2.3 | Orthogonal Factor Rotation

After calculating the factor loading matrix for a data matrix, the new step is to interpret the resulting factors. This step becomes rather challenging if any of the dimensions or rows in this matrix are complex; unfortunately, this is almost always the case. Thankfully, we can rotate a loading matrix so as to decrease and hopefully eliminate any complexities, thereby making the factor interpretation much easier.

2.3.1 | Visualizing Loadings in 2-D

The easiest way to understand the need for and the geometric foundation of *factor rotation* is to first visualize the complex loadings that make it necessary.

Consider again the two-factor representation of our example data.

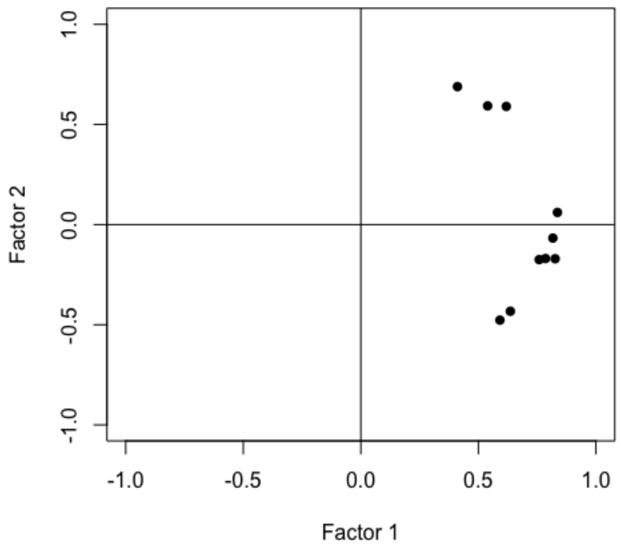
```
A <- pca(r = R, nfactors = 2, rotate = "none")$loadings[]
A
                      PC1         PC2
Hopeless        0.8168518 -0.06733209
Overwhelmed     0.5916281 -0.47691509
Exhausted       0.6358756 -0.43261569
Lonely          0.7854763 -0.16936567
Sad             0.8265731 -0.17045037
Depressed       0.8362622  0.06083607
Anxious         0.7585855 -0.17481308
SelfHarming     0.5393196  0.59271045
SuicidalThoughts 0.6186590  0.59011735
SuicidalAttempts 0.4110085  0.68894644
```

We can see that five of the ten dimensions in our example data are complex under this two-factor representation. This again makes interpreting and naming these factors rather challenging.

Rather than visualizing these loadings using a correlation plot, we can represent them in a two-dimensional space using each pair of loadings as a set of cartesian coordinates. The coordinates for our first dimension, for example, are $(x_1, y_1) = (0.817, -0.067)$.

In R, we can use the plot() function to generate this *factor loading plot*.

```
plot(A, pch = 16, xlab = "Factor 1", ylab = "Factor 2",
     xlim = c(-1,1), ylim = c(-1,1))
abline(h = 0, v = 0)
```



Given that factor loadings are essentially correlations and, as such, are bound between -1 and 1, we use the xlim and ylim arguments to limit our plotting window to be between these two values. The abline() function can then be used to visualize the horizontal and vertical axes.

In this instance, each axis represents one of the two factors, and each data point represents the correlation between the factor and of the original dimensions. Data points that are closer to one axis can then be thoughts of as being more correlated with that factor. These data points often form several distinct clouds of points, each of which is meant to align with a single factor.

Using this framework, dimensions whose points fall in the spaces between the axes are likely to be complex because they exhibit substantial loadings on to both factors. Recall that the goal of factor rotation is to reduce these complexities. We achieve this goal by amplifying high loadings and reducing low ones. Geometrically, we want each data point to align as closely as possible with one axis while being as far away from the other axis as possible.

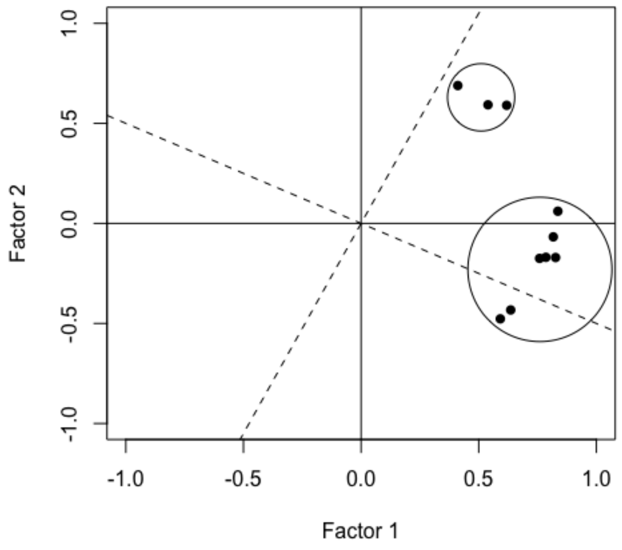
Before moving on, it is important to recognize that these clouds of data points can occur toward either the positive or the negative end of a given axis. In fact, two distinct clouds of points can align with the same axis and, as such, be associated with the same factor, if they occur on opposite ends of said axis.

Using *orthogonal rotation*, we reduce complexity by simply rotating the axes, while keeping them perpendicular to each other, until all data points are as far from the spaces in between axes as possible. We will discuss how to quantify this distance across all data points simultaneously in Section A.2. For now, let's return to our example data.

We can visually estimate that a clockwise orthogonal rotation of approximately 25 degrees should align each axis closer to the centroid of the cloud of data points that it is closest to.

In R, we can use a bit of trigonometry and clever plotting to display these rotated axes.

```
plot(A, pch = 16, xlab = "Factor 1", ylab = "Factor 2",
     xlim = c(-1,1), ylim = c(-1,1))
abline(h = 0, v = 0)
abline(coef = c(0, 2.1), lty = "dashed")
abline(coef = c(0,-0.5), lty = "dashed")
points(0.51, 0.63, cex = 7); points(0.76, -0.23, cex = 15)
```

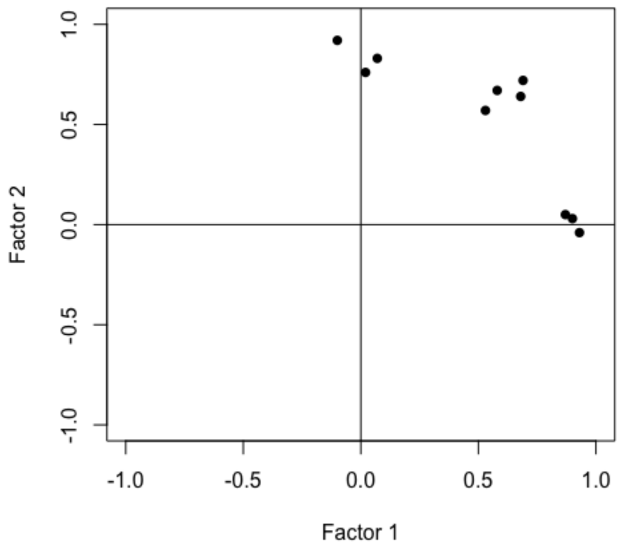


Following this orthogonal rotation, the cloud of three data points at the top now would more closely align with the factor represented by the newly rotated diagonal axis running from bottom left to upper right. The other larger cloud of seven data points would then better align with the other rotated factor axis.

Before moving on to the numerical representation of orthogonal rotation, it is important to note that this or any other rotation may not always be successful in fully eliminating all complexities from a loading matrix.

If, for example, the loading matrix contains three distinct clouds of data points instead of two, there is no way to rotate the two axes so as to align each cluster with just one axis.

```
plot(matrix(c( 0.87,  0.05,
               0.93, -0.04,
               0.90,  0.03,
               0.07,  0.83,
              -0.10,  0.92,
               0.02,  0.76,
               0.53,  0.57,
               0.68,  0.64,
               0.69,  0.72,
               0.58,  0.67), nrow = 10, ncol = 2, byrow = T),
     pch = 16, xlab = "Factor 1", ylab = "Factor 2",
     xlim = c(-1,1), ylim = c(-1,1))
abline(h = 0, v = 0)
```

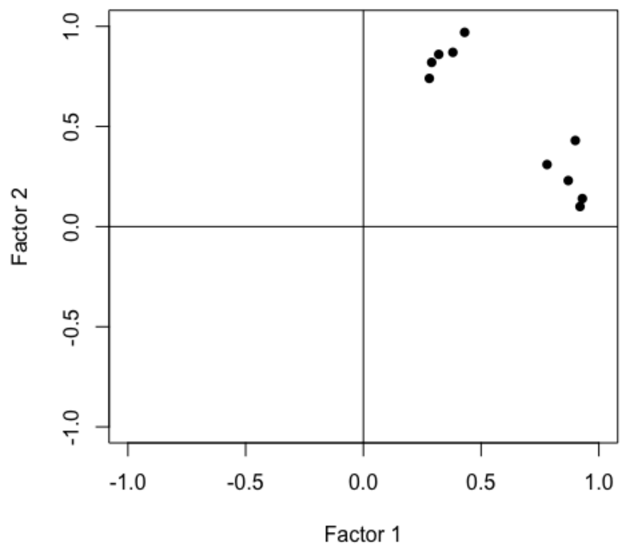


In such situations, the only way to ensure that each cloud of points aligns with a separate axis is to add a third axis. In terms of exploratory factor analysis, this is achieved by increasing the estimated intrinsic dimensionality of the data matrix and adding an additional factor to the analysis.

Remember that all of the different criteria for identifying the intrinsic dimensionality and, by extension, the number of factors to retain in factor analysis are heuristic in nature, which means that one is not inherently better than any other. In this way, Kaiser's, Jolliffe's, and inflection point criteria should serve merely as a starting point from which to further explore whether additional factors are required or even if the number of retained factors should be decreased. Investigating the patterns in a factor loading plot is just one way to conduct this further exploration.

Another situation where orthogonal rotation may be insufficient is if both clouds of points clearly fall between two axes in such a way that no set of perpendicular axes could adequately capture both clusters.

```
plot(matrix(c(0.87, 0.23,
              0.93, 0.14,
              0.90, 0.43,
              0.92, 0.10,
              0.78, 0.31,
              0.32, 0.86,
              0.43, 0.97,
              0.28, 0.74,
              0.29, 0.82,
              0.38, 0.87), nrow = 10, ncol = 2, byrow = T),
     pch = 16, xlab = "Factor 1", ylab = "Factor 2",
     xlim = c(-1,1), ylim = c(-1,1))
abline(h = 0, v = 0)
```



In such situations, the only way to better align one axis with one and only one cloud of points is to relax the limitation of needing to maintain a perpendicular state between the two axes. We will discuss these non-orthogonal rotation methods in more detail in Section 2.5

Another thing to note is that although this geometric representation can certainly be expanded into three-dimensional space for a three-factor representation and even higher dimensional space for higher factor representations, visualizing such loading matrices using a factor loading plot quickly becomes untenable. As such, these factor loading plots are rarely used in research; instead, a quantitative and matrix-based equivalency is used to achieve a similar goal.

2.3.2 | Orthogonally Rotating a Loading Matrix

Geometrically, orthogonal rotation is achieved by rotating the factor axes about the origin while maintaining them in a perpendicular state. Numerically, we achieve the same result by using a *projection* or *rotation matrix* $\boldsymbol{\Psi}$:

$$\boldsymbol{\Psi} = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix} \tag{2.25}$$

This rotation matrix contains only sines and cosines of a *rotation angle* ψ, which are used to project the loadings on to a new set of factor axes. The new factor axes are rotated from the original by a counterclockwise rotation of ψ.

For our example data, we visually estimated that a rotation of approximately 25 degrees clockwise would be sufficient to decrease the observed complexities among dimensions. However, because factor rotations are measured counterclockwise, the angle of rotation that we would want to use would actually be equal to negative 25 degrees.

In R, we can use this rotation angle to quickly construct the corresponding rotation matrix.

```
psi <- -25 * pi / 180
PSI <- matrix(c(cos(psi), -sin(psi),
                sin(psi),  cos(psi)), nrow = 2, ncol = 2, byrow = T)
PSI
           [,1]      [,2]
[1,]  0.9063078 0.4226183
[2,] -0.4226183 0.9063078
```

By multiplying the unrotated loading matrix by this transformation matrix (which we note uses radians rather than degrees in R), we can arrive at the *rotated loading matrix* $\boldsymbol{A}$:

$$\boldsymbol{A} = \boldsymbol{A}_{unrotated} \times \boldsymbol{\Psi} \tag{2.26}$$

We can quickly implement this equation in R using simple matrix multiplication.

```
A1 <- A %*% PSI
A1
                        [,1]       [,2]
Hopeless          0.76877492  0.2841929
Overwhelmed       0.73775018 -0.1821990
Exhausted         0.75913032 -0.1233503
Lonely            0.78346033  0.1784592
Sad               0.82116508  0.1948444
Depressed         0.73220051  0.4085559
Anxious           0.76139117  0.1621576
SelfHarming       0.23829931  0.7651044
SuicidalThoughts  0.31130112  0.7962845
SuicidalAttempts  0.08133888  0.7980972
```

Comparing the correlations in this rotated loading matrix to those in our original unrotated loading matrix, we can see that four of the five originally complex dimensions are now pure (the only exception being SuicidalThoughts). Unfortunately, one of the dimensions that was originally pure (feeling Depressed) is now complex. Although this is not ideal, the total number of complex dimensions has decreased from five to two and the remaining complexities are now not as substantial as those seen earlier.

The question now becomes whether or not this visually estimated rotation angle is truly the best option for an orthogonal rotation.

2.3.3 | Finding the Optimal Rotation

Recall that the goal of orthogonal rotation is to reduce complexities among dimensions by amplifying high loadings and minimizing low ones. For our example data, an orthogonal rotation of negative 25 degrees was sufficient to purify most but not all of our dimensions.

But could a different rotation angle have decreased the number of remaining complex dimensions even lower? If not, could it have otherwise reduced these complexities to be less substantial even if they still exceed the 0.30 threshold of being technically complex?

Before moving on, it is important to note that rotation is used only to improve the interpretability of factors, not to improve the fit between the original correlation matrix and the one that can be reproduced directly from the corresponding loading matrix. In fact, all orthogonally rotated solutions are mathematically equivalent in terms of variance explained and other indices of model fit, which we will discuss in Section 2.4.

Numerous methods or algorithms have been proposed to identify the optimal angle of rotation and thereby produce the least complex loading matrix possible. However, if the correlation structure (a standardized version of the covariance structure) is fairly clear, most of these methods will yield very similar if not identical results. As such, only a few of these methods are generally used (see Section A.2 for more details on some of the other algorithms).

The most common orthogonal rotation algorithm by far is known as *varimax*. This method seeks to maximize the sum of the variances of the squared loadings in each column by making high loadings higher and low loadings lower for each factor.

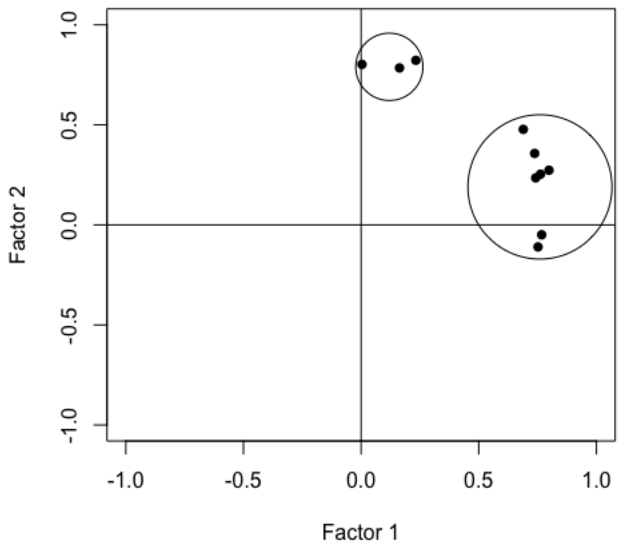
In R, we can implement this algorithm by setting the `rotate` argument equal to `"varimax"` in the `pca()` function.

```
A2 <- pca(r = R, nfactors = 2, rotate = "varimax")$loadings[]
A2
                         RC1         RC2
Hopeless         0.737684419  0.35721446
Overwhelmed      0.751913330 -0.10999159
Exhausted        0.767501553 -0.04935094
Lonely           0.762527268  0.25339680
Sad              0.798470525  0.27335188
Depressed        0.689253418  0.47745704
Anxious          0.742138215  0.23503718
SelfHarming      0.163183240  0.78456520
SuicidalThoughts 0.232827147  0.82265971
SuicidalAttempts 0.003767682  0.80222254
```

Much like the earlier rotated loading matrix, this new rotated loading matrix also contains two complex dimensions and eight pure dimensions, although which dimensions are complex has now changed. The differences between high and low correlations, however, are more pronounced than they were earlier.

The benefit of using this algorithm to identify an optimal angle of rotation is best seen when we visualize the new geometric alignment between the factors and original dimensions using a factor loading plot.

```
plot(A2, pch = 16, xlab = "Factor 1", ylab = "Factor 2",
     xlim = c(-1,1), ylim = c(-1,1))
abline(h = 0, v = 0)
points(0.12, 0.79, cex = 7); points(0.76, 0.19, cex = 15)
```



Note that whereas our visually estimation rotation yielded a set of factor axes where only one axis went directly through the cloud of data points, the varimax rotation ensured that both axes now go through their respective clouds of points. Such optimized alignment makes interpreting and naming factors much easier.

For our example data, the three dimensions corresponding to engaging in SelfHarming behavior, SuicidalThoughts, and SuicidalAttempts all clearly align with and load substantially on to a single factor, which we can think of as measuring aspects of or being a byproduct of suicidal ideation. The other seven dimensions corresponding to feeling Hopeless, Overwhelmed, Exhausted, Lonely, Sad, Depressed, and Anxious all clearly align with the other factor, which we had earlier named as psychological distress.

Once dimensions have been optimally rotated and the corresponding factors properly named, we can again return to calculating the numerical value associated with these factors for each case in the original data matrix. For example, which students exhibit above-average suicidal ideation or psychological distress?

Thankfully, the process of calculating factor scores via a matrix of regression coefficients and standardized data matrices is exactly the same regardless of whether the loading matrix is rotated or not, allowing us to simply use equations 2.23 and 2.24 from Section 2.2 to this end.

Once these factor scores have been extracted and properly interpreted in terms of distance from the mean, the next step is to validate how successfully they capture the correlation structure of the original data matrix, something that we will explore in Section 2.4.

Exercises for Section 2.3

2.09 Online Banking Performance (continued). Exercise 2.07 introduced data on the cost efficiency focused operational orientations of 32 banks in Taiwan that were published in 2009 in *Computer and Operations Research*, which are recreated below:

	PC1	PC2
A	0.744	-0.563
B	0.588	-0.221
C	0.756	-0.129
D	0.708	-0.374
AB	0.739	-0.574
BC	0.755	-0.193
AC	0.802	-0.409
AD	0.825	-0.548
BD	0.842	-0.422
CD	0.856	-0.400
ABC	0.783	-0.456
BCD	0.855	-0.405
ABD	0.830	-0.525
ACD	0.850	-0.460
ABCD	0.839	-0.487

The loading matrix above includes the correlations between two extracted factors or principal components (PC) and dimensions related to the estimated efficiency of deposits (A), operational costs (B), number of employees (C), and equipment (D) at each bank in 2005.

a. Construct this loading matrix in R with the proper row and column names
b. Create a factor loading plot of this loading matrix
c. Visually estimate the counterclockwise rotation angle that would better align each axis with the cloud(s) of points
d. Explain if it appears that two factors were indeed necessary to summarize the correlation structure between the original 13 dimensions

2.10 Multidimensional Poverty Measure (continued). Exercise 1.02 introduced a sample of data collected by the World Bank Group on different South American countries' access to education, basic infrastructure, and other dimensions of poverty, which is recreated below:

	Water	Electricity	Sanitation	Education
Argentina	0.3	0.0	0.4	1.5
Bolivia	7.4	4.9	16.3	13.2
Brazil	1.7	0.2	34.2	16.0
Chile	0.1	0.3	0.6	4.0
Colombia	2.4	1.3	8.2	5.1
Ecuador	4.3	1.4	3.6	3.9
Paraguay	2.1	0.3	9.0	6.3
Peru	6.2	4.1	12.1	5.4
Uruguay	0.5	0.1	1.0	2.0

The values above denote what percent of each country's population did not have access to the designated dimension of poverty in 2019.

a. Construct this data matrix in R with the proper row and column names
b. Find the unrotated loading matrix using Jolliffe's criterion
c. Create a factor loading plot of this loading matrix
d. Explain why orthogonal rotation appears to be needed
e. Find the rotation matrix for a counterclockwise rotation angle of 45 degrees
f. Use matrix algebra to find the rotated loading matrix and corresponding factor loading plot

2.11 Facework in Intercultural Communication (continued). Exercise 2.04 introduced data on the mediating effects of situations factors on the use of avoiding and corrective facework during intercultural miscommunication in face-to-face contexts, which are recreated below:

	1	2	3	4	5	6	7	8	9	10
1. Self-positive face	1.0	.35	.36	.19	.12	.18	.21	.23	-.15	-.13
2. Self-negative face	.35	1.0	.28	.41	.17	.03	.06	.24	.01	-.12
3. Other-positive face	.36	.28	1.0	.40	.23	.27	.25	.32	-.21	-.07
4. Other-negative face	.19	.41	.40	1.0	.24	.24	.27	.21	-.09	.05
5. Severity	.12	.17	.23	.24	1.0	.75	.74	-.01	.34	.29
6. Threatened positive face	.18	.03	.27	.24	.75	1.0	.78	.12	.12	.26
7. Threatened negative face	.21	.06	.25	.27	.74	.78	1.0	.17	.06	.24
8. Unintentional attribution	.23	.24	.32	.21	-.01	.12	.17	1.0	-.35	-.27
9. Intentional attribution	-.15	.01	-.21	-.09	.34	.12	.06	-.35	1.0	.27
10. Incidental attribution	-.13	-.12	-.07	.05	.29	.26	.24	-.27	.27	1.0

The correlation matrix above includes the covariability between ten different situational factors that were measured on 103 undergraduate students who were U.S. citizens.

a. Construct this correlation matrix in R with the proper row and column names
b. Create a factor loading plot of the unrotated loading matrix for an intrinsic dimensionality of two, which corresponds to the original theoretical framework of this article, which hypothesized that dimensions 1 through 4 would form a single coherent cluster (referred to as individual face needs) and dimensions 5 through 10 would form another cluster (referred to as situational factors)
c. Explain if there appears to be any complexity among the original dimensions
d. Find the optimally rotated loading matrix using the "varimax" algorithm and the corresponding factor loading plot
e. Explain if there appears to be any remaining complexity among the original dimensions following this rotation
f. Explain if it appears that non-orthogonal rotation may be effective in further decreasing any remaining complexities among the original dimensions

2.12 Residential Property Values in Ames, Iowa (continued). Exercise 1.16 introduced a sample of data collected by the Ames Assessor's Office on residential properties, which are recreated below:

	Area	Price	Subclass	Lot Frontage	Lot Area	Quality	Built	Remodel
526301100	1656	215000	20	141	31770	6	1960	1960
526350040	896	105000	20	80	11622	5	1961	1961
526351010	1329	172000	20	81	14267	6	1958	1958
526353030	2110	244000	20	93	11160	7	1968	1968
527105010	1629	189900	60	74	13830	5	1997	1998
527105030	1604	195500	60	78	9978	6	1998	1998
527127150	1338	213500	120	41	4920	8	2001	2001
527145080	1280	191500	120	43	5005	8	1992	1992
527146030	1616	236500	120	39	5389	8	1995	1996
527162130	1804	189000	60	60	7500	7	1999	1999
527163010	1655	175900	60	75	10000	6	1993	1994
527165230	1187	185000	20	NA	7980	6	1992	2007
527166040	1465	180400	60	63	8402	6	1998	1998
527180040	1341	171500	20	85	10176	7	1990	1990
527182190	1502	212000	120	NA	6820	8	1985	1985
527216070	3279	538000	60	47	53504	8	2003	2003
527225035	1752	164000	50	152	12134	8	1988	2005
527258010	1856	394432	20	88	11394	9	2010	2010
527276150	864	141000	20	140	19138	4	1951	1951
527302110	2073	210000	20	85	13175	6	1978	1988

The data above include information on the first 20 rows of residential properties sold in Ames, IA from 2006 to 2010, including the price they sold for.

a. Construct this data matrix in R with the proper row and column names
b. Find the total number of complex dimensions in an unrotated four-factor loading matrix solution derived from a pairwise complete correlation matrix
c. Find the total number of complex dimensions in an optimally rotated (using the `"varimax"` algorithm) four-factor loading matrix solution derived from a pairwise complete correlation matrix
d. Explain if orthogonal rotation was successful in decreasing complexity among the original dimensions