Principal Component Analysis in R

STATS 503

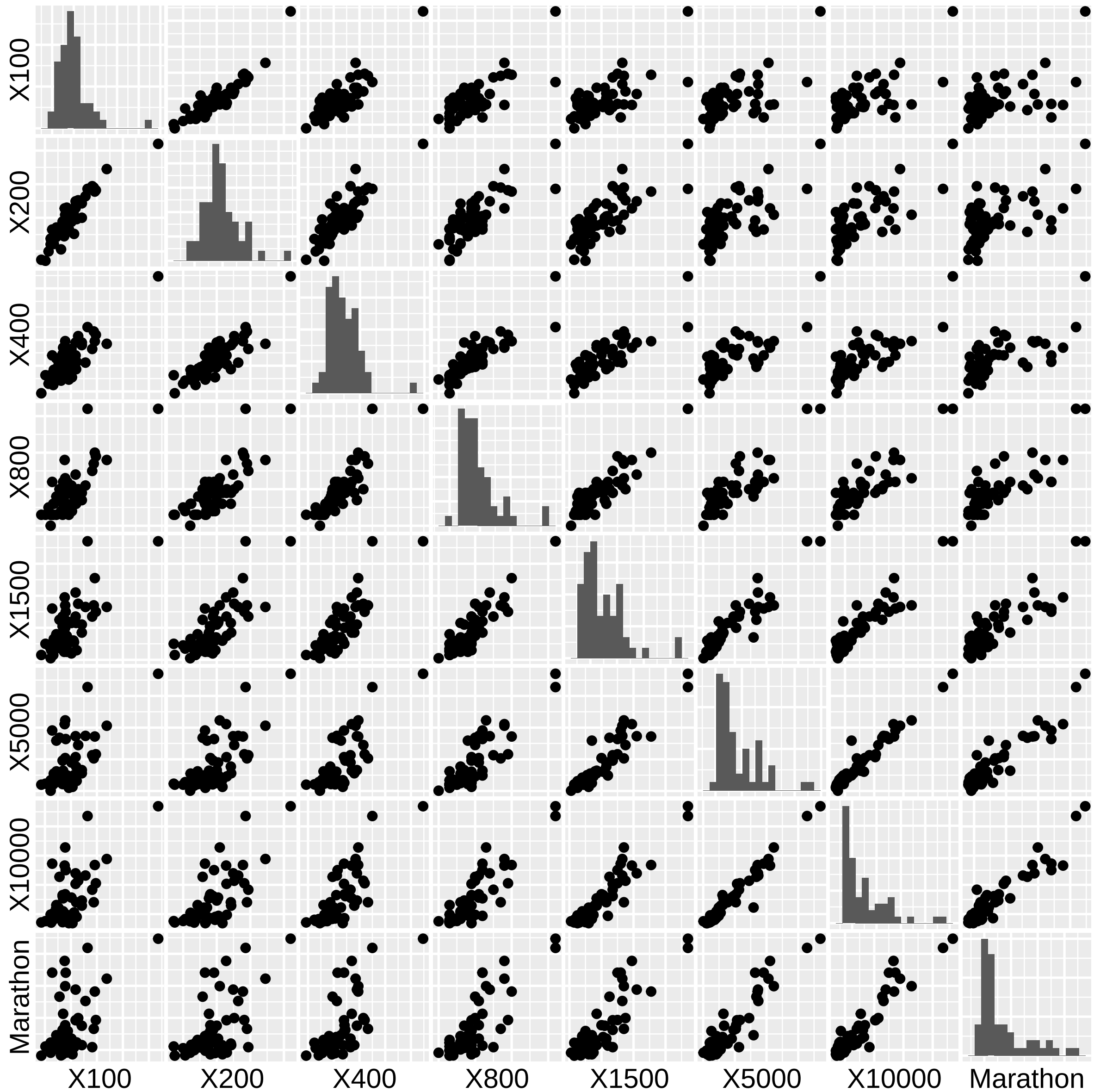
Example: Athletic Performance Data

We start reading the file `athletic.txt`. The first two columns correspond to the code and name of a country. The next columns correspond to the records for each competition.

```
Event <- c("X100", "X200", "X400", "X800", "X1500", "X5000", "X10000", "Marathon")
athletic_data <- read.table("athletic.txt", col.names=c("Code", "Country", Event))
Code <- athletic_data$Code
Country <- athletic_data$Country
numerical.data <- athletic_data[, 3:ncol(athletic_data)]
```

To produce pairwise scatterplots and histograms, we use the function `ggpairs` from the library `GGally`.

```
library(GGally)
ggpairs(numerical.data, axisLabels="none", diag=list(continuous=wrap('barDiag', bins=15)),
  upper=list(continuous="points"),
  lower=list(continuous="points"))
```

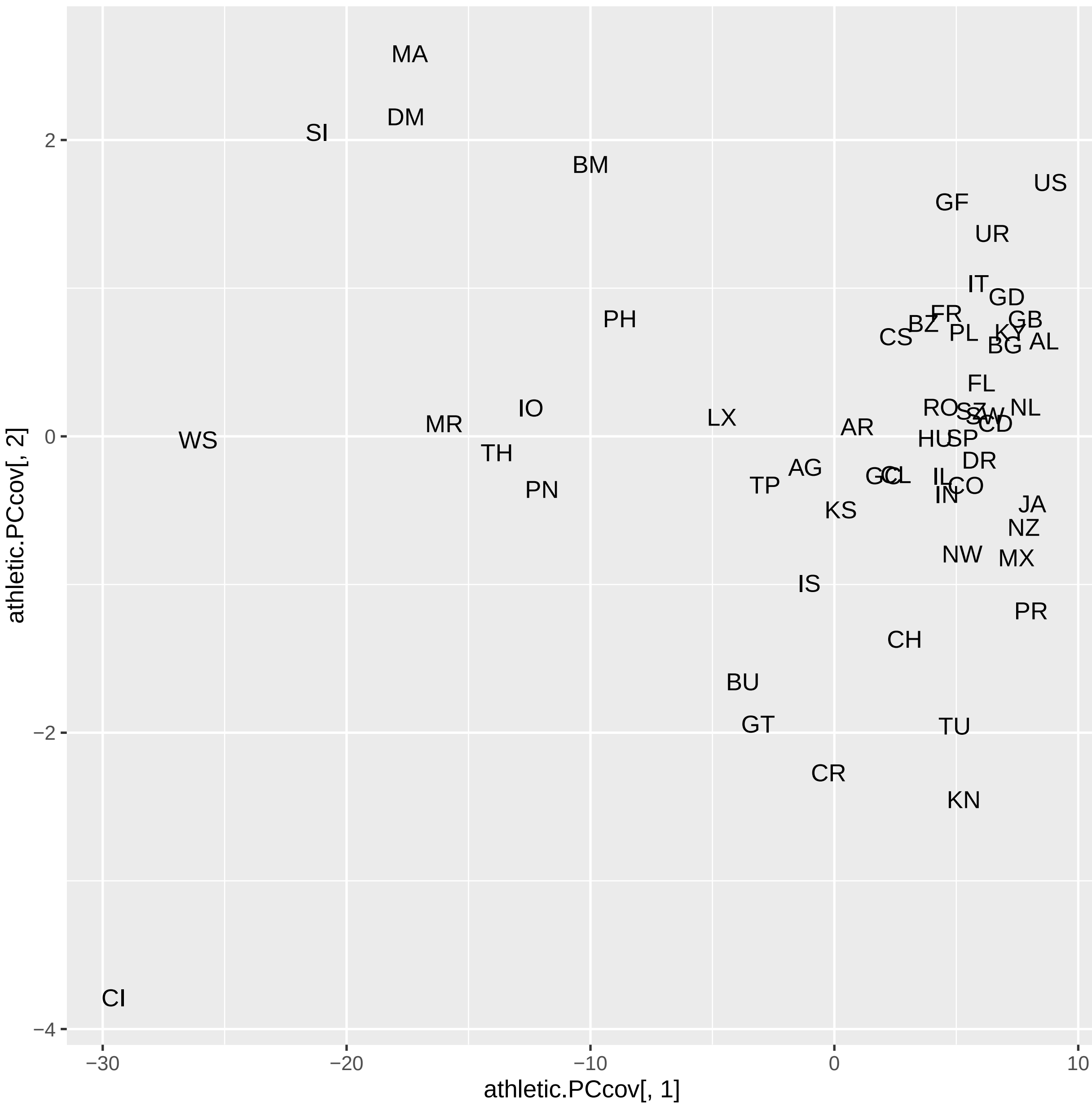


The function **princomp** returns object with principal component analysis of the data set. The argument **cor** decides whether it works with correlation or covariance matrix.

```
athletic.PCAcov <- princomp(numerical.data, cor=F)
loadings(athletic.PCAcov)[,1:2]

##                Comp.1      Comp.2
## X100     -0.019865414 -0.21068624
## X200     -0.041566107 -0.35895599
## X400     -0.110631839 -0.82784910
## X800     -0.005487697 -0.02317370
## X1500    -0.014386824 -0.04465503
## X5000    -0.079308429 -0.12996775
## X10000   -0.181098930 -0.29885158
## Marathon -0.972786963  0.18081152
```

To plot a projection of the athletic events on the space spanned by the first two principal components, we can use the function `qplot`.

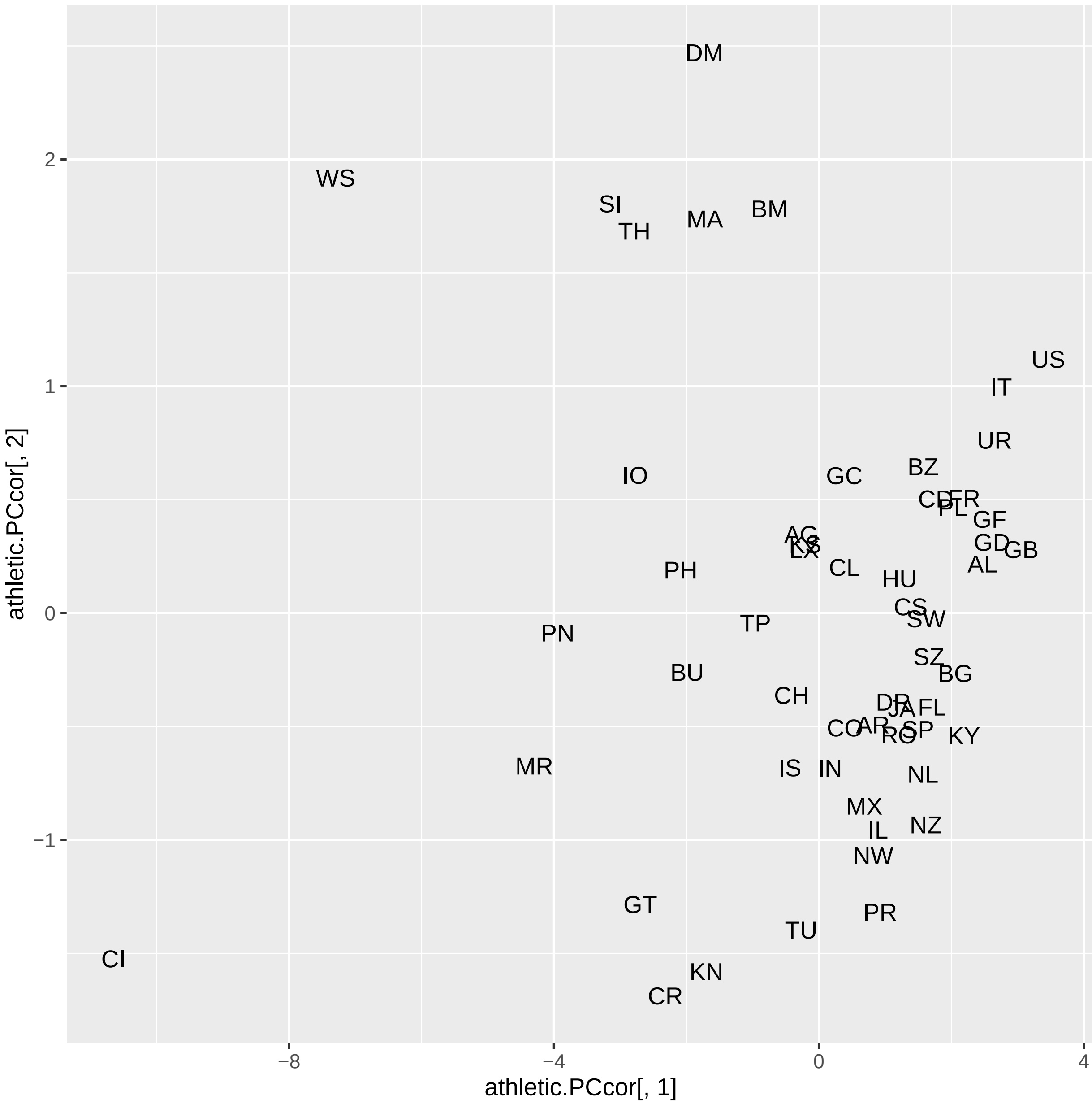
```
library(ggplot2)
athletic.PCcov <- predict(athletic.PCAcov)
qplot(x=athletic.PCcov[,1], y=athletic.PCcov[,2], label=Code, geom="text")
```



Now, we do the same way with the correlation matrix.

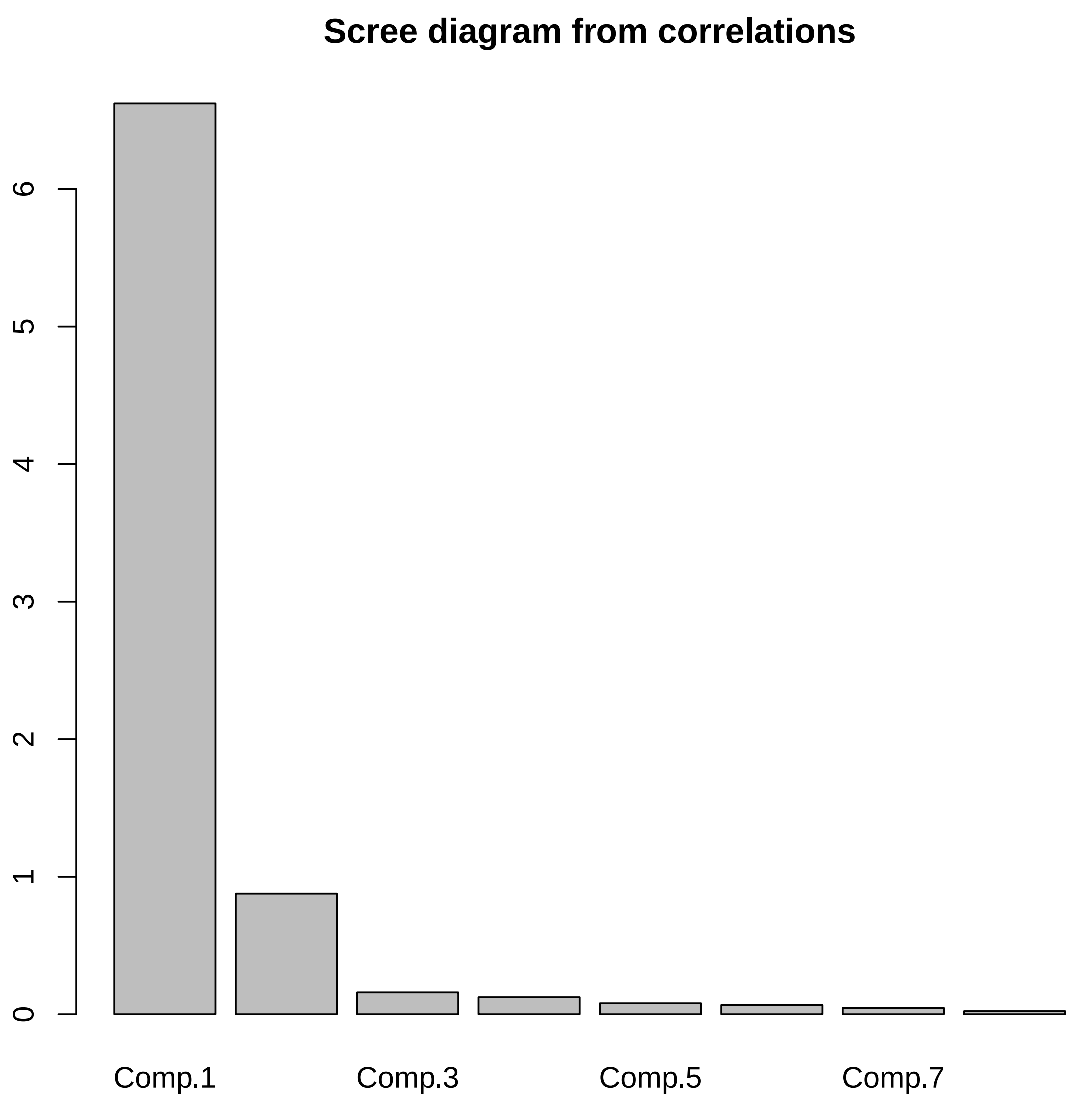
```
athletic.PCAcor <- princomp(numerical.data, cor=T)
loadings(athletic.PCAcor)[,1:2]

##               Comp.1      Comp.2
## X100      -0.3175502 -0.56692566
## X200      -0.3369774 -0.46154846
```

```
## X400     -0.3556427 -0.24831328
## X800     -0.3686786 -0.01239489
## X1500    -0.3728159  0.13973247
## X5000    -0.3643787  0.31201256
## X10000   -0.3667744  0.30686895
## Marathon -0.3419290  0.43898707

athletic.PCcor <- predict(athletic.PCAcor)
qplot(x=athletic.PCcor[,1], y=athletic.PCcor[,2], label=Code, geom="text")
```



Finally, the next code shows how to construct a barplot of the principal components' variance.

```
barplot(athletic.PCAcor$sdev^2)
title("Scree diagram from correlations")
```

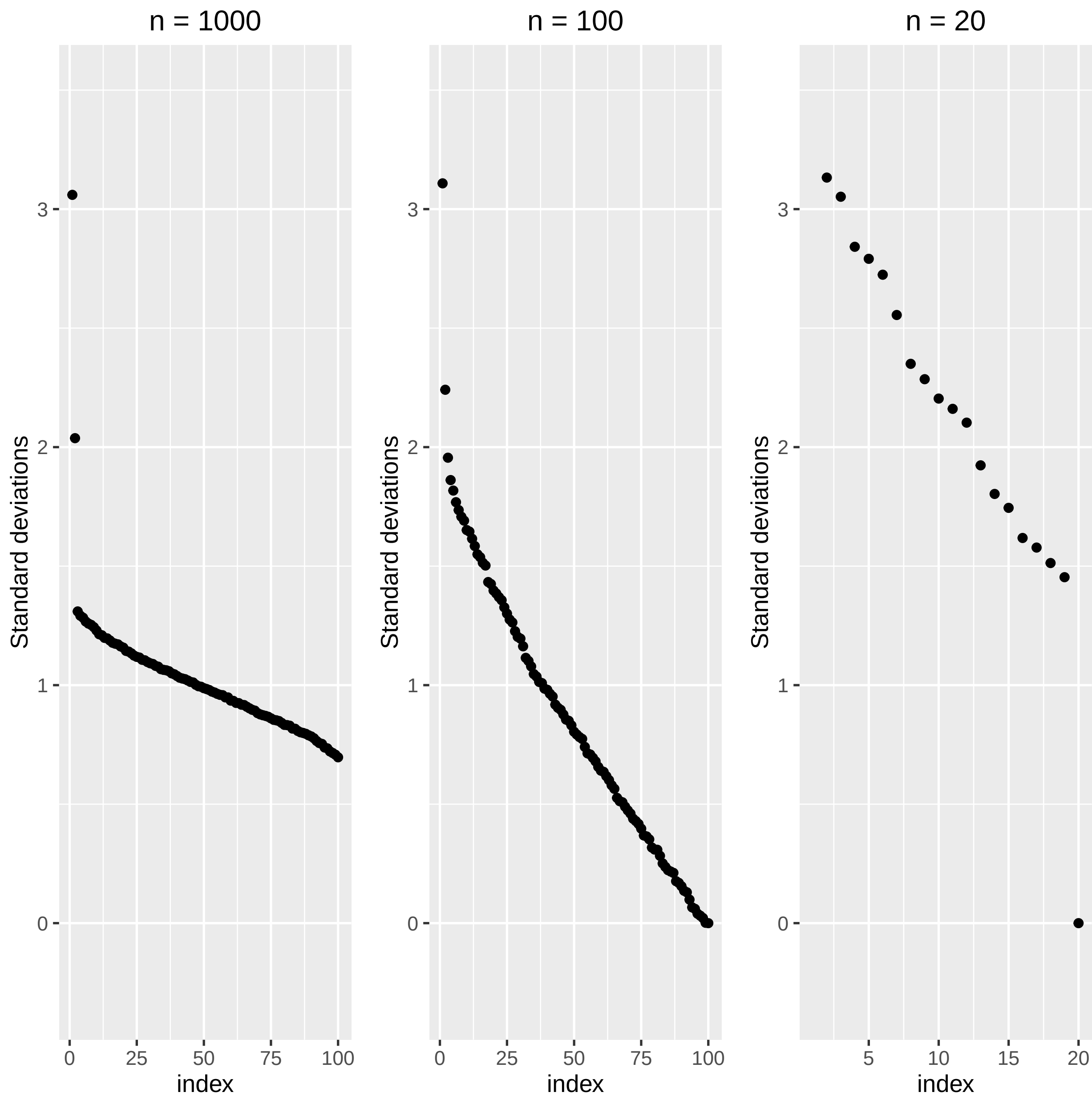


An example in high dimensions

The next code simulates data from a multivariate normal distribution with mean zero and covariance $\Sigma = diag(3^2, 2^2, 1, \ldots, 1)$.

```
scree_plot <- function(n) {
  p <- 100
  X <- sweep(matrix(rnorm(n*p),ncol = p), 2, c(3,2,rep(1,p-2)), "*")
  W <- prcomp(X)
  qplot(y=W$sdev, xlab="index", ylab="Standard deviations", main=paste("n =",n),
        ylim = c(-0.3,3.5))
}
```

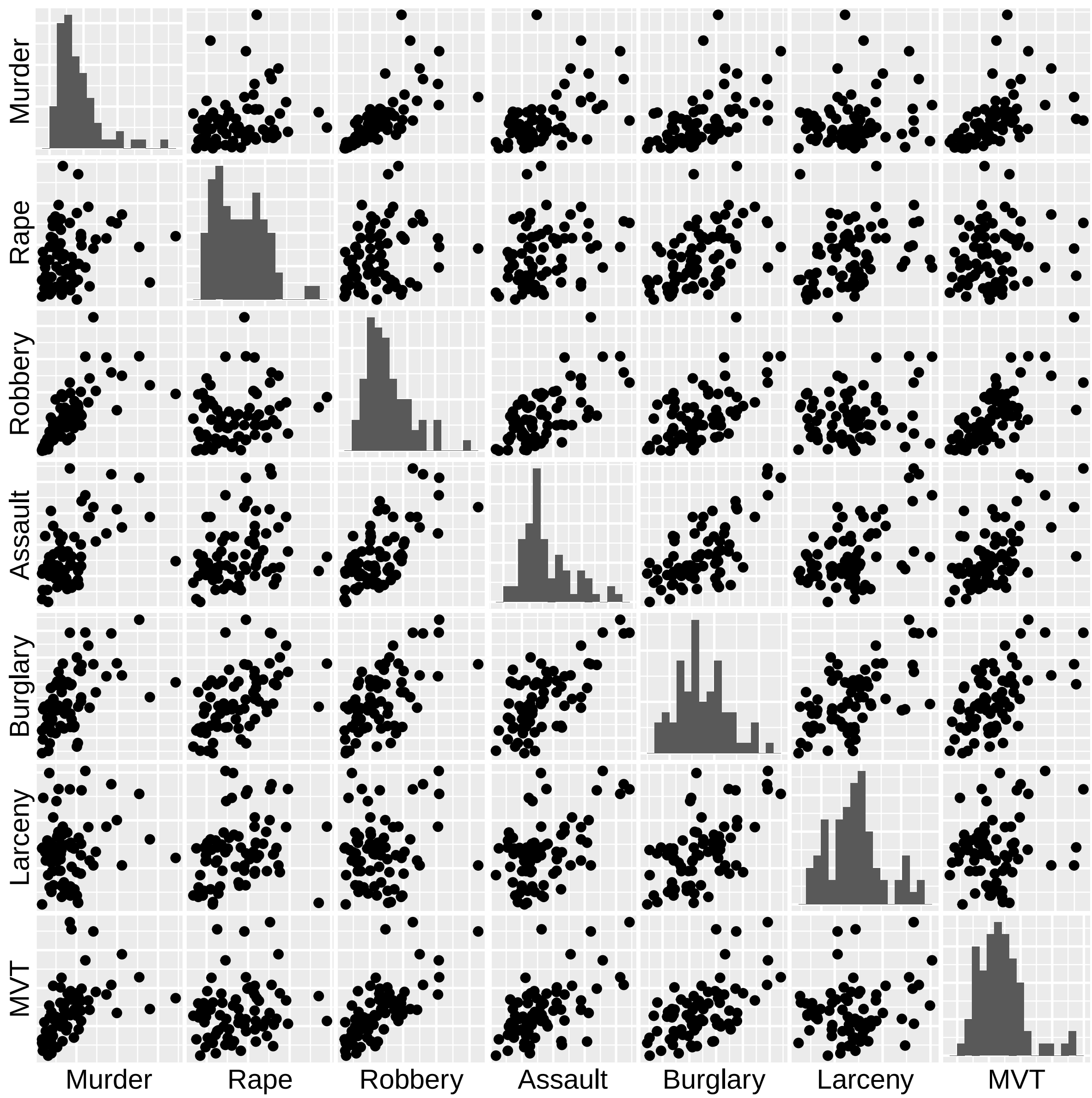
```r
plotlist = lapply(c(1000,100,20),scree_plot)

library(gridExtra)
grid.arrange(plotlist[[1]],plotlist[[2]],plotlist[[3]],ncol=3)

## Warning:  Removed 1 rows containing missing values (geom_point).
```



Example: City crime data

```r
data_crime <- read.table("citycrime.txt", header = TRUE)
ggpairs(data_crime,axisLabels="none", diag=list(continuous=wrap('barDiag', bins=15)),
  upper=list(continuous = "points"),
```

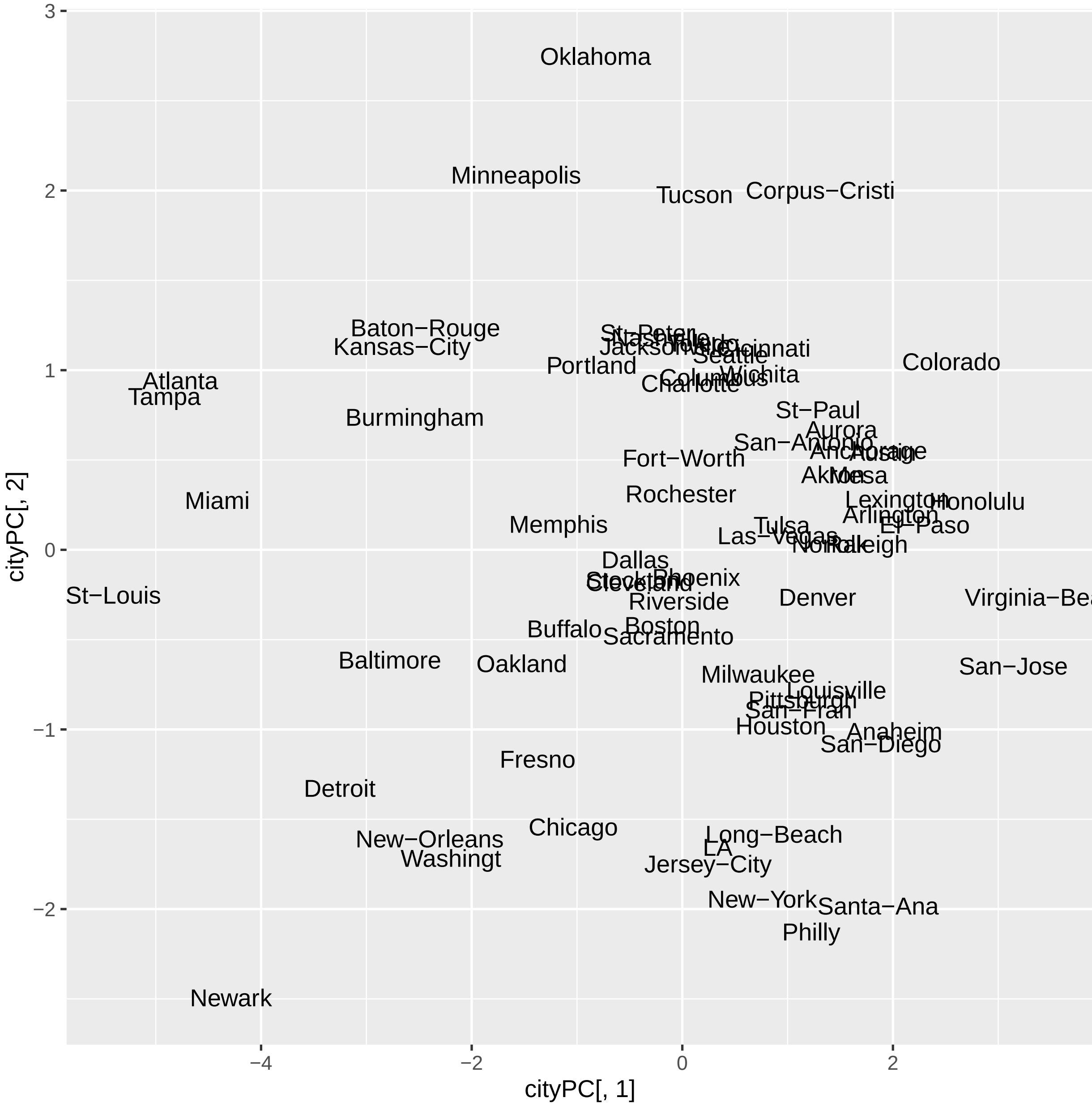
```
  lower=list(continuous = "points"))
```




```
city.PCA <- princomp(data_crime,cor=T)
loadings(city.PCA)[,1:3]

##              Comp.1     Comp.2      Comp.3
## Murder   -0.3703162 -0.3393305  0.20197401
## Rape     -0.2494347  0.4665068  0.78285390
## Robbery  -0.4260252 -0.3878604  0.07906463
## Assault  -0.4340165  0.0424326 -0.28183161
## Burglary -0.4497241  0.2382532  0.01503743
## Larceny  -0.2759217  0.6055442 -0.49241798
## MVT      -0.3903791 -0.3025585 -0.13403135
```

```r
summary(city.PCA)

## Importance of components:
##                           Comp.1    Comp.2    Comp.3     Comp.4     Comp.5
## Standard deviation     1.948024 1.0950164 0.8771129 0.71434047 0.57690506
## Proportion of Variance 0.542114 0.1712944 0.1099039 0.07289747 0.04754564
## Cumulative Proportion  0.542114 0.7134084 0.8233123 0.89620976 0.94375539
##                           Comp.6     Comp.7
## Standard deviation     0.4617666 0.42483396
## Proportion of Variance 0.0304612 0.02578341
## Cumulative Proportion  0.9742166 1.00000000
```

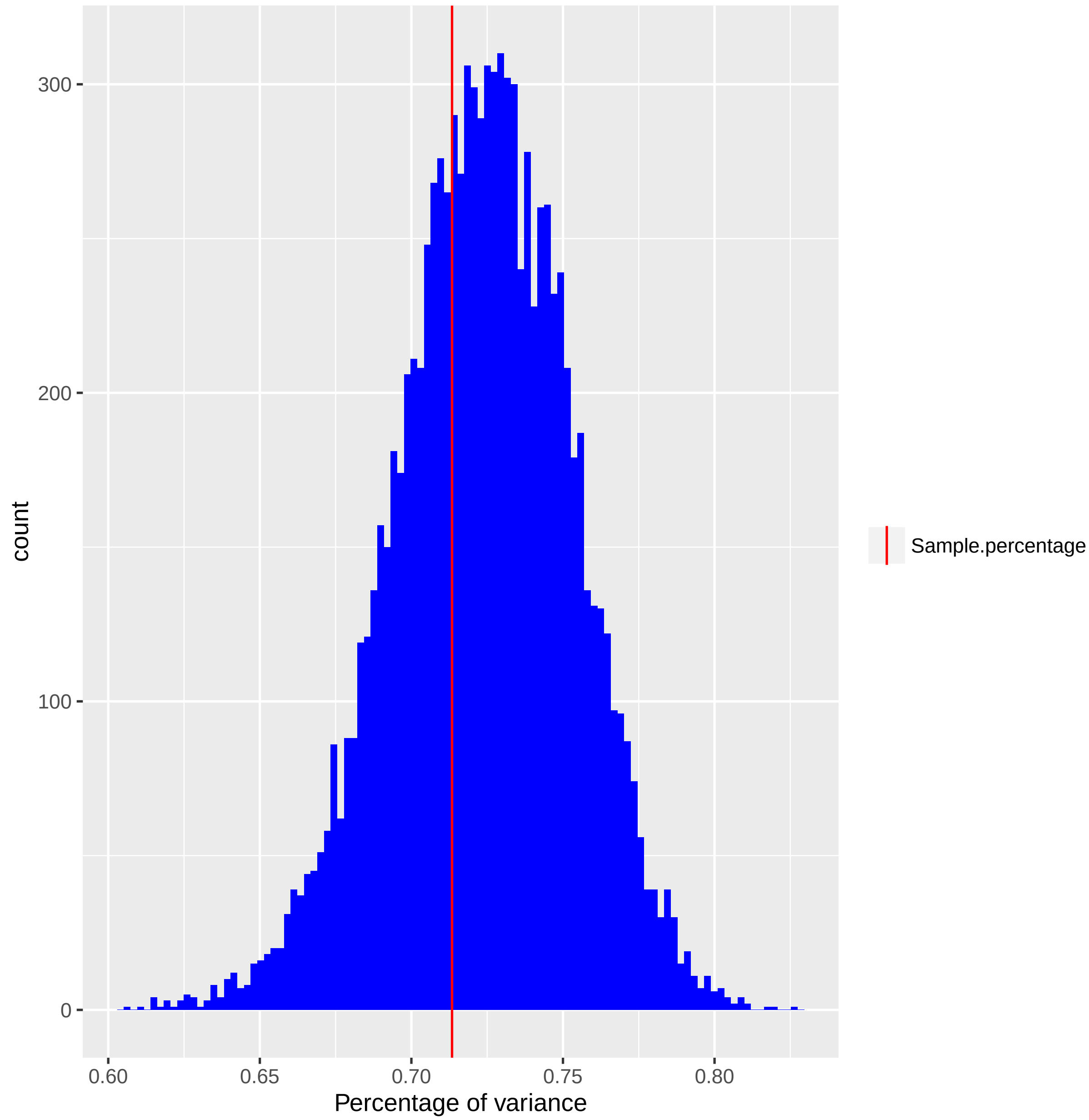
```r
cityPC <- predict(city.PCA)
qplot(x=cityPC[,1], y=cityPC[,2], label=row.names(data_crime), geom="text")
```



The next code generates 10,000 data sets X^* taken from the dataset `data_crime` with replacement. Then, it calculates the percentage of the variance explained by the first two principal components.

```
N <- 10000
M <- length(data_crime[[1]])
data_crime = as.data.frame(data_crime)
theta_samples <- sapply(1:N, function(i){
  X_star <- data_crime[sample(1:length(data_crime[[1]]),size=M,replace=TRUE),]
  u <- summary(princomp(X_star,cor = T))
  return(sum(u[[1]][1:2]^2)/sum(u[[1]]^2))
  })

sample_percentage <- sum(city.PCA$sdev[1:2]^2)/(sum(city.PCA$sdev^2))

qplot(theta_samples, geom="histogram",bins=100, fill=I("blue"),
```

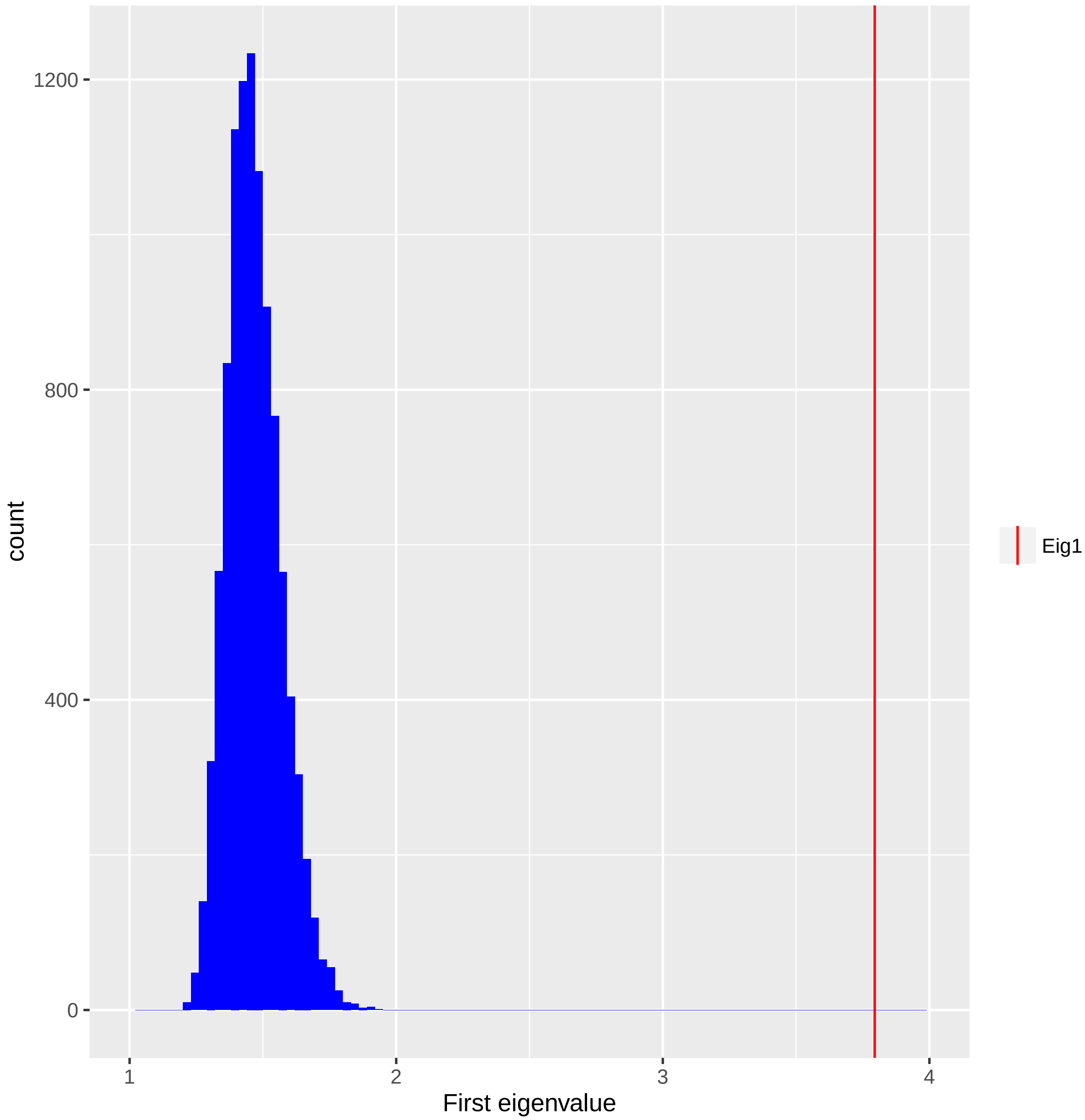
```
    xlab="Percentage of variance") +
  geom_vline(aes(xintercept=sample_percentage,color="Sample.percentage"), show.legend = TRUE)+
  scale_colour_manual(name="", values=c(Sample.percentage="red"))
```



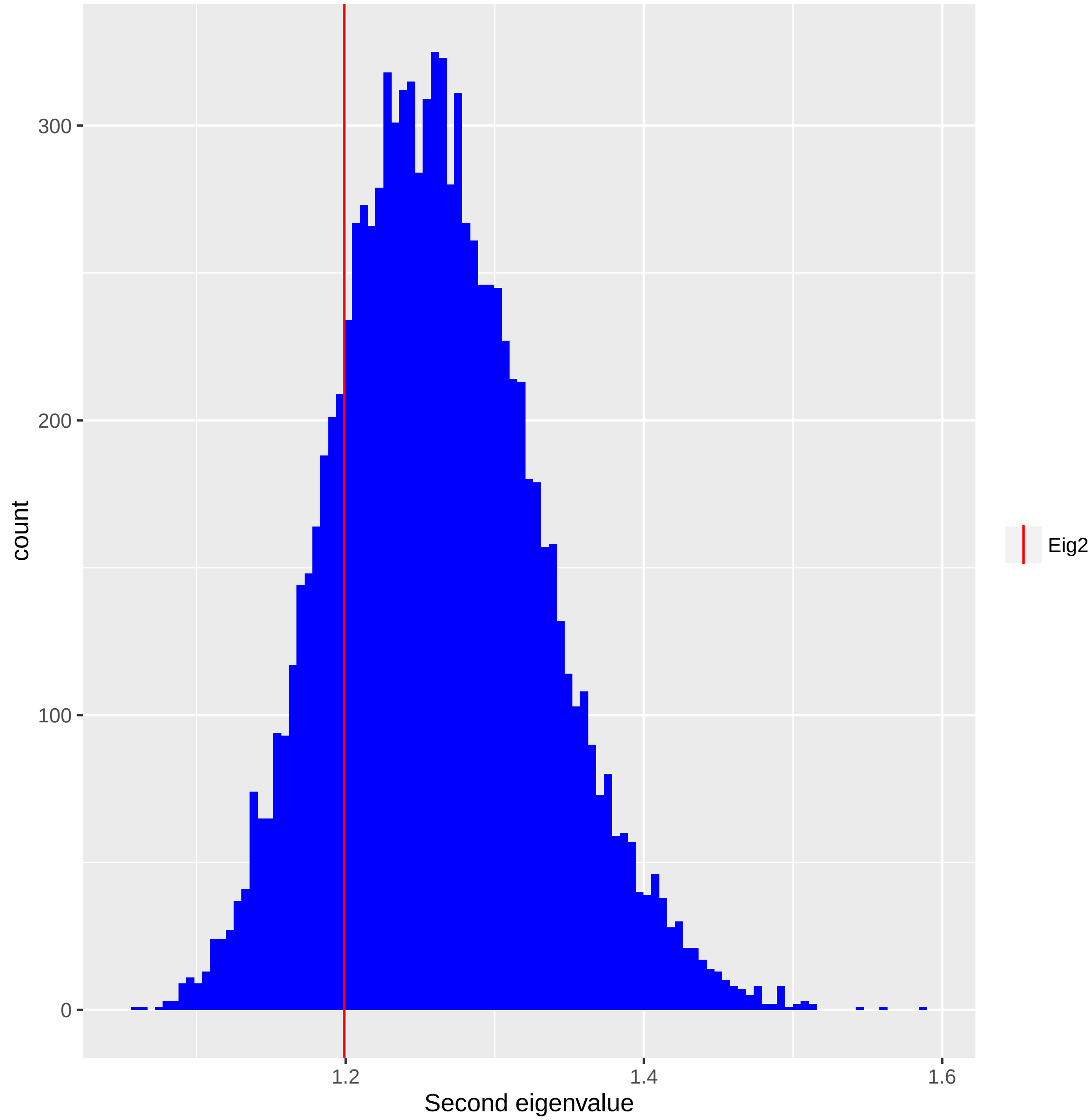

```
quantile(theta_samples,probs = 0.025)

##      2.5%
## 0.6633778

quantile(theta_samples,probs = 0.975)

##     97.5%
## 0.7775128
```

Now, we calculate the variance explained by components 1 and 2 in 10,000 generated data sets.

```r
lambdas <- sapply(1:N, function(i){
  X_permuted <- sapply(data_crime, function(x){return(x[sample(M)])})
  pca <- princomp(X_permuted, cor=T)
  return(pca$sdev^2)})
eig1 <- city.PCA$sdev[1]^2
eig2 <- city.PCA$sdev[2]^2
qplot(lambdas[1,], geom="histogram",bins=100, fill=I("blue"),xlab="First eigenvalue",
      xlim =  c(1,4)) + geom_vline(aes(xintercept=eig1,color="Eig1"), show.legend = TRUE)+
  scale_colour_manual(name="", values=c(Eig1="red"))

## Warning:  Removed 3 rows containing missing values (geom_bar).
```

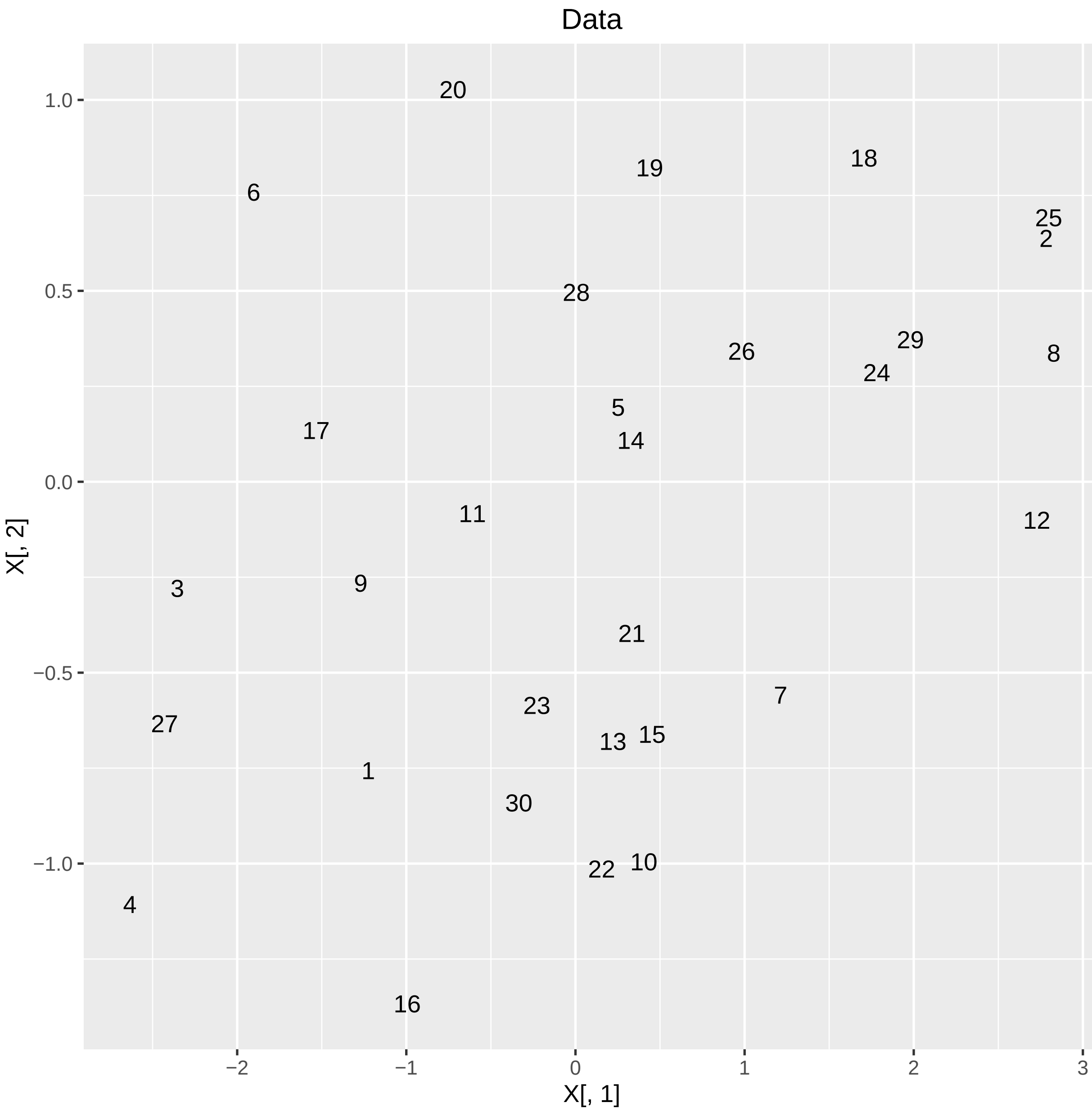


```
qplot(lambdas[2,], geom="histogram",bins=100, fill=I("blue"),xlab="Second eigenvalue") + geom_vline(aes
  scale_colour_manual(name="", values=c(Eig2="red"))
```

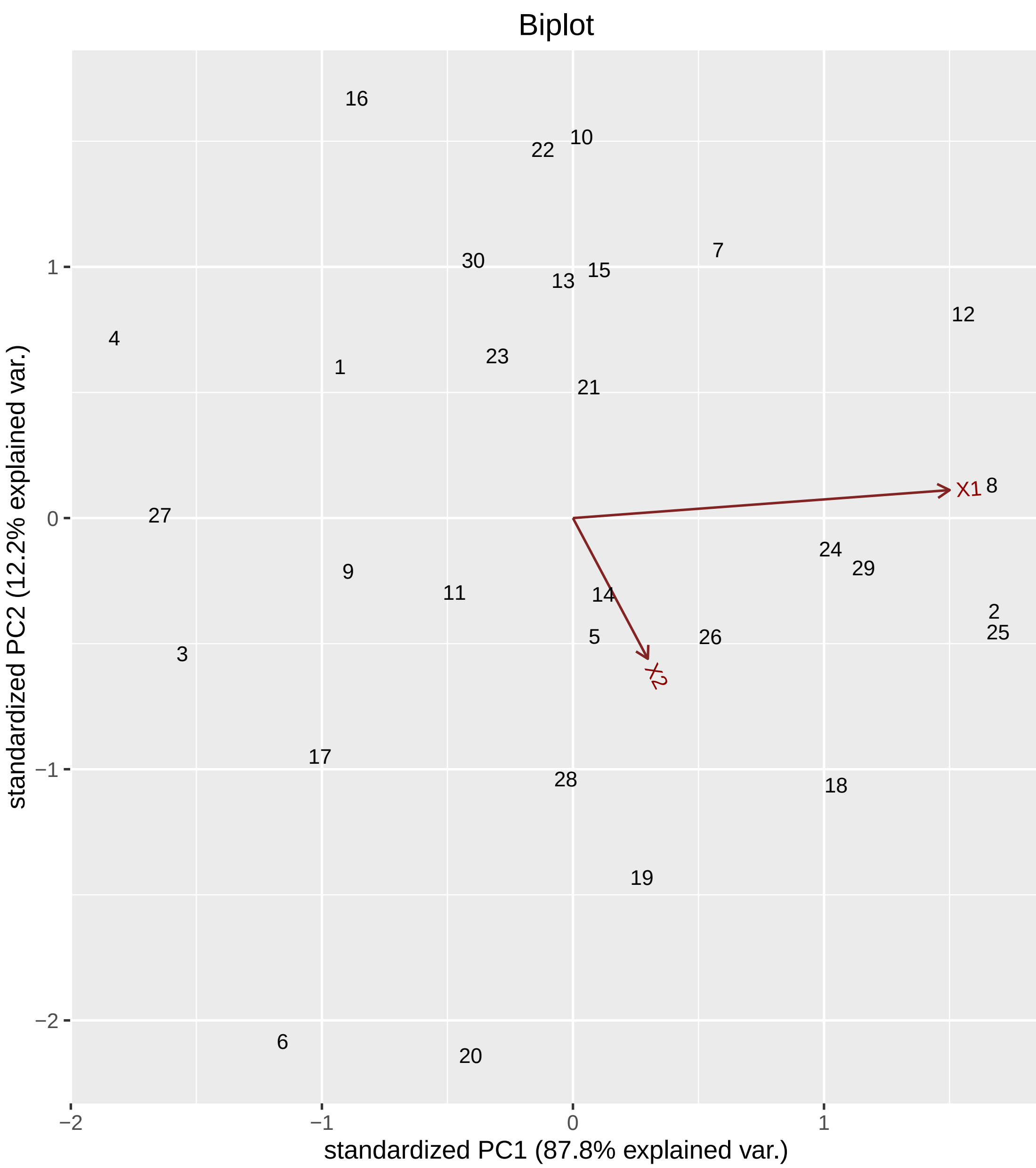


Toy example

```
p <- 2
Sigma <- matrix(c(2, 1, 1, 1), nrow=2)
n <- 30
X <-  data.frame(t(chol(Sigma)%*%matrix(rnorm(p*n),nrow=p)))
pca <- prcomp(X)

qplot(x=X[,1], y=X[,2], label=as.character(1:n), geom="text", main="Data")
```




```r
#library(devtools)
#install_github("vqv/ggbiplot")
library(ggbiplot)

## Loading required package:  plyr
## Loading required package:  scales
## Loading required package:  grid

ggbiplot(pcobj=pca, labels=as.character(1:n)) + ggtitle("Biplot")
```



Example: Applicant's qualifications

```
applicants <- read.table("applicants_qualifications.txt", header=TRUE)
applicantsPCA <- princomp(applicants, cor=T)
loadings(applicantsPCA)[,1:4]
```

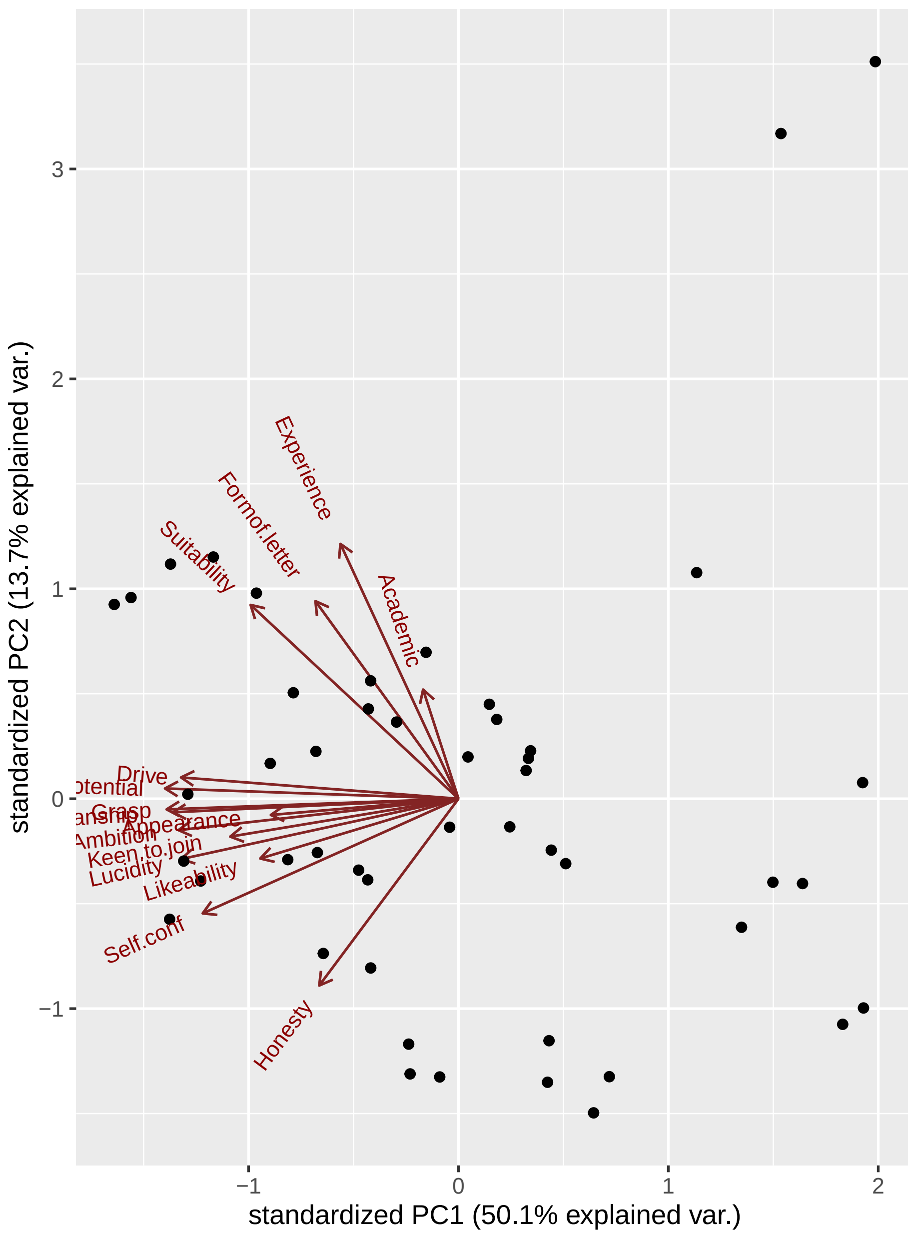
```
##                     Comp.1      Comp.2      Comp.3      Comp.4
## Formof.letter -0.16244045  0.42884630 -0.31537452  0.09434720
## Appearance    -0.21310790 -0.03526587  0.02287769 -0.26217502
## Academic      -0.04018392  0.23691912  0.43046955 -0.63627435
## Likeability   -0.22507847 -0.12979571 -0.46582469 -0.34537465
## Self.conf     -0.29048107 -0.24889621  0.24102627  0.17280371
```

```
## Lucidity      -0.31487036 -0.13099029  0.15003717  0.07103303
## Honesty       -0.15811663 -0.40544998 -0.28392797 -0.41649052
## Salesmanship  -0.32425606 -0.02949230  0.18597471  0.19822750
## Experience    -0.13406824  0.55313856 -0.08259060 -0.06775209
## Drive         -0.31507146  0.04624321  0.07963541  0.15598748
## Ambition      -0.31802398 -0.06815490  0.20865130  0.19929112
## Grasp         -0.33149687 -0.02315034  0.11714220 -0.07472588
## Potential     -0.33328897  0.02225729  0.07254382 -0.18814004
## Keen.to.join  -0.25920844 -0.08227158 -0.46720556  0.20137601
## Suitability   -0.23603667  0.42066207 -0.08915180  0.01991326
```

```r
summary(applicantsPCA)
```

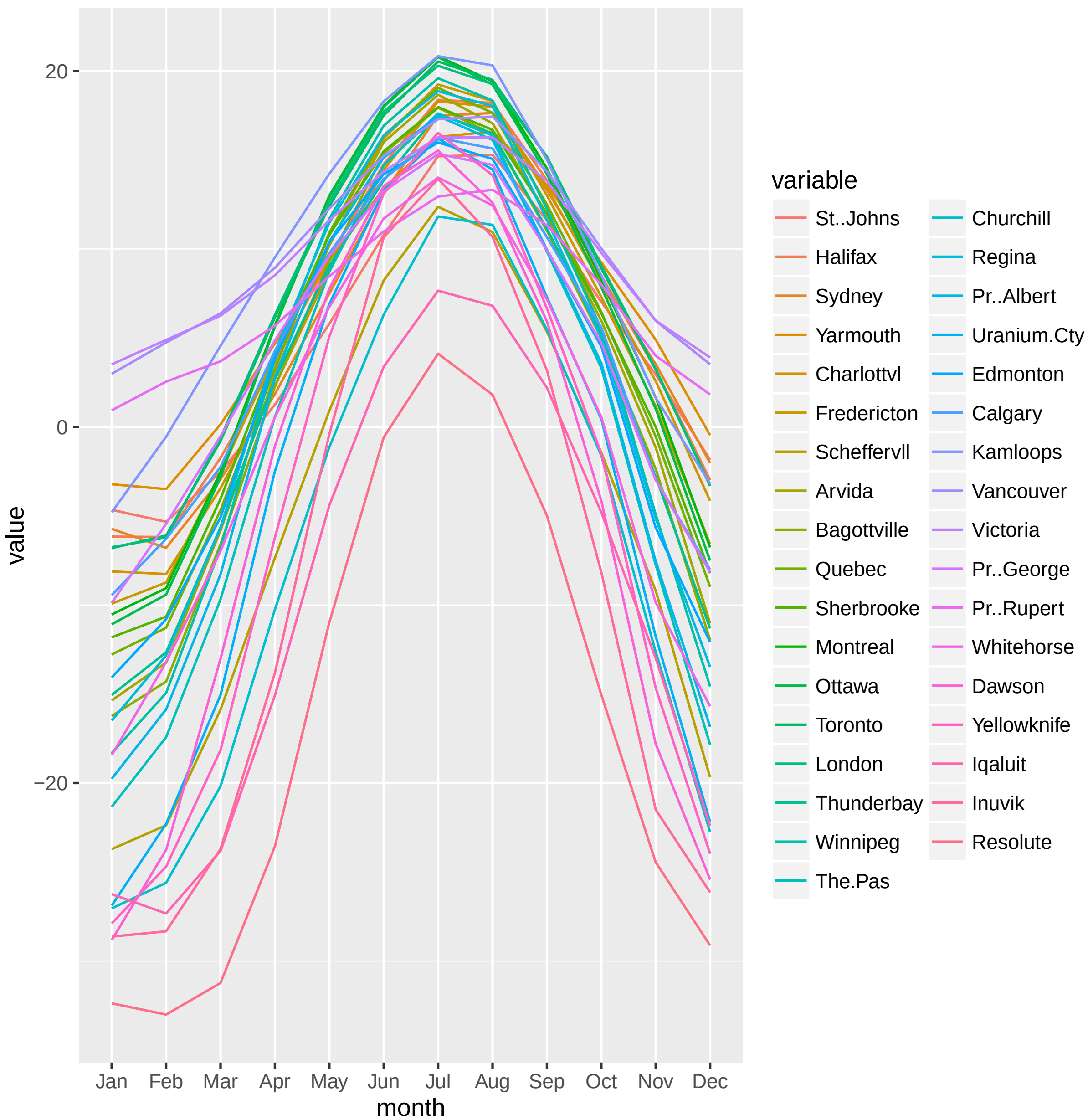
```
## Importance of components:
##                           Comp.1    Comp.2     Comp.3     Comp.4
## Standard deviation     2.7411301 1.4339809 1.20657345 1.09448513
## Proportion of Variance 0.5009196 0.1370867 0.09705463 0.07985985
## Cumulative Proportion  0.5009196 0.6380064 0.73506099 0.81492084
##                            Comp.5     Comp.6     Comp.7     Comp.8
## Standard deviation     0.85973985 0.70326316 0.59267346 0.55668844
## Proportion of Variance 0.04927684 0.03297194 0.02341746 0.02066013
## Cumulative Proportion  0.86419768 0.89716961 0.92058707 0.94124720
##                            Comp.9    Comp.10    Comp.11     Comp.12
## Standard deviation     0.50691374 0.43001206 0.39074335 0.312350893
## Proportion of Variance 0.01713077 0.01232736 0.01017869 0.006504205
## Cumulative Proportion  0.95837797 0.97070533 0.98088402 0.987388228
##                            Comp.13     Comp.14     Comp.15
## Standard deviation     0.298024834 0.254230665 0.189009390
## Proportion of Variance 0.005921253 0.004308882 0.002381637
## Cumulative Proportion  0.993309481 0.997618363 1.000000000
```

```r
ggbiplot(applicantsPCA)
```



Example: Canadian temperatures

```
canadian_weather <- data.frame(t(read.table("canadian_weather.txt", header=TRUE)))
canadian_weather$month = factor(row.names(canadian_weather))
library(reshape2)
meltdf <- melt(canadian_weather,"month")
meltdf$month = factor(meltdf$month,levels=row.names(canadian_weather))
ggplot(meltdf,aes(x=month,y=value,group=variable,color=variable)) + geom_line()
```



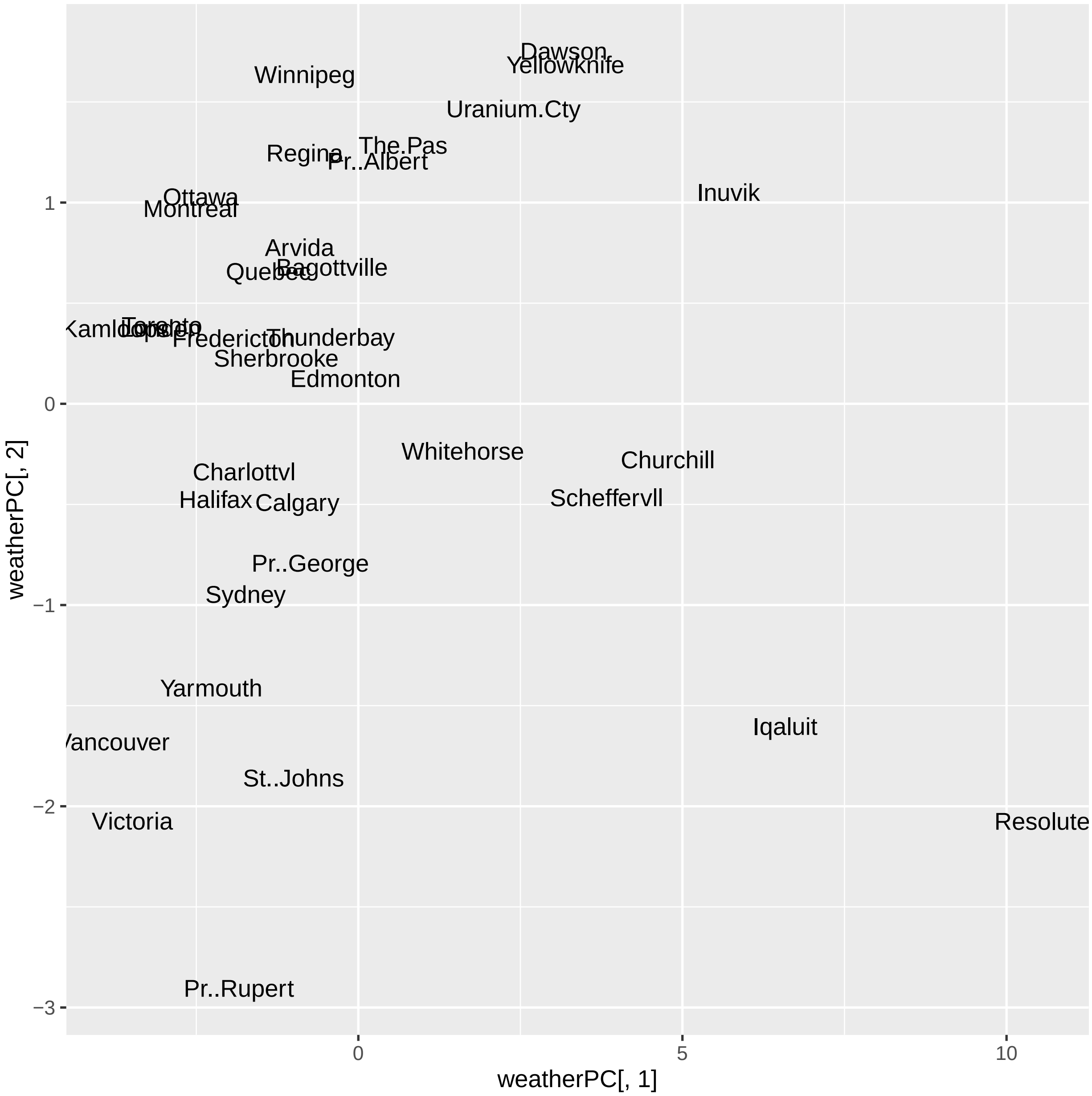

```
weatherPCA <- princomp(t(canadian_weather[,1:35]), cor=T)
summary(weatherPCA)

## Importance of components:
##                          Comp.1    Comp.2     Comp.3      Comp.4
## Standard deviation     3.195162 1.2076746 0.47997719 0.261666684
## Proportion of Variance 0.850755 0.1215398 0.01919818 0.005705788
## Cumulative Proportion  0.850755 0.9722949 0.99149305 0.997198833
##                             Comp.5       Comp.6       Comp.7       Comp.8
## Standard deviation     0.134133729 0.0876024923 0.0629457104 0.0402357196
## Proportion of Variance 0.001499321 0.0006395164 0.0003301802 0.0001349094
## Cumulative Proportion  0.998698154 0.9993376709 0.9996678511 0.9998027605
##                              Comp.9      Comp.10      Comp.11      Comp.12
## Standard deviation     3.227973e-02 2.744603e-02 1.877478e-02 1.480256e-02
```

```
## Proportion of Variance 8.683175e-05 6.277372e-05 2.937436e-05 1.825966e-05
## Cumulative Proportion  9.998896e-01 9.999524e-01 9.999817e-01 1.000000e+00
```

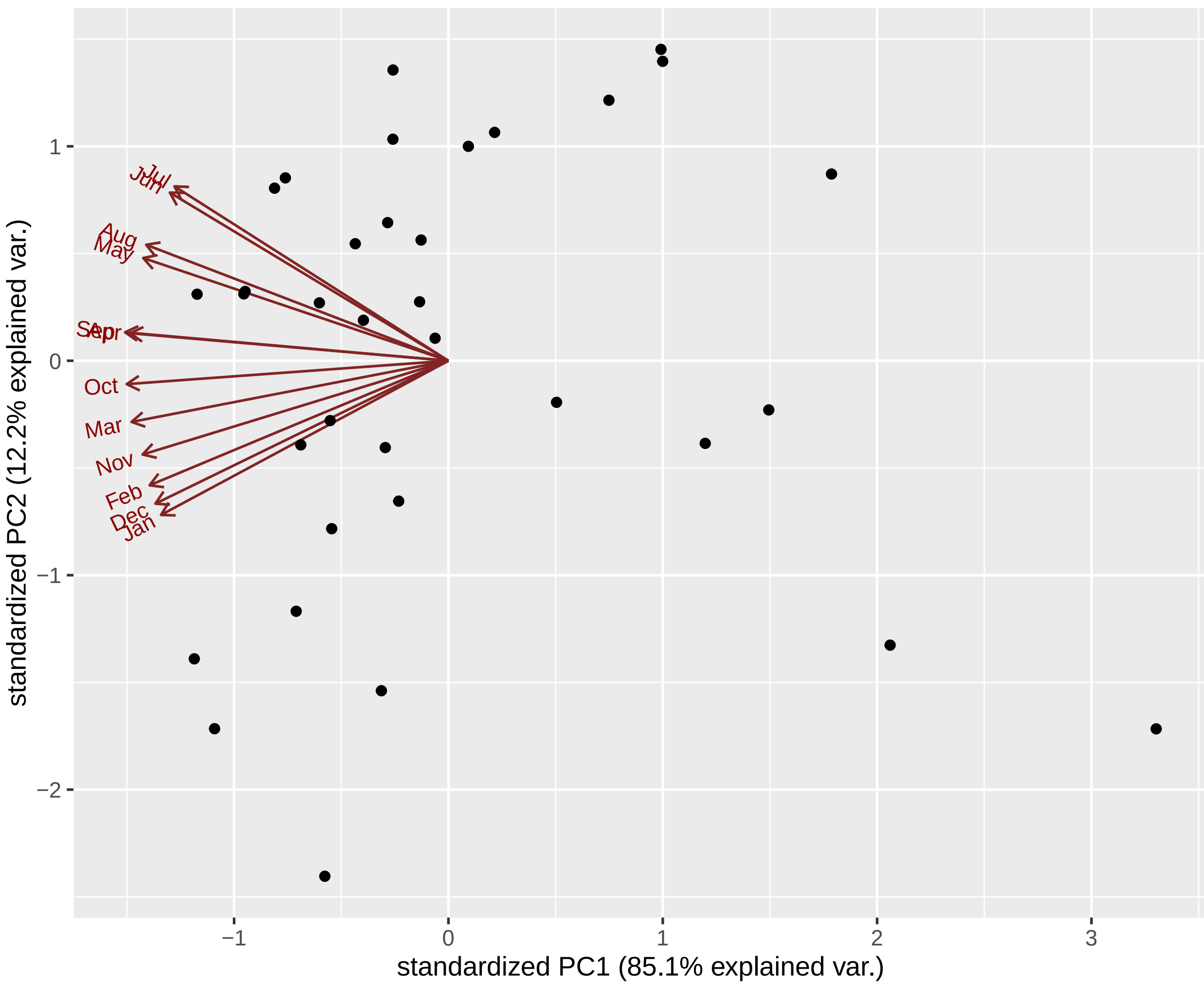
```r
loadings(weatherPCA)[,1:3]
```

```
##         Comp.1      Comp.2      Comp.3
## Jan -0.2742230 -0.38903639  0.00253309
## Feb -0.2851561 -0.31426293 -0.29287615
## Mar -0.3023976 -0.15439433 -0.35285691
## Apr -0.3036991  0.06946101 -0.42350861
## May -0.2912423  0.25922541 -0.37012616
## Jun -0.2658215  0.42435087 -0.13737141
## Jul -0.2614086  0.44005199  0.22378267
## Aug -0.2884328  0.29264101  0.29102415
## Sep -0.3085248  0.07180214  0.27528430
## Oct -0.3069895 -0.05898804  0.24294377
## Nov -0.2918766 -0.23661894  0.40088871
## Dec -0.2796017 -0.36062087  0.15555290
```

```r
weatherPC <- predict(weatherPCA)
qplot(x=weatherPC[,1], y=weatherPC[,2], label=colnames(canadian_weather)[1:35],
      geom = "text")
```




```
ggbiplot(weatherPCA)
```



Jul
Jun
Aug
May
Sep
Apr
Oct
Mar
Nov
Feb
Dec
Jan
1
0
−1
−2
−1
0
1
2
3
standardized PC1 (85.1% explained var.)
standardized PC2 (12.2% explained var.)