

Singular Value Decomposition

Suppose that X is an $n \times p$ matrix. There exists an $n \times p$ matrix U , a $p \times p$ diagonal matrix D and a $p \times p$ matrix V such that:

1. $X = UDV' = \sum d_i u_i v_i'$, where u_i and v_i are respectively the i th columns of U and V , and d_i is the i th diagonal of D .
2. The matrix U has orthogonal columns, so $U'U = I_p$ and UU' is an $n \times n$ projection on the column space of X .
3. The diagonals $d_1 \geq d_2 \geq \dots \geq d_p \geq 0$ of D are called *singular values*. The number of nonzero diagonals is the rank of X .
4. The matrix V is a $p \times p$ orthogonal matrix, so $VV' = V'V = I_p$.
5. Writing out $X'X$ we get:

$$X'X = [DUV']' [VUD'] = VDU'UDV' = VD^2V'$$

We see immediately that the spectral decomposition of $X'X$ is VD^2V' .

6. The right singular vectors V are the eigenvectors of $X'X$.
7. The singular values are the square roots of the eigenvalues of $X'X$.
8. The left singular vectors U are the nonzero eigenvalues of XX' .

In R, the function `svd` computes the singular value decomposition.

Banknotes

This follows the results in Chapter 9 of Härdle and Simar on principal component analysis. The first example is the banknote data from Flury, B. and Riedwyl, H. (1988). *Multivariate Statistics: A practical approach*. London: Chapman & Hall, and discussed in many books and articles. It has to do with dimensional characteristics of genuine and counterfeit Swiss banknotes. All the variables except Y are lengths, while Y is an indicator of whether or not the bill was counterfeit. We ignore Y except in graphics. We think of the 6 measurements as a $p \times 1$ vector x presumably as a sample from a population with mean μ and covariance matrix Σ . The goal is to describe the banknotes by a few linear combinations/principal components.



```
> data(banknote, package="alr3")
> banknote$Y <- factor(c("0-genuine", "1-counterfeit")[1 + banknote$Y])
> library(psych)
> describe(banknote[, -7], ranges=FALSE)
```

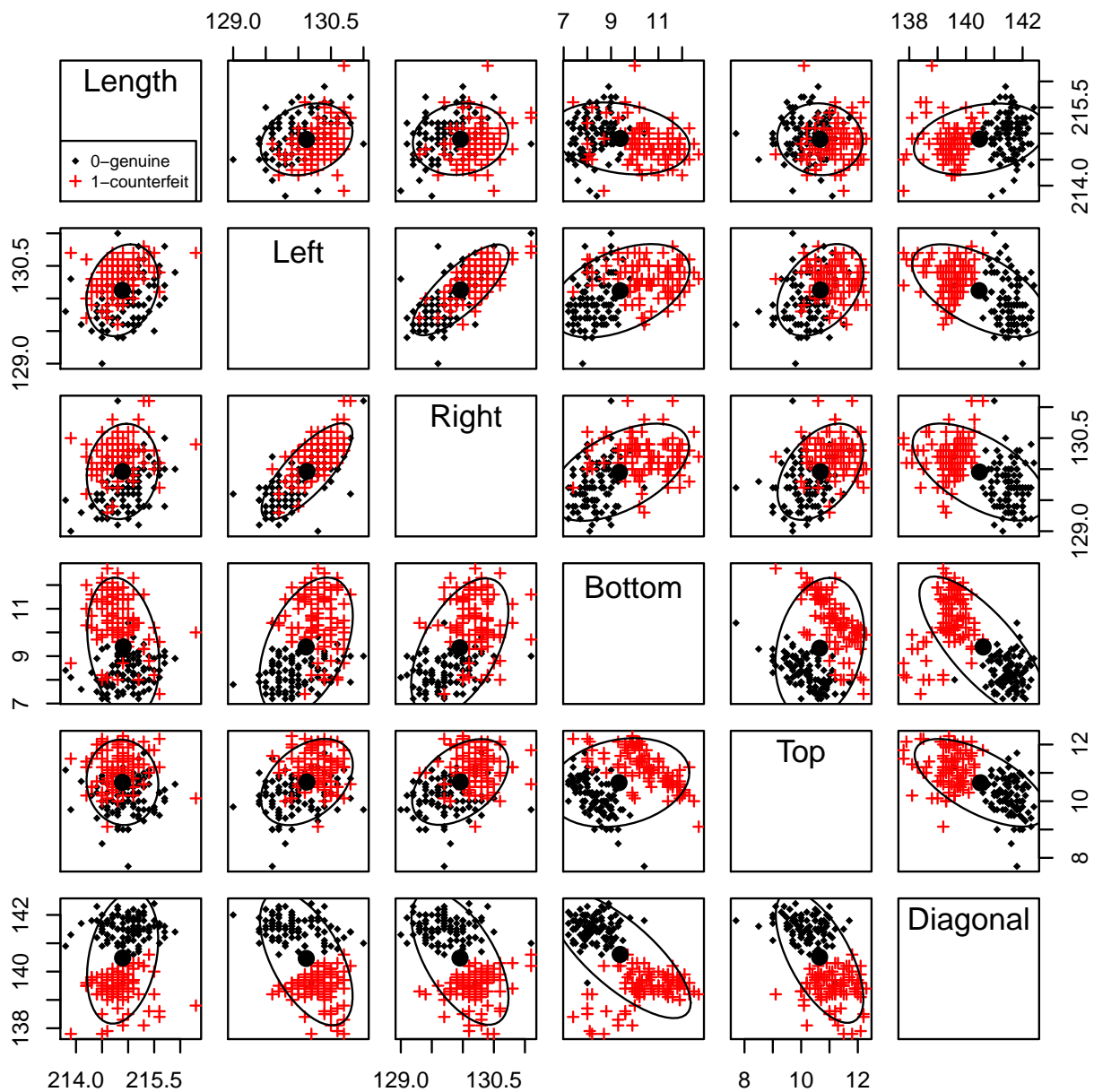
	var	n	mean	sd	skew	kurtosis	se
Length	1	200	214.90	0.38	0.19	0.71	0.03
Left	2	200	130.12	0.36	-0.19	-0.59	0.03
Right	3	200	129.96	0.40	0.04	-0.19	0.03
Bottom	4	200	9.42	1.44	0.37	-1.05	0.10
Top	5	200	10.65	0.80	-0.23	0.15	0.06
Diagonal	6	200	140.48	1.15	-0.19	-1.15	0.08

```
> print(cor(banknote[, -7]), digits=3)
```

	Length	Left	Right	Bottom	Top	Diagonal
Length	1.0000	0.231	0.152	-0.190	-0.0613	0.194
Left	0.2313	1.000	0.743	0.414	0.3623	-0.503
Right	0.1518	0.743	1.000	0.487	0.4007	-0.516
Bottom	-0.1898	0.414	0.487	1.000	0.1419	-0.623
Top	-0.0613	0.362	0.401	0.142	1.0000	-0.594
Diagonal	0.1943	-0.503	-0.516	-0.623	-0.5940	1.000

```
> library(car)
```

```
> print(scatterplotMatrix(~ . - Y | Y, data=banknote, smooth=FALSE, reg.line=FALSE, pch=c(18, 3),
+   cex=.75, ellipse=TRUE, levels= c(0.9), by.groups=FALSE, diagonal="none"))
```



We will use the `prcomp` function in R for PCA that uses the `svd` function to do its computing (there is also a similar function called `princomp` that uses `eigen` and will not be discussed). We will standardize and work with the correlation matrix by setting `center=TRUE`, the default, and `scale=TRUE`, which is not the default.

```
> print(p1 <- prcomp(~ . - Y, data = banknote, center=TRUE, scale=TRUE), digits=3)
```

Standard deviations:

```
[1] 1.716 1.131 0.932 0.671 0.518 0.435
```

Rotation:

	PC1	PC2	PC3	PC4	PC5	PC6
Length	0.00699	-0.8155	0.0177	0.575	-0.0588	0.0311
Left	-0.46776	-0.3420	-0.1034	-0.395	0.6395	-0.2977
Right	-0.48668	-0.2525	-0.1235	-0.430	-0.6141	0.3492
Bottom	-0.40676	0.2662	-0.5835	0.404	-0.2155	-0.4624
Top	-0.36789	0.0915	0.7876	0.110	-0.2198	-0.4190
Diagonal	0.49346	-0.2739	-0.1139	-0.392	-0.3402	-0.6318

As a change of notation, let X now be the **standardized data matrix** so $X1 = 0$ and $(X'X)/(n-1)$ is a correlation matrix. The `prcomp` command with `center=TRUE` and `scale=TRUE` computes the SVD of

$$(n-1)^{-1/2}X = UDV'$$

With the constant $(n-1)^{-1/2}$, the spectral decomposition of the sample correlation matrix is VD^2V' .

```
> n1 <- dim(banknote)[1] - 1
> s <- svd(sqrt(1/n1) * scale(banknote[, -7]))
```

The singular values of s

```
> s$d
[1] 1.7163 1.1305 0.9322 0.6706 0.5183 0.4346
```

are the same as the **Standard deviations** in the `prcomp` output. The **Rotation** is the orthogonal matrix V of the right singular vectors (the matrix `s$v`), or equivalently eigenvectors of the sample correlation matrix. The first rotation vector gives the linear combination of the standardized variables with the largest variance, so it explains the most variation. It essentially ignores **Length**, averages the next 5 variables but with **Diagonal** with a negative sign. The second component is effectively **Length**. Because V is orthogonal, the sum of squares of the elements in each row and the sum of squares of elements in each column, is equal to 1.

```
> summary(p1)
```

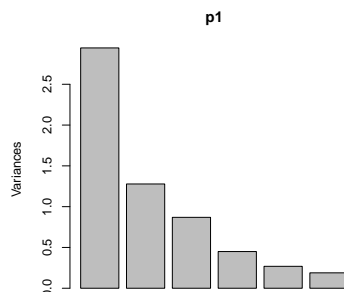
Importance of components:

	PC1	PC2	PC3	PC4	PC5	PC6
Standard deviation	1.716	1.131	0.932	0.671	0.5183	0.4346
Proportion of Variance	0.491	0.213	0.145	0.075	0.0448	0.0315
Cumulative Proportion	0.491	0.704	0.849	0.924	0.9685	1.0000

The summary repeats the singular values $\sqrt{\lambda_i}, i = 1 \dots, p$ in the first row (these are also the square roots of the eigenvalues λ_i of the correlation matrix in normalized PCA). The **Cumulative Proportions** are the scaled inertia values $\psi_j = \sum_{i=1}^j \lambda_i / \sum_{i=1}^p \lambda_i = \sum_{i=1}^j \lambda_i / p$ if the correlation matrix is used. The first two PCs account for 70.4% of the variance in the data.

The default plot method for `prcomp` produces a *scree plot*, a bar chart of the PC variances λ_i , the squares of the standard deviations.

```
> plot(p1)
```



The scree plot is supposed to help you choose the number of PCs to use by looking for an “elbow” in the plot. Here 2 or 3 PCs makes sense.

The *principal component scores* are $Y = XV = UDV'V = UD$. The first 6 rows of this $n \times p$ matrix are:

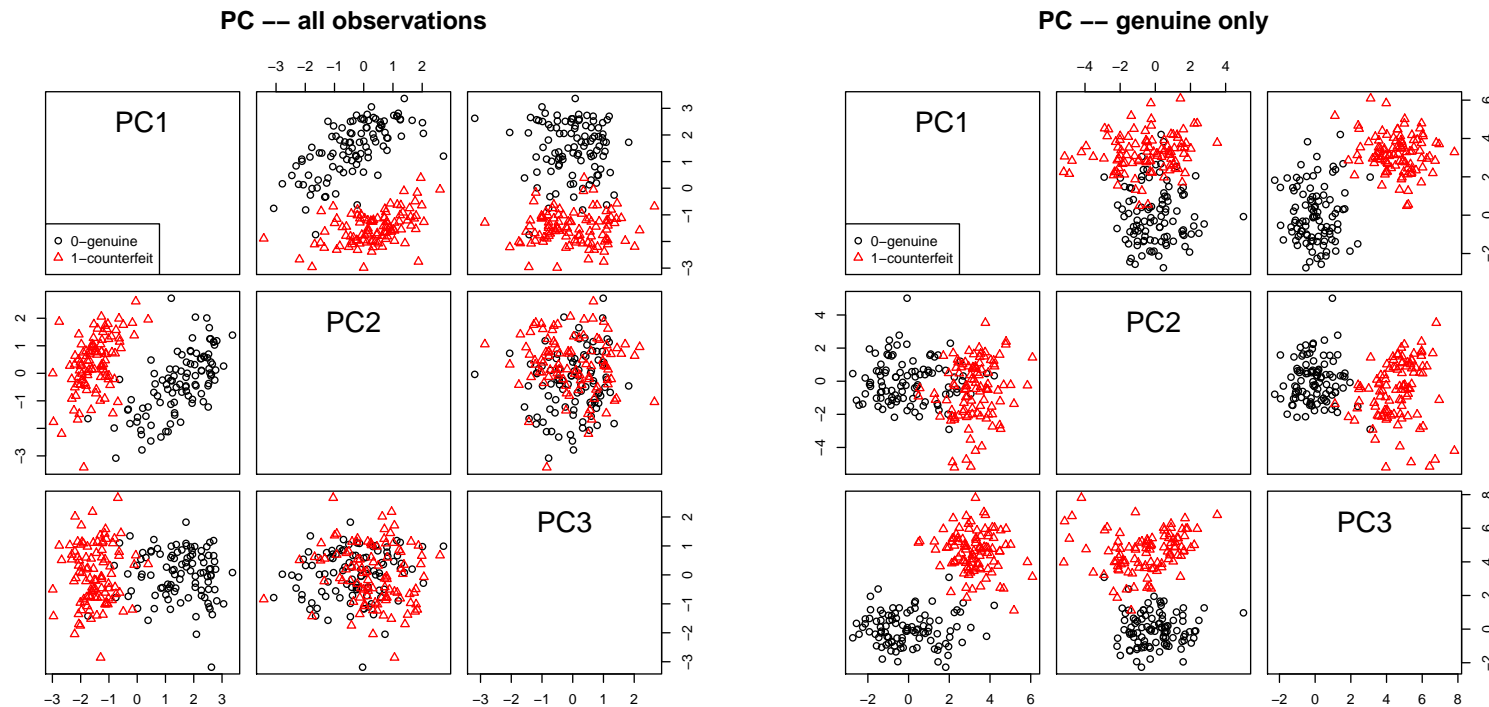
```
> print(head(p1$x, n=4), digits=3)
```

	PC1	PC2	PC3	PC4	PC5	PC6
1	-1.74	-1.6467	-1.420	-2.748	0.00329	0.6020
2	2.27	0.5374	-0.531	-0.657	-0.15817	0.4565
3	2.27	0.1074	-0.716	-0.341	-0.45388	-0.0453
4	2.28	0.0874	0.604	-0.392	-0.28291	-0.0554

The results here differ from those in the text because I used the scaled data and correlations rather than the unscaled data and covariances. In the text using the correlation matrix is called *normalized principal component analysis*. I see no justification for using the unscaled data in this problem (and I'm not so keen on the scaled, either!).

Here are scatterplots of the first three PC scores. The PCs are of course uncorrelated, but the colors here reveal an interesting picture. There is no reason in theory why the PCs should separate groups not used in computing them, as they appear to do in this example.

```
> scatterplotMatrix(~ PC1 + PC2 + PC3|banknote$Y, main="PC -- all observations",
+   data=predict(p1), diagonal="none", smooth=FALSE, reg.line=FALSE)
```



An interesting exercise in this problem is to compute PC analysis with only the genuine bills

```
> (p2 <- update(p1, subset=Y=="0-genuine"))
```

Standard deviations:

```
[1] 1.4845 1.3026 0.9827 0.7635 0.5716 0.4734
```

Rotation:

	PC1	PC2	PC3	PC4	PC5	PC6
Length	0.44955	0.07400	-0.47875	-0.741568	0.06494	0.09532
Left	0.58505	-0.10733	0.04828	0.286154	-0.69957	0.26943
Right	0.57218	-0.03576	-0.07727	0.455925	0.67581	0.02766
Bottom	0.28120	0.61628	0.27956	-0.044657	-0.11601	-0.66897
Top	0.08239	-0.70886	-0.17047	-0.005562	-0.09547	-0.67270
Diagonal	-0.20583	0.31535	-0.80950	0.397869	-0.16460	-0.13231

```
> scatterplotMatrix(~ PC1 + PC2 + PC3/banknote$Y, main="PC -- genuine only",  
+ data=predict(p2, banknote), diagonal="none", smooth=FALSE, reg.line=FALSE)
```

Objects created by `prcomp` have the following components:

```
> names(p1)
```

```
[1] "sdev"      "rotation" "center"   "scale"    "x"         "call"
```

The square roots of the eigenvalues are returned by `p1$sdev`. The eigenvectors are returned by `p1$rotation` and `p1$x`, or `predict(p1)`, returns PC scores. `p1$center` returns the column means of the input matrix, and if `scale=TRUE`, `p1$scale` returns the column SDs.

Tests and Intervals

Tests/confidence statements generally are based on the sample eigenvalues. For example, from (9.18) on p. 226, if ℓ_i is the observed value of the i -th eigenvalue, and λ_i is the unobserved population value, then $\sqrt{n-1}(\log(\ell_j) - \log(\lambda_j))$ is asymptotically $N(0, 2)$. Marginal 95% confidence intervals for the standard deviations (square roots of the eigenvalues) are

```
> n <- dim(p1$x)[1]  
> cis <- data.frame(obs = log(p1$sdev^2), upper = log(p1$sdev^2) - 1.96 * sqrt(2/(n-1)),  
+ lower = log(p1$sdev^2) + 1.96 * sqrt(2/(n-1)))  
> sqrt(exp(cis))
```

	obs	upper	lower
1	1.7163	1.5557	1.8934
2	1.1305	1.0247	1.2472

```

3 0.9322 0.8450 1.0285
4 0.6706 0.6079 0.7399
5 0.5183 0.4698 0.5719
6 0.4346 0.3939 0.4795

```

We can do a similar calculation concerning variance explained by the first q PCs, from Theorem 9.5. Let $\psi_q = \sum_1^q \lambda_i / \sum_1^n \lambda_i$ be the (scaled) population variance explained by the first q PCs, and $\hat{\psi}_q$ is its estimator. Then $\sqrt{n-1}(\hat{\psi}_q - \psi_q)$ is asymptotically $N(0, \omega^2)$, where ω^2 is

$$\omega^2 = \frac{2\text{tr}(\Sigma^2)}{(\text{tr}(\Sigma))^2} [\psi_q^2 - 2\beta\psi_q + \beta^2]$$

and $\beta_q = \sum_1^q \lambda_i^2 / \sum_1^n \lambda_i^2$. The variance is estimated by substituting estimates for parameters.

For example to test if the first three PCs explain 90% of the variance,

```

> evals <- p1$sdev^2 # for convenience
> psihat <- sum(evals[1:3]) / sum(evals)
> betahat <- sum(evals[1:3]^2) / sum(evals^2)
> trS <- sum(evals)
> trS2 <- sum(evals^2)
> omegahat2 <- (2 * trS2/(trS^2)) * (psihat^2 - 2*betahat * psihat + betahat^2)
> psihat + c(-1, 1) * sqrt(omegahat2)/(sqrt(n-1))

[1] 0.8418 0.8558

```

We would reject 0.90 as it is outside the confidence interval.

Correlations between PCs and Original Variables

The correlations between the original variables and the PCs are given by (9.15) in the text. Let x be a $p \times 1$ vector of the original (scaled) variables, and $y = V'x$ be a $p \times 1$ vector of the corresponding principal components (we need the transpose on V because then the first element of y is the linear combination v'_1x , and so on). Then assuming that x is centered,

$$\text{cov}(x, y) = E((xy')) = E(xx'V) = E(xx')V = \Sigma V = VDV'V = VD$$

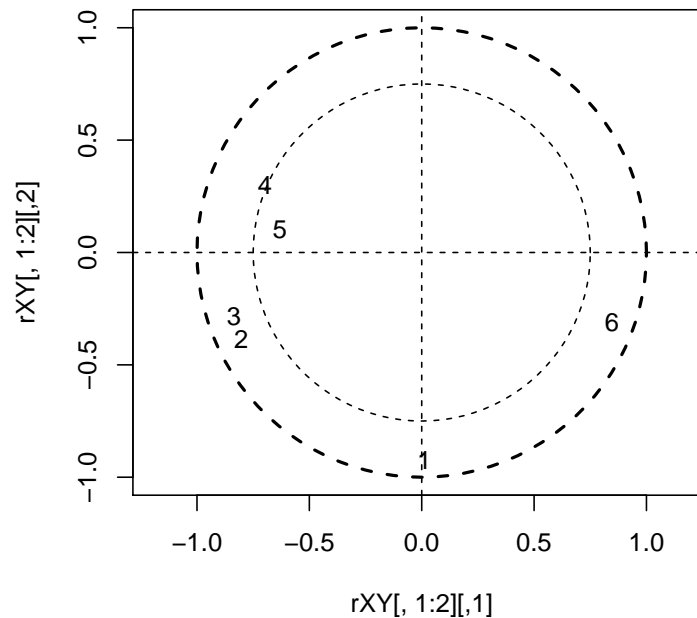
We can get a correlation between the i th predictor and the j th PC score by dividing the (i, j) element of VD by the product of the corresponding standard deviations, which is just $\sqrt{\text{Var}(x_i)\lambda_j} = \sqrt{\lambda_j}$ if X is centered and scaled, as in this and many examples.


```
> (rXY <- with(p1, rotation %*% diag( sdev)))
```

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]
Length	0.01199	-0.9219	0.01648	0.38537	-0.03048	0.0135
Left	-0.80280	-0.3866	-0.09638	-0.26485	0.33148	-0.1294
Right	-0.83527	-0.2854	-0.11511	-0.28857	-0.31831	0.1517
Bottom	-0.69810	0.3010	-0.54399	0.27072	-0.11169	-0.2009
Top	-0.63140	0.1034	0.73419	0.07392	-0.11396	-0.1821
Diagonal	0.84690	-0.3097	-0.10616	-0.26285	-0.17632	-0.2746

The squared correlation can be viewed as *the proportion of the variance in the original variable “explained” by each of the orthogonal PCs*, and so the sum of the squared entries of each row of this table is 1. With a two-dimensional solution, the text suggests plotting the correlations r_{XY_1} between each of the variables and the first principal vector versus r_{XY_2} . Points close to the boundary of the unit circle in this plot correspond to variables that are explained by the first two PCs. The code below uses `asp=1` to assure the aspect ratio is one, and `draw.circle` in the `plotrix` package to draw a circle. In this example, all six variables are fairly close to the boundary circle, and for all variables except for the fifth (Top) at least 75% of the variation is captured by the first two principal component vectors.

```
> plot(rXY[,1:2], asp=1, xlim=c(-1, 1), ylim=c(-1, 1), pch=as.character(1:6))
> abline(h=0, lty=2)
> abline(v=0, lty=2)
> require(plotrix)
> draw.circle(0, 0, 1, lty=2, lwd=2)
> draw.circle(0, 0, 0.75, lty=2)
```



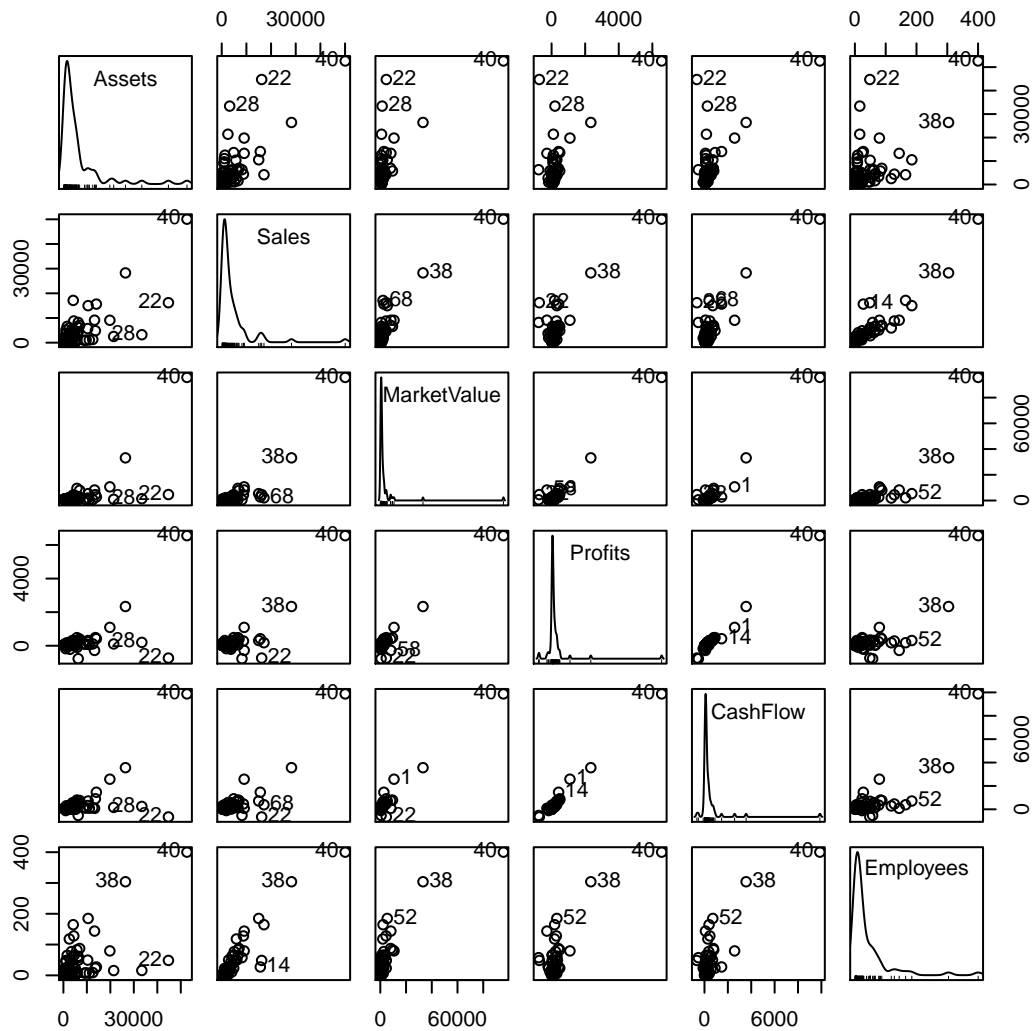
Example 9.9

We have data on 79 US companies and six variables: assets, sales, market value, profits, cash flow and number of employees. PC analysis could be useful here to reduce dimensionality to discover common characteristics among companies, or unusual companies. Because the variables are in completely different units, correlation scale seems appropriate.

```
> loc <- "http://www.stat.umn.edu/~sandy/courses/8053/Data/uscomp1.dat"
> head(uscomp <- read.table(url(loc), header=TRUE), n=4)
```

	Assets	Sales	MarketValue	Profits	CashFlow	Employees
1	19788	9084	10636	1092.9	2576.8	79.4
2	5074	2557	1892	239.9	578.3	21.9
3	13621	4848	4572	485.0	898.9	23.4
4	1117	1038	478	59.7	91.7	3.8

```
> scatterplotMatrix(uscomp, id.n=3, smooth=FALSE, reg=FALSE)
```



For future reference,

Case No.	Company
14	Phillips Petroleum
22	Cigna
28	Marine Corp.
38	GE
40	IBM
52	United Technologies

```
> print(pr1 <- prcomp(uscomp, center=TRUE, scale=TRUE), digits=3)
```

Standard deviations:

```
[1] 2.2447 0.7189 0.5991 0.2225 0.1706 0.0829
```

Rotation:

	PC1	PC2	PC3	PC4	PC5	PC6
Assets	-0.340	0.84921	-0.339	0.2050	0.0770	-0.00593
Sales	-0.423	0.17011	0.379	-0.7833	-0.0057	-0.18560
MarketValue	-0.434	-0.19013	-0.192	0.0708	-0.8437	0.14859
Profits	-0.420	-0.36370	-0.324	0.1556	0.2606	-0.70319
CashFlow	-0.428	-0.28528	-0.267	-0.1215	0.4523	0.66686
Employees	-0.397	-0.00987	0.726	0.5481	0.0983	0.06523

```
> summary(pr1)
```

Importance of components:

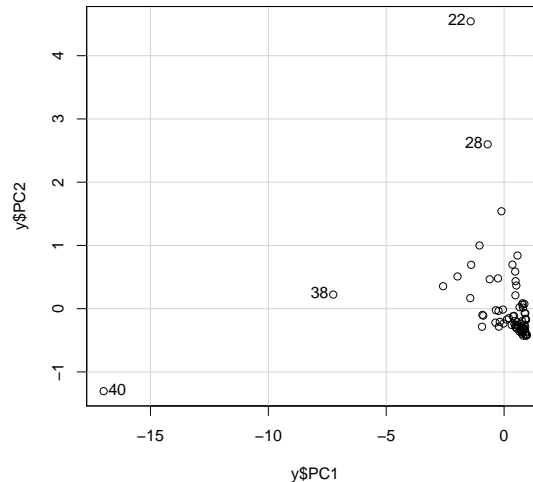
	PC1	PC2	PC3	PC4	PC5	PC6
Standard deviation	2.24	0.7189	0.5991	0.22249	0.17060	0.08289
Proportion of Variance	0.84	0.0862	0.0598	0.00825	0.00485	0.00115
Cumulative Proportion	0.84	0.9259	0.9858	0.99400	0.99885	1.00000

The first PC accounts for 84% of the variance and the first two account for 93%. We examine the plot of the first two PC scores, using sector label as the plotting symbol:

```
> y <- as.data.frame(pr1$x)
> scatterplot(y$PC1, y$PC2, smooth=FALSE, reg=FALSE, box=F, id.n=4)
```

```
22 28 38 40
```

```
22 28 38 40
```



Cases 38 and 40 are very different on the first principal component, while two financial companies are different on the second component. The first two PCs merely serve to discover that these are different from the others. Excluding 38 and 40, and figuring out why they are different, is probably sensible.

```
> print(pr2 <- prcomp(uscomp[-c(38, 40), ],center=TRUE, scale=TRUE), digits=3)
```

Standard deviations:

```
[1] 1.786 1.239 0.889 0.541 0.386 0.203
```

Rotation:

	PC1	PC2	PC3	PC4	PC5	PC6
Assets	-0.263	0.40757	0.7997	0.0675	-0.333	0.0988
Sales	-0.438	0.40712	-0.1616	0.5094	0.441	-0.4028
MarketValue	-0.500	0.00289	0.0351	-0.8013	0.265	-0.1903
Profits	-0.331	-0.62348	0.0803	0.1920	-0.426	-0.5262
CashFlow	-0.443	-0.45013	0.1234	0.2379	0.335	0.6456
Employees	-0.427	0.27711	-0.5580	-0.0209	-0.575	0.3132

```
> summary(pr2)
```

Importance of components:

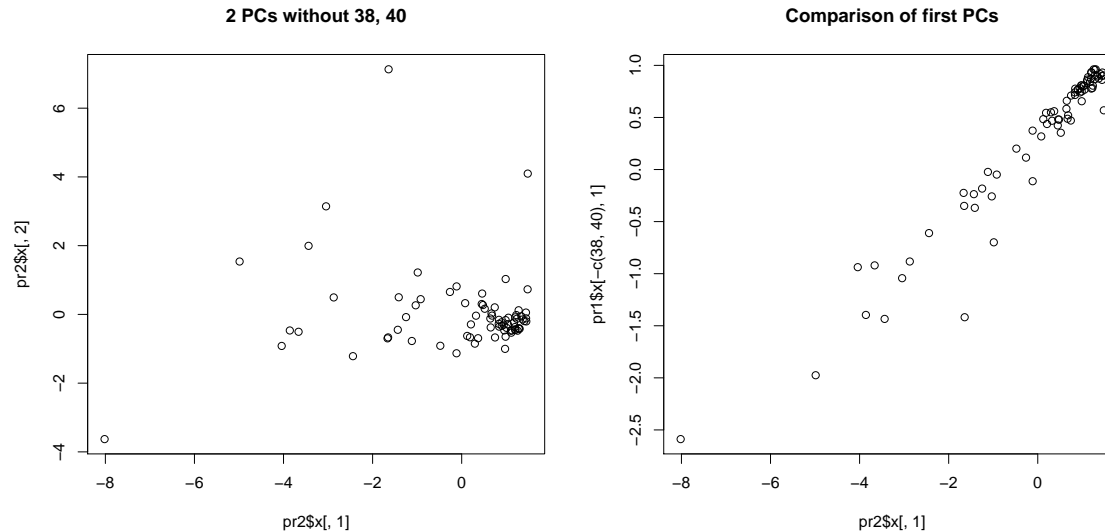
PC1	PC2	PC3	PC4	PC5	PC6
-----	-----	-----	-----	-----	-----

```
Standard deviation      1.786 1.239 0.889 0.5406 0.3859 0.2035
Proportion of Variance 0.532 0.256 0.132 0.0487 0.0248 0.0069
Cumulative Proportion  0.532 0.788 0.920 0.9683 0.9931 1.0000
```

```
> cor(pr2$x[, 1:2],uscomp[-c(38, 40), ])
```

```
Assets Sales MarketValue Profits CashFlow Employees
PC1 -0.4701 -0.7832 -0.894126 -0.5907 -0.7914 -0.7632
PC2 0.5050 0.5044 0.003579 -0.7725 -0.5577 0.3433
```

```
> par(mfrow=c(1, 2))
> plot(pr2$x[, 1],pr2$x[, 2], main="2 PCs without 38, 40")
> plot(pr2$x[, 1],pr1$x[-c(38, 40), 1], main="Comparison of first PCs")
```



The first principal component remains close to the average of the variables.

As an alternative, start with the US company data, and consider first transforming for normality using the Box-Cox method and the function `powerTransform` in `car`.

```
> #pairs(uscomp)
> library(car)
> summary(powerTransform(uscomp,family="yjPower"))
```

yjPower Transformations to Multinormality

	Est.Power	Std.Err.	Wald Lower Bound	Wald Upper Bound
Assets	0.0556	0.0810	-0.1032	0.2143
Sales	0.1083	0.0589	-0.0072	0.2238
MarketValue	0.1783	0.0658	0.0494	0.3072
Profits	0.9501	0.0106	0.9294	0.9708
CashFlow	0.9195	0.0121	0.8958	0.9433
Employees	0.0398	0.0700	-0.0974	0.1770

Likelihood ratio tests about transformation parameters

	LRT	df	pval
LR test, lambda = (0 0 0 0 0 0)	2654.298	6	0.000
LR test, lambda = (1 1 1 1 1 1)	573.579	6	0.000
LR test, lambda = (0 0 0.18 0.95 0.92 0)	3.687	6	0.719

```
> print(pr3<-prcomp(~log(Assets) + log(Sales) + log(MarketValue) + Profits +
+ CashFlow + log(Employees), data=uscomp, scale=TRUE),
+ digits=3)
```

Standard deviations:

```
[1] 1.9625 1.0873 0.7790 0.5135 0.2972 0.0857
```

Rotation:

	PC1	PC2	PC3	PC4	PC5	PC6
log(Assets)	-0.343	0.195	0.9080	-0.040	-0.1347	-0.0162
log(Sales)	-0.436	0.387	-0.1279	0.370	0.7074	-0.0798
log(MarketValue)	-0.447	0.085	-0.2120	-0.860	0.0878	-0.0155
Profits	-0.387	-0.589	-0.0374	0.156	-0.0819	-0.6856
CashFlow	-0.413	-0.531	-0.0172	0.156	0.0331	0.7223
log(Employees)	-0.415	0.419	-0.3353	0.269	-0.6826	0.0357

```
> summary(pr3)
```

Importance of components:

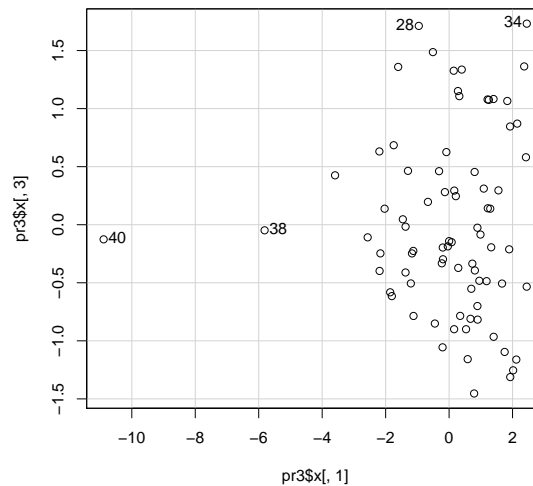
```
PC1 PC2 PC3 PC4 PC5 PC6
```

```
Standard deviation      1.963 1.087 0.779 0.514 0.2972 0.08572
Proportion of Variance 0.642 0.197 0.101 0.044 0.0147 0.00122
Cumulative Proportion  0.642 0.839 0.940 0.984 0.9988 1.00000
```

```
> scatterplot(pr3$x[, 1], pr3$x[, 3], box=FALSE, reg=FALSE,
+             smooth=FALSE, id.n=4)
```

```
28 34 38 40
```

```
28 34 38 40
```



```
> round(rXY <- with(pr3, rotation %*% diag( sdev)), 2)
```

	[,1]	[,2]	[,3]	[,4]	[,5]	[,6]
log(Assets)	-0.67	0.21	0.71	-0.02	-0.04	0.00
log(Sales)	-0.86	0.42	-0.10	0.19	0.21	-0.01
log(MarketValue)	-0.88	0.09	-0.17	-0.44	0.03	0.00
Profits	-0.76	-0.64	-0.03	0.08	-0.02	-0.06
CashFlow	-0.81	-0.58	-0.01	0.08	0.01	0.06
log(Employees)	-0.81	0.46	-0.26	0.14	-0.20	0.00


```

> plot(rXY[,1:2], asp=1, xlim=c(-1, 1), ylim=c(-1, 1), pch=as.character(1:6))
> abline(h=0, lty=2)
> abline(v=0, lty=2)
> require(plotrix)
> draw.circle(0, 0, 1, lty=2, lwd=2)
> draw.circle(0, 0, .75, lty=2, lwd=1)

```

