



MONASH University

ETC3250

Business Analytics

Week 7

Principal Components Analysis

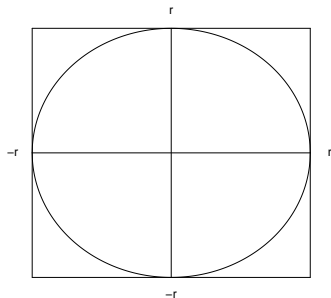
17 April 2018

Outline

Week	Topic	Chapter	Lecturer
1	Introduction	1	Souhaib
2	Statistical learning	2	Souhaib
3	Regression	3	Souhaib
4	Classification	4	Souhaib
5	Clustering	10	Souhaib
Semester break			
6	Model selection and resampling methods	5	Souhaib
7	Dimension reduction	6,10	Souhaib
8	Advanced regression	6	Souhaib
9	Advanced regression	6	Souhaib
10	Advanced classification	9	Souhaib
11	Tree-based methods	8	Souhaib
12	Project presentation		Souhaib

The Strange Geometry in High Dimensions

Consider the **hyper-cube** $[-r, r]^d$ and the inscribed **hyper-sphere**. What does your intuition tell you about the relative sizes of these two objects as $d \rightarrow \infty$?



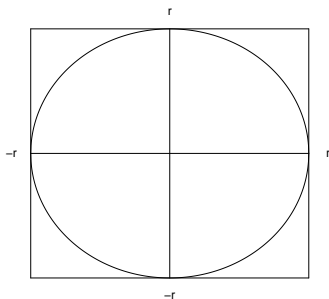
- 1 volume of sphere \gg volume of cube
- 2 volume of sphere \approx volume of cube
- 3 volume of sphere \ll volume of cube

The Strange Geometry in High Dimensions

$$s_d = \frac{\text{Volume(hyper-sphere)}}{\text{Volume(hyper-cube)}} = \frac{\frac{r^d \pi^{d/2}}{\Gamma(\frac{d}{2}+1)}}{(2r)^d} = \underbrace{\left(\frac{\sqrt{\pi}}{2}\right)^d}_{<1} \frac{1}{\Gamma(\frac{d}{2}+1)}$$

where $\Gamma(\cdot)$ is the Gamma function.

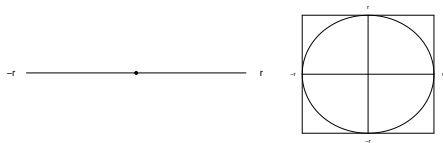
- s_d does not depend on r , just on d
- As the dimension increases, the volume of the sphere is much smaller (infinitesimal) than that of the cube
- How is this going against intuition?



The Strange Geometry in High Dimensions

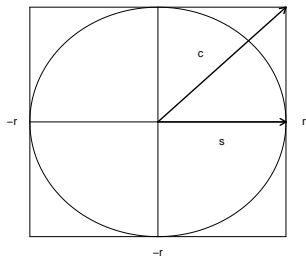
d	1	2	3	4	5	6
s_d	1	.785	.524	.308	.164	.080

- Not very surprising. we can see it even in lower dimensions. For $d = 1$, they have the same volume. For $d = 2$, area of the circle is already smaller.



- As the dimension increases, the volume of the area between the cube and the sphere becomes larger
- In high dimensions, all the volume of the cube is in “the spikes”

The Strange Geometry in High Dimensions



$$\blacksquare \mathbf{c} = (r, r, \dots, r) \in \mathbb{R}^d$$

$$\blacksquare \mathbf{s} = (r, 0, \dots, 0) \in \mathbb{R}^d$$

$$\blacksquare \frac{\|\mathbf{c}\|_2^2}{\|\mathbf{s}\|_2^2} = \frac{dr^2}{r^2} = d \rightarrow \infty$$

The dot product of two vectors \mathbf{c} and \mathbf{s} is defined by $\mathbf{c} \cdot \mathbf{s} = \mathbf{c}'\mathbf{s} = \|\mathbf{c}\|_2 \|\mathbf{s}\|_2 \cos(\theta)$ where $\cos(\theta)$ is the angle between \mathbf{c} and \mathbf{s} .

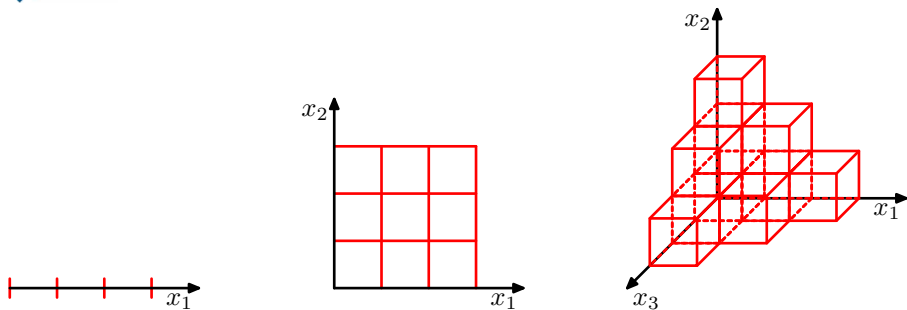
$$\cos(\theta) = \frac{\mathbf{c}'\mathbf{s}}{\|\mathbf{c}\|_2 \|\mathbf{s}\|_2} = \frac{r^2}{\sqrt{dr^2}r} = \frac{1}{\sqrt{d}} \rightarrow 0$$

$\rightarrow \mathbf{c}$ is orthogonal to \mathbf{s} as d increases, and \mathbf{c} is infinitely larger than \mathbf{s}

The curse of dimensionality

- High-dimensional balls (the space bounded by a sphere) have a vanishing volume
- The volume of a high-dimensional ball is concentrated in its crust
 - Most data points are closer to the boundary of the sample space than to any other data point
 - Prediction is much more difficult near the edges of the training sample: extrapolation vs interpolation
- All the points are at a similar distance one from the others
 - The notion of nearest-points vanishes
- High dimensional spaces are strange → never trust your intuition in high dimensions!

The curse of dimensionality



If we divide a region of a space into regular cells, then the number of such cells **grows exponentially** with the dimensionality of the space \implies We would need an **exponentially large quantity of training data points** in order to ensure that the cells are not empty.

Can we learn in high dimension?

Yes!

- First, real data will often be confined to a region of the space having *lower intrinsic dimensionality*. The data *lives* in a low dimensional subspace.
- Second, real data will typically exhibit some smoothness properties (at least locally)

→ Dimensionality reduction

Why dimensionality reduction?

- Curse of dimensionality
- Intrinsic dimensionality
- Visualization
- Reduce computation and storage

We avoid unnecessary dimensions which can be measured in two ways:

- features are not useful
- features are not independent

Dimension reduction methods

- Feature selection vs feature extraction
- Unsupervised vs supervised
- Linear vs nonlinear

Principal components analysis (PCA) is an unsupervised linear feature extraction method.

Principal components analysis

PCA produces a **low-dimensional representation** of a dataset. It finds a sequence of **linear combinations of the variables** that have **maximal variance**, and are **mutually uncorrelated**.

Why?

- We may have **too many predictors** for a regression. Instead, we can use the first few principal components.
- **Understanding** relationships between variables.
- We can **plot a small number of variables**

Principal components analysis

PCA produces a **low-dimensional representation** of a dataset. It finds a sequence of **linear combinations of the variables** that have **maximal variance**, and are **mutually uncorrelated**.

Why?

- We may have **too many predictors** for a regression. Instead, we can use the first few principal components.
- **Understanding** relationships between variables.
- We can **plot a small number of variables**

Principal components analysis

The first principal component of a set of features x_1, x_2, \dots, x_p is the linear combination

$$Z_1 = \phi_{11}x_1 + \phi_{21}x_2 + \dots + \phi_{p1}x_p$$

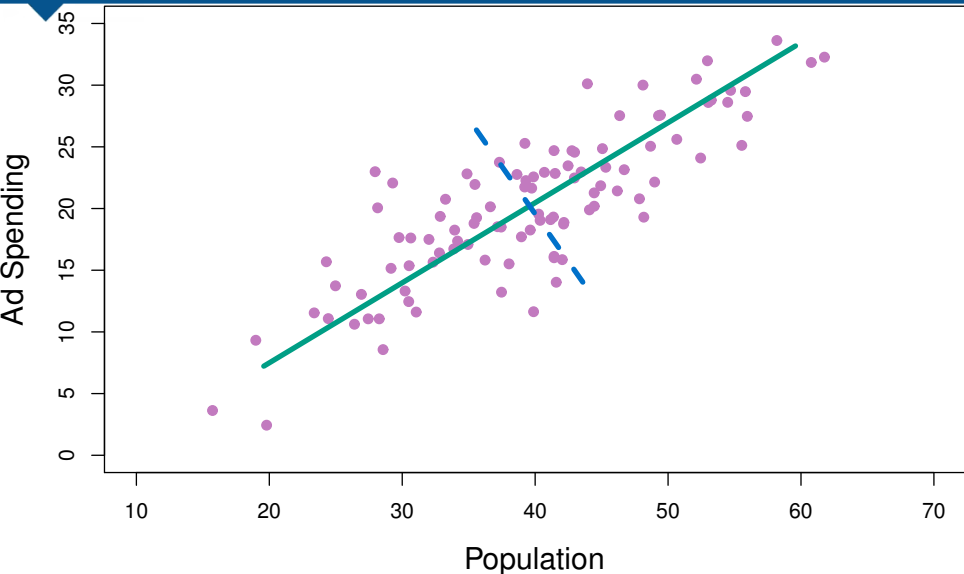
that has the largest variance such that $\sum_{j=1}^p \phi_{j1}^2 = 1$.

$\phi_{11}, \dots, \phi_{p1}$ are the loadings of the first principal component.

Geometry of PCA

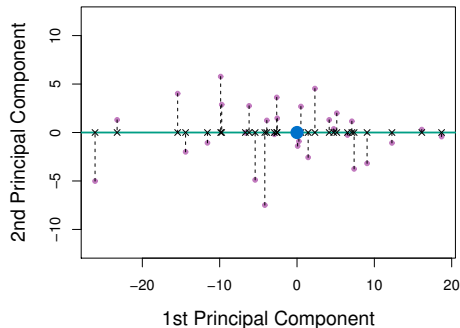
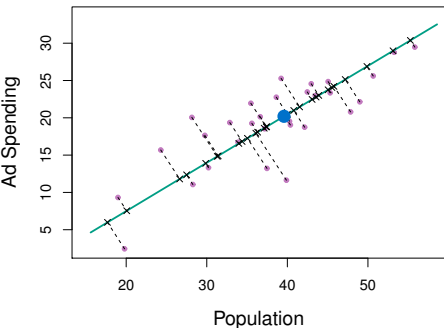
- The loading vector $\phi_1 = [\phi_{11}, \dots, \phi_{p1}]'$ defines a direction in feature space along which the data vary the most.
- If we project the n data points $\mathbf{x}_1, \dots, \mathbf{x}_n$ onto this direction, the projected values are the principal component scores z_{11}, \dots, z_{n1} .
- The second principal component is the linear combination $z_{i2} = \phi_{12}x_{i1} + \phi_{22}x_{i2} + \dots + \phi_{p2}x_{ip}$ that has maximal variance among all linear combinations that are *uncorrelated* with z_1 .
- Equivalent to constraining ϕ_2 to be orthogonal (perpendicular) to ϕ_1 . And so on.
- There are at most $\min(n - 1, p)$ PCs.

PCA Example



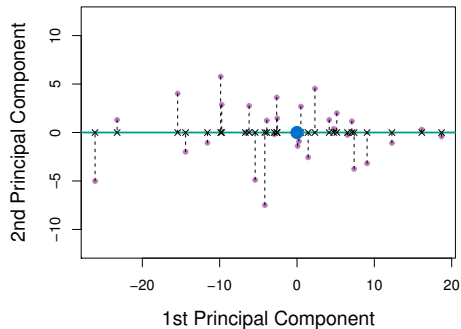
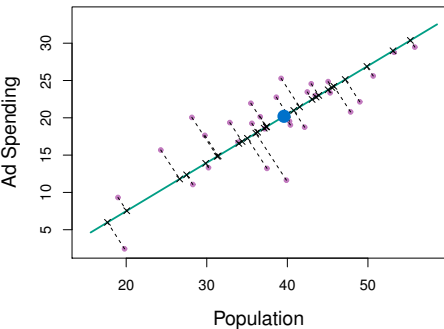
Green: first PC. Blue: second PC

Further principal components



PCA can be thought of as fitting an n -dimensional ellipsoid to the data, where each axis of the ellipsoid represents a principal component.

Further principal components



PCA can be thought of as fitting an n -dimensional ellipsoid to the data, where each axis of the ellipsoid represents a principal component.

Computation of PCs

Suppose we have a $n \times p$ data set $\mathbf{X} = [x_{ij}]$.

- Centre each of the variables to have mean zero (i.e., the column means of \mathbf{X} are zero).
- $z_{i1} = \phi_{11}x_{i1} + \phi_{21}x_{i2} + \cdots + \phi_{p1}x_{ip}$
- Sample variance of z_{i1} is $\frac{1}{n} \sum_{i=1}^n z_{i1}^2$.

$$\underset{\phi_{11}, \dots, \phi_{p1}}{\text{maximize}} \frac{1}{n} \sum_{i=1}^n \left(\sum_{j=1}^p \phi_{j1} x_{ij} \right)^2 \quad \text{subject to} \quad \sum_{j=1}^p \phi_{j1}^2 = 1$$

Computation of PCs

- 1 Compute the covariance matrix (after scaling the columns of \mathbf{X})

$$\mathbf{C} = \mathbf{X}'\mathbf{X}$$

- 2 Find eigenvalues and eigenvectors:

$$\mathbf{C} = \mathbf{V}\mathbf{D}\mathbf{V}'$$

where columns of \mathbf{V} are orthonormal (i.e., $\mathbf{V}'\mathbf{V} = \mathbf{I}$)

- 3 Compute PCs: $\Phi = \mathbf{V}$. $\mathbf{Z} = \mathbf{X}\Phi$.

Computation of PCs

Singular Value Decomposition

$$\mathbf{X} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}'$$

- \mathbf{X} is $n \times p$ matrix
- \mathbf{U} is $n \times r$ matrix with orthonormal columns ($\mathbf{U}'\mathbf{U} = \mathbf{I}$).
The columns of \mathbf{U} are called the left-singular vectors of \mathbf{X} .
- $\mathbf{\Lambda}$ is $r \times r$ diagonal matrix with non-negative elements (called singular values).
- \mathbf{V} is $p \times r$ matrix with orthonormal columns ($\mathbf{V}'\mathbf{V} = \mathbf{I}$).
The columns of \mathbf{V} are called the right-singular vectors of \mathbf{X}

It is always possible to uniquely decompose a matrix in this way.

Computation of PCs

- 1 Compute SVD: $\mathbf{X} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}'$.
- 2 Compute PCs: $\Phi = \mathbf{V}$. $\mathbf{Z} = \mathbf{X}\Phi$.

Relationship with covariance:

$$\mathbf{C} = \mathbf{X}'\mathbf{X} = \mathbf{V}\mathbf{\Lambda}\mathbf{U}'\mathbf{U}\mathbf{\Lambda}\mathbf{V}' = \mathbf{V}\mathbf{\Lambda}^2\mathbf{V}' = \mathbf{V}\mathbf{D}\mathbf{V}'$$

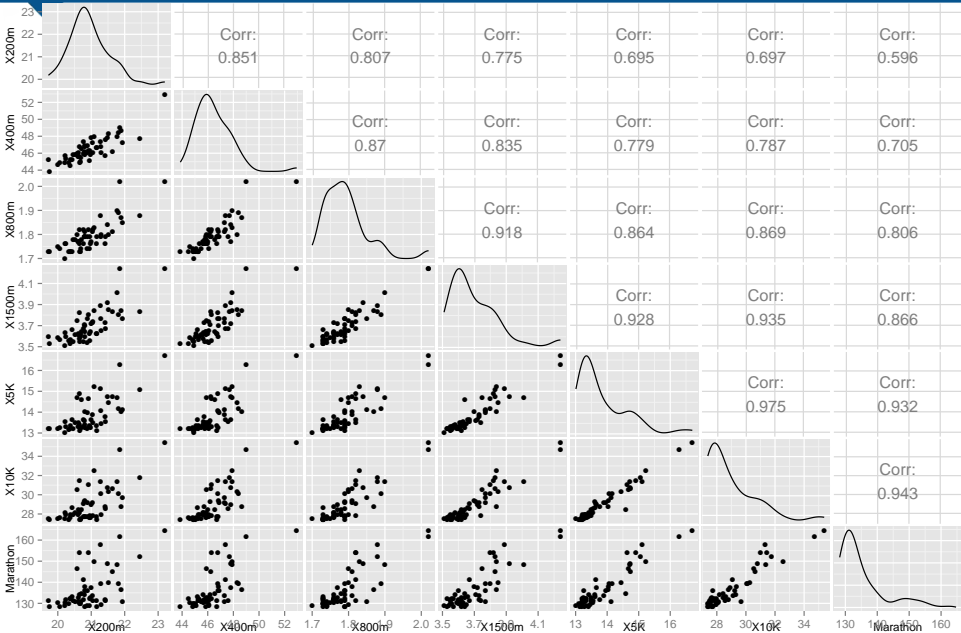
- Eigenvalues of \mathbf{C} are squares of singular values of \mathbf{X} .
- Eigenvectors of \mathbf{C} are the right-singular vectors of \mathbf{X} .
- The PC directions $\phi_1, \phi_2, \phi_3, \dots$ are the right-singular vectors of \mathbf{X} .

Example: National track records

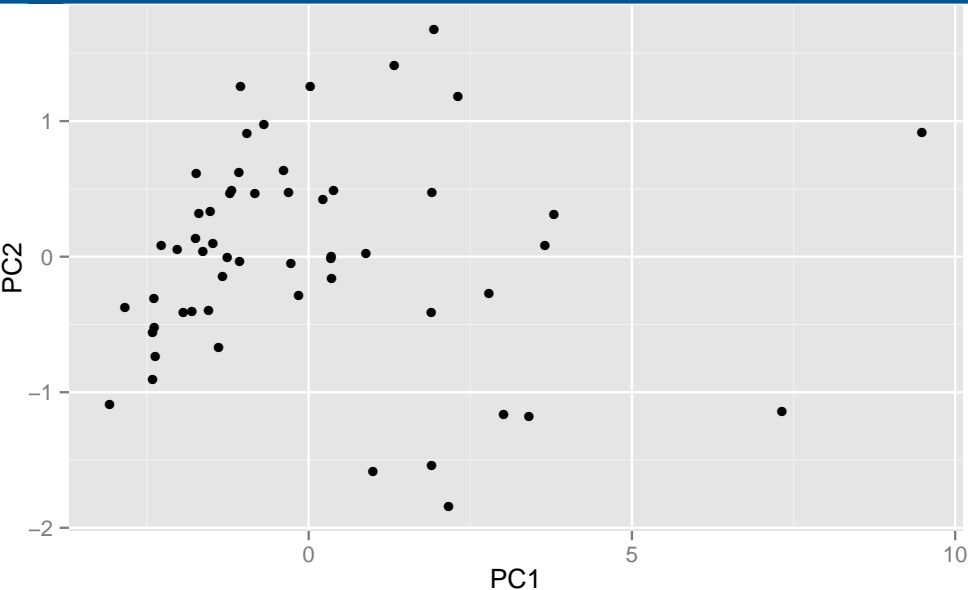
The data on national track records for men are listed in the following table (as at 1984):

Country	100m	200m	400m	800m	1500m	5000m	10000m	Marathon
	(s)	(s)	(s)	(min)	(min)	(min)	(min)	(min)
Argentina	10.39	20.81	46.84	1.81	3.70	14.04	29.36	137.72
Australia	10.31	20.06	44.84	1.74	3.57	13.28	27.66	128.30
Austria	10.44	20.81	46.82	1.79	3.60	13.26	27.72	135.90
Belgium	10.34	20.68	45.04	1.73	3.60	13.22	27.45	129.95
Bermuda	10.28	20.58	45.91	1.80	3.75	14.68	30.55	146.62
Brazil	10.22	20.43	45.21	1.73	3.66	13.62	28.62	133.13
⋮								
Turkey	10.71	21.43	47.60	1.79	3.67	13.56	28.58	131.50
USA	9.93	19.75	43.86	1.73	3.53	13.20	27.43	128.22
USSR	10.07	20.00	44.60	1.75	3.59	13.20	27.53	130.55
W.Samoa	10.82	21.86	49.00	2.02	4.24	16.28	34.71	161.83

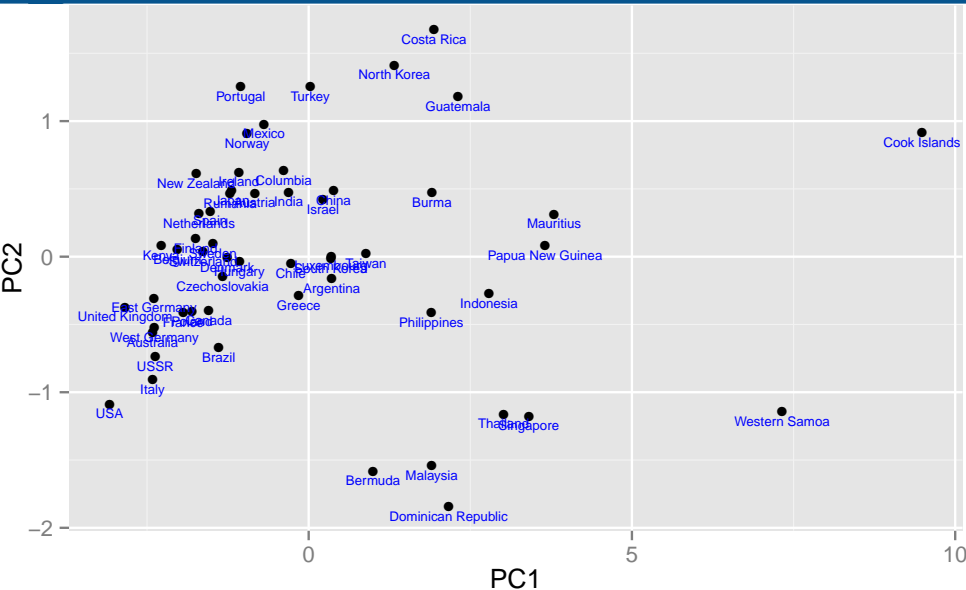
Example: National track records



Example: National track records



Example: National track records



Example: National track records

```
> pca
```

Standard deviations:

```
[1] 2.573 0.937 0.399 0.352 0.283 0.261 0.215 0.150
```

Rotation:

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8
X100m	0.318	0.5669	0.332	-0.1276	0.263	-0.5937	0.13624	-0.105542
X200m	0.337	0.4616	0.361	0.2591	-0.154	0.6561	-0.11264	0.096054
X400m	0.356	0.2483	-0.560	-0.6523	-0.218	0.1566	-0.00285	0.000127
X800m	0.369	0.0124	-0.532	0.4800	0.540	-0.0147	-0.23802	0.038165
X1500m	0.373	-0.1398	-0.153	0.4045	-0.488	-0.1578	0.61001	-0.139291
X5K	0.364	-0.3120	0.190	-0.0296	-0.254	-0.1413	-0.59130	-0.546697
X10K	0.367	-0.3069	0.182	-0.0801	-0.133	-0.2190	-0.17687	0.796795
Marathon	0.342	-0.4390	0.263	-0.2995	0.498	0.3153	0.39882	-0.158164

Proportion of variance explained

Total variance in data (assuming variables centered at 0):

$$TV = \sum_{j=1}^p \text{Var}(x_j) = \sum_{j=1}^p \frac{1}{n} \sum_{i=1}^n x_{ij}^2$$

Variance explained by m th PC:

$$V_m = \text{Var}(z_m) = \frac{1}{n} \sum_{i=1}^n z_{im}^2$$

$$TV = \sum_{m=1}^M V_m \quad \text{where } M = \min(n - 1, p).$$

Proportion of variance explained:

$$PVE_m = V_m / TV$$

Scree plots and biplots

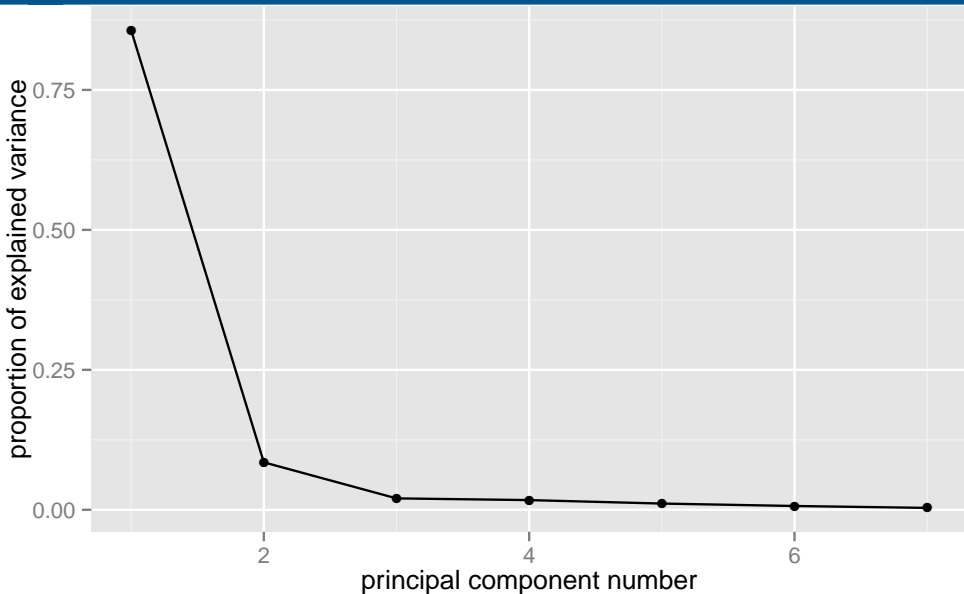
Scree plot

Plot of variance explained by each component vs number of component.

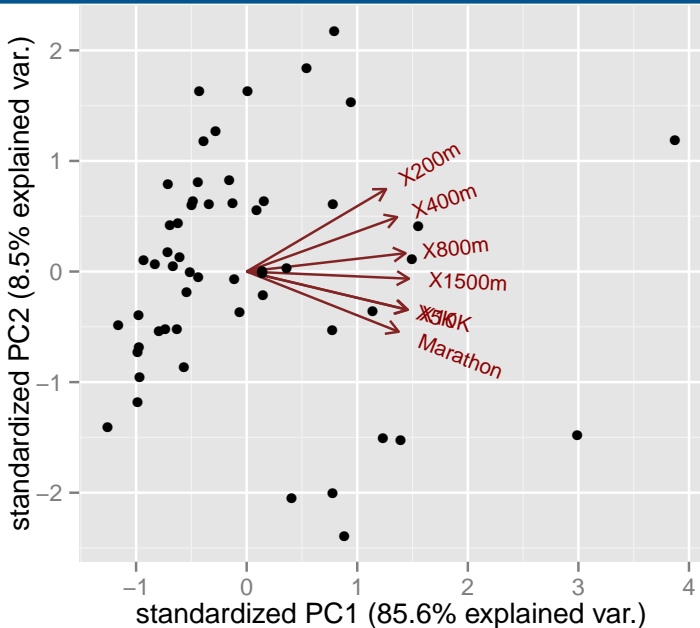
Biplot

Plot of PC2 vs PC1, overlaid with directions of the loading vectors (ϕ_{j1}, ϕ_{j2}).

Scree plot



Biplot



Scaling

- If the variables are in different units, scaling each to have standard deviation equal to one is recommended.
- If they are in the same units, you might or might not scale the variables.