# Chapter 4

# Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a traditional multivariate statistical technique for dimension or variables reduction. It is to find linear combinations of original variables such that the information in the original data is preserved. Before we go into the details of it, we first look at an interesting application of PCA to identify the important components of the term structure of interest rate.

### 4.1 US zero-coupon rate example

The file *"us-rate.csv"* contains US semi-annualized zero-coupon rates, measured in basic points (b.p.), with maturities between 1M to 15Y, monthly data from 1944 to 1992. The following **R** commands will read in the data, apply labels and compute the correlation matrix of these rates.

```
> d<-read.csv("us-rate.csv")  # read in data
> label<-c("1m","3m","6m","9m","12m","18","2y","3y","4y","5y","7y","10y","15y")
> names(d)<-label             # apply labels
> options(digits=2)           # display the number using 2 digits
> cor(d)                      # compute correlation matrix

      1m   3m   6m   9m  12m  18m   2y   3y   4y   5y   7y  10y  15y
1m  1.00 0.99 0.99 0.99 0.98 0.98 0.97 0.96 0.95 0.95 0.94 0.92 0.91
3m  0.99 1.00 1.00 0.99 0.99 0.99 0.98 0.97 0.96 0.96 0.95 0.94 0.92
6m  0.99 1.00 1.00 1.00 1.00 0.99 0.99 0.98 0.97 0.96 0.96 0.94 0.93
9m  0.99 0.99 1.00 1.00 1.00 1.00 0.99 0.99 0.98 0.97 0.96 0.95 0.94
12m 0.98 0.99 1.00 1.00 1.00 1.00 1.00 0.99 0.98 0.98 0.97 0.96 0.94
18m 0.98 0.99 0.99 1.00 1.00 1.00 1.00 1.00 0.99 0.99 0.98 0.97 0.96
2y  0.97 0.98 0.99 0.99 1.00 1.00 1.00 1.00 0.99 0.99 0.98 0.98 0.96
3y  0.96 0.97 0.98 0.99 0.99 1.00 1.00 1.00 1.00 1.00 0.99 0.99 0.98
4y  0.95 0.96 0.97 0.98 0.98 0.99 0.99 1.00 1.00 1.00 1.00 0.99 0.98
5y  0.95 0.96 0.96 0.97 0.98 0.99 0.99 1.00 1.00 1.00 1.00 1.00 0.99
7y  0.94 0.95 0.96 0.96 0.97 0.98 0.98 0.99 1.00 1.00 1.00 1.00 0.99
10y 0.92 0.94 0.94 0.95 0.96 0.97 0.98 0.99 0.99 1.00 1.00 1.00 1.00
15y 0.91 0.92 0.93 0.94 0.94 0.96 0.96 0.98 0.98 0.99 0.99 1.00 1.00
```

First note that these 13 variables are highly correlated. This implies that we can use only few variables (say 2 or 3) to represent this dataset without causing loss of large amount of information. This is exactly the aim of dimension or variable reduction! A

natural question to ask is which variables to use so that they can contain the information in the dataset as much as possible? The answer is not from any of these original variables but a linear combination of the original variables.

Let the original variables be $x_1,...,x_p$ and we are looking for a new variable

$$y_1 = \alpha_1 x_1 + \cdots + \alpha_p x_p = \alpha' x \tag{4.1}$$

such that the variance of $y_1$

$$Var(y_1) = \alpha' S \alpha \tag{4.2}$$

is maximum subject to the constraint:

$$\alpha_1^2 + \cdots + \alpha_p^2 = \alpha'\alpha = 1. \tag{4.3}$$

where S is the sample covariance matrix of x. This is called the first **principal component** (PC) and the $p \times l$ vector $\alpha$ is called the **loadings** of the 1st PC. We can continue to find the second principal component

$$y_2 = \beta_1 x_1 + \cdots + \beta_p x_p = \beta' x \tag{4.4}$$

such that the variance of $y_2$

$$Var(y_2) = \beta' S \beta \tag{4.5}$$

is maximum subject to

$$\beta_1^2 + \cdots + \beta_p^2 = \beta'\beta = 1 \tag{4.6}$$

and $\alpha'\beta = 0$ (This means that α and β is **orthogonal**.). (4.7)

The third PC can be found similarly, and the process can continue up to the *p-th* PC. This gives the basic idea of PCA. This optimization has a nice closed form solution. It turns out that these loadings are the **normalized eigenvectors** of the correlation matrix or the variance-covariance matrix of *x* and the variance of *y* is the corresponding **eigenvalue**. Details of the theory of eigenvealues and eigenvectors will be given in the appendix; also refer to Chapter 1.

The reason for maximizing the variance of the transformed variable y is that we want to preserve as much information in x as possible. Conceptually, variation represents information. We want the variance of the transformed variable y as large as possible, so that the original information in x is preserved. The eigenvalue, which is equal to the variance of y, indicates the amount of information being retained in the PC.

### 4.2 PCA using R

PCA is implemented in R's built-in function `princomp()`. First let us illustrate it by using the *"us-rate.csv"*.

```
> pca<-princomp(d,cor=T)  # perform PCA using correlation matrix
                          # and save the result to the object pca.
> pca$loadings[,1:6]      # display the loadings of the first six PCAs

     Comp.1   Comp.2   Comp.3   Comp.4   Comp.5   Comp.6
1m  -0.2732  0.40434  0.59756  0.56592 -0.27195  0.06041
3m  -0.2758  0.34462  0.28225 -0.29408  0.61221 -0.34923
6m  -0.2770  0.30272 -0.02848 -0.36767  0.16994  0.38335
9m  -0.2783  0.23612 -0.17710 -0.23317 -0.21616  0.25111
12m -0.2786  0.20227 -0.25195 -0.16505 -0.41027  0.18186
18  -0.2798  0.09148 -0.27652  0.03453 -0.20085 -0.31179
2y  -0.2799  0.03527 -0.28849  0.13555 -0.09394 -0.56147
3y  -0.2798 -0.09978 -0.22264  0.22118  0.13877 -0.10916
4y  -0.2791 -0.16815 -0.18806  0.26537  0.25584  0.12151
5y  -0.2785 -0.22976 -0.08357  0.20610  0.21373  0.20016
7y  -0.2772 -0.30150  0.03910  0.13771  0.16234  0.29454
10y -0.2754 -0.37538  0.21868 -0.10556 -0.04935  0.08244
15y -0.2729 -0.44384  0.40821 -0.40615 -0.31461 -0.24199
```

From the output, the 1st PC is $y_1 = -0.2732x_1 - \ldots - 0.2729x_{13}$, the 2nd PC is $y_2 = 0.4043x_1 + \ldots - 0.4438x_{13}$, and so on. Using the previous notations, $\alpha = (-0.2732, \ldots, -0.2729)'$ and $\beta = (0.4043, \ldots, -0.4438)'$. It is easy to check that $\alpha'\alpha = \beta'\beta = 1$(unit length) and $\alpha'\beta = 0$(orthogonal). Actually any of these PC are of unit length and any pairs of PC are orthogonal.

```
> pc1<-pca$loadings[,1]   # save the loading of PC1
> pc2<-pca$loadings[,2]   # save the loading of PC2
> pc1 %*% pc1             # compute α'α (should be 1, unit length)
[1] 1
> pc2 %*% pc2             # compute β'β (should be 1)
[1] 1
> pc1 %*% pc2             # compute α'β (should be 0, orthogonal)
[1] 1.388e-17
```

We can display all these 13 PCs. However, the purpose of PCA is variable reduction. We want to represent this "us-rate" dataset by using only few variables instead of all 13 variables. How many PC should we use? Before we answering this question, first let us look at the variance or standard deviation (s.d.) of these PC.

```
> s<-pca$sdev     # save the s.d. of all PC to s
> s               # display s
  Comp.1   Comp.2   Comp.3   Comp.4   Comp.5   Comp.6   Comp.7   Comp.8
3.567055 0.487587 0.157683 0.086441 0.053838 0.038195 0.031348 0.026454
  Comp.9  Comp.10  Comp.11  Comp.12  Comp.13
0.000880 0.000854 0.000831 0.000798 0.000768

> round(s^2,4)    # display variance
 Comp.1  Comp.2  Comp.3  Comp.4  Comp.5  Comp.6  Comp.7  Comp.8  Comp.9 Comp.10
12.7239  0.2377  0.0249  0.0075  0.0029  0.0015  0.0010  0.0007  0.0000  0.0000
Comp.11 Comp.12 Comp.13
 0.0000  0.0000  0.0000

> t<-sum(s^2)     # compute total variance (should equals 13)
> round(s^2/t,4) # proportion of variance explained by each PC
 Comp.1  Comp.2  Comp.3  Comp.4  Comp.5  Comp.6  Comp.7  Comp.8  Comp.9 Comp.10
 0.9788  0.0183  0.0019  0.0006  0.0002  0.0001  0.0001  0.0001  0.0000  0.0000
Comp.11 Comp.12 Comp.13
 0.0000  0.0000  0.0000

> cumsum(s^2/t)  # cumulative sum of proportion of variance
 [1] 0.979 0.997 0.999 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
[13] 1.000
```

From the above output, we know that the s.d. and variance of the 1st PC is 3.567 and 12.7239 respectively and so on. The 1st PC explained 12.7239/13 = 97.88% of the total variance. The 2nd PC explained 1.83%, the 3rd PC explained 0.19%, and so on. If we only use the 1st PC, this PC already preserved almost all (97.88%) the information. If we use the first two PCs, we can preserve 99.7% of the information and so on.

In Chapter 1, we see that the total variation is equal to the **trace** of *S*, denoted by *tr(s)*, i.e., the sum of all the diagonal elements in *S*. Note that in our example, we use the correlation matrix instead of covariance matrix. Therefore all the diagonal elements is equal to 1 and the sum is equal to p=13.

Note that $y_i$'s are orthogonal, this is because

$Cov(y) = Cov(H'x) = H'Cov(x)H = H'H\Lambda H'H = \Lambda$, which is a diagonal matrix.

In general, if $x_i = \sum_{j=1}^{p} e_{ji} y_j$, we have:

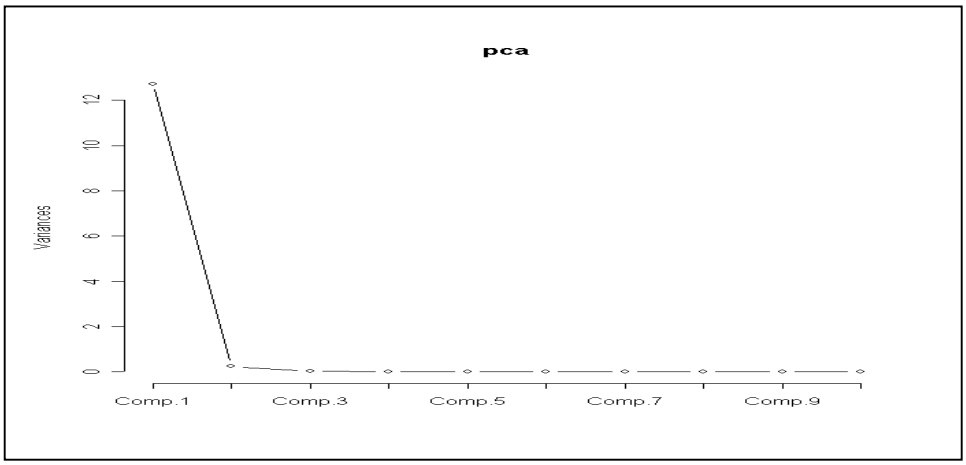
$$Var(x_i - \sum_{j=1}^{m} e_{ji} y_j) = Var(\sum_{j=m+1}^{p} e_{ji} y_j) = \sum_{j=m+1}^{p} e_{ji}^2 Var(y_j) = \sum_{j=m+1}^{p} e_{ji}^2 \lambda_i \le \sum_{j=m+1}^{p} \lambda_i ,$$

which is 0.01% of the total variance in our case. Here the last inequality follows as $e_{ji}^2 \le 1$.

Even more, define $\Delta_i = x_i - \sum_{j=1}^{m} e_{ji} y_j$, then $Var(\Delta_i) = \sum_{j=m+1}^{p} e_{ji}^2 \lambda_j$. Therefore, the total $L^2$-error should be $\sum_i Var(\Delta_i) = \sum_{j=m+1}^{p} \left( \sum_{i=1}^{p} e_{ji}^2 \right) \lambda_j = \sum_{j=m+1}^{p} \lambda_j$, which is 0.01% of the total variance in our example.

A plot of variance (called **Scree plot**, "scree" means large loose broken "gems") can help us determine the suitable number of PC used. This plot represents the information retained in each PC graphically.

```
> screeplot(pca,type="lines")
```




For most applications, we can look at the scree plot to determine the number of PC to be used. In this example, no doubt that we can just used the 1st PC to represent all the 13 variables. However, the first three PCs have interesting interpretations so that we may want to use the first three PCs to represent the term structure of the US zero rate.

### 4.3 Interpretation of PCs

Recall that the 1st PC is $y_1 = -0.2732x_1 - \ldots - 0.2729x_{13}$. The loadings are almost constant at a -0.27 level for each maturity time. In fact, these loadings are not unique. They are unique only up to a positive or negative sign. That is, if we define a new 1st PC as $y_1 = 0.2732x_1 + \ldots + 0.2729x_{13}$, this PC has the same variance as the old one. This PC represents a **parallel shift** (caused by a "level" shock for example) of the

yield curve. It also explained a majority of variance exists in the dataset. In other word, the US zero-coupon rate exhibits a parallel shift of term structure of the interest rate of different maturity. We can find more information about the parallel shift of the yield curve in the internet:

*A shift in economic conditions in which the change in the interest rate on all maturities is the same number of basis points. In other words, if the three month T-bill increases 100 basis points (one %), then the 6-month, 1-year, 5-year, 10-year, 20-year, and 30-year rates all increase by 100 basis points as well. Related: Non-parallel shift in the yield curve.*

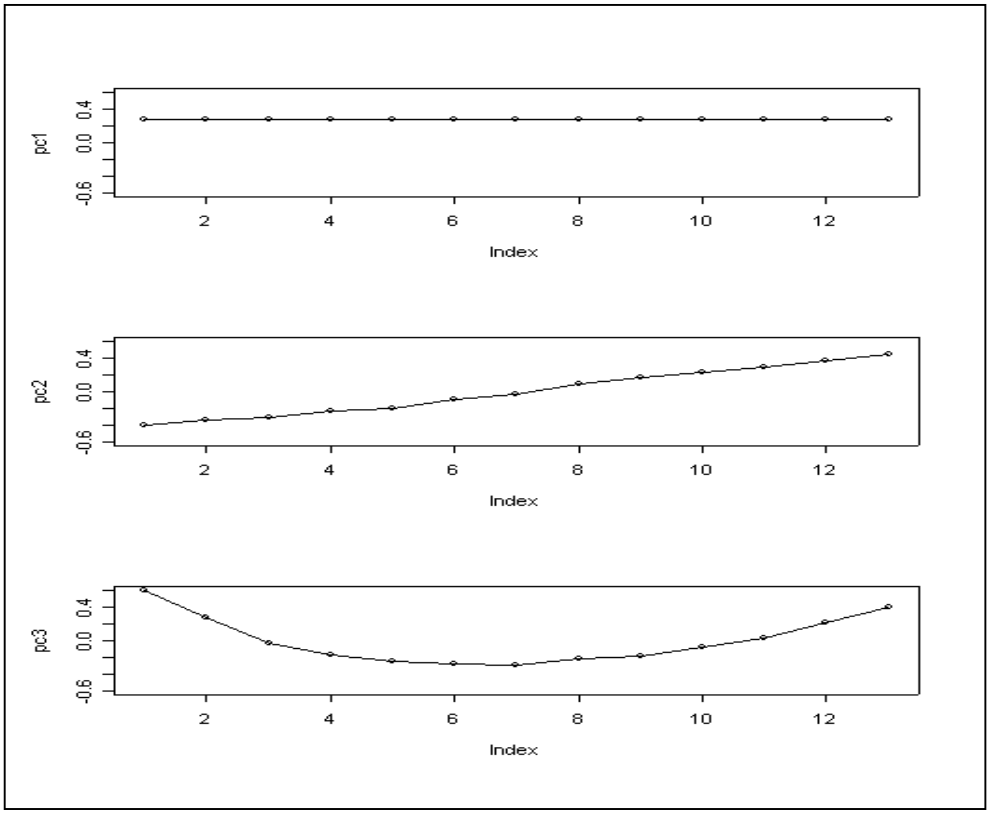
Source: http://www.nasdaq.com/investing/glossary/p/parallel-shift-in-the-yield-curve#ixzz3163N0aF6

The 2nd PC is $y_2 = 0.4043x_1 + \ldots - 0.4438x_{13}$ where the loadings are decreasing with maturity time. Again, the loadings are not unique. In particular the new 2nd PC $y_2 = -0.4043x_1 - \ldots + 0.4438x_{13}$ seems more reasonable since it agrees with the **liquidity preference theory**. This theory states that investors prefer to preserve their liquidity and invest funds for short period of time. This component is termed as the **tilt** component of the yield curve.

The 3rd PC is $y_3 = 0.5976x_1 + \ldots + 0.4082x_{13}$. The loadings decrease with maturity up to certain point and starts increasing again with maturity. This can be interpreted as the **curvature** of the yield curve; this effect is mainly caused by the higher in demand for very short-term and very long-term bonds.

Let us plot the first three PC in R.

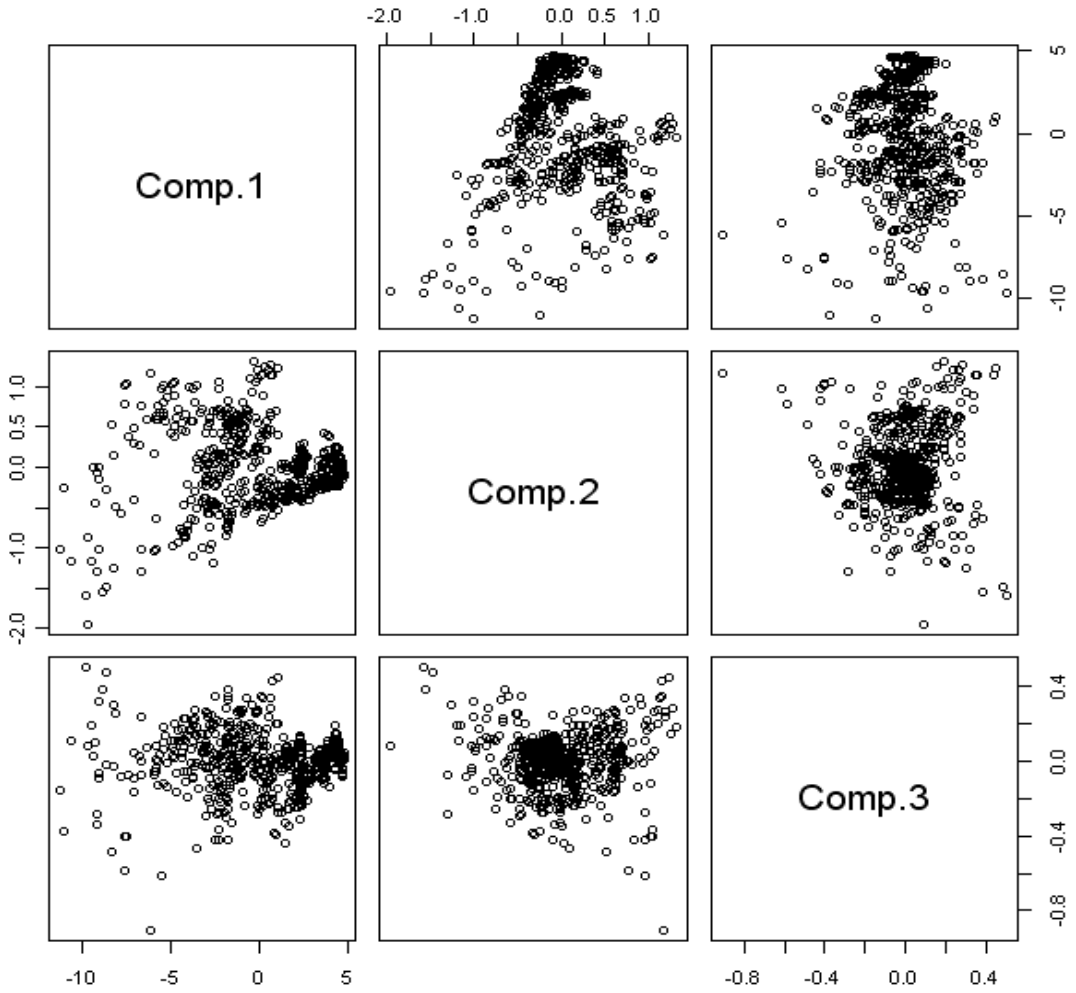
```
pc1<- -pca$loadings[,1]                 # PC1 parallel shift
pc2<- -pca$loadings[,2]                 # PC2 tilt
pc3<- pca$loadings[,3]                  # PC3 curvature
par(mfrow=c(3,1))                       # multiframe for plotting
plot(pc1,ylim=c(-0.6,0.6),type="o")     # plot with y-axis from -0.6 to 0.6
plot(pc2,ylim=c(-0.6,0.6),type="o")     # see help(plot) for details
plot(pc2,ylim=c(-0.6,0.6),type="o")
```




Identifying these components is important in understanding the term structure of the interest rate. Many investment instruments are very sensitive to the movement of interest rate, such as stocks, bonds, forward contract, real estates and swap. These components are very useful in valuating these instruments.

The original data are transformed into $y_1, y_2$ and $y_3$ according to the loadings of PC1, PC2 and PC3. These are called the principal component scores of PC1, PC2 and PC3. The geometrical interpretation of PCA may be illustrated by the scatterplot matrix of these scores.

```
score<-pca$scores[,1:3]          # save the scores of PC1, PC2 and PC3
pairs(score)                     # scatterplot of scores
```




From these plots, each principal component is the direction that the corresponding sample variance of transformed variables $y_1, y_2$ and $y_3$ is maximized. Note that the loadings of *i-th* PC $y_i = \beta_{i1}x_1 + \ldots + \beta_{ip}x_p = \beta_i' x$ are the *i-th* eigenvectors of *S*. Let us form a $p \times p$ matrix $H = (h_{ij}) = [\beta_1 \vdots \beta_2 \vdots, \cdots, \vdots \beta_p]$, where the *i-th* column $\beta_i$ is the *i-th* eigenvector of *S*. This matrix H is orthogonal, i.e., $H'H = HH' = I_p$. Furthermore, $y = H'x$, i.e., *y* is $p \times l$ vector with the first element is the PC1 score of x, and so on. It follows that the original *x* can be expressed in terms of *y* as $x = H\ y$, i.e., $x_i = h_{i1}y_1 + \ldots + h_{ip}y_p$. In practice, we may only use the first *m* PCs to approximate x, $x_i \approx h_{i1}y_1 + \ldots + h_{im}y_m$. This provides an interesting and useful application of PCA to approximate the Value at Risk (**VaR**) of a portfolio.

**4.4 Application of PCA to VaR**

As we have seen in Chapter 3, the standard deviation of the loss in a portfolio is needed to compute the VaR of that portfolio. Let the change in the portfolio be $\Delta P = w'x = w_1 x_1 + \ldots + w_p x_p$ and we can approximate by $\Delta P$ using the first *m* PCs of *x* as follow:

$$\begin{aligned}\Delta P = w'x &\approx w_1(h_{11}y_1 + \ldots + h_{1m}y_m) + \ldots + w_p(h_{p1}y_1 + \ldots + h_{pm}y_m) \\ &= (w_1h_{11} + \ldots + w_ph_{p1})y_1 + \ldots + (w_ph_{p1} + \ldots + w_ph_{pm})y_m = \delta_1 y_1 + \ldots + \delta_m y_m\end{aligned}$$

From chapter 3, we know that the *N*-day 99% VaR of *P* is $z_{0.99}\sqrt{N}\, sd(\Delta P)$. In order to compute the VaR of *P*, we need to compute $Var(\Delta P) = w'\Sigma w$. This can be very complicated and involves all the covariance of *x*. However, if we use the first *m* PC to approximate the variance of $\Delta P$,

$$Var(\Delta P) \approx \delta_1^2 \,\mathrm{var}(y_1) + \ldots + \delta_m^2 \,\mathrm{var}(y_m) = \delta_1^2\lambda_1 + \ldots + \delta_m^2\lambda_m.$$

The expression becomes much simpler due to the fact that all PC's are orthogonal to each other and maintain a large portion of variations in *x*.

To illustrate this, let us continue with our example on US semi-annualized zero-coupon rates. Recall that the variance of the 1st PC is 12.7239 and the s.d. is 3.567. Suppose that there is a 3.567 basis point (b.p. i.e. (3.567/365)%=0.01% per annum) increase in the 1st PC, after changing the sign, (i.e. parallel shift in the yield curve). This change will contribute, since y=H'x and x=Hy, (3.567)(0.2732)=0.9744 b.p. increase in the 1m interest rate, (3.567)(0.2758)=0.9838 b.p. increase in the 3m interest rate, and so on. Similar interpretation can be applied to the 2nd and 3rd PC. With this interpretation in mind, we can apply PCA to calculate VaR of portfolio of instruments depending on interest rate. Suppose that we have a portfolio with the exposures to 1 basis point increase in interest rate as shown.

| 1 b.p. increase in | 1y | 2y | 3y | 4y | 5y |
|---|---|---|---|---|---|
| change in million $ (shares of bond) | +10 | +4 | -8 | -7 | +2 |

Now let us use the 1st PC to calculate the Value at Risk (VaR) of this portfolio. Our exposure to the 1st PC (measured in million $ per b.p.) is, see the entries in the first

column for 1st PC in the first table on P.3, by just consider $x_i=h_{i1}\ y_1$,
(10)(0.2786) + (4)(0.2799) + (-8)(0.2798) + (-7)(0.2791) + (2)(0.2785) = 0.273.
That is, change in 1 b.p. will results in 0.273m increase in the value of the portfolio. Therefore s.d. of $\Delta P$ is s $= \sqrt{Var(0.273y_1)} = (0.273)(3.567) = 0.973$m and the 10-day 99% VaR of this portfolio is $\sqrt{10}\ z_{0.99}\ s = (\sqrt{10})(2.33)(0.973) = 7.157$.

Similarly the exposures to 2nd (the negative counterpart) and 3rd PC are respectively,
(10)(-0.2023) + (4)(-0.0353) + (-8)(0.0998) + (-7)(0.1682) + (2)(0.2298) = -3.681, and
(10)(-0.2512) + (4)(-0.2885) + (-8)(-0.2226) + (-7)(-0.1181) + (2)(-0.0856) = -0.745.

If we only use the 1st and 2nd PC to estimate the interest rate movement, the s.d. of the change in portfolio is

$$s = \sqrt{Var(0.273y_1) + Var((-3.681)y_2)} = \sqrt{(0.273)^2(3.567)^2 + (-3.681)^2(0.4876)^2} = 2.042m$$

.

The 10-day 99% VaR is $\sqrt{10}\ z_{0.99}\ s = (\sqrt{10})(2.33)(2.042) = 15.019m$.
Note that the PC's are orthogonal or uncorrelated, we do not need to include the covariance terms in calculating the variance of $\Delta P$. Similarly, if we use all the first three PCs, the s.d. of $\Delta P$ is

$$s = \sqrt{(0.273)^2(3.567)^2 + (-3.681)^2(0.4876)^2 + (-0.745)^2(0.1577)^2} = 2.045m.$$

and the 10-day 99% VaR is $\sqrt{10}\ z_{0.99}\ s = (\sqrt{10})(2.33)(2.045) = 15.044m$.

In this example, we know that although the 1st PC alone explained almost 98% of the total variance in the dataset, using only the 1st PC will underestimate the VaR, as the component change for each interest rate product is quite different from one another.

**4.5 PCA using EXCEL**

EXCEL does not have built-in function for PCA or computing eigenvalues and eigenvector. However, EXCEL's built-in function for matrix multiplication *mmult()* and *solver* function can help to perform PCA.

1. First we need to compute the correlation matrix of the interest rate. Beside using the built-in *correl()* function and *offset* as in Chapter 1, we can use the built-in procedure in Tools -> Data analysis menu. Let us illustrate this by choosing this procedure and save the output in A1:N14. Note that the output is a lower

triangular matrix. However, we need a full 13x13 matrix to work with at later stage. Therefore we need to transform it into full mode.

2. Highlight the matrix B2:N14 and copy. Place the cursor to B16 and paste special with the option transpose. We have an upper triangular matrix. Delete the diagonal elements in this matrix.
3. In cell B31, enter =B2+B16. Copy the formula to B31:N43. Now we have a full mode correlation matrix.
4. In cell B45:N45, enter some initial values for PC1, say all equal to 1.
5. In cell B46, enter the formula

   =SUMPRODUCT(MMULT(B45:N45,$B$31:$N$43),B45:N45)

   This will compute the variance of PC1.
6. In cell E46, enter the formula =SUMSQ(B45:N45) . This will compute the sum of square of the loadings in PC1.
7. Using *solver* to maximize the cell B46 with B45:N45 as the variable cells and subject to the normalized constraint E46=1.
8. For PC2, we can use similar steps in 2 to 7 with B48:N48 as PC2, B49 as the variance of PC2 and E49 as the normalized constraint.
9. Note that an additional orthogonal constraint is needed. In cell H49, enter the formula =SUMPRODUCT(B45:N45,B48:N48)

   Set this cell to 0 in the *solver's* constraint option.
10. For PC3, we need two orthogonal constraints in H52 and K52.

**4.6 Conclusion**

We have seen the application of PCA in the US zero-coupon rate example. Actually, we can apply PCA to other interest rate such as swap rates, yields on bonds … etc. They all should exhibits similar patterns with three components: parallel shift, tilt and curvature with the parallel shift explain most of the variance in the dataset. One final question is whether we should use correlation matrix or variance-covariance matrix in PCA. As a general rule of thumb, if the units of measurement of the variables in x are very different, we should use correlation matrix instead of variance-covariance matrix. It is to avoid some very large variance exist in certain variables while some other variables are extremely small. On the other hand, if the units of measurement in *x* are the same or similar as in our US zero-coupon rates example, we can either use variance- covariance matrix or correlation matrix for PCA. In general, we can easily verify that the PCA using variance-covariance matrix in our example has similar components.

**Appendix:**

**1.Mathematical details**

The basic idea in PCA is to find a linear combination of the original variables

$$y_1 = \alpha_1 x_1 + \cdots + \alpha_p x_p = \alpha' x$$

so that $Var(y_1) = \alpha' Var(x)\alpha = \alpha' S \alpha$ is maximum subject to $\alpha'\alpha = 1$. The solution of this optimization problem is related to a well-known and very useful result in matrix algebra called the **spectral decomposition** of matrix. Let us review again the definition and some of the important results.

**Definition**:

$|A - \lambda I_P|$ is a polynomial of degree p in $\lambda$. The **eigenvalues** (or **latent roots**) of $A$, denoted by $\lambda_1, \ldots, \lambda_p$ are the roots of the equation $|A - \lambda I_P| = 0$. The nonzero vector $h_i$ such that $(A - \lambda_i I_P)h_i = 0$ *(or* $Ah_i = \lambda_i h_i$ *)* is called the **eigenvector** of A corresponding to $\lambda_i$. If the vector $h_i$ has a unit length (i.e., *h'h=1*), then it is called the **normalized eigenvector** of $A$.

There are some important properties of eigenvalues and eigenvectors.

1. If $A$ is real symmetric matrix then its latent roots are all real.
2. The latent roots of $A$ and $A'$ are the same.
3. If $A$ and $B$ are $p \times p$, $A$ is nonsingular then latent roots of $AB$ and $BA$ are equal.
4. If $\lambda_1, \ldots, \lambda_p$ are the latent roots of a nonsingular matrix $A$, then $\lambda_1^{-1}, \ldots, \lambda_p^{-1}$ are the latent roots of $A^{-1}$.
5. If $A$ is real symmetric matrix and $\lambda_i$ and $\lambda_j$ are two distinct latent roots of $A$, then the corresponding eigenvectors $h_i$ and $h_j$ are orthogonal.
   Proof. By definition, $Ah_i = \lambda_i h_i$ and $Ah_j = \lambda_j h_j$ ,
   => $h_j' Ah_i = \lambda_i h_j' h_i$ and $h_i' Ah_j = \lambda_j h_i' h_j$
   => $(\lambda_i - \lambda_j) h_j' h_i = 0$ => $h_i$ and $h_j$ are orthogonal.
6. (Spectral decomposition of A) Let $H$ be a $p \times p$ matrix whose column is eigenvectors $h_1, \ldots, h_p$ of $A$,
   i.e. $H = (h_1, \ldots h_p)$ then $H'AH = D = diag(\lambda_1, \ldots, \lambda_p)$ or $A = HDH'$
   Proof. Write $AH = (Ah_1, \ldots, Ah_p) = (\lambda_1 h_1, \ldots, \lambda_p h_p) = HD \Rightarrow H'AH = D$.

The following lemma shows that the maximum value of the variance of $y_1$ is the

largest eigenvalue of *S* and α is the corresponding normalized eigenvector of *S*. Also, the 2nd PC is the linear combination

$$y_2 = \beta_1 x_1 + \cdots + \beta_p x_p = \beta' x$$

so that $Var(y_2) = \beta' S\beta$ is maximum subject to $\beta'\beta = 1$ and $\alpha'\beta = 0$. It turns out that this maximum value of the variance of $y_2$ is the second largest eigenvalue of *S* and β is the corresponding normalized eigenvector of *S*.

**Lemma**. Suppose that the $p \times p$ positive definite matrix B has the ordered eigenvalues $\lambda_1 \geq \cdots \geq \lambda_p > 0$ and the corresponding unit (or normalized) eigenvectors are $\alpha_1, \ldots, \alpha_p$. Then

1. $\max_{l \neq 0} \frac{l'Bl}{l'l} = \lambda_1$ and the maximum is attained at $l = \alpha_1$,
2. $\min_{l \neq 0} \frac{l'Bl}{l'l} = \lambda_p$ and the minimum is attained at $l = \alpha_p$. Futhermore,
3. $\max_{l \perp \alpha_1, \ldots, \alpha_k} \frac{l'Bl}{l'l} = \lambda_{k+1}$ and the maximum is attained at $l = \alpha_{k+1} \quad for \quad k = 1,2,\ldots, p-1$

**Proof**. Let $B = P\Lambda P'$ be the spectral decomposition of B, i.e., $P = (\alpha_1, \ldots, \alpha_p)$,

$\Lambda = diag(\lambda_1, \ldots, \lambda_p)$, $B^{1/2} = P\Lambda^{1/2}P'$.

Let *u* and *l* be $p \times l$ vector such that $u = P'l$,

$$\frac{l'Bl}{l'l} = \frac{l'B^{1/2}B^{1/2}l}{l'PP'l} = \frac{l'P\Lambda^{1/2}P'P\Lambda^{1/2}P'l}{u'u} = \frac{u'\Lambda u}{u'u} = \frac{\sum_{i=1}^{p}\lambda_i u_i^2}{\sum_{i=1}^{p} u_i^2} \leq \lambda_1 \frac{\sum u_l^2}{\sum u_i^2} = \lambda_1.$$

When $l = \alpha_1$, $y = P'\alpha_1 = (1,0,\ldots 0)'$ and $\frac{\alpha_1' B \alpha_1}{\alpha_1'\alpha_1} = \frac{y'\Lambda y}{y'y} = \lambda_1$.

Similarly, we can prove 2.
For 3, recall that $u = P'l$,

$$l = Pu = u_1\alpha_1 + \ldots + u_p\alpha_p. \quad l \perp \alpha_1, \ldots, \alpha_k \Rightarrow 0 = \alpha_i' l = u_1\alpha_i'\alpha_1 + \ldots + u_p\alpha_i'\alpha_p = u_i, \quad i \leq k.$$

Therefore, $\frac{l'Bl}{l'l} = \frac{\sum_{i=k+1}^{p}\lambda_i u_i^2}{\sum_{i=k+1}^{p} u_i^2} \leq \lambda_{k+1}$.

## 2. Recommender system

A recommender system that seeks to predict the "rating" or "preference" that a user would give to an item. It has become increasingly popular in recent years, and are utilized in a variety of areas including movies, music, news, books, research articles, search queries, social tags, and products in general. There are also recommender systems for restaurants, garments, financial services, life insurance, romantic partners (online dating), and Twitter pages.


Suppose we found a startup that provides online movie viewing. Our company has attracted $i = 1, 2, ..., m$ subscribers, and has acquired $j = 1, 2, ..., n$ movies. Each subscriber is invited to rate a movie from $r = 1, 2, .., 5$, after viewing the movie, These ratings are stored in a $m \times n$ sparse matrix $R(i, j) = r_{i,j}$, containing records of each subscriber's ratings on all movies in our database.

Note that the rating remain at $r = 0$ for movies that the subscriber hasn't watched or rated. Based on this existing data, we want to build a recommender system to learn about the "taste of the subscriber, so that we can make predictions on a particular subscriber's ratings of these movies that he/she has not watched.

In a method known as collaborative filtering, each movie is evaluated against certain general features, labelled as $k = 1, 2, .., p$, that describe movies. Typically, only a few features are needed and $p \le n$ (provided our company acquires a lot of movies).

These "hidden" features, which arise in our implementation algorithm, depend entirely on the data, and are not necessarily related to typical classifications such as comedy, tragedy, actions etc. More concretely, for a particular movie $j'$, we are to evaluate a series of scores $y_{j',k}$ against all hidden features $k = 1,2,..,p$ that best describe the movie. The collection of all such scores is to be stored in the $n \times p$ matrix $Y(j,k)$, which we are to calculate.

On the other side, these hidden features can also be used to describe the subscribers' preferences on movies. For a subscriber $i'$, we want to evaluate a series of scores $x_{i',k}$ against all hidden features $k = 1,2,..,p$ that best describe the subscriber's taste on movies. These scores are to be collectively stored in the $m \times p$ matrix $X(i,k)$, which are also to be found.

Now our task is to find the matrices $X(i,k)$ and $Y(j,k)$, which amount to $(m+n) \times p$ unknown parameters. The product matrix, $X \cdot Y^T = \sum_{k=1:p} X(i,k) \cdot Y(j,k)^T$ of $m \times n$ dimensions can be compared to the subscriber's rating $R(i,j)$. While $R(i,j)$ is sparse (since each subscriber only had viewed or rated a small amount of the entire movie database), our algorithm is to minimize the difference between the known (non-zero) ratings in $R(i,j)$ and the corresponding rating predicted on $\sum_{k=1:p} X(i,k) \cdot Y(j,k)^T$. In other words, we are to minimize the error function, defined as:

$$J(x_{i,k}, y_{j,k}) = \sum_{\substack{i=1:m, j=1:n \\ R(i,j) \neq 0}} [\sum_{k=1:p} X(i,k) \cdot Y(j,k)^T - R(i,j)]^2 \quad ,$$

for $i = 1,2,..,m$, $j = 1,2,..,n$, $k = 1,2,..,p$, and $p \le m,n$

Optimization scheme are used to find the set of parameters $x_{i,k}$ and $y_{j,k}$ for all $i = 1,2,..,m$ , $j = 1,2,..,n$ and $k = 1,2,...,p$ that gives the global minimum of this error function.

Upon this optimization procedures that minimize $J$ , both the subscribers' tastes on

movies $X(i,k)$, and the nature of movies $Y(j,k)$ can be estimated. The $m \times n$ matrix $X \cdot Y^T \big|_{\min(J)}$ then stores the predicted ratings of all movies to all subscribers. For instance, the $i$'-th row of $X \cdot Y^T \big|_{\min(J)}$ is a row vector that lists the predicted ratings of all movies $j = 1, 2, ..., n$ to that $i$'-th subscriber. These predicted ratings of $X \cdot Y^T \big|_{\min(J)}$ should be nonzero on those $(i, j)$ entries with unknown ratings, where $r_{i,j} = 0$, while on those $(i, j)$ entries where $r_{i,j} \neq 0$, should roughly be equaled to the known ratings of $R(i, j)$. By sorting these ratings in descending order, our system should be able to show a subscriber our top recommendations that best match to his/her taste.

**Reference:**


1. Chapter 8 of Applied Multivariate Statistical Analysis, 5th ed., Richard Johnson and Dean Wichern, Prentice Hall.
2. Chapters 4 and 10 of Risk Management and Financial Institutions, by John Hull, Pearson Education.