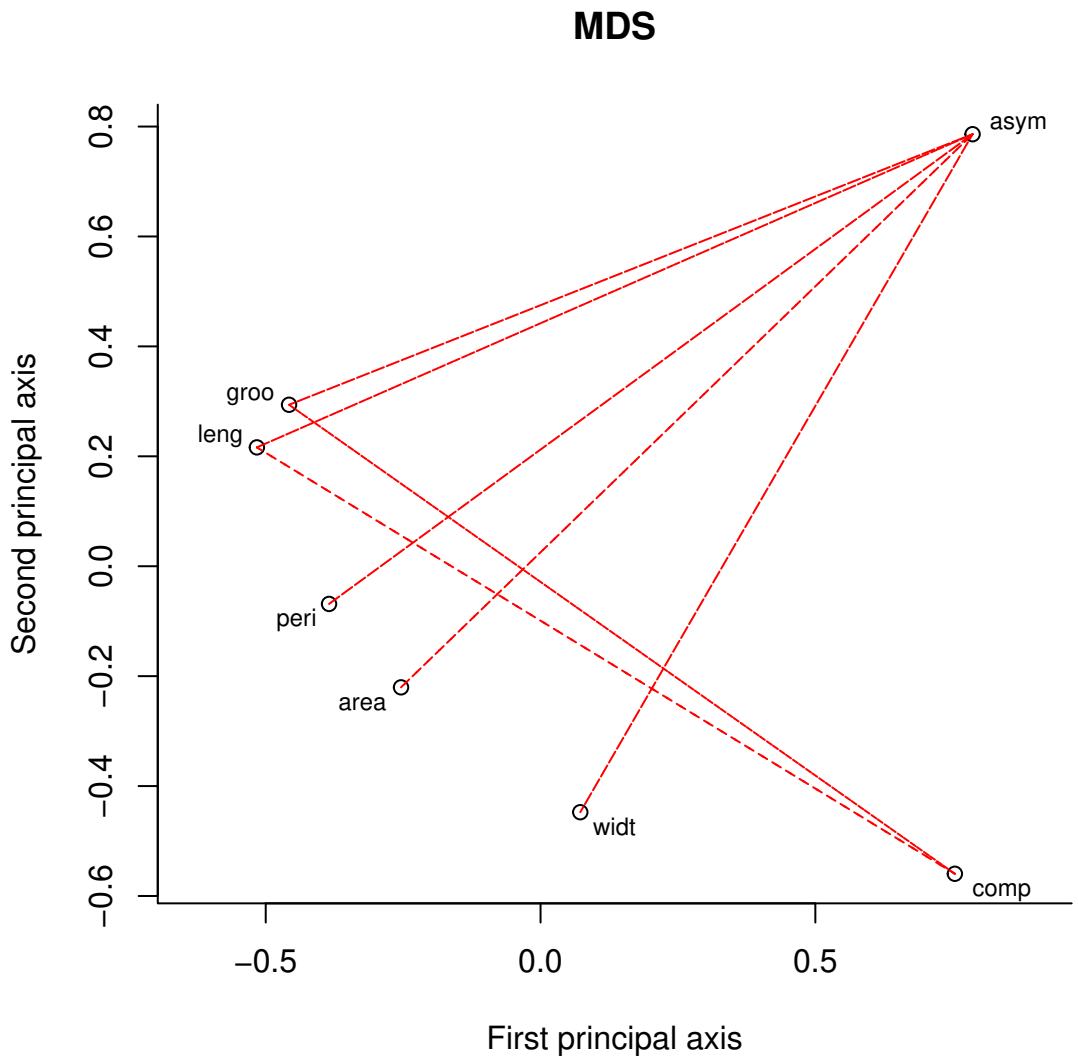

Figure 6: MDS map of the correlation matrix of the wheat kernel data.

```
out.pfa <- Correlplot::pfa(X,verbose=FALSE)
L <- out.pfa$La
```

The biplot of the correlation matrix obtained by PFA is in fact the same as what is known as a factor loading plot in factor analysis, to which a unit circle can be added. The approximation to the correlation matrix and its RMSE are calculated as:

```
Rhat.pfa <- L[,1:2]%*%t(L[,1:2])
rmse.pfa <- rmse(R,Rhat.pfa)
rmse.pfa
#> [1] 0.01119688
```

In this case, the diagonal is excluded, for PFA explicitly avoids fitting the diagonal. To make the factor loading plot, aka PFA biplot of the correlation matrix:

```
bplot(L,L,pch=NA,xl=c(-1,1),yl=c(-1,1),
     xlab="Factor 1",ylab="Factor 2",main="PFA",rowch=NA,
     colch=NA)
circle()
```

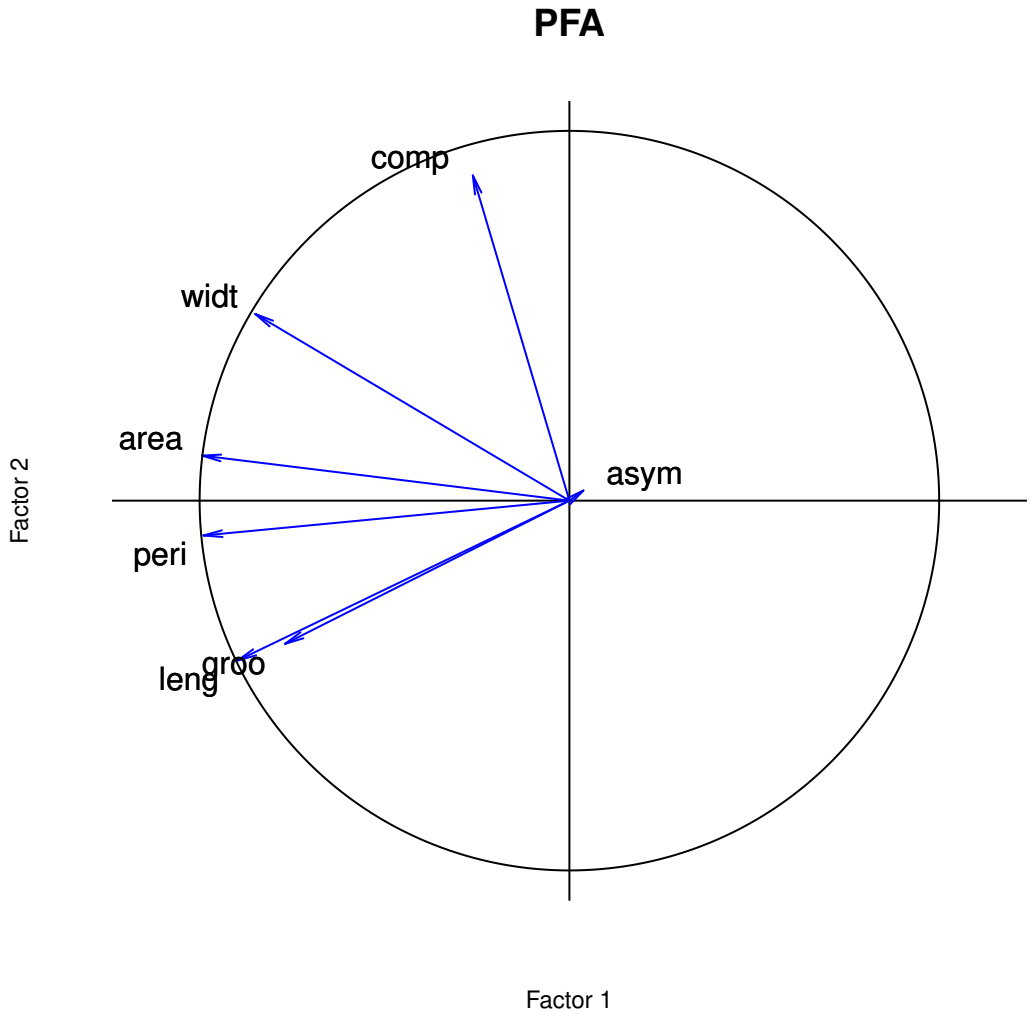

Figure 7: PFA biplot of the correlation matrix of the wheat kernel data.

The RMSE of the plot obtained by PFA is 0.0112, lower than the RMSE obtained by PCA. Note that variable *area* reaches the unit circle for having a communality of 1, or, equivalently, specificity 0, i.e. PFA produces

what is known as a *Heywood case* in factor analysis. The specificities are given by:

```
diag(out.pfa$Psi)
#>        area        peri        comp        leng        widt        asym        groo
#> 0.000000000 0.011494664 0.158097108 0.007885055 0.022509545 0.997799829 0.258768379
```

6. The WALS biplot of a correlation matrix

The correlation matrix can also be factored using weighted alternating least squares, avoiding the fit of the ones on the diagonal of the correlation matrix by assigning them weight 0, using function `ipSymLS` (De Leeuw (2006)).

```
W <- matrix(1,nrow(R),nrow(R))
diag(W) <- 0
Fp.als <- ipSymLS(R,w=W,eps=1e-15)
```

```
bplot(Fp.als,Fp.als,rowch=NA,colch=NA,collab=colnames(R),
      xl=c(-1.1,1.1),yl=c(-1.1,1.1),main="WALS")
circle()
```

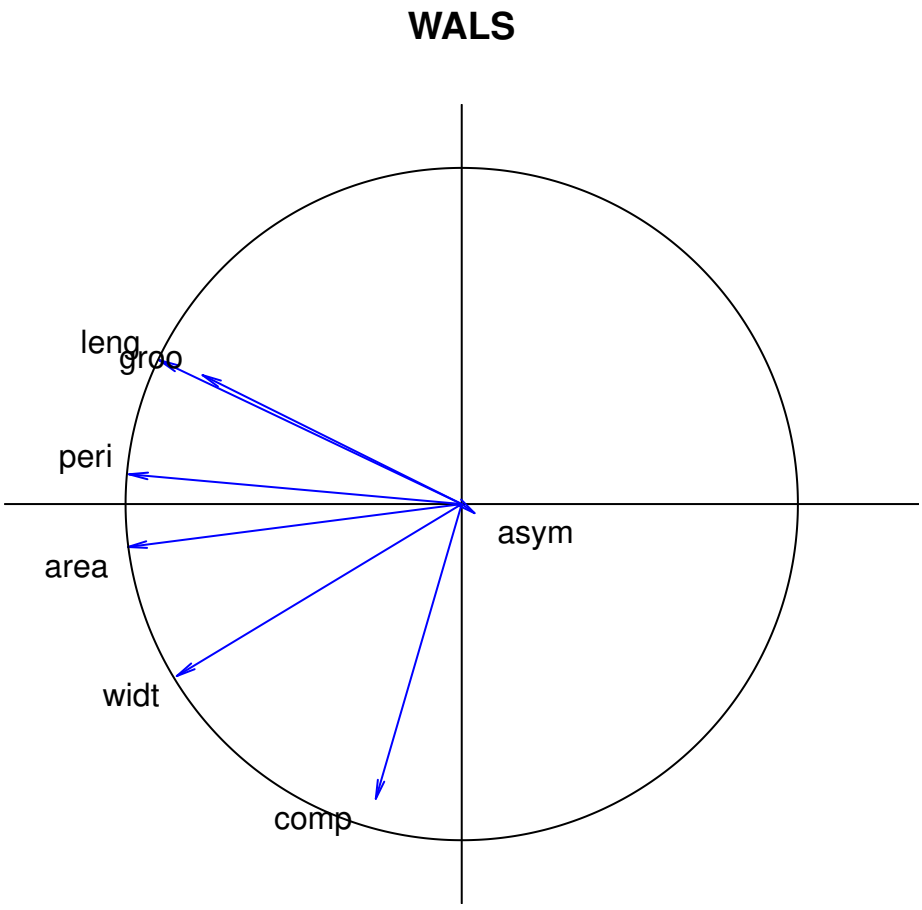

Figure 8: WALS biplot of the correlation matrix of the wheat kernel data.

Weighted alternating least squares has, in contrast to PFA, no restriction on the vector length. If the vector lengths in the biplot are calculated, then variable *area* is seen to just move out of the unit circle.

```
Rhat.wals <- Fp.als%*%t(Fp.als)
sqrt(diag(Rhat.wals))
```

```
#> [1] 1.00124368 0.99394213 0.91345321 0.99646265 0.99026217 0.04686397 0.86124152
rmse.als <- rmse(R,Rhat.wals)
rmse.als
#> [1] 0.01118619
```

The RMSE of the approximation obtained by WALS is 0.011186, slightly below the RMSE of PFA. The WALS low rank approximation to the correlation matrix can also be obtained by the more generic function `wAddPCA`, which allows for non-symmetric matrices and adjustments (see the next section). In order to get uniquely defined biplot vectors for each variable, corresponding to symmetric input, an eigendecomposition is applied to the fitted correlation matrix.

```
p <- nrow(R)
W <- matrix(1,p,p)
diag(W) <- 0
out.wals <- wAddPCA(R, W, add = "nul", verboseout = FALSE, epsout = 1e-10)
Rhat.wals <- out.wals$a%*%t(out.wals$b)
out.eig <- eigen(Rhat.wals)
Fp.adj <- out.eig$vectors[,1:2]%*%diag(sqrt(out.eig$values[1:2]))
rmse.als <- rmse(R,Rhat.wals)
rmse.als
#> [1] 0.01118619
```

7. The WALS biplot using an adjustment of the correlation matrix

A scalar adjustment can be employed to improve the approximation of the correlation matrix, and the adjusted correlation matrix is factored for a biplot representation. That means we seek the factorization

$$\mathbf{R}_a = \mathbf{R} - \delta\mathbf{J} = \mathbf{G}\mathbf{G}',$$

where both δ and $\mathbf{G}$ are chosen such that the (weighted) residual sum of squares is minimal. This problem is solved by using function `wAddPCA`.

```
p <- nrow(R)
W <- matrix(1,p,p)
diag(W) <- 0
out.wals <- wAddPCA(R, W, add = "one", verboseout = FALSE, epsout = 1e-10)
delta <- out.wals$delta[1,1]
Rhat <- out.wals$a%*%t(out.wals$b)
out.eig <- eigen(Rhat)
Fp.adj <- out.eig$vectors[,1:2]%*%diag(sqrt(out.eig$values[1:2]))
```

The optimal adjustment δ is 0.071. The corresponding biplot is shown below.

```
bplot(Fp.adj,Fp.adj,rowch=NA,colch=NA,collab=colnames(R),
      xl=c(-1.2,1.2),yl=c(-1.2,1.2),main="WALS adjusted")
circle()
```

Note that, when calculating the fitted correlation matrix, adjustment δ is added back. The fitted correlation matrix and its RMSE are now calculated as:

```
Rhat.adj <- Fp.adj%*%t(Fp.adj)+delta
rmse.adj <- rmse(R,Rhat.adj)
rmse.adj
#> [1] 0.005560242
```

This gives RMSE 0.0056. This is smaller than the RMSE obtained by WALS without adjustment. We summarize the values of the RMSE of all methods in a table below:

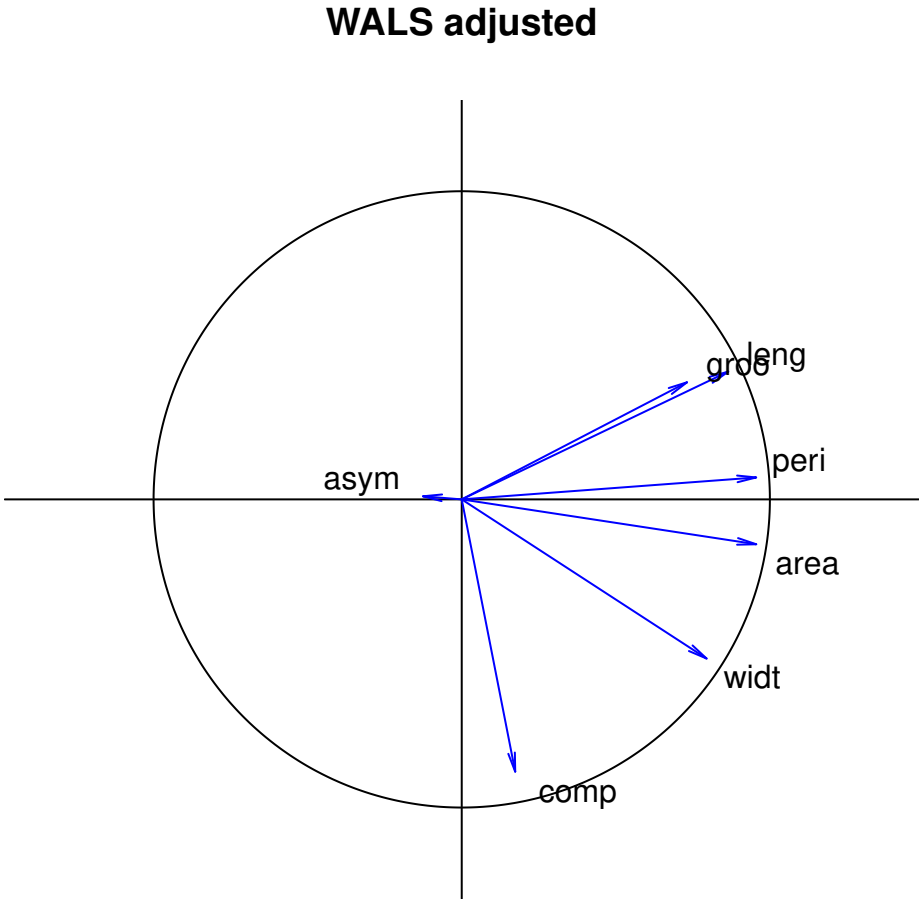

Figure 9: WALS biplot of the correlation matrix of the wheat kernel data, with the use of an additive adjustment.

```
rmsevector <- c(rmse.crg,rmse.lin,rmse.pca,rmse.mds,rmse.pfa,rmse.als,rmse.adj)
methods <- c("CRG (cos)","CRG (lin)","PCA","MDS",
"PFA","WALS R","WALS Radj")
results <- data.frame(methods,rmsevector)
results <- results[order(rmsevector),]
results
#>     methods  rmsevector
#> 7 WALS Radj 0.005560242
#> 6    WALS R 0.011186190
#> 5       PFA 0.011196885
#> 4       MDS 0.068374694
#> 3       PCA 0.145959040
#> 2 CRG (lin) 0.166755585
#> 1 CRG (cos) 0.243753461
```

A summary of RMSE calculations for four methods (PCA and WALS, both with and without δ adjustment) can be obtained by the functions `FitRwithPCAandWALS` and `rmsePCAandWALS`; the first applies the four methods to the correlation matrix, and the latter calculates the RMSE statistics for the four approximations. The bottom line of the table produced by `rmsePCAandWALS` gives the overall RMSE for each methods.

```
output <- FitRwithPCAandWALS(R,eps=1e-15)
rmsePCAandWALS(R,output)
#>         PCA  PCA-A   WALS WALS-A
#> area 0.0140 0.0535 0.0081 0.0043
#> peri 0.0216 0.0399 0.0088 0.0064
#> comp 0.0326 0.0602 0.0110 0.0054
#> leng 0.0248 0.0467 0.0067 0.0045
#> widt 0.0215 0.0639 0.0076 0.0068
#> asym 0.3761 0.1253 0.0181 0.0049
#> groo 0.0698 0.0695 0.0135 0.0061
#> All  0.1460 0.0706 0.0112 0.0056
```

References

Charytanowicz, M., J. Niewczas, P. Kulczycki, P. A. Kowalski, S. Lukasik, and S. Zak. 2010. "A Complete Gradient Clustering Algorithm for Features Analysis of x-Ray Images." In *Information Technologies in Biomedicine*, edited by Ewa Pietka and Jacek Kawa, 15–24. Berlin-Heidelberg: Springer-Verlag.

De Leeuw, J. 2006. "A Decomposition Method for Weighted Least Squares Low-Rank Approximation of Symmetric Matrices." *Department of Statistics, UCLA*. https://escholarship.org/uc/item/1wh197mh.

Friendly, M. 2002. "Corrgrams: Exploratory Displays for Correlation Matrices." *The American Statistician* 56 (4): 316–24. https://doi.org/10.1198/000313002533.

Graffelman, J. 2013. "Linear-Angle Correlation Plots: New Graphs for Revealing Correlation Structure." *Journal of Computational and Graphical Statistics* 22 (1): 92–106. https://doi.org/10.1080/15533174.2012.707850.

Graffelman, J., and J. De Leeuw. 2022. "Improved Approximation and Visualization of the Correlation Matrix." https://doi.org/10.48550/arXiv.2211.13150.

Trosset, M. W. 2005. "Visualizing Correlation." *Journal of Computational and Graphical Statistics* 14 (1): 1–19. https://doi.org/10.1198/106186005X27004.