Food Price Data - PCA in R

Isabel M. Rodrigues

12/12/2022

Principal Component Analysis

Introduction

- **Principal Component Analysis (PCA)**: Initially proposed by **Pearson** (1901), with a different name. In 1933, **Hotelling** independently, got (and named) the same results.
- The main concern of PCA is to explain the **associations** among a set of variables through **linear combinations** of these variables.

  Aims:

  (i) Data reduction

  (ii) Interpretation
- Frequently used as an **intermediate step** in larger investigations.

Let $\mathbf{X} = (X_1, \ldots, X_p)^{\mathrm{T}}$ be a random vector describing a given population, with expected vector $\boldsymbol{\mu} = \mathbf{0}$ and covariance matrix $\boldsymbol{\Sigma}$.

- **Principal Components (PC)** (see Figure 1) are:
  - **Algebraically**, are non-correlated linear combinations of the original variables
  - **Geometrically**, correspond to a new coordination system of axes (change of base) to represent the data
- No distributional hypotheses are required

The first PC, $Y_1 = \boldsymbol{\gamma}_1^{\mathrm{T}}\mathbf{X} = \gamma_{11}X_1 + \ldots + \gamma_{1p}X_p$, is the linear combination of $\mathbf{X}$, with maximum variability, i.e.

$$\boldsymbol{\gamma}_1 \quad = arg\, max_{\mathbf{a}} \mathrm{Var}(\mathbf{a}^{\mathrm{T}}\mathbf{X}) = arg\, max_{\mathbf{a}}(\mathbf{a}^{\mathrm{T}}\boldsymbol{\Sigma}\mathbf{a})$$

- **Problem**: For any $c > 1$, $\mathrm{Var}(c\mathbf{a}^{\mathrm{T}}\mathbf{X}) = c^2\mathrm{Var}(\mathbf{a}^{\mathrm{T}}\mathbf{X}) > \mathrm{Var}(\mathbf{a}^{\mathrm{T}}\mathbf{X})$ i.e. $\mathrm{Var}(cY_1) > \mathrm{Var}(Y_1)$
- **Solution**: Impose the restriction: $||\mathbf{a}|| = 1 \Leftrightarrow \mathbf{a}^{\mathrm{T}}\mathbf{a} = 1$
- **First PC,** Y_1: is the linear combination of $\mathbf{X}$, $\mathbf{a}^{\mathrm{T}}\mathbf{X}$, with maximum variance, such that $||\mathbf{a}|| = 1$
- **Second PC,** Y_2: is the linear combination of $\mathbf{X}$, $\mathbf{a}^{\mathrm{T}}\mathbf{X}$, with maximum variance, such that:

  (i) $||\mathbf{a}|| = 1$

  (ii) $\mathrm{Cov}(\mathbf{a}^{\mathrm{T}}\mathbf{X}, \boldsymbol{\gamma}_1^{\mathrm{T}}\mathbf{X}) = 0 \Leftrightarrow \mathbf{a}^{\mathrm{T}}\boldsymbol{\Sigma}\boldsymbol{\gamma}_1 = 0$ $(\Leftrightarrow \mathbf{a}^{\mathrm{T}}\boldsymbol{\gamma}_1 = 0)$
- ***i*-th PC,** Y_i: is the linear combination of $\mathbf{X}$, $\mathbf{a}^{\mathrm{T}}\mathbf{X}$, with maximum variance, such that:

  (i) $||\mathbf{a}|| = 1$.

  (ii) $\mathrm{Cov}(\mathbf{a}^{\mathrm{T}}\mathbf{X}, \boldsymbol{\gamma}_k^{\mathrm{T}}\mathbf{X}) = 0, \Leftrightarrow \mathbf{a}^{\mathrm{T}}\boldsymbol{\Sigma}\boldsymbol{\gamma}_k = 0$ $(\Leftrightarrow \mathbf{a}^{\mathrm{T}}\boldsymbol{\gamma}_k = 0)$, $k = 1, \ldots, i-1$

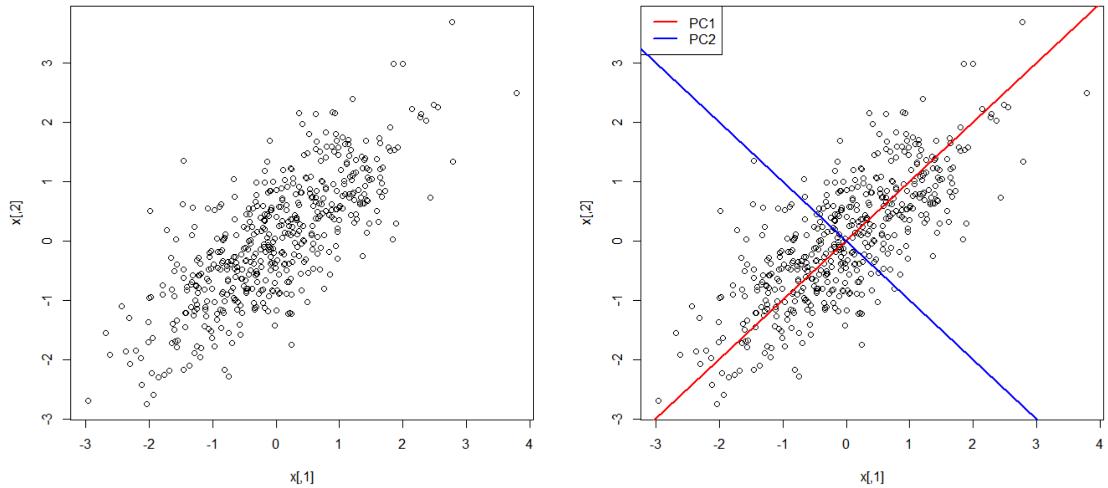


Figure 1: Interesting directions to project the data.

Population Principal Components

Theorem: Let $\mathbf{\Sigma}$ be a covariance matrix associated with the random vector $\mathbf{X}$ with expected value $\boldsymbol{\mu} = \mathbf{0}$. Let $(\lambda_1, \boldsymbol{\gamma}_1)$, ..., $(\lambda_p, \boldsymbol{\gamma}_p)$ be the eigenvalues/eigenvectors pairs of $\mathbf{\Sigma}$, where $\lambda_1 \geq \lambda_2 \geq \ldots \lambda_p \geq 0$. The i-th principal component is given by:

$$Y_i = \boldsymbol{\gamma}_i^{\mathrm{T}} \mathbf{X} = \gamma_{i1} X_1 + \ldots + \gamma_{ip} X_p$$

with

$$\mathrm{Var}(Y_i) = \boldsymbol{\gamma}_i^{\mathrm{T}} \mathbf{\Sigma} \boldsymbol{\gamma}_i = \lambda_i$$
$$\mathrm{Cov}(Y_i, Y_j) = 0, \quad \forall i \neq j$$

γ_i are called **loadings**. If some λ_i are equal, then $\boldsymbol{\gamma}_i$ and Y_i are not unique.

Comment: If $\mathrm{E}(\mathbf{X}) = \boldsymbol{\mu} \neq \mathbf{0}$, the PCs are defined as: $Y_i = \boldsymbol{\gamma}^{\mathrm{T}}(\mathbf{X} - \boldsymbol{\mu})$.

Properties:

1. $\mathrm{E}(Y_i) = \boldsymbol{\gamma}_i^{\mathrm{T}} \mathrm{E}(\mathbf{X}) = \boldsymbol{\gamma}_i^{\mathrm{T}} \boldsymbol{\mu}$
2. $\mathrm{Var}(Y_i) = \boldsymbol{\gamma}_i^{\mathrm{T}} \mathbf{\Sigma} \boldsymbol{\gamma}_i = \lambda_i$
3. $\mathrm{Cov}(Y_i, Y_j) = 0, \quad \forall i \neq j$
4. $\mathrm{Var}(Y_1) \geq \mathrm{Var}(Y_2) \geq \ldots \geq \mathrm{Var}(Y_p) \geq 0$
5. $\mathrm{Var}(\mathbf{Y}) = \mathbf{\Lambda} = \mathrm{Diag}(\lambda_1, \ldots, \lambda_p)$, where $\mathbf{Y} = (Y_1, \ldots Y_p)^{\mathrm{T}}$
6. $\mathrm{tr}(\mathbf{\Sigma}) = \sum_{i=1}^{p} \mathrm{Var}(X_i) = \mathrm{tr}(\mathbf{\Lambda}) = \sum_{i=1}^{p} \lambda_i$
7. $\det(\mathbf{\Sigma}) = \prod_{i=1}^{p} \lambda_i = \det(\mathbf{\Lambda})$
8. $\mathrm{Cov}(X_i, Y_j) = \lambda_j \gamma_{ij}$
9. $\mathrm{Cov}(\mathbf{X}, \mathbf{Y}) = \boldsymbol{\Gamma} \mathbf{\Lambda}$, where $\boldsymbol{\Gamma} = (\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_p)$ is a $(p \times p)$ matrix
10. $\mathrm{Cor}(X_i, Y_j) = \gamma_{ij} \sqrt{\lambda_j} / \sqrt{\sigma_{ii}}$

Comment: Properties 5. and 6. are the theoretical support for the use of PCA as a data reduction technique. In fact, as $\text{tr}(\boldsymbol{\Sigma}) = \sum_{i=1}^{p} \text{Var}(\mathbf{X}) = \text{tr}(\boldsymbol{\Lambda}) = \sum_{i=1}^{p} \lambda_i$ then

$$\frac{\lambda_j}{\sum_{i=1}^{p} \lambda_i}$$

is the **proportion of population total variance due to the j-th principal component**.

Sample Principal Components

Sample PC can be obtained in a similar way just by considering the sample covariance matrix, $\mathbf{S}$, and the rest of the properties follows. This results is summarized in the following theorem.

Theorem: Let $\mathbf{S}$ be the $p \times p$ sample covariance matrix of the random vector $\mathbf{X}$ with $\text{E}(\mathbf{X}) = \boldsymbol{\mu} = \mathbf{0}$. Let $(\hat{\lambda}_1, \hat{\boldsymbol{\gamma}}_1)$, ..., $(\hat{\lambda}_p, \hat{\boldsymbol{\gamma}}_p)$ be the eigenvalues/eigenvectors pairs of $\mathbf{S}$, such that $\hat{\lambda}_1 \geq \hat{\lambda}_2 \geq \ldots \hat{\lambda}_p \geq 0$. The i-th sample principal component is given by:

$$\hat{y}_i = \hat{\boldsymbol{\gamma}}_i^{\text{T}} \mathbf{x} = \hat{\gamma}_{i1} x_1 + \ldots + \hat{\gamma}_{ip} x_p.$$

$$\begin{aligned}
\text{Sample Variance}(\hat{Y}_i) &= \hat{\lambda}_i \\
\text{Sample Covariance}(\hat{Y}_i, \hat{Y}_j) &= 0, \quad \forall i \neq j \\
\text{Total Sample Variance} &= \sum_{i=1}^{p} s_{ii} = \sum_{i=1}^{p} \hat{\lambda}_i \\
\text{Sample Covariance}(X_i, \hat{Y}_j) &= \hat{\gamma}_{ij} \hat{\lambda}_j \\
\text{Sample Correlation}(X_i, \hat{Y}_j) &= \frac{\hat{\gamma}_{ij} \sqrt{\hat{\lambda}_j}}{\sqrt{s_{ii}}}
\end{aligned}$$

Sample Score: Let $\mathbf{x}_l$ represents the observed values on the l-th object. Then $\hat{y}_{li} = \hat{\boldsymbol{\gamma}}_i^{\text{T}} \mathbf{x}_l$ are called the **score** of the l-th object on the i-th PC.

Principal Components obtained from Standardized Variables

Let $Z_i = \frac{X_i - \mu_i}{\sqrt{\sigma_{ii}}}$ be the i-th standardized variable. Then

$$\mathbf{Z} = (Z_1, \ldots, Z_p)^{\text{T}} = \mathbf{D}^{-1/2}(\mathbf{X} - \boldsymbol{\mu}),$$

where $\mathbf{D} = \text{Diag}(\sigma_{11}, \ldots, \sigma_{pp})$

Note that:

- $\text{E}(\mathbf{Z}) = \mathbf{0}$
- $\text{Var}(\mathbf{Z}) = \mathbf{D}^{-1/2} \boldsymbol{\Sigma} \mathbf{D}^{-1/2} = \boldsymbol{\rho} = \text{Cor}(\mathbf{X})$

The principal components of $\mathbf{Z}$ may be obtained from the eigenvectors of the correlation matrix of $\mathbf{X}$, $\text{Cor}(\mathbf{X}) = \boldsymbol{\rho}$

Theorem: Let $\boldsymbol{\rho}$ be a correlation matrix associated with the random vector $\mathbf{X}$. The i-th principal component of the standardized variables $\mathbf{Z} = \mathbf{D}^{-1/2}(\mathbf{X} - \boldsymbol{\mu})$ is given by:

$$Y_i^* = \boldsymbol{\gamma}_i^{*\mathrm{T}}\mathbf{Z} = \gamma_{i1}^* \frac{X_1 - \mu_1}{\sqrt{\sigma_{11}}} + \ldots + \gamma_{ip}^* \frac{X_p - \mu_p}{\sqrt{\sigma_{pp}}}$$

with

$$\mathrm{Var}(Y_i^*) = \boldsymbol{\gamma}_i^{*\mathrm{T}} \boldsymbol{\rho} \boldsymbol{\gamma}_i^* = \lambda_i^*$$
$$\mathrm{Cov}(Y_i^*, Y_j^*) = 0, \quad \forall i \neq j$$

If some λ_i^* are equal, then $\boldsymbol{\gamma}_i^*$ and Y_i^* are not unique.

Sample PC obtained from **standardized** variables follow a similar reasoning.

Additional Properties

PC are not scale invariant: If we change the scale each variable is measured we will obtain a different set of PC. So, **PC's obtained from no Standardized Variables and obtained from Standardized Variables are different**.

Principal Components Analysis (PCA) with R

R commands prcomp() and princomp()

output	prcomp()	princomp()
square root of the eigenvalues	$sdev	$sdev
eigenvectors	$rotation	$loadings
means of each variable	$center	$center
scaling used or FALSE	$scale	$scale
objects in PC's system (scores)	$x	$scores/predict

Exercise 7

1. Table 1 provides the average price in cents per pound of five food items in 24 U.S cities[1].
   (a) Using principal components analysis, define price index measure(s) based on the five food items.
   (b) Identify the most and the least expensive cities (based on the above price index measures). Do the most and the least expensive cities change when standardized data are used as against mean-correlated data? Which type of data should be used to define price index measures? Why?
   (c) Plot the data using principal components scores and identify distinct groups of cities. How are these groups different from each other?
   Hint: Use R and explore the functions `princomp` and `prcomp`.
   (d) Anchorage is characterized by high prices, and can be seem as a potential outlier. Eliminate this observation from your dataset and repeat the analysis. What can you conclude?

Read the data set in R

```
dat<-read.table("Food_Data Ex7.txt",header=TRUE)
  dat
```

[1]U.S. Department of Labor, Bureau of Labor Statistics, Washington, D.C., 1978.

Table 2: Food Price Data. Average price in cents per pound.

City	Bread	Hamburger	Butter	Apples	Tomatoes
Anchorage	70.9	135.6	155	63.9	100.1
Atlanta	36.4	111.5	144.3	53.9	95.9
Baltimore	28.9	108.8	151	47.5	104.5
Boston	43.2	119.3	142	41.1	96.5
Buffalo	34.5	109.9	124.8	35.6	75.9
Chicago	37.1	107.5	145.4	65.1	94.2
Cincinnati	37.1	118.1	149.6	45.6	90.8
Cleveland	38.5	107.7	142.7	50.3	83.2
Dallas	35.5	116.8	142.5	62.4	90.7
Detroit	40.8	108.8	140.1	39.7	96.1
Honolulu	50.9	131.7	154.4	65	93.9
Houston	35.1	102.3	150.3	59.3	84.5
Kansas City	35.1	99.8	162.3	42.6	87.9
Los Angeles	36.9	96.2	140.4	54.7	79.3
Milwaukee	33.3	109.1	123.2	57.7	87.7
Minneapolis	32.5	116.7	135.1	48	89.1
New York	42.7	130.8	148.7	47.6	92.1
Philadelphia	42.9	126.9	153.8	51.9	101.5
Pittsburgh	36.9	115.4	138.9	43.8	91.9
St. Louis	36.9	109.8	140	46.7	79
San Diego	32.5	84.5	145.9	48.5	82.3
San Francisco	40	104.6	139.1	59.2	81.9
Seattle	32.2	105.4	136.8	54	88.6
Washington	31.8	116.7	154.81	57.6	86.6

```
##             City Bread Hamburger Butter Apples Tomatoes
## 1      Anchorage  70.9     135.6 155.00   63.9    100.1
## 2        Atlanta  36.4     111.5 144.30   53.9     95.9
## 3      Baltimore  28.9     108.8 151.00   47.5    104.5
## 4         Boston  43.2     119.3 142.00   41.1     96.5
## 5        Buffalo  34.5     109.9 124.80   35.6     75.9
## 6        Chicago  37.1     107.5 145.40   65.1     94.2
## 7     Cincinnati  37.1     118.1 149.60   45.6     90.8
## 8      Cleveland  38.5     107.7 142.70   50.3     83.2
## 9         Dallas  35.5     116.8 142.50   62.4     90.7
## 10       Detroit  40.8     108.8 140.10   39.7     96.1
## 11      Honolulu  50.9     131.7 154.40   65.0     93.9
## 12       Houston  35.1     102.3 150.30   59.3     84.5
## 13    KansasCity  35.1      99.8 162.30   42.6     87.9
## 14    LosAngeles  36.9      96.2 140.40   54.7     79.3
## 15     Milwaukee  33.3     109.1 123.20   57.7     87.7
## 16   Minneapolis  32.5     116.7 135.10   48.0     89.1
## 17       NewYork  42.7     130.8 148.70   47.6     92.1
## 18  Philadelphia  42.9     126.9 153.80   51.9    101.5
## 19    Pittsburgh  36.9     115.4 138.90   43.8     91.9
## 20      St.Louis  36.9     109.8 140.00   46.7     79.0
## 21      SanDiego  32.5      84.5 145.90   48.5     82.3
## 22 SanFrancisco  40.0     104.6 139.10   59.2     81.9
## 23       Seattle  32.2     105.4 136.80   54.0     88.6
## 24    Washington  31.8     116.7 154.81   57.6     86.6
```

```
dim(dat)
```

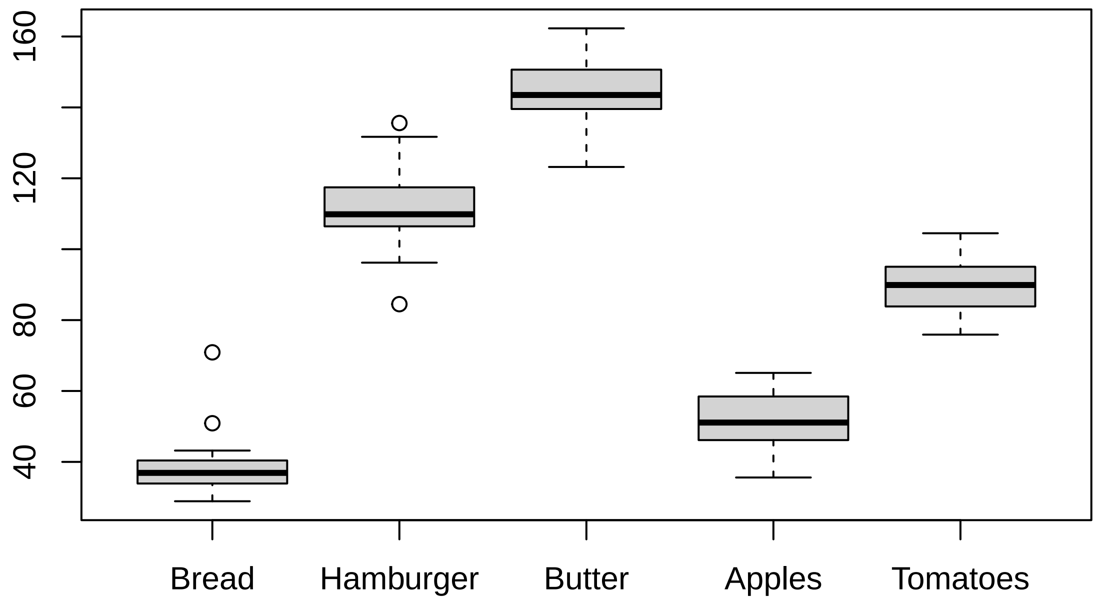
```
## [1] 24  6
```

```
data<-dat[,2:6]
rownames(data)<-dat[,1]  # row names to identify the cities
data
```

```
##              Bread Hamburger Butter Apples Tomatoes
## Anchorage     70.9     135.6 155.00   63.9    100.1
## Atlanta       36.4     111.5 144.30   53.9     95.9
## Baltimore     28.9     108.8 151.00   47.5    104.5
## Boston        43.2     119.3 142.00   41.1     96.5
## Buffalo       34.5     109.9 124.80   35.6     75.9
## Chicago       37.1     107.5 145.40   65.1     94.2
## Cincinnati    37.1     118.1 149.60   45.6     90.8
## Cleveland     38.5     107.7 142.70   50.3     83.2
## Dallas        35.5     116.8 142.50   62.4     90.7
## Detroit       40.8     108.8 140.10   39.7     96.1
## Honolulu      50.9     131.7 154.40   65.0     93.9
## Houston       35.1     102.3 150.30   59.3     84.5
## KansasCity    35.1      99.8 162.30   42.6     87.9
## LosAngeles    36.9      96.2 140.40   54.7     79.3
## Milwaukee     33.3     109.1 123.20   57.7     87.7
## Minneapolis   32.5     116.7 135.10   48.0     89.1
## NewYork       42.7     130.8 148.70   47.6     92.1
## Philadelphia  42.9     126.9 153.80   51.9    101.5
## Pittsburgh    36.9     115.4 138.90   43.8     91.9
## St.Louis      36.9     109.8 140.00   46.7     79.0
## SanDiego      32.5      84.5 145.90   48.5     82.3
## SanFrancisco  40.0     104.6 139.10   59.2     81.9
## Seattle       32.2     105.4 136.80   54.0     88.6
## Washington    31.8     116.7 154.81   57.6     86.6
```

Descriptive statistics

```
boxplot(data)
```



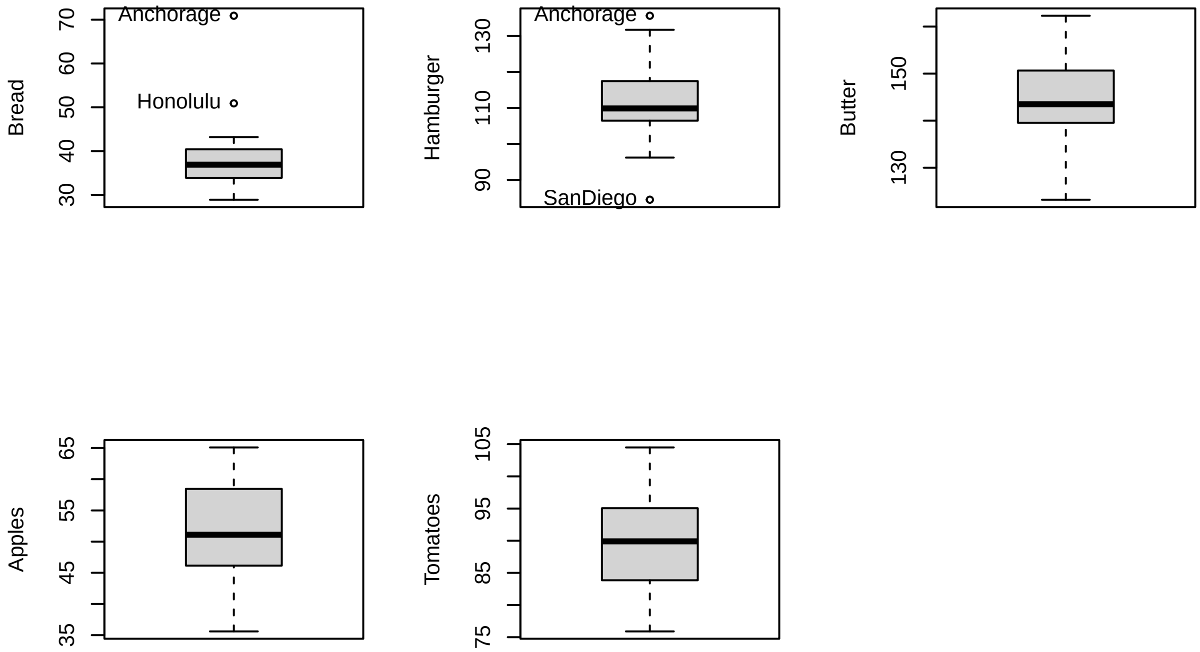

```
library(car)
par(mfrow=c(2,3))
Boxplot(~Bread, data = data, id.n = Inf)
```

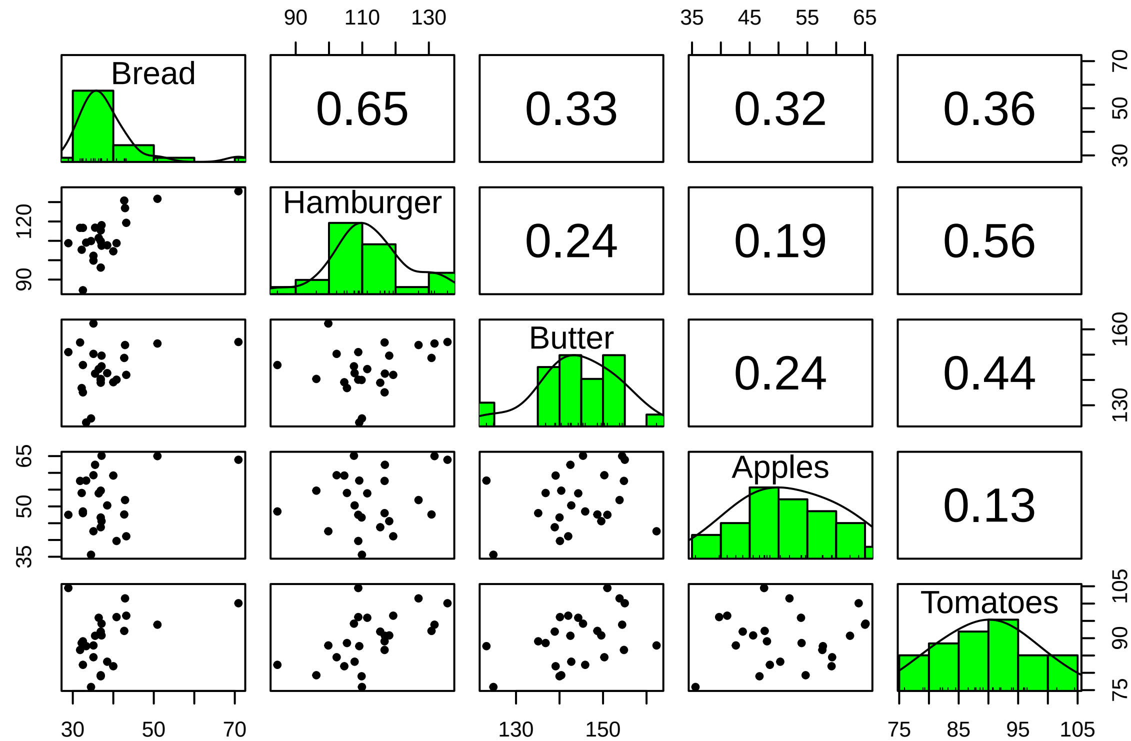
```
## [1] "Anchorage" "Honolulu"
```

```
Boxplot(~Hamburger, data = data, id.n = Inf)
```

```
## [1] "SanDiego"  "Anchorage"
```

```
Boxplot(~Butter, data = data, id.n = Inf)
Boxplot(~Apples, data = data, id.n = Inf)
Boxplot(~Tomatoes, data = data, id.n = Inf)
library(psych)
pairs.panels(data,smooth =FALSE,ellipses=FALSE,lm=FALSE,digits = 2,hist.col="green")
```


Bread
30
40
50
60
70
Anchorage
Honolulu
Hamburger
90
110
130
Anchorage
SanDiego
Butter
130
150
Apples
35
45
55
65
Tomatoes
75
85
95
105




```
summary(data)
```

```
##       Bread          Hamburger          Butter           Apples
##  Min.   :28.90   Min.   : 84.5   Min.   :123.2   Min.   :35.60
##  1st Qu.:34.20   1st Qu.:107.0   1st Qu.:139.8   1st Qu.:46.42
##  Median :36.90   Median :109.8   Median :143.5   Median :51.10
##  Mean   :38.44   Mean   :112.2   Mean   :144.2   Mean   :51.74
##  3rd Qu.:40.20   3rd Qu.:117.1   3rd Qu.:150.5   3rd Qu.:58.08
##  Max.   :70.90   Max.   :135.6   Max.   :162.3   Max.   :65.10
##     Tomatoes
##  Min.   : 75.90
##  1st Qu.: 84.17
##  Median : 89.90
##  Mean   : 89.76
##  3rd Qu.: 94.62
##  Max.   :104.50
```

```
library(psych)
describe(data)
```

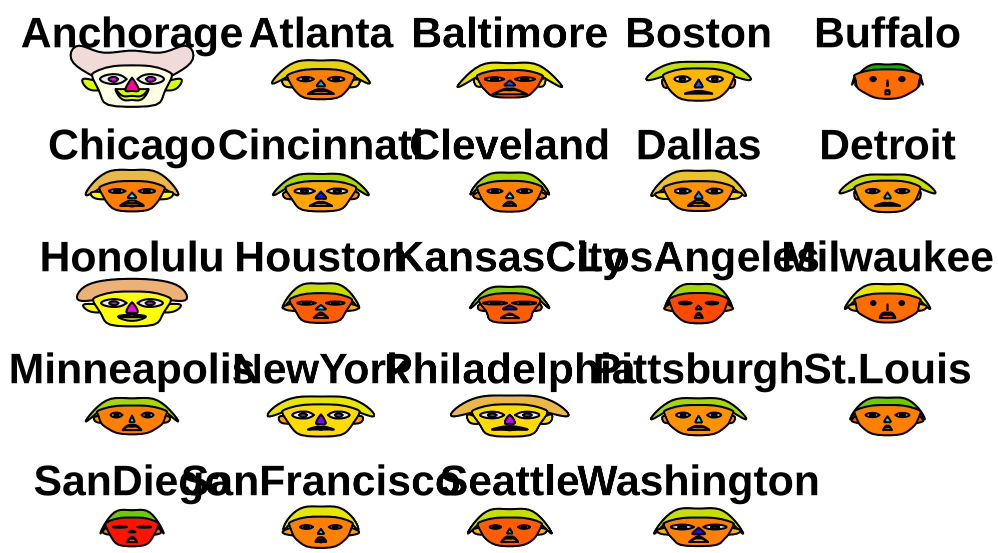
```
##              vars  n   mean    sd median trimmed   mad   min   max range  skew
## Bread           1 24  38.44  8.37  36.90   37.01  4.97  28.9  70.9  42.0  2.44
## Hamburger       2 24 112.25 11.63 109.85  112.30 10.16  84.5 135.6  51.1  0.03
## Butter          3 24 144.21  9.23 143.50  144.79  8.38 123.2 162.3  39.1 -0.38
## Apples          4 24  51.74  8.36  51.10   51.81  9.71  35.6  65.1  29.5  0.00
## Tomatoes        5 24  89.76  7.40  89.90   89.67  8.45  75.9 104.5  28.6  0.05
##              kurtosis   se
```

```
## Bread          6.79 1.71
## Hamburger     -0.02 2.37
## Butter        -0.09 1.88
## Apples        -1.10 1.71
## Tomatoes      -0.84 1.51
```

```
apply(data, 2 ,sd)
```

```
##     Bread Hamburger    Butter    Apples  Tomatoes
##  8.366544 11.633160  9.226894  8.359000  7.398879
```

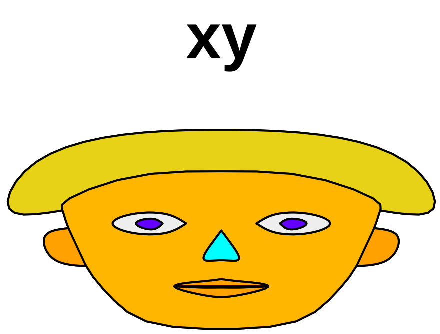
```
library(aplpack)
faces(data)
```




```
## effect of variables:
##  modified item       Var
##  "height of face   " "Bread"
##  "width of face    " "Hamburger"
##  "structure of face" "Butter"
##  "height of mouth  " "Apples"
##  "width of mouth   " "Tomatoes"
##  "smiling          " "Bread"
##  "height of eyes   " "Hamburger"
##  "width of eyes    " "Butter"
##  "height of hair   " "Apples"
##  "width of hair   "  "Tomatoes"
##  "style of hair   "  "Bread"
##  "height of nose  "  "Hamburger"
```

```
##  "width of nose   "  "Butter"
##  "width of ear    "  "Apples"
##  "height of ear   "  "Tomatoes"
```

```r
# plot the face for the mean vector
means<- apply(data, 2 ,mean)
faces(means)
```




```
## effect of variables:
##  modified item       Var
##  "height of face   " "Bread"
##  "width of face    " "Hamburger"
##  "structure of face" "Butter"
##  "height of mouth  " "Apples"
##  "width of mouth   " "Tomatoes"
##  "smiling          " "Bread"
##  "height of eyes   " "Hamburger"
##  "width of eyes    " "Butter"
##  "height of hair   " "Apples"
##  "width of hair   "  "Tomatoes"
##  "style of hair   "  "Bread"
##  "height of nose  "  "Hamburger"
##  "width of nose   "  "Butter"
##  "width of ear    "  "Apples"
##  "height of ear   "  "Tomatoes"
```

```r
# data-mean(data) - the program R compute the PC for the data-mean(data)
datascale=scale(data, center = TRUE, scale = FALSE) # subtracted the mean
datascale
```

```
##                    Bread  Hamburger       Butter  Apples    Tomatoes
## Anchorage    32.45833333  23.3541667  10.78708333  12.1625  10.3416667
## Atlanta      -2.04166667  -0.7458333   0.08708333   2.1625   6.1416667
## Baltimore    -9.54166667  -3.4458333   6.78708333  -4.2375  14.7416667
## Boston        4.75833333   7.0541667  -2.21291667 -10.6375   6.7416667
## Buffalo      -3.94166667  -2.3458333 -19.41291667 -16.1375 -13.8583333
## Chicago      -1.34166667  -4.7458333   1.18708333  13.3625   4.4416667
## Cincinnati   -1.34166667   5.8541667   5.38708333  -6.1375   1.0416667
## Cleveland     0.05833333  -4.5458333  -1.51291667  -1.4375  -6.5583333
## Dallas       -2.94166667   4.5541667  -1.71291667  10.6625   0.9416667
## Detroit       2.35833333  -3.4458333  -4.11291667 -12.0375   6.3416667
## Honolulu     12.45833333  19.4541667  10.18708333  13.2625   4.1416667
## Houston      -3.34166667  -9.9458333   6.08708333   7.5625  -5.2583333
## KansasCity   -3.34166667 -12.4458333  18.08708333  -9.1375  -1.8583333
## LosAngeles   -1.54166667 -16.0458333  -3.81291667   2.9625 -10.4583333
## Milwaukee    -5.14166667  -3.1458333 -21.01291667   5.9625  -2.0583333
## Minneapolis  -5.94166667   4.4541667  -9.11291667  -3.7375  -0.6583333
## NewYork       4.25833333  18.5541667   4.48708333  -4.1375   2.3416667
## Philadelphia  4.45833333  14.6541667   9.58708333   0.1625  11.7416667
## Pittsburgh   -1.54166667   3.1541667  -5.31291667  -7.9375   2.1416667
## St.Louis     -1.54166667  -2.4458333  -4.21291667  -5.0375 -10.7583333
## SanDiego     -5.94166667 -27.7458333   1.68708333  -3.2375  -7.4583333
## SanFrancisco  1.55833333  -7.6458333  -5.11291667   7.4625  -7.8583333
## Seattle      -6.24166667  -6.8458333  -7.41291667   2.2625  -1.1583333
## Washington   -6.64166667   4.4541667  10.59708333   5.8625  -3.1583333
## attr(,"scaled:center")
##     Bread Hamburger    Butter    Apples  Tomatoes
##  38.44167 112.24583 144.21292  51.73750  89.75833
```

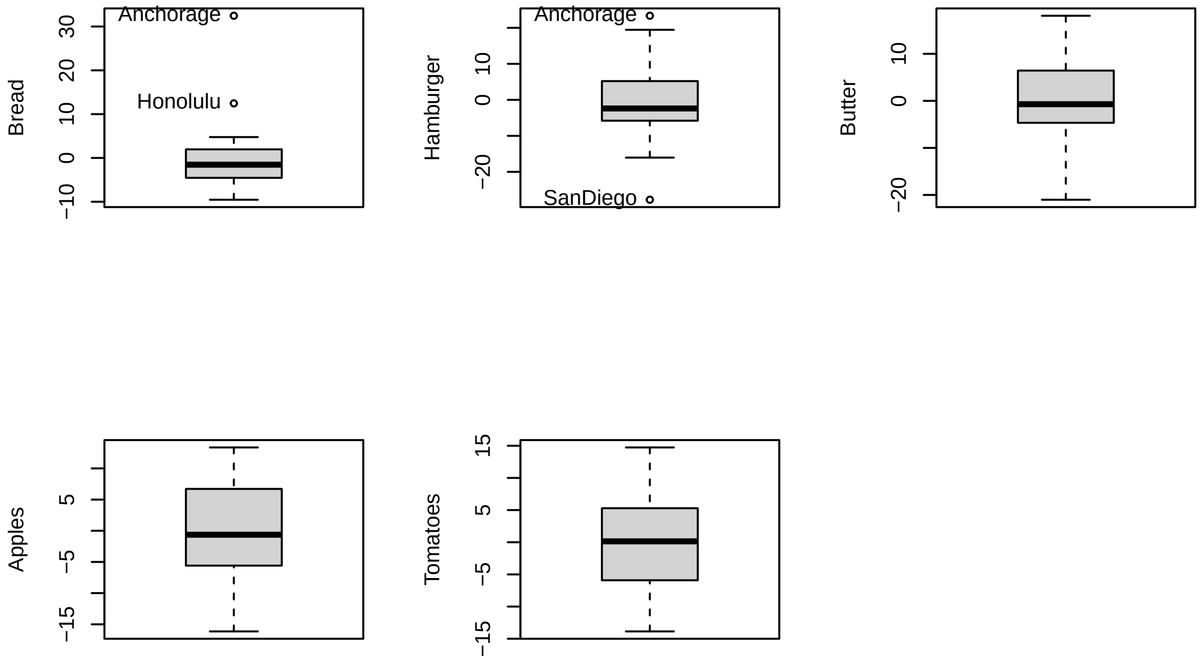
```r
par(mfrow=c(2,3))
Boxplot(~Bread, data = datascale, id.n = Inf)
```

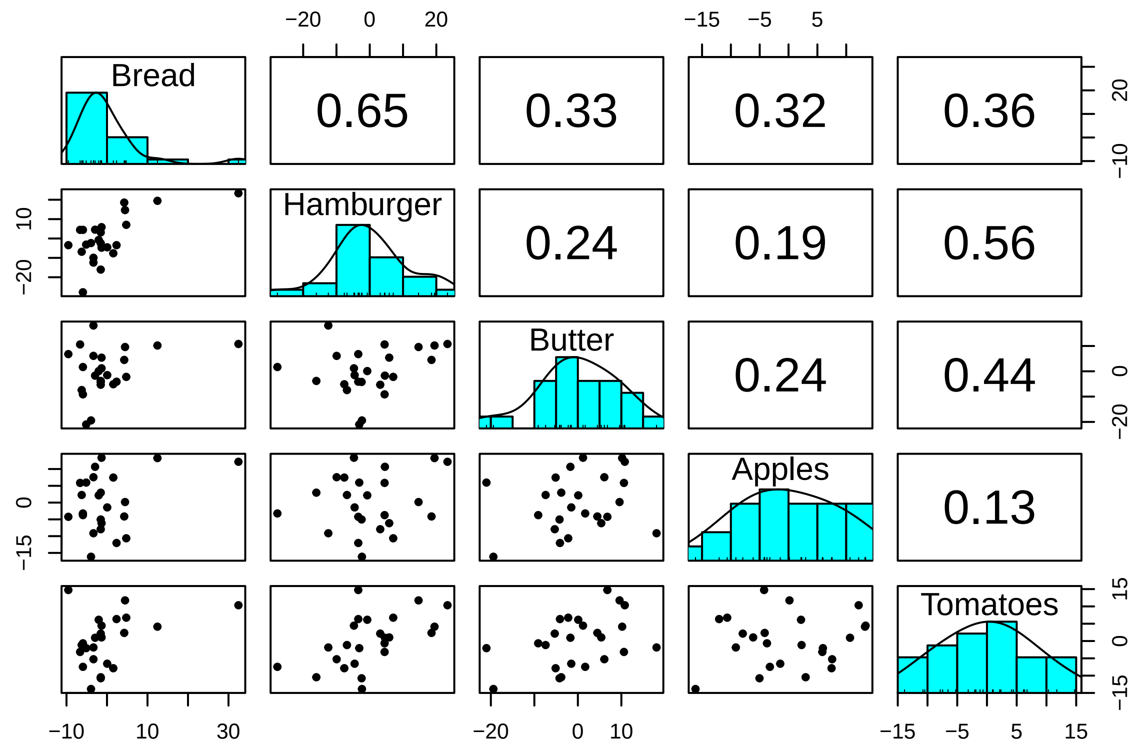
```
## [1] "Anchorage" "Honolulu"
```

```r
Boxplot(~Hamburger, data = datascale, id.n = Inf)
```

```
## [1] "SanDiego"  "Anchorage"
```

```r
Boxplot(~Butter, data = datascale, id.n = Inf)
Boxplot(~Apples, data = datascale, id.n = Inf)
Boxplot(~Tomatoes, data = datascale, id.n = Inf)
pairs.panels(datascale,smooth =FALSE,ellipses=FALSE,lm=FALSE)
```

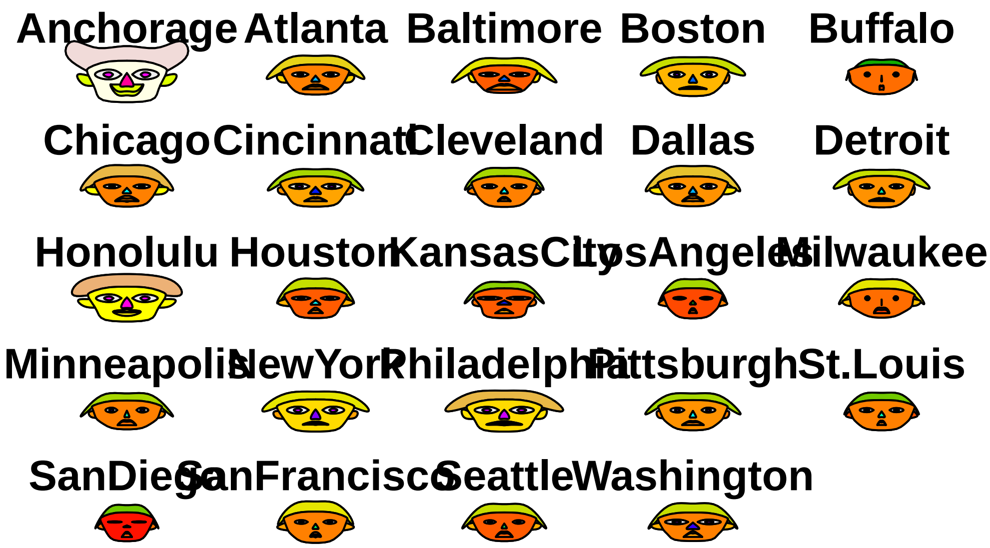

Bread
Anchorage
Honolulu
Hamburger
Anchorage
SanDiego
Butter
Apples
Tomatoes




```
summary(datascale)
```

```
##      Bread          Hamburger          Butter             Apples
##  Min.   :-9.542   Min.   :-27.746   Min.   :-21.0129   Min.   :-16.1375
##  1st Qu.:-4.242   1st Qu.: -5.271   1st Qu.: -4.4379   1st Qu.: -5.3125
##  Median :-1.542   Median : -2.396   Median : -0.7129   Median : -0.6375
##  Mean   : 0.000   Mean   :  0.000   Mean   :  0.0000   Mean   :  0.0000
##  3rd Qu.: 1.758   3rd Qu.:  4.879   3rd Qu.:  6.2621   3rd Qu.:  6.3375
##  Max.   :32.458   Max.   : 23.354   Max.   : 18.0871   Max.   : 13.3625
##     Tomatoes
##  Min.   :-13.8583
##  1st Qu.: -5.5833
##  Median :  0.1417
##  Mean   :  0.0000
##  3rd Qu.:  4.8667
##  Max.   : 14.7417
```

```
faces(datascale)
```




```
## effect of variables:
##  modified item       Var
##  "height of face   " "Bread"
##  "width of face    " "Hamburger"
##  "structure of face" "Butter"
##  "height of mouth  " "Apples"
##  "width of mouth   " "Tomatoes"
##  "smiling          " "Bread"
##  "height of eyes   " "Hamburger"
##  "width of eyes    " "Butter"
##  "height of hair   " "Apples"
##  "width of hair   "  "Tomatoes"
##  "style of hair   "  "Bread"
##  "height of nose  "  "Hamburger"
##  "width of nose   "  "Butter"
##  "width of ear    "  "Apples"
##  "height of ear   "  "Tomatoes"
```

1. PCA for the original dataset / PCA based on the covariance matrix

```r
round(cov(datascale),3)
```

eigen decomposition of covariance matrix

```
##             Bread Hamburger Butter Apples Tomatoes
## Bread      69.999    63.172 25.489 22.289   22.413
```

```
## Hamburger 63.172   135.330 26.274 18.563   47.839
## Butter    25.489    26.274 85.136 18.136   29.774
## Apples    22.289    18.563 18.136 69.873    8.249
## Tomatoes  22.413    47.839 29.774  8.249   54.743
```

```
eigen(cov(datascale))
```

```
## eigen() decomposition
## $values
## [1] 216.79440  79.12794  62.26846  34.67047  22.22005
##
## $vectors
##            [,1]        [,2]        [,3]       [,4]        [,5]
## [1,] -0.4529089  0.05515147  0.21435116  0.6856702  0.52511130
## [2,] -0.7146773  0.48679539 -0.02261341 -0.1338116 -0.48358009
## [3,] -0.3391656 -0.75632931 -0.43256354  0.1976187 -0.29456459
## [4,] -0.2203644 -0.42895099  0.81242257 -0.3231370 -0.05470465
## [5,] -0.3471543 -0.06289354 -0.32619100 -0.6070257  0.63296724
```

```
val<-round(sqrt(eigen(cov(datascale))$values),3)
val
```

```
## [1] 14.724  8.895  7.891  5.888  4.714
```

```
vec<-round(eigen(cov(datascale))$vectors,3)
vec
```

```
##        [,1]   [,2]   [,3]   [,4]   [,5]
## [1,] -0.453  0.055  0.214  0.686  0.525
## [2,] -0.715  0.487 -0.023 -0.134 -0.484
## [3,] -0.339 -0.756 -0.433  0.198 -0.295
## [4,] -0.220 -0.429  0.812 -0.323 -0.055
## [5,] -0.347 -0.063 -0.326 -0.607  0.633
```

```
round(cov(data),3)  # equals cov(datascale)
```

```
##            Bread Hamburger Butter Apples Tomatoes
## Bread     69.999    63.172 25.489 22.289   22.413
## Hamburger 63.172   135.330 26.274 18.563   47.839
## Butter    25.489    26.274 85.136 18.136   29.774
## Apples    22.289    18.563 18.136 69.873    8.249
## Tomatoes  22.413    47.839 29.774  8.249   54.743
```

PCA with prcomp()

```
food.pca_1 <- prcomp(data)
food.pca_1
```

```
## Standard deviations (1, .., p=5):
## [1] 14.723940  8.895389  7.891037  5.888164  4.713815
##
## Rotation (n x k) = (5 x 5):
##                  PC1         PC2         PC3        PC4         PC5
## Bread     -0.4529089  0.05515147 -0.21435116 -0.6856702  0.52511130
## Hamburger -0.7146773  0.48679539  0.02261341  0.1338116 -0.48358009
## Butter    -0.3391656 -0.75632931  0.43256354 -0.1976187 -0.29456459
## Apples    -0.2203644 -0.42895099 -0.81242257  0.3231370 -0.05470465
```

```
## Tomatoes  -0.3471543 -0.06289354  0.32619100  0.6070257  0.63296724
```

```
summary(food.pca_1)
```

```
## Importance of components:
##                            PC1    PC2    PC3     PC4     PC5
## Standard deviation     14.7239 8.8954 7.8910 5.88816 4.71381
## Proportion of Variance  0.5223 0.1906 0.1500 0.08353 0.05353
## Cumulative Proportion   0.5223 0.7129 0.8629 0.94647 1.00000
```

```
food.pca_1$sdev
```

```
## [1] 14.723940  8.895389  7.891037  5.888164  4.713815
```

```
food.pca_1$rotation
```

```
##                  PC1         PC2         PC3        PC4         PC5
## Bread     -0.4529089  0.05515147 -0.21435116 -0.6856702  0.52511130
## Hamburger -0.7146773  0.48679539  0.02261341  0.1338116 -0.48358009
## Butter    -0.3391656 -0.75632931  0.43256354 -0.1976187 -0.29456459
## Apples    -0.2203644 -0.42895099 -0.81242257  0.3231370 -0.05470465
## Tomatoes  -0.3471543 -0.06289354  0.32619100  0.6070257  0.63296724
```

```
food.pca_1$center
```

```
##     Bread Hamburger    Butter    Apples  Tomatoes
##  38.44167 112.24583 144.21292  51.73750  89.75833
```

```
food.pca_1$scale
```

```
## [1] FALSE
```

```
### Scores
scores_1<-round(food.pca_1$x,3)
scores_1
```

```
##                 PC1     PC2     PC3     PC4    PC5
## Anchorage   -41.320  -0.867  -8.271 -11.055  8.454
## Atlanta      -1.180  -1.855   0.705   5.710  3.032
## Baltimore     0.298  -6.446  13.154  12.319  4.219
## Boston       -6.442   9.509   9.024  -1.226  4.588
## Buffalo      18.413  21.117   0.985  -7.402 -3.106
## Chicago      -0.890  -9.293  -8.713   7.064  3.321
## Cincinnati   -4.412   1.269   8.076  -0.712 -4.127
## Cleveland     6.329  -0.036  -1.741  -4.795 -1.398
## Dallas       -4.018  -1.283  -8.363   6.982 -3.230
## Detroit       3.241   6.328   9.486  -1.306  8.789
## Honolulu    -27.361  -3.497  -7.248  -1.153 -3.970
## Houston       6.716 -12.543  -4.735  -0.991 -2.480
## KansasCity    6.932 -15.886  15.076  -7.029 -1.740
## LosAngeles   16.437  -5.625  -7.500  -5.728  1.291
## Milwaukee    11.104  11.650 -13.574   7.934  3.382
## Minneapolis   3.651  10.378   0.254   4.864 -2.802
## NewYork     -16.612   7.501   5.573  -1.239 -6.350
## Philadelphia -19.856  -0.680   7.221   4.189 -0.146
## Pittsburgh    1.252   8.739   5.251   1.264  1.020
## St.Louis      8.720   4.748  -0.964  -6.596 -4.920
## SanDiego     25.251 -13.252   1.573  -5.546  5.257
```

```
## SanFrancisco   7.576  -2.476 -11.345  -3.440   0.639
## Seattle       10.137   1.032  -4.239   4.857   1.360
## Washington    -3.965  -8.529   0.315   3.033 -11.083
```

```
rownames(scores_1)<-rownames(data)
scores_1
```

```
##                 PC1     PC2     PC3     PC4     PC5
## Anchorage    -41.320  -0.867  -8.271 -11.055   8.454
## Atlanta       -1.180  -1.855   0.705   5.710   3.032
## Baltimore      0.298  -6.446  13.154  12.319   4.219
## Boston        -6.442   9.509   9.024  -1.226   4.588
## Buffalo       18.413  21.117   0.985  -7.402  -3.106
## Chicago       -0.890  -9.293  -8.713   7.064   3.321
## Cincinnati    -4.412   1.269   8.076  -0.712  -4.127
## Cleveland      6.329  -0.036  -1.741  -4.795  -1.398
## Dallas        -4.018  -1.283  -8.363   6.982  -3.230
## Detroit        3.241   6.328   9.486  -1.306   8.789
## Honolulu     -27.361  -3.497  -7.248  -1.153  -3.970
## Houston        6.716 -12.543  -4.735  -0.991  -2.480
## KansasCity     6.932 -15.886  15.076  -7.029  -1.740
## LosAngeles    16.437  -5.625  -7.500  -5.728   1.291
## Milwaukee     11.104  11.650 -13.574   7.934   3.382
## Minneapolis    3.651  10.378   0.254   4.864  -2.802
## NewYork      -16.612   7.501   5.573  -1.239  -6.350
## Philadelphia -19.856  -0.680   7.221   4.189  -0.146
## Pittsburgh     1.252   8.739   5.251   1.264   1.020
## St.Louis       8.720   4.748  -0.964  -6.596  -4.920
## SanDiego      25.251 -13.252   1.573  -5.546   5.257
## SanFrancisco   7.576  -2.476 -11.345  -3.440   0.639
## Seattle       10.137   1.032  -4.239   4.857   1.360
## Washington    -3.965  -8.529   0.315   3.033 -11.083
```

```
### Ranking the cities according to the 1st sample PC
o<-order(scores_1[,1])  # Scores sorted by 1st PC
scores.pc1<-scores_1[o,]
scores.pc1
```

```
##                 PC1     PC2     PC3     PC4     PC5
## Anchorage    -41.320  -0.867  -8.271 -11.055   8.454
## Honolulu     -27.361  -3.497  -7.248  -1.153  -3.970
## Philadelphia -19.856  -0.680   7.221   4.189  -0.146
## NewYork      -16.612   7.501   5.573  -1.239  -6.350
## Boston        -6.442   9.509   9.024  -1.226   4.588
## Cincinnati    -4.412   1.269   8.076  -0.712  -4.127
## Dallas        -4.018  -1.283  -8.363   6.982  -3.230
## Washington    -3.965  -8.529   0.315   3.033 -11.083
## Atlanta       -1.180  -1.855   0.705   5.710   3.032
## Chicago       -0.890  -9.293  -8.713   7.064   3.321
## Baltimore      0.298  -6.446  13.154  12.319   4.219
## Pittsburgh     1.252   8.739   5.251   1.264   1.020
## Detroit        3.241   6.328   9.486  -1.306   8.789
## Minneapolis    3.651  10.378   0.254   4.864  -2.802
## Cleveland      6.329  -0.036  -1.741  -4.795  -1.398
## Houston        6.716 -12.543  -4.735  -0.991  -2.480
```

```
## KansasCity     6.932 -15.886  15.076  -7.029  -1.740
## SanFrancisco   7.576  -2.476 -11.345  -3.440   0.639
## St.Louis       8.720   4.748  -0.964  -6.596  -4.920
## Seattle       10.137   1.032  -4.239   4.857   1.360
## Milwaukee     11.104  11.650 -13.574   7.934   3.382
## LosAngeles    16.437  -5.625  -7.500  -5.728   1.291
## Buffalo       18.413  21.117   0.985  -7.402  -3.106
## SanDiego      25.251 -13.252   1.573  -5.546   5.257
```

```
### Correlation between the original variables and each sample PC
round(cor(data,scores_1),3)
```

```
##              PC1    PC2    PC3    PC4    PC5
## Bread     -0.797  0.059 -0.202 -0.483  0.296
## Hamburger -0.905  0.372  0.015  0.068 -0.196
## Butter    -0.541 -0.729  0.370 -0.126 -0.150
## Apples    -0.388 -0.456 -0.767  0.228 -0.031
## Tomatoes  -0.691 -0.076  0.348  0.483  0.403
```

```
(food.pca_1$sdev[1]*food.pca_1$rotation[1,1])/sd(datascale[,1])
```

```
## [1] -0.7970559
```

PCA with princomp()

```
food.pca_2 <- princomp(data)
food.pca_2
```

```
## Call:
## princomp(x = data)
##
## Standard deviations:
##    Comp.1    Comp.2    Comp.3    Comp.4    Comp.5
## 14.413927  8.708096  7.724891  5.764189  4.614566
##
##  5  variables and  24 observations.
```

```
summary(food.pca_2)
```

```
## Importance of components:
##                            Comp.1    Comp.2    Comp.3     Comp.4    Comp.5
## Standard deviation     14.4139273 8.7080964 7.7248912 5.76418865 4.6145655
## Proportion of Variance  0.5222938 0.1906324 0.1500151 0.08352694 0.0535318
## Cumulative Proportion   0.5222938 0.7129262 0.8629413 0.94646820 1.0000000
```

```
food.pca_2$sdev
```

```
##    Comp.1    Comp.2    Comp.3    Comp.4    Comp.5
## 14.413927  8.708096  7.724891  5.764189  4.614566
```

```
loadings<-round(loadings(food.pca_2),3)
loadings
```

```
##
## Loadings:
##           Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## Bread      0.453         0.214  0.686  0.525
## Hamburger  0.715  0.487        -0.134 -0.484
```

```
## Butter     0.339 -0.756 -0.433  0.198 -0.295
## Apples     0.220 -0.429  0.812 -0.323
## Tomatoes   0.347        -0.326 -0.607  0.633
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## SS loadings       1.0    1.0  0.999  1.001  1.001
## Proportion Var    0.2    0.2  0.200  0.200  0.200
## Cumulative Var    0.2    0.4  0.600  0.800  1.000
```

```
scores_2<-round(food.pca_2$scores,3)
scores_2
```

```
##                Comp.1  Comp.2  Comp.3  Comp.4  Comp.5
## Anchorage      41.320  -0.867   8.271  11.055   8.454
## Atlanta         1.180  -1.855  -0.705  -5.710   3.032
## Baltimore      -0.298  -6.446 -13.154 -12.319   4.219
## Boston          6.442   9.509  -9.024   1.226   4.588
## Buffalo       -18.413  21.117  -0.985   7.402  -3.106
## Chicago         0.890  -9.293   8.713  -7.064   3.321
## Cincinnati      4.412   1.269  -8.076   0.712  -4.127
## Cleveland      -6.329  -0.036   1.741   4.795  -1.398
## Dallas          4.018  -1.283   8.363  -6.982  -3.230
## Detroit        -3.241   6.328  -9.486   1.306   8.789
## Honolulu       27.361  -3.497   7.248   1.153  -3.970
## Houston        -6.716 -12.543   4.735   0.991  -2.480
## KansasCity     -6.932 -15.886 -15.076   7.029  -1.740
## LosAngeles    -16.437  -5.625   7.500   5.728   1.291
## Milwaukee     -11.104  11.650  13.574  -7.934   3.382
## Minneapolis    -3.651  10.378  -0.254  -4.864  -2.802
## NewYork        16.612   7.501  -5.573   1.239  -6.350
## Philadelphia   19.856  -0.680  -7.221  -4.189  -0.146
## Pittsburgh     -1.252   8.739  -5.251  -1.264   1.020
## St.Louis       -8.720   4.748   0.964   6.596  -4.920
## SanDiego      -25.251 -13.252  -1.573   5.546   5.257
## SanFrancisco   -7.576  -2.476  11.345   3.440   0.639
## Seattle       -10.137   1.032   4.239  -4.857   1.360
## Washington      3.965  -8.529  -0.315  -3.033 -11.083
```

```
scores_2_alt<-round(predict(food.pca_2),3)
scores_2_alt
```

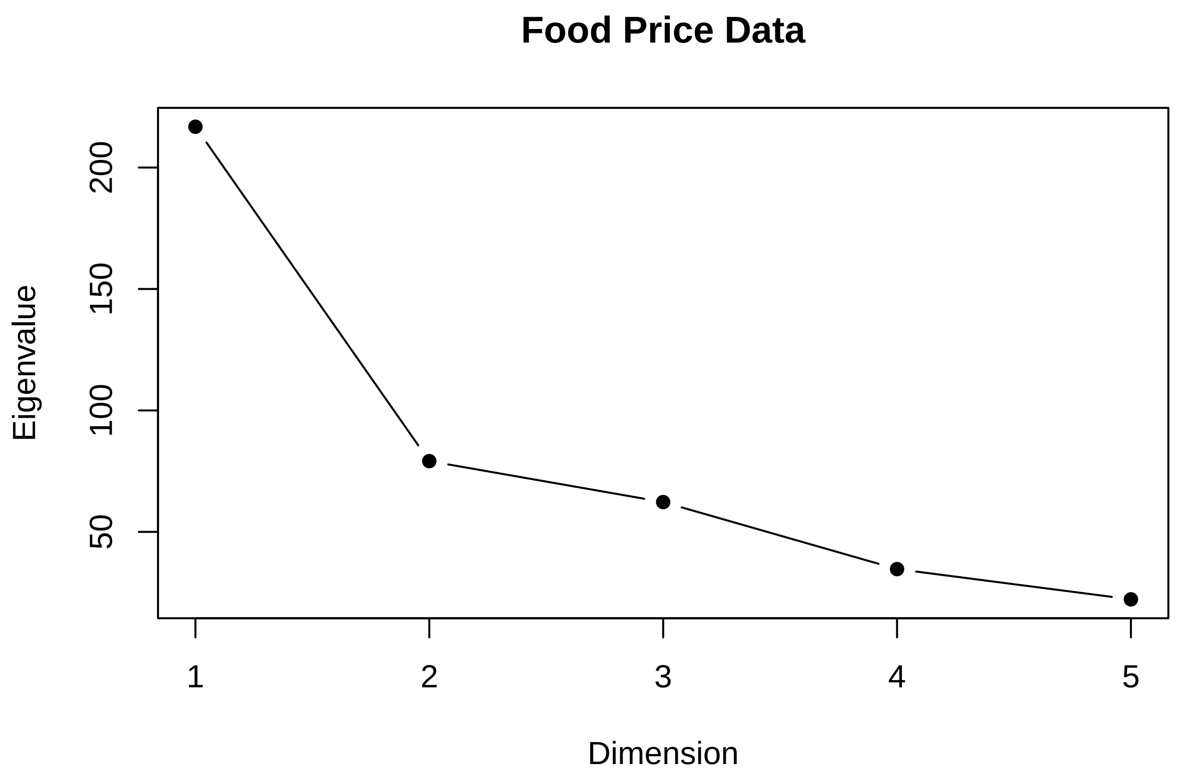
```
##                Comp.1  Comp.2  Comp.3  Comp.4  Comp.5
## Anchorage      41.320  -0.867   8.271  11.055   8.454
## Atlanta         1.180  -1.855  -0.705  -5.710   3.032
## Baltimore      -0.298  -6.446 -13.154 -12.319   4.219
## Boston          6.442   9.509  -9.024   1.226   4.588
## Buffalo       -18.413  21.117  -0.985   7.402  -3.106
## Chicago         0.890  -9.293   8.713  -7.064   3.321
## Cincinnati      4.412   1.269  -8.076   0.712  -4.127
## Cleveland      -6.329  -0.036   1.741   4.795  -1.398
## Dallas          4.018  -1.283   8.363  -6.982  -3.230
## Detroit        -3.241   6.328  -9.486   1.306   8.789
## Honolulu       27.361  -3.497   7.248   1.153  -3.970
## Houston        -6.716 -12.543   4.735   0.991  -2.480
## KansasCity     -6.932 -15.886 -15.076   7.029  -1.740
```

```
## LosAngeles   -16.437  -5.625   7.500   5.728   1.291
## Milwaukee    -11.104  11.650  13.574  -7.934   3.382
## Minneapolis   -3.651  10.378  -0.254  -4.864  -2.802
## NewYork       16.612   7.501  -5.573   1.239  -6.350
## Philadelphia  19.856  -0.680  -7.221  -4.189  -0.146
## Pittsburgh    -1.252   8.739  -5.251  -1.264   1.020
## St.Louis      -8.720   4.748   0.964   6.596  -4.920
## SanDiego     -25.251 -13.252  -1.573   5.546   5.257
## SanFrancisco  -7.576  -2.476  11.345   3.440   0.639
## Seattle      -10.137   1.032   4.239  -4.857   1.360
## Washington     3.965  -8.529  -0.315  -3.033 -11.083
```

Scree plot - number of PC's (library(psy))

Scores plot

```
library(psy)
scree.plot(data,title="Food Price Data",type="V") # covariance matrix
```

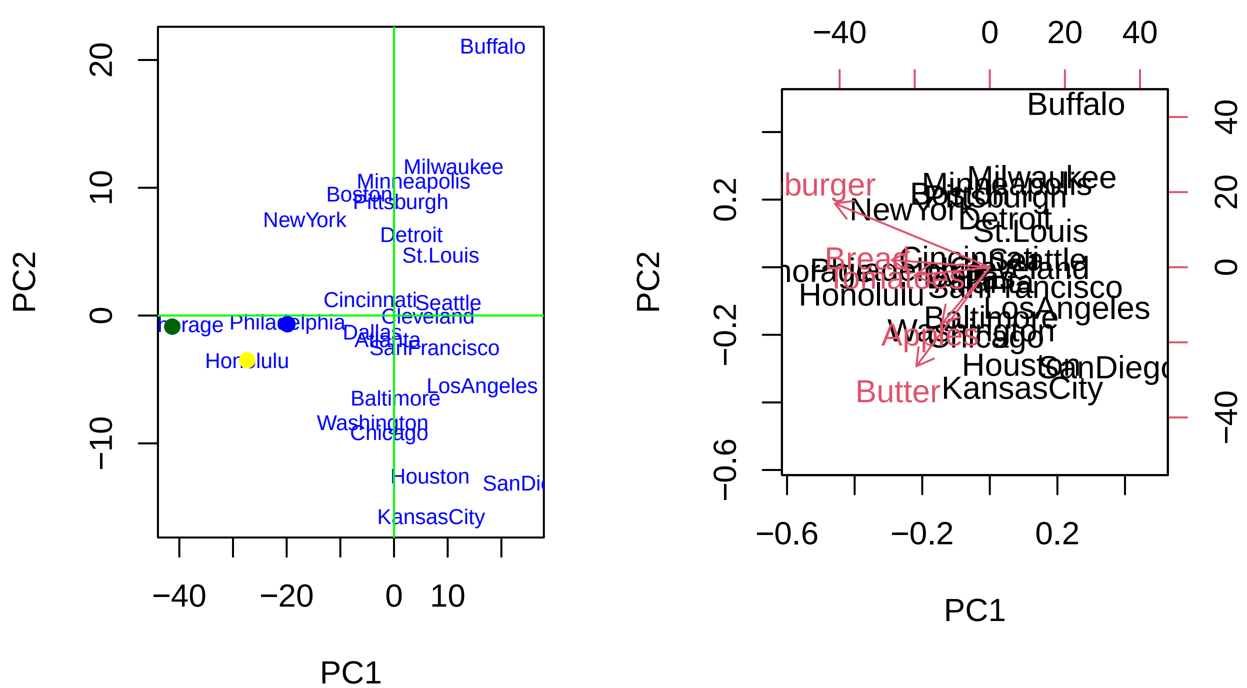



```
par(mfrow=c(1,2))
plot(scores_1[,1], scores_1[,2], xlab="PC1", ylab="PC2",type="n",
xlim=c(min(scores_1[,1]), max(scores_1[,1])),ylim=c(min(scores_1[,2]),
max(scores_1[,2])))
text(scores_1[,1],scores_1[,2], rownames(scores_1), col="blue",cex=0.7)
abline(h=mean(scores_1[,2]),col="green")
abline(v=mean(scores_1[,1]),col="green")
```

```r
points(scores_1["Anchorage",1],scores_1["Anchorage",2],pch=19,col="darkgreen")
points(scores_1["Honolulu",1],scores_1["Honolulu",2],pch=19,col="yellow")
points(scores_1["Philadelphia",1],scores_1["Philadelphia",2],pch=19,col="blue")

# Biplot:
#
# A biplot plots on the same plane the principal component scores
# and loadings representing the contribution of each of the original
# variables to these components.
# In PC analysis, it plots the first two components
# and the original variables, the latter as direction vectors,
# with the direction indicating the relationship between
# the principal components and the original variables.

biplot(food.pca_1)
```



2. PCA for Standardized Variables / PCA based on Correlation matrix

```r
food.pca_1.cor <- prcomp(data,scale = TRUE)
food.pca_1.cor
```

```
## Standard deviations (1, .., p=5):
## [1] 1.5618436 0.9641531 0.9128186 0.7299813 0.5147260
##
## Rotation (n x k) = (5 x 5):
##                 PC1         PC2        PC3         PC4         PC5
## Bread    -0.5099267  0.05649609 -0.4017162 -0.53197875 -0.54074549
```

```
## Hamburger -0.5200718 -0.27761601 -0.4074371  0.07148075  0.69378685
## Butter    -0.3973106  0.09940133  0.7684773 -0.43041438  0.23759156
## Apples    -0.2909422  0.87675286 -0.0713631  0.37257819  0.05243928
## Tomatoes  -0.4764421 -0.37571444  0.2774329  0.62275040 -0.40872301
```

```
summary(food.pca_1.cor)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4     PC5
## Standard deviation     1.5618 0.9642 0.9128 0.7300 0.51473
## Proportion of Variance 0.4879 0.1859 0.1666 0.1066 0.05299
## Cumulative Proportion  0.4879 0.6738 0.8404 0.9470 1.00000
```

```
food.pca_1.cor$sdev
```

```
## [1] 1.5618436 0.9641531 0.9128186 0.7299813 0.5147260
```

```
food.pca_1.cor$rotation
```

```
##                  PC1         PC2        PC3         PC4         PC5
## Bread     -0.5099267  0.05649609 -0.4017162 -0.53197875 -0.54074549
## Hamburger -0.5200718 -0.27761601 -0.4074371  0.07148075  0.69378685
## Butter    -0.3973106  0.09940133  0.7684773 -0.43041438  0.23759156
## Apples    -0.2909422  0.87675286 -0.0713631  0.37257819  0.05243928
## Tomatoes  -0.4764421 -0.37571444  0.2774329  0.62275040 -0.40872301
```

```
food.pca_1.cor$center
```

```
##     Bread Hamburger    Butter    Apples  Tomatoes
##  38.44167 112.24583 144.21292  51.73750  89.75833
```

```
food.pca_1.cor$scale
```

```
##     Bread Hamburger    Butter    Apples  Tomatoes
##  8.366544 11.633160  9.226894  8.359000  7.398879
```

```
scores_1.cor<-round(food.pca_1.cor$x,3)
rownames(scores_1.cor)<-rownames(data)
scores_1.cor
```

```
##                PC1    PC2    PC3    PC4    PC5
## Anchorage   -4.576  0.529 -1.194 -1.011 -0.922
## Atlanta     -0.317 -0.080  0.343  0.734 -0.236
## Baltimore   -0.358 -1.102  1.733  1.321 -0.255
## Boston      -0.574 -1.618 -0.316 -0.063 -0.383
## Buffalo      2.635 -1.169 -1.727 -0.744  0.279
## Chicago     -0.508  1.293  0.382  0.970 -0.327
## Cincinnati  -0.265 -0.787  0.400 -0.316  0.479
## Cleveland    0.737  0.275 -0.203 -0.577  0.039
## Dallas      -0.382  0.924 -0.217  0.849  0.432
## Detroit      0.198 -1.531  0.005  0.018 -0.890
## Honolulu    -2.796  0.910 -0.389 -0.208  0.472
## Houston      0.462  1.341  0.754 -0.238  0.117
## KansasCity   0.419 -0.395  2.111 -1.271 -0.015
## LosAngeles   1.546  1.173 -0.099 -0.571 -0.359
## Milwaukee    1.284  0.544 -1.521  1.380 -0.245
## Minneapolis  0.728 -0.603 -0.622  0.608  0.428
## NewYork     -1.289 -0.919 -0.357 -0.353  0.792
```

```
## Philadelphia -2.101 -0.796  0.510  0.355  0.185
## Pittsburgh    0.320 -1.084 -0.331  0.192 -0.017
## St.Louis      1.253  0.021 -0.552 -0.851  0.408
## SanDiego      2.123  0.679  1.146 -0.643 -0.836
## SanFrancisco  0.713  1.320 -0.591 -0.236 -0.207
## Seattle       1.002  0.337 -0.141  0.704 -0.118
## Washington   -0.251  0.738  0.877 -0.049  1.179
```

```
# Scores sorted by PC1
o<-order(scores_1.cor[,1])
scores.pc1.cor<-scores_1.cor[o,]
scores.pc1.cor
```

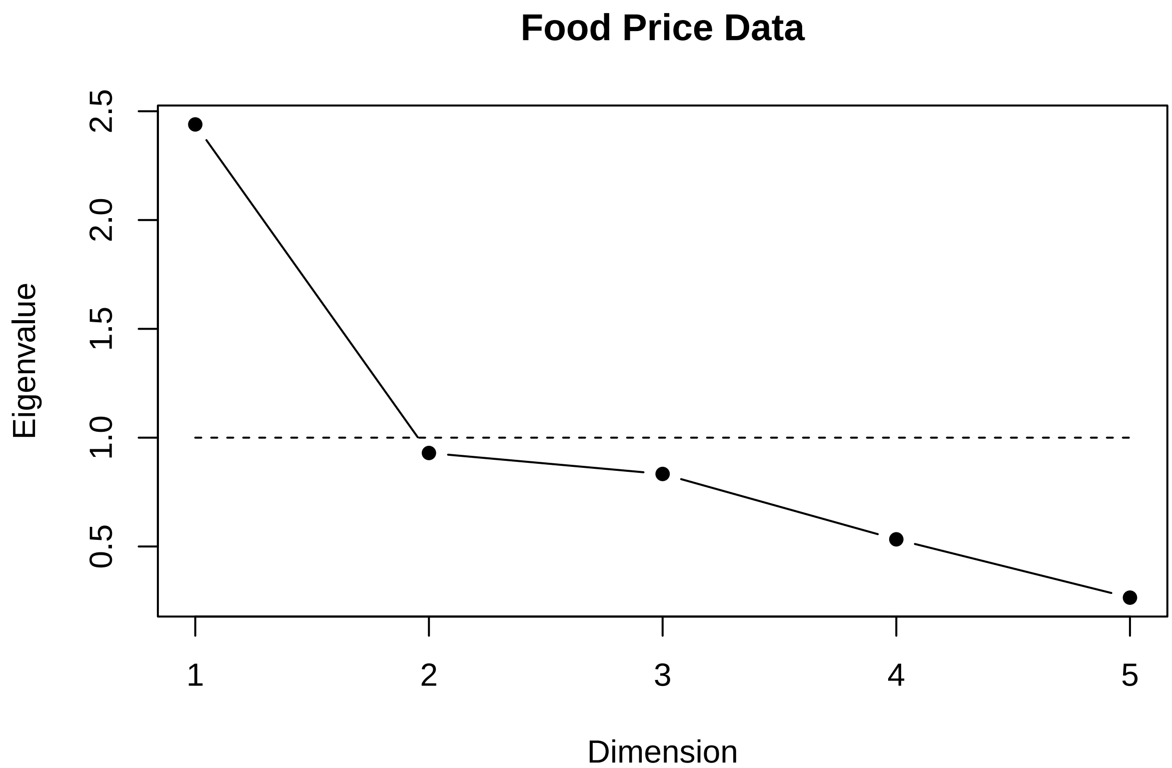
```
##                 PC1    PC2    PC3    PC4    PC5
## Anchorage    -4.576  0.529 -1.194 -1.011 -0.922
## Honolulu     -2.796  0.910 -0.389 -0.208  0.472
## Philadelphia -2.101 -0.796  0.510  0.355  0.185
## NewYork      -1.289 -0.919 -0.357 -0.353  0.792
## Boston       -0.574 -1.618 -0.316 -0.063 -0.383
## Chicago      -0.508  1.293  0.382  0.970 -0.327
## Dallas       -0.382  0.924 -0.217  0.849  0.432
## Baltimore    -0.358 -1.102  1.733  1.321 -0.255
## Atlanta      -0.317 -0.080  0.343  0.734 -0.236
## Cincinnati   -0.265 -0.787  0.400 -0.316  0.479
## Washington   -0.251  0.738  0.877 -0.049  1.179
## Detroit       0.198 -1.531  0.005  0.018 -0.890
## Pittsburgh    0.320 -1.084 -0.331  0.192 -0.017
## KansasCity    0.419 -0.395  2.111 -1.271 -0.015
## Houston       0.462  1.341  0.754 -0.238  0.117
## SanFrancisco  0.713  1.320 -0.591 -0.236 -0.207
## Minneapolis   0.728 -0.603 -0.622  0.608  0.428
## Cleveland     0.737  0.275 -0.203 -0.577  0.039
## Seattle       1.002  0.337 -0.141  0.704 -0.118
## St.Louis      1.253  0.021 -0.552 -0.851  0.408
## Milwaukee     1.284  0.544 -1.521  1.380 -0.245
## LosAngeles    1.546  1.173 -0.099 -0.571 -0.359
## SanDiego      2.123  0.679  1.146 -0.643 -0.836
## Buffalo       2.635 -1.169 -1.727 -0.744  0.279
```

```
### Correlation between the standardized variables and each sample PC
cor(data,scores_1.cor)
```

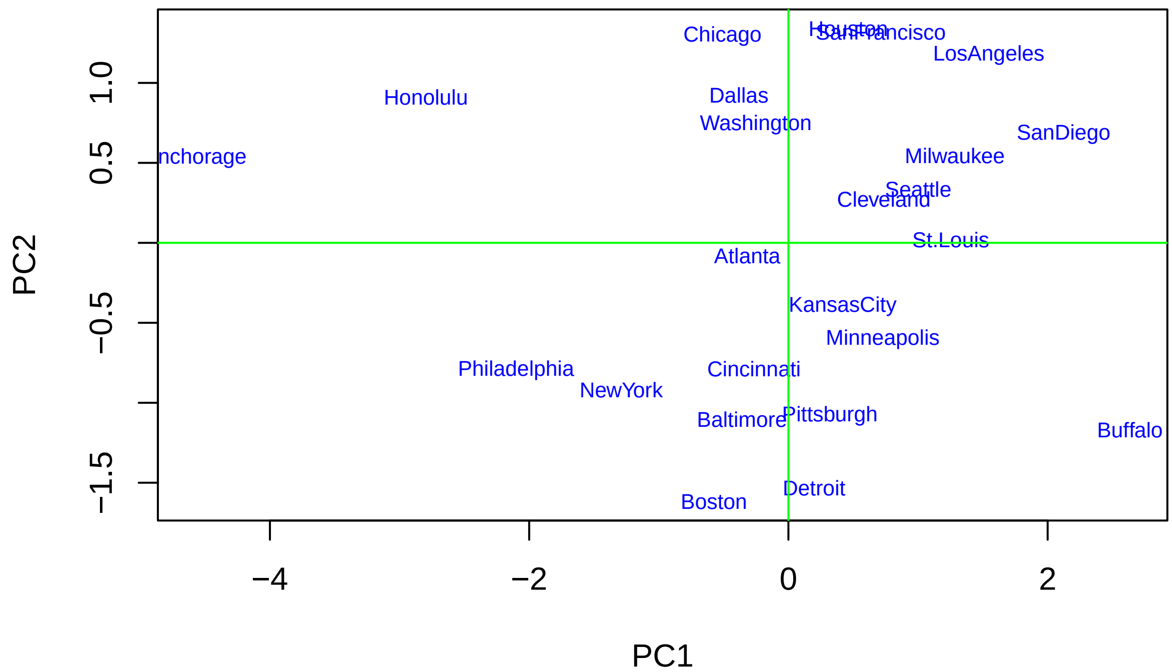
```
##                  PC1         PC2         PC3         PC4         PC5
## Bread     -0.7964682  0.05450718 -0.36670048 -0.38838458 -0.27812531
## Hamburger -0.8122849 -0.26763260 -0.37191678  0.05212669  0.35731280
## Butter    -0.6205082  0.09579836  0.70148851 -0.31407936  0.12238258
## Apples    -0.4543785  0.84534267 -0.06518677  0.27196235  0.02707546
## Tomatoes  -0.7441080 -0.36223093  0.25320137  0.45464664 -0.21025268
```

##Scree plot and Scores plot

```
scree.plot(data,title="Food Price Data",type="R") # correlation matrix
```




```
plot(scores_1.cor[,1], scores_1.cor[,2], xlab="PC1", ylab="PC2",type="n", xlim=c(min(scores_1.cor[,1]),
max(scores_1.cor[,1])),ylim=c(min(scores_1.cor[,2]),
max(scores_1.cor[,2])))
text(scores_1.cor[,1],scores_1.cor[,2], rownames(scores_1.cor),
col="blue",cex=0.7)
abline(h=mean(scores_1.cor[,2]),col="green")
abline(v=mean(scores_1.cor[,1]),col="green")
```



PCA with data without Anchorage - prcomp

```
data.new<-data[-1,]
food.pca_1.new <- prcomp(data.new)
food.pca_1.new
```

```
## Standard deviations (1, .., p=5):
## [1] 12.403356  9.092870  7.822538  5.503907  3.534388
##
## Rotation (n x k) = (5 x 5):
##                  PC1         PC2         PC3        PC4         PC5
## Bread     -0.2375126  0.06580119 -0.06573635 -0.3539556 -0.89980672
## Hamburger -0.7801241  0.47037147 -0.12698583 -0.2093723  0.33195615
## Butter    -0.3820564 -0.77107550  0.37577114 -0.3139027  0.14048718
## Apples    -0.1330537 -0.41675693 -0.88972015  0.1290422  0.01888248
## Tomatoes  -0.4139127 -0.07859468  0.21621156  0.8459858 -0.24507098
```

```
summary(food.pca_1.new)
```

```
## Importance of components:
##                            PC1    PC2    PC3     PC4     PC5
## Standard deviation     12.4034 9.0929 7.8225 5.50391 3.53439
## Proportion of Variance  0.4518 0.2428 0.1797 0.08897 0.03669
## Cumulative Proportion   0.4518 0.6946 0.8743 0.96331 1.00000
```

```
food.pca_1.new$sdev
```

```
## [1] 12.403356  9.092870  7.822538  5.503907  3.534388
```

```r
food.pca_1.new$rotation
```

```
##                 PC1         PC2         PC3        PC4         PC5
## Bread     -0.2375126  0.06580119 -0.06573635 -0.3539556 -0.89980672
## Hamburger -0.7801241  0.47037147 -0.12698583 -0.2093723  0.33195615
## Butter    -0.3820564 -0.77107550  0.37577114 -0.3139027  0.14048718
## Apples    -0.1330537 -0.41675693 -0.88972015  0.1290422  0.01888248
## Tomatoes  -0.4139127 -0.07859468  0.21621156  0.8459858 -0.24507098
```

```r
food.pca_1.new$center
```

```
##     Bread Hamburger    Butter    Apples  Tomatoes
##  37.03043 111.23043 143.74391  51.20870  89.30870
```

```r
food.pca_1.new$scale
```

```
## [1] FALSE
```

```r
scores_1.new<-round(food.pca_1.new$x,3)
rownames(scores_1.new)<-rownames(data.new)
scores_1.new
```

```
##                   PC1     PC2     PC3    PC4    PC5
## Atlanta        -3.359  -1.983  -0.753  5.916 -0.830
## Baltimore      -4.740  -6.922  10.154 13.482  3.735
## Boston         -8.726   9.194   8.463  1.453 -5.071
## Buffalo        16.503  21.374   4.205 -6.237  2.165
## Chicago        -1.612  -9.201 -10.210  6.167 -2.005
## Cincinnati     -7.484   0.941   6.636 -2.763  2.569
## Cleveland       5.453   0.100  -0.553 -4.738 -1.161
## Dallas         -5.571  -1.295 -10.730  2.387  2.922
## Detroit         1.113   6.177  10.399  4.579 -6.592
## Honolulu      -27.070  -3.784 -10.785 -6.876 -5.053
## Houston         5.834 -12.377  -4.514 -2.529  1.025
## KansasCity      4.015 -16.113  15.906 -5.051  0.732
## LosAngeles     16.712  -5.168  -4.610 -3.774 -2.823
## Milwaukee      10.199  12.014 -13.327  7.692  0.280
## Minneapolis     0.625  10.293  -0.835  2.581  4.668
## NewYork       -19.182   7.041   2.819 -5.764  1.339
## Philadelphia -22.598  -1.244   3.424  1.888 -1.642
## Pittsburgh     -1.458   8.572   4.811  1.930  0.046
## St.Louis        7.444   4.895   0.566 -7.782  1.558
## SanDiego       24.367 -12.854   5.397  0.245 -2.827
## SanFrancisco    8.245  -2.091  -9.810 -3.442 -3.559
## Seattle         8.271   1.186  -4.188  4.871  1.662
## Washington     -6.982  -8.755  -2.465 -4.234  8.861
```

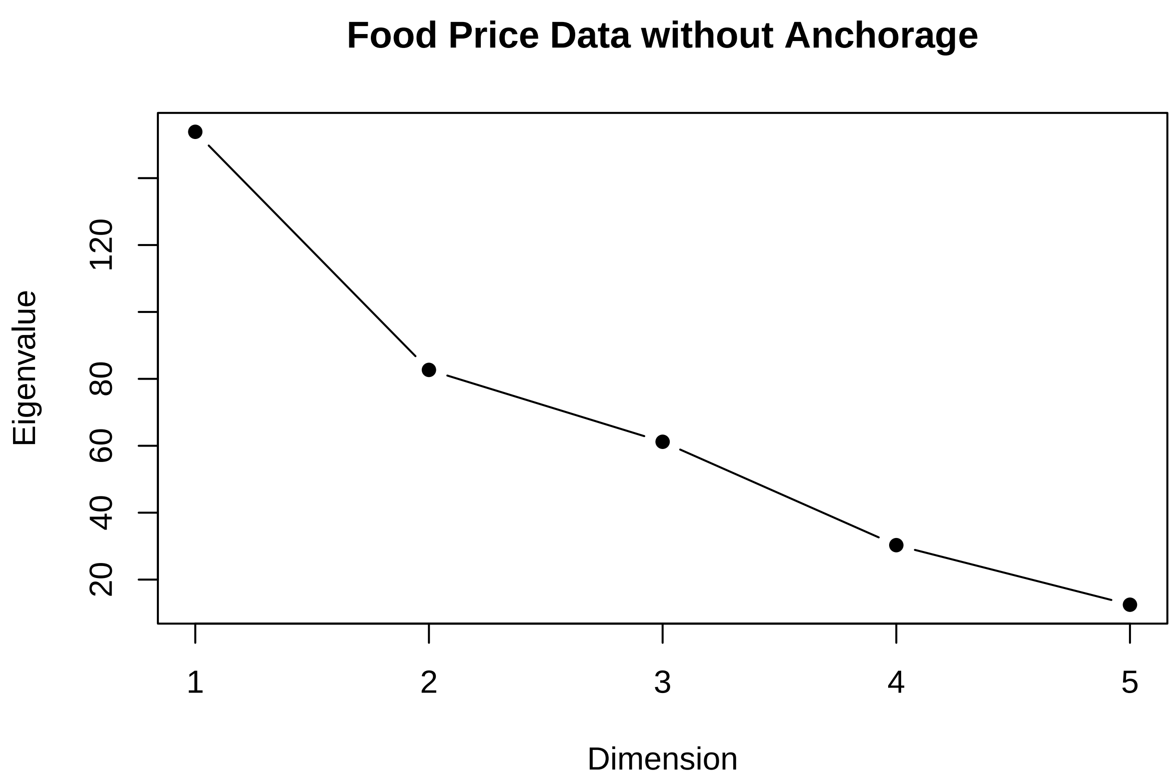
```r
o<-order(scores_1.new[,1]) # Scores sorted by PC1
scores.new.pc1<-scores_1.new[o,]
scores.new.pc1
```

```
##                   PC1     PC2     PC3    PC4    PC5
## Honolulu      -27.070  -3.784 -10.785 -6.876 -5.053
## Philadelphia -22.598  -1.244   3.424  1.888 -1.642
## NewYork       -19.182   7.041   2.819 -5.764  1.339
## Boston         -8.726   9.194   8.463  1.453 -5.071
```

```
## Cincinnati    -7.484   0.941   6.636 -2.763  2.569
## Washington    -6.982  -8.755  -2.465 -4.234  8.861
## Dallas        -5.571  -1.295 -10.730  2.387  2.922
## Baltimore     -4.740  -6.922  10.154 13.482  3.735
## Atlanta       -3.359  -1.983  -0.753  5.916 -0.830
## Chicago       -1.612  -9.201 -10.210  6.167 -2.005
## Pittsburgh    -1.458   8.572   4.811  1.930  0.046
## Minneapolis    0.625  10.293  -0.835  2.581  4.668
## Detroit        1.113   6.177  10.399  4.579 -6.592
## KansasCity     4.015 -16.113  15.906 -5.051  0.732
## Cleveland      5.453   0.100  -0.553 -4.738 -1.161
## Houston        5.834 -12.377  -4.514 -2.529  1.025
## St.Louis       7.444   4.895   0.566 -7.782  1.558
## SanFrancisco   8.245  -2.091  -9.810 -3.442 -3.559
## Seattle        8.271   1.186  -4.188  4.871  1.662
## Milwaukee     10.199  12.014 -13.327  7.692  0.280
## Buffalo       16.503  21.374   4.205 -6.237  2.165
## LosAngeles    16.712  -5.168  -4.610 -3.774 -2.823
## SanDiego      24.367 -12.854   5.397  0.245 -2.827
```

```
# Scree plot  and Scores plot

scree.plot(data.new,title="Food Price Data without Anchorage",type="V")
```




```
plot(scores_1.new[,1], scores_1.new[,2], xlab="PC1", ylab="PC2",type="n", xlim=c(min(scores_1.new[,1]),
    max(scores_1.new[,2])))
    text(scores_1.new[,1],scores_1.new[,2], rownames(scores_1.new), col="blue",cex=0.7)
```

```
abline(h=mean(scores_1.new[,2]),col="green")
abline(v=mean(scores_1.new[,1]),col="green")
```