LSIRM Statistical/Machine Learning
SVD and PCA

Dave Armstrong
University of Western Ontario
Department of Political Science

e: dave.armstrong@uwo.ca
w: www.quantoid.net/teachwlu/

1 Introduction

- Discuss the Singular Value Decomposition as a tool to reveal the "basic structure of a matrix".
- Build biplots that can visually reveal low-dimensional structure in multivariate data.
  - We will tend to discuss both numerical and visual ways of investigating structure in data.
- Discuss PCA as a method of finding (calculating, not estimating) orthogonal, conditionally variance-maximizing linear combinations.
- Discuss Common Factor Analysis as a model of inter-relationships among multivariate data.
  - Discuss theoretical and empirical differences between PCA and CFA.
  - Discuss which you should use to tackle various problems.

2 Singular Value Decomposition

Suppose we have a matrix $\boldsymbol{X}$ of rank r (the smaller of # rows and # columns of $\boldsymbol{X}$). We want to approximate the information in $\boldsymbol{X}$ with a matrix called $\hat{\boldsymbol{X}}$ which has rank $p < r$.

- For any rank p, we want to minimize: trace $\left[(\boldsymbol{X} - \hat{\boldsymbol{X}})(\boldsymbol{X} - \hat{\boldsymbol{X}})'\right] = \sum_i \sum_m (x_{ij} - \hat{x_{ij}})^2$
- This is minimizing the sum of squared differences between our approximation and our observed matrix.
- The singular value decomposition does this for us.

The Singular Value Decomposition shows us the "basic structure of a matrix". First, some terminology:

- $\boldsymbol{X}$ is our $n \times m$ matrix of data where x_{ij} refers to an individual element (cell) of that matrix.

We are trying to find the underlying independent sources of variation in a matrix of multivariate data. The idea is relatively simple, we can decompose a matrix $\boldsymbol{X}$ into the product of three matrices:

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}' \tag{1}$$

where:

- $\boldsymbol{U}$ (an $n \times m$ matrix) summarizes information in the rows of $\boldsymbol{X}$, where the rows of $\boldsymbol{U}$ correspond to the rows of $\boldsymbol{X}$ and the columns of $\boldsymbol{U}$ correspond to the independent sources of variation in the rows of X.
- $\boldsymbol{D}$ (an $m \times m$ diagonal matrix) has non-negative entries on the diagonal and zeros on the off-diagonal. These entries d_{ii}, $i = \{1, \ldots, m\}$, are called *singular values*. The first entry in $\boldsymbol{D}$ (d_{11}) corresponds to the first column of $\boldsymbol{U}$ and the first column of $\boldsymbol{V}$. These are weights providing the relative importance of these independent sources of underlying variability ordered from largest to smallest.
- $\boldsymbol{V}$ (an $m \times m$ matrix) summarizes the information in the columns of $\boldsymbol{X}$, where the rows of $\boldsymbol{V}$ correspond to the columns of $\boldsymbol{X}$ and the columns of $\boldsymbol{V}$ correspond to the independent sources of information in the column variables.

2.1 Properties of the SVD Solution

- $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthonormal.
  - Each of the corresponding vectors (columns of $\boldsymbol{U}$ and $\boldsymbol{V}$) is of unit length. $\sum_{i=1}^{m} v_{i,j}^2 = 1$ and $\sum_{i=1}^{n} u_{i,j}^2 = 1$
  - Each of the corresponding vectors (columns of $\boldsymbol{U}$ and $\boldsymbol{V}$) is mutually independent of all other vectors in $\boldsymbol{U}$ and $\boldsymbol{V}$, respectively.
  - Put another way: $\boldsymbol{U}'\boldsymbol{U} = \boldsymbol{I}$ and $\boldsymbol{V}\boldsymbol{V}' = \boldsymbol{I}$.
- $d_{11} \geq d_{22} \geq \ldots \geq d_{mm}$

2.2 Example: Simple SVD

Here is a trivially simple example of an SVD, but it highlights some interesting properties. First, I want to make some data that has certain properties. I want there to be two variables, I want them to be uncorrelated and I want each to have a variance of 0.5 (so that all of the variance across the two variables equals 1).

```r
# load the MASS library for the mvrnorm() function
library(MASS)
# set the random seed so we always get the same answer
set.seed(123)
# make a diagonal matrix of order 2 (2 rows and columns)
# with 0 on off-diagonal and 0.5 on diagonal
sig1 <- diag(2)*.5
# draw 3 obs from a MVN with mean 0 and variance-covariance
# equal to sig1
X <- mvrnorm(3, c(0,0), sig1, empirical=T)
# print X
X
```

```
##             [,1]       [,2]
## [1,] -0.66293943 -0.4766319
## [2,]  0.74424506 -0.3358064
## [3,] -0.08130564  0.8124383
```

```r
# Do SVD of the X matrix
svd(X)
```

```
## $d
## [1] 1 1
##
## $u
##            [,1]        [,2]
## [1,] -0.4766319 -0.66293943
## [2,] -0.3358064  0.74424506
## [3,]  0.8124383 -0.08130564
##
## $v
##      [,1] [,2]
## [1,]    0    1
## [2,]    1    0
```

Things to note:

- The two entries in `$d` are both 1, meaning that to perfectly reproduce $\boldsymbol{X}$, the first and second columns of $\boldsymbol{U}$ and $\boldsymbol{V}$ are equally weighted in the linear combination.
- $\boldsymbol{V}$ (as is $\boldsymbol{V}'$) is a 2×2 matrix with zeros on the diagonal and ones on the off-diagonal. Indicating that the information in the first column of $\boldsymbol{U}$ (i.e., the left singular vectors) correspond to the second column of $\boldsymbol{X}$ and that the second column of $\boldsymbol{U}$ corresponds with the first column of $\boldsymbol{X}$.

- $\boldsymbol{U}$ looks like $\boldsymbol{X}$ only with the columns interchanged; since $\boldsymbol{X} = \boldsymbol{UDV}'$ and $\boldsymbol{X}$ is already orthonormal.

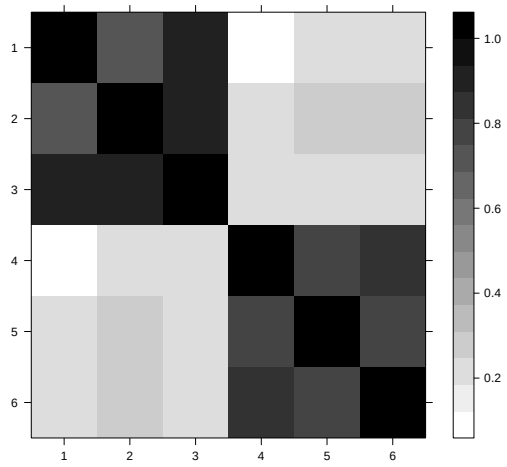
We can also look at a slightly more complicated example that will hopefully display some of the other properties of the technique.

- Here, we consider more than one underlying source of variation.

2.3 Simulated SVD Example

Here is a slightly more complicated example, but one that continues to highlight some important features of the SVD, but with more real-like data. Again, we want to generate some data that have the appropriate properties. We are going to generate 6 variables, where the first three are highly intercorrelated, the second three are also highly intercorrelated, but the off-block-diagonal correlations are relatively small (Figure 3 will make this clearer). Figure 3 shows the correlation matrix of the six variables.

Figure 1: Visual Correlation Matrix



Next, we can actually generate some data and do the SVD on those data.

Remember, that the $\boldsymbol{D}$ matrix with diagonal elements d_{mm} (Table 2.3) give the relative importance of the columns of $\boldsymbol{U}$ and $\boldsymbol{V}$ in accounting for variance in $\boldsymbol{X}$. So, what we want to know is how much variance can we capture if we use some number of columns of singular vectors and values. The output shows how much variance is captured by using $1:m$ singular vectors and values. Note that it takes all six to perfectly account for the total variance in the six variables.

```
d <- my.svd$d
cat("Cumulative Percentage of Variance Capture by Singular Vectors 1-m\n")

## Cumulative Percentage of Variance Capture by Singular Vectors 1-m
```

```
library(MASS)
X <- mvrnorm(250, mu=rep(0, nrow(sig)), Sigma=sig, empirical=T)
my.svd <- svd(X)
d.mat <- matrix(my.svd$d, ncol=1)
colnames(d.mat) <- "d"
d.mat
```

```
##              d
## [1,] 28.548200
## [2,] 22.230910
## [3,]  8.560075
## [4,]  8.043814
## [5,]  6.579173
## [6,]  1.877118
```

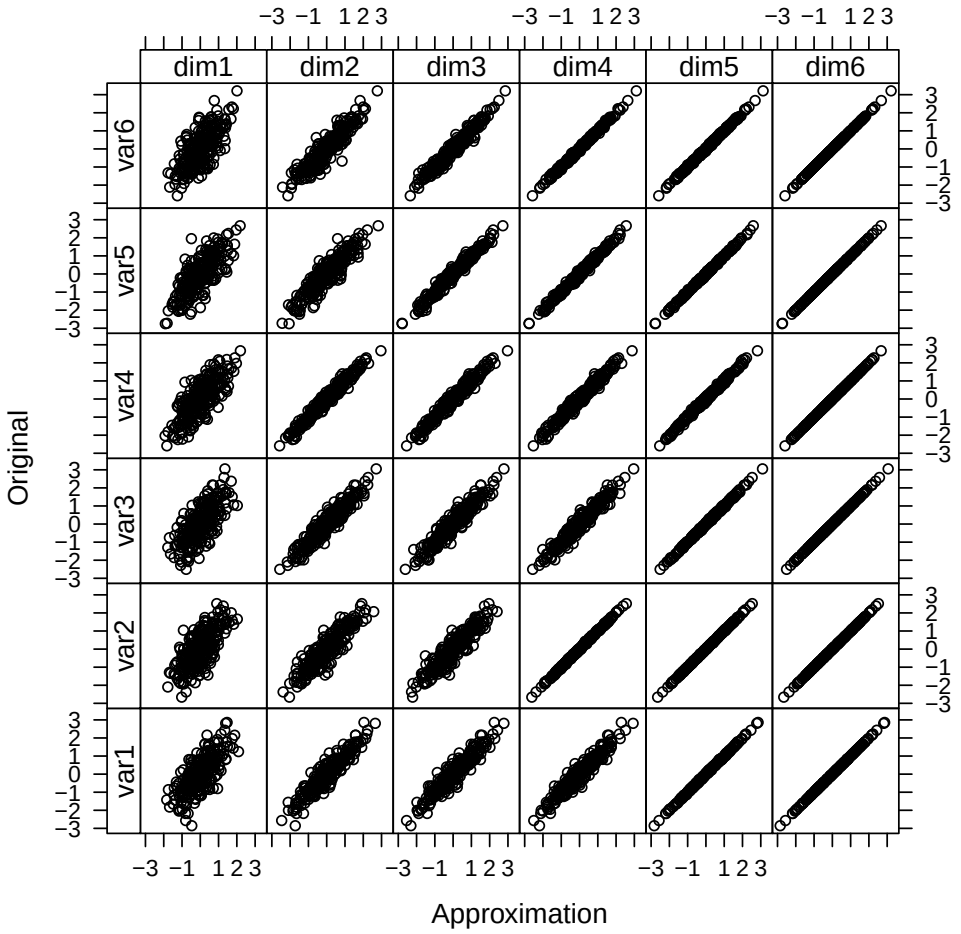
```
cumul.var <- cumsum(d^2)/sum(d^2)
cumul.var
```

```
## [1] 0.5455152 0.8763140 0.9253601 0.9686686 0.9976415 1.0000000
```

Note that 55% of the variance is accounted for by the first singular column of the vectors and the first singular value and that increases to 88% when using the first two columns of singular vectors and the first two singular values.

Another way of visualizing the fit of the lower-dimensioned approximation to the original data is to plot the original data against the low-dimensioned approximation as in Figure 4.

Figure 2: Comparison of Original X with Lower-Dimensioned Approximations



We can also see the correlation between the original and approximated variables. In the output below, the rows represent the variables and the columns represent the dimensions used to make the approximation. The entries are correlations.

```
##       dim1  dim2  dim3  dim4  dim5 dim6
## var1 0.739 0.937 0.949 0.961 1.000    1
## var2 0.716 0.908 0.929 0.999 1.000    1
## var3 0.716 0.947 0.951 0.968 0.999    1
## var4 0.776 0.981 0.981 0.985 0.997    1
## var5 0.771 0.917 0.985 0.994 0.999    1
## var6 0.710 0.925 0.976 0.998 0.999    1
```

2.4 SVD: Real Data Example

Now, here's a real data example

Now that we have an intuition for what the SVD does, let's look at it using some real data. Here, we are using data from Arthur Banks' Cross-National Time-Series Data Archive. (**?**). We are using the following data:

Variable	Description
ccode	COW code
year	year
country	country
importspc	Imports Per Capita
exportspc	Exports Per Capita
enprodkgpc	Energy Production in Kilograms Per Capita
enconskgpc	Energy Consumption, in Kilograms Per Capita
wfpcgind	Percent Work Force in Industry
newspaper	Daily Newspaper Circulation Per Capita
literate	Percent Literate
gdppcmp	Gross National Product Per Capita (Market Prices)

We could imagine that these variables together have relatively few underlying, independent sources of variation. Because these variables have such different variances, we standardize them to ensure that no one variable is having undue influence on the result.

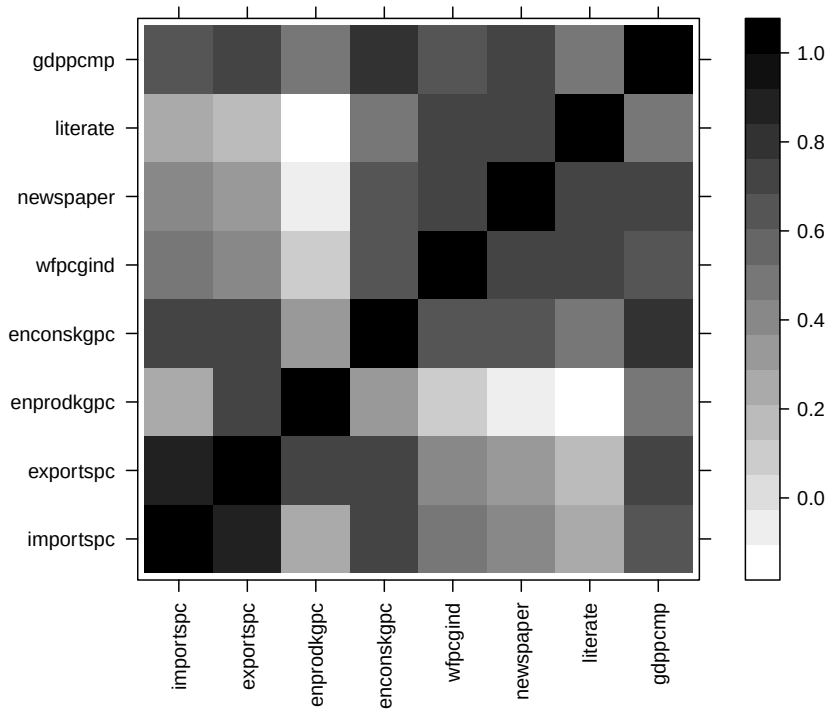
We could look at a picture of the correlation matrix to get a sense of what is going on here as in Figure 3.

Next, we can perform the SVD on our standardized data. The output below shows the singular values. You can see that the first two dimensions have the greatest relative importance

```
my.svd <- svd(tmp2)
d.mat <- matrix(my.svd$d, ncol=1)
colnames(d.mat) <- "d"
d.mat
```

```
##               d
## [1,] 19.2484470
## [2,] 11.8427368
## [3,]  6.9686341
## [4,]  5.2092127
## [5,]  4.4020936
## [6,]  4.1890078
## [7,]  2.7680921
## [8,]  0.9800848
```

Figure 3: Visual Correlation Matrix




```
## Total Variance in X
## [1] 8
## Cumulative Percentage of Variance Capture by Singular Vectors 1-m
## [1] 0.5862385 0.8081537 0.8849921 0.9279286 0.9585907 0.9863562 0.9984801
## [8] 1.0000000
```

Note that 59% of the variance is accounted for by the first singular vector and value and that increases to 81% when using the first two singular vectors and values. Another way of visualizing the fit of the lower-dimensioned approximation to the original data is to plot the original data against the low-dimensioned approximation as in Figure 4.
We can also see the correlation between the original and approximated variables. In Table 1, the rows represent the variables and the columns represent the dimensions used to make the approximation. The entries are correlations.

Figure 4: Comparison of Original X with Lower-Dimensioned Approximations

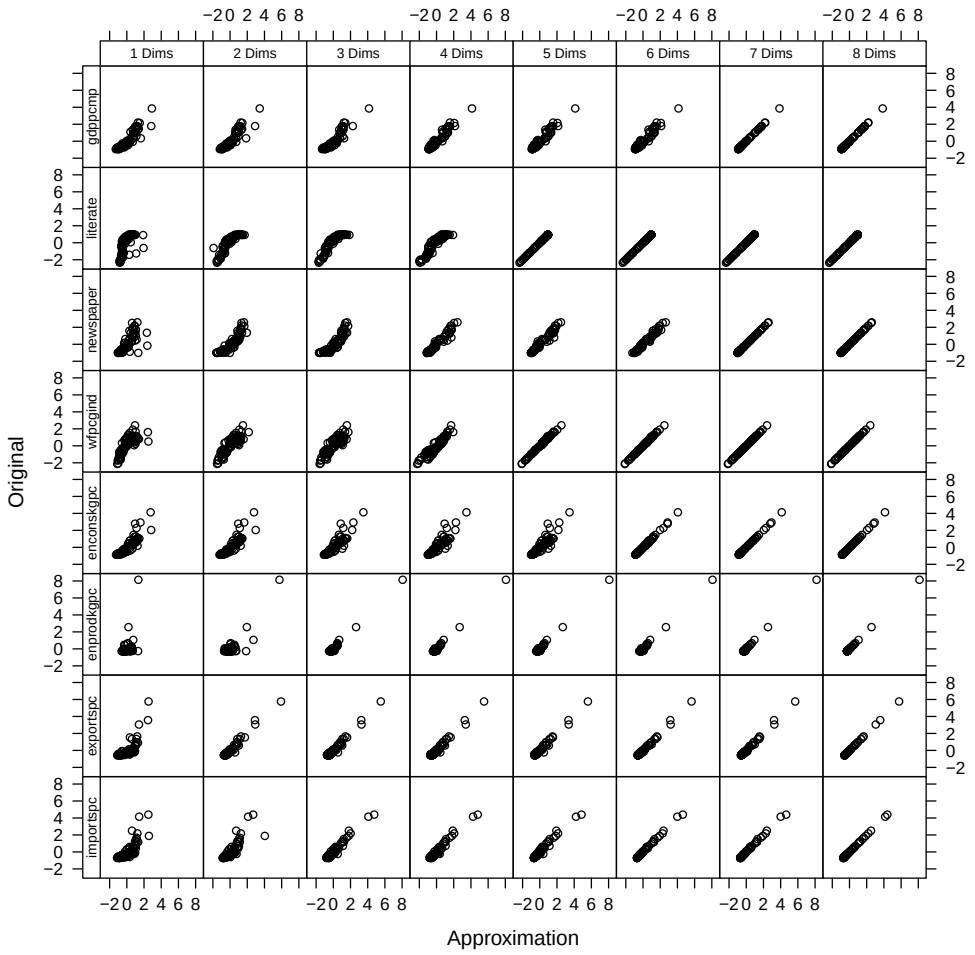


Table 1: Correlations between original and approximated variables (rows) for different dimensional approximations (columns)

	1	2	3	4	5	6	7	8
1	0.81	0.85	0.98	0.98	0.99	1.00	1.00	1.00
2	0.80	0.98	0.99	0.99	0.99	1.00	1.00	1.00
3	0.42	0.86	0.99	1.00	1.00	1.00	1.00	1.00
4	0.89	0.89	0.91	0.91	0.91	1.00	1.00	1.00
5	0.79	0.88	0.88	0.95	1.00	1.00	1.00	1.00
6	0.75	0.91	0.91	0.97	0.98	0.99	1.00	1.00
7	0.61	0.89	0.92	0.93	1.00	1.00	1.00	1.00
8	0.92	0.92	0.94	0.97	0.97	0.97	1.00	1.00

3 Eigen Decomposition

The Eigen-decomposition is a special case of an SVD. Specifically, it is an SVD performed on a square symmetric matrix.

If our data are $\boldsymbol{X}$, then we could consider two square, symmetric matrices derived from $\boldsymbol{X}$; namely $\boldsymbol{X}'\boldsymbol{X}$, which will be an $m \times m$ matrix of cross-products of the variables (columns) and $\boldsymbol{X}\boldsymbol{X}'$, which is an $n \times n$ matrix of cross-products on the observations (rows). Let's put this in terms of the SVD.

- First, we know:

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}' \tag{2}$$

- Consider, then, $\boldsymbol{X}'\boldsymbol{X}$:

$$\begin{aligned} \boldsymbol{X} &= \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}' && (3)\\ \boldsymbol{X}'\boldsymbol{X} &= (\boldsymbol{U}\boldsymbol{D}\boldsymbol{V}')'\boldsymbol{U}\boldsymbol{D}\boldsymbol{V}' && (4)\\ &= \boldsymbol{V}\boldsymbol{D}\boldsymbol{U}'\boldsymbol{U}\boldsymbol{D}\boldsymbol{V}' && (5)\\ &= \boldsymbol{V}\boldsymbol{D}\boldsymbol{I}\boldsymbol{D}\boldsymbol{V}' && (6)\\ &= \boldsymbol{V}\boldsymbol{D}^2\boldsymbol{V}' && (7) \end{aligned}$$

  where $\boldsymbol{D}^2$ is a diagonal matrix with d_{mm}^2 on the diagonal.

  - Now, the elements in $\boldsymbol{D}^2$ are referred to as eigenvalues and what were the right singular vectors are called eigenvectors.
  - When we have a square symmetric matrix, the left and right singular vectors are the same.

- Now, consider $\boldsymbol{X}\boldsymbol{X}'$:

$$\begin{aligned} \boldsymbol{X} &= \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}' && (8)\\ \boldsymbol{X}\boldsymbol{X}' &= \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}'(\boldsymbol{U}\boldsymbol{D}\boldsymbol{V}')' && (9)\\ &= \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}'\boldsymbol{V}\boldsymbol{D}\boldsymbol{U}' && (10)\\ &= \boldsymbol{U}\boldsymbol{D}^2\boldsymbol{U}' && (11) \end{aligned}$$

Some things to note:

1. The eigenvectors and singular vectors (either left or right, where appropriate) have the same properties, since it is easy to recast one solution as another.
2. The eigenvalues are squared singular values.
3. We will put a more substantive interpretation on these quantities when we use the tools to solve for the parameters of the CFA model.

4 Biplots

Biplots allow us to visually depict the relationship among variables on the dimensions and (can) show the observations on the relevant dimensions as well.

- Variables are depicted as vectors where their lengths are equal to their respective variances and the cosines of the angles between the vectors are the correlations between the low-dimensional depiction of the representations of the variables.

4.1 Biplot Coordinates

First, we recognize that we can re-state the singular value decomposition as:

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{D}^{\alpha}\boldsymbol{D}^{(1-\alpha)}\boldsymbol{V}' \tag{12}$$

- We can plot the observations (i.e., the rows of $\boldsymbol{X}$) with $\boldsymbol{U}_k\boldsymbol{D}_k^{\alpha}$
- We can plot the variables (i.e., the columns of $\boldsymbol{X}$) with $\boldsymbol{D}_k^{(1-\alpha)}\boldsymbol{V}'$.

where k represents the dimensionality of the approximation of $\boldsymbol{X}$ and different values of α represent different solutions.

- When $\alpha = 1$, the biplot is referred to as the Principal Components biplot because, as we will see later, this results in principal component scores and principal component coefficients.
- When $\alpha = 0.5$, it is called a "symmetric factorization" which *tends* to place points and vectors in the plot in a manner that is easy to see both.
- Any value of α, such that $0 \leq \alpha \leq 1$ are possible, but values 0, 0.5 and 1 are most common.

[See biplot with variable alpha in Scaling1.R]

The placement of the points relative to the variable vectors is meaningful. Points with greater values on each vector have greater values on the two-dimensional approximation of those variables.

4.1.1 Example: Biplot for Simulated Sata

We can use our previous simulated SVD solution in Handout 2 to discuss the first biplot. Remember, we can find coordinates in the following way:

```
set.seed(123)
sig <- makeMVsigma(k=c(3,3), m=2, within=matrix(c(.7,.9),
    ncol=2, nrow=2, byrow=T), between=c(.1,.3))

library(MASS)
X <- mvrnorm(250, mu=rep(0, nrow(sig)), Sigma=sig, empirical=T)
```

rows $\boldsymbol{U}_k\boldsymbol{D}_k^{\alpha}$

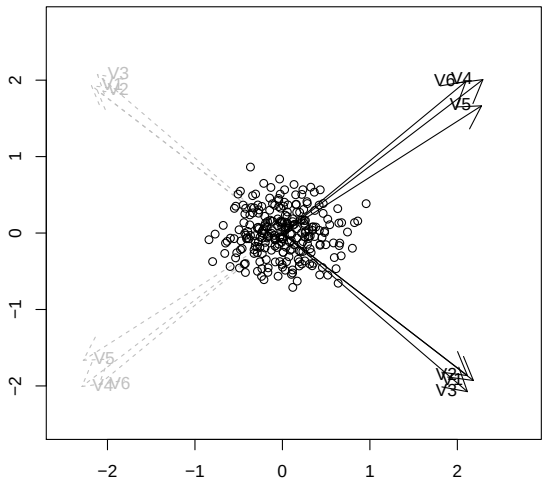
columns $\boldsymbol{D}_k^{(1-\alpha)}\boldsymbol{V}'$.

Our data are in the object `X`.

```
my.svd <- svd(X)
alpha <- .5
k <- 2
row.vals <- my.svd$u[,1:k] %*% diag(my.svd$d[1:k]^alpha)
col.vals <- t(diag(my.svd$d[1:k]^(1-alpha)) %*% t(my.svd$v[,1:k]))
```

We can use the `row.vals` and `col.vals` objects to plot the points and vectors. You can do this yourself using `points()` and `arrows()` commands, or you could do `biplot(row.vals, col.vals)` which will give you something similar, but not exactly the same.

Figure 5: Biplot with $\alpha = .5$



4.2 Example: SVD - Banks Data

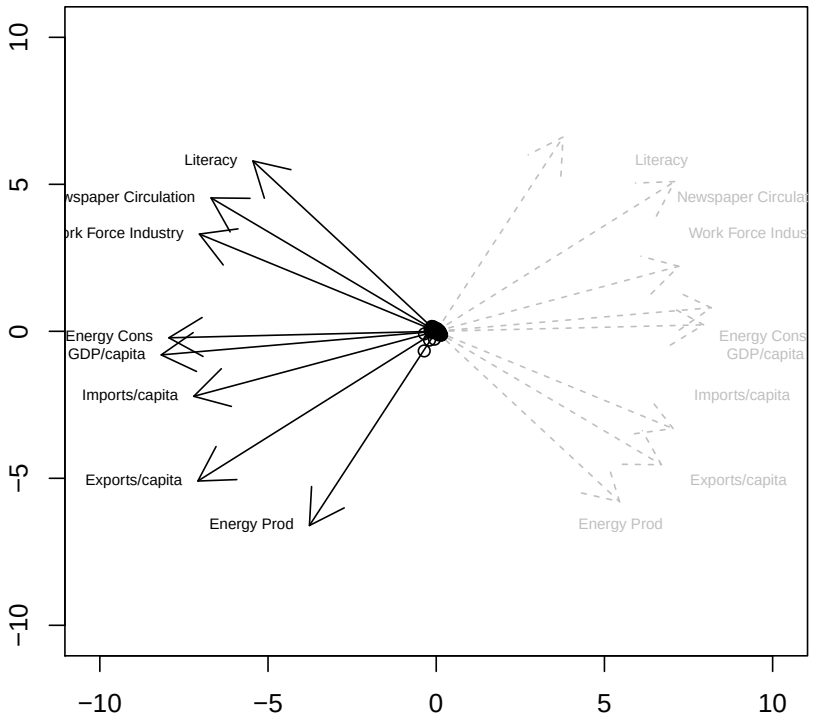
To make a biplot of the Banks data we used in Handout 3, we first need to read in the data and compute the SVD.

```
library(foreign)
tmp <- read.dta("http://www.quantoid.net/files/essex/banks_biplot.dta")
label.names <- c("Imports/capita", "Exports/capita",
```

```
        "Energy Prod", "Energy Cons", "Work Force Industry",
        "Newspaper Circulation", "Literacy", "GDP/capita")
tmp2 <- scale(tmp[,-(1:3)])
my.svd <- svd(tmp2)
alpha <- 0
k <- 2
row.vals <- my.svd$u[,1:k] %*% diag(my.svd$d[1:k]^alpha)
col.vals <- t(diag(my.svd$d[1:k]^(1-alpha)) %*% t(my.svd$v[,1:k]))
rownames(col.vals) <- label.names
```

Next, we can construct the biplot using the SVD results

Figure 6: Biplot with Banks Data



5 Principal Components

Principal Components Analysis finds orthogonal, variance maximizing linear combinations.

- intuitively, we are trying to find the natural axes of multivariate data.

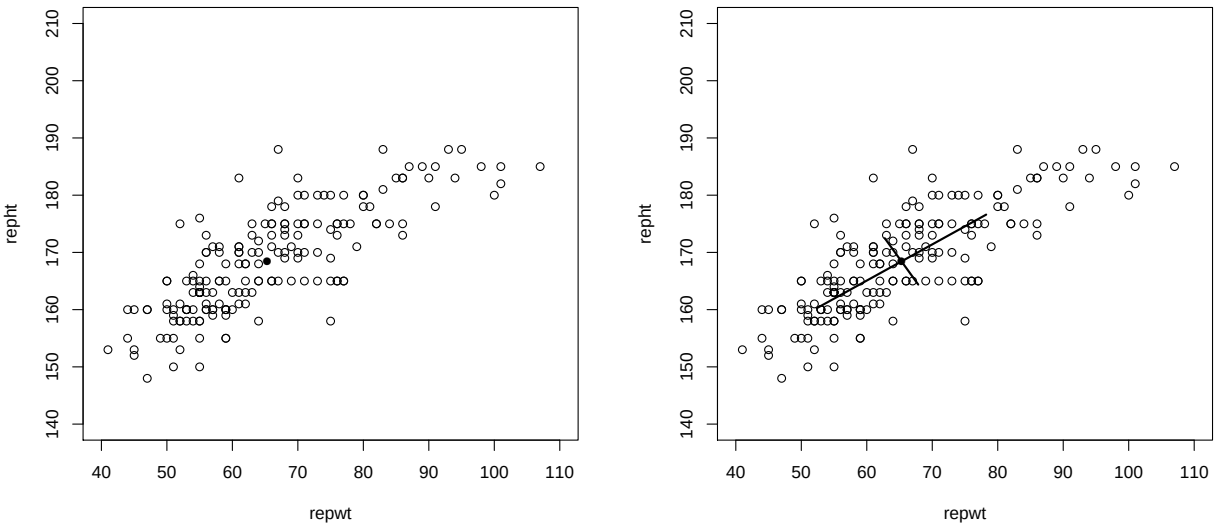
- Trying to maximize the variance of the perpendicular projections on each axis (subject to orthogonality). The major axis could be seen as a regression line where the goal is to minimize perpendicular distance to the line.
- Easy to visualize in 2 dimensions.

First, let's try to generate some intuition about what we are trying to do. We can discuss the details of how we get here later.

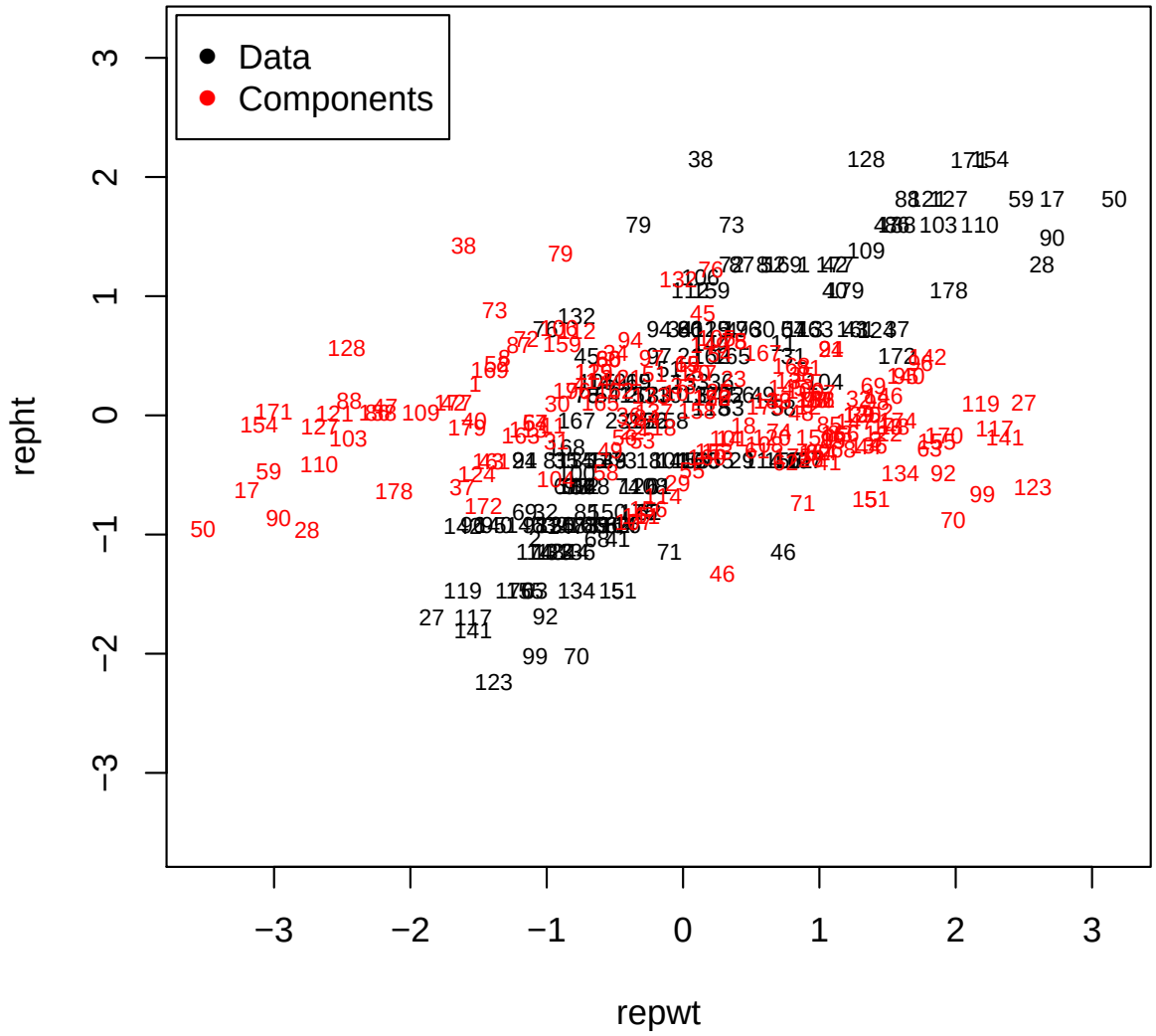
- Trying to express the information in variables in such a way that:
  - The new variables capturing the information are uncorrelated
  - The first variable has the most variance and the second variable has the second most variance.

This amounts to a (potential reflection), rotation and translation of the original data.

- When we start with k variables and use k components, you'll notice that the information is not changing, we're just expressing it a different way.
- This is not necessarily true if we use $m < k$ components.



We can also think about this in terms of rotating and translating the initial data matrix.



5.1 Details of Calculation

First, this is not really "estimation" in the sense of estimating a population parameter.

- This is more like "calculation".
- We want to know *the* values, subject to some constraints we will discuss, that make a particular thing happen.
- As we will show later, this particular thing is not necessarily returning some population parameter, but finding values that make things happen for this sample.

We are trying to do the following:

$$C_k = a_{k1}x_1 + a_{k2}x_2 + \ldots + a_{kk}x_k \tag{13}$$

such that

- $var(C_1) \geq var(C_2) \geq \ldots \geq var(C_k)$.
- $cor(C_j C_m) = 0 \quad j \neq m$
- $\sum_m a_{km}^2 = 1$
- As a result of the above, these also maximize: $\sum_m r_{c_k x_m}^2$ subject to the constraints above.

We can also give the goal in matrix form:

$$\boldsymbol{C} = \boldsymbol{X}\boldsymbol{A} \tag{14}$$

We need to think about what tools we have that might allow us to solve this problem. Not surprisingly, the SVD can help us. Let's think about what we need and what the SVD gives us.

- We need:

$$\boldsymbol{C} = \boldsymbol{X}\boldsymbol{A} \tag{15}$$

- The SVD gives us:

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}' \tag{16}$$

Now, what if we think about $\boldsymbol{V} = \boldsymbol{A}$, where does that get us:

$$\begin{aligned} \boldsymbol{X} &= \boldsymbol{U}\boldsymbol{D}\boldsymbol{A}' && (17)\\ \boldsymbol{X}\boldsymbol{A} &= \boldsymbol{U}\boldsymbol{D}\boldsymbol{A}'\boldsymbol{A} && (18)\\ &= \boldsymbol{U}\boldsymbol{D}\boldsymbol{I} && (19)\\ &= \underbrace{\boldsymbol{U}\boldsymbol{D}}_{\boldsymbol{C}} && (20) \end{aligned}$$

Note that $\boldsymbol{U}$ is ortho-normal, meaning the columns of $\boldsymbol{U}$ are mutually orthogonal - all with mean 0 and unit variance. We know that $var(\boldsymbol{c}_1) \geq var(\boldsymbol{c}_2) \ldots$, so we need to post-multiply the $\boldsymbol{U}$ by $\boldsymbol{D}$ to "spread out" the values of $\boldsymbol{c}_1$ relative to $\boldsymbol{c}_2$ and so on.

You might notice from this the reason that the "principal component biplot" is one where α is set to 1. Remember:

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{D}^{\alpha}\boldsymbol{D}^{(1-\alpha)}\boldsymbol{V}' \tag{21}$$

- We can plot the observations (i.e., the rows of $\boldsymbol{X}$) with $\boldsymbol{U}_k\boldsymbol{D}_k^{\alpha}$
- We can plot the variables (i.e., the columns of $\boldsymbol{X}$) with $\boldsymbol{D}_k^{(1-\alpha)}\boldsymbol{V}'$.

5.2 Important Note about PCA

- PCA (and the underlying SVD) are scale-*dependent*.
- If variables have wildly different variances, this could result in some unwanted behavior of the SVD (i.e., items with bigger variances will get bigger weights on $\boldsymbol{c}_1$ to maximize variance explained).
- As such, it is often wise to center and standardize variables unless you have a particularly good reason to do otherwise.
- Above is how the `prcomp()` command in R proceeds.
- Stata does what we will talk about next.

5.2.1 Another Way: Correlation or Covariance Matrices

Though this is not generally the way we proceed,

First, let's re-cast the problem in terms of a square, symmetric matrix.

$$\begin{aligned}
\boldsymbol{X} &= \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}' && (22)\\
\boldsymbol{X}'\boldsymbol{X} &= (\boldsymbol{U}\boldsymbol{D}\boldsymbol{V}')'\boldsymbol{U}\boldsymbol{D}\boldsymbol{V}' && (23)\\
&= \boldsymbol{V}\boldsymbol{D}\boldsymbol{U}'\boldsymbol{U}\boldsymbol{D}\boldsymbol{V}' && (24)\\
&= \boldsymbol{V}\boldsymbol{D}\boldsymbol{I}\boldsymbol{D}\boldsymbol{V}' && (25)\\
&= \boldsymbol{V}\underbrace{\boldsymbol{D}^{\boldsymbol{2}}}_{\boldsymbol{\Lambda}}\boldsymbol{V}' && (26)
\end{aligned}$$

We know from before, that the eigen decomposition is a way to decompose a square, symmetric matrix into eigen-vectors (similar to the right singular-vectors from an SVD) and eigen-values (similar to the singular values). In fact, an SVD and eigen decomposition are equivalent on a square, symmetric matrix.

So, we could simply do the eigen decomposition on the covariance matrix (if we want items to contribute different variances to the total variance) or correlation matrix (if we want all items to contribute the same variance to the total variance). Then, the $\boldsymbol{V}$ matrix is the matrix of coefficients such that $\boldsymbol{X}\boldsymbol{V}$ are the components.

5.3 Example: PCA with Banks Data

We can use the Banks data to perform a Principal Components Analysis. First, let's load the data and do a bit of investigating.

```
library(foreign)
dat <- read.dta("https://quantoid.net/files/essex/banks_biplot.dta")
apply(dat[,-(1:3)], 2, sd)
```

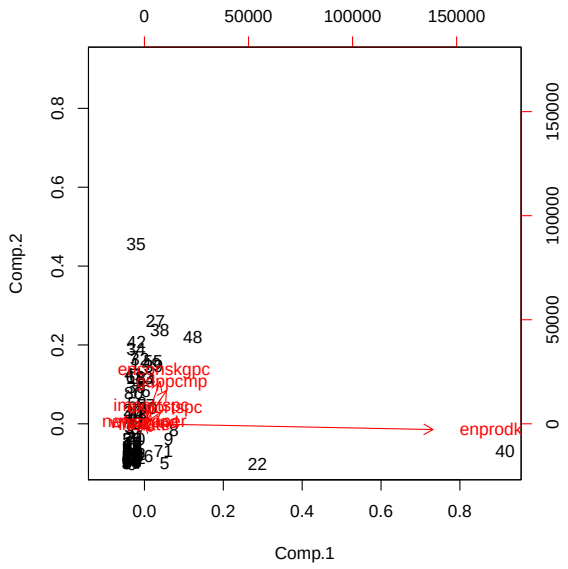
```
##    importspc    exportspc   enprodkgpc   enconskgpc      wfpcgind
## 1.235312e+03 1.505682e+03 1.950021e+04 3.155460e+03 1.127838e+01
##    newspaper     literate      gdppcmp
## 1.512383e-01 2.651987e+01 2.888253e+03

round(apply(dat[,-(1:3)], 2, sd)/max(apply(dat[,-(1:3)], 2, sd)), 2)

##  importspc  exportspc enprodkgpc enconskgpc   wfpcgind  newspaper
##       0.06       0.08       1.00       0.16       0.00       0.00
##   literate    gdppcmp
##       0.00       0.15
```

Now, we could do the Biplot with $\alpha = 1$, generating the PCA solution:

```
pdf("biplot_banks_noscale.pdf", height=6, width=6)
biplot(princomp(dat[,-(1:3)]))
invisible(dev.off())
```



Let's see what the PCA solution looks like:

```
pca1 <- princomp(dat[,-(1:3)], center=T, scale=F)
summary(pca1, loadings=T, cutoff=0)

## Importance of components:
##                             Comp.1       Comp.2       Comp.3       Comp.4
## Standard deviation     1.949281e+04 3.825690e+03 1.304910e+03 9.921425e+02
## Proportion of Variance 9.563275e-01 3.683641e-02 4.285669e-03 2.477456e-03
## Cumulative Proportion  9.563275e-01 9.931639e-01 9.974495e-01 9.999270e-01
##                             Comp.5       Comp.6       Comp.7       Comp.8
## Standard deviation     1.690254e+02 1.980468e+01 6.546857e+00 7.142626e-02
```

```
## Proportion of Variance 7.190542e-05 9.871735e-07 1.078757e-07 1.284026e-11
## Cumulative Proportion  9.999989e-01 9.999999e-01 1.000000e+00 1.000000e+00
##
## Loadings:
##            Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
## importspc   0.019  0.240  0.123  0.701  0.659  0.007  0.005  0.000
## exportspc   0.056  0.207  0.084  0.619 -0.751 -0.015 -0.004  0.000
## enprodkgpc  0.994 -0.101  0.023 -0.024  0.029  0.000  0.000  0.000
## enconskgpc  0.054  0.742  0.572 -0.345 -0.012  0.003  0.001  0.000
## wfpcgind    0.000  0.002 -0.001 -0.001  0.010 -0.226 -0.974 -0.003
## newspaper   0.000  0.000  0.000  0.000  0.000 -0.002 -0.003  1.000
## literate    0.000  0.004 -0.004 -0.006  0.014 -0.974  0.226 -0.001
## gdppcmp     0.076  0.582 -0.806 -0.074  0.015  0.007  0.000  0.000
```

Rather than use this solution that chases the variance in the first three variables, we could use the correlation matrix.

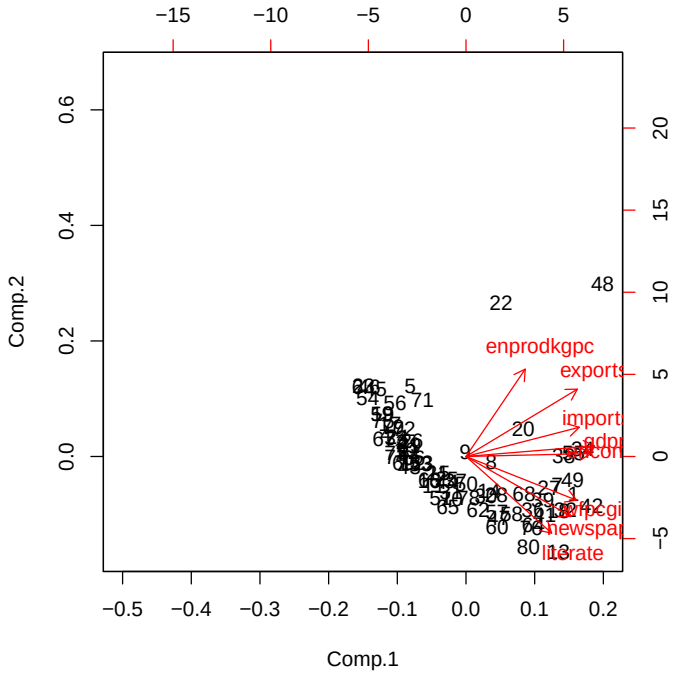
```
pca2 <- princomp(dat[,-c(1:3)], cor=TRUE)
```

```
summary(pca2, loadings=T, cutoff=0)

## Importance of components:
##                            Comp.1    Comp.2     Comp.3     Comp.4
## Standard deviation     2.1656195 1.3324120 0.78403259 0.58608222
## Proportion of Variance 0.5862385 0.2219152 0.07683839 0.04293655
## Cumulative Proportion  0.5862385 0.8081537 0.88499207 0.92792862
##                            Comp.5     Comp.6     Comp.7      Comp.8
## Standard deviation     0.49527422 0.47130020 0.31143470 0.110268160
## Proportion of Variance 0.03066207 0.02776548 0.01212395 0.001519883
## Cumulative Proportion  0.95859069 0.98635617 0.99848012 1.000000000
##
## Loadings:
##            Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
## importspc   0.374  0.187  0.627  0.126  0.160  0.323  0.032  0.533
## exportspc   0.368  0.430  0.110  0.091  0.146  0.231  0.135 -0.753
## enprodkgpc  0.196  0.558 -0.643  0.108  0.022 -0.014  0.278  0.383
## enconskgpc  0.413  0.019  0.218 -0.064 -0.059 -0.862  0.174 -0.008
## wfpcgind    0.366 -0.279 -0.127  0.622 -0.595  0.104 -0.137 -0.032
## newspaper   0.348 -0.383 -0.105 -0.557 -0.235  0.294  0.519  0.006
## literate    0.283 -0.489 -0.261  0.269  0.735 -0.007  0.021 -0.003
## gdppcmp     0.425  0.068 -0.195 -0.437 -0.005  0.034 -0.764  0.038
```

We could also make a biplot using this version of the PCA.

```
pdf("biplot_banks_scale.pdf", height=6, width=6)
biplot(pca2, xlim=c(-.5, .2))
invisible(dev.off())
```



5.4 Assessing Dimensionality

- The goal of PCA is (or we argue, ought to be) dimension reduction: fewer variables that capture much of the original *variance* across a set of observed variables.
- We are not theorizing about a particular number of dimensions (i.e., there is no underlying model of the data).
- Ultimately, what we find out is that the first m components explain $w\%$ of the variance. The question is: "is that enough?". Analogously, let me ask:

  Q Can you get where you need to go on half a tank of gas?

  A Depends on where you are trying to go.

  Dimensionality in PCA is similar - sometimes we may need only one dimension, sometimes more. Since PCA is not a "model" per se, it may not be particularly useful in its own right. However, it is good for reducing dimensionality of a set of correlated variables.

5.5 Example: PCA with Democracy Data

Another example, close to my own research, is using democracy data. Let's take a look at the polity data.

Table 2: Codebook of Polity Data (1999)

Variable	Description
ccode	COW country code
scode	Three-letter country abbreviation
country	Full country name
democ_polity	Polity democracy score
autoc	Polity autocracy score
polity	Democracy - Autocracy
xrreg	Regulation of Executive Recruitment
xrcomp	Competitiveness of Executive Recruitment
xropen	Openness of Executive Recruitment
xconst	Executive Constraints
parcomp	Competitiveness of Political Participation

```
library(foreign)
pol99 <- read.dta("https://quantoid.net/files/lsirm/polity_1999.dta")
dem99 <- pol99[,7:11]
pol.pca <- princomp(scale(dem99), center=T)
summary(pol.pca, loadings=T, cutoff=0)

## Importance of components:
##                           Comp.1    Comp.2     Comp.3     Comp.4
## Standard deviation     1.9102614 0.7777094 0.69609638 0.40060495
## Proportion of Variance 0.7347509 0.1217837 0.09756483 0.03231374
## Cumulative Proportion  0.7347509 0.8565346 0.95409947 0.98641321
##                            Comp.5
## Standard deviation     0.25976533
## Proportion of Variance 0.01358679
## Cumulative Proportion  1.00000000
##
## Loadings:
##         Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## xrreg    0.417  0.269  0.791  0.184  0.306
## xrcomp   0.504 -0.112  0.126 -0.131 -0.837
## xropen   0.389 -0.836 -0.089  0.231  0.297
## xconst   0.481  0.169 -0.271 -0.741  0.343
## parcomp  0.435  0.432 -0.526  0.588  0.033
```

We can attach the component scores to an existing dataset that has some interesting data regarding state repression and some state characteristics. After adding in the data, we can also run some models to figure out how these work.

```
data <- read.dta("https://quantoid.net/files/lsirm/polity_model_data.dta")
data[, c("comp1", "comp2", "comp3", "comp4", "comp5")] <-
        pol.pca$scores[match(data$ccode, pol99$ccode), 1:5]
data[, names(dem99)] <- dem99[match(data$ccode, pol99$ccode),]
mod1 <- lm(rep1 ~ gdppc + logpop + polity_democ , data=data)
mod2 <- lm(rep1 ~ comp1 + comp2 + gdppc + logpop, data=data)
mod2a <- lm(rep1 ~ comp1  + gdppc + logpop, data=data)
```

Table 3: Codebook of Repression Data (1999)

Variable	Description
ccode	COW Country Code
country	Full Country Name
rep1	State Repression of Physical Integrity Rights
democ	Democracy score from **?**
polity_democ	Polity IV Democracy Variable
gdppc	GDP/capita (in $10,000)
logpop	Natural logarithm of population
cwar	Civil War dummy variable
iwar	Interstate War dummy variable
polpris	Political Imprisonment dummy variable

As mentioned in class, sometimes the dimensionality of data changes as a function of the problem. Consider a binary logit of political imprisonment as a function of GDP/capita, population and democracy. We can adopt a strategy similar to the one above:

```
lmod1 <- glm(polpris ~ gdppc + logpop + polity_democ ,
         data=data, family=binomial)
lmod2 <- glm(polpris ~ comp1 + comp2 + gdppc + logpop,
         data=data, family=binomial)
```

There are some other things we need to care about:

Linear Combinations

By linear combination (as Bill suggested earlier), we mean

$$Y = a_1X_1 + a_2X_2 + \ldots + a_kX_k \tag{27}$$

where a_j are constants (scalars) and X_j are variables (i.e., things with variances).

Table 4: State Repression Model Results using Polity and PCA Scores

	Model 1	Model 2	Model 3
Intercept	−2.67*	−3.55*	−3.61*
	(0.58)	(0.59)	(0.58)
GDP/capita (in $10000)	−1.15*	−1.12*	−1.15*
	(0.13)	(0.13)	(0.13)
log(Population)	0.47*	0.47*	0.48*
	(0.06)	(0.06)	(0.06)
Democracy (Polity)	−0.16*		
	(0.03)		
Component 1		−0.34*	−0.34*
		(0.05)	(0.05)
Component 2		−0.12	
		(0.12)	
N	147	147	147
R^2	0.66	0.67	0.67
adj. R^2	0.65	0.66	0.66
Resid. sd	1.10	1.09	1.09

Standard errors in parentheses

* indicates significance at $p < 0.05$

Table 5: Political Imprisonment Model Results using Polity and PCA Scores

	Model 1	Model 2
Intercept	−3.91*	−5.73*
	(1.41)	(1.55)
GDP/capita (in $10000)	−0.40	−0.26
	(0.29)	(0.30)
log(Population)	0.69*	0.67*
	(0.17)	(0.17)
Democracy (Polity)	−0.37*	
	(0.07)	
Component 1		−0.78*
		(0.16)
Component 2		−0.77*
		(0.33)
N	147	147
AIC	139.49	143.20
BIC	187.34	203.01
$\log L$	−53.75	−51.60

Standard errors in parentheses

* indicates significance at $p < 0.05$

In matrix form, a linear combination is as follows:

$$\boldsymbol{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_k \end{bmatrix} \quad \boldsymbol{X} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1k} \\ x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & \vdots \\ x_{k1} & x_{k2} & \cdots & x_{kk} \end{bmatrix}$$

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{a}$$

If, instead of a vector of $\boldsymbol{a}$ coefficients, we have an $m \times k$ matrix of coefficients, such that:

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1k} \\ a_{21} & a_{22} & \cdots & a_{2k} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mk} \end{bmatrix}$$

the linear composite would be:

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{A}'$$

We know that PCA provides linear composites of our original variables of the form above.

Mean

If we want to know the mean of our linear composite, we want to know:

$$E(\boldsymbol{Y}) = E\left(\boldsymbol{X}\boldsymbol{A}'\right) \tag{28}$$
$$= E(\boldsymbol{X})\boldsymbol{A}' \tag{29}$$

If, as our variables usually are, the variables in $\boldsymbol{X}$ are centered, then

$$E(\boldsymbol{Y}) = E(\boldsymbol{X})\boldsymbol{A}' = \boldsymbol{0}\boldsymbol{A}' = \boldsymbol{0}$$

The mean of a linear combination is the linear combination of the means using the same coefficients.

5.5.1 Variance

We know that if we have $aX + bY$ where a and b are constants and X and Y are variables that:

$$var(aX + bY) = a^2 var(X) + b^2 var(Y) + 2(ab)cov(X, Y)$$

This gets complicated when we have many variables and many linear combinations. This is where linear algebra makes our lives a bit simpler. We can rely on the following definition of variance:

$$var(\boldsymbol{X}) = E(\boldsymbol{X}\boldsymbol{X}') - E(\boldsymbol{X})E(\boldsymbol{X}')$$

With the following linear composite:

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{A}'$$

we can try to find the variance:

$$\begin{aligned}
var(\boldsymbol{Y}) &= E(\boldsymbol{Y}\boldsymbol{Y}') - E(\boldsymbol{Y})E(\boldsymbol{Y}') && (30)\\
&= E(\boldsymbol{X}\boldsymbol{A}'(\boldsymbol{X}\boldsymbol{A}')') - E(\boldsymbol{X}\boldsymbol{A}')E((\boldsymbol{X}\boldsymbol{A}')') && (31)\\
&= E(\boldsymbol{X}\boldsymbol{A}'\boldsymbol{A}\boldsymbol{X}') - E(\boldsymbol{X}\boldsymbol{A}')E(\boldsymbol{A}\boldsymbol{X}') && (32)\\
&= \boldsymbol{A}'E(\boldsymbol{X}\boldsymbol{X}')\boldsymbol{A} - \boldsymbol{A}'E(\boldsymbol{X})E(\boldsymbol{X}')\boldsymbol{A} && (33)\\
&= \boldsymbol{A}'\left[E(\boldsymbol{X}\boldsymbol{X}') - E(\boldsymbol{X})E(\boldsymbol{X}')\right]\boldsymbol{A} && (34)\\
&= \boldsymbol{A}'var(X)\boldsymbol{A} && (35)\\
&= \boldsymbol{A}'\boldsymbol{\Sigma}\boldsymbol{A} && (36)
\end{aligned}$$

Think about what this means for our PCA:

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}'$$

The component scores are given by $\boldsymbol{X}\boldsymbol{V}$ where each column of $\boldsymbol{X}$ has mean zero and unit variance.

- $E(XV) = E(\boldsymbol{X}V) = E(\boldsymbol{X})\boldsymbol{V} = \boldsymbol{0}$
- $var(XV) = \boldsymbol{V}'\boldsymbol{\Sigma}\boldsymbol{V}$. To generate some intuition, let's think about the components in a slightly different way. We know that $\boldsymbol{X}\boldsymbol{V} = \boldsymbol{U}\boldsymbol{D}$, so let's think about $var(UD)$ where $\boldsymbol{U}$ is the variable and $\boldsymbol{D}$ is the matrix of coefficients. If we calculated the PCA on the variance-covariance matrix We know that $var(\boldsymbol{U})$ will be diagonal, and in this case (if we decompose the standardized data matrix) it is scaled by $\frac{1}{n-1}$ where n is the number of rows of $\boldsymbol{X}$. The variance of the components, then, is $\frac{1}{n-1}\boldsymbol{D}^2$.v

The scaling by $\frac{1}{n-1}$ can be thought of as follows. If we decompose $\boldsymbol{X} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}'$, then the variance-covariance matrix of $\boldsymbol{X}$ is $\frac{1}{n-1}\boldsymbol{X}'\boldsymbol{X}$. So,

$$\begin{aligned}
\boldsymbol{X} &= \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}'\\
\frac{1}{n-1}\boldsymbol{X}'\boldsymbol{X} &= \frac{1}{n-1}\boldsymbol{V}\boldsymbol{D}\boldsymbol{U}'\boldsymbol{U}\boldsymbol{D}\boldsymbol{V}'\\
&= \frac{1}{n-1}\boldsymbol{V}\boldsymbol{D}^2\boldsymbol{V}'\\
&= \boldsymbol{V}\left[\frac{1}{n-1}\boldsymbol{D}^2\right]\boldsymbol{V}'
\end{aligned}$$

5.6 Reproducing the Correlation Matrix

PCA is explicitly *not* a technique that tries to explain inter-relationships among variables. Rather, it tries to capture variance. While we can reproduce the variance-covariance matrix of the observed variables (with any m-dimensional subspace), the extent to which it is a "good" reproduction of the variance-covariance matrix is incidental to the technique.

```
library(foreign)
banks <- read.dta("https://quantoid.net/files/essex/banks_biplot.dta")
svd.data <- scale(banks[,4:11])
s1 <- svd(svd.data)
banks.var <- var(svd.data)

lam <- (1/(nrow(svd.data)-1))*diag(s1$d^2)
allcomps.var <- s1$v%*%lam %*% t(s1$v)
comp2.var <- s1$v[,1:2]%*%lam[1:2,1:2]%*% t(s1$v[,1:2])
resid.cor <- round(banks.var-comp2.var, 2)

cat("\nResidual Correlation Matrix\n")

##
## Residual Correlation Matrix

resid.cor[,1:4]

##            importspc exportspc enprodkgpc enconskgpc
## importspc       0.28      0.06      -0.24       0.02
## exportspc       0.06      0.04      -0.04      -0.03
## enprodkgpc     -0.24     -0.04       0.27      -0.08
## enconskgpc      0.02     -0.03      -0.08       0.20
## wfpcgind       -0.04     -0.01       0.07      -0.04
## newspaper      -0.05     -0.01       0.03      -0.05
## literate       -0.06      0.02       0.12      -0.05
## gdppcmp        -0.09     -0.04       0.04      -0.04

resid.cor[,5:8]

##            wfpcgind newspaper literate gdppcmp
## importspc     -0.04     -0.05    -0.06   -0.09
## exportspc     -0.01     -0.01     0.02   -0.04
## enprodkgpc     0.07      0.03     0.12    0.04
## enconskgpc    -0.04     -0.05    -0.05   -0.04
## wfpcgind       0.23     -0.08    -0.03   -0.07
## newspaper     -0.08      0.17    -0.08    0.06
## literate      -0.03     -0.08     0.20   -0.01
## gdppcmp       -0.07      0.06    -0.01    0.15
```

Linearity

These methods leverage linear relationships in the data - even more so in CFA, but also in PCA. The extent to which variables are linearly related will be the extent to which underlying structure will be identified.

Assumptions, Caveats, etc...

There are very few assumptions here.

- One "assumption" is that you want to explain variance (i.e., that it makes sense to get the correlation matrix among observations).
- Again, not really an assumption, but that a linear combination of observed variables is "best" or at least suitable for your purposes.
- No assumptions of normality (multivariate or otherwise) because we're not interested in sampling distributions here.

- The principal components coefficients $\boldsymbol{V}$ are *the* values that produce sequentially variance-maximized, orthogonal linear combinations of the observed variables, period. We don't care about a population, and as we will consider after talking about FA, it may not even make sense to talk about population parameters with PCA.
- Along the same line, principal components don't get estimated; they get calculated. Principal components are a well-defined mathematical transformation of observed variables. As such, there is nothing to estimate.
- We haven't talked about rotations yet, but technically PCA solutions shouldn't be rotated because then they aren't PCA solutions any more (i.e., it no longer has the properties of a PCA).