Outline: **Cluster Analysis**
(F. Chiaromonte)

Introduction to Statistical Learning
Chapter 10 section 3 and 4 Lab 2

Units\Features	X_1	X_2	…	X_p
Unit 1	x_{11}	x_{12}		x_{1p}
Unit 2	x_{21}	x_{22}		x_{2p}
. . .				
Unit n	x_{n1}	x_{n2}		x_{np}

p features measured on n units:

- An n x p data matrix X
- A data cloud of n points in R^p .

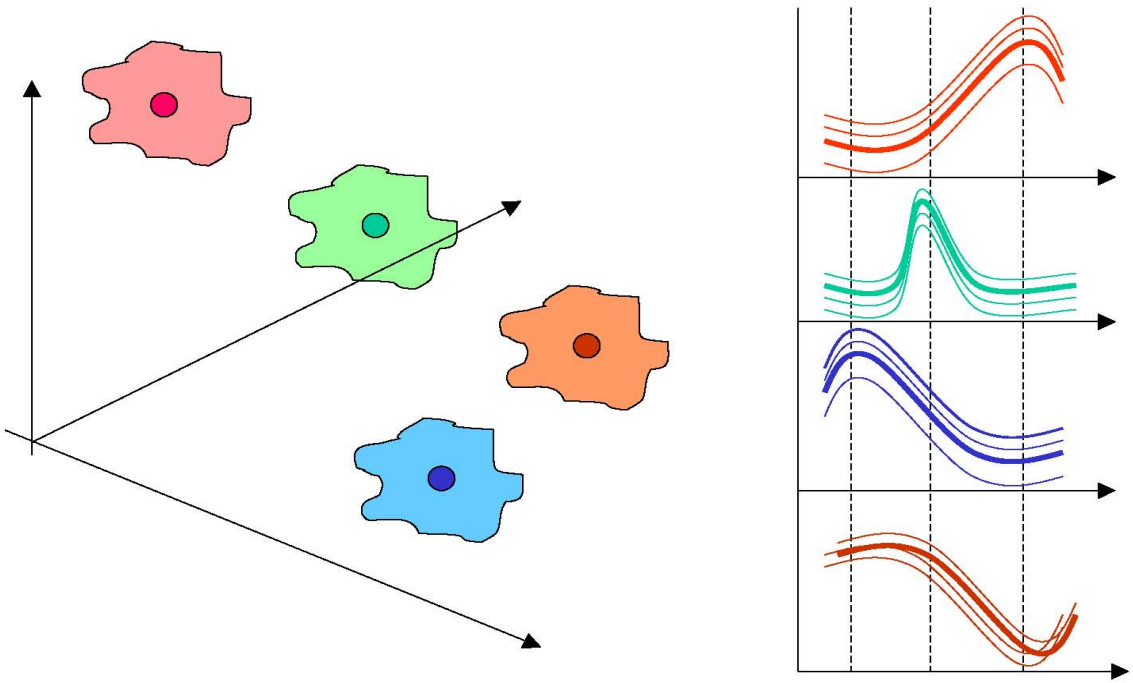
General idea: form groups of features (columns) or more commonly groups of units (rows) that are similar within groups and distinct across groups.

Unsupervised classification (define group labels on the basis of similarity).

"Discover" group structure in the data.

Simple methods determine a **hard partition**: every point belongs to one group, and one group only.

- **(Agglomerative) Hierarchical clustering**
- **K-means clustering**



Basic cartoon: partition and characteristic profiles

Pre-processing of the data matters, for instance:

• Center and/or scale by row/unit? Then a typical Euclidean distance will score as similar units with similar profiles across features, regardless of their location and scale. The same is achieved using a correlation distance.

• Center and/or scale by column/feature? (then a typical Euclidean distance will be insensitive to original differences in variability scales for different features)

(Agglomerative) Hierarchical Clustering

Set of n data points in $\boldsymbol{R}^p$.

• Choose a **distance function** for points $d(x_1,x_2)$ (Euclidean distance; correlation distance; other more complicated distances). Sometimes the point distance is not defined explicitly as a function, but provided through an nxn matrix.

• Choose a distance function for clusters, i.e. a **linkage function**, $D(C_1,C_2)$ based on a summary of the distances between points as measured by $d(x_1,x_2)$ (for clusters formed by just one points, D reduces to d).

Proceeding in an agglomerative fashion ("bottom-up"), generate a sequence of nested partitions of the data – progressively less fine:

- Start from n clusters, each containing one data point.
- At each iteration:

1. Using the current matrix of cluster distances, find the two closest clusters.
2. Update the list of clusters by merging the two closest.
3. Update the matrix of cluster distances accordingly

- Repeat until all data points are joined in one cluster.

Remarks:

- The method is sensitive to anomalous data points/outliers
- Mergers are irreversible: "bad" mergers occurring early on affect the structure of the nested sequence (path dependence).
- If two pairs of clusters are equally (and maximally) close at a given iteration, we have to choose arbitrarily; the choice will affect the structure of the nested sequence.

Defining cluster distance: the linkage function

$$D(C_1, C_2) = \min_{x_1 \in C_1, x_2 \in C_2} d(x_1, x_2)$$

Single (string-like, long)

$$D(C_1, C_2) = \frac{1}{\#(C_1)\#(C_2)} \sum_{x_1 \in C_1, x_2 \in C_2} d(x_1, x_2)$$

Average

$$D(C_1, C_2) = \max_{x_1 \in C_1, x_2 \in C_2} d(x_1, x_2)$$

Complete (ball-like, compact)

$$D(C_1, C_2) = d(\bar{x}_1, \bar{x}_2)$$

Centroid

Single and complete linkages produce nested sequences invariant under monotone transformations of *d* – not the case for average linkage. However, the latter is a compromise between "long", "stringy" clusters produced by single, and "round", "compact" clusters produced by complete. Centroid linkage can cause inversions.

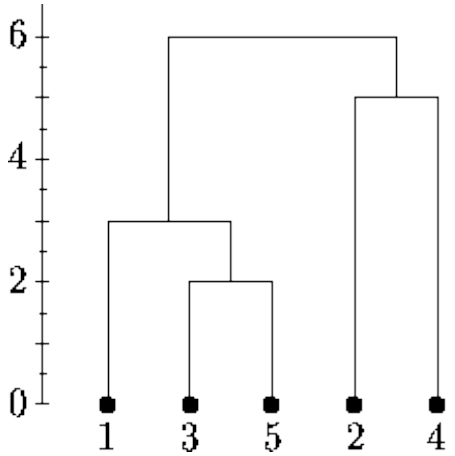
Example

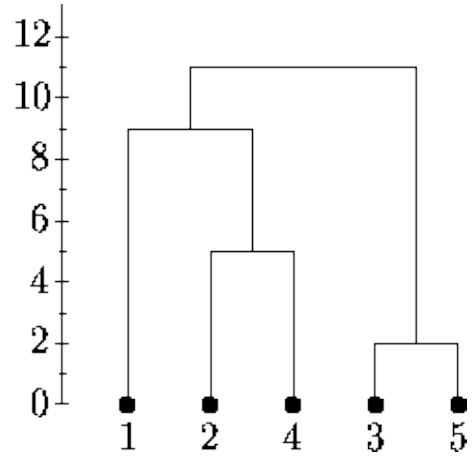
Agglomeration step in constructing the nested sequence (first iteration): Left: matrix of distances among 5 data points. 3 and 5 are the closest, and are therefore merged in cluster "35". Right: new distance matrix computed with complete linkage.

	1	2	3	4	5
1	0				
2	9	0			
3	3	7	0		
4	6	5	9	0	
5	11	10	2	8	0

	35	1	2	4
35	0			
1	11	0		
2	10	9	0	
4	9	6	5	0

Dendrogram representing the nested sequence produced by single linkage (left) and complete linkage (right). Ordinate: distance, or hight, at which each merger occurred. Horizontal ordering of the data points is any order preventing intersections of branches.





IMPORTANT: hierarchical clustering, per se, does not dictate a partition and a number of clusters; it provides a nested sequence of partitions (this is more informative than just one partition). To settle on one partition, we have to **"cut" the dendrogram**.

Loosely: read the hight associated with various numbers of clusters as a measure of "fit" of the partitioning in K groups (more later).

K-means Clustering (a partitioning algorithm).

Set of n data points in $\boldsymbol{R}^p$.

- Choose a distance function for points $d(x_1, x_2)$.
- Choose K = number of clusters.
- Initialize the algorithm; two options
    - Initialize the K cluster centroids (e.g. K data points chosen at random)
    - Initialize the labels (e.g. each data point gets a random label from {1,2…K}), then compute centroids $\bar{x}_1(0), \ldots \bar{x}_K(0)$

Use the data to iteratively relocate centroids, and reallocate points to closest centroid.

- At each iteration:

1. Compute distance of each data point from each current centroid

$$d(x, \bar{x}_k) \quad \forall x, k = 1 \ldots K$$

1. Update current cluster membership of each data point, selecting the centroid to which the point is closest

$$m(x) = \arg\min_{k=1\ldots K} d(x, \bar{x}_k) \quad \forall x$$

1. Update current centroids, as averages of the new clusters formed in (2).

$$\bar{x}_k = \frac{1}{\#(x : m(x) = k)} \sum_{x : m(x) = k} x \quad k = 1 \ldots K$$

- Repeat until cluster memberships, and thus centroids, stop changing.

Remarks:

- Also this method is sensitive to anomalous data points/outliers.
- Points can move from one cluster to another, but the final solution depends strongly on initialization (another form of path dependence). ***Run algorithm on multiple initializations*** and select solution with best outcome (smallest total within cluster sum of squares).
- If two centroids are equally (and maximally) close to an observation at a given iteration, we have to choose arbitrarily; the problem here is not so serious because points can move.
- There are several "variants" of the K-means algorithm.

Objective function

K-means converges to a local minimum of the **total within cluster sum of squares** – not necessarily a global one; will depend on initialization.

$$W(K, start) = \sum_{k=1}^{K} \sum_{x:m(x)=k} d^2(x, \bar{x}_k) \propto \sum_{k=1}^{K} \sum_{x,\tilde{x}:m(x)=k} d^2(x, \tilde{x})$$

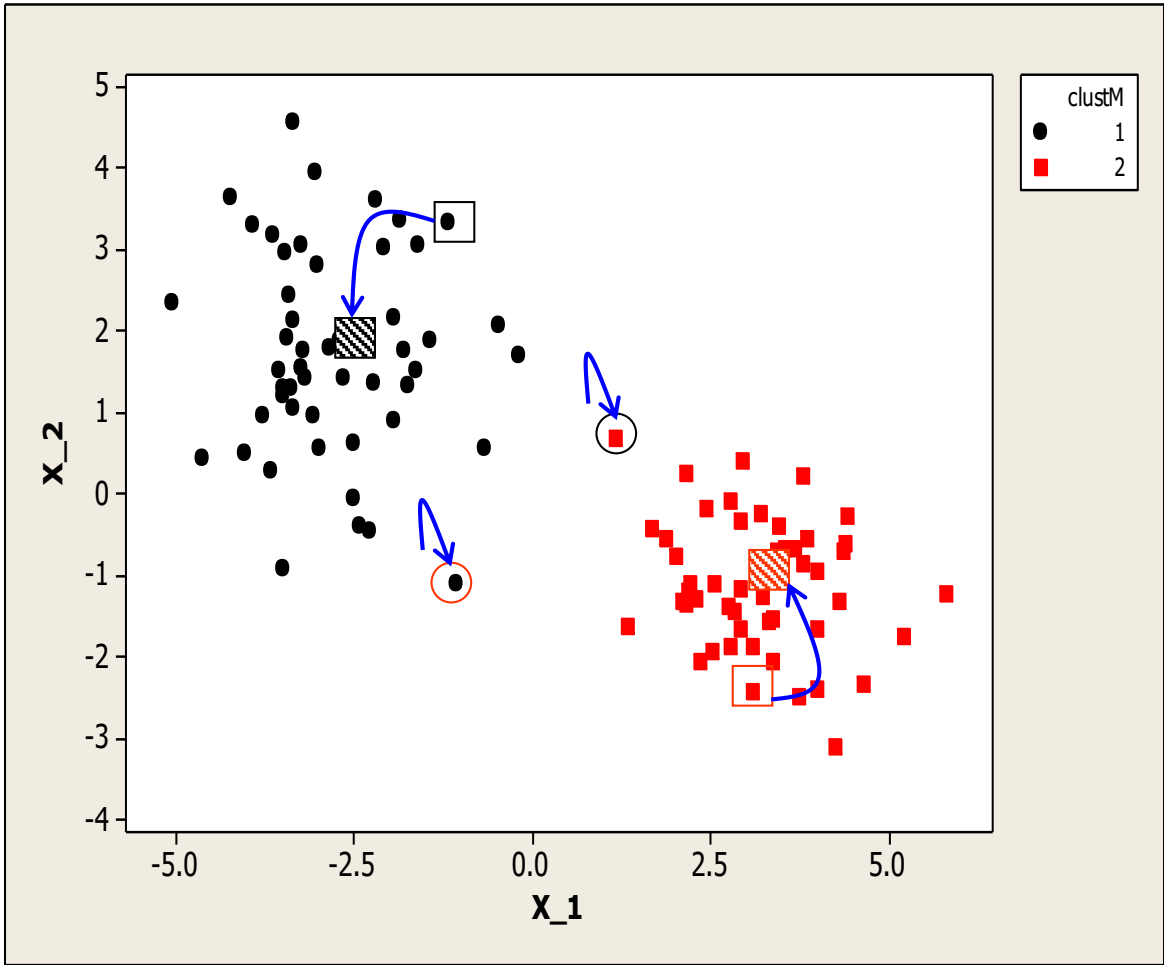
Clusters tend to be ball-shaped with respect to the chosen distance.

Example

Iteration step (first) in updating centroids and memberships.

Rectangles: random centroid initializations, and updated centroids.

Circles: updated memberships.



IMPORTANT: K-means creates only one partition with the specified number of clusters; to get partitions with different K's need to run the algorithm again (and nesting is not guaranteed).

Loosely: read the final sum of squares value associated with various numbers of clusters as a measure of "fit" of the partitioning in K groups (more later).

Variants on partitioning algorithms:

• Partitioning around medioids (PAM): instead of averages, use multivariate medians as centroids (cluster "prototypes"). Dudoit and Freedland (2002).

• Self-organizing maps (SOM): add an underlying "*topology*" (neighboring structure on a lattice) that relates cluster centroids to one another. Kohonen (1997), Tamayo et al. (1999).

• Fuzzy k-means: allow for a "gradation" of points between clusters; *soft partitions*. Gash and Eisen (2002).

• Mixture-based clustering: implemented through an EM (Expectation-Maximization) algorithm. This provides *soft partitioning*, and allows for *modeling* of cluster centroids and shapes. Yeung et al. (2001), McLachlan et al. (2002)

Evaluating a clustering solution:

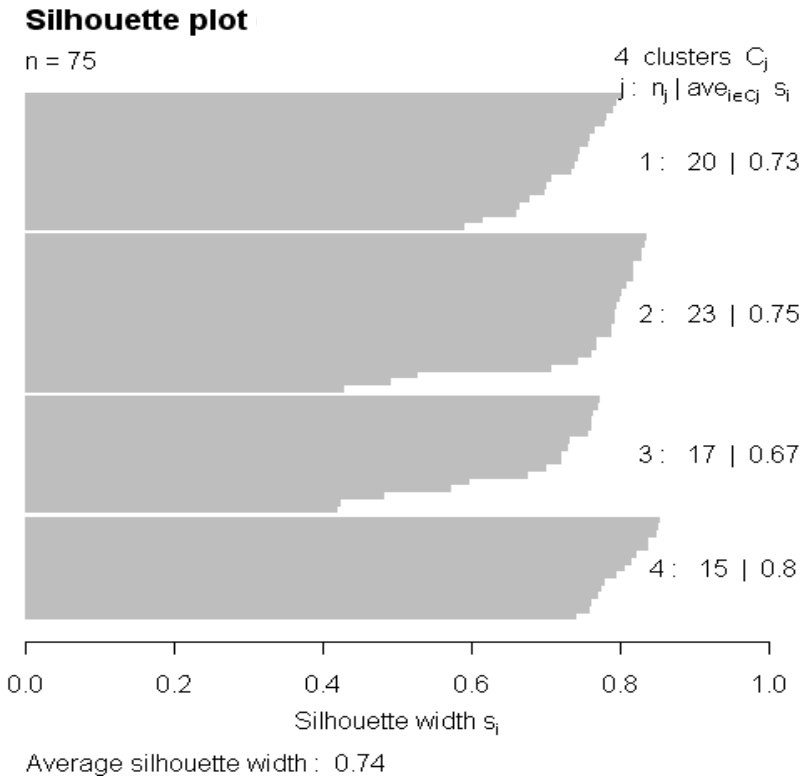
Besides dendrogram cut hight, or final value of the total within cluster sum of squares, a clustering can be evaluated trough **Silhouette widths**. For each point i=1…n define:

$$\delta(i,C) = \frac{1}{\#(C)} \sum_{j \in C} d(x_i, x_j)$$

$$a_i = \delta(i, C(i)) \quad C(i) = C \text{ st } i \in C$$

$$b_i = \min_{C \neq C(i)} \delta(i,C)$$

$$sil_i = \frac{b_i - a_i}{\max\{a_i, b_i\}}$$



Silhouette of 75 units in 4 clusters:
Cluster sizes and average silhouette widths:

20	23	17	15
0.7262347	0.7548344	0.6691154	0.8042285

Individual silhouette widths:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
0.4196	0.7145	0.7642	0.7377	0.7984	0.8549

Observations with large and positive *sil* (~1) are well clustered, those with *sil* around 0 lie between clusters, and those with negative *sil* are placed in the "wrong" cluster.

Summary statistics for sil_i , i=1…N; plots of them (most commonly, by cluster).

Computational assessments:

Because methods are so sensitive to possibly "anomalous" positions of points, and distributional assumptions hard to make, computational assessments (stability, boot-strapping) are also very important:

- Perturb *adding noise* (drawn from what distribution?) to the data.

- Perturb *deleting points* (resample without replacement) from the data.

- Bootstrap (resample with replacement) the data.

Is the clustering solution (dendrogram and chosen partition; partition in K groups and choice of K) stable to perturbations? What is their sampling variability?

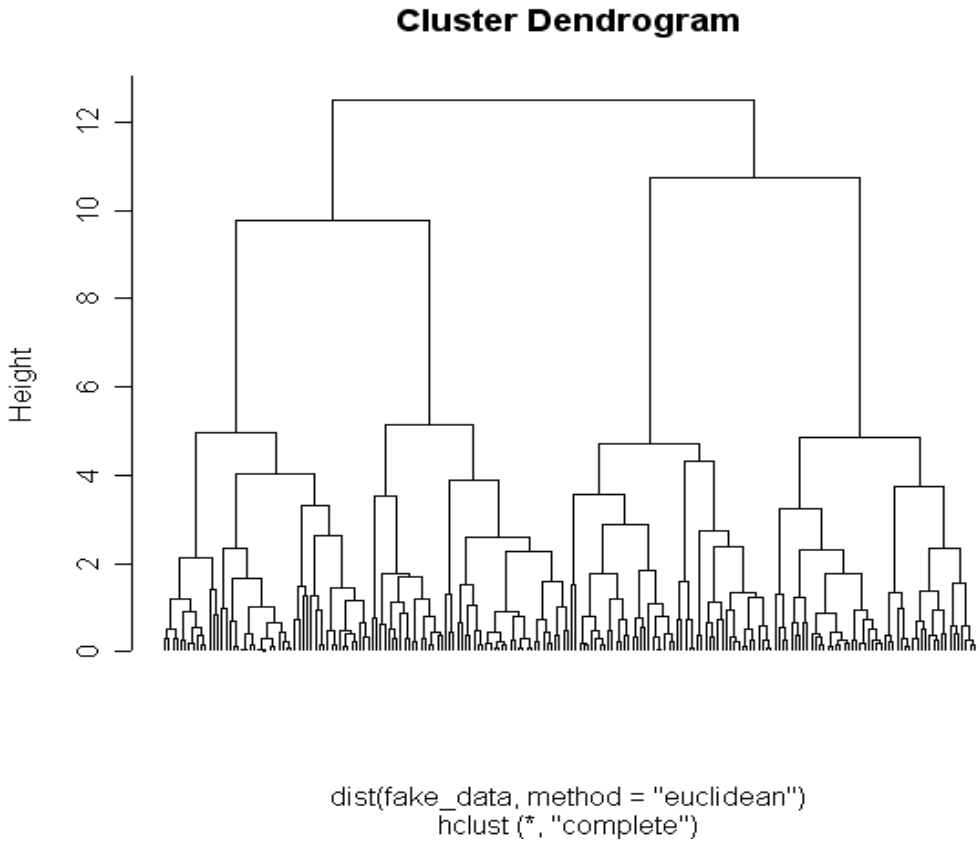
Approaches to determine the number of clusters in a data set

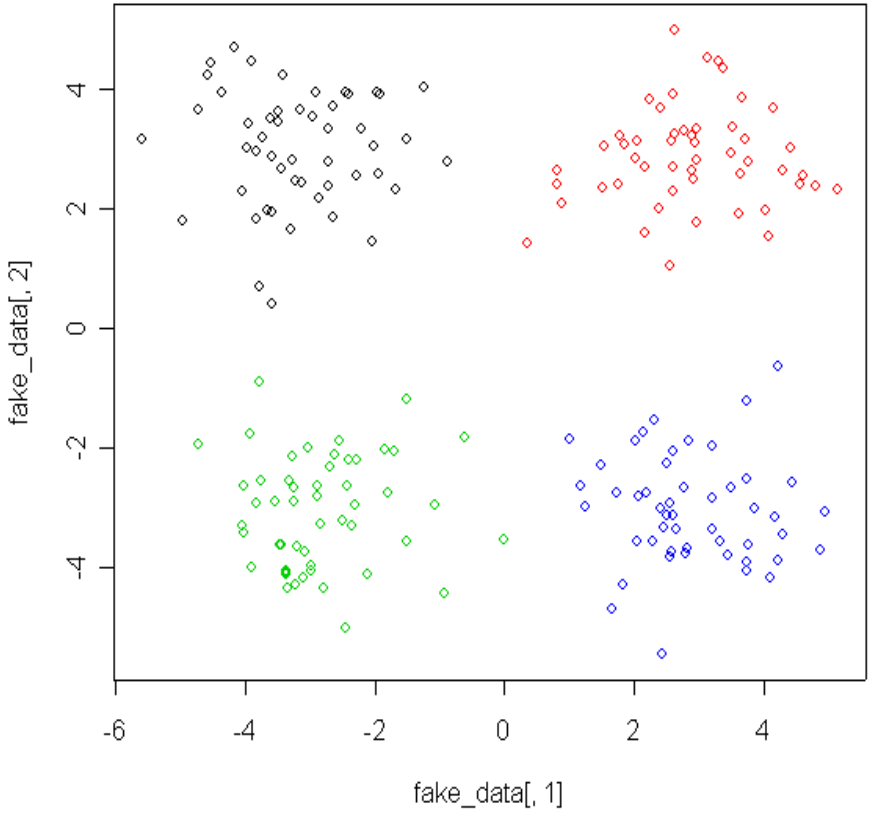
Data set: x_i , i=1…n points in $\boldsymbol{R}^p$ (each coordinate is a feature for the clustering)

Clustering method: e.g. hierarchical with given choices of distance (e.g. Euclidean) and linkage (e.g. complete); K-means with given choice of distance (e.g. Euclidean).

With given method and K (# clusters), we obtain a partition of the points: $P(K) = \{C_1 \ldots C_K\}$

For instance, fake 2D data set (n=200, mixture of four $N(\mu_j, I_2)$)





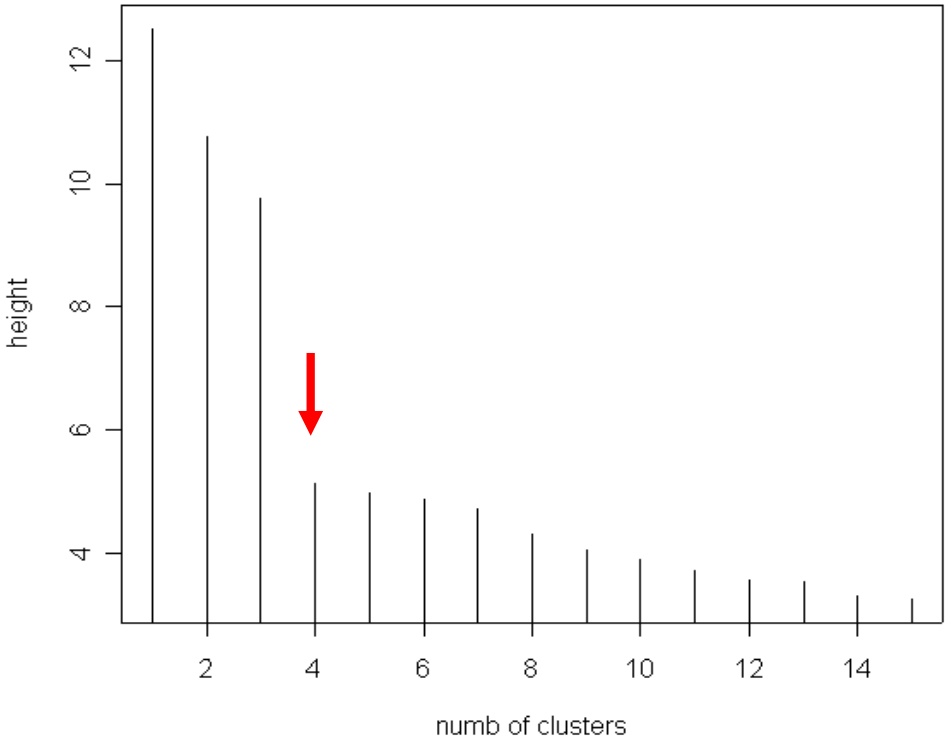
(Hierarchical and K-means give identical results here)

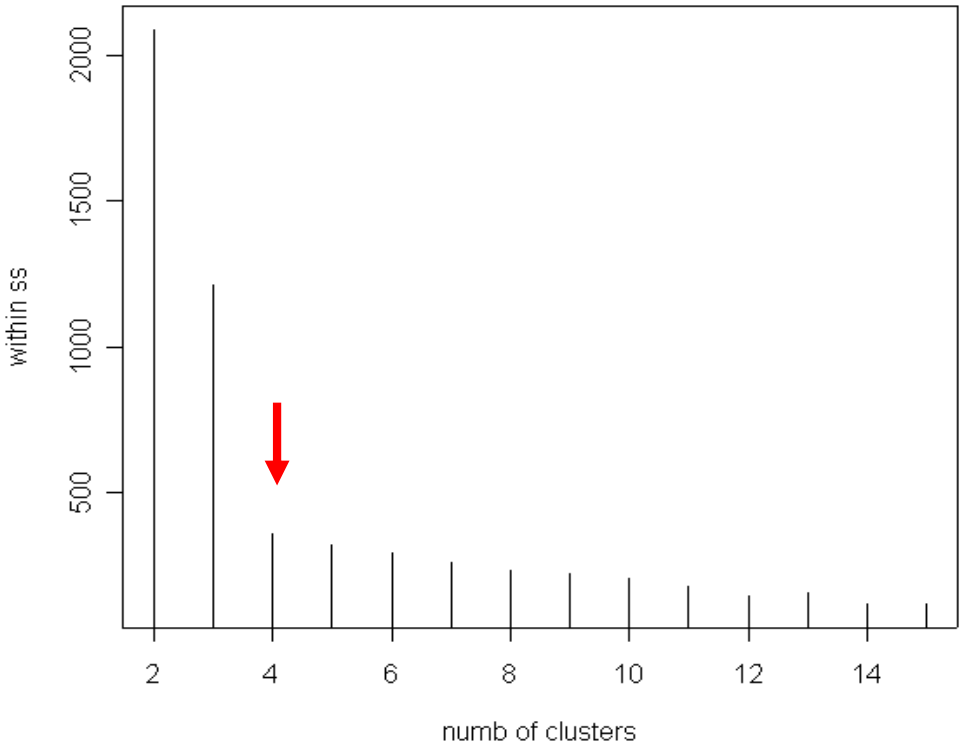
Within cluster dissimilarity/distance

Hierarchical: Dissimilarity levels (heights) at which clusters are formed by cutting.

K-means: Within clusters sum of squares (what the algorithm finds a local minimum for)

$$W(K) = \sum_{j=1\ldots K} \sum_{i \in C_j} d^2(x_i; \bar{x}_j)$$



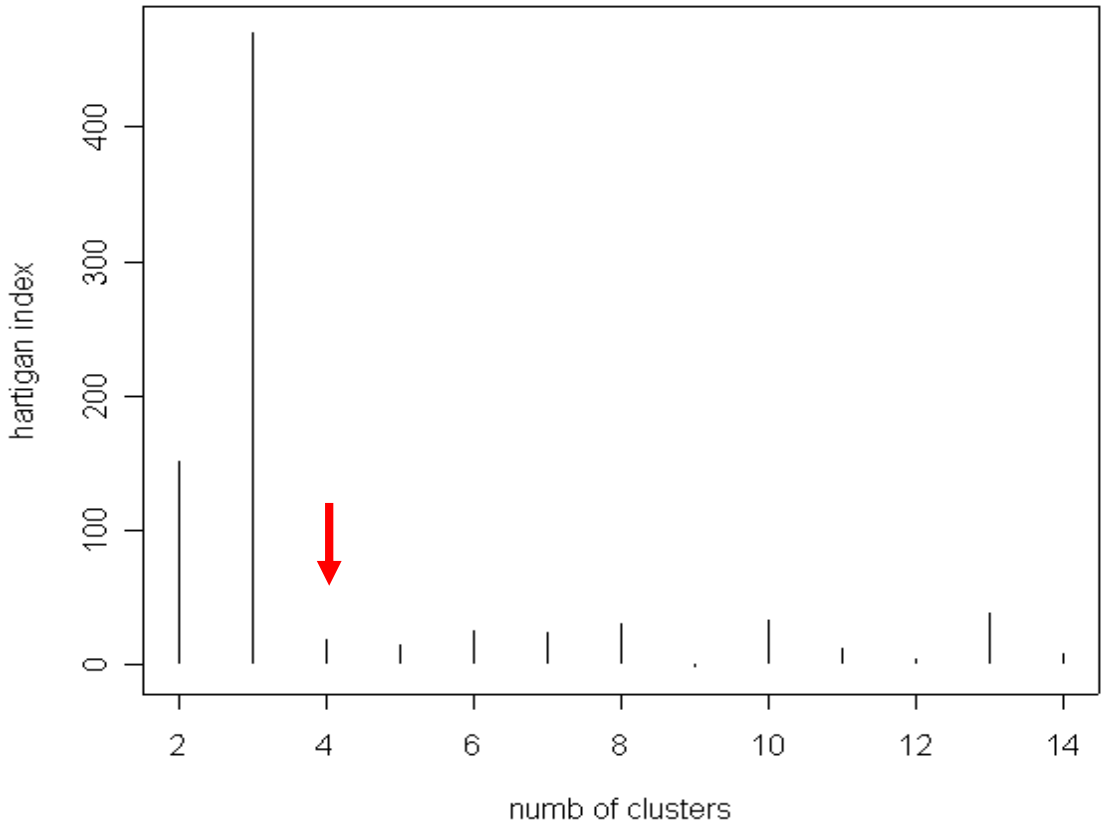


Low values when the partition is good, BUT these are by construction monotone non-increasing (more clusters always makes ≤ within cluster dissimilarity); look for "bends".

Hartigan Index

$$H(K) = \gamma(K)\frac{W(K) - W(K+1)}{W(K+1)}$$

correction $\gamma(K) = n - K - 1$



(corrected) relative improvement when passing from K to K +1. High value (right before) followed by low value when the partition is good. NOT monotone.

Average Silhouette

$$d_{i,C} = \frac{1}{\#(C)} \sum_{l \in C} d(x_i, x_l)$$

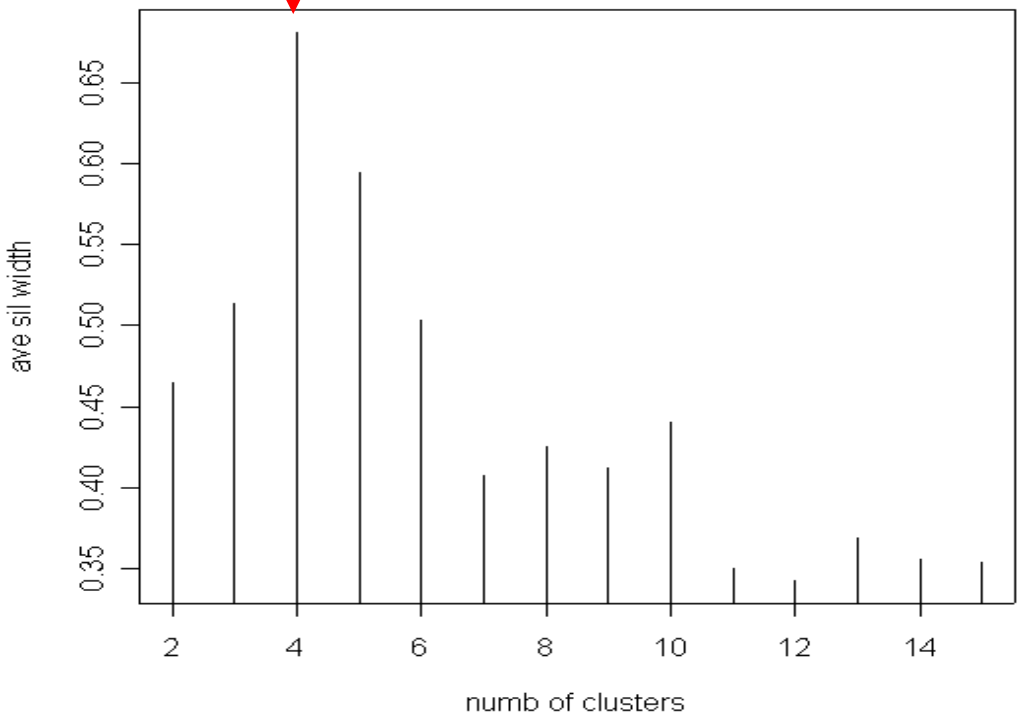
$$a_i = d_{i,C(i)} \qquad b_i = \min_{C \neq C(i)} d_{i,C}$$

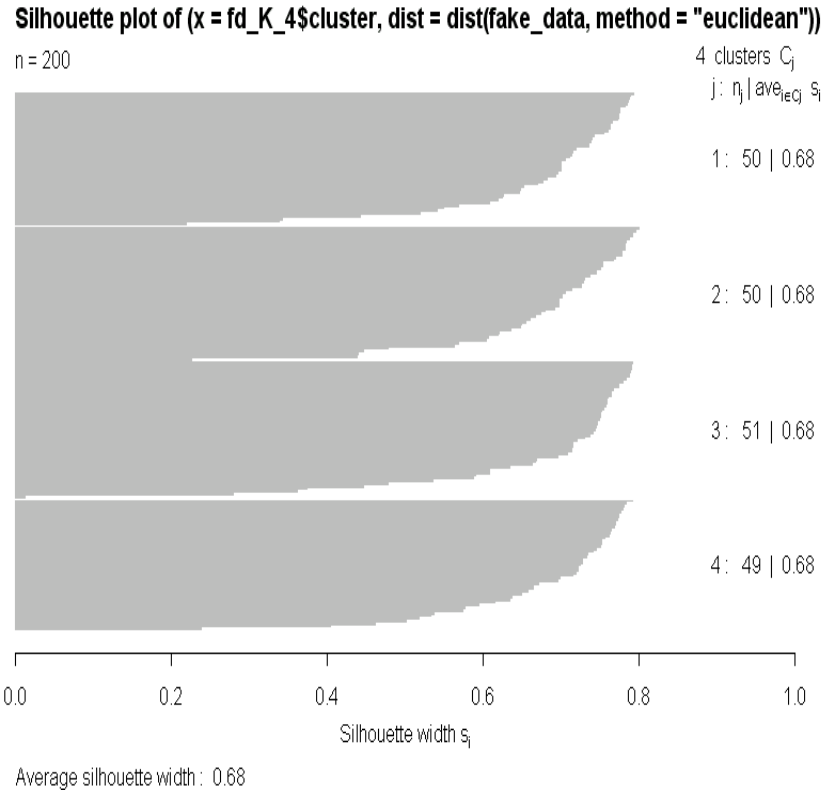
$$Sil_i = \frac{b_i - a_i}{\max\{a_i, b_i\}}$$

How well a point is clustered

$$Sil(K) = \frac{1}{N} \sum_{i=1\ldots N} sil_i$$

Averaging over points, overall partition quality





High value when the partition is good. NOT monotone.

Remarks:

- Many other approaches are possible, e.g., for each *K*, benchmark one of these quality indexes based on stability or reproducibility of the clustering solution.

- Also, for Mixture-based clustering, Bayes Information Criterion (BIC) can be computed for each *K* and used to select a *K*.

- In many real applications the “right” number of clusters is not at all evident; the data points do not show distinct groups – clustering is still useful; think of it more like a way to segment a data cloud in (not well separated) subparts.

Some additions: **Cluster Analysis**

Clustering with Gaussian Mixtures (model-based soft partitioning)

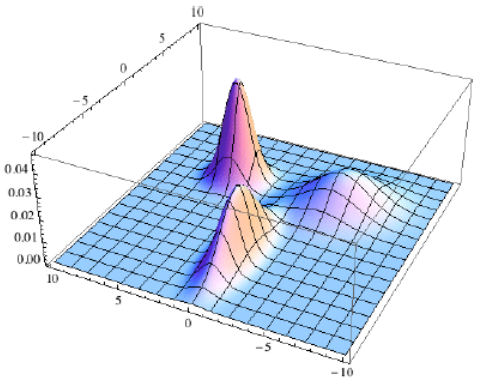
Postulated **stochastic mechanism**:
each data point in $\mathbf{R}^p$ is drawn from

$$f(x) = \sum_{k=1}^{K} \pi_k f(x|z=k) = \sum_{k=1}^{K} \pi_k \varphi(x; \mu_k, \Sigma_k)$$

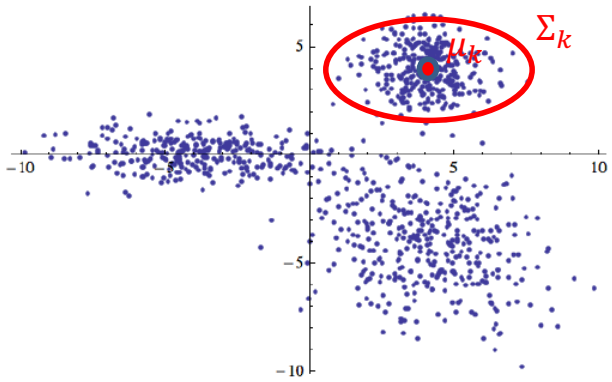
prior probabilities $Pr\{z=k\}$

latent component labels

Gaussian components



(a) A probability distribution on $\mathbb{R}^2$.



(b) Data sampled from this distribution.

$$\theta = \{(\pi_k; \mu_k; \Sigma_k), k = 1 \ldots K\}$$

PARAMETERS
- Prevalence
- Center (location)
- Shape and orientation

of each cluster

$$P = \{p_{i,k}, i = 1 \ldots n, k = 1 \ldots K\}$$

POSTERIOR PROBABILITIES
For each data point and component
$Pr\{z_i=k \mid x_i\}$
non-negative, rows add up to 1

The Expectation-Maximization (EM) algorithm

A very important algorithm to fit models comprising latent (unobservable) variables – here, the component labels.

Produces estimates for

- all parameters – prevalence, location and shape/orientation of the clusters
- and for the posterior probabilities – which express a soft, probabilistic partition of the data points

Similar to the *k-means algorithm*, EM iteratively seeks (an) optimum of an objective function; namely, a maximum for the likelihood on the log scale. After an initialization (of ~~centroids~~ parameters or ~~memberships~~ posterior probabilities), the algorithm iterates between two steps until convergence:

E-step: compute the expectation of the log-likelihood, evaluated given the observable variables and the current parameter estimates

$$h_t(\theta) = E[logL(\theta|\boldsymbol{X}, \boldsymbol{Z})|\boldsymbol{X}, \theta_t]$$

(compute the ~~memberships~~ posterior probabilities, given the current ~~centroids~~ parameters)

M-step: compute parameter estimates maximizing the expected log-likelihood found in the E step.

$$\theta_{t+1} = argmax_\theta \, h_t(\theta)$$

(compute the ~~centroids~~ parameters given the current ~~memberships~~ posterior probabilities)

The Rand Index

Comparing two partitions. Evaluating a clustering solution against a known partition.

- Set of elements {1, 2… n} (the data points)
- A first partition ***A***, e.g., as generated by a clustering algorithm (in r groups)
- A second partition ***B***, e.g., known and used for benchmarking ***A*** (in s groups)

Share of agreement between the two partitions, Rand Index $0 \leq RI \leq 1$

$$RI = \frac{\#\{(i,j) together\ in\ \boldsymbol{A}\ and\ in\ \boldsymbol{B}\} + \#\{(i,j)\ not\ together\ in\ \boldsymbol{A}\ and\ in\ \boldsymbol{B}\}}{\binom{n}{2}}$$

If r and s are different, maximal agreement cannot be 1. Adjusted Rand index

$$ARI = \frac{\sum_{ij}\binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}\right] \Big/ \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \left[\sum_i \binom{a_i}{2}\sum_j \binom{b_j}{2}\right] \Big/ \binom{n}{2}}$$

CONTINGENCY TABLE

n_{11}	n_{12}	$\cdots$	n_{1s}	a_1
n_{21}	n_{22}	$\cdots$	n_{2s}	a_2
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
n_{r1}	n_{r2}	$\cdots$	n_{rs}	a_r
b_1	b_2	$\cdots$	b_s	

Outline: Principal Components Analysis (F. Chiaromonte)

Introduction to Statistical Learning
Chapter 10 sections 1, 2 and 4 Lab 1

Principal Component Analysis (PCA); Some context:

Unsupervised (no response, no labels) dimension reduction technique:

1. Capture the main signals, patterns, structure in a multivariate data set without any prior target or specified predictive aim – Exploratory Data Analysis (EDA).
2. Represent the data in lower dimension, perhaps to visualized them.
3. De-noise the data by removing negligible variation.
4. Use low-dimensional/de-noised data in input for supervised analyses.

Note: if the purpose is to create low-dimensional representations relevant for other analyses (4), other approaches exist:

- Linear Discriminant Analysis (LDA) prior to classification
- Sufficient Dimension Reduction (SDR) prior to regression
- Multi-Dimensional Scaling (MDS) prior to clustering

… the strength of PCA is its "general purpose" nature.

Also connections to:

- Independent Components Analysis (extract independent, as opposed to uncorrelated, components)
- Factor Analysis (extract latent components in the framework of a probabilistic model for the mechanism generating the data.)

Units\Features	X_1	X_2	…	X_p
Unit 1	x_{11}	x_{12}		x_{1p}
Unit 2	x_{21}	x_{22}		x_{2p}
. . .				
Unit n	x_{n1}	x_{n2}		x_{np}

p features measured on n units:

- An n x p data matrix X
- A data cloud of n points in R^p .

Location is irrelevant to PCA; assume each feature is centered, mean 0.

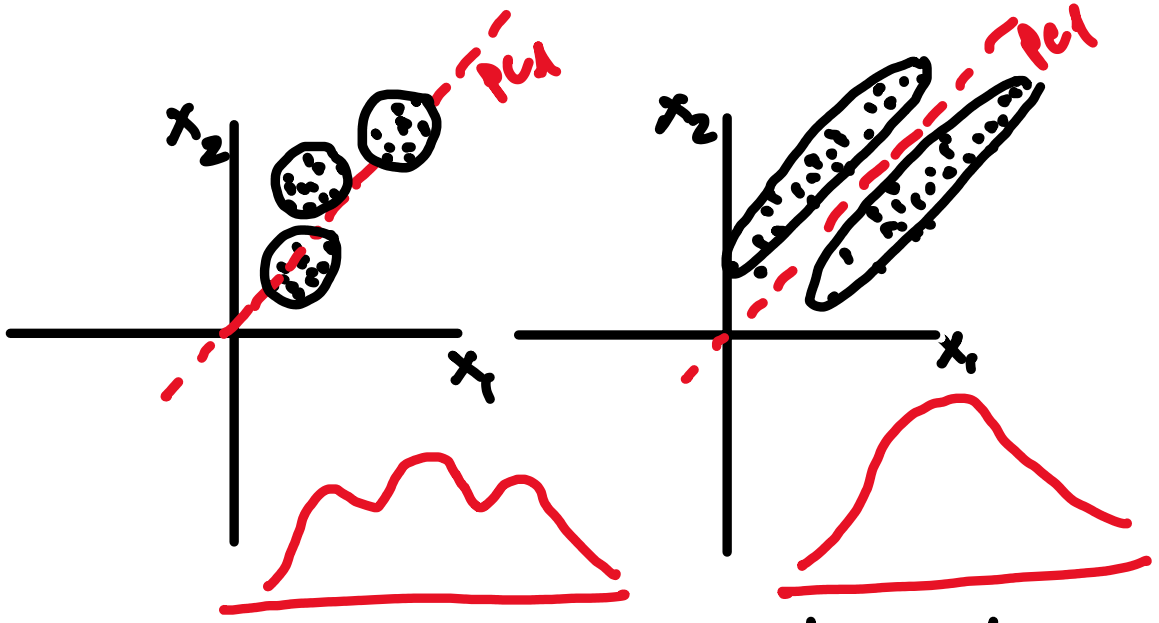
$\bar{X} = 0_p$ mean vector in R^p; cloud is centered/located at origin

$S = X'X$ (sample) p x p variance/covariance matrix

proportional to; divide by n or (n-1)

Find directions (and corresponding linear combinations) in R^p that are most interesting, in the sense of capturing variability of the data cloud.

General purpose, in many applications (but not all) these are the most informative, capturing structure in the data.



Projecting on PC1 preserves the cluster structure

does not

Sequentially, find orthogonal directions (orthonormal vectors) maximizing the variance of the corresponding projections:

$$\text{for } m = 1, 2 \ldots p:$$

$$\max_{\phi_m} \text{var}(\phi_m^T X)$$

$$\text{subject to } \| \phi_m \| = \phi_m^T \phi_m = 1, \phi_m^T \phi_k = 0 \ (k = 1, 2 \ldots m-1)$$

Equivalent to taking the Eigen decomposition of S:

$$S = \sum_{m=1}^{p} \lambda_m \phi_m \phi_m^T$$

$$\lambda_1 \geq \ldots \geq \lambda_p \geq 0 \quad \text{(eigenvalues)}$$

eigenvalues are variances along the directions identified by eigenvectors; in non-increasing order.

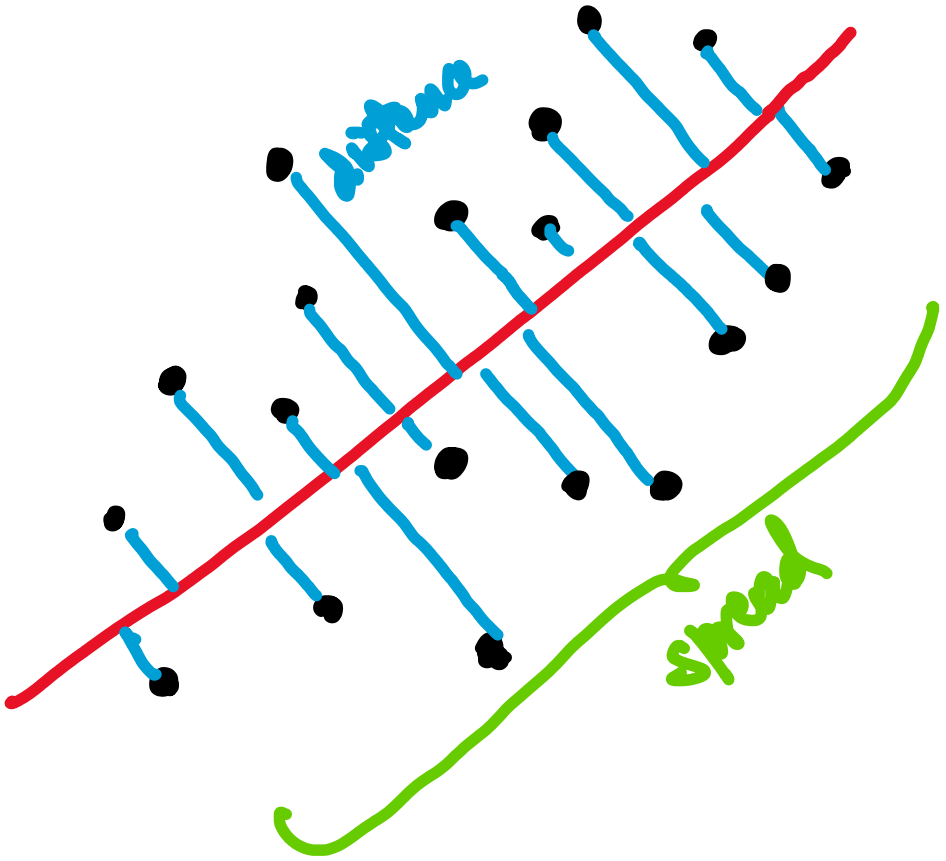
$$\| \phi_m \| = \phi_m^T \phi_m = 1, \phi_m^T \phi_k = 0 \ m, k = 1, 2 \ldots p \quad \text{(eigenvectors)}$$

For any given dimension *m*, identify the linear subspaces closest to the data:

$$\| P_{Span(\Phi_1 \ldots \Phi_m)} X \|^2 = \max$$

m-dimensional representation most likely to be useful in capturing structure, most informative (not always though!)

Note: here proper distance, least squares in regression uses "vertical" distance because one has a response to predict.



PC 1

minimizes the sum of squared distances

⇕ (equiv.)

maximizes the variance of the projected data

LOADINGS: coefficients expressing the m-th component in terms of the original features

$$\phi_{m1}, \quad \phi_{m2} \dots \phi_{mp}$$

(coordinates of each element of the new basis in terms of the original basis)

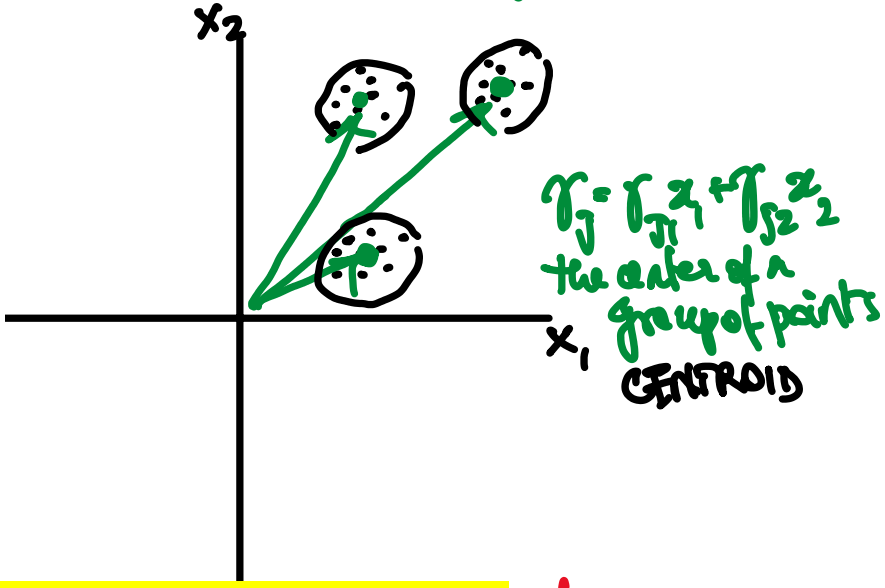
SCORES: values of the m-th component on the n units

$$z_{im} = \phi_{m1} x_{i1} + \phi_{m2} x_{i2} \dots + \phi_{mp} x_{ip} \quad , \quad i = 1, 2 \dots n$$

(coordinates of the n data points in terms of each element of the new basis)

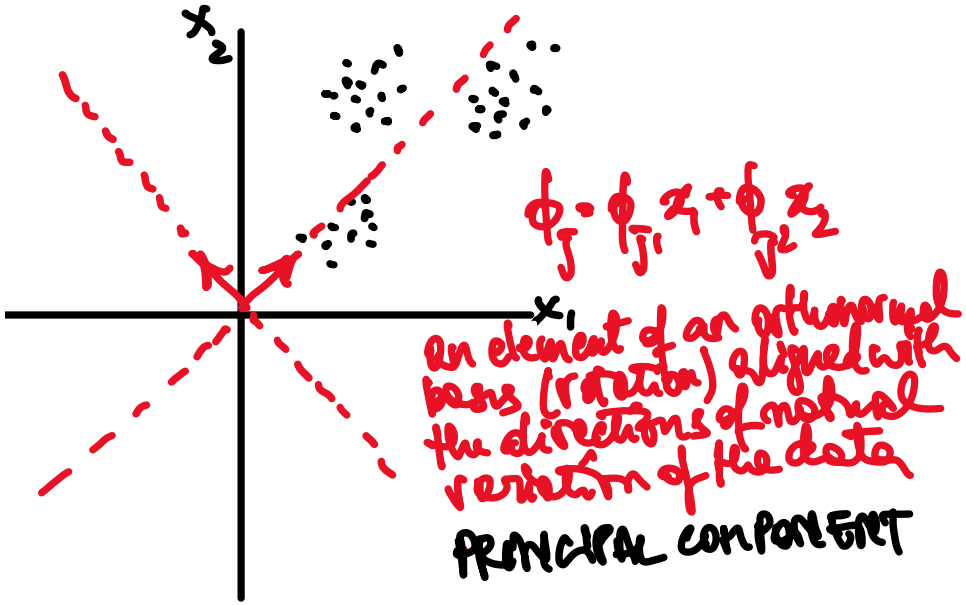
"PROTOTYPICAL" PATTERNS

$r_J, J=1,2,3$

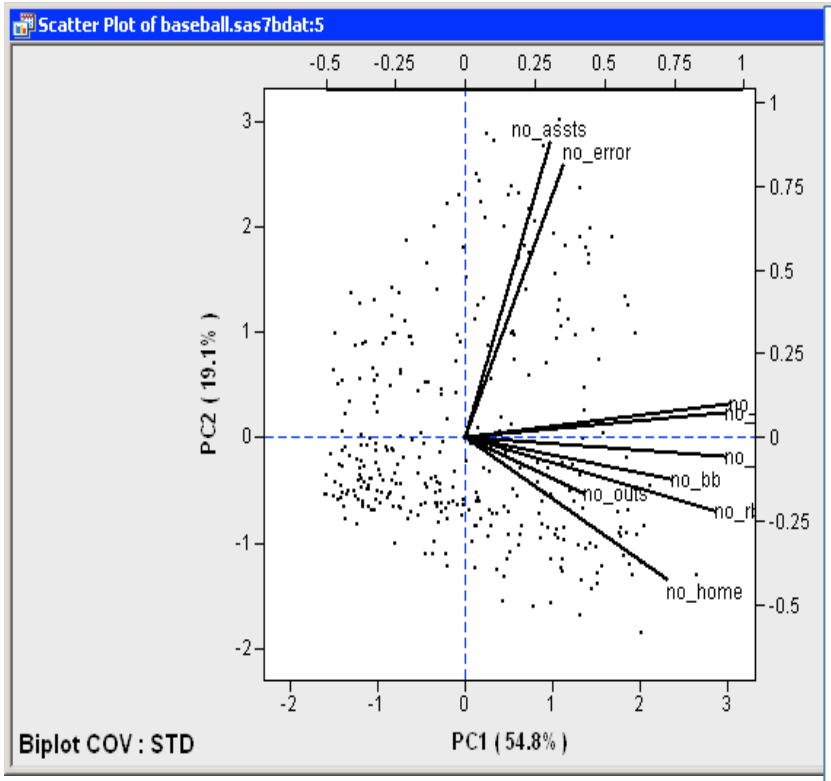


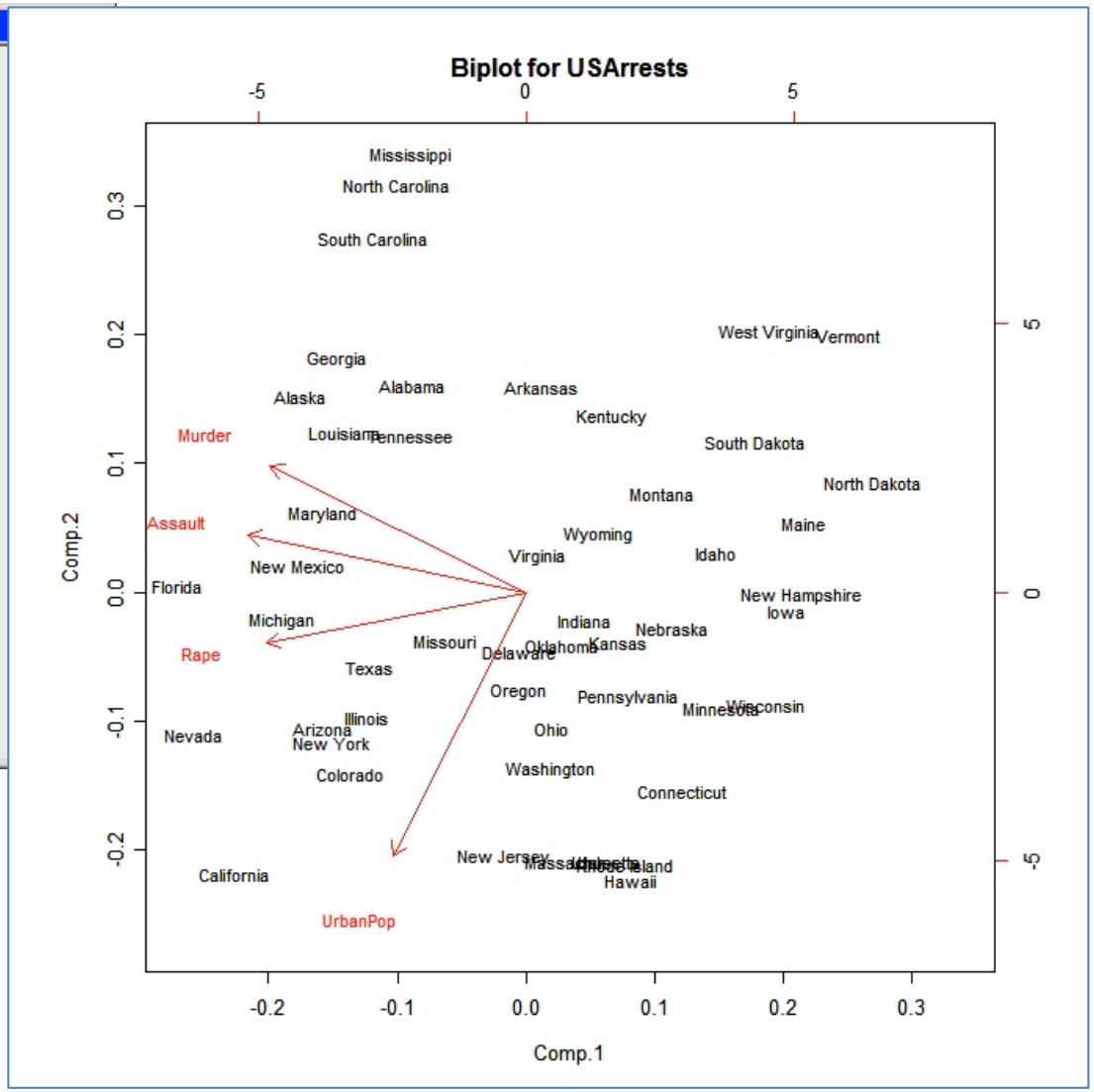
"COMPONENT" PATTERNS

$\phi_J, J=1,2$



Biplot: shows the scores for two specified components (e.g. 1st and 2nd; projection of the points on the first PCA plane) along with the loadings represented by arrows.





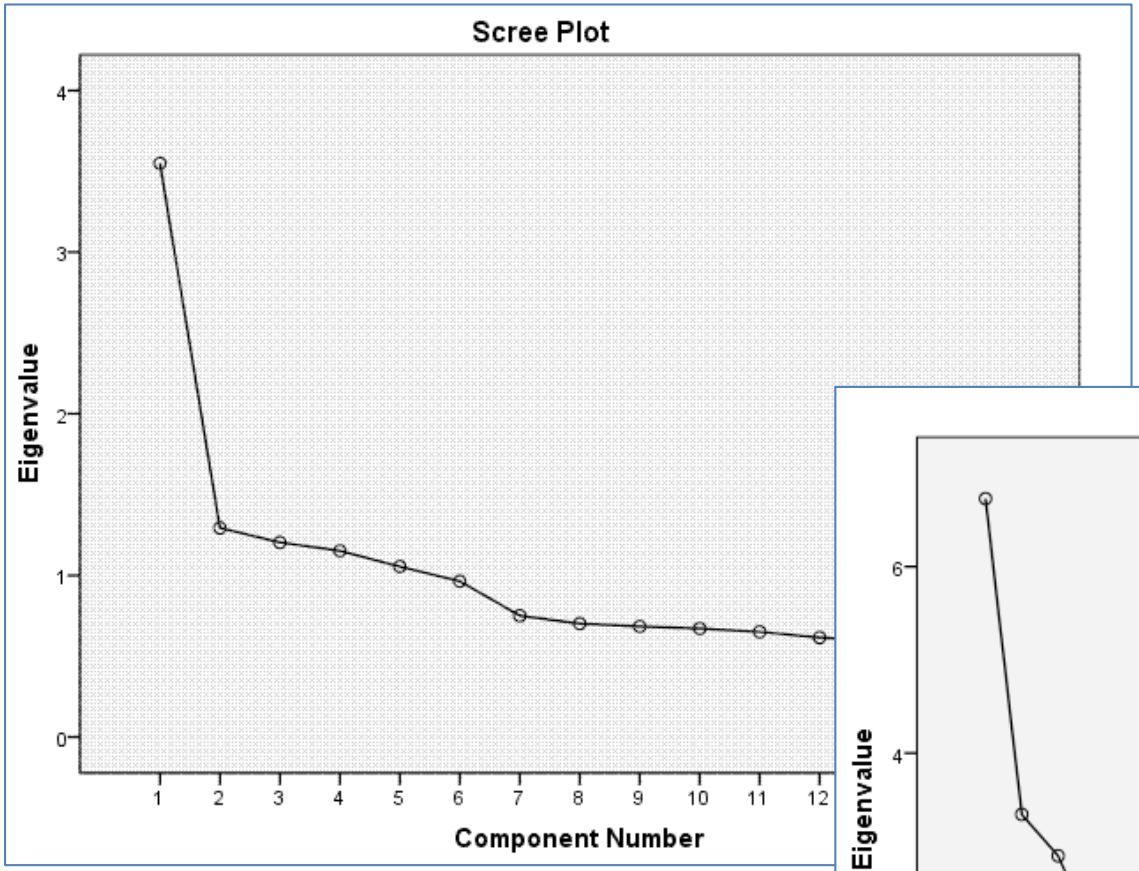
PERCENTAGE OF VARIANCE EXPLAINED (PVE) by the m-th component

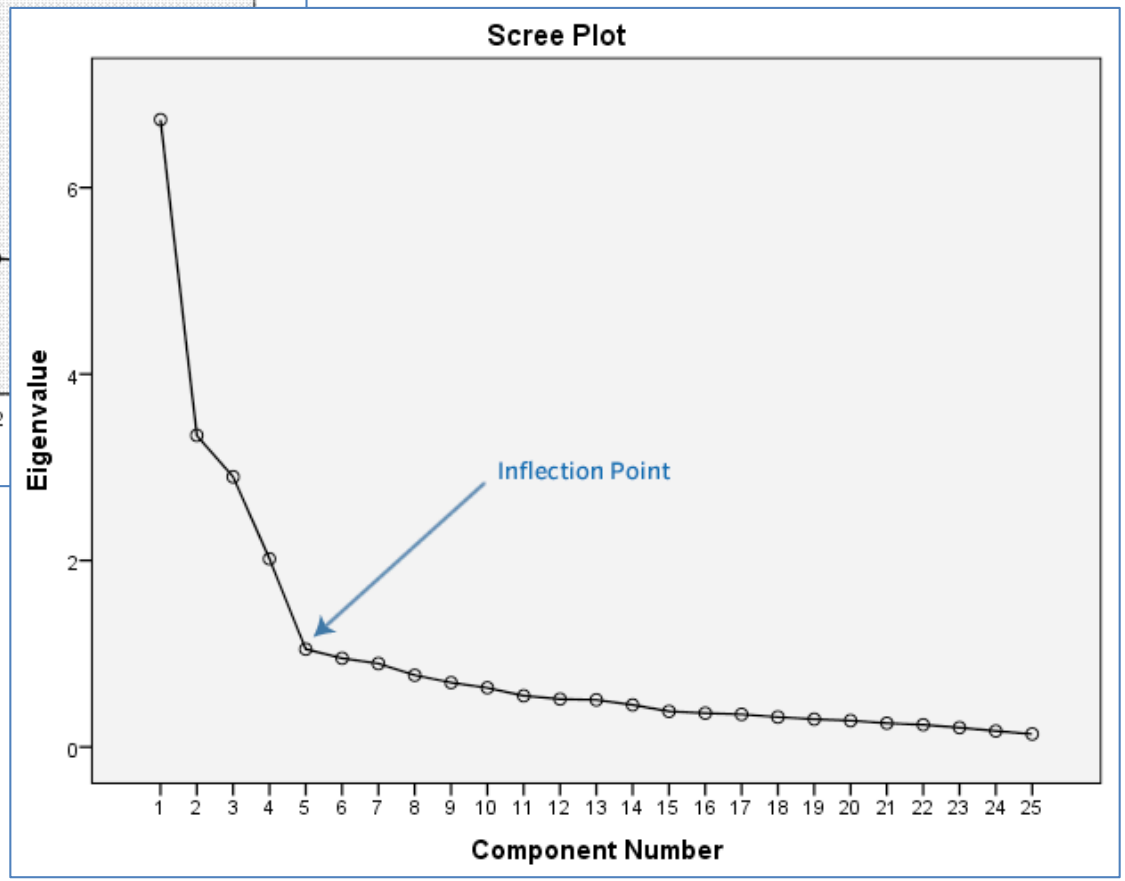
$$PVM_m = \frac{\lambda_m}{\sum_{k=1}^{p} \lambda_k} = \frac{\mathrm{var}(\phi_m^T X)}{\sum_{k=1}^{p} \mathrm{var}(\phi_k^T X)}$$

Also, **CUMULATIVE PVE** up to the m-th component

$$CPVM_m = \sum_{j=1}^{m} PVM_j = \frac{\sum_{j=1}^{m} \lambda_j}{\sum_{k=1}^{p} \lambda_k}$$

Scree plot: Show the variances (eigenvalues) or PVEs in non-increasing order.





How many components to consider?

$M \leq \min\{ p, n-1 \}$

Rule of thumb: look where the PVEs become very small and/or level off (stop decreasing substantially).

The book says this is necessarily ad-hoc because an unsupervised analysis does not have a prediction outcome that allows to select "tuning" parameters of the procedure through cross-validation.

But some less subjective approaches are available! Technically, assessing how many (tail) eigenvalues are not significantly > 0:

- Rigorous tests exist if the features are multivariate Gaussian
- Sampling variability of the eigenvalues can be gaged by bootstrapping (put bootstrap "bands" around the scree plot)
- Empirical null distribution of the eigenvalues can be simulated under appropriate assumptions
- Information-type criteria can be formed under appropriate assumptions, comprising a penalty for complexity.

Scaling: In some (but not all) circumstances, it is good to eliminate from consideration not just location but also units of measurement and/or magnitude of variation of the original features:

- Normalize each dividing by its standard deviation; $X \leftarrow \text{Diag}(s_j)^{-1/2} X$
- S becomes a correlation matrix (diagonal entries = 1; off diagonal entries in (-1,1)).

Note: unlike other analyses (e.g. regression) PCA is affected in non-trivial ways by scaling; the Eigen decompositions of $X'X$ and $X'DX$ (where D is diagonal) can be very different. Unless D is a multiple of the identity:

- Eigenvectors can change
- Eigenvalues can differ by more than just a scaling factor.

De-noised representation of the data:

based on a reconstruction in M components, take

$$x_{ij} \approx \sum_{m=1}^{M} z_{im}\phi_{jm1} \quad , \quad i = 1,2 \ldots n \, , \, j = 1,2 \ldots p$$

eliminates the “detail” (noise) in the directions that have been discarded; de-noise.

Some additions: **Multidimensional Scaling**

Given a matrix $D = \{d_{ij}\}$ containing the distances (dissimilarities) between each pair of n objects in a set, and a chosen number of dimensions, k, MDS places each object into a k-dimensional Euclidian space as to retain as much as possible the distances between objects. If k = *1, 2* or *3*, the resulting data cloud can be visualized.

Stress function

$$S_k(\boldsymbol{X}) = \sqrt{\frac{\sum_{i<j}(d(x_i; xj) - dij)^2}{\sum_{i<j} d(x_i; xj)^2}} \longrightarrow \boldsymbol{X}_k{}^* = argmin\ S_k(\boldsymbol{X})$$

The minimization can be performed numerically under a variety of specifications of the problem.

In the simplest, where D is a true Euclidian distance matrix, this reduces to an Eigen decomposition problem like in PCA. Form the scores in the Eigen-space spanned by the eigenvectors of the k largest eigenvalues of

$$A = \left\{-\frac{1}{2}d^2{}_{ij}\right\}$$

Centered Matrix

$$B = \{a_{ij} - ai_{\cdot}\_a_{\cdot j} + a_{\cdot\cdot}\}$$

Can pick a reasonable k using the eigenvalues.

Notes:

- The stress can be brought to 0 for k $\geq$ n-1.
- Non-uniqueness, e.g., any translation or rotation of the solution produces an equally good solution.

Outline: **Supervised Classification**
(F. Chiaromonte)

Introduction to Statistical Learning
Chapter 4

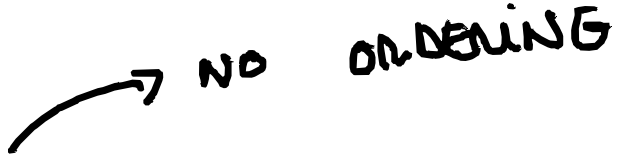
Units\Features	X_1	X_2	…	X_p	Y
Unit 1	x_{11}	x_{12}		x_{1p}	y_1
Unit 2	x_{21}	x_{22}		x_{2p}	y_2
.					
.					
.					
Unit n	x_{n1}	x_{n2}		x_{np}	y_n

p features and one **categorical response** measured on n units:

- An n x p data matrix X, plus a columns of labels
- A data cloud of n labeled points in R^p .

General idea: using the data available, train (estimate the parameter of) an algorithm (model) that will allow one to predict the label of a new unit based on the values in its feature vector $x_{(new)} = (x_{1(new)} \dots x_{p(new)})^T$

Simple methods (Chapter 4):



- **Logistic Regression**
- **Linear and Quadratic Discriminant Analysis** (LDA, QDA)
- **K-Nearest Neighbors** (see also Chapter 2)

Many more sophisticated algorithms exist, e.g., Tree-Based methods (Chapter 8), Support Vector Machines (SVM, Chapter 9)

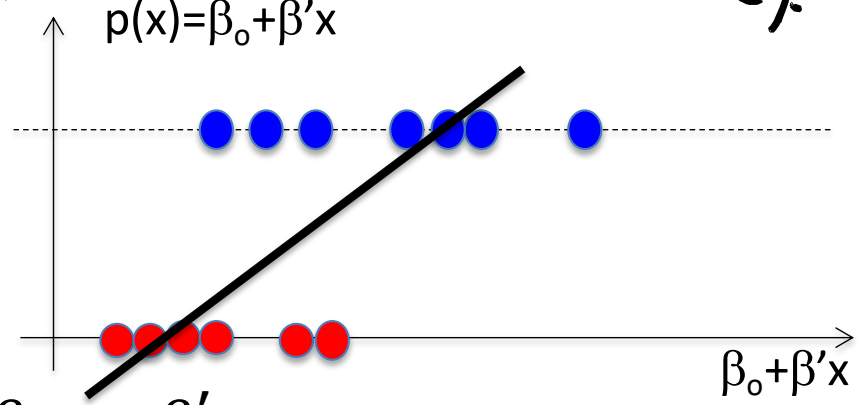
$$\beta_0 + \beta' x = \log\left(\frac{p(x)}{1-p(x)}\right)$$

$$y = \beta_0 + \beta' x + \varepsilon \quad \text{error} \quad \varepsilon \sim N(0, \sigma^2)$$

LOGISTIC REGRESSION

Binary Y, encoded (arbitrarily) as {0,1}

The standard Linear Model would be:



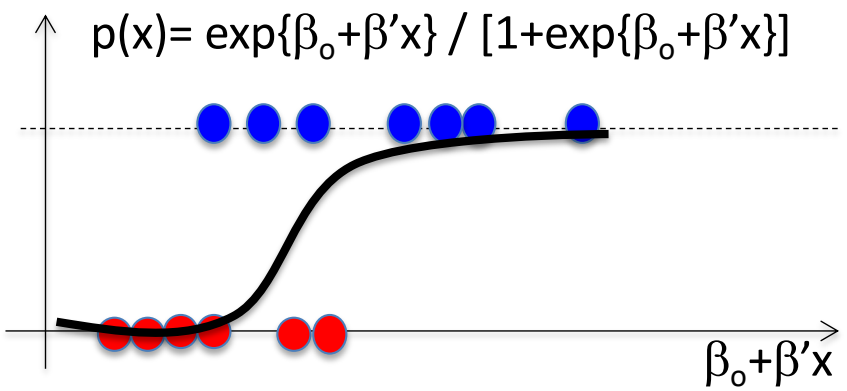
$$E(Y|x) = \Pr(Y = 1|x) = p(x) = \beta_o + \beta' x$$

$$Y|x \sim p(x) + \varepsilon, \varepsilon \sim N(0, \beta\sigma^2)$$

… can produce p(x) estimates smaller than 0 or larger than 1, and adding Gaussian error makes no sense. Instead, use a *link function* (Generalized Linear Model); the LOGIT:

$$\log\left(\frac{p(x)}{1-p(x)}\right) = \beta_o + \beta' x$$

ODD RATIO

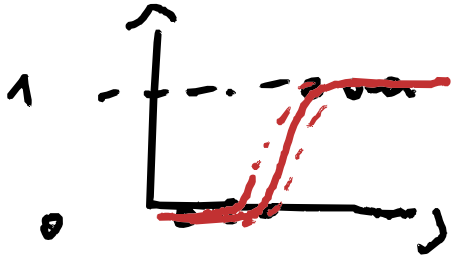


Log of the *odds ratio* of a "1" modeled as a linear function of x.

Stochastic component in the mechanism producing the data is represented through a Bernoulli scheme, not through a Gaussian error about the expected value.

Parameters are estimated by *maximum likelihood*; p(x) can be predicted for any level of x:

$$\hat{p}(x) = \frac{\exp\{\hat{\beta}_o + \hat{\beta}' x\}}{1 + \exp\{\hat{\beta}_o + \hat{\beta}' x\}}$$



and the label can be predicted (classification) based on whether this is > 0.5 (a threshold).

Remark: likewise a standard linear model, some of the features included as predictors could be *binary or categorical* (through dummy variables).

Remark: In the credit default example used in the book, student status (=0,1) has a positive effect on the log-odds ratio of default when considered by itself (being a student, marginally, increases the odds of defaulting) but a negative value when considered together with credit card balance (given any given level of outstanding balance, being a student decreases the odds of defaulting). An example of *confounding*. Due to associations among predictors; the effects of a feature considered by itself or in the context of other features are different!

Remark: MLEs in Logistic regression are highly *unstable* when the two classes are well separated.

LINEAR DISCRIMINANT ANALYSIS

Very close relative of Logistic regression that:

- Works also when Y has K > 2 than two classes
- Overcomes instability presented by the MLEs in Logistic regression when the two classes are well separated.

For k=1,2… K, instead of modeling $p_k(x) = Pr\{Y=k|x\}$, model $f_k(x)$ = density of X given Y = k .

Let π_k = *prior* Pr{Y=k}, by Bayes Theorem we have:

$$p_k(x) = \frac{\pi_k f_k(x)}{\sum_{j=1}^{K} \pi_j f_j(x)}$$

The *Bayes classifier rule* (optimal performance under certain assumptions) will attribute a new unit to the class $k^*(x_{(new)})$ for which $p_k(x_{(new)})$ is highest.

LDA assumes a model for $f_k(x)$ and implements estimation of $p_k(x)$ accordingly.

Assume $X \mid Y = k \sim N_p(\mu_k, \Sigma)$ p-variate Gaussians with different mean vectors but the same variance-covariance matrix

$$f_k(x) = \frac{1}{(2\pi)^{p/2}\det(\Sigma)^{1/2}} \exp\left\{-\frac{1}{2}(x-\mu_k)'\Sigma^{-1}(x-\mu_k)\right\}$$

Bayes classifier *discriminant function* (classify to highest)

$$\delta_k(x) = x'\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k'\Sigma^{-1}\mu_k + \log \pi_k$$

Note: linear function of x

Estimate $\hat{\pi}_k = \frac{n_k}{n}$, $k = 1, 2 \ldots K$

$$\hat{\mu}_k = \frac{1}{n_k}\sum_{i:y_i=k} x_i \quad , \quad k = 1, 2 \ldots K$$

$$\hat{\Sigma} = \frac{1}{n-K}\sum_{k=1}^{K}\sum_{i:y_i=k}(x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)'$$

pooled estimate of the (common) "within class" Σ

LDA discriminant function (classify to highest) approximates Bayes classifier

$$\hat{\delta}_k(x) = x'\hat{\Sigma}^{-1}\hat{\mu}_k - \frac{1}{2}\hat{\mu}_k'\hat{\Sigma}^{-1}\hat{\mu}_k + \log \hat{\pi}_k$$

Example p=1, K=2. *Boundary*: x s.t. $\delta_1(x) = \delta_2(x)$ (here $x = (\mu_1+\mu_2)/2$ since $\pi_1=\pi_2=1/2$)

↳ Feature

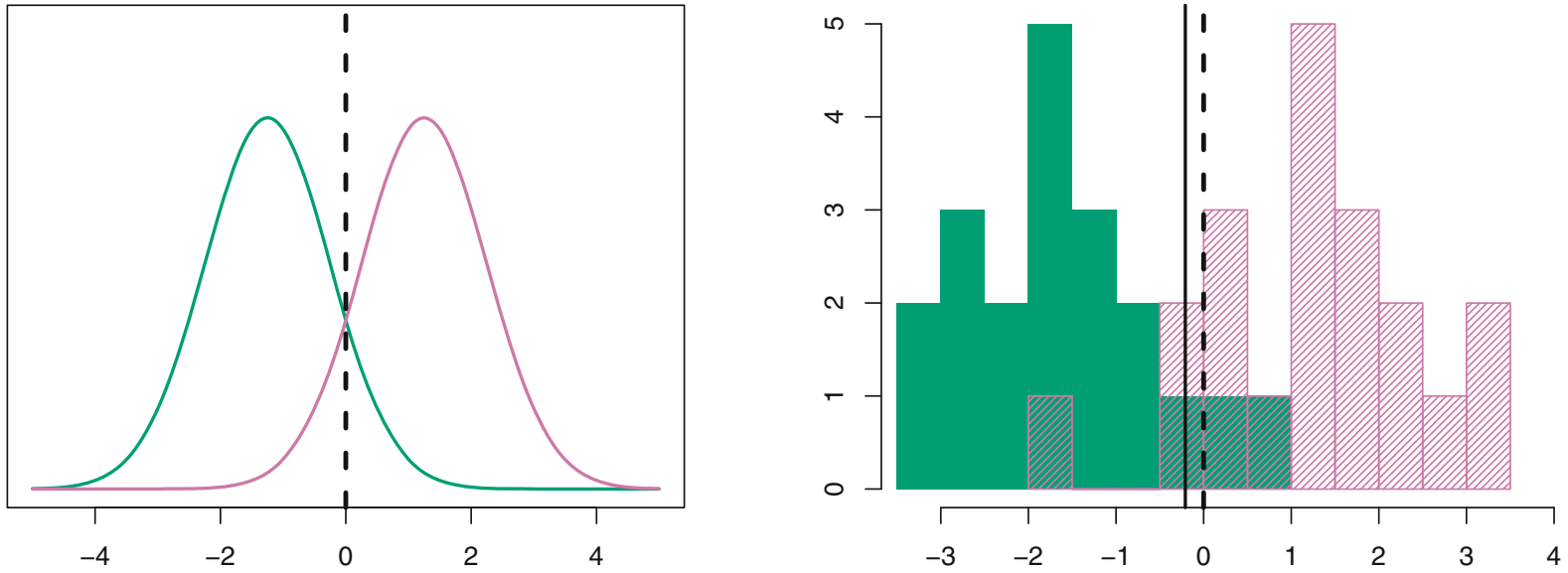


FIGURE 4.4. Left: *Two one-dimensional normal density functions are shown. The dashed vertical line represents the Bayes decision boundary.* Right: *20 observations were drawn from each of the two classes, and are shown as histograms. The Bayes decision boundary is again shown as a dashed vertical line. The solid vertical line represents the LDA decision boundary estimated from the training data.*

Example p=2, K=3. *Boundaries*: x s.t. $\delta_k(x) = \delta_j(x)$, for each k, j pair

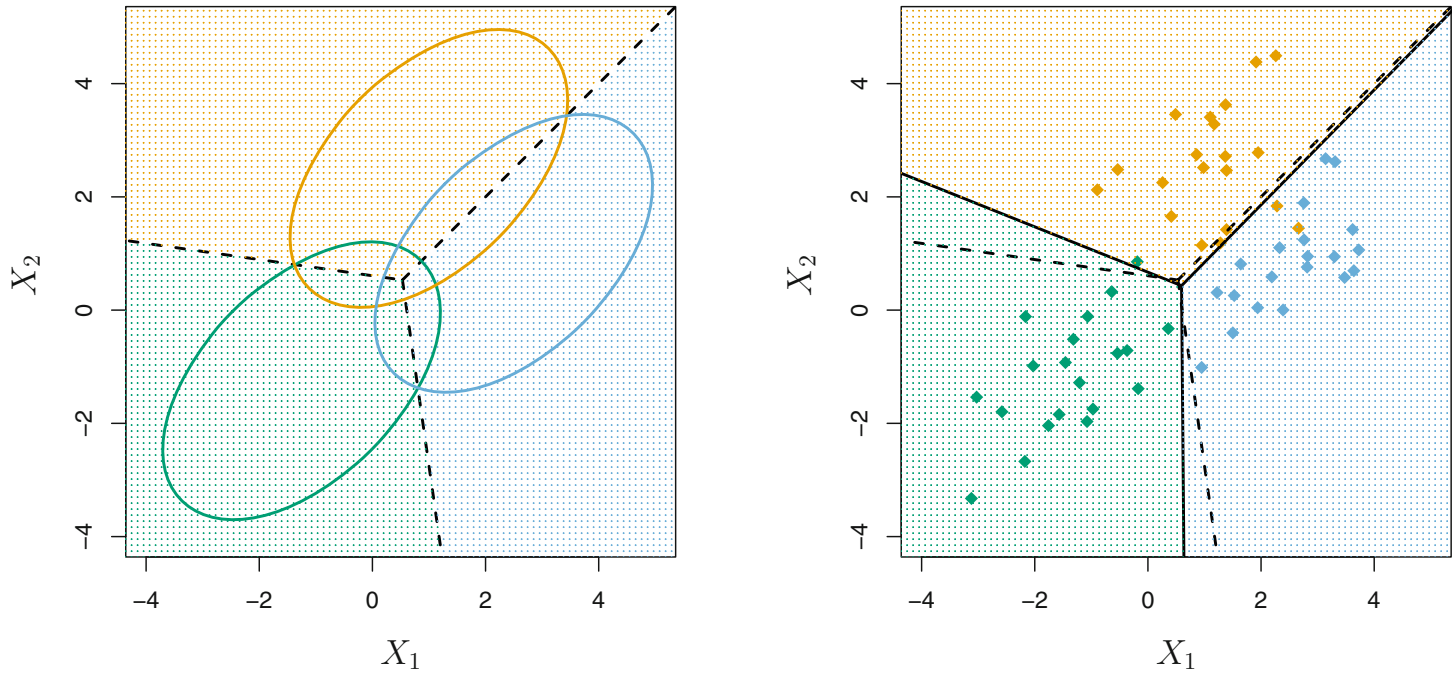


FIGURE 4.6. *An example with three classes. The observations from each class are drawn from a multivariate Gaussian distribution with* $p = 2$*, with a class-specific mean vector and a common covariance matrix.* Left: *Ellipses that contain* 95 % *of the probability for each of the three classes are shown. The dashed lines are the Bayes decision boundaries.* Right: 20 *observations were generated from each class, and the corresponding LDA decision boundaries are indicated using solid black lines. The Bayes decision boundaries are once again shown as dashed lines.*

QUADRATIC DISCRIMINANT ANALYSIS

Works exactly the same way, but variance-covariance matrices are allowed to differ among the classes

- We need to estimate a much larger number of parameters!
- The boundaries are allowed to curve; more flexible classifier
- Less *bias* but more *variance*
- More prone to overfitting

In a given application, do you have a large enough sample size to use QDA?

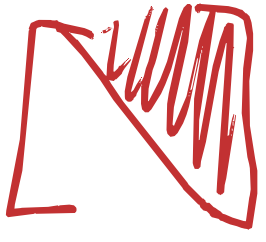
Is it really a stretch to assume common variance-covariance across classes and use LDA?

LDA classifier discriminant function (classify to highest) approximates Bayes classifier

$$\hat{\delta}_k(x) = x'\hat{\Sigma}_k^{-1}\hat{\mu}_k - \frac{1}{2}x_k'\hat{\Sigma}_k^{-1}x_k - \frac{1}{2}\hat{\mu}_k'\hat{\Sigma}_k^{-1}\hat{\mu}_k - \frac{1}{2}\log\det(\hat{\Sigma}_k) + \log\hat{\pi}_k$$

Note: quadratic term in x

PARAMS IN $\frac{p(p+1)}{2}$



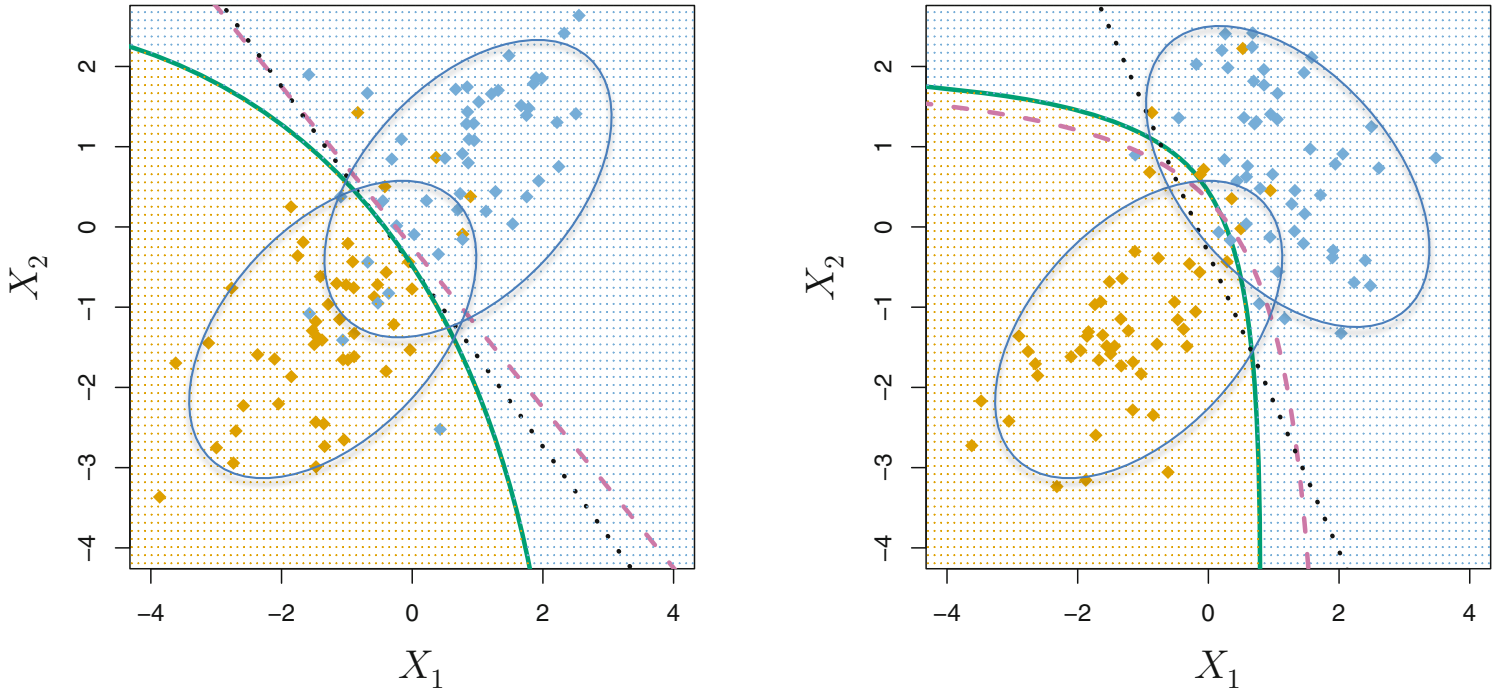


FIGURE 4.9. Left: *The Bayes (purple dashed), LDA (black dotted), and QDA (green solid) decision boundaries for a two-class problem with* $\mathbf{\Sigma}_1 = \mathbf{\Sigma}_2$. *The shading indicates the QDA decision rule. Since the Bayes decision boundary is linear, it is more accurately approximated by LDA than by QDA.* Right: *Details are as given in the left-hand panel, except that* $\mathbf{\Sigma}_1 \neq \mathbf{\Sigma}_2$. *Since the Bayes decision boundary is non-linear, it is more accurately approximated by QDA than by LDA.*

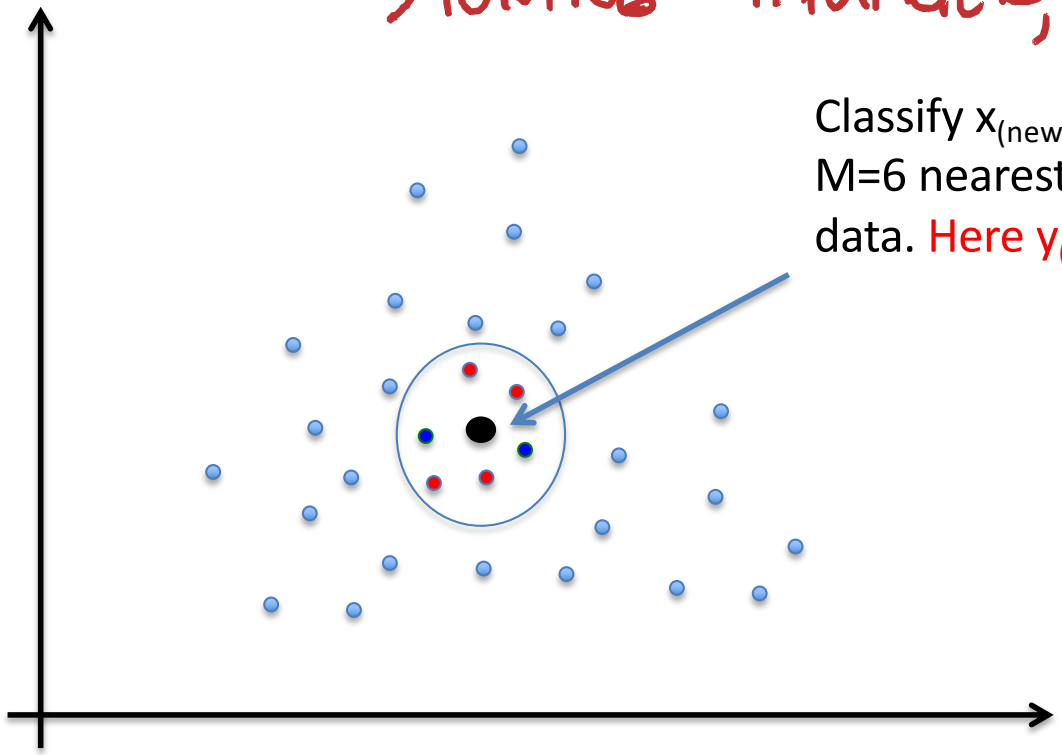
K-NEAREST NEIGHBORS CLASSIFICATION

NON PARAMETRIC!! ~ NOT p(x|y) nor p(y|x)

Say M nearest, K used for number of classes (Chapter 2)

- Completely *non-parametric*!
- The boundaries can take any shape; no modeling
- Without modeling one may have less power and/or accuracy
- Need to *select M* appropriately (*tuning parameter* of the algorithm; a "degree of smoothing")

→ TUNING PARAMETER, THRESHOLD



Classify $x_{(new)}$ with a majority vote by the M=6 nearest neighbors in the training data. Here $y_{(new)}$ = red.

CLASSIFICATION ACCURACY: Training set and Testing set error (misclassification) rates

$$Err_{(train)} = \frac{1}{n_{(train)}} \sum_{i \in Train} Ind(\hat{y}_i \neq y_i)$$

$$Err_{(test)} = \frac{1}{n_{(test)}} \sum_{i \in Test} Ind(\hat{y}_i \neq y_i)$$

Better estimate of the performance of the classifier (on training set, may overfit)

Bayes classifier (and L(Q)DA approximations of it) have optimality properties in terms of overall misclassification rates.

But what if the training set is very unbalanced, misclassification rates for different classes may differ; even with a good overall classification performance, some classes (the least sampled) may be predicted very poorly!

e.g., in the credit default example used in the book, overall misclassification rate is rather low with both Logistic regression and LDA, but misclassification rate for people who do default (a very small fraction of the total) is as bad as ~75%! (and this is likely what the researchers care about the most).

Can move boundaries (thresholds) as to improve classification for a class of interest.

		True default status		
		No	Yes	Total
Predicted default status	No	9,432	138	9,570
	Yes	235	195	430
	Total	9,667	333	10,000

TABLE 4.5. *A confusion matrix compares the LDA predictions to the true default statuses for the* 10,000 *training observations in the* `Default` *data set, using a modified threshold value that predicts default for any individuals whose posterior default probability exceeds* 20 %.

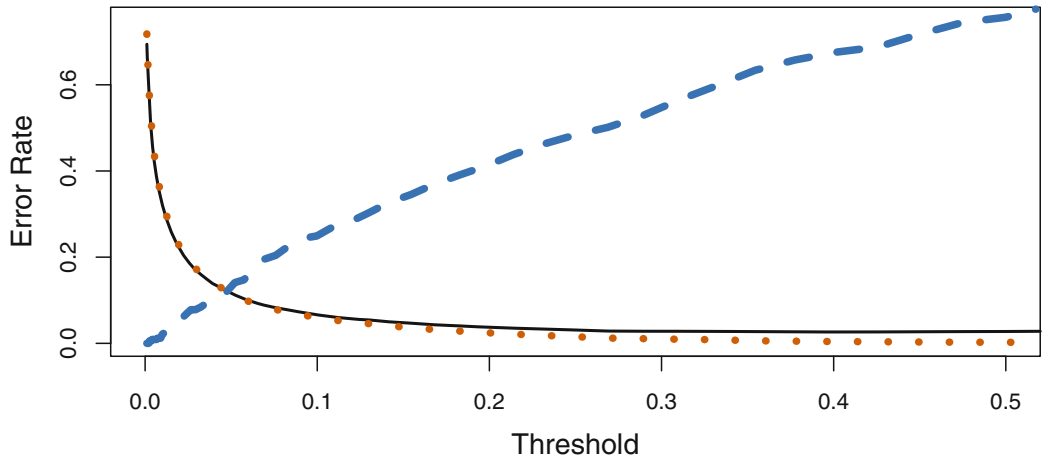
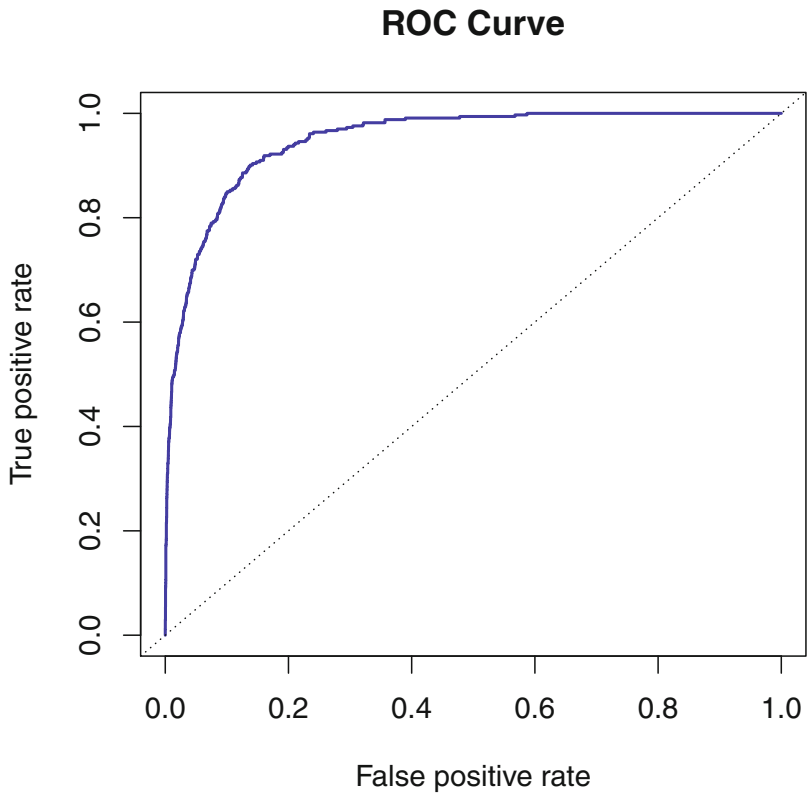


FIGURE 4.7. *For the* `Default` *data set, error rates are shown as a function of the threshold value for the posterior probability that is used to perform the assignment. The black solid line displays the overall error rate. The blue dashed line represents the fraction of defaulting customers that are incorrectly classified, and the orange dotted line indicates the fraction of errors among the non-defaulting customers.*



Area Under the Curve (AUD) is a measure of the quality of the classifier – across choices of boundaries/ thresholds

FIGURE 4.8. *A ROC curve for the LDA classifier on the* `Default` *data. It traces out two types of error as we vary the threshold value for the posterior probability of default. The actual thresholds are not shown. The true positive rate is the sensitivity: the fraction of defaulters that are correctly identified, using a given threshold value. The false positive rate is 1-specificity: the fraction of non-defaulters that we classify incorrectly as defaulters, using that same threshold value. The ideal ROC curve hugs the top left corner, indicating a high true positive rate and a low false positive rate. The dotted line represents the "no information" classifier; this is what we would expect if student status and credit card balance are not associated with probability of default.*

Nomenclature on classifiers' performance; K=2

		Predicted class		
		− or Null	+ or Non-null	Total
True class	− or Null	True Neg. (TN)	False Pos. (FP)	N
	+ or Non-null	False Neg. (FN)	True Pos. (TP)	P
	Total	N^*	P^*	

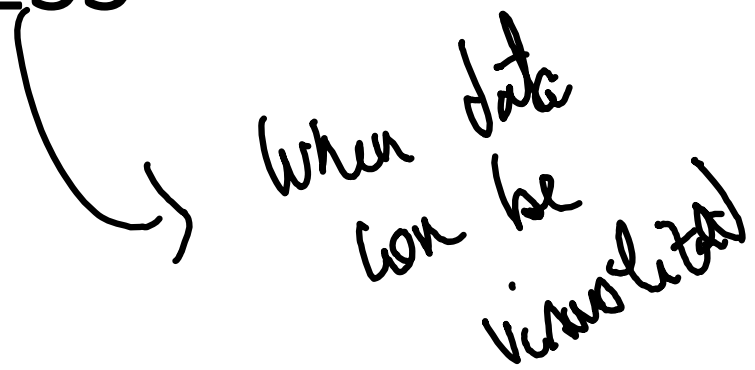
TABLE 4.6. *Possible results when applying a classifier or diagnostic test to a population.*

Name	Definition	Synonyms
False Pos. rate	FP/N	Type I error, 1−Specificity
True Pos. rate	TP/P	1−Type II error, power, sensitivity, recall
Pos. Pred. value	TP/P^*	Precision, 1−false discovery proportion
Neg. Pred. value	TN/N^*	

TABLE 4.7. *Important measures for classification and diagnostic testing, derived from quantities in Table 4.6.*

Outline: **Smoothing**
(F. Chiaromonte)

Non-parametric regression with the LOWESS



IF i don't want to specify equations ... ?

$$E(Y|X) = Y = f(X) \sim \beta_0 + \beta_1 X + \varepsilon \sim \beta_0 + \beta_1 X^2 + \varepsilon$$

(f(X) ~ continuous; $\beta_0 + \beta_1 X + \varepsilon$ ~ fit to squares, estimate them; $\beta_0 + \beta_1 X^2 + \varepsilon$ ~ parametric model)

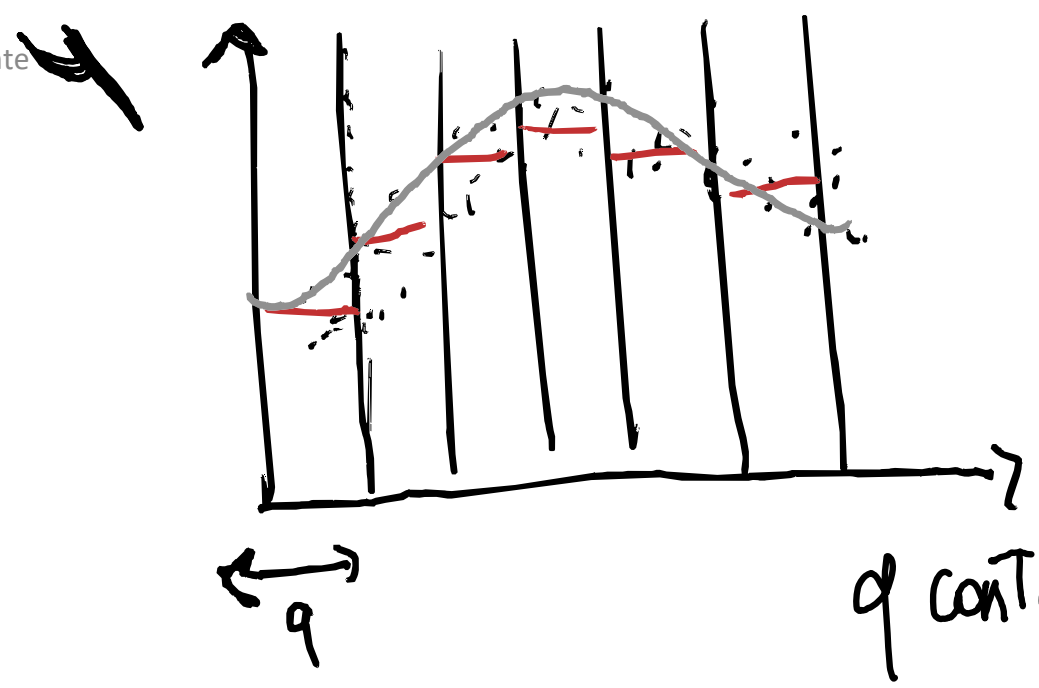
Non-parametric regression

Do without the specification of a parametric model (an explicit equation), for the regression function. If p is very small (p=1,2) and we have a fairly large n, we can instead let the data suggest the shape of the systematic relationship linking the response to the features – using a smoothing technique.

For instance, **LOWESS**: *Locally Weighted Scatter Plot Smoothing.* Implemented in most statistical software packages, including R. See
https://stat.ethz.ch/R-manual/R-devel/library/stats/html/lowess.html

The procedure uses local Least Square fits of linear or quadratic forms, with weights and iterations (to dampen effect of outliers).

To implement it, one needs to specify:
- the fraction of data in each local neighborhood (smoothing parameter, q in (0,1))
- Degree of locally fitted polynomial (1=linear most common, but also 2=quadratic)
- Weight function for the least square fit
- Number of iterative weighted least square fits

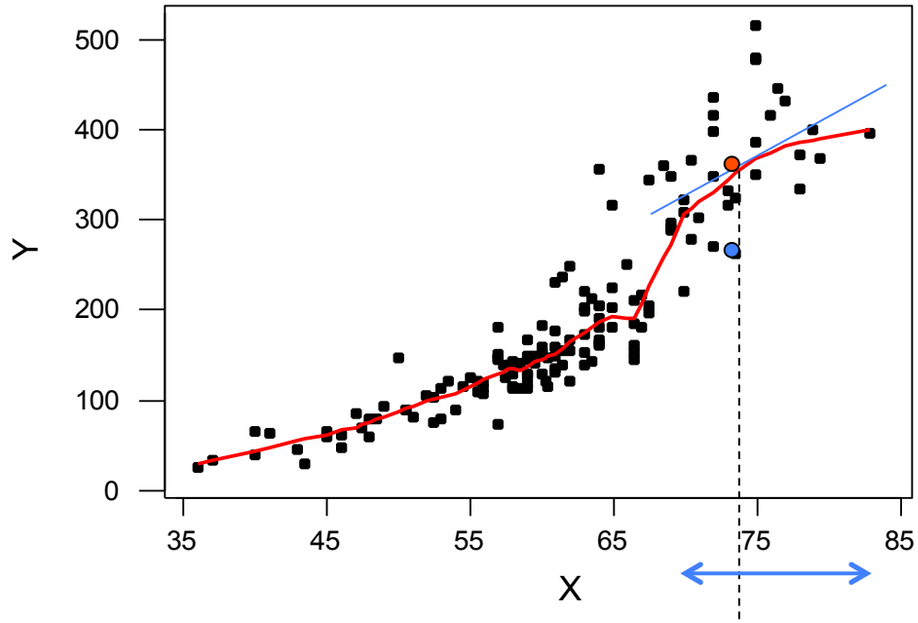


(on regions, more robust)

- Compute AVG of Y in bin
- Moving windows, avg Y
- Don't fix q, fix # things in q

q controls the smoothingness

For instance:
Smoothing param = 0.2
Degree = 1
# iterations = 2



VERTICALLY → ROBUST TO FLUX (moving median)

→ DROP POINTS WITH HIGH RESIDUALS

Weighted least squares linear fit

$$\min_{\beta_0,\beta_1} \sum_{j \in N_i} w_j (y_j - (\beta_0 + \beta_1 x_j))^2$$

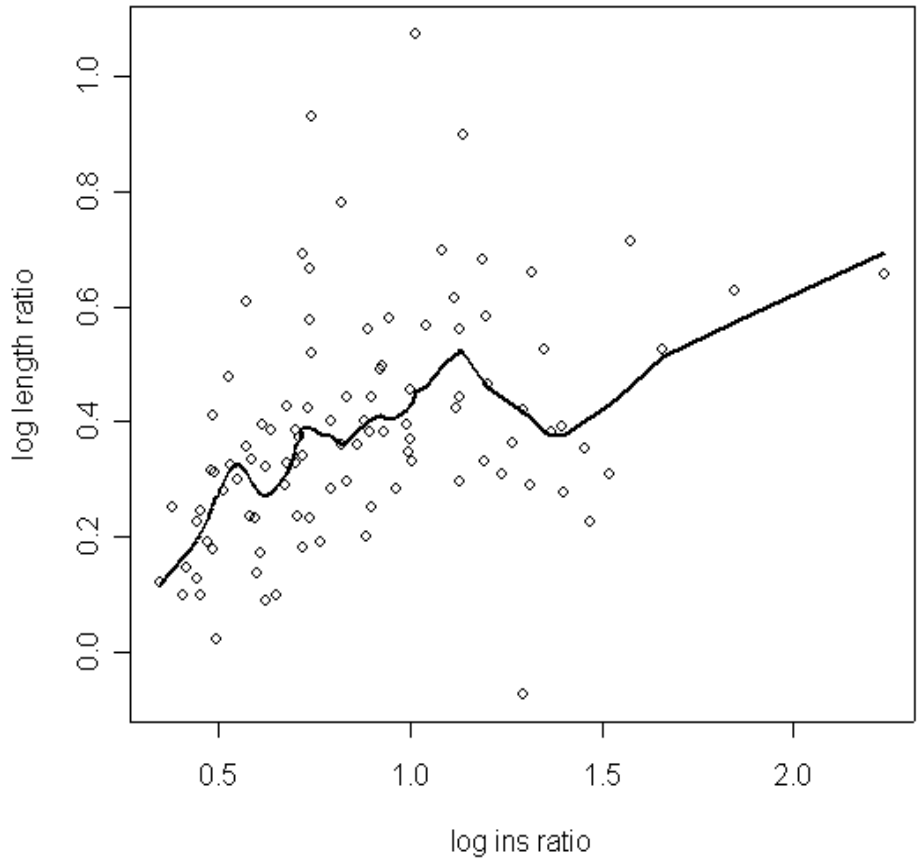
→ locally

Weight function

$$w_i = \left(1 - \left(\frac{d(x_j, x_i)}{\max_{l \in N_i} d(x_l, x_i)}\right)^3\right)^3$$

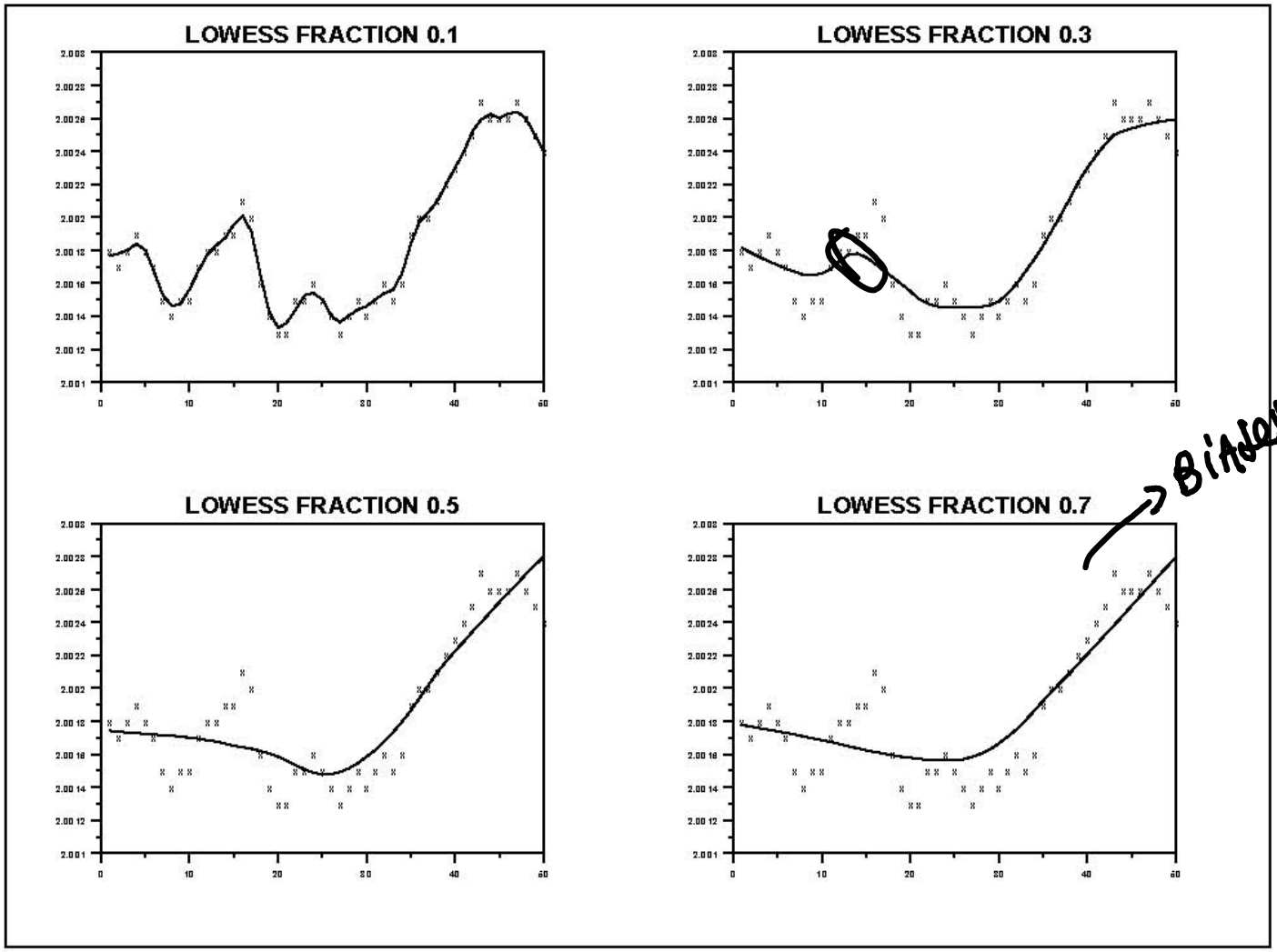
→ weight more point close

An example: first degree, q=0.2, # iterations = 2



Systematic or error

Fraction controls degree of smoothing:



→ Biased

Helps you catch a pattern!!

Bias - Variance Trade off

Regression doesn't imply causality !!

Uses of the LOWESS together with *parametric* regression:

→ AN AID

• On the plot of Y vs X, can suggest an appropriate regression function (the form of an explicit equation may be suggested by the smoother, and one can then fit the corresponding parametric model).
• On the plot of residuals vs X or vs the fitted values; helps diagnosing departures from whatever regression funxtion was postulated (e.g. a line) – visualizing systematic mean patterns in the residuals.

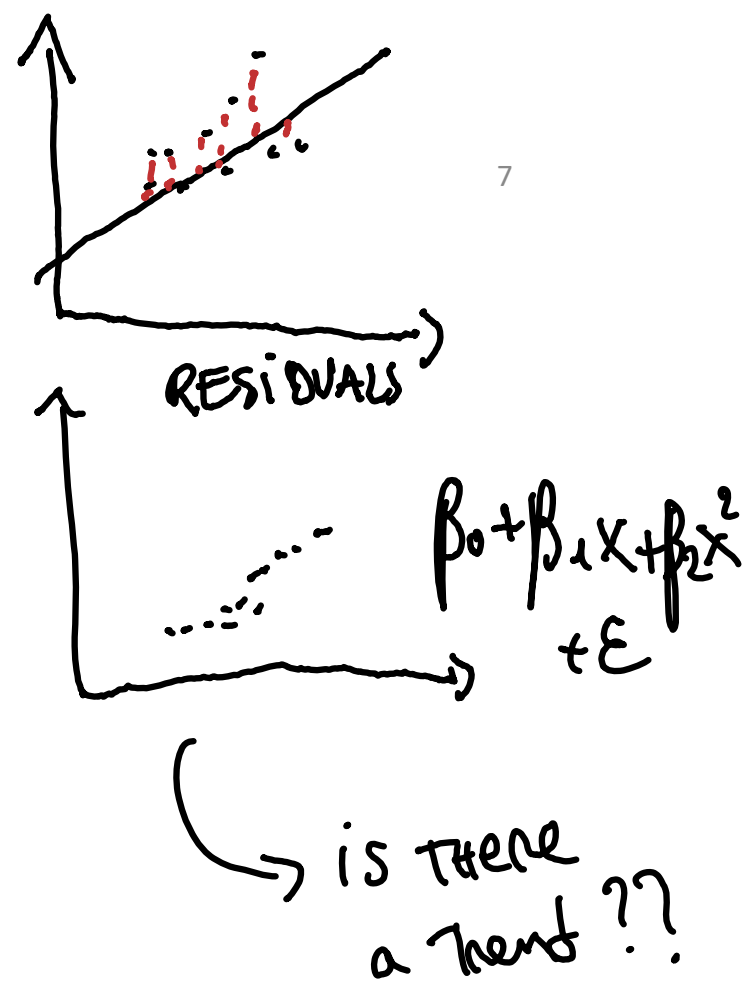
→ Are the Residuals SISTEMATIC ??

Drawbacks of non-parametric regression approaches:

• No explicit equation is obtained for the systematic relationship between response and predictors (but the form of an explicit equation may be suggested)
• The smoothing parameter (a tuning parameter of the procedure) is critical; how does one chose an appropriate level of smoothing? What's systematic signal and what is noise?

Note: also Smoothing Splines and Kernel Smoothing are used in regression. These non-parametric smoothing technique are also used to estimate density functions (less so lowess).

→ lowess is specialized for REGRESSION (it's a R. LOCALLY)



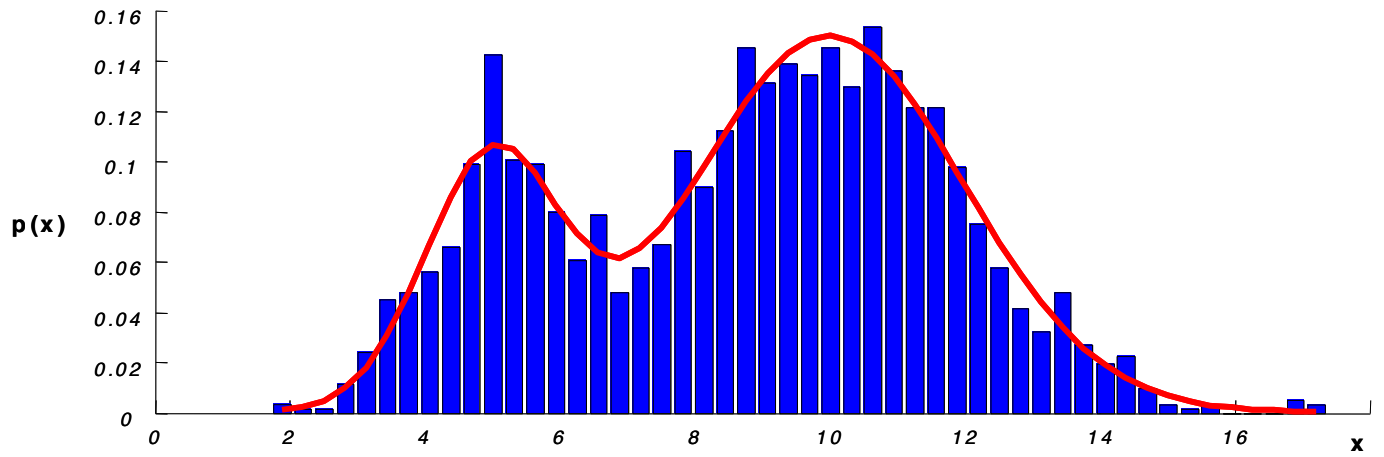
→ is there a trend ??

Density estimation with Kernels

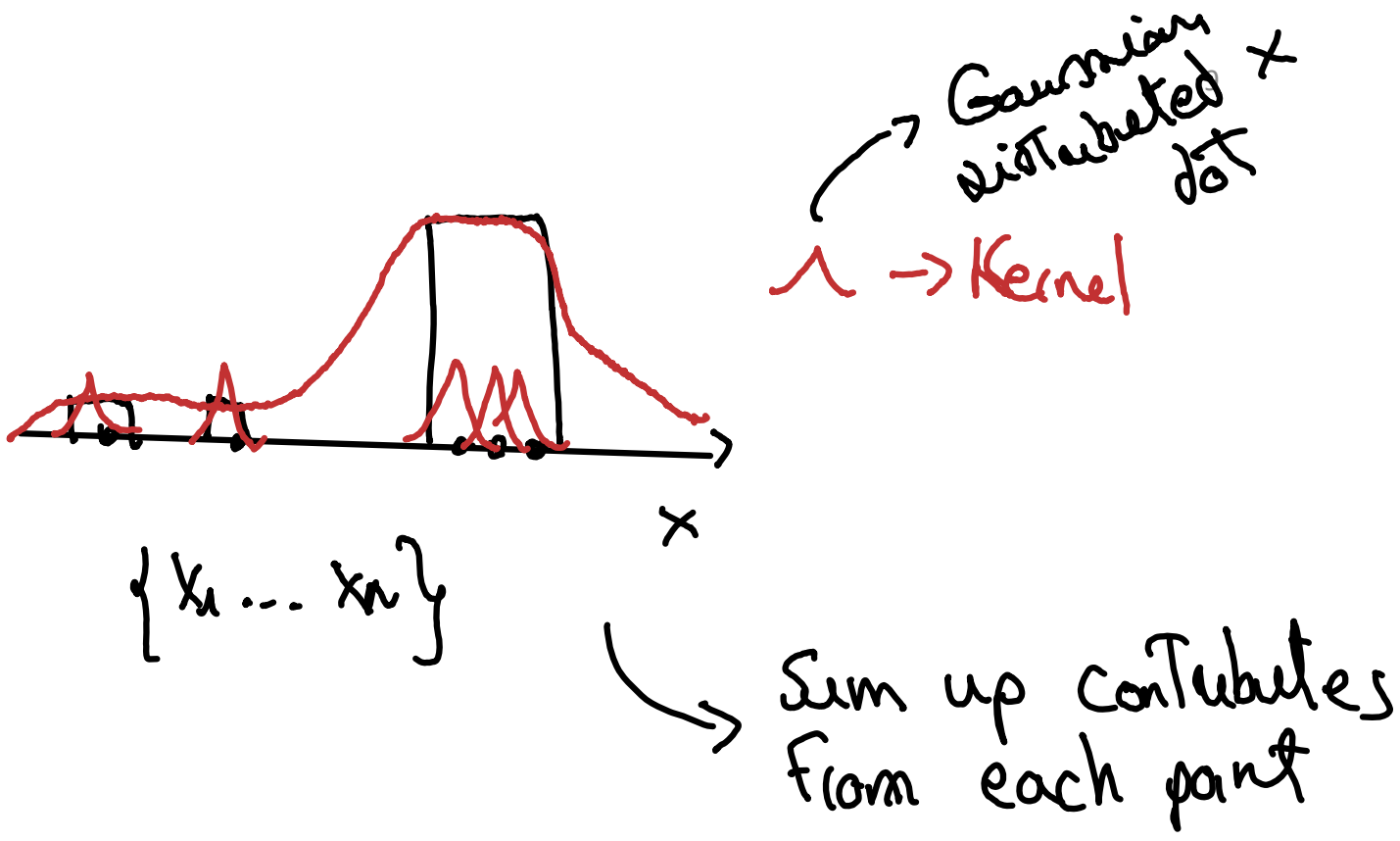
Kernel Density Estimation

DENSITY ESTIMATION

Bimodal density



Instead of the histogram, produce a continuous, smooth estimate of the underlying density based on the empirical distribution (the observed data)



$$\hat{f}_h(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x - x_i) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$

where I put Kernel

where I observe

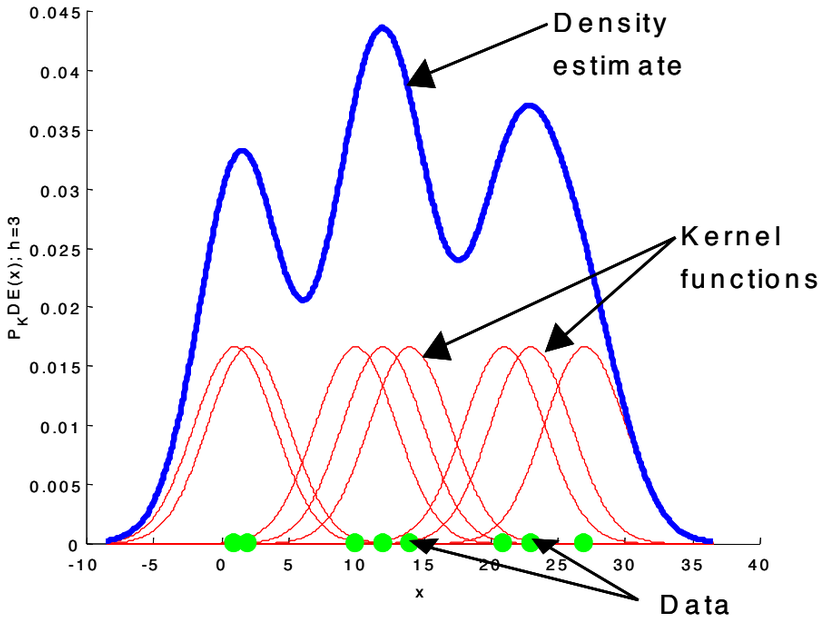
σ^2 of Gaussian

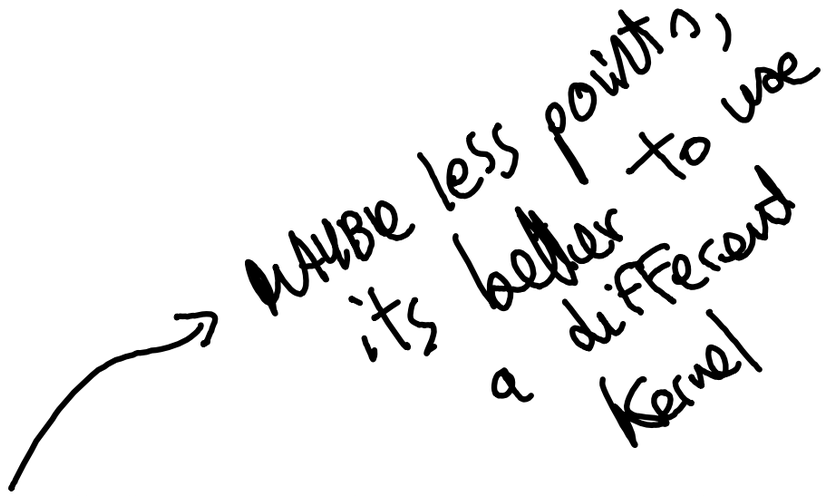
Smoothing

the smooth kernel estimate is a sum of "bumps"

The kernel function determines the shape of the bumps

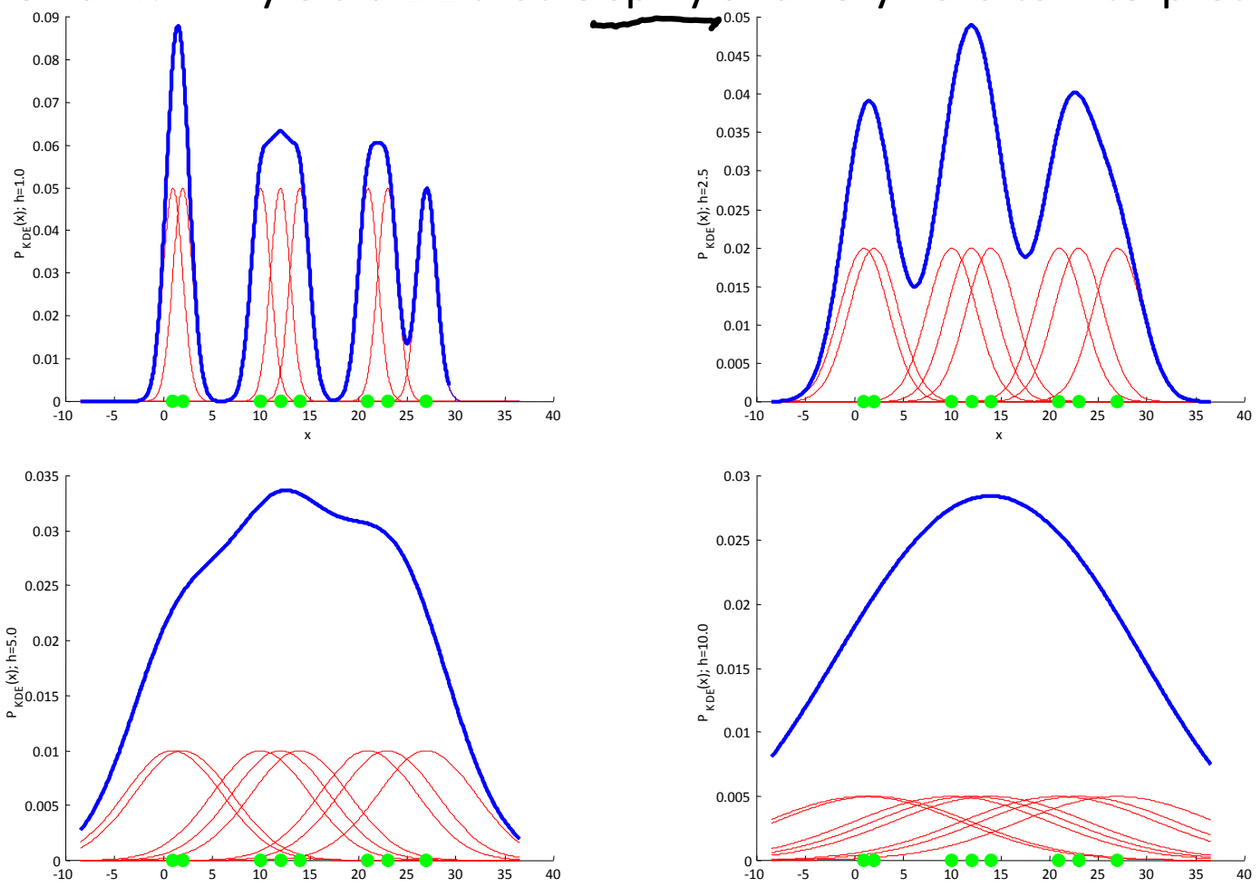
The parameter h, also called the smoothing parameter or bandwidth, determines their width





The problem of choosing h is crucial in density estimation

- A large h will over-smooth the DE and mask the structure of the data
- A small h will yield a DE that is spiky and very hard to interpret



We would like to find a value of h that minimizes the error between the estimated density and the true density

- A natural measure is the MSE at the estimation point x, defined by

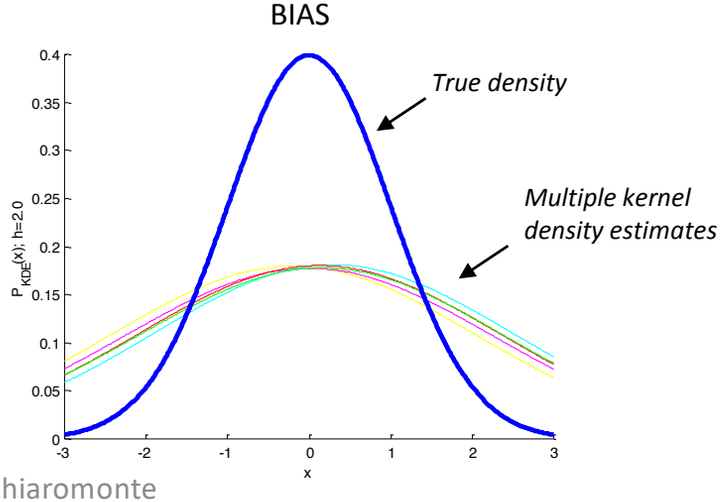
$$E[(p_{KDE}(x) - p(x))^2] = \underbrace{E[p_{KDE}(x) - p(x)]^2}_{bias} + \underbrace{var(p_{KDE}(x))}_{variance}$$

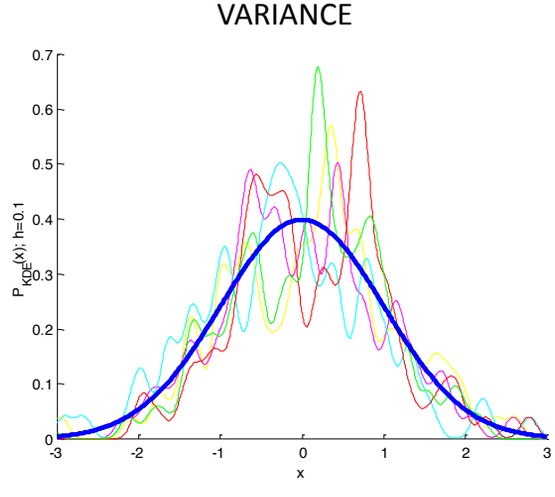
↳ estimated

↳ h

bias-variance tradeoff

- A large bandwidth will reduce the differences among the estimates of $p_{KDE}(x)$ for different data sets (the variance), but it will increase the bias of $p_{KDE}(x)$ with respect to the true density $p(x)$
- A small bandwidth will reduce the bias of $p_{KDE}(x)$, at the expense of a larger variance in the estimates $p_{KDE}(x)$





TOO SMOOTH → UNDERFIT

Outline: **Cross-Validation** (F. Chiaromonte)

Introduction to Statistical Learning Chapter 5 Section 1 Lab 3

COMPUTATIONAL ~~ASSESSMENT~~/TUNING OF STATISTICAL PROCEDURES

For a very long time, the properties of statistical procedures were assessed using mathematical manipulations, facts about probability distributions and asymptotic results. $n \to \infty$

Mathematical tractability defined a narrow scope for statistics:

- Consider only simple procedures
- Introduce strong assumptions on the stochastic mechanism generating the data, and/or
- Prove properties only for large samples

'70s onward: replacing math with computation has substantially expanded the scope.

IN PROCEDURES WHOSE PERFORMANCE DEPENDS CRUCIALLY ON TUNING PARAMETERS, HOW DO WE IDENTIFY THEIR OPTIMAL/SATISFACTORY VALUES… BASED ON THE DATA?

→ h, M, degree of polinomial

OUT-OF-SAMPLE ACCURACY

CROSS VALIDATION: A computational approach to evaluate the out-of-sample accuracy of a statistical procedure used for prediction (supervised problems). Computationally expensive but now viable.

- Given a model or procedure comprising a tuning parameter (e.g. a polynomial regression of degree r; a lowess with smoothing parameter s; a classifier with threshold t)… how do we choose the tuning parameter?
- More generally, given a set of alternative models… how do we chose among them?

E.G. **Mean Squared Error (MSE)** for a regression; prediction of a continuous response (the quantity that is minimized in-sample by Least Squares fitting in the case of parametric linear models).

$$MSE(in) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$\hat{y}_i = \begin{cases} \hat{\beta}_o + \hat{\beta}' x_i & \text{(parametric)} \\ \ell(x_i) & \text{(lowess fit)} \end{cases}$$

continuous

Or E.G. **Misclassification rate (Err)** for a classifier; prediction of a categorical response.

$$Err(in) = \frac{1}{n}\sum_{k=0}^{n} Ind(\hat{y}_i \neq y_i)$$

$$\hat{y}_i = classpred(x_i) \quad \text{(classifier)}$$

categorical

"in" = in-sample, same as "train" (training data for the procedure)

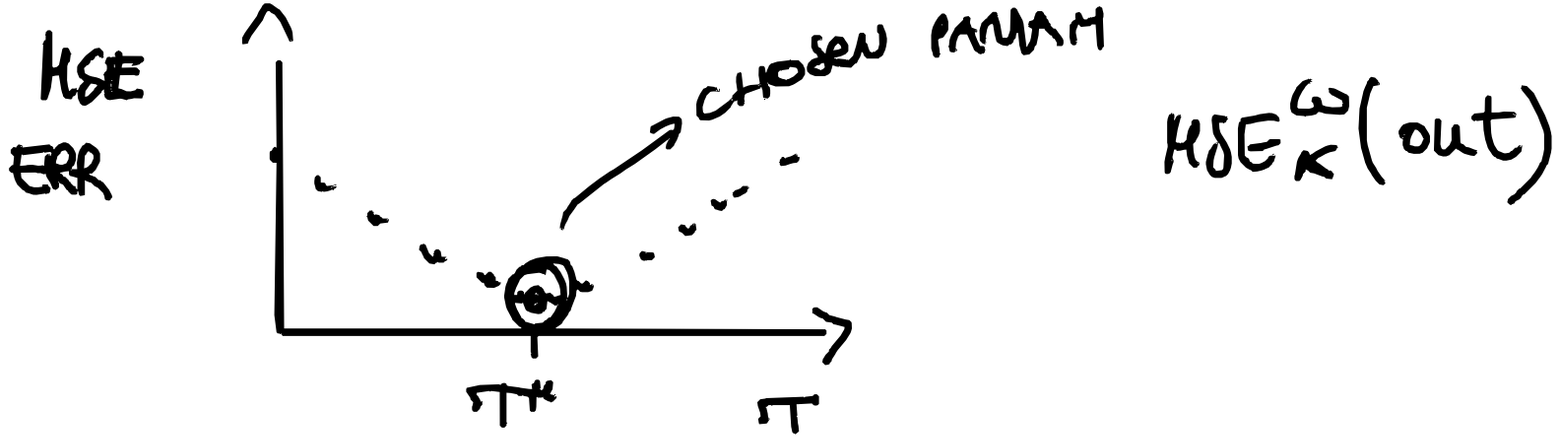
In-sample MSE (Err) can be a poor approximation of **out-of-sample MSE (Err)**; how closely we manage to "reproduce" the training data, especially with a large or complex model having lots of degrees of freedom, can say little about how accurately we would predict the response on independent test data.

In-sample MSE (Err) can substantially underestimate out-of-sample MSE (Err); if we overfit we are learning idiosynchratic components of the training data, not the underlying mechanism.

Traditionally, formulae were developed for specific problems (e.g. Mallow's C_p for regression – good estimator of the out-of-sample MSE under assumptions and for large samples).

Cross Validation (CV) allows us to produce a reliable estimate of the out-of-sample MSE for a collection of values of the tuning parameter (a collection of possible models), and then select the one that provides the lowest.

"2-fold" … k-fold … "n-fold"
(validation or test set) LOOCV (leave-one-out) LOOCV



- Divide the data at random in k subsets of equal size S_1, S_2 … S_k

- For j=1,2 … k

  Form Training and Test Sets
  (sizes n(k-1)/k) and n/k , respectively)

  $$TrSet = \bigcup_{h \neq j} S_h \qquad TsSet = S_j$$

  Train model/procedure on Training Set $\hat{M}_j$

  Compute MSE on Test Set

  $$MSE_j = \frac{1}{n/k} \sum_{i \in S_j} (y_i - \hat{M}_j(x_i))^2$$

- Average measurements over the folds

$$MSE(out)_k^{CV} = \frac{1}{k} \sum_{j=1}^{k} MSE_j$$

Estimates the out-of-sample MSE (Err) without the *underestimation* one has in-sample, but with an *overestimation* due to using a smaller sample size in training!

Select k based on computational burden, as well as a variance-bias trade off.

$\bar{X} = \mu$ $\quad Var(\bar{x}) = \frac{S^2}{n}$

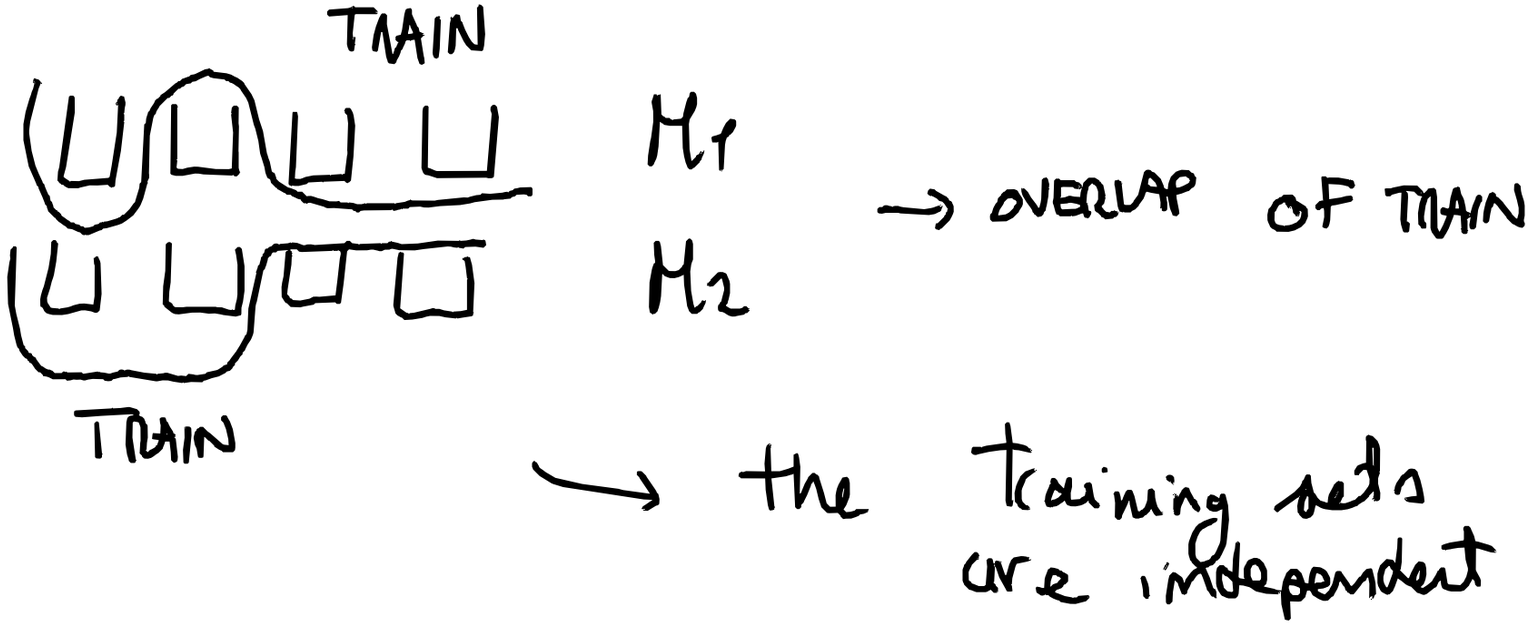
“2-fold” k-fold “n-fold”
(validation or test set) (leave-one-out) LOOCV

As k increases (from 2 to n)

- Computational burden increases.
- Overestimation bias for MSE(out) decreases: training sets used in each fold grow in size, up to n-1 ~ n.
- Variance for estimation of MSE(out) increases: training sets used in each fold have larger overlaps; *while we average more numbers (k) they are more dependent* – less actual replication.

Common reasoning: using k>10 often induces an increase in computational burden and variance that is not justified by a substantial decrease in the overestimation bias…

Note: in the old fashioned “validation (test) set approach”, the calculation is not even repeated flipping the roles of the two folds as training and testing sets.



Outline: **Resampling**
(F. Chiaromonte)

Introduction to Statistical Learning Chapter 5 Section 2 Lab 3

COMPUTATIONAL ASSESSMENT/~~TUNING~~ OF STATISTICAL PROCEDURES

For a very long time, the properties of statistical procedures were assessed using mathematical manipulations, facts about probability distributions and asymptotic results.

Mathematical tractability defined a narrow scope for statistics:

- Consider only simple procedures
- Introduce strong assumptions on the stochastic mechanism generating the data, and/or
- Prove properties only for large samples

’70s onward: replacing math with computation has substantially expanded the scope.

HOW DO WE EVALUATE THE ACCURACY AND SAMPLING VARIABILITY OF PROCEDURES THAT WE CANNOT TACKLE ANALYTICALLY?

~> LOTS OF POINTS

A computational approach to evaluate the accuracy of a statistic used as estimator for a quantity of interest. We can estimate its standard error; in fact, we can approximate its sampling distribution by ***simulating*** its variability across samples.

Setup:

- Sample of n independent observations from a stochastic mechanism

$$x = (x_1 \dots x_n) \ , \ x_i \ iid \sim F$$

independent and identically distributed

- Statistics (function of the observations) produces a real valued estimate of θ

$$\hat{\theta} = g(x)$$

- What is its accuracy for θ? Standard error

$$se(\hat{\theta}) = \sqrt{E[(\hat{\theta} - \theta)^2]}$$

— if $\hat{\theta}$ is unbiased

equal to the standard deviation of the sampling distribution of the estimator, if it is unbiased for θ.

How do we estimate this standard error?

We know the answer for the sample mean as an estimator of the population mean:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\mathrm{E}(\bar{x}) = \mu \qquad \text{UNBIASED}$$

$$se(\bar{x}) = sd(\bar{x}) = \frac{\sigma}{\sqrt{n}}$$

$$\widehat{se}(\bar{x}) = \frac{s}{\sqrt{n}}$$

This is the skeleton of the traditional approach…

- We can proceed similarly for other estimators based on averaging, e.g. estimators of the slope parameters of a regression model.
- We can also extend this logic to estimators that are smooth (differentiable) functions of averages; the Delta method (based on Taylor expansions).

But what if the picture is more complex? e.g. estimating

- A quantile of a univariate population
- The correlation coefficient of a bivariate population
- The covariance eigen-ratio (largest/sum) for a multivariate population…

(note here we are still referring to populations in Euclidian spaces)

- An index defined on a population of curves

which may be hard to do with estimators based on averaging or smooth functions of averages… Whatever

- The nature of the statistics and its domain space
- The nature of the stochastic process F generating the data, and/or
- The size of the sample n (need not resort to limit theorems)

We can use the (one-sample, non-parametric) ***Bootstrap to estimate the standard error***.

The bootstrap algorithm: NON PARAMETRIC

$\hat{\theta} = g(x)$

$E(\hat{\theta}) = \theta$

- Generate B bootstrap samples of size n drawing with replacement from the data

  $x^*_{(b)} \quad b = 1 \dots B$

- On each, compute the statistic – producing B bootstrap values; "copies" that mimic its sampling variability

  $\hat{\theta}^*_{(b)} = g(x^*_{(b)}) \quad b = 1 \dots B$

- Estimate the standard error as the standard deviation of the bootstrap values

$$\widehat{se}_{BT} = \sqrt{\frac{1}{B-1}\sum_{b=1}^{B}(\hat{\theta}^*_{(b)} - \hat{\theta}^*_{(\cdot)})^2} \qquad \hat{\theta}^*_{(\cdot)} = \frac{1}{B}\sum_{b=1}^{B}\hat{\theta}^*_{(b)}$$

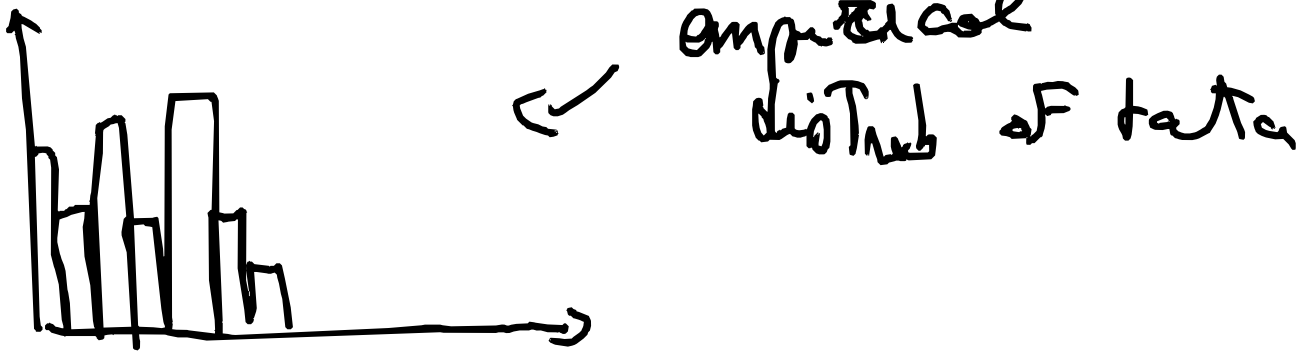
UNBIASED CORRECTION

Already rather good with **B=200**.

Rationale 1: since we cannot take more samples of size n from F, we do the next best thing, i.e. generate them from the empirical distribution $\hat{F}_n$. This is our best estimate of F based on the available sample of n observations (in fact, its non-parametric MLE).

PERTURBATION

Rationale 2: the bootstrap lets us gauge the variability of $\hat{\theta}$ creating perturbations of the original data by resampling (with replacement). These are less local of the perturbations created with the Jackknife (create n samples of size n-1 deleting one observation at a time).



$\hat{\theta}$ computed on the original sample of size n has true standard error $se(\hat{\theta}) = \rho(F; n)$.

The "ideal" bootstrap estimate would be $\rho(\hat{F}_n; n)$ replacing F with $\hat{F}_n$.

As n increases

- The true standard error $\rho(F; n)$ likely gets smaller, closer to 0
- The empirical distribution $\hat{F}_n$ gets closer to F
- The ideal bootstrap estimate $\rho(\hat{F}_n; n)$ gets closer to the (shrinking) true $\rho(F; n)$.

However, for any given n, $\widehat{se}_{BT}$ is an approximation of the ideal $\rho(\hat{F}_n; n)$ based on B bootstrap samples. As B increases

- The bootstrap estimate $\widehat{se}_{BT}$ gets closer to the ideal $\rho(\hat{F}_n; n)$.

One step forward, from estimating standard error to producing ***Confidence Intervals***.

Pivot-based construction of the (1-α) coverage CI for the population mean:

$$CI(\alpha) = \bar{x} \pm m_{\frac{\alpha}{2}} \frac{s}{\sqrt{n}} \qquad m_{(\cdot)} = \Phi^{-1}(\cdot)$$

inverse of the cdf of a N(0,1)

More generally, if we can assume that the sampling distribution is approximately $\hat{\theta} \sim N(\theta; sd(\hat{\theta}))$ (symmetric around quantity of interest, bell shaped, with very fast vanishing tails), we construct the interval as

$$CI(\alpha) = \hat{\theta} \pm m_{\frac{\alpha}{2}} \widehat{se}(\hat{\theta})$$

This works if the estimator is based on averaging or is a smooth function of averages because of the Central Limit Theorem. But if the picture is more complex the interval will be bad; coverage not guaranteed.

SAMPLING VARIABILITY

Traditional approach:

- Figure out an invertible transformation $\varphi(\theta)$ and an estimator for it such that $\hat{\varphi} \sim N\left(\varphi; sd(\hat{\varphi})\right)$
- Build a pivot-based (1-α) coverage interval for the transformation

$$LOW = \hat{\varphi} - m_{\frac{\alpha}{2}} \widehat{se}(\hat{\varphi}) \qquad UP = \hat{\varphi} + m_{\frac{\alpha}{2}} \widehat{se}(\hat{\varphi})$$

- Back-transform (invariance of coverage) to a (1-a) coverage interval for the quantity of interest

$$LOW = \varphi^{-1}(\hat{\varphi} - m_{\frac{\alpha}{2}} \widehat{se}(\hat{\varphi})) \qquad UP = \varphi^{-1}(\hat{\varphi} + m_{\frac{\alpha}{2}} \widehat{se}(\hat{\varphi}))$$

But what if it is hard or impossible to figure out an appropriate “normalizing” transformation?

In the bootstrap world, we could naively take (**Bootstrap Standard Confidence Interval**)

$$CI(\alpha) = \hat{\theta} \pm m_{\frac{\alpha}{2}} \widehat{se}_{BT}$$

but this is unlikely to be a good choice as it relies on assumptions on the nature of the sampling distribution that may not be met.

if we are willing to use computer power and generate **B=1000, 2000** bootstrap samples (instead of B=200) we can approximate the traditional approach without having to figure out the normalizing transformation.

Once again, replace math with computation and expand the scope!

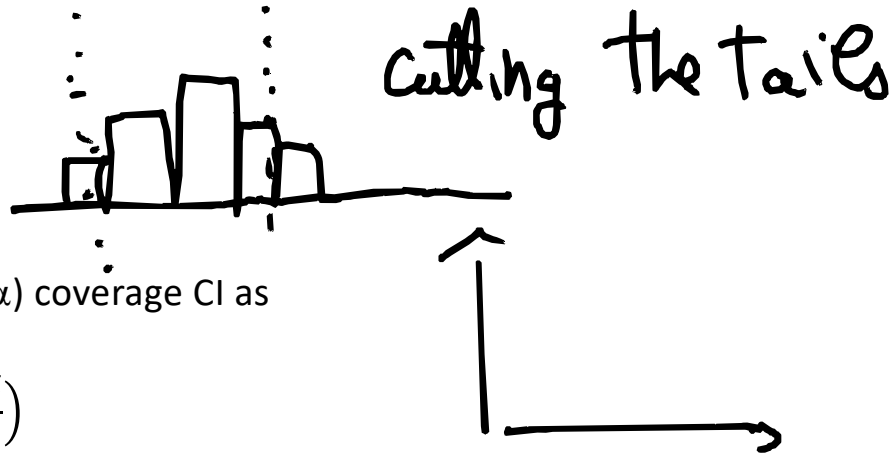
Bootstrap Percentile Confidence Interval:

- Generate B bootstrap samples of size n drawing with replacement from the data

  $x^*_{(b)} \quad b = 1 \ldots B$

- On each, compute the statistic – producing B bootstrap values; "copies" that mimic its sampling variability

  $\hat{\theta}^*_{(b)} = g(x^*_{(b)}) \quad b = 1 \ldots B$



- Build the corresponding empirical cdf $\hat{G}$

- Use its percentiles to define a non-pivot-based (1-α) coverage CI as

  $LOW = G^{-1}\left(\frac{\alpha}{2}\right) \qquad UP = G^{-1}\left(1 - \frac{\alpha}{2}\right)$

Not the ultimate answer; this can be further improved introducing a correction for bias and potential non-constant variance in "tracking" θ with $\hat{\theta}^*_{(\cdot)}$.

Some additions:

variations on the bootstrap and random permutations

Multi-sample Bootstrap:

→ Reproduce mechanism of original sampling

Samples from two univariate populations, e.g. cholesterol for individuals with/out metabolic syndrome:

$x^{(1)} = (x_1^{(1)} \dots x_{n1}^{(1)})$, $x_i^{(1)}$ $iid \sim F^{(1)}$ $\qquad$ $x^{(2)} = (x_1^{(2)} \dots x_{n2}^{(2)})$, $x_i^{(2)}$ $iid \sim F^{(2)}$

Of interest θ = shift in medians between the populations

Estimator: $\hat{\theta} = g\left(x^{(1)}, x^{(2)}\right) = \text{Med}\left(x^{(2)}\right) - \text{Med}\left(x^{(1)}\right)$

- Generate B pairs of bootstrap samples of size n_1 and n_2 drawing with replacement from the data in $x^{(1)}$ and $x^{(2)}$, respectively

  $x_{(b)}^{(1)*}, x_{(b)}^{(2)*} \quad b = 1 \dots B$

- On each pair, compute the statistic – producing B bootstrap values; "copies" that mimic its sampling variability

  $\hat{\theta}_{(b)}^{*} = g(x_{(b)}^{(1)*}, x_{(b)}^{(2)*}) \quad b = 1 \dots B$

The bootstrap must reproduce the original sampling; if we drew samples of size $n_1 + n_2$ with replacement from $x^{(1)} \cup x^{(2)}$, we could get more/less observations of each type, adding inappropriately to the variability of $\hat{\theta}$.

→ if I have some bias more

Parametric Bootstrap:

Sample from a population for which we can postulate a parametric form, e.g. income following a Pareto distribution with $\tau=(\eta,\alpha)$; *minimum* (scale) and *shape* parameters.

$x \quad =(x_1 \ \ldots x_n)$, $x_i \ \ iid \sim F(\tau) \in \mathcal{F}(\tau)$

Of interest θ = 90th percentile ,

Estimator: $\hat{\theta} \quad = g(x) = q_{0.95}(x)$

Compute the MLE $\hat{\tau} = \left(\min\{x_i\}; \frac{n}{\sum \ln(x_i) - \ln(\min\{x_i\})}\right)$ and use $F(\hat{\tau})$ instead of $\hat{F}_n$.

- Generate B bootstrap samples of size n from $F(\hat{\tau})$

  $x^*_{(b)} \quad b = 1 \ldots B$

- On each, compute the statistic – producing B bootstrap values; ”copies” that mimic its sampling variability

  $\hat{\theta}^*_{(b)} = g(x^*_{(b)}) \quad b = 1 \ldots B$

If we drew samples of size n with replacement from x, we would not utilize our knowledge of $\mathcal{F}(\tau)$, adding inappropriately to the variability of $\hat{\theta}$.

$H_0: \rho = 0$

$H_1: \rho \neq 0$

Random permutations:

Studying a relationship, e.g., Y (continuous or categorical) as a function of X (or X's)
A statistic that estimates an association parameter ρ (e.g., correlation coefficient or regression coefficient)
Instead of simulating the sampling distribution of the statistic, simulate the null distribution of that statistic under
H_o: there is no association

Setup:

- Sample of n independent paired measurements from a stochastic mechanism

$$(y, x) \quad = ((y_1, x_1) \dots (y_n, x_n)) \ , \ \ (y_i, x_i)\ iid \sim F$$

- Statistics (function of the observations) produces a real valued estimate of ρ , $\hat{\rho} = r(y, x)$

Algorithm:

- Generate B no-association samples permuting the n entries of y at random

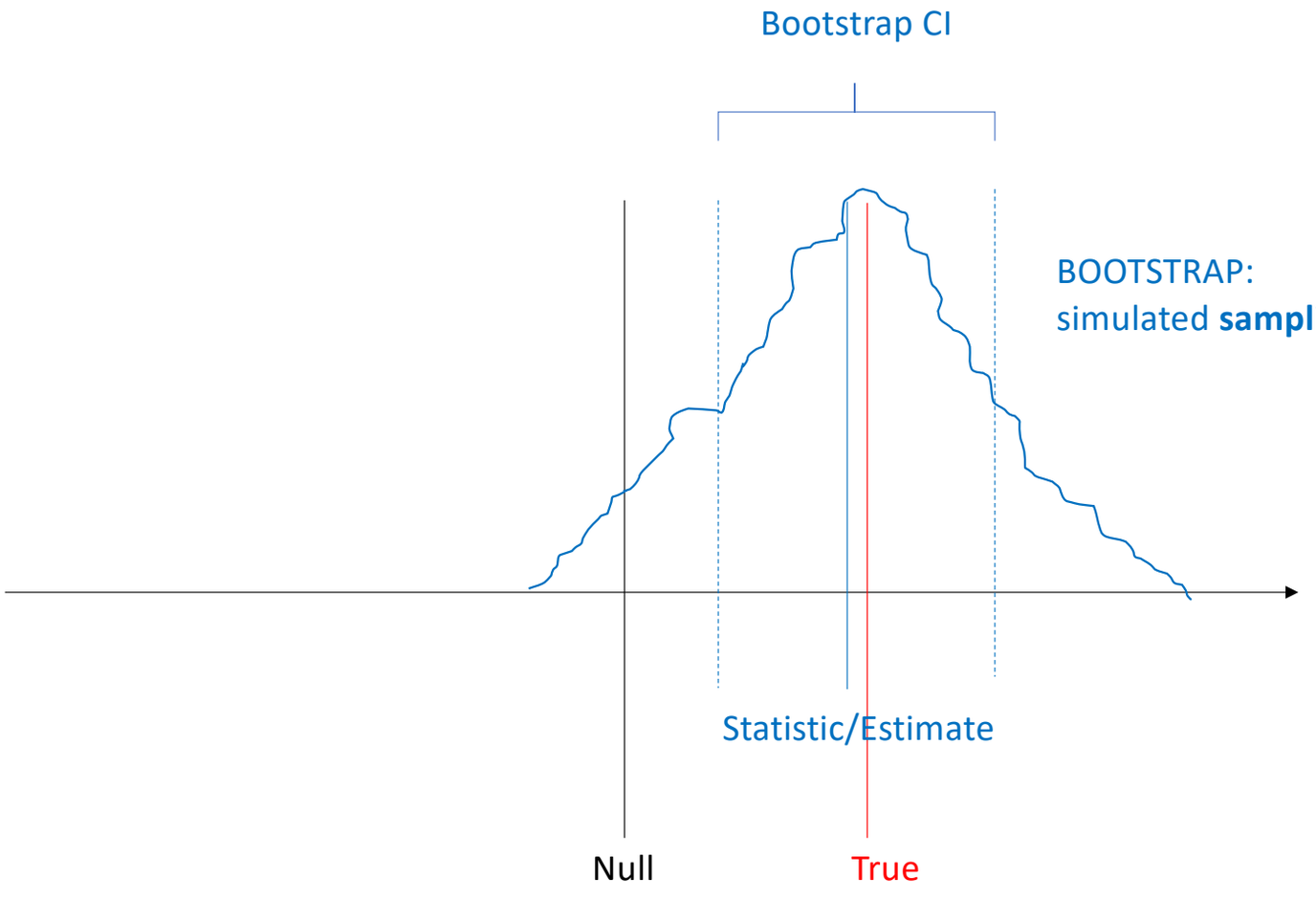
$(y^*_{(b)}, x) \quad b = 1 \dots B$

- On each, compute the statistic – producing B "copies" that mimic its null distribution

$\hat{\rho}^*_{(b)} = r(y^*_{(b)}, x) \quad b = 1 \dots B$

X_1	X_2	…	X_p	Y
x_{11}	x_{12}		x_{1p}	y_1
x_{21}	x_{22}		x_{2p}	y_2
x_{n1}	x_{n2}		x_{np}	y_n

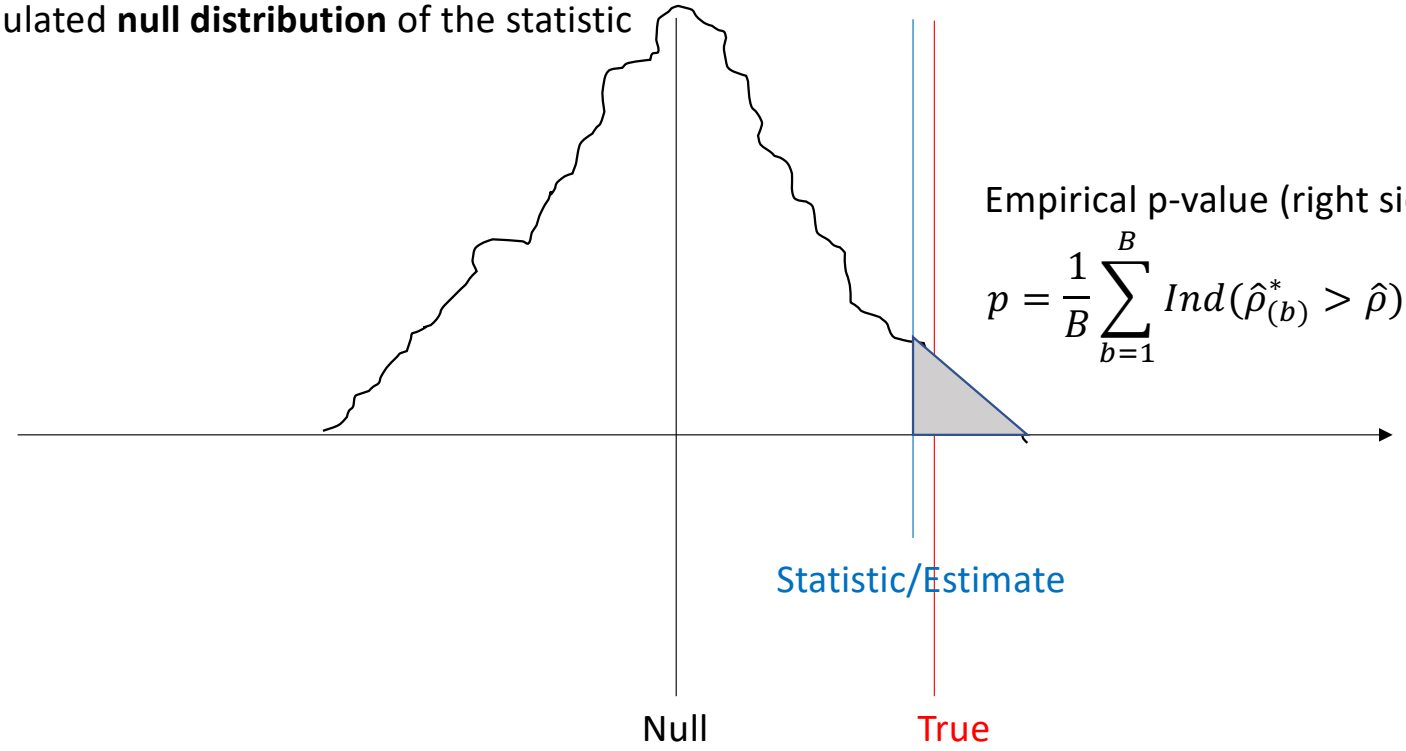
is it significant?



BOOTSTRAP:
simulated **sampling distribution** of the estimator

CL ⟷ HYPOTHESIS TESTING

RANDOM PERMUTATIONS:
simulated **null distribution** of the statistic



Empirical p-value (right sided alternative)

$$p = \frac{1}{B}\sum_{b=1}^{B} Ind(\hat{\rho}^*_{(b)} > \hat{\rho})$$

Outline: (Linear) Models in Large Feature Spaces
Review of Concepts and Notation

(F. Chiaromonte)

Introduction to Statistical Learning

Review (Linear and Generalized Linear Models): Chapters 3, 4

High-dimension, considerations: Chapter 6 Section 4

A QUICK REVIEW OF SOME CONCEPTS AND NOTATION CONCERNING LINEAR MODELS

A classical **linear model** expresses a continuous response Y through an additive function comprising

- an intercept (constant)
- p terms (linear in the slope parameters)
- a random error independent from the random mechanisms underlying the terms.

On a generic observation "i":

$$y_i = \beta_o + \beta' x_i + \varepsilon_i$$

Terms: X = $(X_1 \ldots X_p)$' features (predictors) or specified transformations and functions of the features (e.g., powers, products; observable).

Coefficient parameters: β_o intercept, $\beta = (\beta_1 \ldots \beta_p)$' slopes for each term.

Regression function: the conditional expected value $E(Y|X_1 \ldots X_p) = \beta_o + \beta'X$

Independent random error: added to the regression function and assumed to have mean $E(\varepsilon)=0$ and the same variance parameter $var(\varepsilon)=\sigma^2$ on all observations (homoschedasticity).

In order to develop **inferential procedures**, the random error is also often assumed to be Gaussian:

$$\varepsilon \sim N_1(0,\sigma^2)$$

(T-based confidence intervals and tests for the slope coefficients and the mean response, T-based prediction intervals, F-based ANOVA tests).

The observations are assumed to be iid from the joint distribution of $(Y, X_1 \ldots X_p)$. If $X_1 \ldots X_p$ are viewed as fixed (conditioning; not random), the errors are assumed to be iid across observations. Thus one has, for the vector of errors associated to the n observations in the sample:

$$\underline{\varepsilon}_{(n)} \sim N_n(0,\sigma^2 I)$$

Model in **matrix notation**:

$$\underline{Y}_{(n)} = \underline{1}_{(n)}\beta_o + \underline{X}_{(n,p)}\beta + \underline{\varepsilon}_{(n)}$$

$$\text{for simplicity } \underline{Y}_{(n)} = \underline{X}_{(n,p+1)}\beta + \underline{\varepsilon}_{(n)}$$

$$\underline{X}_{(n,p+1)} = (\underline{1}_{(n)}\ \underline{X}_{(n,p)}) \quad \beta = \begin{pmatrix} \beta_o \\ \beta \end{pmatrix}$$

---

* Intercept = “slope” of the constant term 1 (or assume response is centered, intercept = 0)

Estimating model parameters (fitting):

Because of linearity in the coefficient parameters, whatever the terms represent, fitting can be performed through **least squares** with an explicit, close form solution.

An estimate of the error variance is obtained dividing the minimized sum of squared deviations (Residual Sum of Squares; RSS) by the appropriate number of degrees of freedom.

$$\hat{\beta} = \operatorname{argmin} \| \underline{Y} - \underline{X}\beta \|^2 = (\underline{X}'\underline{X})^{-1}\underline{X}'\underline{Y}$$

$$\hat{\sigma}^2 = \frac{1}{n-(p+1)} \| \underline{Y} - \underline{X}\hat{\beta} \|^2 = \frac{1}{n-(p+1)} RSS$$

Implemented in most statistical software packages, including R. See https://stat.ethz.ch/R-manual/R-devel/library/stats/html/lm.html

If the model is correct (i.e., not misspecified, though some of the coefficients may in fact be 0) the LS coefficient estimators, as well as the LS estimator of the error variance, are **unbiased** for the corresponding parameters.

$$E(\hat{\beta}) = \beta \qquad E(\hat{\sigma}^2) = \sigma^2$$

Moreover, again if the model is not misspecified, the LS coefficient estimators are **BLUE**, i.e., the best (most accurate; minimum sampling variance) among unbiased estimators expressed as linear functions of the response observations – **Gauss-Markov Theorem**.

Finally, if one assumes Gaussian errors, the LS coefficient estimators coincide with the **Maximum Likelihood** (ML) estimators – and the ML estimator for the error variance, which is biased, divides the minimized sum of squares by n instead of the degrees of freedom.

Why? The exponent of the Gaussian likelihood is inversely proportional to the sum of squared deviations:

$$L(\beta \mid \underline{Y}; \underline{X}) \propto \exp\left\{-\frac{1}{2\sigma^2} \| \underline{Y} - \underline{X}\beta \|^2\right\}$$

Residuals diagnostics and **remedial measures:**

Lots of techniques that, based on residuals (observable proxies for the unobservable errors) aim to detect departures from

$$\underline{\varepsilon}_{(n)} \sim N_n(0,\sigma^2 I)$$

i.e., Gaussian iid random errors with 0 mean and shared variance σ^2.

Lots of approaches to manipulate the data, e.g.:

- variance stabilizing transformations of the response to address heteroschedasticity
- identification and removal of outliers/influential observations (*case diagnostics*)

and the model specification, e.g.:

- add/remove terms to address mean patterns in the residuals

to come closer to meeting the assumptions.

Also important, **Multicollinearity**:

Linear dependencies among predictors increase the variance in effect estimates, may even make LS solution unstable.

Diagnose through

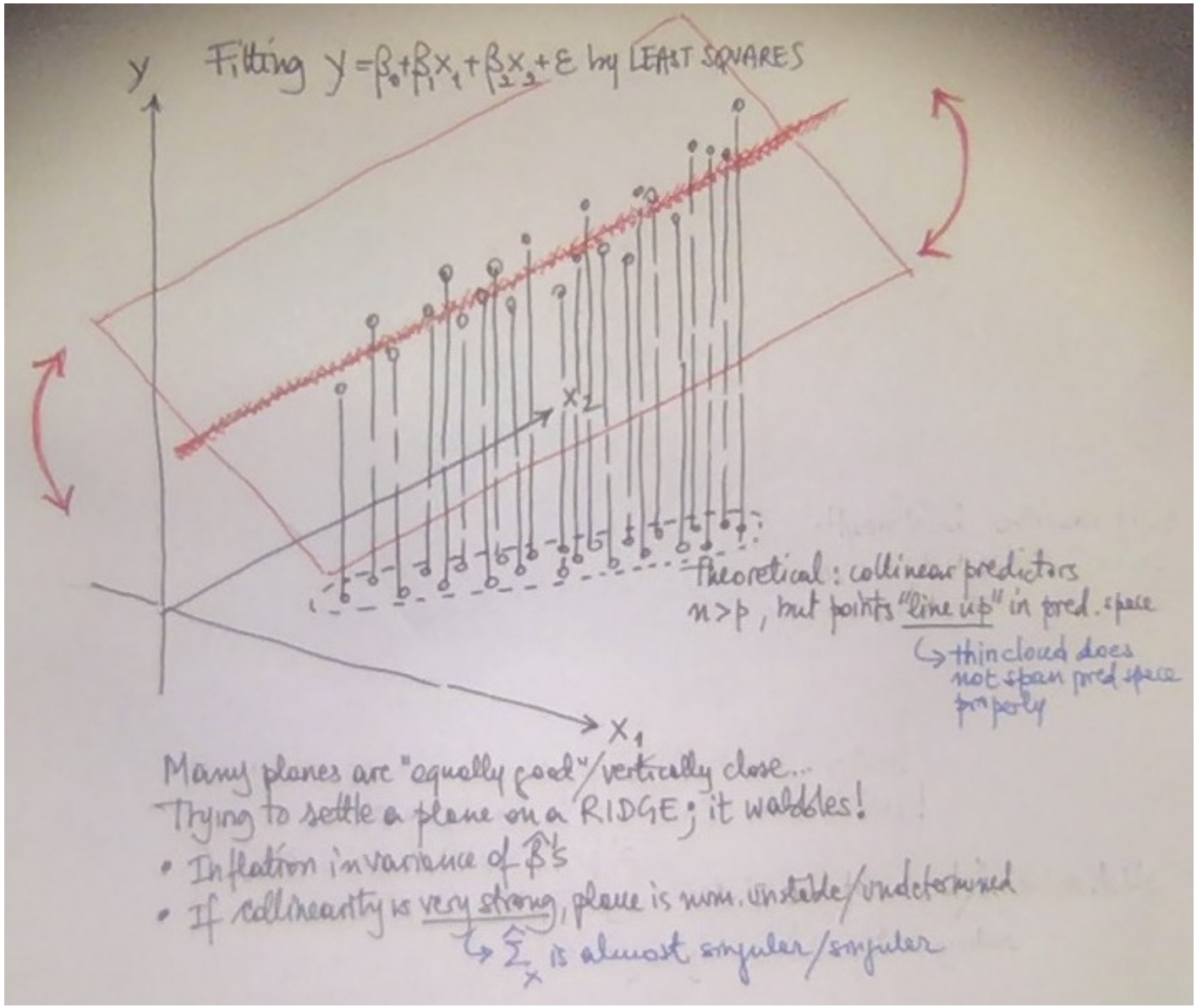
- Pair-wise correlations or scatter-plot matrices
- More complete, **partial R-squares** and **Variance Inflation Factors** (do not depend on the response)

$$R^2_{X_j|\ other\ Xs} \qquad VIF_j = \frac{1}{1 - R^2_{X_j|\ other\ Xs}}$$

Remedies

- stabilize the fit (Ridge)
- eliminate some predictors (LASSO, Best Subsets – feature selection)
- reduce dimension creating "composite" predictors (unsupervised, e.g., Principal Components Analysis; supervised, e.g., Sliced Inverse Regression)

… coming next!


Fitting $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \varepsilon$ by LEAST SQUARES
y
x_1
x_2
theoretical: collinear predictors
$n > p$, but points "line up" in pred. space
↳ thin cloud does not span pred. space for poly
Many planes are "equally good"/vertically close...
Trying to settle a plane on a RIDGE; it wobbles!
• Inflation in variance of $\hat{\beta}$'s
• If collinearity is very strong, plane is num. unstable/undetermined
↳ $\hat{\Sigma}_x$ is almost singular/singular

GENERALIZATIONS

- Linear models can be extended to comprise **categorical predictors**, properly encoding (dummies) their main effects and interactions with the continuos predictors.

- In **Generalized Linear Models** link functions are introduced to represent the dependence of a non-continuos response Y on the predictors $X_1 \ldots X_p$, and the stochasticity of $Y \mid X_1 \ldots X_p$ is modeled differently, not through an additive random error:
  - Counts (Poisson regression)
  - Binary labels (Logit or Binomial regression; Probit regression)
  - Multi-class labels (Multinomial regression)

  All (including the standard Normal regression) are implemented in the R package **GLM**
  https://stat.ethz.ch/R-manual/R-devel/library/stats/html/glm.html

  For instance, Logistic regression: $g(x_i) = \ln\left(\frac{p(x_i)}{1-p(x_i)}\right) = \beta_o + \beta' x_i \qquad y_i \sim Bernoulli(p(x_i))$

  Note: $p(X)=E(Y|X_1 \ldots X_p)$. LS estimation is replaced by ML estimation – often implemented numerically; more computation.

- In heteroschedastic cases, and/or when modeling non-iid data (time series, spatial data), special covariance structures for the errors are introduced to represent varying variance and/or correlations among observations: $\underline{\varepsilon}_{(n)} \sim N_n(0, \Sigma(\eta))$.

  LS estimation can be adapted to recover BLUE; **Generalized Least Squares**.
  More sophisticated estimation approaches (based on ML or Bayesian techniques) also exist; more computation necessary.

SOME ADDITIONS: OMITTED VARIABLE BIAS

The OMITTED VARIABLE BIAS (OVB) phenomenon

Important focus for Economics (less so for other fields): Trying to move towards causation, parsing controllable/exogenous and endogenous features. Related emphasis on model misspecification. Suppose an omitted feature Z

(i) affects the response (non-zero coefficient in the true model), and

(ii) correlates with one or more of the X's in the working model (non-zero Cov(X,Z))

then:

$$\underline{Y} = \underline{X}\beta^* + Z\beta_Z + \underline{\varepsilon}^*$$

$$\underline{\varepsilon}^* : \quad E(\underline{\varepsilon}^*) = 0, Cov(\underline{\varepsilon}^*) = \theta^2 I, \text{ indep of X,Z}$$

true model

$$\underline{Y} = \underline{X}\beta + \underline{\varepsilon}$$

$$\underline{\varepsilon} = Z\beta_Z + \underline{\varepsilon}^* : \quad E(\underline{\varepsilon}) \neq 0, Cov(\underline{\varepsilon}) \neq \sigma^2 I, \text{ not indep of X}$$

working model

$$E(\hat{\beta}) = \beta^* + (\underline{X}'\underline{X})^{-1}\underline{X}'Z\beta_Z$$

BIAS (fct of Cov(X,Z))

- The error of the working model includes Z which correlates with the X's; thus, this error is not independent from the random mechanism underlying the X's.
- The LS coefficient estimators for the features in the working model are biased; they subsume parts of the effects of the omitted Z through its correlations with the X's.
- The bias does not vanish with increasing n – causing inconsistency.

IMPORTANT REMARKS:

- Omitting anything that does not carry interdependencies with X's may induce a bigger and/or non-spherical error variability in the model – but does not introduce a bias in the LS estimators of the coefficients of the X's.

- Including more features in the model reduces the risk of OVB – but inflates the variability of the LS coefficients estimators (more below). *Under-specifying vs overfitting; a variance/bias trade-off*.

Outline: (Linear) Models in Large Feature Spaces
Introducing Penalizations: Ridge and LASSO

(F. Chiaromonte)

Introduction to Statistical Learning

Ridge & LASSO: Chapter 6 Section 2, Lab 2

→ Not a strong Feature selection!!

x_1 x_2 ... x_p y

1
2
3
⋮

We have reviewed a very powerful, rich and versatile framework for supervised data analysis.

VARIANCE INFLATION

Why do we need to go beyond it?

Even when all the assumptions comprised in the modeling are right, so in particular the LS coefficient estimators are unbiased, their accuracy (notwithstanding the Gauss-Markov Theorem) may deteriorate severely when the feature space is large relative to the (iid) data at our disposal...

$n \to \infty$

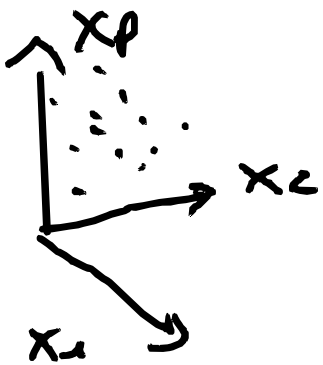
When does this happen?

When the features (terms) have strong linear associations to each other (multicollinearity) and/or n is not very large relative to p (possibly n<p) the data cloud does not "span" the feature space properly; it is "thin", possibly it collapses in lower dimension.

→ numerically instable

As a consequence

- The LS coefficient estimators undergo variance inflation; if the data collapses (almost collapses) in lower dimension the estimates are non-uniquely determined (numerically unstable)
- Relatedly, we run into overfitting; the in-sample MSE may be very small, but the out-of-sample MSE is very large.



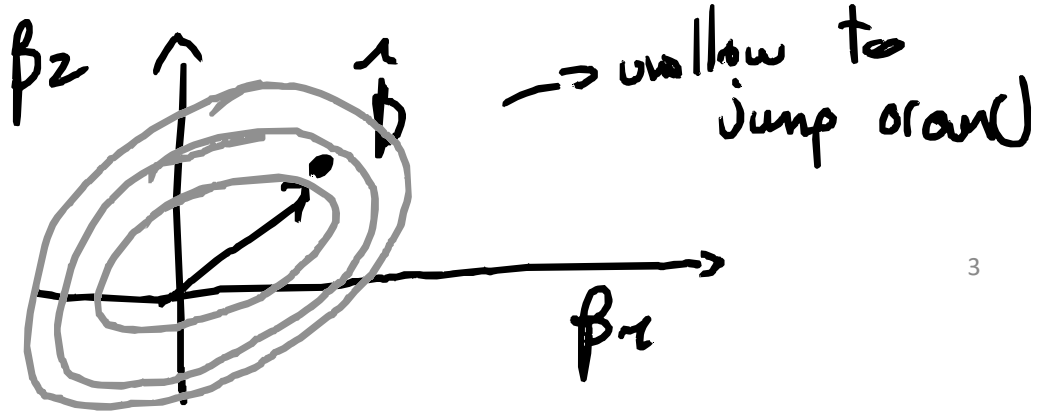
We will focus on a variety of techniques to overcome this problem.

Overarching idea: *constrain the LS as to reduce the estimators' variance and emiliorate overfitting*.

This *introduces a bias*, but the bias may be minor relative to the gain in reducing variance – so overall accuracy improves.

Additionally, constraining may result in a smaller, more *parsimonious and interpretable model* (some features are eliminated, or the focus is shifted to a small number of composite features – linear combinations).

Note: the same approach can be extended to the case of Generalized Linear Models (e.g., a logistic or multinomial regression for binary or multi-class classification, as a measure of accuracy one can considers in-sample and out-of-sample misclassification rates).



- **Shrinkage/Regularization**
  - Ridge Regression
  - LASSO Regression

  Penalized version of LS produces an alternative estimator.

Ridge ('60-'70s): fitting procedure to *overcome multicollinearity* in regressions with highly interdependent features.

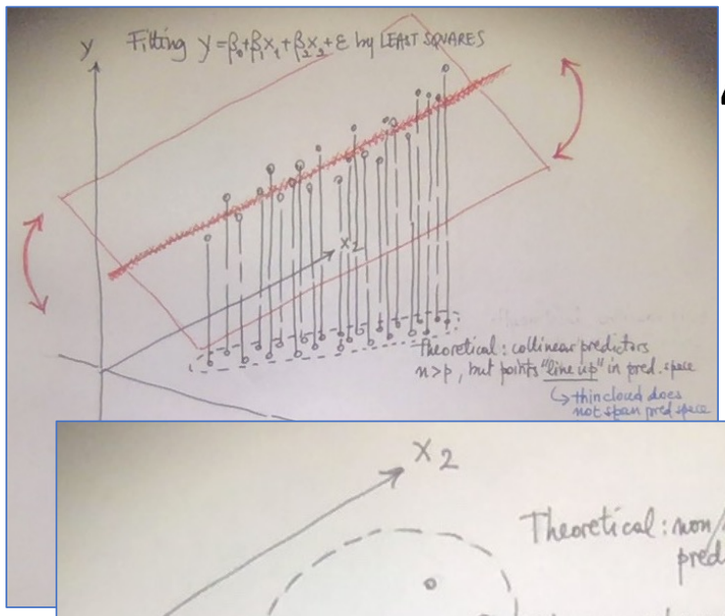
Loosely, in the LS, replaces $S_X=\underline{X}'\underline{X}$ (pxp sample covariance matrix of centered X's) with $S_X(\lambda)=\underline{X}'\underline{X} + \lambda I_p$, $\lambda>0$ which is always invertible – spherically "increasing" the features' spread in the data decreases the sampling variability of the estimator.

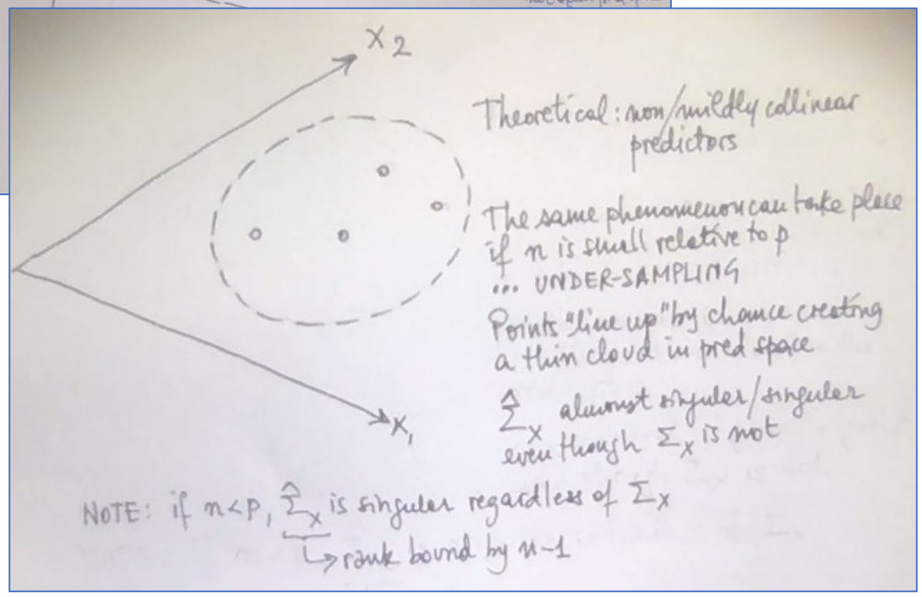
LASSO ('90s): fitting procedure to *produce sparse solutions* in regressions with a large number of features, only a subset of which is expected to matter.

Loosely, it performs "soft" features selection (more below), without specifying how many coefficients should be set to 0.

Both are formulated as a constrained LS: *Size constraint* on β using different norms in R^p.

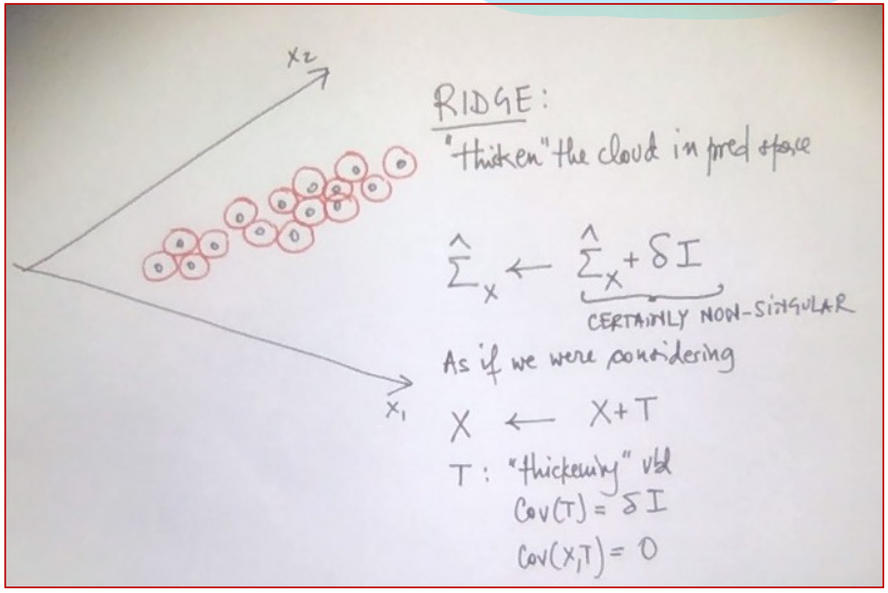
x_1 & x_2 are correlated





UNCORRELATED

$\tilde{X} = X + T$ $\quad Cov(\tilde{X}) = \Sigma_x + \delta I$



$$\hat{\beta}_L = (X^T X)^{-1} X^T y$$

$$\alpha\ \hat{\Sigma}_x + \lambda I$$

Constrained Least Squares (using different norms)

Ridge $$\hat{\beta}_{Ridge} = \text{argmin}\left\{ \| \underline{Y} - \underline{X}\beta \|^2 + \lambda \| \beta \|_{(2)} \right\}$$

$$\| \beta \|_{(2)} = \sum_{j=1}^{p} \beta_j^2 \quad \text{L2 Norm}$$

LASSO $$\hat{\beta}_{LASSO} = \text{argmin}\left\{ \| \underline{Y} - \underline{X}\beta \|^2 + \lambda \| \beta \|_{(1)} \right\}$$

$$\| \beta \|_{(1)} = \sum_{j=1}^{p} | \beta_j | \quad \text{L1 Norm}$$

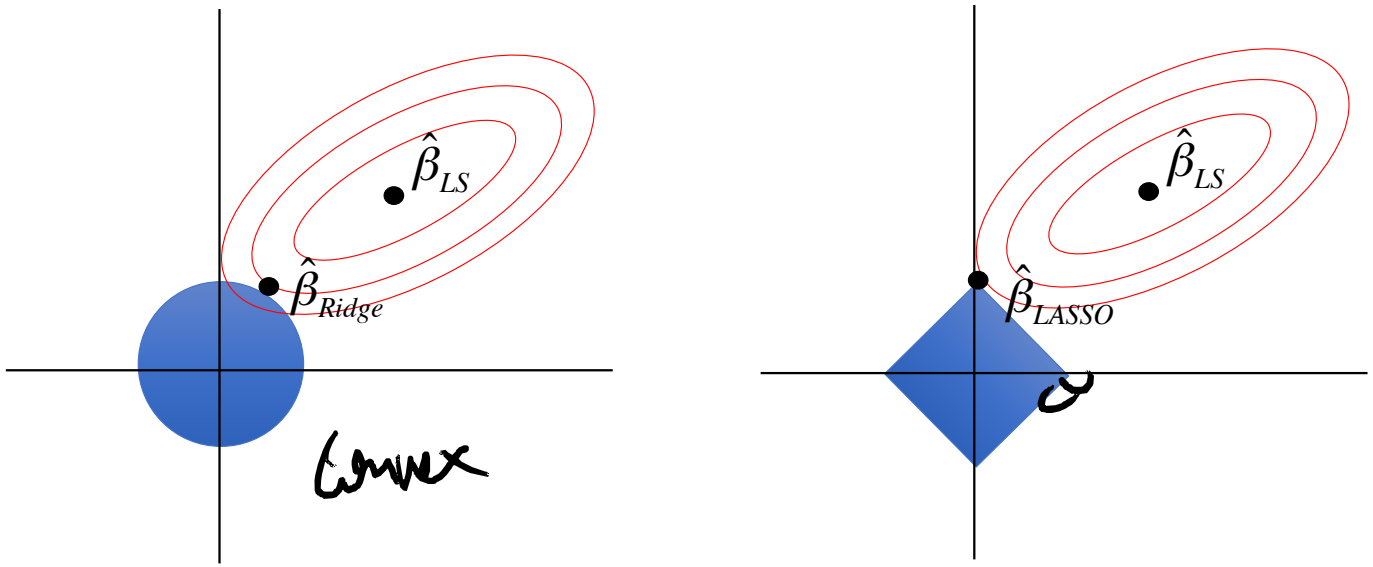
size of β

Note: while LS is scale equivariant, Ridge and LASSO are not – *important to perform them after scaling features* (e.g., divide each by its sd, so all will have the same unitary spread).

↳ need STD

Cartoons with p=2

the diamond shape of the L1 constraint makes it more likely that the solution lies in a corner



$$\| \underline{Y} - \underline{X}\beta \|^2 = \min_{\beta \in R^p}$$

$$\sum_{j=1}^{p} \beta_j^2 \leq s_\lambda$$

size constraint

$$\| \underline{Y} - \underline{X}\beta \|^2 = \min_{\beta \in R^p}$$

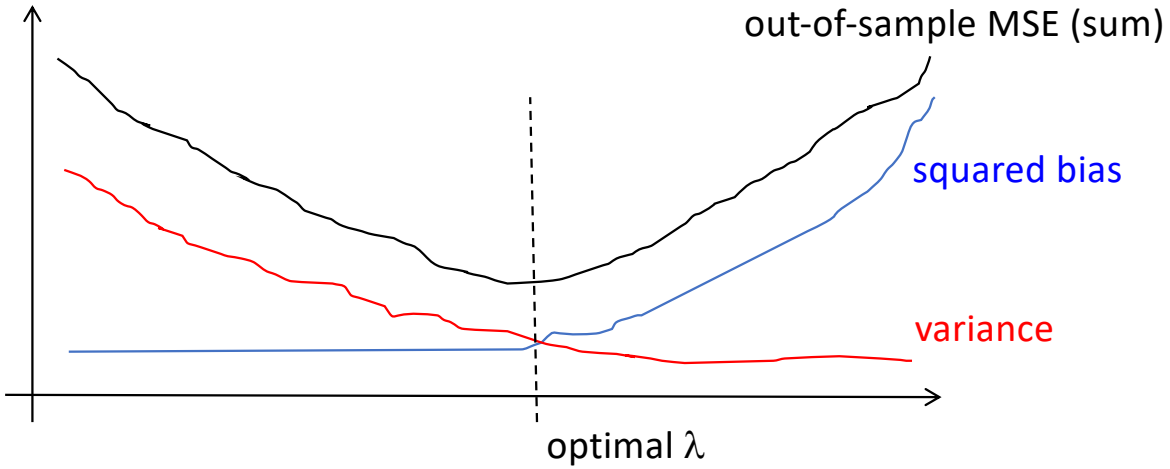
$$\sum_{j=1}^{p} | \beta_j | \leq s_\lambda$$

(maximal) VALUE OF THE SIZE CONSTRAINT (PENALTY PARAMETER) FIXED…

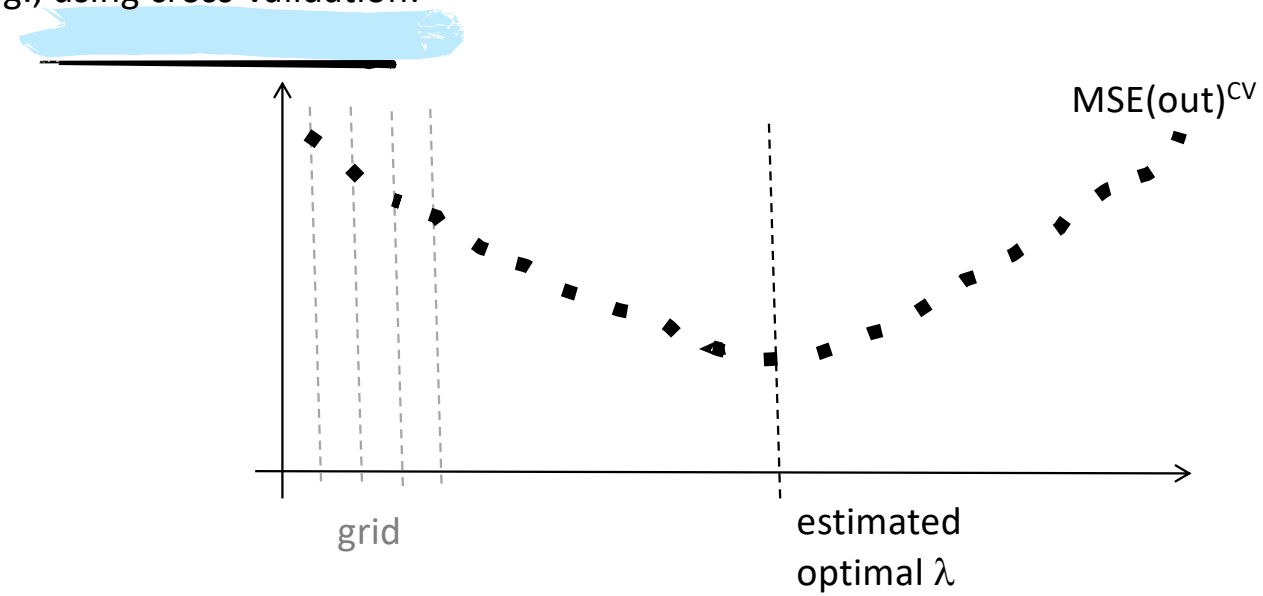
Extremely efficient algorithms exist to minimize the objective function with penalty and find the constrained solutions (convex optimization). LASSO is of course more computationally expensive that LS (with its close form solution), but not prohibitive also for very large p.

The critical part is ***selecting an appropriate λ (i.e., s_λ)***.

Ridge and LASSO can improve over LS because they reduce the sample variability of the coefficient estimators and emiliorate overfitting. But they create (if LS is unbiased) or increase a bias component. As λ (and thus the weight of the penalty vs the RSS term in the objective function) grows, the variance decreases but the bias increases – in the limit for infinite λ all coefficient estimates will be 0, with no variance! We want the "sweet spot" where the out-of-sample MSE is minimized:



In applications we don't know how the out-of-sample MSE varies as a function of λ. But we can estimate it on a grid of values, e.g., using cross-validation:



The selection of an appropriate λ by cross-validation substantially adds to the computational burden (we need to iterate the optimization algorithm many times) but it is essential for an effective use of Ridge and LASSO…

… and algorithms for running LASSO are very fast!
(implemented in standard statistical software)

Note: we have introduced the geometry of the constraints taking λ as fixed, then discussed the selection of λ.

One could imagine selecting (e.g., by cross-validation) also some aspects of the geometry of the constraints? For instance, L1 or L2 norm? A combination of norms?

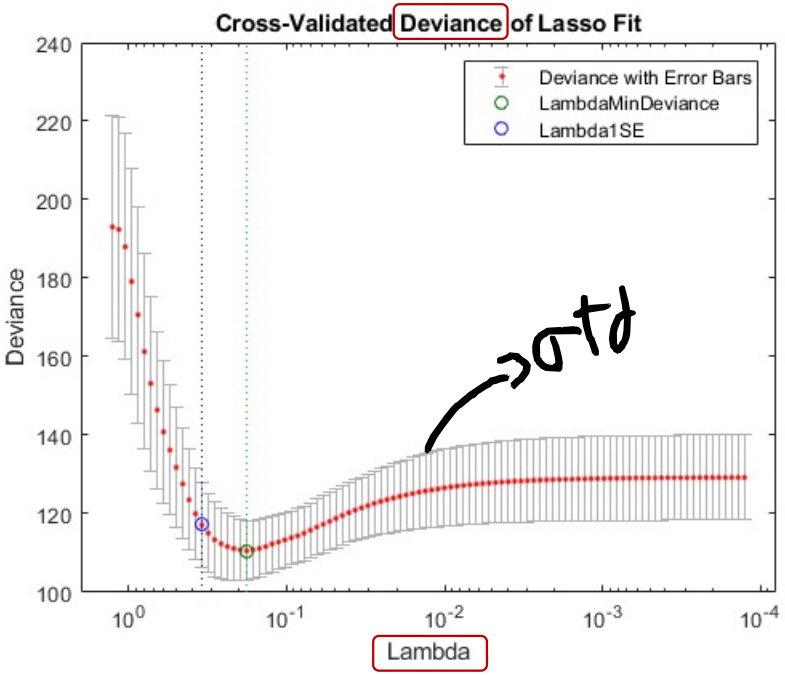
Elastic Nets – combine Ridge and LASSO penalization (not in ISLR).

Implementation for Linear (Gaussian) as well as Generalized Linear Models in the R package **GLMnet** (Ridge, LASSO, and Elastic Nets in between)

https://cran.r-project.org/web/packages/glmnet/glmnet.pdf
https://cran.r-project.org/web/packages/glmnet/index.html

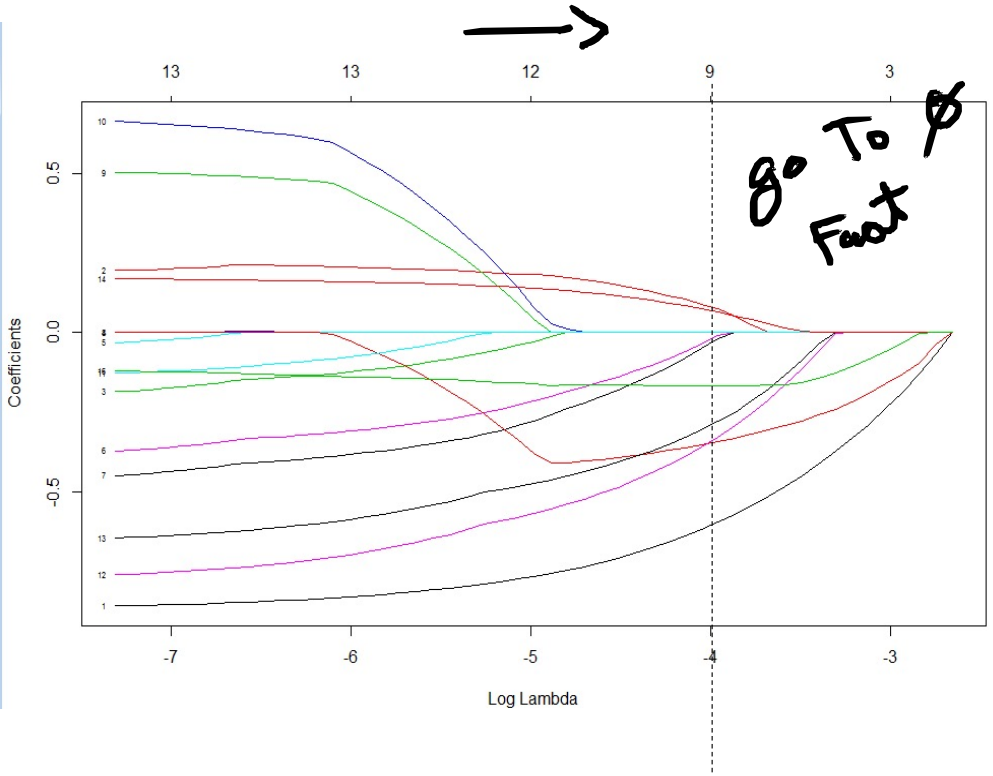
Choosing the right penalty parameter

or MSE



or Log Lambda

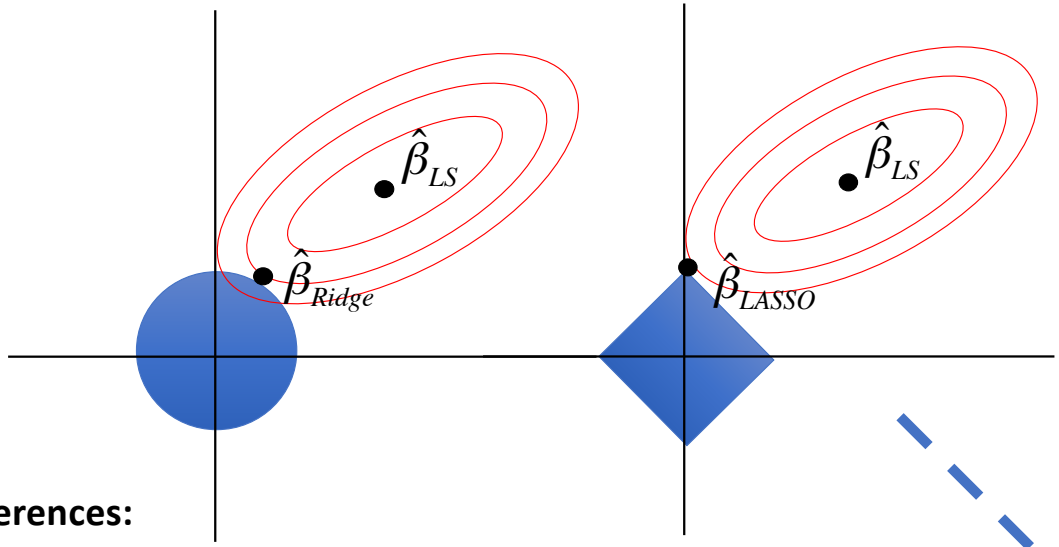
How coefficient estimates shrink with penalization



at a given level of penalization, how many and which coefficients are non-0?

SOME ADDITIONS: NON-CONVEXITY AND ISSUES WITH THE LASSO

FURTHER CONSIDERATIONS AND REFERENCES ABOUT PENALIZATIONS



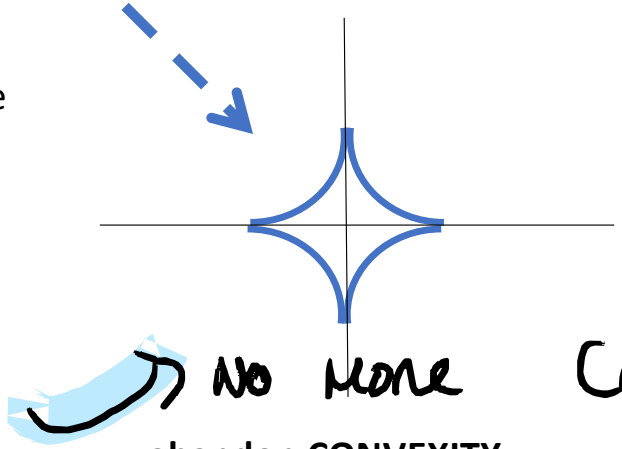
CONVEX constrained optimization
very efficient computational approaches.

Estimates are biased (even when the LS for the full model is not)… but **more stable**, and **sparse** (L1).

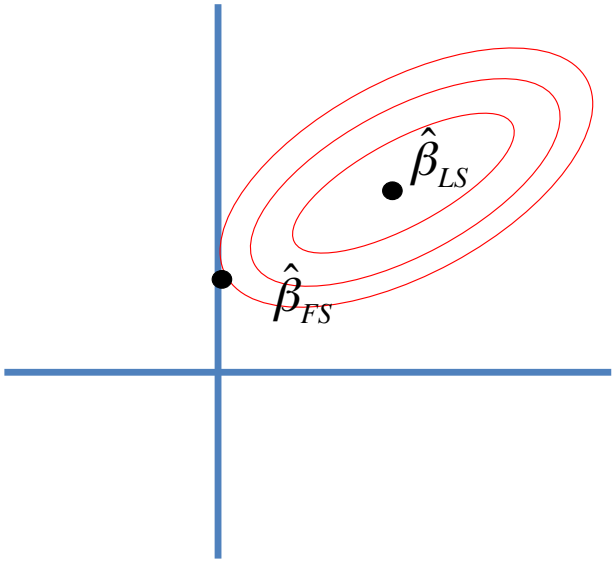
Very broad literature, lots of variants, including those to **incorporate group or order structure** for features.

Some references:

- Hastie T., Tibshirani R., Friedman J. (2009). Elements of Statistical learning 2nd ed. Springer.
- Tibshirani, R. (1996) Regression shrinkage and selection via the lasso. JRSS B, 58(1) 267-288.
- Zou H. Hastie T. (2005) Regularization and variable selection via the elastic net. JRSS B, 67(2) 301–320.
- Tibshirani R., Saunders M. (2005) Sparsity and smoothness via the fused lasso. JRSS B, 67(1) 91–108
- Yuan M., Lin Y. (2006) Model selection and estimation in regression with grouped variables. JRSS B, 68(1) 49–67.
- Fan J. Li R. (2001) Variable selection via nonconcave penalized likelihood and its oracle properties. JASA, 96(456) 1348-1360



… abandon CONVEXITY
reduced bias, but harder computational problem.



Traditional ("hard") feature selection: (most) **NON-CONVEX constrained optimization**

L0 norm counts non-0 coefficients, but size of each non-0 coefficient is unconstrained.

Much harder, previously not computationally viable; now *Mixed Integer Optimization*.

Some (non-traditional) references:

- Bertsimas D., King A., Mazumder R. (2016) Best subset selection via a modern optimization lens. AOS 44(2) 813-852.
- Kenney A., Chiaromonte F., Felici G. (2020) MIP-BOOST: Efficient and Effective L_0 Feature Selection for Linear Regression. JCGS.

$$\| \underline{Y} - \underline{X}\beta \|^2 = \min_{\beta \in R^p}$$

$$\sum_{j=1}^{p} Ind(\beta_j \neq 0) \leq c$$ size constraint

SIZE CONSTRAINT TUNED BY CROSS VALIDATION

… the traditional **Best Subset Selection** problem

Important remarks on the potential of MIP:

- Further integer constraints to capture structure, e.g., feature groups.
- Incorporate election of units/observations to robustify the fit; Insolia L., Kenney S., Chiaromonte F., Felici G. (2021) Simultaneous feature selection and outlier detection with optimality guarantees. Biometrics.

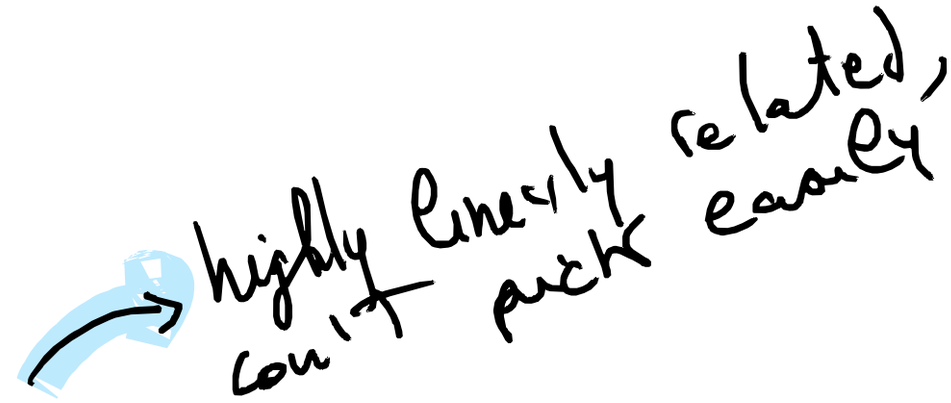
MIP-BOOST: Efficient and Effective L_0 Feature Selection for Linear Regression

Abstract: Recent advances in mathematical programming have made mixed integer optimization a competitive alternative to popular regularization methods for selecting features in regression problems. The approach exhibits unquestionable foundational appeal and versatility, but also poses important challenges. Here, we propose MIP-BOOST, a revision of standard mixed integer programming feature selection that reduces the computational burden of tuning the critical sparsity bound parameter and improves performance in the presence of feature collinearity and of signals that vary in nature and strength. The final outcome is a more efficient and effective L_0 feature selection method for applications of realistic size and complexity, grounded on rigorous cross-validation tuning and exact optimization of the associated mixed integer program. Computational viability and improved performance in realistic scenarios is achieved through three independent but synergistic proposals.

First: a novel bisection procedure designed specifically for tuning the sparsity bound, which significantly reduces the needed number of evaluations, cutting the computational burden of the MIP approach.

Second: a novel cross-validation scheme that exploits the structure of the MIP and of the simplex algorithm used for its solution to significantly reduce the computational effort of repeating calculations across folds (also warm starts and surrogate lower bounds).

Third: whitening, a pre-processing step that, by handling feature collinearities, can both reduce computational burden and improve solution quality. Whitening can be applied prior to any feature selection technique but it benefits MIP more than it does other approaches.



More important remarks:

- Even when introducing some Ridge for stabilization through an Elastic Net, collinearities can hinder the performance of LASSO (as of any other feature selection approach). Strong associations, especially between relevant and irrelevant features, can deteriorate the ability of any feature selection approach to identify the former.

Zhao P., Yu B. (2006) On model selection consistency of LASSO. Journal of Machine Learning Research, 7, 2541-2563.

Abstract: Sparsity or parsimony of statistical models is crucial for their proper interpretations, as in sciences and social sciences. Model selection is a commonly used method to find such models, but usually involves a computationally heavy combinatorial search. Lasso (Tibshirani, 1996) is now being used as a computationally feasible alternative to model selection. Therefore it is important to study Lasso for model selection purposes. In this paper, we prove that a single condition, which we call the Irrepresentable Condition, is almost necessary and sufficient for Lasso to select the true model both in the classical fixed *p* setting and in the large *p* setting as the sample size *n* gets large. Based on these results, sufficient conditions that are verifiable in practice are given to relate to previous works and help applications of Lasso for feature selection and sparse representation. This Irrepresentable Condition, which depends mainly on the covariance of the predictor variables, states that Lasso selects the true model consistently if and (almost) only if the predictors that are not in the true model are "irrepresentable" (in a sense to be clarified) by predictors that are in the true model. Furthermore, simulations are carried out to provide insights and understanding of this result.

- Addressing biases and post-selection inference issues for penalized methods such as the LASSO.

Belloni A., Chernozhukov V. (2013) Least squares after model selection in high-dimensional sparse models. Bernoulli, 19(2), 521 – 547.

→ reduces the bias

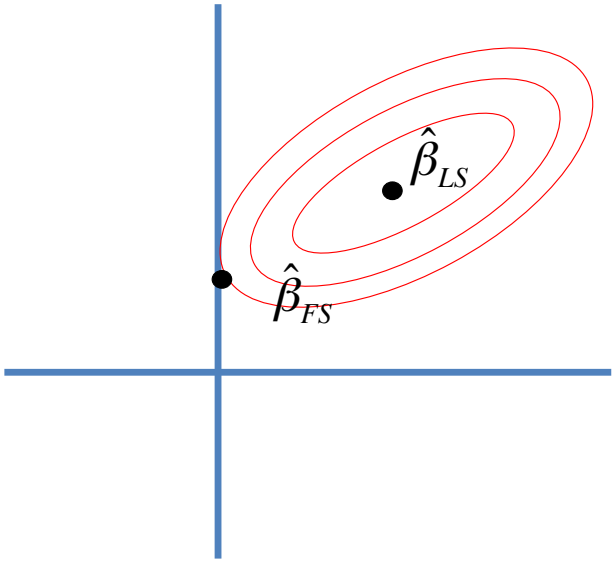
Abstract: In this article we study post-model selection estimators that apply ordinary least squares (OLS) to the model selected by first-step penalized estimators, typically Lasso. It is well known that Lasso can estimate the nonparametric regression function at nearly the oracle rate, and is thus hard to improve upon. We show that the OLS post-Lasso estimator performs at least as well as Lasso in terms of the rate of convergence, and has the advantage of a smaller bias. Remarkably, this performance occurs even if the Lasso-based model selection "fails" in the sense of missing some components of the "true" regression model. By the "true" model, we mean the best s-dimensional approximation to the nonparametric regression function chosen by the oracle. Furthermore, OLS post-Lasso estimator can perform strictly better than Lasso, in the sense of a strictly faster rate of convergence, if the Lasso-based model selection correctly includes all components of the "true" model as a subset and also achieves sufficient sparsity. In the extreme case, when Lasso perfectly selects the "true" model, the OLS post-Lasso estimator becomes the oracle estimator. An important ingredient in our analysis is a new sparsity bound on the dimension of the model selected by Lasso, which guarantees that this dimension is at most of the same order as the dimension of the "true" model. Our rate results are nonasymptotic and hold in both parametric and nonparametric models. Moreover, our analysis is not limited to the Lasso estimator acting as a selector in the first step, but also applies to any other estimator, for example, various forms of thresholded Lasso, with good rates and good sparsity properties. Our analysis covers both traditional thresholding and a new practical, data-driven thresholding scheme that induces additional sparsity subject to maintaining a certain goodness of fit. The latter scheme has theoretical guarantees similar to those of Lasso or OLS post-Lasso, but it dominates those procedures as well as traditional thresholding in a wide variety of experiments.

Outline: (Linear) Models in Large Feature Spaces
Traditional Feature Selection

(F. Chiaromonte)

Introduction to Statistical Learning

Chapter 6 Section 1, Lab 1



Traditional ("hard") feature selection: (most) **NON-CONVEX constrained optimization**

L0 norm counts non-0 coefficients, but size of each non-0 coefficient is unconstrained.

Much harder, previously not computationally viable; now *Mixed Integer Optimization*.

Some (non-traditional) references:

- Bertsimas D., King A., Mazumder R. (2016) Best subset selection via a modern optimization lens. AOS 44(2) 813-852.
- Kenney A., Chiaromonte F., Felici G. (2020) MIP-BOOST: Efficient and Effective L_0 Feature Selection for Linear Regression. JCGS.

$$\| \underline{Y} - \underline{X}\beta \|^2 = \min_{\beta \in R^p}$$

$$\sum_{j=1}^{p} Ind(\beta_j \neq 0) \leq c$$

size constraint

→ # of coefficients

SIZE CONSTRAINT TUNED BY CROSS VALIDATION

… the traditional **Best Subset Selection** problem

Important remarks on the potential of MIP:

- Further integer constraints to capture structure, e.g., feature groups.
- Incorporate election of units/observations to robustify the fit; Insolia L., Kenney S., Chiaromonte F., Felici G. (2021) Simultaneous feature selection and outlier detection with optimality guarantees. Biometrics.

Back to traditional statistical approaches:

- **Feature (Subset) Selection**
  - Best Subset Selection
  - Step-wise (sequential) Selection

  Followed by LS fit of the (smaller) model comprising the selected features.

- **Dimension Reduction**
  - Principal Components (unsupervised reduction)
  - Sufficient Dimension Reduction (supervised reduction)

  Followed by LS fit of the (smaller) model comprising the selected linear combinations

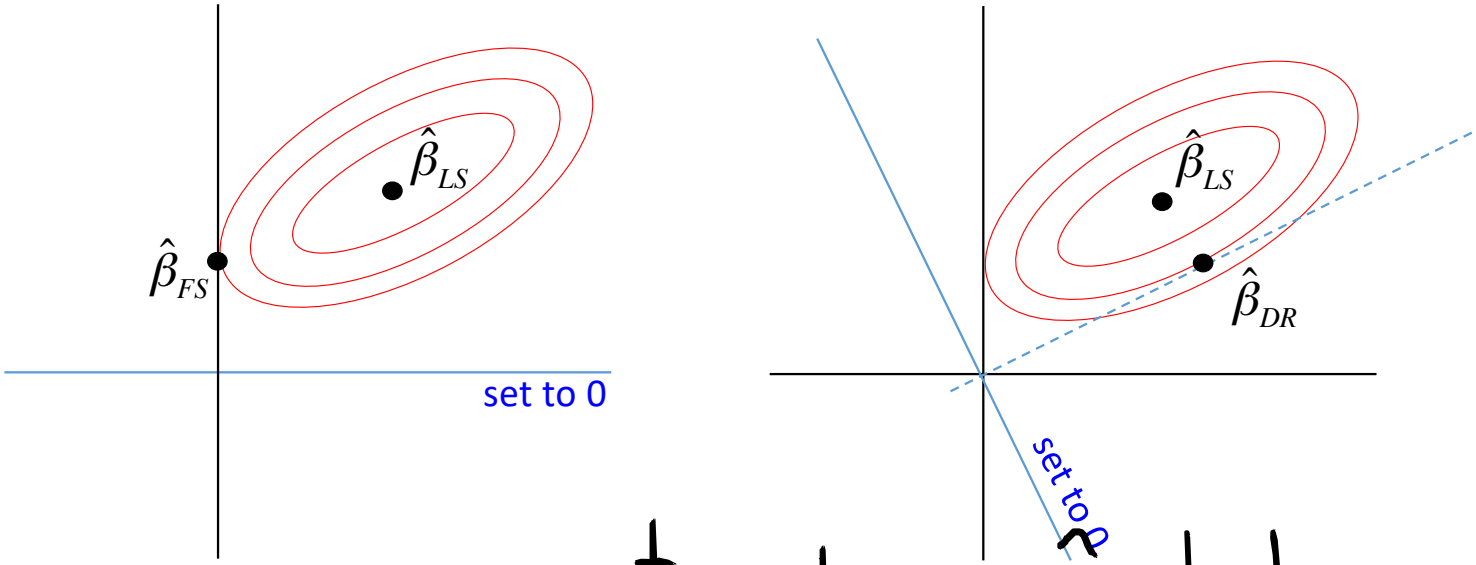
Also these can be though of in the framework of constrained LS: *Linear constraints* to force β in a coordinate space, or a generic linear subspace, of R^p. ***BUT WE NEED AN ADDITIONAL "INGREDIENT"!***

==========

Dimension Reduction: ISLR also describes Partial Least Squares. But does not describe Sufficient Dimension Reduction techniques.

Vor H in a basis of space

Cartoons with p=2 and c=1 (d=1)



move to optimal β, slowly

$$\| \underline{Y} - \underline{X}\beta \|^2 = \min_{\beta \in R^p}$$

$$H_{FS}\beta = 0_{(p-c)}$$

linear constraints

$$\| \underline{Y} - \underline{X}\beta \|^2 = \min_{\beta \in R^p}$$

$$V_{DR}\beta = 0_{(p-d)}$$

THE (maximal) **COUNT** OF FEATURES *c* OR THE **DIMENSION** *d* IS FIXED… SO IS THE **(O.N.) BASIS** OF THE FEATURE SPACE FROM WHICH WE FORM THE (*p*-*c*) or (*p*-*d*) CONSTRAINTS.

$X_{(1)}$ X_2 ... X_p

$\{e_1, e_2, \ldots e_p\}$

$X_{(1)} \to X_{(2)} \to$ ... $X_{(p)}$ Nested seq. of subsets

Feature Selection: How do we produce a pxp matrix H expressing an O.N. basis in the feature space?

(we then focus on its last (*p-c*) rows to create the linear constraints with H_{FS})

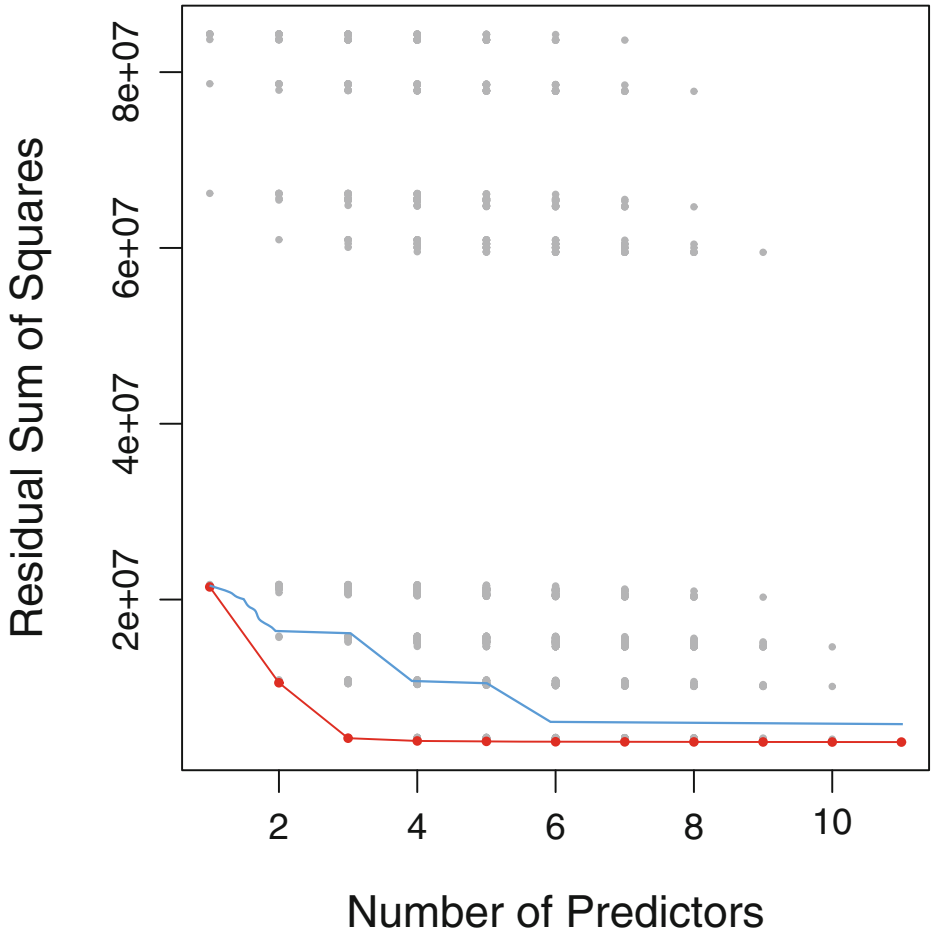
Feature (subset) Selection:

Make a sequence of nested subsets adding one feature at a time, i.e., make an ordering of the features.

This corresponds to a permutation of the elements of the Canonical O.N. basis $\{e_1, \ldots e_p\}$, and provides

H (a permutation matrix)

Rigorous for feature selection implemented through **stepwise (sequential) methods**, not for **best subset methods** which find the best subset of each size – in principle such subsets may not be nested (but the intuition still works).

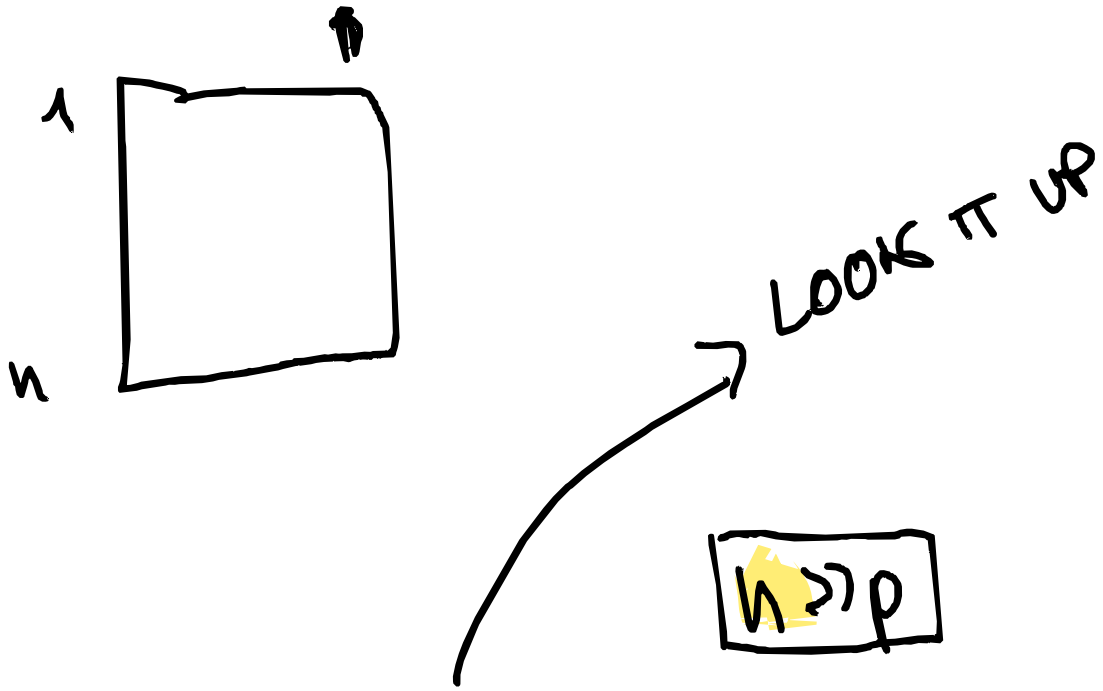


The RSS of 2^p possible models (subsets of features).

For each size c, there are $\binom{p}{c}$ possible models.

On the lower boundary (red) are the best models of each size; best subsets. May or may not be a nested sequence.

The blue path represents a nested sequence of "good" models obtained stepwise. May depart from the lower boundary.



Algorithm 6.1 *Best subset selection*

1. Let $\mathcal{M}_0$ denote the *null model*, which contains no predictors. This model simply predicts the sample mean for each observation.
2. For $k = 1, 2, \ldots p$:
   (a) Fit all $\binom{p}{k}$ models that contain exactly k predictors.
   (b) Pick the best among these $\binom{p}{k}$ models, and call it $\mathcal{M}_k$. Here *best* is defined as having the smallest RSS, or equivalently largest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

Best Subset approach requires fitting ALL possible models ("brute" enumeration).

Algorithm 6.2 *Forward stepwise selection*

1. Let $\mathcal{M}_0$ denote the *null* model, which contains no predictors.
2. For $k = 0, \ldots, p-1$:
   (a) Consider all $p-k$ models that augment the predictors in $\mathcal{M}_k$ with one additional predictor.
   (b) Choose the *best* among these $p-k$ models, and call it $\mathcal{M}_{k+1}$. Here *best* is defined as having smallest RSS or highest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

Stepwise (sequential) approaches do not; they explore fewer models. Computationally leaner.

Hybrid stepwise approaches mix forward and backward progression.

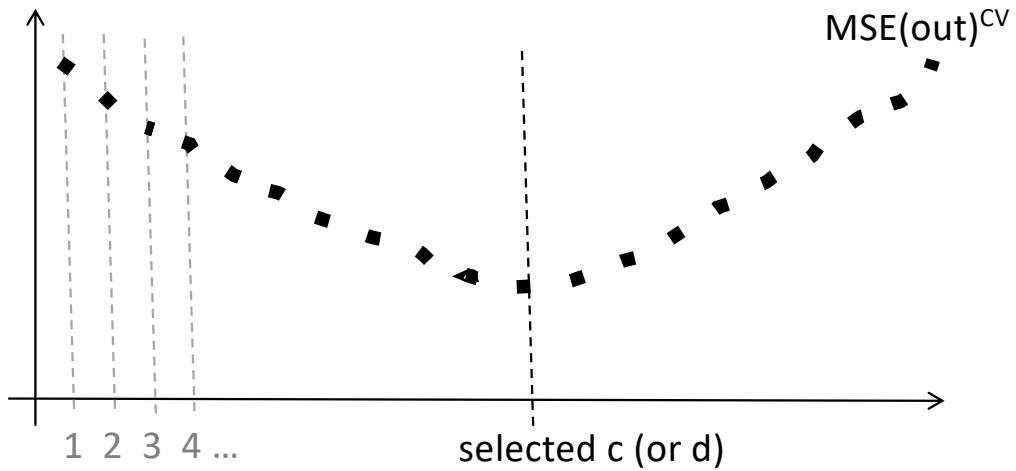
Algorithm 6.3 *Backward stepwise selection*

1. Let $\mathcal{M}_p$ denote the *full* model, which contains all p predictors.
2. For $k = p, p-1, \ldots, 1$:
   (a) Consider all k models that contain all but one of the predictors in $\mathcal{M}_k$, for a total of $k-1$ predictors.
   (b) Choose the *best* among these k models, and call it $\mathcal{M}_{k-1}$. Here *best* is defined as having smallest RSS or highest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

(Step 3 in each algorithm; next…)

How do we select the (maximal) count c, or dimension d, along an ordered sequence of features or linear combinations?

Of course, once H (or V) are fixed we could perform the selection estimating the out-of-sample MSE associated to each value of *c* or *d*, e.g., by cross-validation (same as for the penalty parameter in Ridge or LASSO).



Historically though other (non-computation based) approaches and criteria have been used.

CRITERIA FOR FEATURE (SUBSET) SELECTION

→ OUT OF SAMPLE PERFORMANCE

The number of features c (= **d** in formulae below) is selected optimizing a criterion that approximates an estimate of the out-of-sample performance – but indirectly, not directly as in cross-validation; no intensive computation required. For instance:

$$\text{Adjusted } R^2 = 1 - \frac{\text{RSS}/(n-d-1)}{\text{TSS}/(n-1)}.$$

Adjusted R-squared provides the simplest solution, each sum of squares is divided by its degrees of freedom.

$$R^2 = 1 - \frac{RSS}{TSS}$$

→ Residual Sum Squares

→ Total sum squares

- $C_p = \frac{1}{n}\left(\text{RSS} + 2d\hat{\sigma}^2\right)$

↓ UNBIASED

Mallow's C_p formula in the book, proportional to original formula. If $\hat{\sigma}^2$ is an unbiased estimate of the error variance, C_p is an unbiased estimate of the out-of-sample MSE (can take the s^2 estimate from the full model, least likely to carry bias.

Strike a balance between RSS (error on the training data) and d (size, complexity of the model)

Akaike and Bayes Information Criteria can be used for model selection whenever models are fitted by Maximum Likelihood.

For linear models with additive Gaussian errors their expressions become:

→ Require normality of error

$$\mathrm{AIC} = \frac{1}{n\hat{\sigma}^2}\left(\mathrm{RSS} + 2d\hat{\sigma}^2\right)$$

Behaves exactly like C_p here!

$$\mathrm{BIC} = \frac{1}{n}\left(\mathrm{RSS} + \log(n)d\hat{\sigma}^2\right)$$

The weight of the adjustment uses log(n), depending on the sample size, instead of 2. For n>7 BIC has a steeper penalty than AIC (C_p); it favors smaller models.

General equations for Information Criteria:

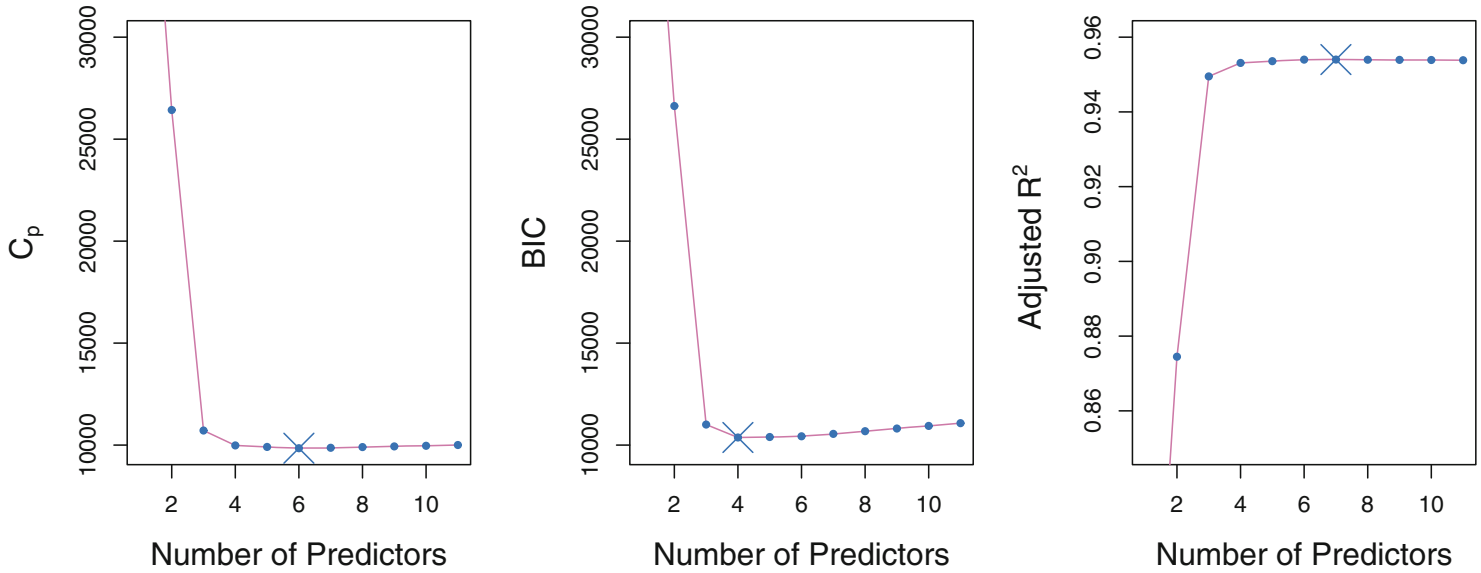
$$AIC = -2\log L(\hat{\vartheta}_{ML}) + 2d$$

$$BIC = -2\log L(\hat{\vartheta}_{ML}) + \log(n)d$$

$L(\hat{\vartheta}_{ML})$ = optimal likelihood value on MLE of parameter vector

d = # of free parameters being estimated

HOW DO THE TRADITIONAL CRITERIA LOOK IN PRACTICE?



This behavior is rather common; even though the criteria are NOT monotone and ought to have a minimum/maximum, in many applications they do not clearly discriminate unambiguously the number of features (the model) to be used – rather flat ranges.

Would you pick 3 instead of the “technical” optima, which btw differ across criteria?

Outline: (Linear) Models in Large Feature Spaces
Supervised Dimension Reduction

(F. Chiaromonte)

Introduction to Statistical Learning
PCR: Chapter 6 Section 3, Lab 3
LDA: Chapter 4 Section 4, Lab (part 3)

SDR: References throughout the slides

INFORM THE REDUCTION

Back to more traditional statistical approaches:

COMPOSITION OF BEAT COMPONENTS

- **Feature (Subset) Selection**
  - Best Subset Selection
  - Step-wise Selection

  Followed by LS fit of the (smaller) model comprising the selected features.

- **Dimension Reduction**
  - Principal Components (unsupervised reduction)
  - Sufficient Dimension Reduction (supervised reduction)

  Followed by LS fit of the (smaller) model comprising the selected linear combinations

  Also, in the case of a binary or categorical response (Generalized Linear Models)
  - Linear Discriminant Analysis (seen as supervised reduction)

  Followed by ML fit of the (smaller) model comprising the selected linear combinations

Also these can be though of in the framework of constrained LS: *Linear constraints* to force β in a coordinate space, or a generic linear subspace, of R^p. ***BUT WE NEED AN ADDITIONAL "INGREDIENT"!***

==========

Dimension Reduction: ISLR also describes Partial Least Squares. But does not describe Sufficient Dimension Reduction techniques.

Dimension Reduction: ***How do we produce a pxp matrix V expressing an O.N. basis in the feature space?***

(we then focus on its last (*p*-*d*) rows to create the linear constraints V_{DR})

One very effective approach: take the O.N. basis provided by the Eigen decomposition of an appropriate positive definite matrix. The eigenvectors express the directions, and the corresponding eigenvalues their ordering.

This corresponds to a rotation of the elements of the Canonical O.N. basis $\{e_1, \ldots e_p\}$, and provides

V – a rotation matrix

TAKE eigen decomposition ← → graph from previous slides

Other approaches to dimension reduction exist, but some of the most popular (e.g. **Principal Components Analysis**, unsupervised; **Sliced Inverse Regression** supervised) use Eigen decompositions.

SIR

POSITIVE DEFINITE

PRINCIPAL COMPONENTS (see also ISLR Chapter 10, Sections 1,2)

Units\Features	X_1	X_2	…	X_p
Unit 1	x_{11}	x_{12}		x_{1p}
Unit 2	x_{21}	x_{22}		x_{2p}
. . .				
Unit n	x_{n1}	x_{n2}		x_{np}

p features measured on n units:

- An n x p data matrix X
- A data cloud of n points in $\mathbf{R}^p$.

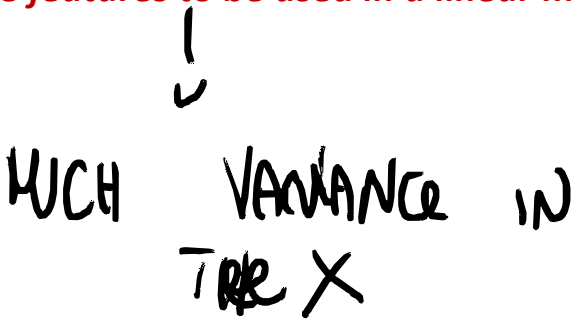
Location is irrelevant; assume each feature is centered, mean 0.

$\bar{X} = 0_p$ mean vector in R^p; cloud is centered/located at origin

$S \propto X'X$ (sample) p x p variance/covariance matrix

Create orthogonal directions in $\mathbf{R}^p$ (and corresponding linear combinations) ranked by variability of the data cloud. In many applications (but not all) these are the most informative, capturing structure in the data.

Here, we think of them as generating composite features to be used in a linear model – but we do not consider Y in creating them!



Take the Eigen decomposition of S:

$$S = \sum_{m=1}^{p} \lambda_m \phi_m \phi_m^T$$

$$\lambda_1 \geq \ldots \geq \lambda_p \geq 0 \quad \text{(eigenvalues)}$$

eigenvalues are variances along the directions identified by eigenvectors; in non-increasing order.

$$\| \phi_m \| = \phi_m^T \phi_m = 1 \,,\, \phi_m^T \phi_k = 0 \; m,k = 1,2 \ldots p \quad \text{(eigenvectors)}$$

For any given dimension *m*, identify the linear subspaces closest to the data:

$$\| P_{Span(\phi_1 \ldots \phi_p)} X \|^2 = \max$$

m-dimensional representation most likely to be useful in capturing structure, most informative (not always!)

Here, the reduced feature space we use to formulate a linear model once we chose m*=d.

$V = (\phi_1,\ \phi_2 \dots \phi_p) \quad (p \times p)$ **ROTATION MATRIX** provided by the eigenvectors of S

LOADINGS: coefficients expressing the m-th component in terms of the original features

$\phi_m^T = (\phi_{m1},\ \phi_{m2} \dots \phi_{mp})$

coordinates of each element (m-th) of the new rotated basis in terms of the original Canonical basis.

SCORES: values of the m-th component on the n units

$z_{im} = \phi_{m1} x_{i1} + \phi_{m2} x_{i2} \dots + \phi_{mp} x_{ip} \quad , \quad i = 1, 2 \dots n$

coordinates of the n data points in terms of each element (m-th) of the new rotated basis. These express the values of the new composite features on each unit.

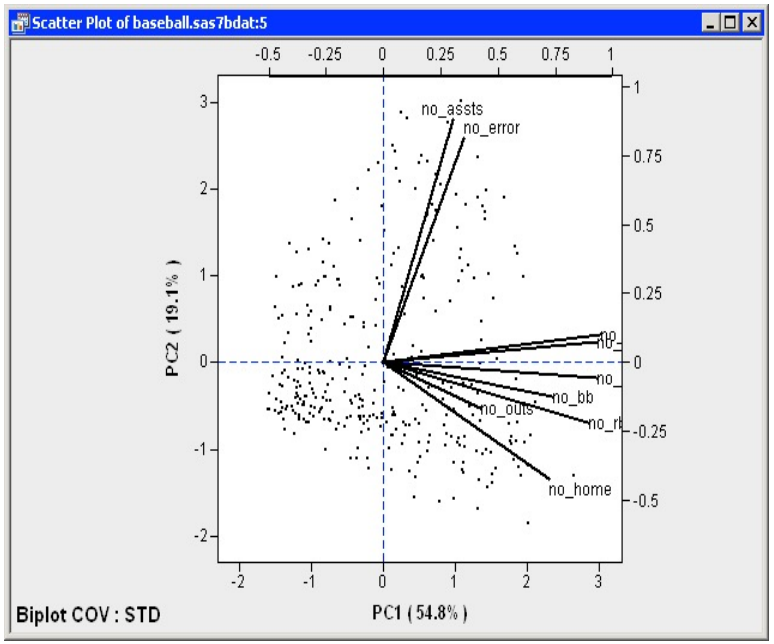
PERCENTAGE OF VARIANCE EXPLAINED (PVE) by the m-th component

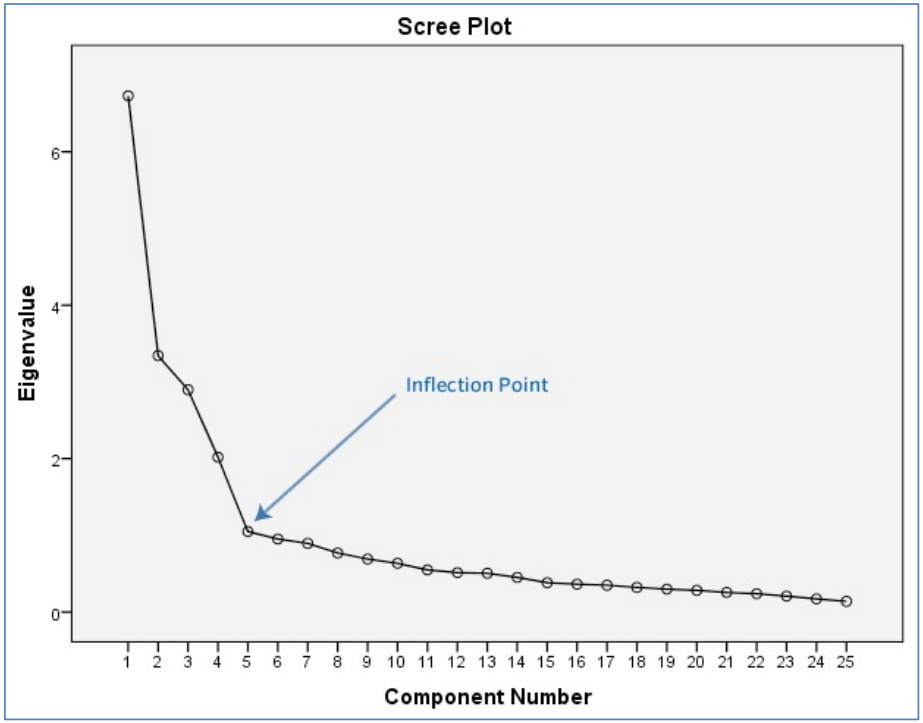
$$PVM_m = \frac{\lambda_m}{\sum_{k=1}^{p} \lambda_k} = \frac{\mathrm{var}(\phi_m^T X)}{\sum_{k=1}^{p} \mathrm{var}(\phi_k^T X)} \cdot 100$$

Also, **CUMULATIVE PVE** up to the m-th component

$$CPVM_m = \sum_{j=1}^{m} PVM_j = \frac{\sum_{j=1}^{m} \lambda_j}{\sum_{k=1}^{p} \lambda_k}$$

Biplot: shows the scores for two specified components (e.g., 1st and 2nd; projection of the points on the first PCA plane) along with the loadings represented by arrows.





Scree plot: Show the variances (eigenvalues) or PVEs in non-increasing order (graphical diagnostic to aid in the selection of m*=d)

SLICED INVERSE REGRESSION (SIR):

A method for *supervised* (sufficient) dimension reduction.

The main limitation of PC regression is that the composite features are identified as to capture variability in the feature space; the response Y plays no role.

Directions of maximal variability are not necessarily directions of maximal explanatory power with respect to Y!

Can we perform ***supervised*** dimension reduction?

… Yes! ~30 years old field of statistical research named ***Sufficient Dimension Reduction*** (SDR)

Some references:

- Li, K. C. (1991) Sliced inverse regression for dimension reduction (with discussion). Journal of the American Statistical Association, 86, 316–327.
- Cook, R. D. (1998) Regression Graphics: Ideas for Studying Regressions through Graphics. New York: Wiley.
- Ma, Y. and Zhu, L. (2013) A review on dimension reduction. International Statistical Review, 81, 134–150.

R package:
https://cran.r-project.org/web/packages/dr/index.html
https://cran.r-project.org/web/packages/dr/vignettes/overview.pdf

$$Y = m(A^T X; \varepsilon)$$

I want ↳ model Free → NON PARAMETRIC METHODS

high dimensional feature vector (p)

X_1

$X_{p=tens}$

$X_{hundreds}$

$X_{thousands\ (etc\ldots)}$

Y

Response (any nature)

Small composite feature vector (d=1,2,3)

$\tilde{X}_1 = \alpha_1^T X$

$\tilde{X}_d = \alpha_d^T X$

Y

(1) Sufficient Dimension Reduction (SDR): inference on the ~~Central Subspace~~ (CS), which retains information on the dependence between Y and X

$S_{Y|X}$ smallest s.t. $Y \perp X \mid P_{S_{Y|X}} X$

$S_{Y|X} = Span(A_{p\times d}),\ \ Y \perp X \mid A^T X$ basis version

$d = \dim(S_{Y|X})$ structural dimension

↳ smallest subspace

(2) Modeling: Y as a function of the composite feature vector and random error

$Y = m(\tilde{X};\varepsilon) \quad e.g. \quad Y = \beta_o + \beta^T \tilde{X} + \varepsilon$

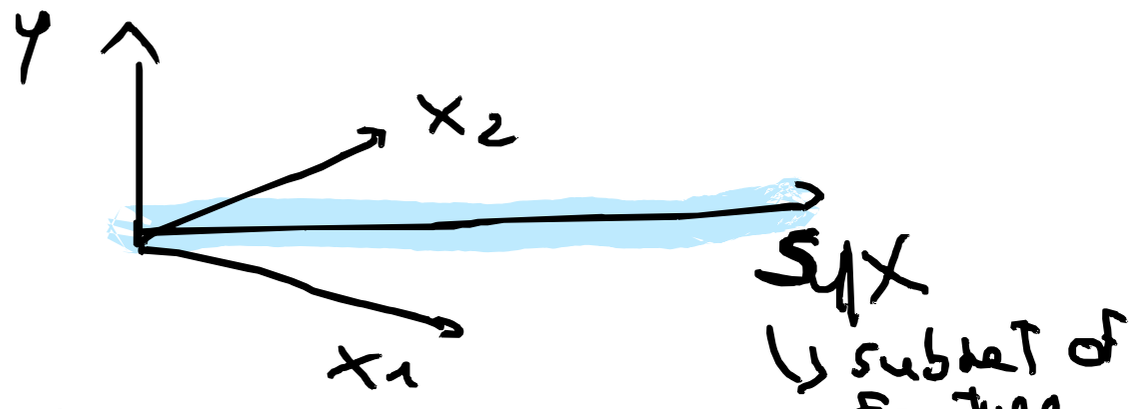
need methods for inference on the CS (on the spanning matrix A) and on d!

↳ sufficiently independent from the rest of data

(Y, X)
$\|$
$p=2$

$(Y|X)$

Regression



$Y \perp X \mid P_S X$

↳ suff dimension red. subspace

min SUBSPACE

(X|Y) inverse regression

SIR: simplest SDR method, based on an Eigen decomposition

- Consider the regression of X on Y (this is the *inverse regression*)
- *Slice* the range of Y in h=1,2… H slices (if continuous, otherwise use the classes) and form the sample covariance matrix of E(X|Y)

↳ equal content to slice

$$\bar{X}_h = \frac{1}{n_h}\sum_{y_i \in h} X_i \quad h = 1,2...H$$ (we always assume overall mean vector = 0)

↳ max H-1 directions

$$y_i = \beta_0 + \beta_i' x_i + \varepsilon_i \Rightarrow \hat{\beta}$$

$$M = \frac{n_h}{n}\sum_{h=1...H} \bar{X}_h \bar{X}_h^T$$

- Take the Eigen decomposition of

$$S^{-1}M = \sum_{m=1}^{p} \lambda_m \phi_m \phi_m^T$$

Rescaling by S^{-1} very important! Weighing directions in the feature space

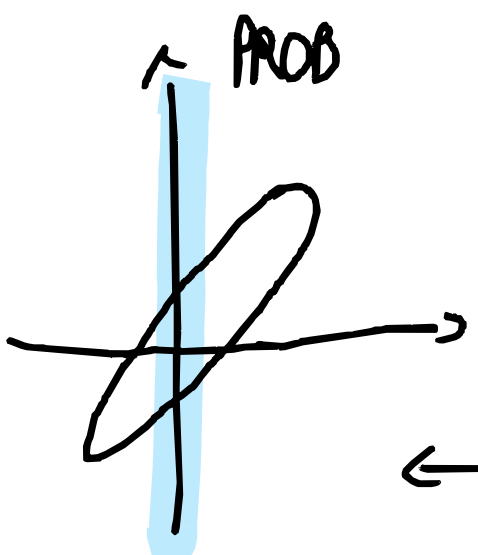
(also, cannot apply as is if n < p, rank(S) cannot exceed n-1)

$\lambda_1 \geq ... \geq \lambda_p \geq 0$ (eigenvalues) Note: only H-1 of the eigenvalues can be > 0, rank(M) cannot exceed H-1

$\| \phi_m \| = \phi_m^T \phi_m = 1, \phi_m^T \phi_k = 0 \; m,k = 1,2...p$ (eigenvectors)

- Under conditions on the joint distribution of X (linearity), for any given dimension *m*, we identify the linear subspaces with highest explanatory power (directions within the CS).
- If we know *d*, we estimate the whole CS.

$\hat{\beta}_{OLS} = (X^T X)^{-1} X^T y$ here $M = S^{-1}M$



LINEARITY

You matter more where it changes less

Remarks:

- Can make bi-plots and scree plots exactly as for PCA.

Also, If d=1,2

- Can plot Y vs the composite features to visualize the data and create a satisfactory model.
- Can use non-parametric regression fits.

- Like in feature selection (LASSO and related techniques), large literature. Some relevant developments:
  - Chiaromonte et al. (2002). *Sufficient dimension reduction in regressions with categorical predictors*. Annals of Statistics.
  - Li et al. (2010). *Groupwise dimension reduction*. JASA.
  - Guo et al. (2014). *Groupwise dimension reduction via envelope methods*. JASA.
  - Liu et al. (2017). *Structured Ordinary Least Squares: A Sufficient Dimension Reduction approach for regressions with partitioned predictors and heterogeneous units*. Biometrics.

Dimension Reduction*: *how do we select the relevant (structural) dimension d?

→ Find d to assure conditional independence

- Based on diagnostic statistics and their plots. For methods based on an Eigen decomposition, statistics and plots can be derived from eigenvalues.

- Using tests. For methods based on an Eigen decomposition, one can test how many tail eigenvalues are significantly > 0. More generally, one can use a sequence of tests, e.g., for each q=0,1... (p-1) test whether Ho: d=q vs Ha: d>q.

- Minimizing BIC or BIC-like criteria, if and when we introduce distributional assumptions and can use likelihoods.

- Assessing stability through the Bootstrap; rather different approach; see Ye and Weiss (2003) JASA:

Using the Bootstrap to Select One of a New Class of Dimension Reduction Methods

→ Recalculate with BOOTSTRAP

Abstract: Dimension reduction in a regression analysis of response *y* given a *p*-dimensional vector of predictors **x** reduces the dimension of **x** by replacing it with a lower-dimensional linear combination β'**x** of the **x**'s without specifying a parametric model and without loss of information about the conditional distribution of *y* given **x**. We unify three existing methods, sliced inverse regression (SIR), sliced average variance estimate (SAVE), and principal Hessian directions (pHd), into a larger class of methods. Each method estimates a particular *candidate matrix*, essentially a matrix of parameters. We introduce broad classes of dimension reduction *candidate matrices*, and we distinguish estimators of the matrices from the matrices themselves. Given these classes of methods and several ways to estimate any matrix, we now have the problem of selecting a particular matrix and estimation method. We propose bootstrap methodology to select among candidate matrices, estimators and dimension, and in particular we investigate linear combinations of different methods.

Not monotone

↓ Stability

LINEAR DISCRIMINANT ANALYSIS (LDA)

As a method for supervised dimension reduction when Y is categorical, again based on an Eigen decomposition.

Works just like SIR, but uses a *different rescaling*.

- Consider predicting Y based on X looking at how X varies in each of the Y classes (inverse approach)
- Say Y has H levels h=1,2… H. Without having to slice, form the sample covariance matrix of E(X|Y)

$$\bar{X}_h = \frac{1}{n_h} \sum_{y_i \in h} X_i \quad h = 1,2 \ldots H$$

(we always assume overall mean vector = 0)

between the means

$$M = \frac{n_h}{n} \sum_{h=1 \ldots H} \bar{X}_h \bar{X}_h^T$$

This is the between sample var/cov matrix S_B

- Take the Eigen decomposition of

S_W^{-1} ~~S^{-1}~~ $M = \sum_{m=1}^{p} \lambda_m \phi_m \phi_m^T$

In rescaling, instead of S (overall sample var/cov matrix) use S_W (within sample var/cov matrix). A different way of weighing directions in the feature space! (cannot apply as is if rank(S_W) < p)

$$\lambda_1 \geq \ldots \geq \lambda_p \geq 0 \quad \text{(eigenvalues)}$$

Note: only H-1 of the eigenvalues can be > 0, rank(M) cannot exceed H-1

$$\| \phi_m \| = \phi_m^T \phi_m = 1 \,,\, \phi_m^T \phi_k = 0 \; m,k = 1,2 \ldots p \quad \text{(eigenvectors)}$$

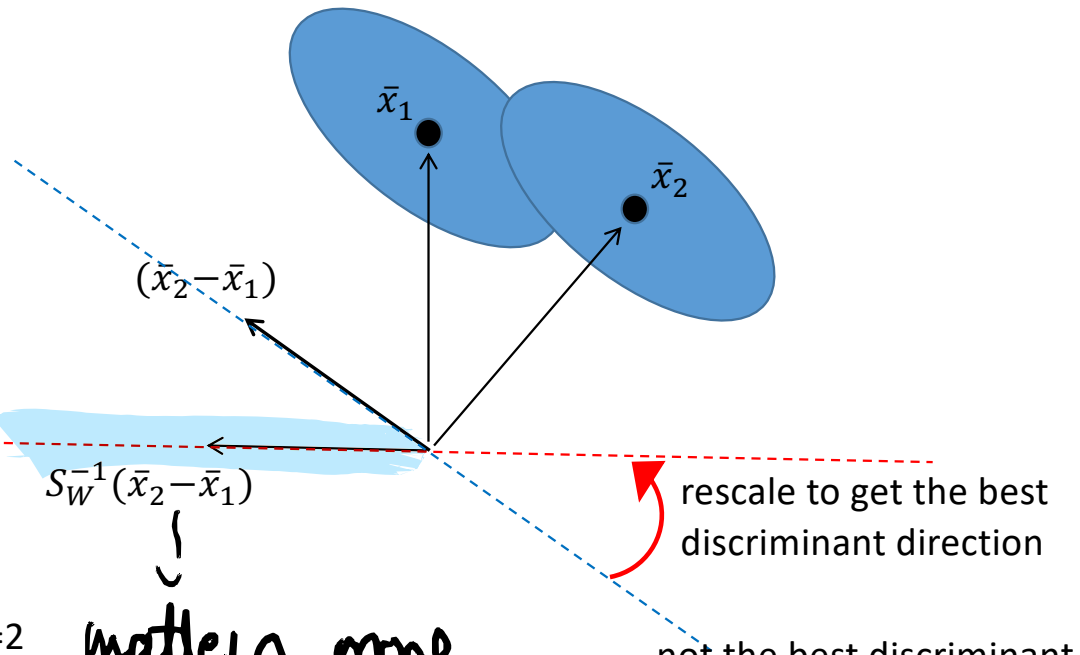
$S_W = S_B$ → when some eigenvalues & eigenvectors

$S = S_W + S_B$

Decomposition of the var/cov matrix: Within and Between variation

$$S = S_B + S_W \propto \sum_{k=1\ldots H} \sum_{i:y_i \in k} (x_i - \bar{x}_k)(x_i - \bar{x}_k)' + \sum_{k=1\ldots H} n_k (\bar{x}_k - \bar{x}\ )(\bar{x}_k - \bar{x}\ )'$$

Use the Within variation structure to rescale



LDA discriminant function (classify to highest)

$$\hat{\delta}_k(x) = (S_W^{-1}\bar{x}_k)'x - \frac{1}{2}\bar{x}_k{}'S_W^{-1}\bar{x}_k + log\frac{n_k}{n}$$

$$\hat{\delta}_2(x) - \hat{\delta}_1(x) = (S_W^{-1}(\bar{x}_2 - \bar{x}_1))'x + const$$

for simplicity H=2

difficult to invertible → where i can

matters more where you change less

The remapping is CRITICAL

$S^{-1} = S^T$ if orthogonal

→ log (domestic) makes them more readable

→ Box - Cox algorithm tries to make the data as Gaussian ~ Elliptical as possible

↓

TRY ON DATA

IF $n \to \infty$, also $p \to \infty$

Outline: (Linear) Models in Ultra-Large Feature Spaces

Feature Screening

(F. Chiaromonte)

- A key reference: Fan J., Lv J. (2008) Sure Independence Screening for ultrahigh dimensional feature spaces. JRSS-B, 70(5) 849-911.
- Additional references: throughout the slides.

Stochastic mechanism:

Y, ε $r.vbls\ in\ R^1$ $X\ r.vct\ in\ R^p$

$E(X) = 0 \quad Cov(X) = \Sigma_X$

$\varepsilon\ indep\ X \quad \varepsilon \sim N(0, \sigma^2)$

GAUSSIAN NOISE

Independent sampling:

$Y_{nx1} = X_{nxp}\beta_{px1} + \varepsilon_{nx1}$

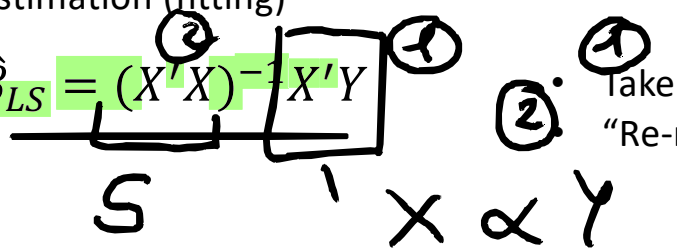
$S = X'X$

variance

covariance

Estimation (fitting)

$\hat{\beta}_{LS} = (X'X)^{-1}X'Y$



Take the marginal (linear) associations between Y and each of the X's; X'Y

"Re-map" it based on the (linear) associations among the X's; X'X ~ S_x .

How would you rank the X's and chose the d << p most likely to have an effect on Y?

On the basis of the effects as estimated by LS; rank the coordinates of the LS vector and pick the largest d.

Does this work?

Yes, if the re-mapping is effective; the ranking comprises two ingredients, marginal associations with Y and re-mapping based on associations among the X's. No problem at all if "no re-mapping" i.e. S_x proportional to I_p .

What can go wrong?

$\lambda_{max}(S_x)$ is large and $\lambda_{min}(S_x)$ is small…

- Linear "concentration" of the data cloud in feature space, collinearity in the sample. Because Σ_x contains collinearity, and/or because n is not large enough wrt p.
- An ineffective re-mapping inflates the sampling variability in the LS estimates of the feature effects, possibly makes them poorly determined or non-unique on any given sample.

What happens as p grows wrt n; p~n, p>n, p>>n?

- Σ_x is likely to comprise more collinearity; we are considering more and more features which may carry linear associations, and
- S_x becomes progressively more collinear than its population counterpart, just because we have insufficient replication

What do we do?

$$\hat{\beta}_{MP} = (X'X)^{+}X'Y$$ More-Penrose generalized inverse

$$\hat{\beta}_{Ridge}(\lambda) = (X'X + \lambda I)^{-1}X'Y$$ Regularize with a size constraint, L2

$$\hat{\beta}_{LASSO}(\lambda) = argmin\{||Y - X\beta||^2 + \lambda\, L1(\beta)\}$$ Regularize and sparsify with a size constraint, L1

… and more sophisticated approaches; SCAD, Adaptive LASSO, Danzig Selector. Also old fashioned Partial Least Squares.

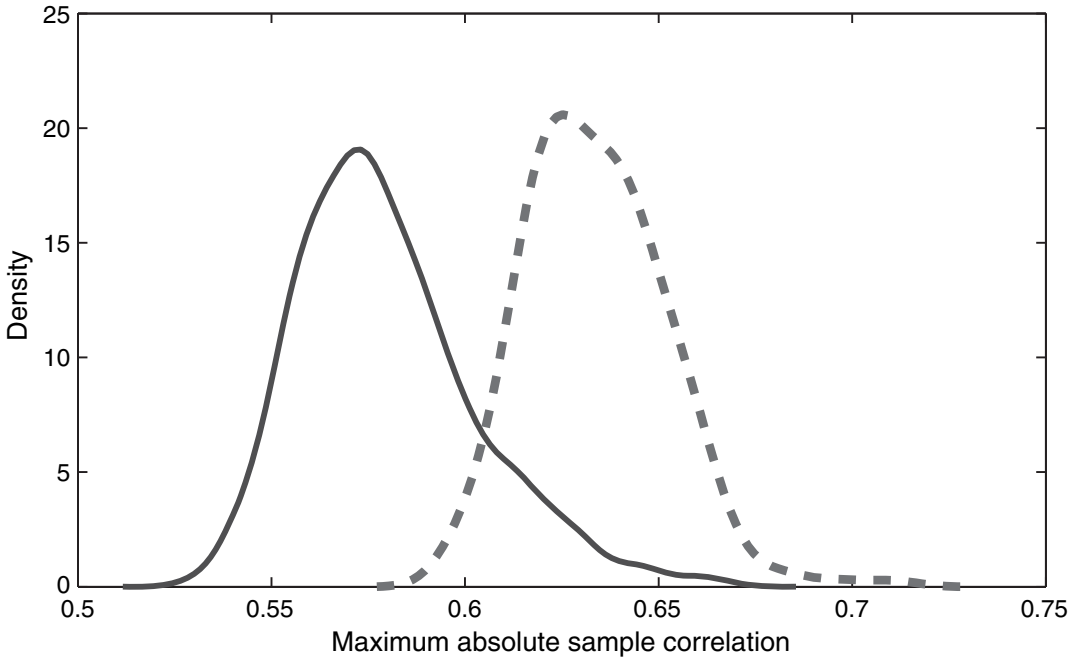


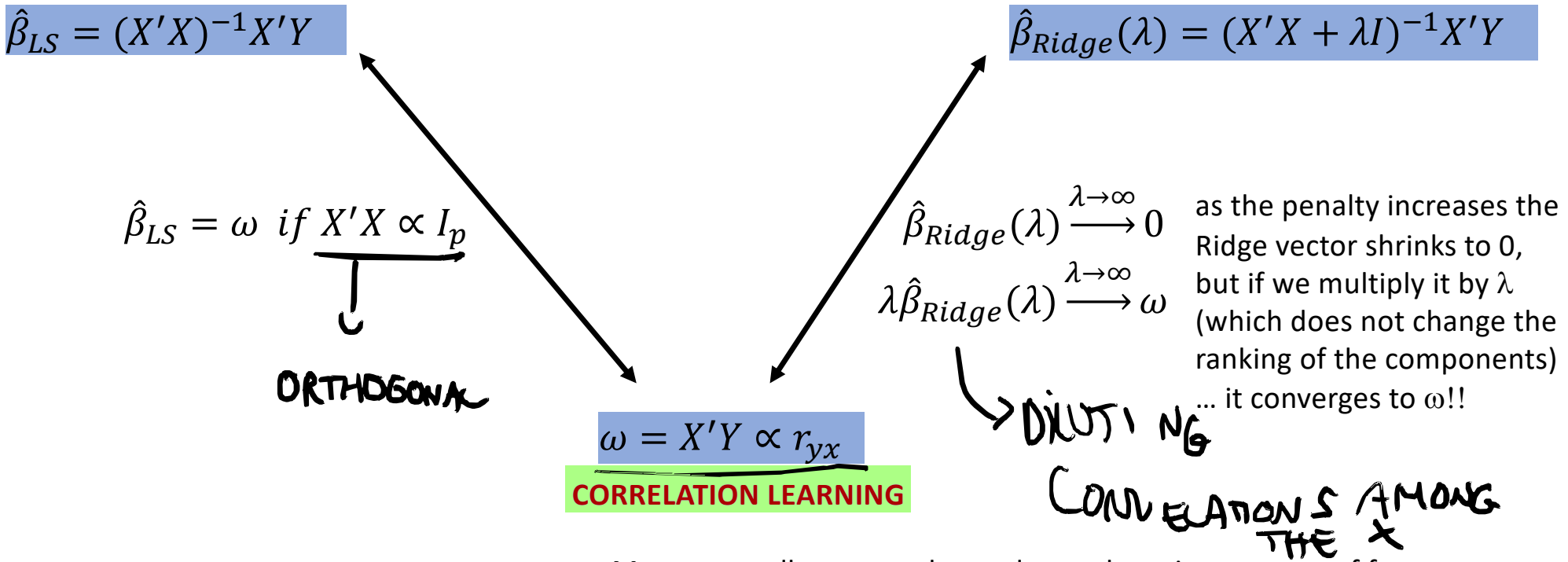
Fig. 1. Distributions of the maximum absolute sample correlation coefficient when $n = 60$ and $p = 1000$ (———) and $n = 60$ and $p = 5000$ (- - - - -), based on 500 simulations

X data simulated with Σ_x proportional to I_p.

No correlations at the population level, yet if p >> n the correlations in S_x are large!

What if we ignored the re-mapping all together?

Base the ranking and choice of d features on the component-wise regression, i.e., on the *marginal correlations*



as the penalty increases the Ridge vector shrinks to 0, but if we multiply it by λ (which does not change the ranking of the components) … it converges to ω!!

More generally, approaches to learn about importance of features ***marginally***; these can be applied with small computational burden and without the curse of dimensionality even when p >> n

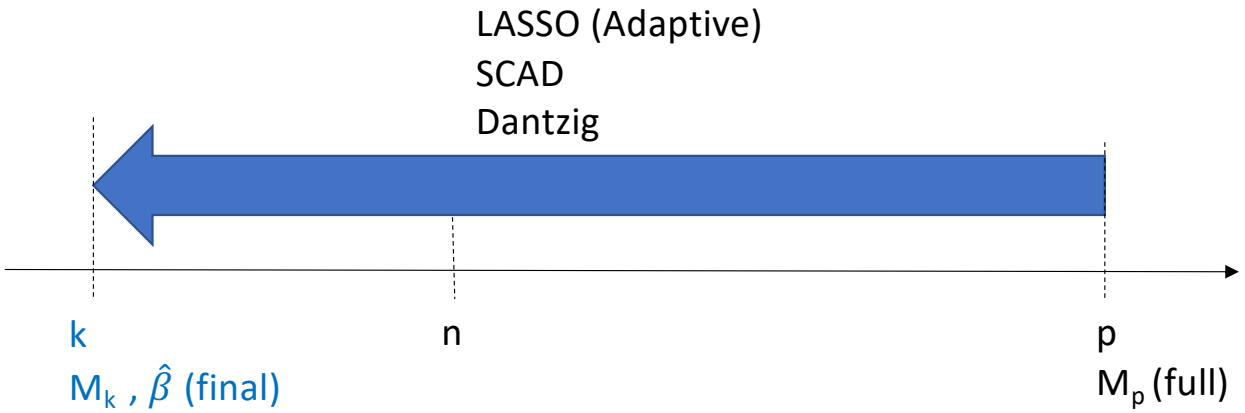
IMPORTANT REMARK: A NEW ASYMPTOTICS

In addition to problems with p >> n, starting with the new century we have problems where $p = p(n)$. As n grows, p grows with it, possibly exponentially $\log(p) = O(n^a)$ though perhaps with small a.

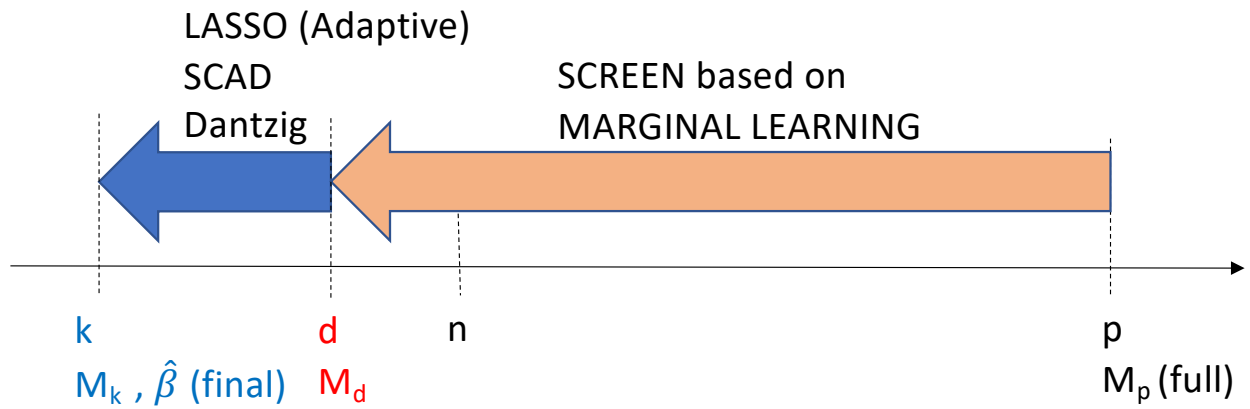
The more we sample, observing more units, the more the dimension of our feature space increases. Think of variants (SNPs) as we sequence genomes of more individuals, or recorded product choices/scores as we branch out over a network of individuals on social media. The ”universe” of the observed SNPs (of products) increases as we include more people.

In Statistics this required the formulation of a **new type of asymptotics**; study the performance of statistical procedures as n grows, and p(n) grows with it – possibly not too fast.

The proposal of **Fan and Lv (2008)** : If we believe the underlying true model is sparse, instead of



use a reasonable d < n, e.g., d = n-1, or d = n/log(n) and proceed in two stages



Screening disregards associations among the X's, but when p >> n our ability to account for them and re-map is so poor that it may hurt more than it helps.

To get to the true sparse model M_* with s features, and to estimate its true β, the two-stage strategy (which uses only marginal information when the dimension is still ultra-high) is more effective!!

Not a new idea, practitioners have used it forever, but now formalized. And its performance established by simulations and "new asymptotics" theoretical results.

SURE INDEPENDENCE SCREENING (SIS)

- SURE SCREENING PROPERTY: with a reasonable d = n-1 or n/log(n) and some assumptions on the nature of the stochastic mechanism generating the data and the speed at which p(n)>>n grows with n, when n is large $Pr(M_* \text{ is contained in } M_d)$ is "overwhelming" (growing to 1).
- INDEPENDENCE: refers to the fact that we operate on marginal information, one X at a time.

The SCREENING algorithm is trivial and computationally inexpensive:

- Compute the marginal correlations $\omega = X'Y \propto r_{yx}$
- Rank the entries in ω and pick the first d to form M_d .

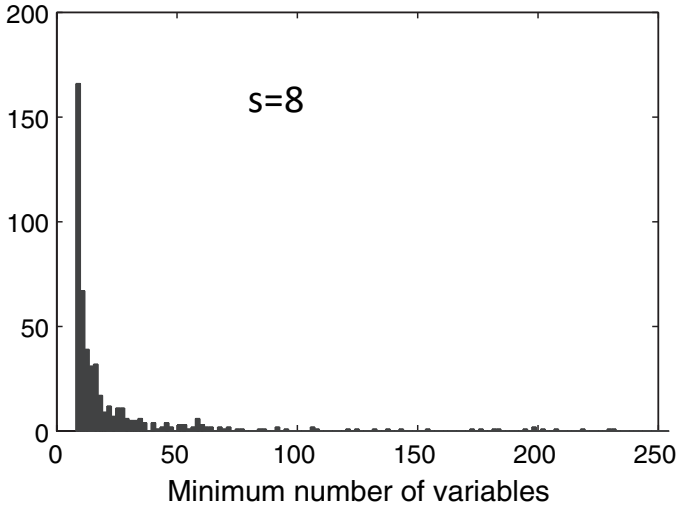
Next, apply feature selection algorithm XXX to the regression of Y on the features contained in M_d to obtain M_k , $\hat{\beta}$ (final).

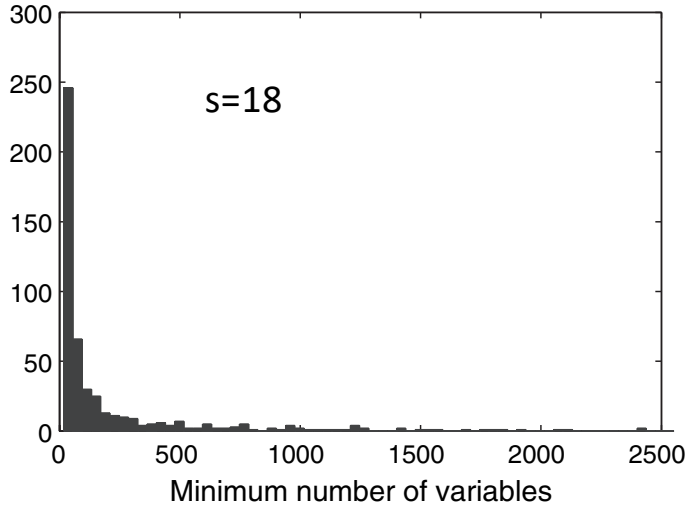
First, use **simulations** to assess whether, on finite (but large) samples

- the sure screening property holds for SIS, and
- M_k is close to M_* (k close to s) and $\hat{\beta}$ close to the true β for SIS-XXX

In particular

- do we do better than with one pass of XXX? and
- is performance affected by the presence of collinearity in Σ_x (population level)?





SIMULATIONS WITH UNCORRELATED X'S

Fig. 5. Distribution of the minimum number of selected variables that is required to include the true model by using SIS when (a) $n = 200$ and $p = 1000$ and (b) $n = 800$ and $p = 20\,000$ in simulation I

OK!

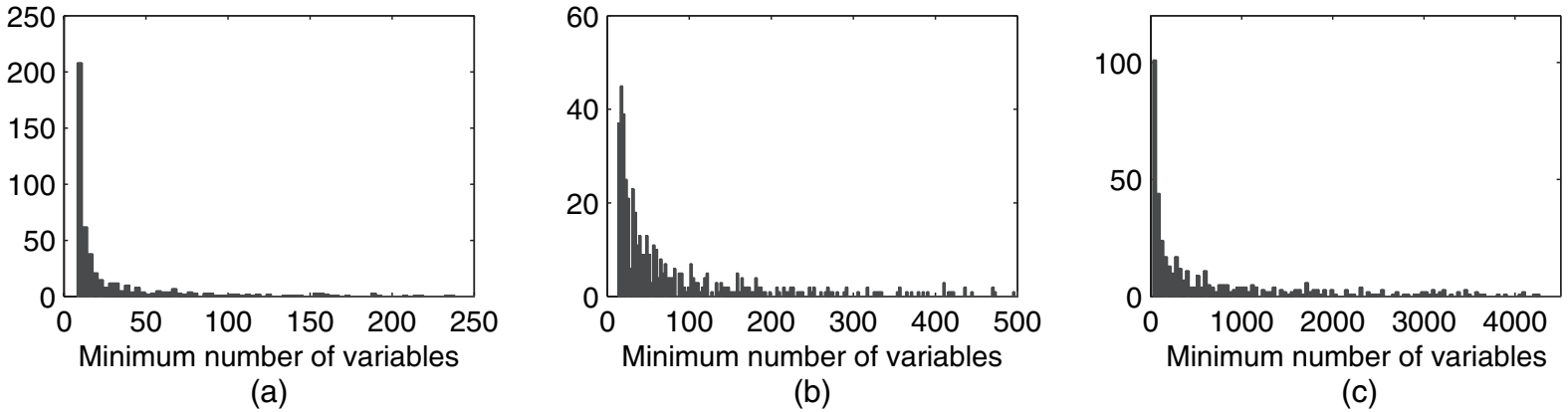
Table 1. Results of simulation I: medians of the selected model sizes and estimation errors (in parentheses)

k $\|\widehat{\beta} - \beta\|^2$

p	*Results for the following methods:*					
	Dantzig selector	*Lasso*	*SIS–SCAD*	*SIS–DS*	*SIS–DS–SCAD*	*SIS–DS–AdaLasso*
1000 s=8	10^3 (1.381)	62.5 (0.895)	15 (0.374)	37 (0.795)	27 (0.614)	34 (1.269)
20000 s=18	— —	— —	37 (0.288)	119 (0.732)	60.5 (0.372)	99 (1.014)

one pass

One pass does very poorly at imposing sparsity and at estimating effects. And for p=20,000 could NOT be run in reasonable time (in 2008).



SIMULATIONS WITH CORRELATED X'S

Fig. 6. Distribution of the minimum number of selected variables that is required to include the true model by using SIS when (a) $n = 200$, $p = 1000$ and $s = 5$, (b) $n = 200$, $p = 1000$ and $s = 8$ and (c) $n = 800$, $p = 20\,000$ and $s = 8$ in simulation II s=14?

Table 2. Results of simulation II: medians of the selected model sizes and estimation errors (in parentheses)

k $\|\hat{\beta} - \beta\|^2$

p	*Results for the following methods:*					
one pass	*Dantzig selector*	*Lasso*	*SIS–SCAD*	*SIS–DS*	*SIS–DS–SCAD*	*SIS–DS–AdaLasso*
1000	10^3	91	21	56	27	52
($s = 5$)	(1.256)	(1.257)	(0.331)	(0.727)	(0.476)	(1.204)
($s = 8$)	10^3	74	18	56	31.5	51
	(1.465)	(1.257)	(0.458)	(1.014)	(0.787)	(1.824)
20000	—	—	36	119	54	86
s=14?	—	—	(0.367)	(0.986)	(0.743)	(1.762)

OK! also with correlated X's

One pass does very poorly at imposing sparsity and at estimating effects. And for p=20,000 could NOT be run in reasonable time (in 2008).

Theoretical Assessments

SURE SCREENING PROPERTY OF SIS:

- Let M_d = SIS(M_p) with d = n/log(n)
- *Impose assumptions on the nature of the stochastic mechanism generating the data, and on the speed at which p(n)>>n grows with n*
- Prove

$$\Pr(M_* \subseteq M_d) \geq 1 - (fast\ vanishing\ as\ n \to \infty)$$

Also, again under appropriate assumption, prove

- CONSISTENCY FOR SIS-DANTZIG $\quad \hat{\beta} \xrightarrow[n\to\infty]{} \beta$

- ORACLE PROPERTY FOR SIS-SCAD (i) $\hat{\beta}_j = 0\ for\ any\ j \notin M_*$

  (ii) $\widehat{\beta}$ does as well for β as if we had run standard LS **knowing** the true M_*

Assumptions

Stochastic mechanism:

$Y, \varepsilon \;\; r.vbls \; in \; R^1 \;\; X, Z = \Sigma_X^{-1/2} X \; r.vct \; in \; R^p$

$E(Z) = 0 \quad Cov(Z) \propto I_p$

$\varepsilon \; indep \; Z \quad \varepsilon \sim N(0, \sigma^2)$

Independent sampling:

$$Y_{nx1} = X_{nxp}\beta_{px1} + \varepsilon_{nx1} = Z_{nxp}\Sigma_X^{1/2}\beta_{px1} + \varepsilon_{nx1}$$

(I) Var(Y) does not grow, and the effects for the relevant features are strong enough, as n grows; for some $\kappa \geq 0$

$$\min_{j \in M_*} |\beta_j| \geq \frac{pos \; const}{n^{\kappa}}$$

(II) The relevant features have a strong enough "trace" in their marginal linear associations with Y

$$\min_{j \in M_*} \left| cov\left(\frac{Y}{\beta_J}, X_j\right) \right| \geq pos \; const$$

p > n grows exponentially, but slow enough relative to the strength of the signals; for some $0 < \alpha <$ (1-2κ)

$$\log(p) = O(n^{\alpha})$$

(III) The feature collinearity (population level) remains weak enough as p(n) grows ; for some $\tau \geq 0$

$$\lambda_{max(\Sigma_X)} = \leq pos \; const \times n^{\tau}$$

Z is spherically symmetric and, for any submatrix with more than n columns $\tilde{Z}_{nxh}, n < h \leq p$, $\tilde{Z}_{nxh}\tilde{Z}_{nxh}'$ has all n eigenvalues of the same order – not too dissimilar (adds to the required symmetry for the features). Nothing much beyond Σ_x. This certainly holds if X is Gaussian (Z standard Gaussian).

SIS-XXX can perform poorly though (even for fairly large n) if:

- The effects of some relevant features are weaker than the spurious marginal traces carried by non-relevant features correlated with strong relevant ones; failure in (I), (III).
- Some relevant features have weak marginal linear traces – they may act only jointly with other features, or act with little linear trend; failure in (II).

Performance can be rescued through another very old and effective idea: iterate the procedure, working with residuals as to magnify weaker signals: **ITERATIVE SIS**, ISIS-XXX. The algorithm:

- SIS for Y on M_p => $M^{(1)}_{dIT}$
- XXX for Y on $M^{(1)}_{dIT}$ => ($M^{(1)}_{k(1)}$, $\hat{\beta}_{(1)}$)
- $A^{(1)} = M^{(1)}_{k(1)}$
- $e^{(1)} = Y - X_{A(1)} \hat{\beta}_{(1)}$

Iteration 1

- SIS for $e^{(1)}$ on $M_p \setminus A^{(1)}$ => $M^{(2)}_{dIT}$
- XXX for $e^{(1)}$ on $M^{(2)}_{dIT}$ => ($M^{(2)}_{k(2)}$, $\hat{\beta}_{(2)}$)
- $A^{(2)} = A^{(1)} \cup M^{(1)}_{k(1)}$
- $e^{(2)} = Y - X_{A(2)} \hat{\beta}_{(2)}$

Iteration 2

… etc. (usually just a few times). Stop iterating when #($A^{(h)}$) reaches n-1 or n/log(n)

XXX for Y on $A^{(h)}$ => (M_k , $\hat{\beta}$) the *final result*.

ITERATIONS HELP

Table 1. Results of simulation I: medians of the selected model sizes and estimation errors (in parentheses)

p	Results for the following methods:					
	Dantzig selector	*Lasso*	*SIS–SCAD*	*SIS–DS*	*SIS–DS–SCAD*	*SIS–DS–AdaLasso*
1000	10^3	62.5	15	37	27	34
s=8	(1.381)	(0.895)	(0.374)	(0.795)	(0.614)	(1.269)
20000	—	—	37	119	60.5	99
s=18	—	—	(0.288)	(0.732)	(0.372)	(1.014)

Table 2. Results of simulation II: medians of the selected model sizes and estimation errors (in parentheses)

p	Results for the following methods:					
	Dantzig selector	*Lasso*	*SIS–SCAD*	*SIS–DS*	*SIS–DS–SCAD*	*SIS–DS–AdaLasso*
1000	10^3	91	21	56	27	52
($s=5$)	(1.256)	(1.257)	(0.331)	(0.727)	(0.476)	(1.204)
($s=8$)	10^3	74	18	56	31.5	51
	(1.465)	(1.257)	(0.458)	(1.014)	(0.787)	(1.824)
20000	—	—	36	119	54	86
s=14?	—	—	(0.367)	(0.986)	(0.743)	(1.762)

FASTER, ITERATIONS DEGRADES

Table 7. Simulations I and II in Section 3.3 revisited: medians of the model sizes selected and the estimation errors (in parentheses) for the ISIS–SCAD method

p	*Results for simulation I*		*Results for simulation II*
1000	13	($s=5$)	11
s=8	(0.329)		(0.223)
		($s=8$)	13.5
			(0.366)
20000	31		27
s=18	(0.246)	s=14?	(0.315)

Iterations work!

In the article more simulation results to demonstrate effectiveness against more specific failure scenarios

After the seminal paper by Fan and Lv (2008), there has been a deluge of literature on screening algorithms for ultra-high dimensional supervised problems.

- **Extension to GLMs**: Fan J., Samworth R., Wu Y. (2009) Ultrahigh dimensional feature selection: beyond the linear model. Journal of Machine Learning Research, 10 2013-2038.

- **Extension to model-free settings**: Zhu L., Li L., Li R., Zhu L. (2011). Model-free feature screening for ultra-high dimensional data. JASA 106(496) 1464-1475.

… and many others. Still a very active area of research.

What changes is the implementation of marginal learning; **ω is defined differently** (based on a GLM framework, or non-parametrically). Also, theoretical results can be different and require alternative proof strategies.

Remark: a screening algorithm must be ***conservative, the focus is on controling false negatives***. We want to make sure we do not leave out any relevant feature; more features can be eliminated as needed by feature selection after screening. If we consider SIS-XXX as a whole, it makes sense to also assess performance in terms of false positives (we do not want to finish with a k much larger than s; the size of the true sparse model) and in terms of estimation quality (how close $\hat{\beta}$ comes to the true β).

A package published in 2018 that implements various versions of sure independence screening:

D.F. Saldana, Y. Feng (2018) SIS: An R Package for Sure Independence Screening in Ultrahigh-Dimensional Statistical Models. *Journal of Statistical Software*, *83*(2), 1–25. https://doi.org/10.18637/jss.v083.i02

Abstract
We revisit sure independence screening procedures for variable selection in generalized linear models and the Cox proportional hazards model. Through the publicly available R package SIS, we provide a unified environment to carry out variable selection using iterative sure independence screening (ISIS) and all of its variants. For the regularization steps in the ISIS recruiting process, available penalties include the LASSO, SCAD, and MCP while the implemented variants for the screening steps are sample splitting, data-driven thresholding, and combinations thereof. Performance of these feature selection techniques is investigated by means of real and simulated data sets, where we find considerable improvements in terms of model selection and computational time between our algorithms and traditional penalized pseudo-likelihood methods applied directly to the full set of covariates.

Some additions: covariate information matrices and numbers for supervised dimension reduction and feature screening

SUFFICIENT DIMENSION REDUCTION BASED ON INFORMATION MATRICES

Theory and Methods

Covariate Information Matrix for Sufficient Dimension Reduction

Weixin Yao, Debmalya Nandy, Bruce G. Lindsay & Francesca Chiaromonte

Pages 1752-1764 | Received 03 Mar 2017, Accepted 10 Aug 2018, Accepted author version posted online: 06 Sep 2018, Published online: 27 Feb 2019

Download citation https://doi.org/10.1080/01621459.2018.1515080 Check for updates

Covariate Information Matrix (CIM): combine "local" non-parametric assessment of densities and "global" Eigen-decomposition, in an ***Information Matrix framework***.

… prior applications to:

- Projection Pursuit
- Spherical Symmetry, Multivariate Structures
- Independent Components Analysis
- Graphical Models

Hui G., Lindsay B. (2010). Projection pursuit via white noise matrices . Sankhya B, 72(2): 123–153 .

Lindsay B., Yao W. (2012). Fisher information matrix: A tool for dimension reduction, projection pursuit, independent component analysis, and more. Canadian Journal of Statistics, 40(4): 712–730.

What is the theory behind this?

$U_{\mathbf{x}}(y) = \nabla_{\mathbf{x}} \log f(y \mid \mathbf{x})$ score vector

$\mathbb{F}_{\mathbf{x}} = \int U_{\mathbf{x}}(y) U_{\mathbf{x}}(y)^T f(y \mid \mathbf{x}) dy$ ***Fisher Information Matrix*** for "parameter" $\boldsymbol{x}$
How $\boldsymbol{Y} \mid \boldsymbol{X}=\boldsymbol{x}$ changes with $\boldsymbol{x}$

$\mathbb{C}_{\mathbf{X}} = \int \mathbb{F}_{\mathbf{x}} f(\mathbf{x}) d\mathbf{x}$ average on X … ***Covariate Information Matrix***

Key result: under very general conditions one has $\boldsymbol{Span[\mathbb{C}_X] \equiv \Sigma_X S_{Y|X}}$ (the Central Subspace of SDR)

$U_f(\mathbf{x}) = \nabla_{\mathbf{x}} \log f(\mathbf{x})$

$\mathbb{J}_{\mathbf{X}} = \int U_f(\mathbf{x}) U_f(\mathbf{x})^T f(\mathbf{x}) d\mathbf{x}$ ***Density Information Matrix*** for $\boldsymbol{X}$
characterization of the $\boldsymbol{X}$ distribution

$U_{f^{(y)}}(\mathbf{x}) = \nabla_{\mathbf{x}} \log f^{(y)}(\mathbf{x})$

$\mathbb{J}_{\mathbf{X}|Y=y} = \int U_{f^{(y)}}(\mathbf{x}) U_{f^{(y)}}(\mathbf{x})^T f^{(y)}(\mathbf{x}) d\mathbf{x}$ Density Information Matrix for $\boldsymbol{X} \mid \boldsymbol{Y}=\boldsymbol{y}$

$\mathbb{J}_{\mathbf{X}|Y} = \int \mathbb{J}_{\mathbf{X}|Y=y} f(y) dy$ average on Y … ***Inverse Regression***

Key result: $\boxed{\mathbb{C}_{\boldsymbol{X}} = \mathbb{J}_{\boldsymbol{X}|Y} - \mathbb{J}_{\boldsymbol{X}}}$ rewrite CIM in terms of X-related DIMs
Inverse Regression, "adjusted" for $\boldsymbol{X}$

What is the implementation?

(2) "global" ***Eigen-decomposition*** $\hat{\mathbb{C}}_X = \sum_{j=1}^{p} \lambda_j v_j v_j^T \rightarrow \hat{S}_{Y|X} = \hat{\Sigma}_X^{-1} Span[v_1 \ldots v_d]$

Key result: **without (L) and (C), one has** $Span[\mathbb{C}_X] \equiv \Sigma_X S_{Y|X}$

(1) Inexpensive, effective estimation of $f(x)$ and $f(x|y_\ell), \ell = 1 \ldots L$ (slices) with the ***f2-method for non-parametric density estimation*** (use of squared surrogates preserving peaks and valleys) $\hat{J}_X , \hat{J}_{X|Y} \rightarrow \hat{\mathbb{C}}_X$

Key result: $\mathbb{C}_X = \mathbb{J}_{X|Y} - \mathbb{J}_X$

What is the implementation?

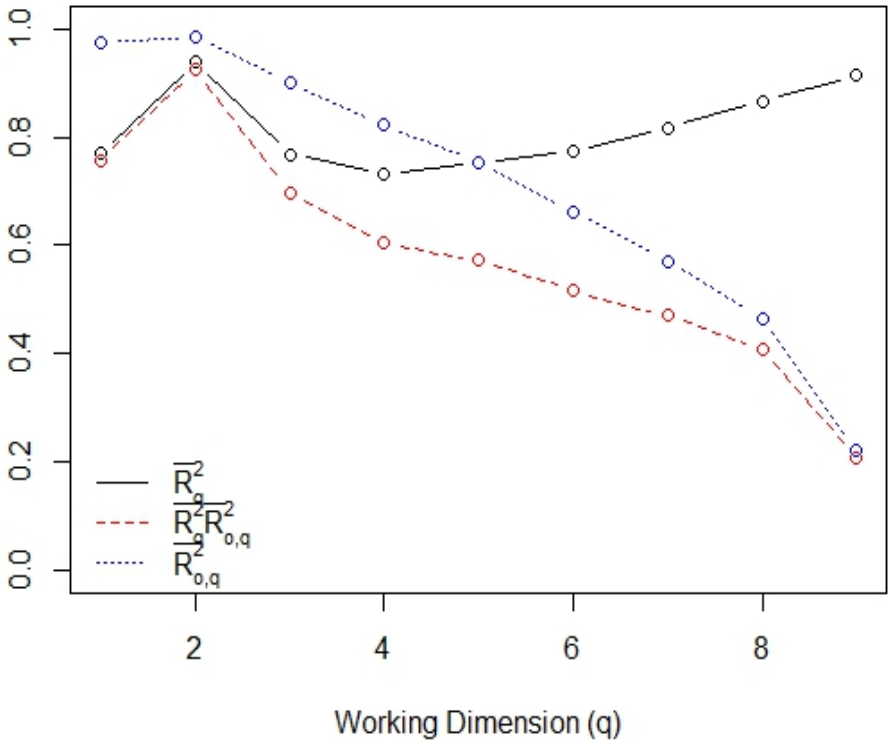
(2) "global" ***Eigen-decomposition*** $\hat{\mathbb{C}}_X = \sum_{j=1}^{p} \lambda_j v_j v_j^T \rightarrow \hat{S}_{Y|X} = \hat{\Sigma}_X^{-1} Span[v_1 \ldots v_d]$

dimension estimation
new bootstrap-based
diagnostic plot (see later)

(1) Inexpensive, effective estimation of $f(x)$ and $f(x|y_\ell), \ell = 1 \ldots L$ (slices) with the ***f2-method for non-parametric density estimation*** (use of squared surrogates preserving peaks and valleys) $\hat{J}_X , \hat{J}_{X|Y} \rightarrow \hat{\mathbb{C}}_X$

Diagnostic plot for dimension estimation

(applicable with <u>any</u> SDR method)



Heteroscedastic Y scenario
$\sigma = 0.2$, *Independent* **X**, $n = 400$
CIM with 5 slices

$R^2_q(\mathcal{S}_1, \mathcal{S}_2) = \frac{1}{q}\text{tr}(P_{\mathcal{S}_1} P_{\mathcal{S}_2})$ ***Squared Trace Correlation***, 1 for q=p

$R^2_{o,q}(\mathcal{S}_1, \mathcal{S}_2) = R^2_q(\mathcal{S}_1^\perp, \mathcal{S}_2^\perp)$ complement version, 1 for q=0

Bootstrap Scheme: For each "working dimension" $1 \le q \le p-1$:

- Estimate $\hat{\mathcal{S}}_q$
- For $j = 1, \ldots, B$ bootstrap replicates, estimate $\hat{\mathcal{S}}_q^{(j)}$ and compute $R_q^{2(j)} = R^2_q(\hat{\mathcal{S}}_q, \hat{\mathcal{S}}_q^{(j)})$, $R_{o,q}^{2(j)} = R^2_{o,q}(\hat{\mathcal{S}}_q, \hat{\mathcal{S}}_q^{(j)})$, and $R_q^{2(j)} R_{o,q}^{2(j)}$.
- Calculate the averages $\bar{R}^2_q$, $\bar{R}^2_{o,q}$, and $\overline{R^2_q R^2_{o,q}}$.

$\bar{R}^2_q$, $\bar{R}^2_{o,q}$ and $\overline{R^2_q R^2_{o,q}}$ all measure 'stability' in estimating $\mathcal{S}_q$.

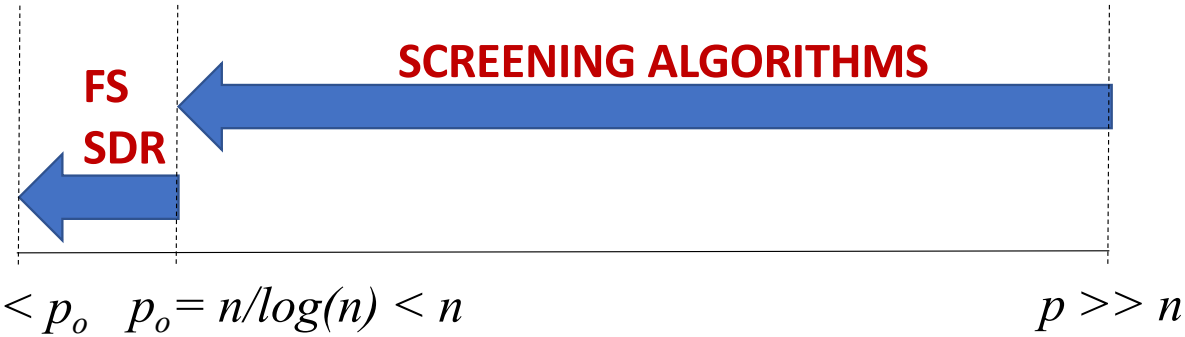
Select $\hat{d}$ where $\overline{R^2_q R^2_{o,q}}$ has a "peak".

What to do when the feature space becomes very high dimensional?

Cut p based on ***marginal utilities***, e.g.

$$\omega_j = corr(Y, X_j) \quad j = 1,2 \dots p$$

Disregard associations among the X's but with $p >> n$ our ability to account for them is too poor!



Covariate Information Number (one feature at a time)

$$\omega_j = \mathbb{C}_{X_j} = \int \mathbb{F}_{x_j} f(x_j) dx_j = \mathbb{J}_{X_j|Y} - \mathbb{J}_{X_j} \quad j = 1,2 \dots p$$

$$\int \left[\frac{\partial}{\partial x_j} \log f(y \mid x_j) \right]^2 f(y \mid x_j)\, dy$$

Theory and Methods

Covariate Information Number for Feature Screening in Ultrahigh-Dimensional Supervised Problems

Debmalya Nandy, Francesca Chiaromonte & Runze Li

Received 01 Nov 2018, Accepted 08 Dec 2020, Accepted author version posted online: 16 Dec 2020, Published online: 10 Feb 2021

Download citation https://doi.org/10.1080/01621459.2020.1864380 Check for updates

Covariate Information Number for Feature Screening in Ultrahigh-Dimensional Supervised Problems

Abstract: Contemporary high-throughput experimental and surveying techniques give rise to ultrahigh-dimensional supervised problems with sparse signals; that is, a limited number of observations (*n*), each with a very large number of covariates ($p \gg n$), only a small share of which is truly associated with the response. In these settings, major concerns on computational burden, algorithmic stability, and statistical accuracy call for substantially reducing the feature space by eliminating redundant covariates before the use of any sophisticated statistical analysis. Along the lines of *Pearson's correlation coefficient-based sure independence screening* and other model- and correlation-based feature screening methods, we propose a model-free procedure called *covariate information number-sure independence screening* (CIS). CIS uses a marginal utility connected to the notion of the traditional Fisher information, possesses the sure screening property, and is applicable to any type of response (features) with continuous features (response). Simulations and an application to transcriptomic data on rats reveal the comparative strengths of CIS over some popular feature screening methods.

How Strong

$$\frac{\text{SIGNAL}}{\text{NOISE}} \qquad \frac{\text{Var}\left(\beta^T x\right)}{\sigma^2}$$

$$y = \beta_0 + \beta^T x + \varepsilon \qquad \text{Var}(\varepsilon) = \sigma^2$$