Principal Component Analysis

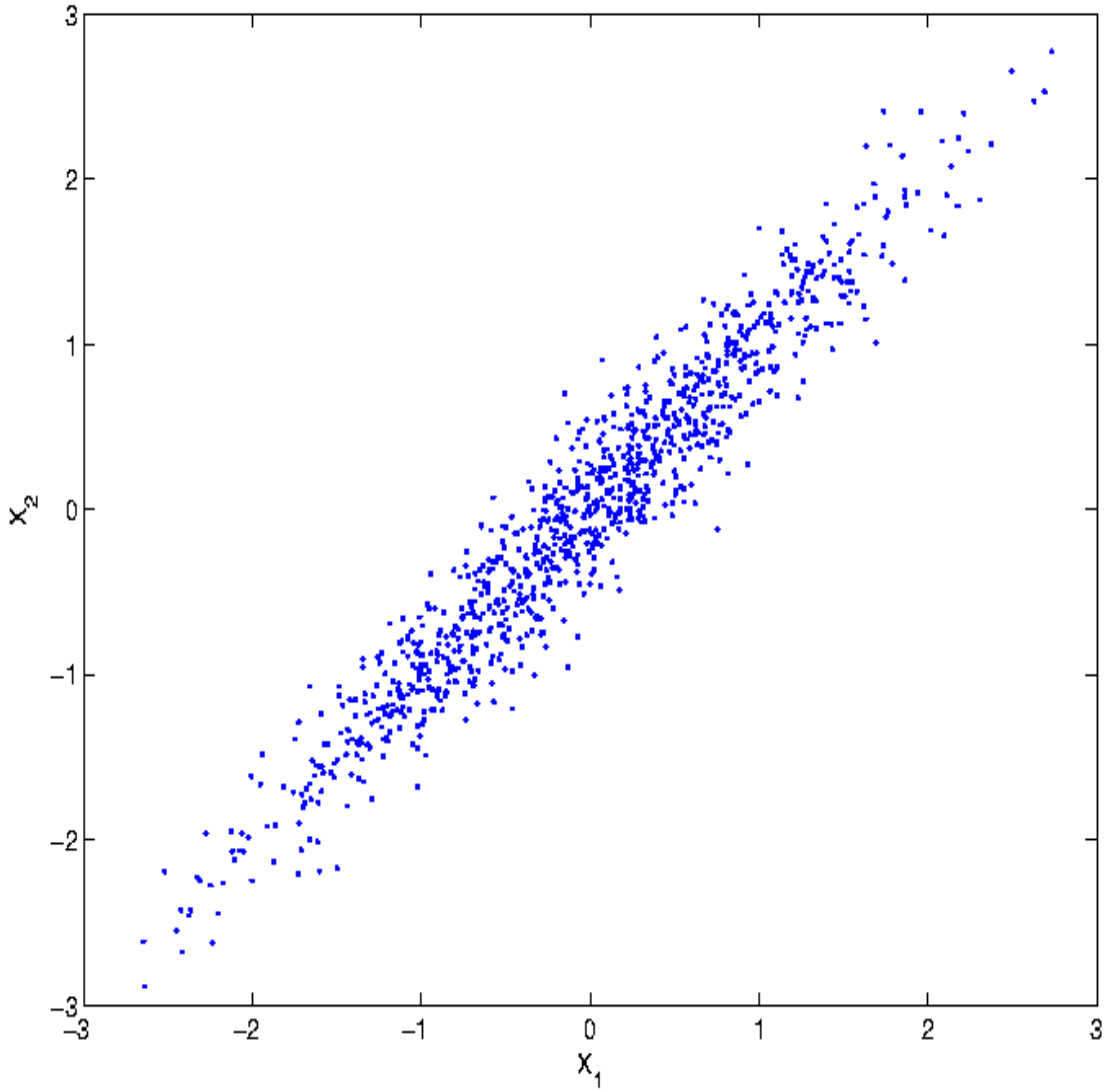
Stats 503

Prof. Liza Levina

Motivation

- The main objective: reduce dimensionality of the data set.
- Replaces the original p variables with $k < p$ linear combinations of the original variables that are a "good representation" of the data (a linear dimension reduction method)
- Belongs to the class of projection methods
- Useful for
  - visualization (project to 2-d or 3-d)
  - as a pre-processing step for other methods that do not deal well with an excessive number of variables (principle component regression (PCR), classification based on principle components)

A Toy Example:



Question: What is a good 1-dim projection of the data?

Some Possibilities

- Could use 1 (2,3,...) of the variables (e.g. X_1 in the toy example). But what if there are many thousands of variables?
- Better idea: use a linear combination of the variables; i.e. a weighted average of the variables. In the toy example,

$$Y_1 = w_1 X_1 + w_2 X_2.$$

- Main issue: what is a good choice for the weights w_i?
- Need a criterion for choosing the weights.

The Criterion for Principal Components

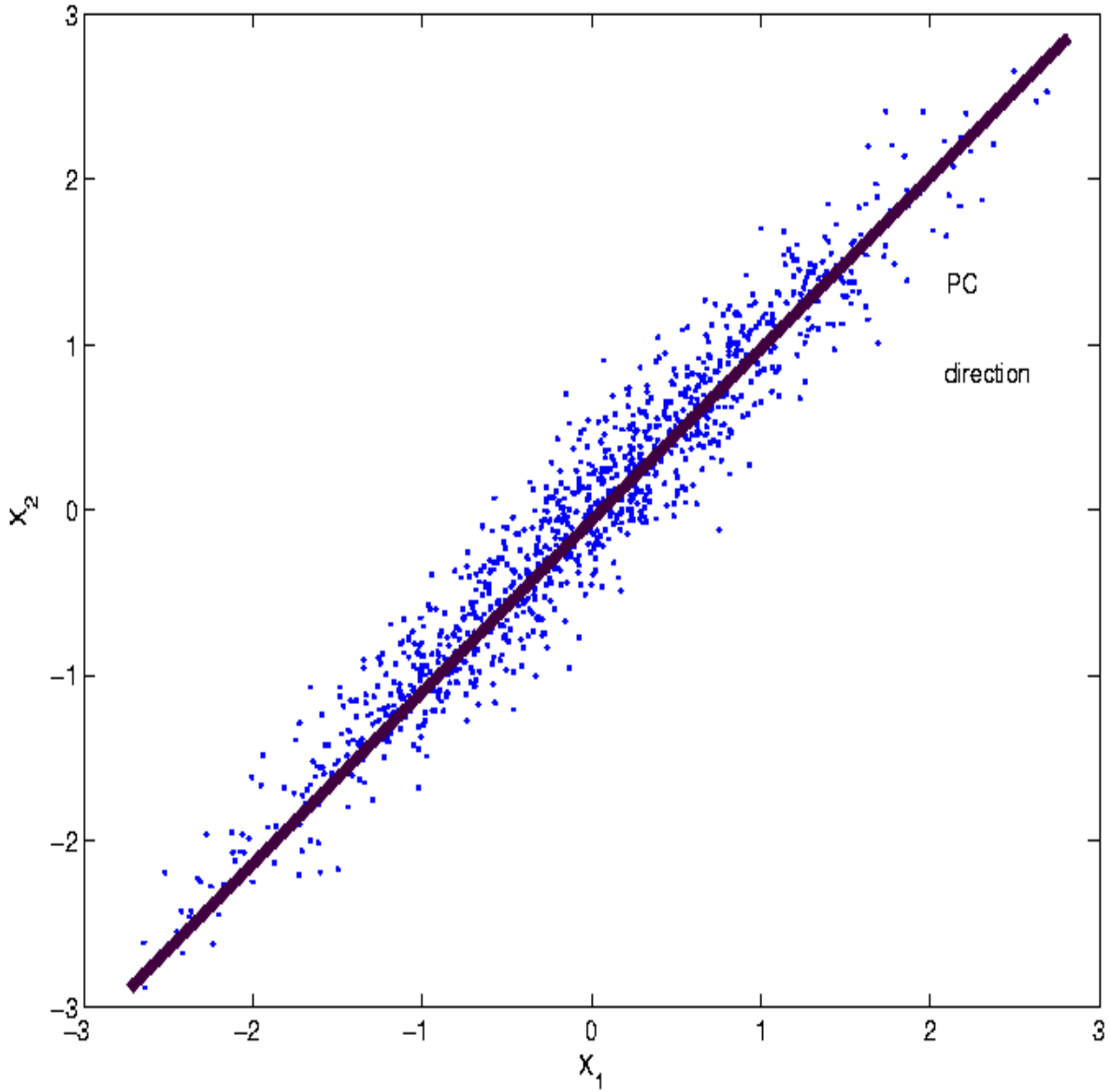
PCA finds the direction vector w that maximizes

$$\max_{w:w^T w=1} \text{Var}\left(\sum_i w_i X_i\right) = \max_{w:w^T w=1} w^T \Sigma w$$

where Σ is the covariance matrix of the random vector X. Check:

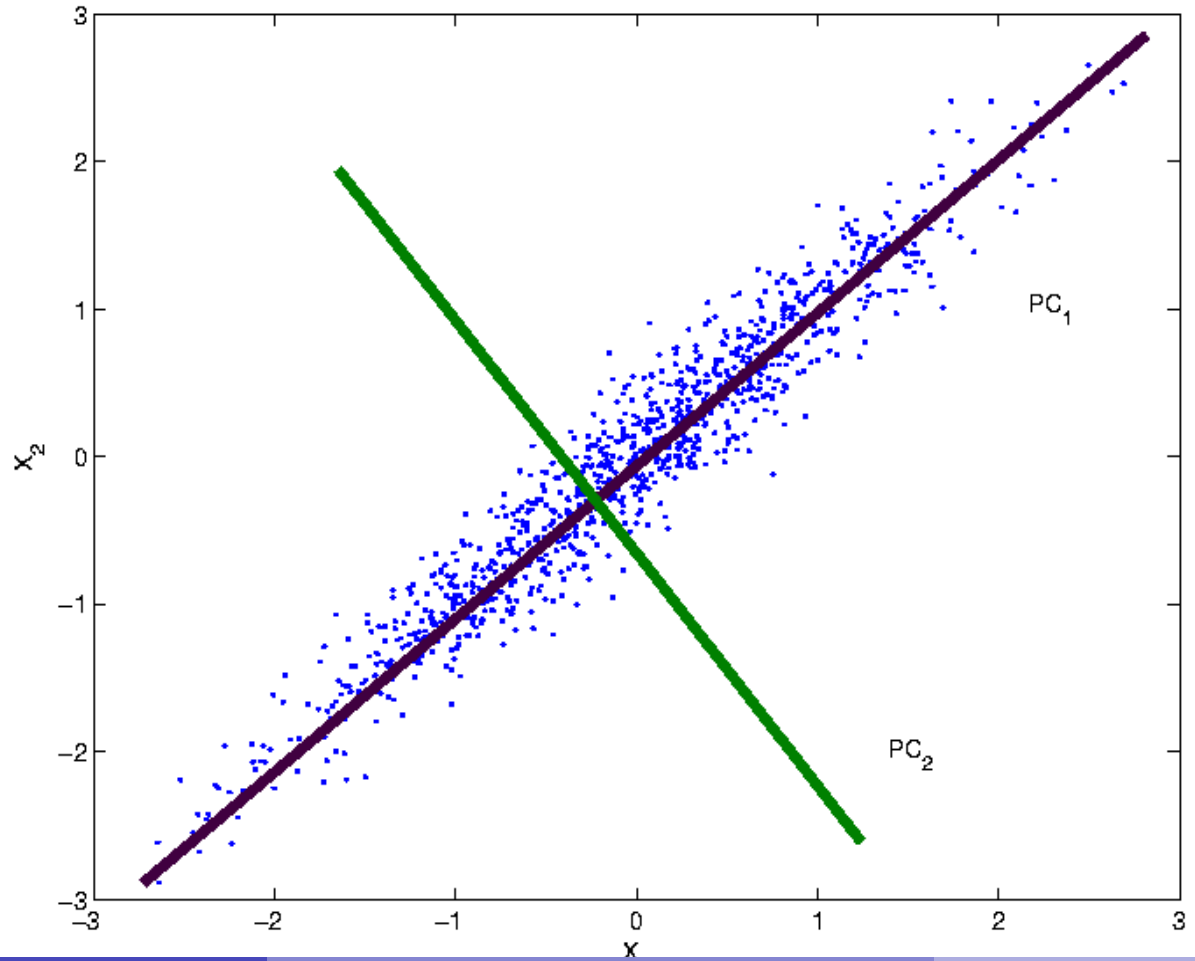
- The "interesting" direction in the data according to the PCA criterion is the one that captures the most variance in the data.

Toy Example: 1-dim PCA solution



Toy Example: 2-dim PCA solution

- What if we wanted a second linear combination, i.e. $Y_2 = v_1 X_1 + v_2 X_2 = v^T X$?
- Require subsequent linear combinations to be orthogonal to previous ones.



Mathematical Formulation of PCA

- Assume that the variables have been centered (i.e. E(X) = 0, or for the data the columns have mean 0).
- The problem: Find p new variables $Y_1, Y_2, \cdots, Y_p$ that are linear combinations of the original variables $X_1, X_2, \cdots, X_p$ (i.e. $Y_i = \sum_{j=1}^{p} w_{ij} X_j$) that maximize

$$w_i^T \Sigma w_i \text{ subject to } w_i^T w_i = 1, w_i^T w_j = 0.$$

- In matrix form: find $Y = XW$ where W solves

$$\max_{W: W^T W = I} W^T \Sigma W.$$

Solution to the PCA problem

Derive the 1st PC:

- Write the constraint $w_1^T w_1 = 1$ via Lagrange multipliers: maximize

$$f(w_1) = w_1^T \Sigma w_1 + \lambda_1 (1 - w_1^T w_1)$$

- Take derivative w.r.t. w_1 and set to 0:

$$\frac{\partial f(w_1)}{\partial w_1} =$$

- Solution: λ_1 must be one of the eigenvalues of Σ, and w_1 the corresponding eigenvector.

- Which eigenvector should we choose? Let

$$\Sigma = W\Lambda W^T,$$

where W is a $p \times p$ matrix of eigenvectors and Λ a $p \times p$ matrix of eigenvalues.

- Want to pick an eigenvector w_1 such that $w_1^T \Sigma w_1$ is as large as possible. Calculate:

- Subsequent PCs are derived similarly with extra Lagrangian constraints for orthogonality to previous PCs.
- The solution is given by the eigendecomposition of Σ:

$$\Sigma = W\Lambda W^T \ , \quad Y = XW$$

Properties of PCs

- $E(Y_i) = 0$
- $\text{Cov}(Y) = \Lambda$, hence PCs are uncorrelated. Check:

- $\text{Var}(Y_i) = \lambda_i$.
- If X is multivariate normal, so is Y, and PCs are independent.
- $\text{Cov}(X, Y) = W\Lambda$. Check:

- Then for a given Y_i we have $E(X^T Y_i) = \lambda_i w_i$, and if $\text{Var}(X_i) = 1$, then $\text{Corr}(X_i, Y_j) = \sqrt{\lambda_j} w_{ji}$.

PCA Terminology

- Let X now be the $n \times p$ data matrix (centered)
- Σ is replaced by the sample covariance matrix $\hat{\Sigma} = 1/(n-1)X^T X$.
- The vectors w_i are called PC directions
- Vectors $Y_i = Xw_i$ are called the principal components of X and are projections of the data onto the PC directions
- Components of Xw_i are also called scores
- The coordinates w_{ij} are called (factor) loadings; sometimes loadings are defined as $\sqrt{\lambda_j} w_{ij}$.

Covariance vs Correlation

- PCA is applied to the covariance matrix of centered data (mean 0)
- Alternative: standardize all variables first (mean 0, sd 1)
- This is equivalent to applying PCA to the correlation matrix
- The PCs from covariance and from correlation are not the same
- Reason to standardize: makes the analysis independent of units
- Reason to not standardize: there is information in the variance, particularly if all variables are measured on the same scale
- No universal rule or consensus on this issue

Example: Athletic Performance Data

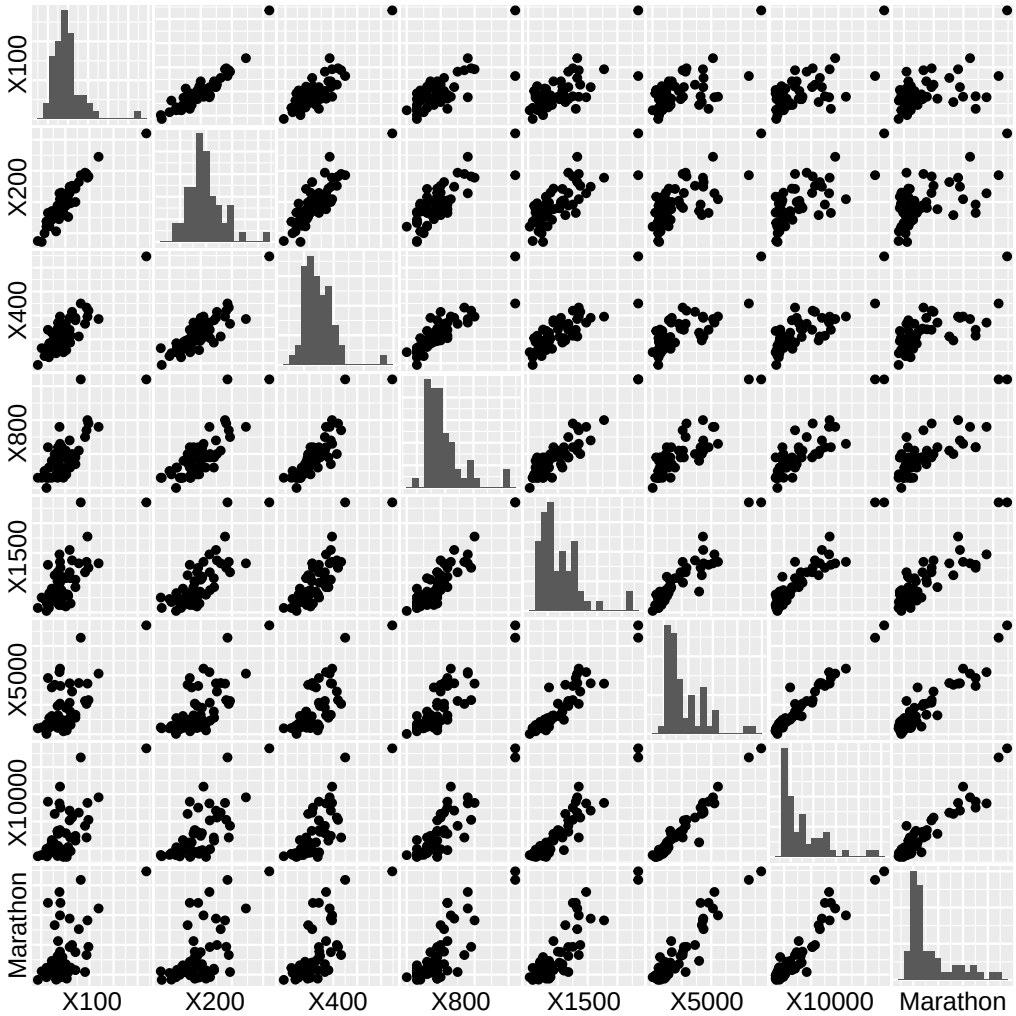
- Data come from Belsham and Hymans (1984)
- Records for 55 countries in the following men's track events: 100, 200, 400, 800, 1500, 5000, 10000 meters and the marathon
- The data are in seconds for the first three events and in minutes for the rest.

Country codes

AG=Argentina AL=Australia AR=Austria BG=Belgium BM=Bermuda
BZ=Brazil BU=Burma CD=Canada CL=Chile CH=China CO=Colombia
CI=Cook.Islands CR=Costa.Rica CS=Czechoslovakia DR=Denmark
DM=Dominican.Rep FL=Finland FR=France GD=German.Dem.Rep
GF=German.Fed.Rep GB=Great.Britain.NI GC=Greece GT=Guatemala
HU=Hungary IN=India IO=Indonesia IL=Ireland IS=Israel IT=Italy JA=Japan
KY=Kenya KS=Korea KN=Korean.DP.Rep LX=Luxemburg MA=Malaysia
MR=Mauritius MX=Mexico NL=Netherlands NZ=New.Zealand NW=Norway
PN=Papua.New.Guinea PH=Philippines PL=Poland PR=Portugal
RO=Romania SI=Singapore SP=Spain SW=Sweden SZ=Switzerland
TP=Taipei TH=Thailand TU=Turkey US=USA UR=USSR
WS=Western.Samoa

Pairwise scatterplots and histograms

```
> ggpairs(athletic.data, diag=list(continuous='barDiag'),
  upper=list(continuous="points"),
  lower=list(continuous="points"))
```

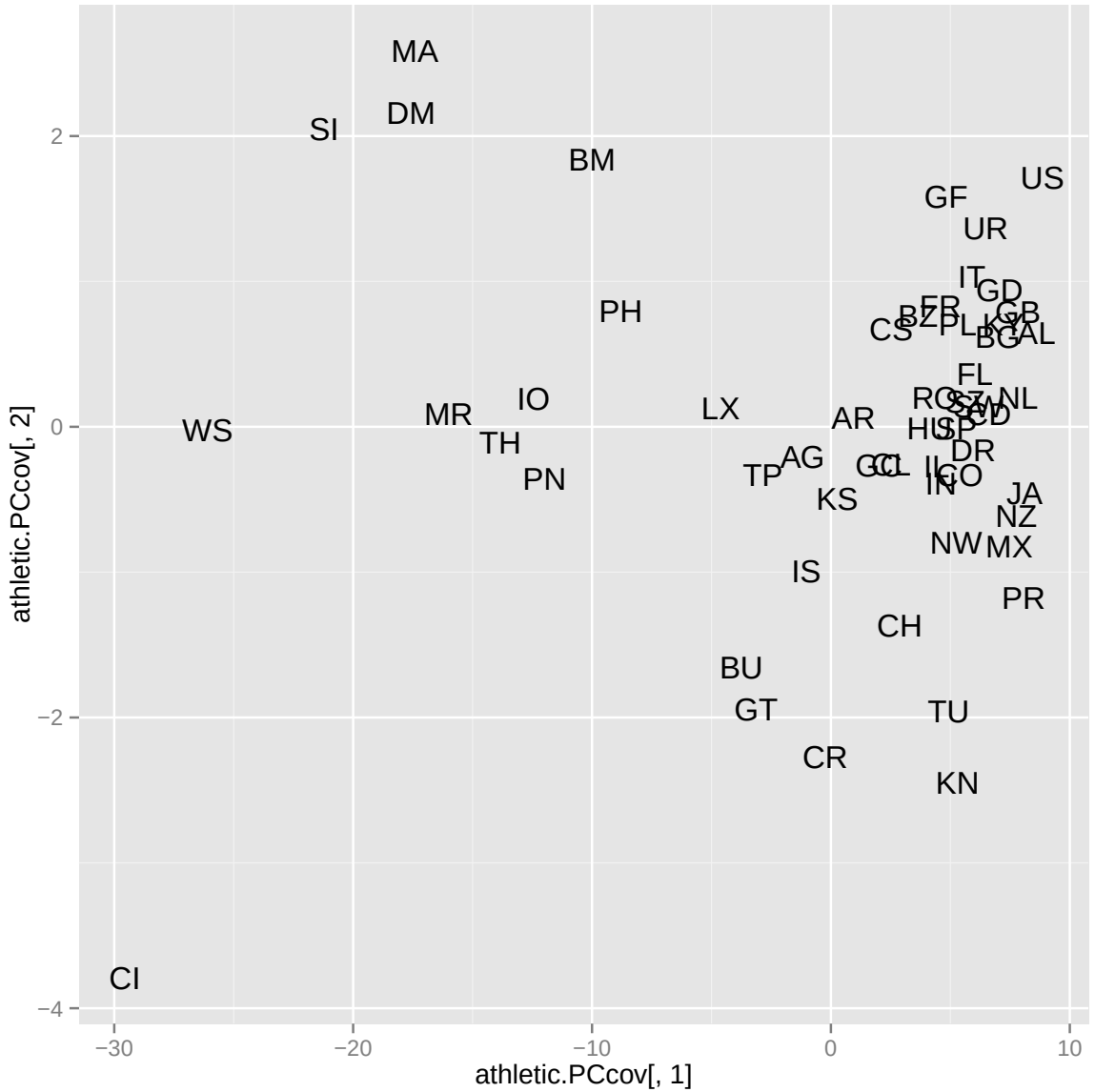


Loadings from the covariance matrix

```
> 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

```
> qplot(x = athletic.PCcov[,1],y=athletic.PCcov[,2],
  label=Code,geom = "text")
```

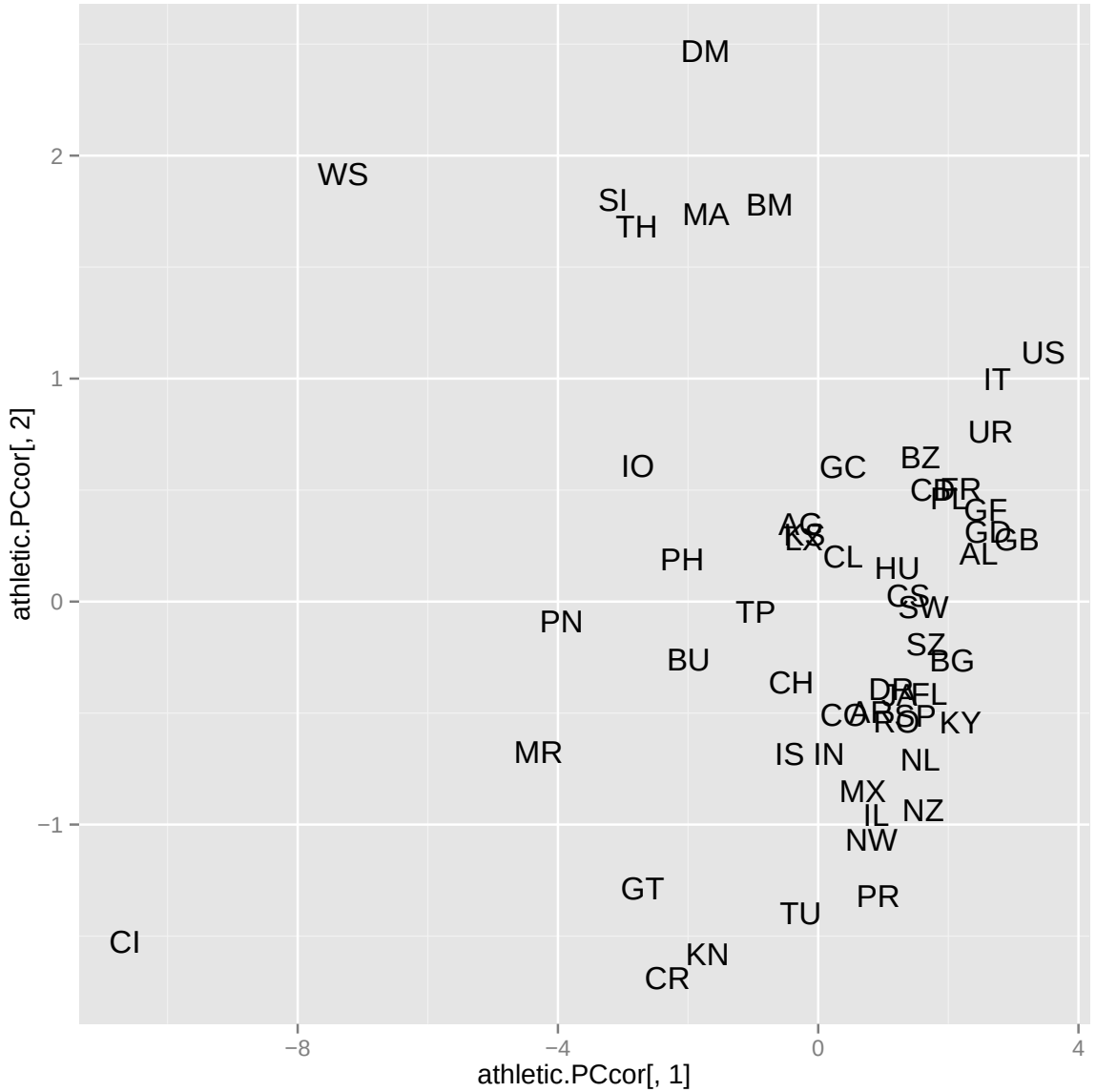


Loadings from the correlation matrix

```
> 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
```

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



How many PCs should we keep?

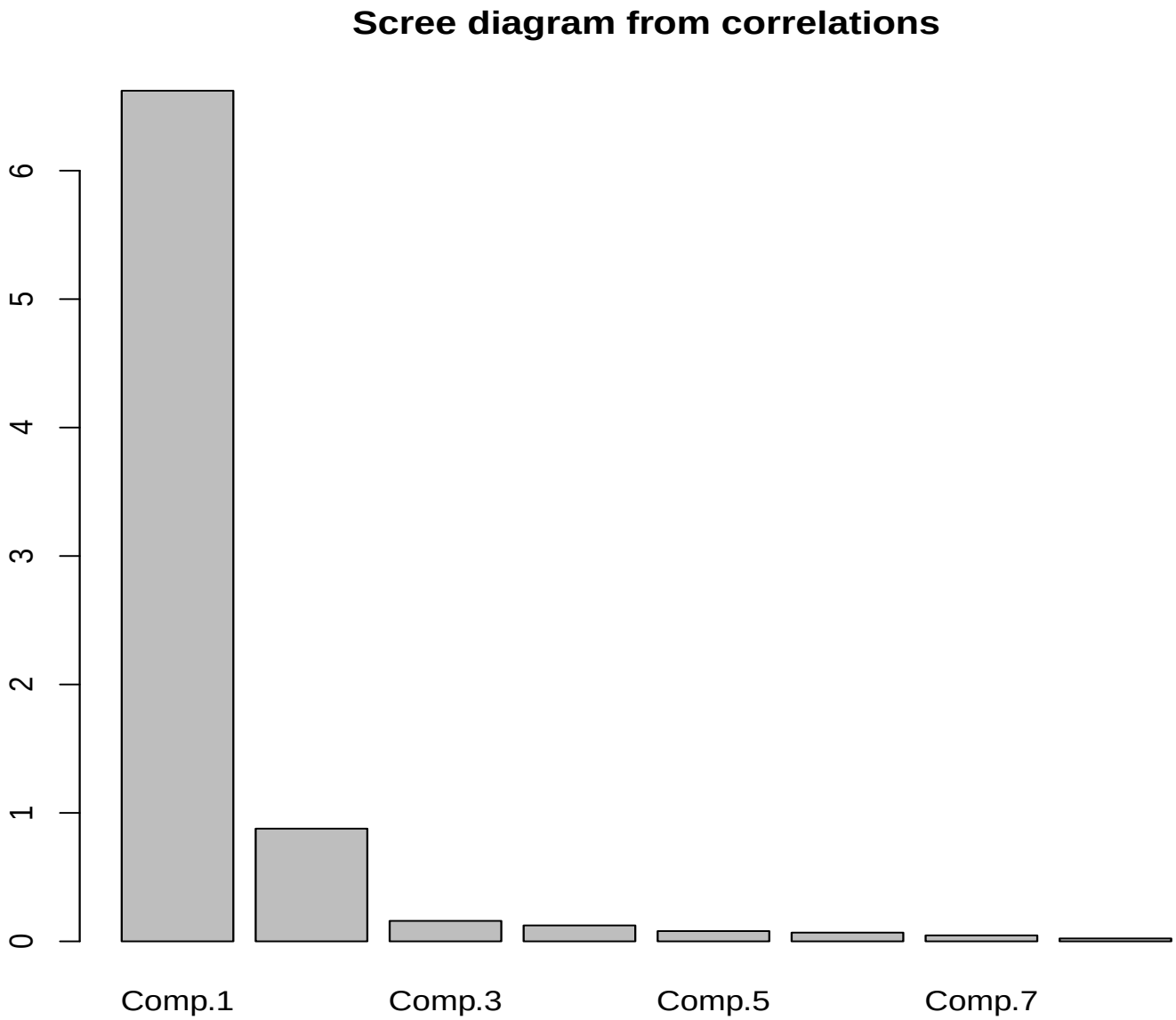
- For visualization, can only use 2 or 3
- For general dimension reduction, want to keep enough to represent the data "well"
- Scree plot: plot λ_i or $\sqrt{\lambda_i}$ against i and look for an "elbow"
- Percentage of variance explained: component i "explains" $\frac{\lambda_i}{\sum_{j=1}^{p} \lambda_j}$, so pick the first k such that

$$\frac{\sum_{j=1}^{k} \lambda_j}{\sum_{j=1}^{p} \lambda_j} \geq 1 - \alpha$$

  for some pre-specified small α (e.g. 0.1)
- Some hypothesis tests have been proposed, but no universal rule

Scree plot for athletic data

```
> barplot(athletic.PCAcor$sdev^2)
```



Computing the PCs: role of the SVD

- In practice, the covariance or correlation matrix is not used
- Let $X = ULB^T$ be the SVD of the data matrix
- Then $(n-1)\hat{\Sigma} = BL^2B^T$. Check:

- B contains the eigenvectors of the sample covariance matrix, and L^2 its eigenvalues.
- Can we also get the PC scores from the SVD?

$$Y = XW =$$

- This also shows that the data projected onto the first k PCs (setting all but the first k elements in L to 0) is the best rank k approximation to the full data matrix X in the Frobenius norm

Some other issues

- PCs with equal variance: if k eigenvalues coincide, their eigenspace is a unique k-dimensional subspace, but within that subspace PC directions cannot be distinguished
- Outliers: PCA is not a robust method, some robust versions exist
- Subsets of variables: sometimes want to express a PC in terms of just a few original variables (for ease of interpretation). Sparse variants of PCA are available (shrink many loadings to exactly 0).
- Singular Σ: then only r PCs are defined, where $r = \text{rank}(\Sigma)$. Always the case when $p > n$, since the sample covariance matrix has rank $\min(p, n-1)$.

Population vs Sample PCA

- Think of data X as an i.i.d. sample from a multivariate distribution with covariance Σ
- True PCs: eigenvectors of Σ, but we only know X, not Σ
- Estimate Σ with the sample covariance matrix (assuming centered data)

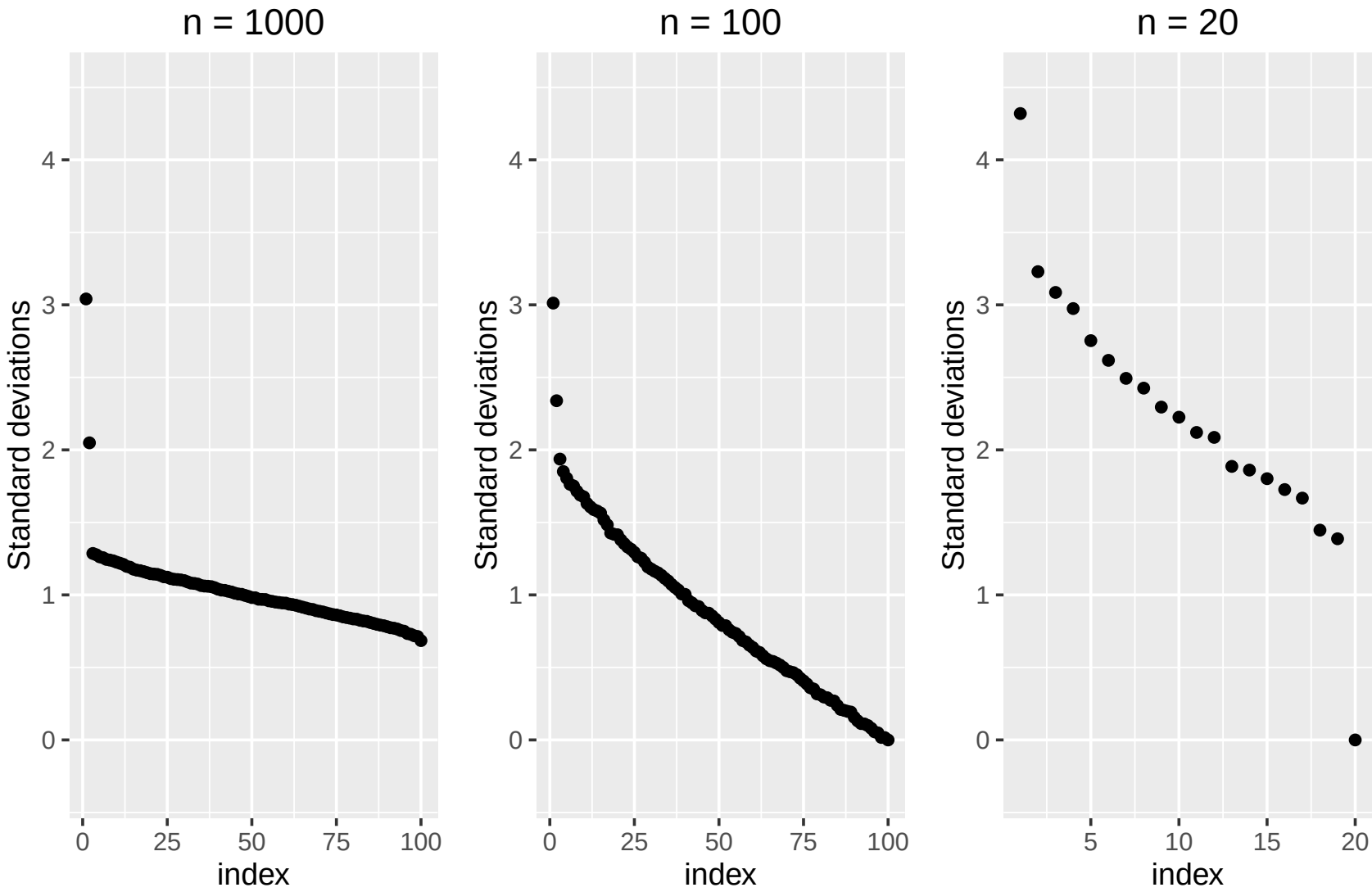
$$\hat{\Sigma} = \frac{1}{n-1} X^T X$$

- The eigenvalues and eigenvectors of $\hat{\Sigma}$ approximate those of Σ, but the quality of the approximation varies and becomes problematic when $p > n$

An example in high dimensions

- Simulate n i.i.d. observations drawn from the multivariate normal distribution with $p = 100$, mean 0, covariance $\Sigma = \text{diag}(3^2, 2^2, 1, \ldots, 1)$
- True population standard deviations, $\sqrt{\lambda_i}$: $3, 2, 1, \ldots, 1$
- True population eigenvectors: $(1, 0, 0, \ldots, 0)$, $(0, 1, 0, \ldots, 0)$, and any basis of the remaining 98-dimensional sphere
- Compute the sample covariance and plot the scree plot with different n

Sample-based scree plots



Classical Inference for Σ

- All based on the assumption X is multivariate normal
- Asymptotic results are based on p fixed, $n \to \infty$
- The distribution of $\hat{\Sigma}$ when the data are an iid sample from $\mathbf{N}(0, \Sigma)$ is called the Wishart distribution and is well studied

Implications for PCA

- This applies to p small, sample size n large
- The estimated eigenvalues $\hat{\lambda}_i$ are approximately normally distributed, $E(\hat{\lambda}_i) = \lambda_i$, and $\hat{\lambda}_i \to \lambda_i$ as $n \to \infty$.
- The estimated eigenvectors also converge to the truth
- All $\hat{\lambda}_i$ are independent from each other, and the eigenvalues are independent from eigenvectors (asymptotically)
- This can be used to construct confidence intervals for λ_i

Computational inference for PCA

Motivation:

- Asymptotics do not work well if sample size n is small or moderate relative to number of variables p
- The normal assumption for the data may not hold

Goals:

- Obtain standard errors and confidence intervals for estimated parameters, e.g. eigenvalues
- Test whether there is any structure in the data (i.e., are the estimated eigenvalues significantly larger than they would be if all variables were independent?)

Fundamentals of Bootstrap

- Applied when there is no theory to compute standard errors, confidence intervals, etc, or the theory cannot be trusted
- May not always work either, but works quite generally
- Fundamental idea: pretend the observed data is the population
- Resample observed data, create multiple samples
- From each sample, estimate parameters and assess variability

Bootstrap vs Simulation

Simulation:

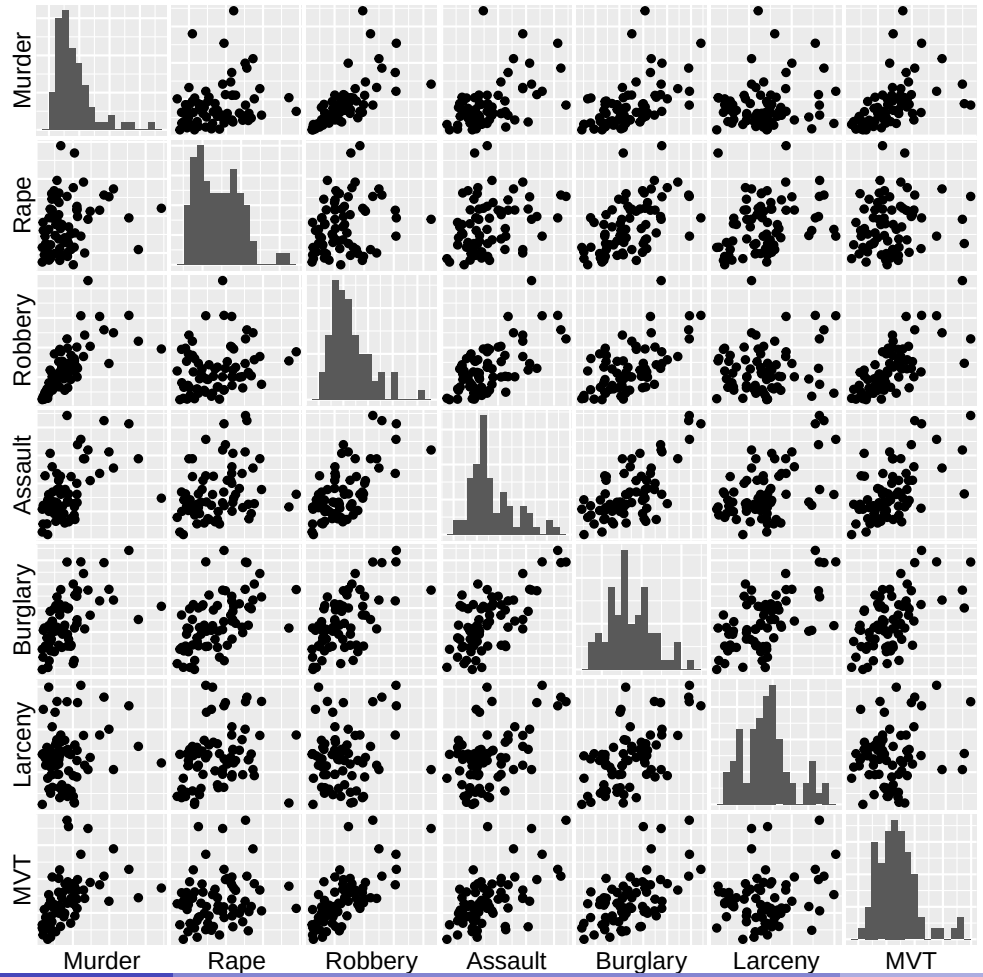
- Assume X are $N(0, \Sigma)$ with known Σ
- Generate n i.i.d. copies of X, perform PCA, and repeat the whole process N times
- To assess behavior of, e.g., $\hat{\lambda}_1$, compute mean and SD of $\hat{\lambda}_1^{(1)}, \ldots, \hat{\lambda}_1^{(N)}$ over N replications and compare to the true λ_1 computed from true Σ
- Useful for testing new methodology and asymptotics

Bootstrap:

- Sample with replacement from data $X_1, \ldots, X_n$ to create new "sample" $X_1^*, \ldots, X_n^*$
- Compute parameter of interest, e.g. $\hat{\lambda}_1^*$, and repeat the whole process N times
- Compute mean and SD of $\hat{\lambda}_1^{*(1)}, \ldots, \hat{\lambda}_1^{*(N)}$ over N replications and compare to $\hat{\lambda}_1$ computed from $\hat{\Sigma}$ of the original sample
- Useful for computing SEs and CIs when there is no other way

Example: City crime data

```
> data_crime <- read.table("citycrime.txt", header = TRUE)
> ggpairs(data_crime,diag=list(continuous='barDiag'),
  upper = list(continuous = "points"),
  lower = list(continuous = "points"))
```



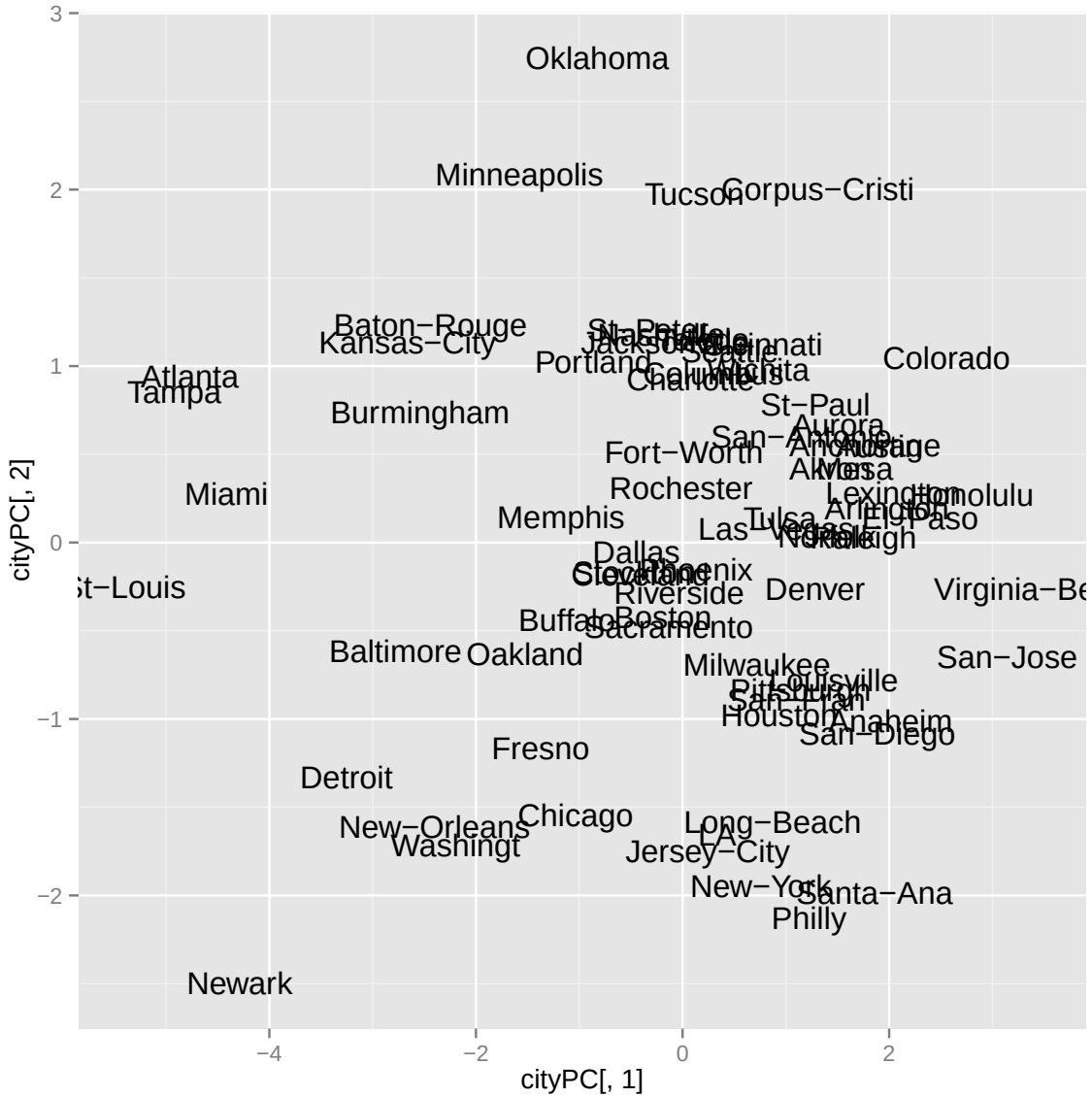
Loadings

```
> 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
```

Projection onto the first two PCs

```
> qplot(x = cityPC[,1], y=cityPC[,2],
  label=row.names(data_crime),geom = "text")
```



Variance explained

```
> summary(city.PCA)
```

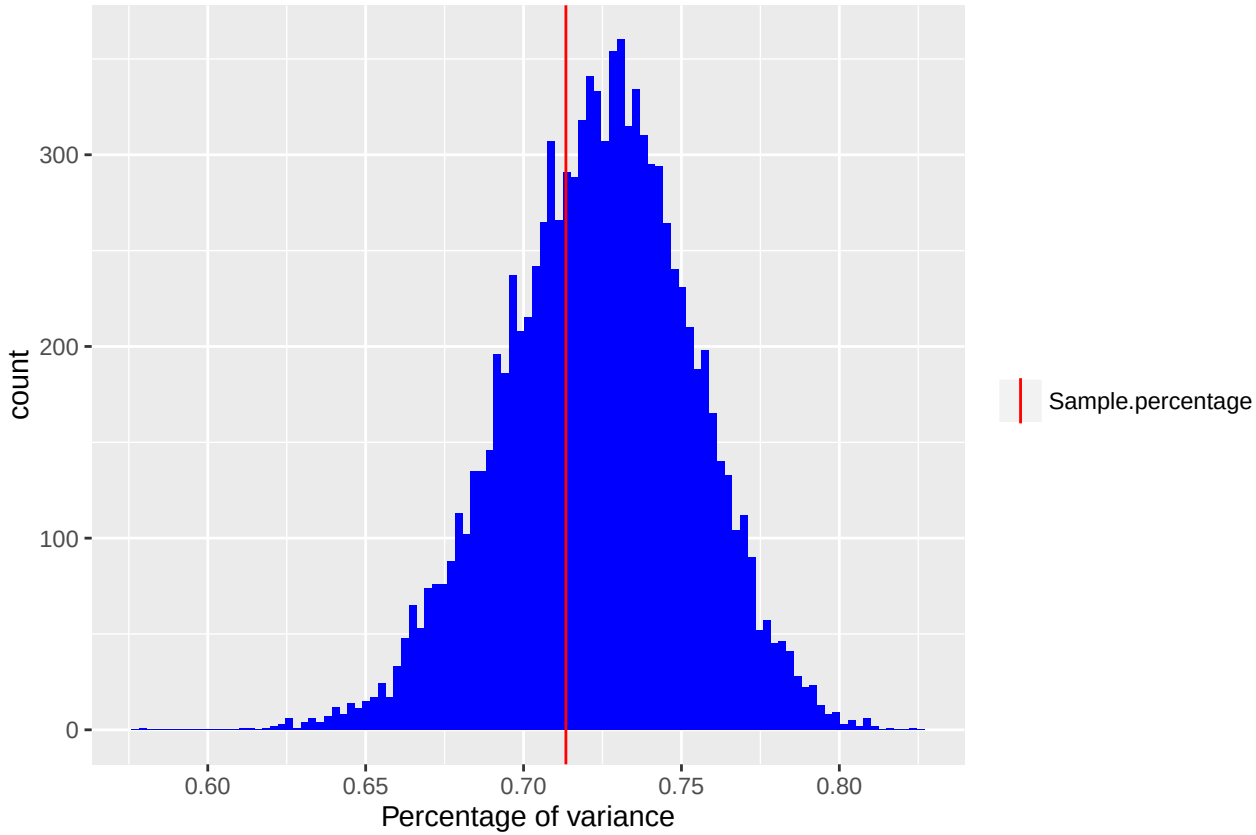
	Comp. 1	Comp. 2	Comp. 3	Comp. 4
Standard deviation	1.948	1.095	0.877	0.714
Proportion of Variance	0.54	0.17	0.11	0.073
Cumulative Proportion	0.54	0.71	0.82	0.89

A question: What is the uncertainty in how much variance the first two components explain ($0.71 \pm ??$)

Bootstrap for percentage of variance explained

- Create new data X^* by resampling rows of X, with replacement
- Compute the variance explained by the first two PCs of X^* (call this quantity $\hat{\theta}^*$)
- Repeat N times and plot a histogram of all N $\hat{\theta}^*$'s
- To get a formal 95% confidence intervals, can take the 2.5-th and 97.5-th percentiles of $\hat{\theta}^*$

Bootstrap histogram of variance explained by first two PCs



$$71\% \pm 5\%$$

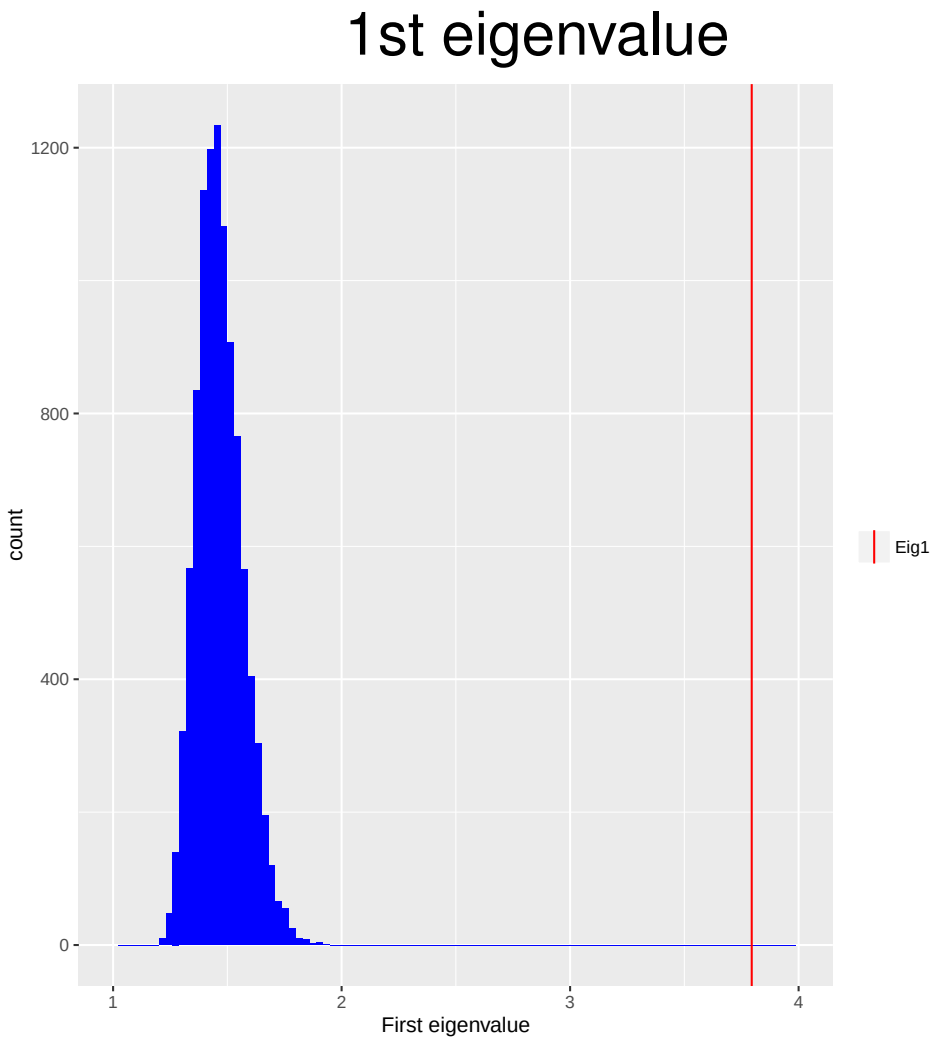
Testing for structure

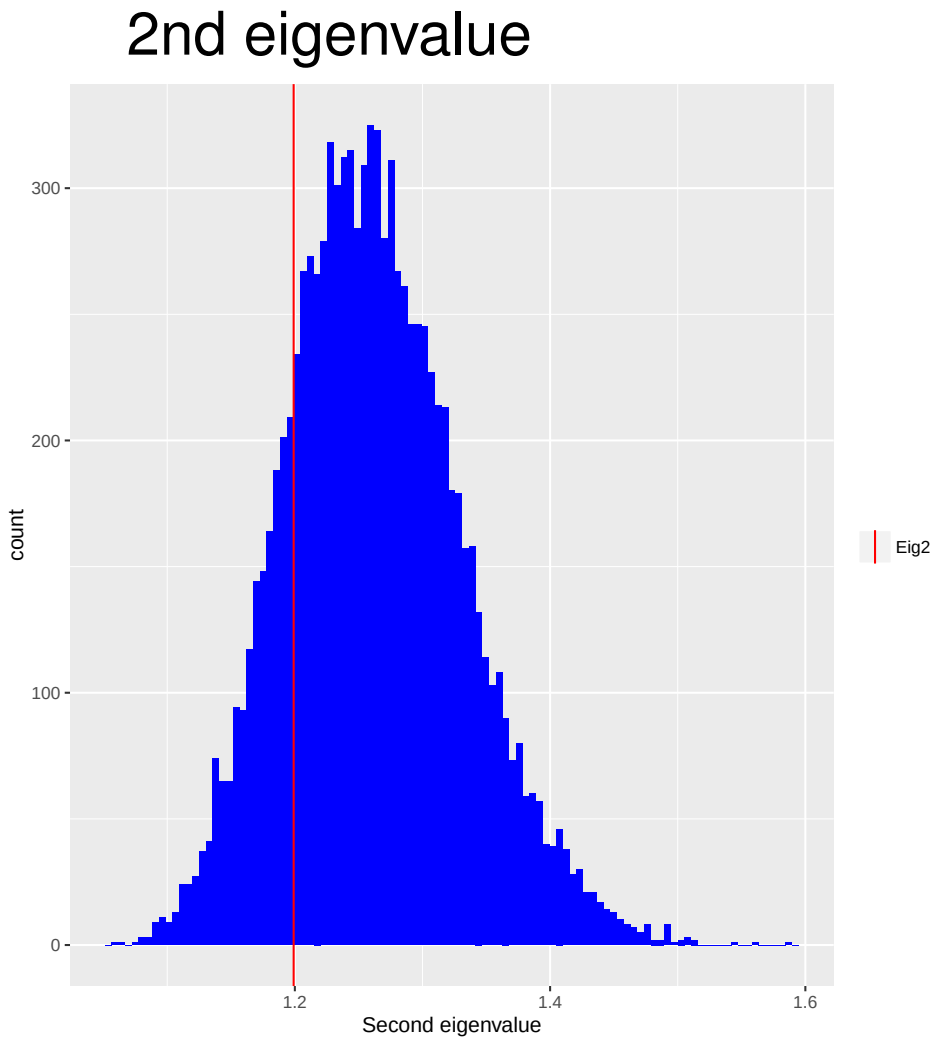
- “No structure” means the distribution is spherical
- Equivalently (for PCA based on the correlation matrix of the multivariate normal), the variables are independent, and the eigenvalues are no larger than independent data would generate
- This may be the case for smaller eigenvalues while the first few correspond to “real structure”

Permutation test for eigenvalues

- Create X^* by randomly permuting each column of X
- Calculate all eigenvalues $\hat{\lambda}_1^*, \ldots, \hat{\lambda}_p^*$
- Repeat N times and construct a histogram of all replications for each eigenvalue
- If the structure is "real", the original eigenvalue $\hat{\lambda}_j$ should be larger than most (e.g., 95%) of the N permutation-based $\hat{\lambda}_j^*$'s

Permutation test for crime data





The biplot

- A way to plot both observations and variables in the same plot
- Many variants exist
- Based on the SVD of the centered data matrix:

$$X = ULB^T = UL^\gamma \cdot L^{1-\gamma}B^T = GH^T$$

- For any $0 \leq \gamma \leq 1$, plotting G and H on the same plot gives a biplot
- Most commonly used values are $\gamma = 0$ and $\gamma = 1$; the standard PCA biplot in R has $\gamma = 1$.

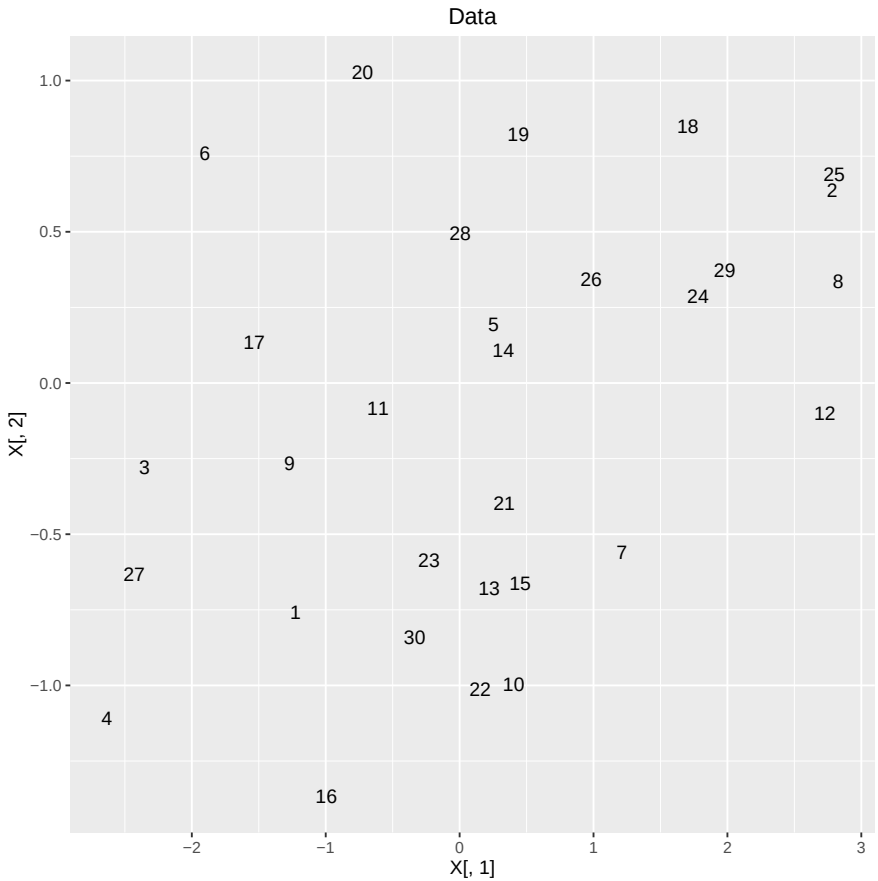
Properties of the standard PCA biplot

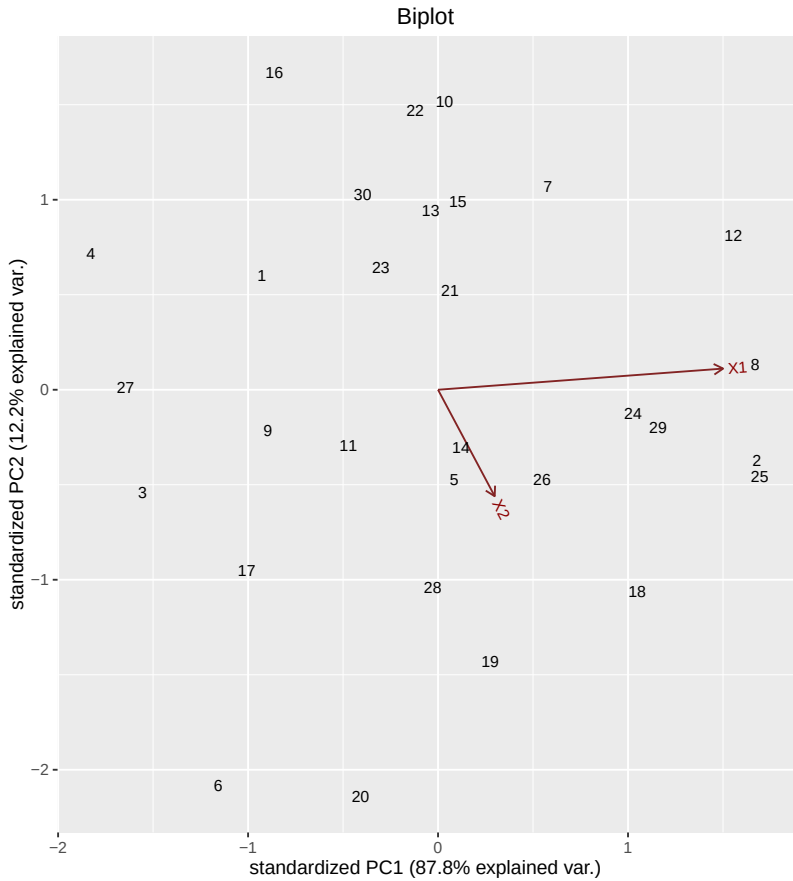
- The variables are represented by directions
- Projecting points onto these directions gives you an idea about the original variable values
- (Cosines of) angles between directions are proportional to correlations between variables
- If the first two components don't explain most of the variance, be cautious in interpretation of the biplot

Toy example

- Two variables X_1, X_2, $\mathrm{Var}(X_1) = 2$, $\mathrm{Var}(X_2) = 1$, $\mathrm{Corr}(X_1, X_2) = 0.5$
- $n = 30$ i.i.d. points
- Compare the data plot to the biplot

Toy example plots





Example: Applicant's qualifications

- A dataset (Kendall 1980) on scores of job applicants for a certain post
- Each score is on the scale from 1 to 10
- There are $p = 15$ variables (experience, potential, etc), and $n = 48$ applicants

Variable loadings for the first 4 PCs

```
> loadings(applicantsPCA)[,1:4]

                Comp.1  Comp.2  Comp.3  Comp.4
Formof.letter -0.1624  0.4288 -0.3154  0.0943
Appearance    -0.2131 -0.0353  0.0229 -0.2622
Academic      -0.0402  0.2369  0.4305 -0.6363
Likeability   -0.2251 -0.1298 -0.4658 -0.3454
Self.conf     -0.2905 -0.2489  0.2410  0.1728
Lucidity      -0.3149 -0.1310  0.1500  0.0710
Honesty       -0.1581 -0.4054 -0.2839 -0.4165
Salesmanship  -0.3243 -0.0295  0.1860  0.1982
Experience    -0.1341  0.5531 -0.0826 -0.0678
Drive         -0.3151  0.0462  0.0796  0.1560
Ambition      -0.3180 -0.0682  0.2087  0.1993
Grasp         -0.3315 -0.0232  0.1171 -0.0747
Potential     -0.3333  0.0223  0.0725 -0.1881
Keen.to.join  -0.2592 -0.0823 -0.4672  0.2014
Suitability   -0.2360  0.4207 -0.0892  0.0199
```

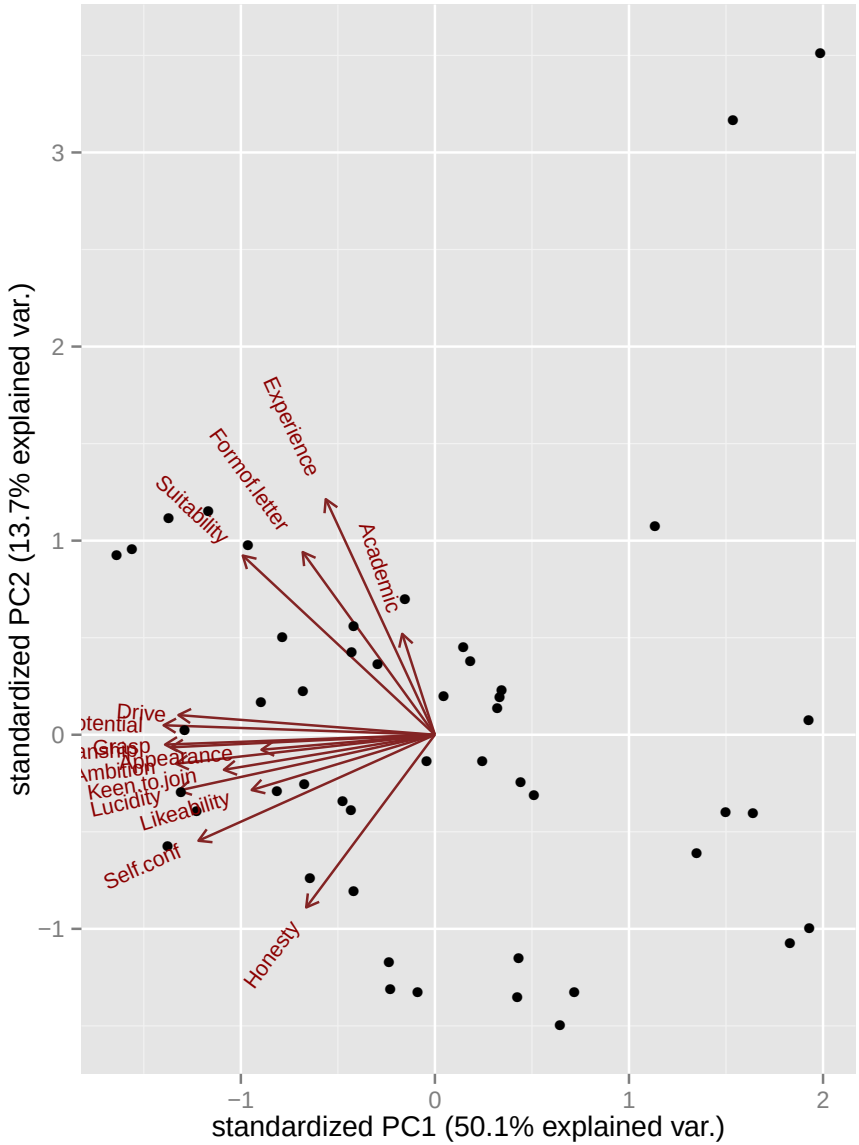
Importance of components

```
> summary(applicantsPCA)

Importance of components:
                          Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
Standard deviation         2.741  1.434 1.2066 1.0945 0.8597
Proportion of Variance     0.501  0.137 0.0971 0.0799 0.0493
Cumulative Proportion      0.501  0.638 0.7351 0.8149 0.8642
                          Comp.6 Comp.7 Comp.8 Comp.9 Comp.10
Standard deviation         0.703 0.5927 0.5567 0.5069  0.4300
Proportion of Variance     0.033 0.0234 0.0207 0.0171  0.0123
Cumulative Proportion      0.897 0.9206 0.9412 0.9584  0.9707
                          Comp.11 Comp.12 Comp.13 Comp.14
Standard deviation         0.3907  0.3124 0.29802 0.25423
Proportion of Variance     0.0102  0.0065 0.00592 0.00431
Cumulative Proportion      0.9809  0.9874 0.99331 0.99762
                          Comp.15
Standard deviation        0.18901
Proportion of Variance    0.00238
Cumulative Proportion     1.00000
```

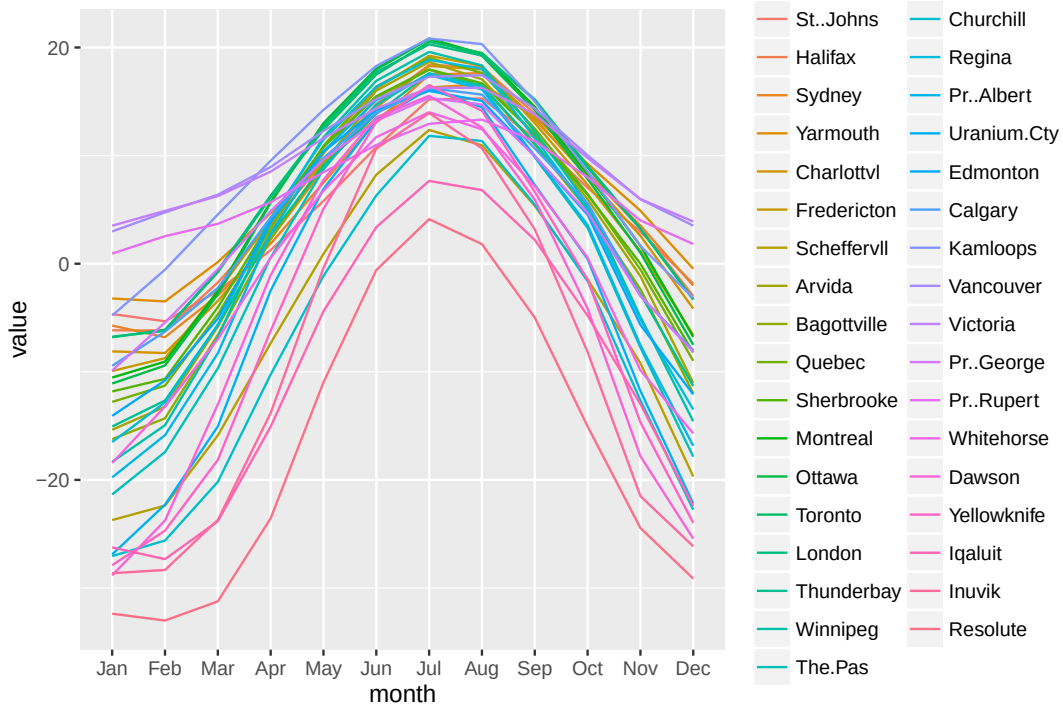
Biplot for applicants data

```
> ggbiplot(applicantsPCA)
```



Example: Canadian temperatures

- Average monthly temperatures (in degrees Celsius)
- $p = 12$ (months), $n = 35$ (weather stations in Canada)
- This is time series data (months are ordered). Specialized variants of PCA for functional data are available, but regular PCA can be applied too.



Percentage of variance explained

```
> weatherPCA = princomp(canadian_weather,cor = T)
> summary(weatherPCA)

Importance of components:
                         Comp.1 Comp.2 Comp.3
Standard deviation        3.195  1.208 0.4800
Proportion of Variance    0.851  0.122 0.0192
Cumulative Proportion     0.851  0.972 0.9915
```

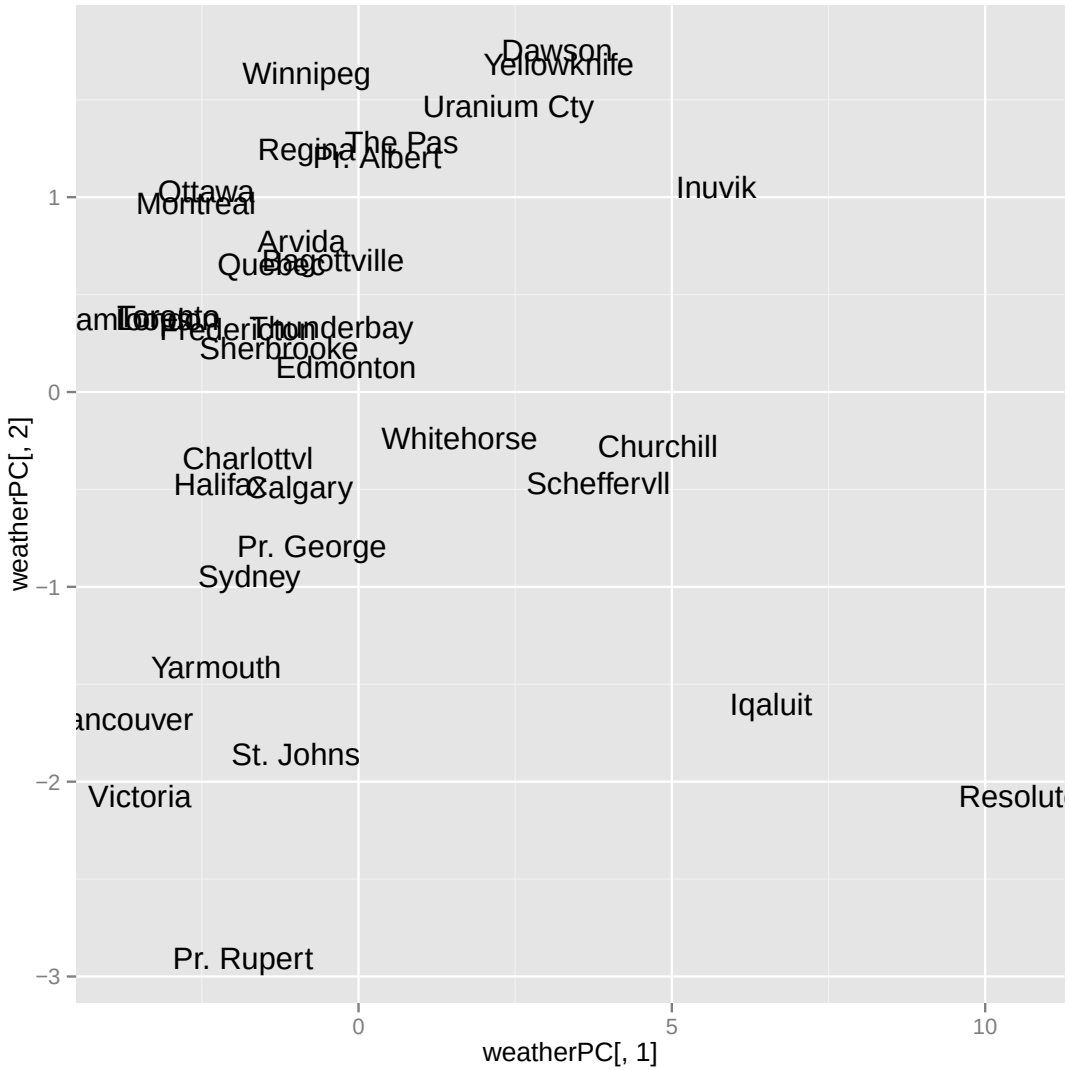
Loadings

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

     Comp.1  Comp.2   Comp.3
Jan -0.274 -0.3890  0.00253
Feb -0.285 -0.3143 -0.29288
Mar -0.302 -0.1544 -0.35286
Apr -0.304  0.0695 -0.42351
May -0.291  0.2592 -0.37013
Jun -0.266  0.4244 -0.13737
Jul -0.261  0.4401  0.22378
Aug -0.288  0.2926  0.29102
Sep -0.309  0.0718  0.27528
Oct -0.307 -0.0590  0.24294
Nov -0.292 -0.2366  0.40089
Dec -0.280 -0.3606  0.15555
```

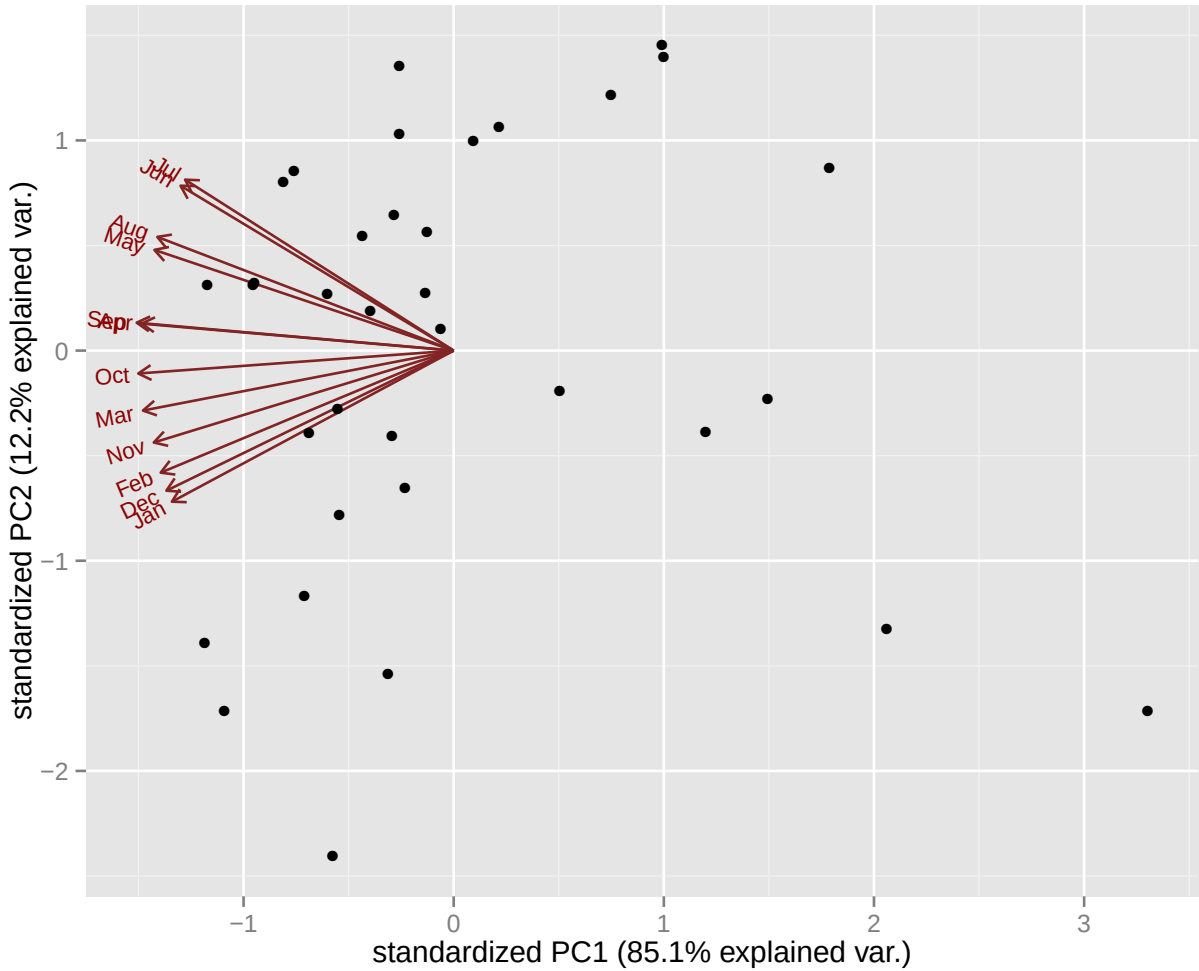
Projections onto the first 2 PCs

```
> qplot(x = weatherPC[,1],y=weatherPC[,2],
  label=row.names(canadian_weather),geom = "text")
```



Biplot for Canadian temperatures

```
> ggbiplot(weatherPCA)
```



Useful R functions

- `prcomp()`, `princomp()` : return all parts of PCA
- `eigen()`, `svd()` : perform eigendecomposition and singular value decomposition of any matrix
- For large matrices, computing all eigenvectors is slow (eigenvalues only is much faster); but computing just a few leading eigenvectors can still be feasible – use ARPACK interface (easy in Matlab, doable in R).
- `sample()`: resampling with replacement (for bootstrap) or without (for permutation tests)
- `biplot()` : make a PCA biplot out of an object created by `prcomp`