Principal Components Analysis: Creating Linear Composites of Predictors

A brief introduction to methods of combining correlated predictors. Example taken from Chatterjee & Hadi (2012).

AUTHOR

Andrew Zieffler

PUBLISHED

Oct. 14, 2020

In 1964, the US Congress passed the Civil Rights Act and also ordered a survey of school districts to evaluate the availability of equal educational opportunity in public education. The results of this survey were reported on in Coleman et al. (1966) and Mosteller & Moynihan (1972). The data in *equal-educational-opportunity.csv* consist of data taken from a random sample of 70 schools in 1965. The variables, which have all been mean-centered and standardized, include:

- `achievement`: Measurement indicating the student achievement level
- `faculty`: Measurement indicating the faculty's credentials
- `peer`: Measurement indicating the influence of peer groups in the school
- `school`: Measurement indicating the school facilities (e.g., building, teaching materials)

We will use these data to mimic one of the original regression analyses performed; examining whether the level of school facilities was an important predictor of student achievement after accounting for the variation in faculty credentials and peer influence.

```
# Load libraries
library(broom)
library(car)
library(corrr)
library(tidyverse)

# Read in data
eeo = read_csv("~/Documents/github/epsy-8264/data/equal-education-opportunity.csv")
head(eeo)
```

```
# A tibble: 6 x 4
  achievement faculty    peer school
        <dbl>   <dbl>   <dbl>  <dbl>
1      -0.431   0.608  0.0351  0.166
2       0.800   0.794  0.479   0.534
3      -0.925  -0.826 -0.620  -0.786
4      -2.19   -1.25  -1.22   -1.04
5      -2.85    0.174 -0.185   0.142
6      -0.662   0.202  0.128   0.273
```

The problem we faced from the last set of notes, was that the predictors in the model were collinear, so we encountered computational issues when trying to estimate the effects and standard errors.

Idea of Principal Components

If the *X*-matrix of the predictors were orthogonal, there would be no collinearity issues and we could easily estimate the effects and standard errors. This is, of course, not the case since the predictors are highly correlated. The idea of principal components analysis is to change the basis vectors (coordinate system) so that they are orthogonal. This is shown in the figure below in which we consider the predictor space composed of two of the predictors.

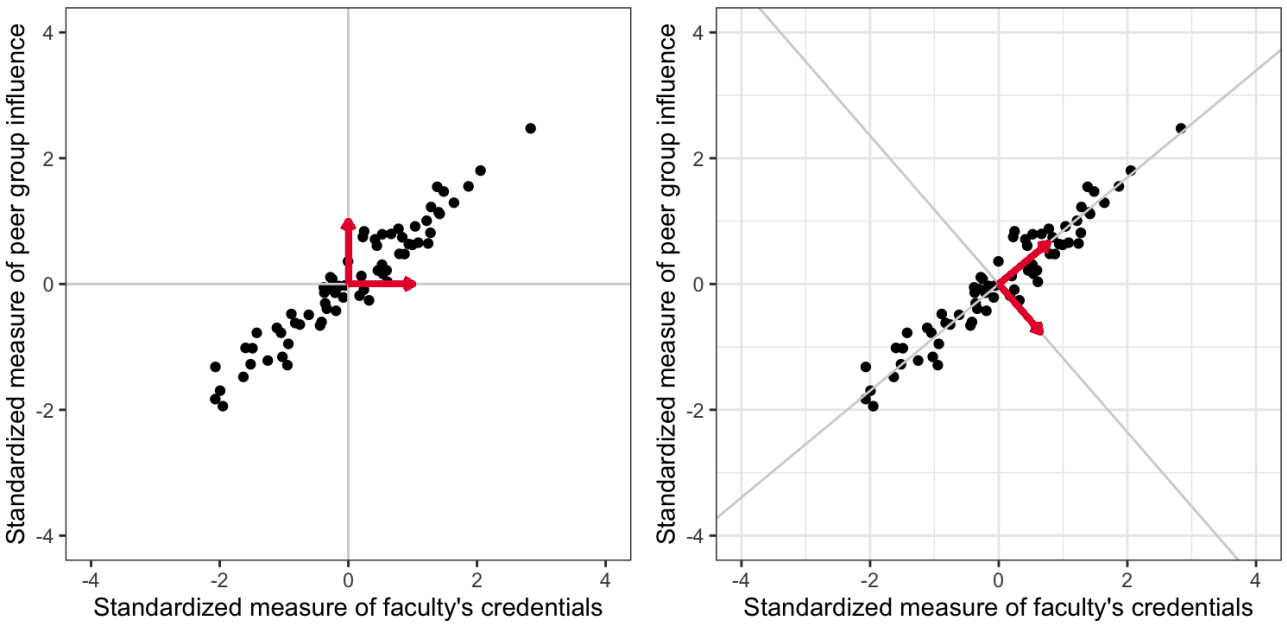


Figure 1: LEFT: Faculty credential and peer influence measures shown in the coordinate system given by the $(1, 0)$-$(0, 1)$ basis. RIGHT: The coordinate system has been rotated based on the $(0.763, 0.647)$-$(0.647, -0.762)$ basis.

If we had considered all three predictors, the plot would be in three-dimensional space and we would need to rotate the coordinate system formed by the basis vectors (1, 0, 0)$-(0, 1, 0)-(0, 0, 1). With three dimensions, of course, we can now rotate in multiple directions. This idea can also be extended

three dimensions, of course, we can now rotate in multiple directions. This idea can also be extended to *k*-dimensional space. (For now, we will continue to work with the predictor space defined by the `faculty` and `peer` predictors.)

Recall from our notes on vector geometry, that transforming the coordinates for a point to a new basis is a simple matter of pre-multiplying the vector of original coordinates by the matrix composed of the basis vectors. For example, the first observation had a `faculty` value of 0.608, and a `peer` value of 0.0351.

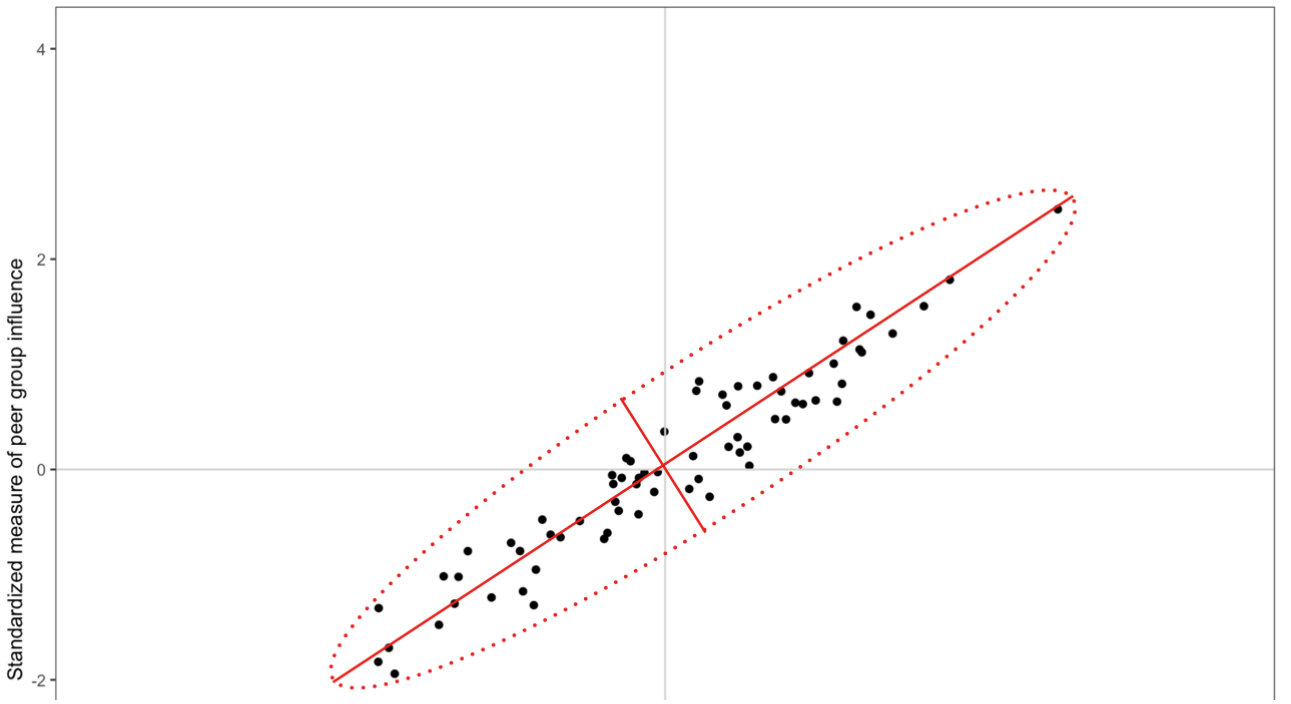
```
# Coordinates in original predictor space (row vector)
old = t(c(0.608, 0.0351))

# Matrix of new basis vectors
basis = matrix(c(0.7625172, 0.6469680, 0.6469680, -0.7625172), nrow = 2)

# Coordinates in rotated predictor space
old %*% basis
```

```
         [,1]      [,2]
[1,] 0.486319 0.3665922
```

Aside from producing an orthogonal basis, we also choose the principal components so that they maximize variance in the predictor space. For example, the direction of the first basis vector (i.e., the first principal component) is chosen to maximize the variation in the predictor space. This is essentially the direction of the major axis in the data ellipse. The second principal component is then chosen to maximize variance in an orthogonal direction to the first principal component. (With only two predictors, there is only one possible direction for the second principal component.) In our data ellipse this would be the direction of the minor axis. This continues until we exhaust the number of principal components.



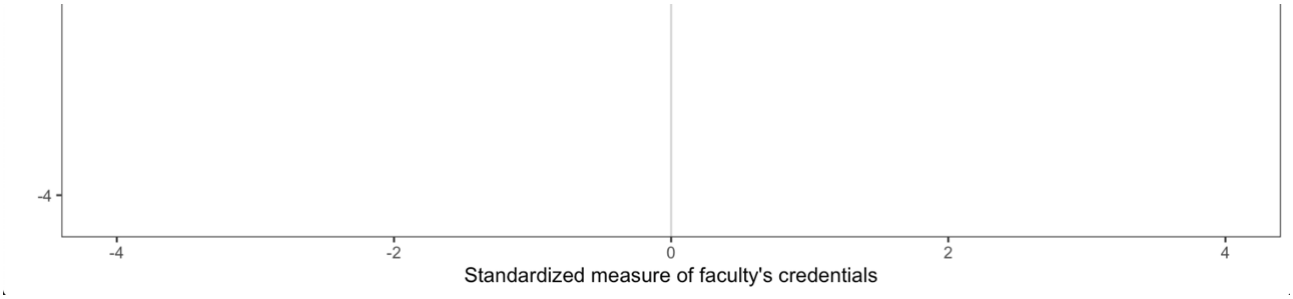


Figure 2: Data ellipse showing the directions of the principal components. The first principal component is in the direction of the major axis and the second principal component is in the direction of the minor axis.

Determining the Principal Components

Looking at the principal components in Figure 1 (RIGHT), it is clear that they lie in the subspace spanned by the original basis vectors. Because of this, we can express each principal component as a linear combination of the original predictor values. (To facilitate interpretations, all the predictors are typically standardized.) As such, we can write the first principal component as:

$$\begin{bmatrix} w_{1_1} \\ w_{1_2} \\ w_{1_3} \\ \vdots \\ w_{1_n} \end{bmatrix} = c_{1_1} \begin{bmatrix} x_{1_1} \\ x_{1_2} \\ x_{1_3} \\ \vdots \\ x_{1_n} \end{bmatrix} + c_{2_1} \begin{bmatrix} x_{2_1} \\ x_{2_2} \\ x_{2_3} \\ \vdots \\ x_{2_n} \end{bmatrix} + \ldots + c_{k_1} \begin{bmatrix} x_{k_1} \\ x_{k_2} \\ x_{k_3} \\ \vdots \\ x_{k_n} \end{bmatrix}$$

$$\underset{n\times 1}{\mathbf{W}_1} = \underset{n\times k}{\mathbf{X}} \underset{k\times 1}{\mathbf{C}_1}$$

where $\mathbf{W}_1$ is the basis vector defined by the first principal component, $\mathbf{X}$ is the matrix of predictor values, and $\mathbf{C}_1$ is a vector of coefficient weights that produce the desired value in $\mathbf{W}_1$. We can also compute the variance of $\mathbf{W}_1$ as:

$$\begin{aligned} S^2_{W_1} &= \frac{1}{n-1}\mathbf{W}_1^\intercal \mathbf{W}_1 \\ &= \frac{1}{n-1}\left(\mathbf{X}\mathbf{C}_1\right)^\intercal \left(\mathbf{X}\mathbf{C}_1\right) \\ &= \frac{1}{n-1}\mathbf{C}_1^\intercal \mathbf{X}^\intercal \mathbf{X}\mathbf{C}_1 \\ &= \mathbf{C}_1^\intercal \mathbf{\Sigma}_{XX}\mathbf{C}_1 \end{aligned}$$

where $\mathbf{R}_{XX}$ is the variance–covariance matrix of the predictors (since $\mathbf{\Sigma}_{XX} = \frac{1}{n-1}\mathbf{X}^\intercal\mathbf{X}$).

Our goal is to maximize the variance, but in order to do that we need to constrain $\mathbf{C}_1^\intercal \mathbf{C}_1 = 1$. To maximize the variance under a constraint, we include a Lagrange multiplier in the formula for variance and then differentiate that with respect to $\mathbf{C}_1$ and the Lagrange multiplier; set those partial derivatives equal to 0; and solve. This produces two equations:

Without the constraint, the variance could be made arbitrarily large by choosing large coefficients for $\mathbf{C}_1$.

$$\begin{aligned} \left(\mathbf{\Sigma}_{XX} - \lambda_1 \mathbf{I}\right)\mathbf{C}_1 &= \mathbf{0} \\ \mathbf{C}_1^\intercal \mathbf{C}_1 &= 1 \end{aligned}$$

This gives solutions to $\mathbf{C}_1$ only when $\mathbf{\Sigma}_{XX} - \lambda_1 \mathbf{I}$ is singular. This implies that:

- λ_1 is an eigenvalue of $\mathbf{R}_{XX}$; and
- $\mathbf{C}_1$ is the corresponding eigenvector (scaled so that $\mathbf{C}_1^\intercal \mathbf{C}_1 = 1$)

One problem is that there will be many eigenvalues/vectors to choose from since there will be *k* such pairs. From the equation based on the first partial derivative,

$$\begin{aligned} \left(\mathbf{\Sigma}_{XX} - \lambda_1 \mathbf{I}\right)\mathbf{C}_1 &= \mathbf{0} \\ \mathbf{\Sigma}_{XX}\mathbf{C}_1 &= \lambda_1 \mathbf{C}_1 \end{aligned}$$

Which we can substitute in to the variance formula (recall this is what we are maximizing).

$$\begin{aligned} S_{W_1}^2 &= \mathbf{C}_1^\intercal \mathbf{\Sigma}_{XX} \mathbf{C}_1 \\ &= \mathbf{C}_1^\intercal \lambda_1 \mathbf{C}_1 \\ &= \lambda_1 \mathbf{C}_1^\intercal \mathbf{C}_1 \\ &= \lambda_1 \end{aligned}$$

Maximizing the variance implies that we will want to choose the largest eigenvalue and corresponding eigenvector.

The second principal component is derived in a similar fashion, with the further constraint that it is orthogonal to the first principal component. (This process continues for the third, fourth, fifth, etc. principal components.) The math (not shown) indicates that maximizing the variance in the second principal component corresponds to choosing the second largest eigenvalue (and corresponding eigenvector) and so on. That is, finding the rotation matrix is equivalent to ordering the eigenvalues based on decomposing $\mathbf{\Sigma}_{XX}$ from largest to smallest, and then producing a matrix made up of the corresponding eigenvectors.

Let's go back to our example.

```
# Compute variance-covariance matrix of predictors
sigma_xx = cov(ggg[, c("faculty", "peer")])
```

```
sigma_xx = cov(eeo[ , c( "faculty" , "peer" )])

# Eigendecomposition
eigen(sigma_xx)
```

```
eigen() decomposition
$values
[1] 1.98899907 0.03947275

$vectors
           [,1]       [,2]
[1,] -0.7625938  0.6468777
[2,] -0.6468777 -0.7625938
```

```
# Matrix of new basis vectors
basis = eigen(sigma_xx)$vectors

# Coordinates in original predictor space (row vector)
# These are often centered
old = t(c(0.608, 0.0351))

# Compute rotated values under the new basis
old %*% basis
```

```
           [,1]      [,2]
[1,] -0.4863624 0.3665346
```

Using `princomp()` to Obtain the Principal Components

We can also use the R function `princomp()` to obtain the principal components based on the eigendecomposition. We provide this function with a data frame of the predictors.

```
# Select predictors
eeo_pred = eeo %>%
  select(faculty, peer)

# Create princomp object
my_pca = princomp(eeo_pred)

# View output
summary(my_pca, loadings = TRUE)
```

```
Importance of components:
                          Comp.1     Comp.2
Standard deviation     1.4002088 0.19725327
Proportion of Variance 0.9805406 0.01945935
```

```
Cumulative Proportion  0.9805406 1.00000000

Loadings:
        Comp.1 Comp.2
faculty  0.763  0.647
peer     0.647 -0.763
```

The values of the principal components are given in the `loadings` output. Note that the signs of the loadings are arbitrary. For example, the vector associated with PC1 could also have been $(-0.763, -0.647)$ and that for PC2 could have been $(-0.647, 0.763)$. The variances of each component can be computed by squaring the appropriate standard deviations in the output.

```
# Compute variance of PC1
1.4002088 ^ 2
```

```
[1] 1.960585
```

```
# Compute variance of PC2
0.19725327 ^ 2
```

```
[1] 0.03890885
```

Remember, the variances are the eigenvalues from the eigendecomposition. Also, since the principal components are orthogonal, we can sum the variances to obtain a total measure of variation in the original set of predictors accounted for by the principal components.

```
# Compute total variation accounted for
total_var = 1.4002088 ^ 2 + 0.19725327 ^ 2
total_var
```

```
[1] 1.999494
```

```
# Compute variation accounted for by PC1
(1.4002088 ^ 2) / total_var
```

```
[1] 0.9805406
```

```
# Compute variation accounted for by PC2
(0.19725327 ^ 2) / total_var
```

```
[1] 0.01945935
```

This suggests that PC1 accounts for 98% of the variance in the original set of predictors and that PC2 accounts for 2% of the variance. Note that these values are also given in the `summary()` output. We

can also obtain the PC scores for each observation by accessing the `scores` element of the `princomp` object. (Below we only show the first 10 scores.)

```
# Get PC scores
my_pca$scores
```

```
          Comp.1      Comp.2
 [1,]  0.41884376  0.37000669
 [2,]  0.84765375  0.15132881
 [3,] -1.09849740 -0.05870659
 [4,] -1.81031365  0.12065755
 [5,] -0.05471761  0.25713367
 [6,]  0.16934323  0.03700331
 [7,]  0.05844541  0.22861604
 [8,]  0.52621778  0.22189095
 [9,] -0.85386180 -0.02140061
[10,]  1.09286966  0.17176523
```

These are slightly different than the scores we obtained by multiplying the original predictor values by the new basis matrix. For example, the PC scores for the first observation were -0.486 and 0.367. The `princomp()` function mean centers each variable prior to multiplying by the basis matrix.

```
# Mimic scores from princomp()
old = t(c(0.608 - mean(eeo$faculty), 0.0351 - mean(eeo$peer)))

# Compute PC scores
old %*% basis
```

```
          [,1]      [,2]
[1,] -0.4187435 0.3699085
```

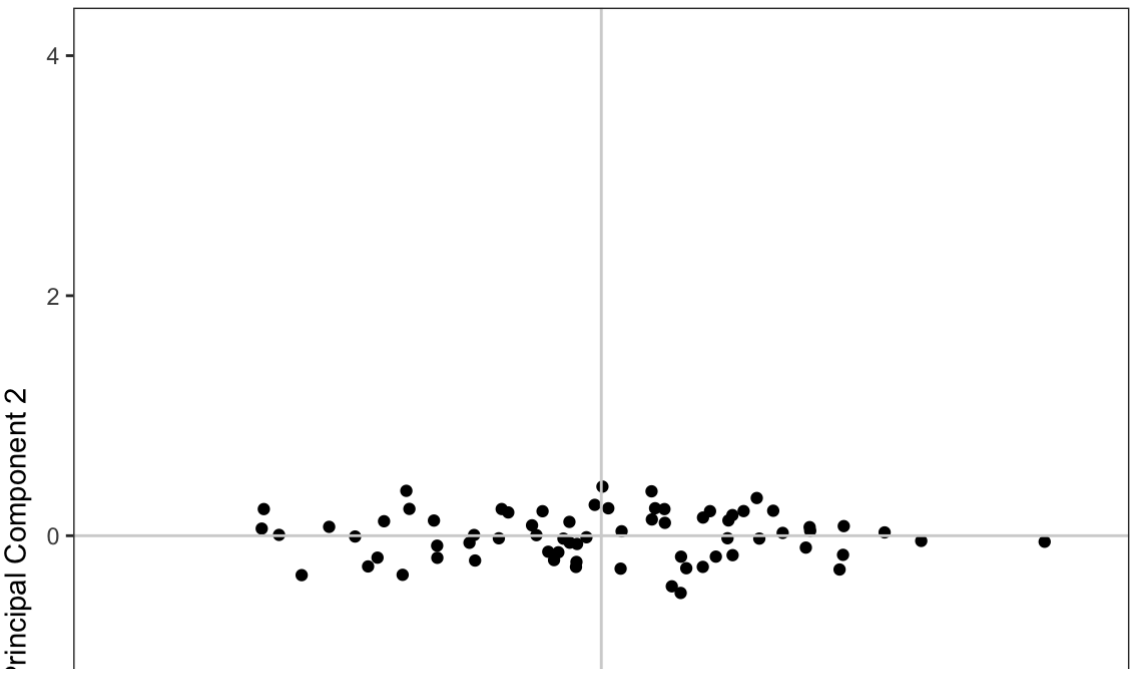
Because we want the prinicpal components to be soley functions of the predictors, we mean center them. Otherwise we would have to include a column of ones (intercept) in the **X** matrix. Mean centering the variables makes each variable orthogonal to the intercept, in which case it can be ignored. (This is simlar to how mean centering predictors removes the intercept from the fitted equation.)

Since we only have two principal components, we can visualize the scores using a scatterplot.

```
# Create data frame of scores
pc = data.frame(my_pca$scores)

# Plot the scores
ggplot(data = pc, aes(x = Comp.1, y = Comp.2)) +
  geom_point() +
  geom_hline(yintercept = 0, color = "lightgrey") +
  geom_vline(xintercept = 0, color = "lightgrey") +
  scale_x_continuous(name = "Principal Component 1", limits = c(-4, 4)) +
  scale_y_continuous(name = "Principal Component 2", limits = c(-4, 4)) +
  theme_bw() +
  theme(
```

```
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank()
  )
```



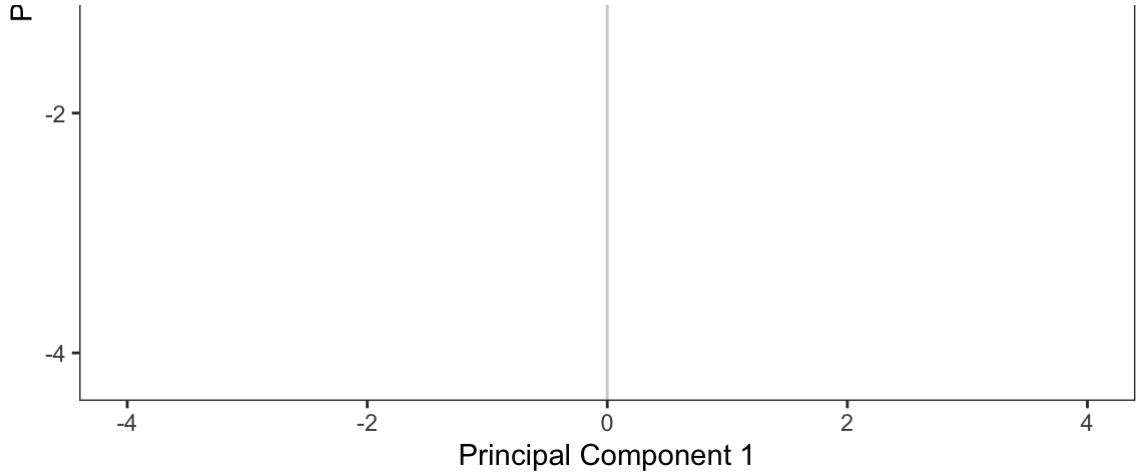


Figure 3: Rotated predictor space using the principal components as the new basis.

Conceptually this visualization depicts rotating the scatterplot showing the initial values to a new set of basis axes. The figure below, shows the rotated predictor space after re-orienting the rotated coordinate system. From this visualization, it is clear that there is much more variation in the values of PC1 than in PC2.

Using All Three Predictors in the PCA

In the example we were focused on only two predictors, as it is pedagogically easier to conceptualize since the plotting is easier. However, PCA is directly extensible to more than two variables. With three variables, the data ellipse is a data ellipsoid and there are three principal components corresponding to the three orthogonal semi-axes of the ellipsoid. With four or more variables the ideas extend although the visual doesn't.

We will again use the `princomp()` function to compute the principal components and rotated scores.

The `cutoff=0` argument in the `summary()` function prints all the loadings. By default, only loadings above 0.1 are displayed.

```
# Select predictors
eeo_pred = eeo %>%
  select(faculty, peer, school)

# Create princomp object
my_pca = princomp(eeo_pred)

# View output
summary(my_pca, loadings = TRUE, cutoff = 0)
```

```
Importance of components:
                          Comp.1     Comp.2     Comp.3
Standard deviation     1.7277218 0.19747823 0.09021075
Proportion of Variance 0.9844548 0.01286135 0.00268389
```

```
Cumulative Proportion  0.9844548 0.99731611 1.00000000

Loadings:
        Comp.1 Comp.2 Comp.3
faculty  0.617  0.670  0.412
peer     0.524 -0.741  0.420
school   0.587 -0.043 -0.809
```

The rotation matrix, or loading matrix, is:

$$\mathbf{W} = \begin{bmatrix} 0.617 & 0.670 & 0.412 \\ 0.524 & -0.741 & 0.420 \\ 0.587 & -0.043 & -0.809 \end{bmatrix}$$

The first principal component again accounts for most of the variance in the three predictors (98.4%). The other two principal components account for an additional 1.3% and 0.3% of the variance in the predictor space.

Remember that each principal component is defining a composite variable composed of all three predictors. The loadings, which are the weights in computing the composite variables, can also be interpreted as the correlations between each particular variable and the composite. And, although the signs are arbitrary, we can try to interpret the differences in direction. So, for example,

- The composite variable formed by the first principal component is moderately and positively correlated with all three predictors.
- The second composite variable is highly positively correlated with the faculty variable, highly negatively correlated with the peer variable, and not correlated with the school variable.
- The third composite variable is positively and moderately correlated with the faculty and peer variables, and highly negatively correlated with the school variable.

Sometimes these patterns of correlations can point toward an underlying latent factor, but this is a subjective call by the researcher based on their substantive knowledge. Other times the patterns make no sense; the results, after all, are just a mathematical result based on the variances and covariances of the variables.

Here, the first composite might be interpreted as an overall measurement of the three predictors since all the loadings are in the same direction and at least of moderate size. The second composite seems to represent a contrast between faculty credentials and peer influence due to the opposite signs on these loadings. While the third composite points toward a more complex contrast between school facilities and the combined faculty credentials/peer group influence.

Using the Principal Components in a Regression Model

Remember, we undertook the PCA because of the collinearity in the original predictors. Rather than

Remember, we undertook the PCA because of the collinearity in the original predictors. Rather than using the original predictor values in our regression, we can use the scores from the PCA. Since we created the principal components to be orthognal, this should alleviate any collinearity problems.

```
# Create data frame of PC scores
pc = data.frame(my_pca$scores)

# Add the PC scores to the original data
eeo2 = eeo %>%
  cbind(pc)

# View data
head(eeo2)
```

```
  achievement  faculty     peer   school      Comp.1      Comp.2
1    -0.43148  0.60814  0.03509  0.16607  0.41775535  0.37701335
2     0.79969  0.79369  0.47924  0.53356  0.98069934  0.15640371
3    -0.92467 -0.82630 -0.61951 -0.78635 -1.36961355 -0.05819740
4    -2.19081 -1.25310 -1.21675 -1.04076 -2.09539768  0.10923278
5    -2.84818  0.17399 -0.18517  0.14229  0.02033702  0.25023453
6    -0.66233  0.20246  0.12764  0.27311  0.27862628  0.03189946
       Comp.3
1  0.11694576
2  0.08270588
3  0.02140965
4 -0.19945874
5 -0.13514941
6 -0.09785376
```

```
# Fit model using PC scores
lm.pc = lm(achievement ~ 1 + Comp.1 + Comp.2 + Comp.3, data = eeo2)

# Check for collinearity -- correlations
eeo2 %>%
  select(Comp.1, Comp.2, Comp.3) %>%
  correlate()
```

```
# A tibble: 3 x 4
  rowname    Comp.1    Comp.2    Comp.3
  <chr>       <dbl>     <dbl>     <dbl>
1 Comp.1  NA         6.35e-16  1.00e-15
2 Comp.2   6.35e-16 NA        -1.12e-15
3 Comp.3   1.00e-15 -1.12e-15 NA
```

```
# Check for collinearity -- VIF
vif(lm.pc)
```

```
Comp.1 Comp.2 Comp.3
     1      1      1
```

Examining some of the collinearity diagnostics we see that the predictors in this model are completely uncorrelated and the VIF values are 1; indicating that the SEs for these coefficients are exactly as large as they would be if the predictors were independent (which they are). Looking at the model- and coefficient-level output:

```
# Model-level output
glance(lm.pc)
```

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic p.value    df logLik   AIC
      <dbl>         <dbl> <dbl>     <dbl>   <dbl> <dbl>  <dbl> <dbl>
1     0.206         0.170  2.07      5.72 0.00153     3  -148.  306.
# … with 4 more variables: BIC <dbl>, deviance <dbl>,
#   df.residual <int>, nobs <int>
```

```
# Coefficient-level output
tidy(lm.pc, conf.int = 0.95)
```

```
# A tibble: 4 x 7
  term        estimate std.error statistic  p.value conf.low conf.high
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
1 (Intercept)   0.0192     0.247    0.0776 0.938      -0.475     0.513
2 Comp.1        0.559      0.143    3.90   0.000226    0.273     0.845
3 Comp.2       -0.883      1.25    -0.705  0.483      -3.39      1.62
4 Comp.3        3.27       2.74     1.19   0.237      -2.20      8.75
```

The model-level output from this model is exactly the same as the model-level output from the model fitted with the original predictors. The coefficient-level output is where we start to see differences. Because there is no collinearity in this model, we now see a statistically significant result at the coefficient level (PC1). The other thing to remember here is that each principal component is a composite variable composed of all three predictors and perhaps had a more substantive interpretation. We need to use those substantive interpretations in the interpretations of coefficients:

- The intercept is the predicted average achievement for cases where all the composite variables are 0.
- The positive slope associated with the first composite indicates that higher values on this composite are associated with higher achievement, on average. In other words, higher values on all three predictors are associated with higher achievement, on average.
- The negative slope associated with the second composite indicates that higher values on this composite are associated with lower achievement, on average. In other words, larger contrasts between faculty credentials and peer influence are associated with lower achievement, on average.
- The positive slope associated with the third composite indicates that higher values on this composite are associated with higher achievement, on average. In other words, larger contrasts between school facilities and faculty credentials/peer influence are associated with higher

achievement, on average.

Again, these interpretations may not be satisfactory—it all depends on whether the loadings offer a reasonable interpretation of the composite variable.

Dimension Reduction

One of the most useful qualities of a PCA, aside from fixing collinearity issues, can be to reduce the size of the predictor space in a regression model; thereby reducing the complexity of the model and improving statistical power. To do this, we consider the variance accounted for by each of the principal components.

In our example, the first principal component accounted for most of the variance in the original predictor space (approximately 98%). The other two principal components did not account for that much variation, which suggests that they may not be necessary. We can capture most of the variation in the original three predictors by just using the first principal component.

```
# Fit reduced model using PC1
lm.pc.2 = lm(achievement ~ 1 + Comp.1, data = eeo2)

# Model-level output
glance(lm.pc.2)
```

```
# A tibble: 1 x 12
  r.squared adj.r.squared sigma statistic p.value    df logLik   AIC
      <dbl>         <dbl> <dbl>     <dbl>   <dbl> <dbl>  <dbl> <dbl>
1     0.183         0.171  2.07      15.2 2.19e-4     1  -149.  304.
# … with 4 more variables: BIC <dbl>, deviance <dbl>,
#   df.residual <int>, nobs <int>
```

```
# Coefficient-level output
tidy(lm.pc.2, conf.int = 0.95)
```

```
# A tibble: 2 x 7
  term        estimate std.error statistic  p.value conf.low conf.high
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
1 (Intercept)   0.0192     0.247    0.0776 0.938      -0.474     0.513
2 Comp.1        0.559      0.143    3.90   0.000219    0.273     0.845
```

In this model, the model R^2 value is slightly smaller (0.183 rather than 0.206) since the first principal component did not account for all of the variance in the original predictors. However, this difference is negligible, and the model is still explains a statistically relevant amount of variation in achievement scores, $F(1, 68) = 15.25, p = 0.0002$.

From the coefficient-level output, we see that the magnitude of the intercept and slope are comparable to those in the model where all three composites were used. The standard error and *p*-value for the composite in this model are slightly smaller than when we used all of the predictors. This represents the additional two degrees of freedom in the error term that we got from reducing the number of predictors. (In this example, the differences are negligible.)

SVD Decomposition

Another decomposition method that creates orthogonal unit vectors (i.e., basis) is SVD decomposition. Recall this decomposition method decomposes a matrix **A** as follows:

$$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$$

where, **U** and **V** are an orthogonal matrices and **D** is a diagonal matrix. In PCA, we are interested in performing SVD on the covariance matrix $\mathbf{X}^\top\mathbf{X}$. To carry out the SVD decomposition, we use the `svd()` function. Below, we create a matrix of the three predictors and then carry out the SVD decomposition on the covariance matrix.

```
# Create matrix of predictors
X = as.matrix(eeo[ , c("faculty", "peer", "school")])

# SVD decomposition
sv_decomp = svd(t(X) %*% X)

# View results
sv_decomp
```

```
$d
[1] 209.3295610   2.7303093   0.5833274

$u
           [,1]       [,2]       [,3]
[1,] -0.6174223  0.6698093 -0.4124865
[2,] -0.5243779 -0.7413256 -0.4188844
[3,] -0.5863595 -0.0423298  0.8089442

$v
           [,1]       [,2]       [,3]
[1,] -0.6174223  0.6698093 -0.4124865
[2,] -0.5243779 -0.7413256 -0.4188844
[3,] -0.5863595 -0.0423298  0.8089442
```

$$\mathbf{X}^\top\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^\top$$

$$\begin{bmatrix} 81.123 & 66.518 & 75.512 \\ \end{bmatrix} \quad \begin{bmatrix} 0.617 & 0.670 & -0.412 \\ \end{bmatrix} \quad \begin{bmatrix} 209.330 & 0 & 0 \\ \end{bmatrix} \quad \begin{bmatrix} -0.6 \\ \end{bmatrix}$$

$$\begin{bmatrix} 66.518 & 59.163 & 64.251 \\ 75.512 & 64.251 & 72.358 \end{bmatrix} = \begin{bmatrix} -0.524 & -0.741 & -0.419 \\ -0.586 & -0.042 & 0.809 \end{bmatrix} \begin{bmatrix} 0 & 2.730 & 0 \\ 0 & 0 & 0.583 \end{bmatrix} \begin{bmatrix} 0.6 \\ -0.4 \end{bmatrix}$$

Mathematically, since any matrix can be decomposed using SVD, we can also decompose $\mathbf{X} = \mathbf{UDV}^\intercal$. Then we can write $\mathbf{X}^\intercal\mathbf{X}$ as:

$$\mathbf{X}^\intercal\mathbf{X} = (\mathbf{UDV}^\intercal)^\intercal(\mathbf{UDV}^\intercal)$$

Re-expressing this we get:

$$\mathbf{X}^\intercal\mathbf{X} = \mathbf{VD}^\intercal\mathbf{U}^\intercal\mathbf{UDV}^\intercal$$

Since **D** is a diagonal matrix, $\mathbf{D}^\intercal\mathbf{D} = \mathbf{D}^2$, so reducing this expression gives

$$\mathbf{X}^\intercal\mathbf{X} = \mathbf{VD}^2\mathbf{V}^\intercal$$

The matrices **V** and $\mathbf{V}^\intercal$ are both orthogonal basis matrices that ultimately act to rotate the Cartesian bases. The $\mathbf{D}^2$ matrix is diagonalizing the covariance matrix which amounts to finding the the major axes in the data ellipse along which our data varies.

Using the SVD Decomposition as PCA

From a dimension reduction/PCA perspective, what is interesting is the **V** matrix.

$$\mathbf{V} = \begin{bmatrix} -0.617 & 0.670 & -0.412 \\ -0.524 & -0.741 & -0.419 \\ -0.586 & -0.042 & 0.809 \end{bmatrix}$$

These are the principal components. The values in the **D** matrix are related to the eigenvalues, and thus the variances of the principal components.

$$\lambda = \frac{d_{ii}^2}{n-1}$$

Squaring each of the diagonal elements of **D** and dividing by the total degrees of freedom gives us the associated eigenvalues. Then we can use these to compute the proportion of variance for each of the principal components.

```
# Compute lambda values (variances)
lambda = sv_decomp$d ^ 2 / (70 - 1)
lambda
```

```
[1] 6.350560e+02 1.080375e-01 4.931462e-03
```

```r
# Compute proportion of variance
lambda / sum(lambda)
```

```
[1] 9.998221e-01 1.700926e-04 7.764017e-06
```

If the SVD is carried out on the correlation matrix of the predictors rather than the variance–covariance matrix, the values in **D** are the λ values. Note also that the `cov()` function in R produced a variance–covariance matrix that has already been divided by $n - 1$, so if that is the input to the SVD, then there is also no need to divide by $n - 1$ in the
The principal component scores can be obtained by postmultiplying the mean centered predictor matrix **X** by the **V** matrix.

```r
# Mean center each predictor
X[, 1] = X[, 1] - mean(X[, 1])
X[, 2] = X[, 2] - mean(X[, 2])
X[, 3] = X[, 3] - mean(X[, 3])

# Compute PC scores
pc_scores = X %*% sv_decomp$v
head(pc_scores)
```

```
           [,1]        [,2]        [,3]
[1,] -0.41777127  0.37690208 -0.11724716
[2,] -0.98071768  0.15636966 -0.08255266
[3,]  1.36960233 -0.05831171 -0.02181284
[4,]  2.09547335  0.10933210  0.19860746
[5,] -0.02027426  0.25039535  0.13486066
[6,] -0.27859048  0.03203317  0.09791201
```

The `prcomp()` function carries out PCS using SVD decomposition. Different elements of the The `prcomp()` object can then be accessed to print output that is important to a PCA.

```r
# PCA using SVD decomposition
my_pca2 = prcomp(X)

# Get standard deviations/proportion of variance
summary(my_pca2)
```

```
Importance of components:
                          PC1     PC2     PC3
Standard deviation     1.7402 0.19890 0.09086
Proportion of Variance 0.9845 0.01286 0.00268
Cumulative Proportion  0.9845 0.99732 1.00000
```

```r
# Get matrix of principal components (eigenvector matrix)
my_pca2$rotation
```

```
               PC1         PC2        PC3
faculty -0.6173237  0.67025631 -0.4119077
peer    -0.5241853 -0.74086406 -0.4199407
school  -0.5866355 -0.04332342  0.8086915
```

```r
# Get PC scores
my_pca2$x
```

```
              PC1          PC2           PC3
 [1,] -0.41775535  0.377013352 -0.1169457640
 [2,] -0.98069934  0.156403711 -0.0827058802
 [3,]  1.36961355 -0.058197398 -0.0214096538
 [4,]  2.09539768  0.109232775  0.1994587407
 [5,] -0.02033702  0.250234525  0.1351494085
 [6,] -0.27862628  0.031899464  0.0978537560
 [7,] -0.05765964  0.229378988 -0.0075729005
 [8,] -0.71167315  0.217255930  0.0974081114
 [9,]  1.08107464 -0.019823990 -0.0380349105
[10,] -1.41400396  0.167294666  0.0983272711
[11,] -1.89400304  0.022075255  0.0392863309
[12,] -0.90101150 -0.272005138  0.0216696271
[13,] -1.25637710 -0.017431356 -0.0595493220
[14,]  1.21156669 -0.001642385  0.1346749133
[15,]  0.50438236 -0.136553019  0.0499154896
[16,] -2.19552202  0.038688617  0.0658700256
[17,] -2.44039555  0.084935412 -0.0706813561
[18,]  0.43441976 -0.138805721  0.0029381853
[19,] -2.06822631 -0.095252658 -0.0557049069
[20,]  2.48143874 -0.011516538  0.0708733872
[21,]  0.54611315 -0.198539188 -0.0936555767
[22,]  0.73037784  0.212917556 -0.1624804340
[23,] -1.68086687  0.215064163 -0.1100237827
[24,] -0.68754546  0.105150870  0.0514664645
[25,]  2.46251906 -0.251775706 -0.1006108111
[26,] -1.31633686 -0.158585661 -0.0569152008
[27,]  1.66269539 -0.085431981  0.0319262840
[28,]  2.86050854  0.077179366 -0.0790868818
[29,] -0.71229957 -0.166118941 -0.1590637380

[30,]  0.28734852 -0.215672978 -0.0554491781
[31,]  0.70445497  0.085572254  0.0161732263
[32,]  0.24118900 -0.035204243  0.2031905172
[33,]  3.29514377  0.004058795  0.0283640145
[34,] -1.16435715 -0.172366106 -0.0294020157
[35,]  0.34465273 -0.253602507 -0.1300963783
[36,]  1.03424153  0.222222713 -0.0005640946
[37,]  0.21749307 -0.008128879 -0.0897987029
[38,]  3.05002011 -0.331570043  0.0274009376
[39,]  2.01447802  0.225267569 -0.0427638210
[40,] -0.50642506  0.136613889 -0.0118769265
[41,] -1.24939036  0.132101509 -0.0742558614
[42,] -1.75078998  0.127586856  0.0328340790
```

```
[42,]  1.75078998  0.127386856 -0.052854979
[43,]  0.72774408  0.008806172 -0.082756405
[44,] -0.12652813 -0.268553127 -0.114897912
[45,]  0.22182854 -0.065802361  0.139008872
[46,] -1.04960073  0.209847482 -0.084319144
[47,]  3.58157257  0.064261501 -0.117998990
[48,] -0.11576350  0.400999804  0.165545558
[49,]  1.98978535 -0.330109349  0.057303483
[50,]  0.30277987  0.113209550  0.040954582
[51,]  0.39704453 -0.057936546 -0.207095079
[52,] -1.08520291 -0.261713483  0.041828924
[53,]  2.33913499 -0.180717119 -0.060030409
[54,]  2.06670464  0.377195713 -0.063253674
[55,] -0.86691998 -0.479475416  0.042927391
[56,]  0.85165358  0.185539523  0.156094151
[57,] -2.52466294 -0.285508747  0.085270397
[58,] -1.62025460 -0.022828544 -0.009118823
[59,]  1.32622917 -0.204817359 -0.049087573
[60,] -1.46612071  0.205383051  0.013507013
[61,]  3.40862545  0.215555063  0.099670719
[62,] -4.61075583 -0.051856944  0.065827185
[63,] -2.94935632  0.026276353  0.048973962
[64,]  1.59576019 -0.191178609  0.116200858
[65,] -2.11792874  0.074192316 -0.031612003
[66,] -0.54071593  0.229741569 -0.005424071
[67,] -0.84569988 -0.429495960  0.142438458
[68,] -2.51771060 -0.160286049  0.033388518
[69,] -3.25511953 -0.039162856 -0.052568607
[70,] -1.54293015  0.318510569 -0.061240994
```

In general it is more efficient to use singular value decomposition than eigendecomposition when carrying out a PCA. As such the use of `prcomp()` is recommended.

PCA from the Correlation Matrix

So far we have been applying the PCA to the covariance matrix. It can often make more sense to apply PCA to the correlation matrix, especially when the variables are measured in different metrics or have varying degrees of magnitude. In these cases, using the covariance matrix will often result in results in which variables with large magnitudes of scale dominate the PCA. To use the correlation matrix rather than the covariance matrix when computing the prinicpal components, we use the argument `scale = TRUE` in the `prcomp()` function.

```
# PCA using SVD decomposition; use correlation matrix of predictors
my_pca3 = prcomp(X, scale = TRUE)

# Get standard deviations/proportion of variance
summary(my_pca3)
```

```
Importance of components:
                          PC1     PC2     PC3
```

```
                          PC1     PC2     PC3
Standard deviation     1.718 0.20012 0.08922
Proportion of Variance 0.984 0.01335 0.00265
Cumulative Proportion  0.984 0.99735 1.00000
```

```
# Get matrix of principal components (eigenvector matrix)
my_pca3$rotation
```

```
               PC1         PC2        PC3
faculty -0.5761385  0.67939712 -0.4544052
peer    -0.5754361 -0.73197527 -0.3648089
school  -0.5804634  0.05130072  0.8126687
```

```
# Get PC scores
my_pca3$x
```

```
              PC1          PC2          PC3
 [1,] -0.36630991  0.366083254 -0.123459444
 [2,] -0.94977721  0.149343548 -0.084727422
 [3,]  1.34412355 -0.063278543 -0.019661363
 [4,]  2.08704153  0.128977212  0.192989279
 [5,]  0.01515640  0.266909829  0.126681564
 [6,] -0.26881618  0.043736173  0.095210936
 [7,] -0.02748806  0.229673318 -0.012776619
 [8,] -0.67241667  0.230646070  0.090457272
 [9,]  1.06386403 -0.026134791 -0.036853020
[10,] -1.37219342  0.181768424  0.092537736
[11,] -1.86607933  0.029940082  0.038043107
[12,] -0.92411758 -0.269230860  0.027612018
[13,] -1.24385723 -0.022169317 -0.057959986
[14,]  1.19879446  0.011689953  0.132069854
[15,]  0.48139865 -0.132336560  0.052120923
[16,] -2.16101996  0.050194885  0.063727714
[17,] -2.39976796  0.081435895 -0.071241348
[18,]  0.41101703 -0.139860379  0.006111503
[19,] -2.05528327 -0.098512321 -0.052359021
[20,]  2.44976450 -0.007725645  0.069715758
[21,]  0.51147071 -0.211127612 -0.087211303
[22,]  0.74497495  0.194033799 -0.164301261
[23,] -1.63401661  0.206352504 -0.112869183
[24,] -0.66406402  0.112717626  0.048024774
[25,]  2.39634941 -0.268708397 -0.092825678
[26,] -1.32120750 -0.163572431 -0.052082379
[27,]  1.63110467 -0.085034538  0.033266746
[28,]  2.83216679  0.063550398 -0.079403717
[29,] -0.72809081 -0.183882030 -0.152079301

[30,]  0.25463606 -0.223518554 -0.049343509
[31,]  0.70684120  0.086607838  0.013848414
[32,]  0.23798665 -0.012507535  0.200054088
[33,]  3.25414842  0.001650850  0.027654460
[34,] -1.17235034 -0.174526888 -0.024785850
```

```
[35,]  0.30468755 -0.270273220 -0.121654459
[36,]  1.04967957  0.221405970 -0.005758073
[37,]  0.21170748 -0.018827486 -0.087866077
[38,]  2.96882863 -0.335220713  0.034546120
[39,]  2.01685524  0.217942570 -0.047226556
[40,] -0.48257905  0.136758500 -0.014823814
[41,] -1.21798429  0.126359531 -0.075869619
[42,]  1.74415056  0.121400936 -0.035206637
[43,]  0.71777230 -0.001886201 -0.081365699
[44,] -0.16206634 -0.282740528 -0.106393837
[45,]  0.21352776 -0.050565960  0.137835477
[46,] -1.01093624  0.202968753 -0.087554998
[47,]  3.54149592  0.044873349 -0.117271128
[48,] -0.05893310  0.422019659  0.152970310
[49,]  1.92299129 -0.328501725  0.063853075
[50,]  0.31440849  0.117903698  0.037510439
[51,]  0.37998150 -0.082611286 -0.201721609
[52,] -1.10418661 -0.256265047  0.047142482
[53,]  2.28459622 -0.192459709 -0.054690472
[54,]  2.08757269  0.368137396 -0.070863738
[55,] -0.91677116 -0.475285807  0.053296498
[56,]  0.86810421  0.202812372  0.148711680
[57,] -2.52736408 -0.272714928  0.090321507
[58,] -1.60267993 -0.021187984 -0.008377765
[59,]  1.28177134 -0.213673386 -0.043378834
[60,] -1.42057581  0.210406950  0.008480000
[61,]  3.39502328  0.222099386  0.092636766
[62,] -4.55705387 -0.036614805  0.065844608
[63,] -2.90718418  0.037088167  0.047464410
[64,]  1.55321495 -0.181502142  0.118370378
[65,] -2.08196024  0.074562941 -0.032687618
[66,] -0.50427367  0.231115777 -0.010668820
[67,] -0.88720610 -0.413712532  0.149704902
[68,] -2.50547162 -0.152867180  0.036527049
[69,] -3.21969291 -0.039759058 -0.050569804
[70,] -1.48343272  0.315628485 -0.067451885
```

The variance accounted for by the principal components is comparable; the scaling does not change this. The actual principal components are different because of the scaling, but the interpretations are the same as when we used the covariance matrix. Similarly, because of the different scaling, the PC scores are different.

Here the results from using the correlation matrix are not that different as the variables were already *z*-scores.

Typically applied researchers will use the correlation matrix rather than the covariance matrix to perform a PCA. Working on the correlation matrix is akin to working with standardized variables. (In fact the correlation matrix is simply the variance–covariance matrix of the standardized predictors.) This protects the PCA from being dominated by variables with numerically large scales. It also is helpful when the predictors are measured using qualitatively different scales (e.g., one is measured in dollars and another in years of education).

References

References

Chatterjee, S., & Hadi, A. S. (2012). *Regression analysis by example*. Wiley.

Coleman, J. S., Cambell, E. Q., Hobson, C. J., McPartland, J., Mood, A. M., Weinfield, F. D., & York, R. L. (1966). *Equality of educational opportunity*. U.S. Government Printing Office.

Mosteller, F., & Moynihan, D. F. (1972). *On equality of educational opportunity*. Random House.