Mathematical Tools
for Neural and Cognitive Science

Fall semester, 2022

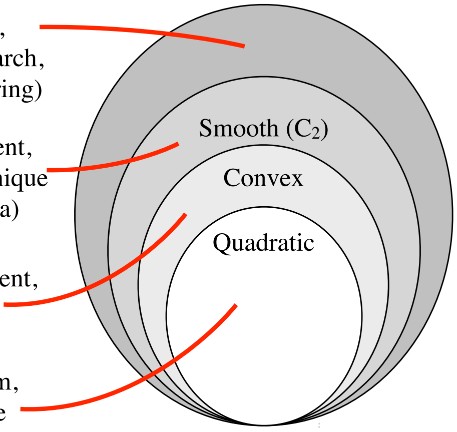
Section 6

Model fitting:
comparison, selection and regularization

Taxonomy of model-fitting errors

- Unexplainable variability (due to finite/noisy measurements)
- Overfitting (too many params, not enough data)
- Optimization failures (e.g., local minima)
- Model failures (what you’d really like to know)

Optimization...



Heuristics, exhaustive search, (pain & suffering)

Iterative descent, possibly non-unique (local minima)

Iterative descent, unique

Closed-form, and unique

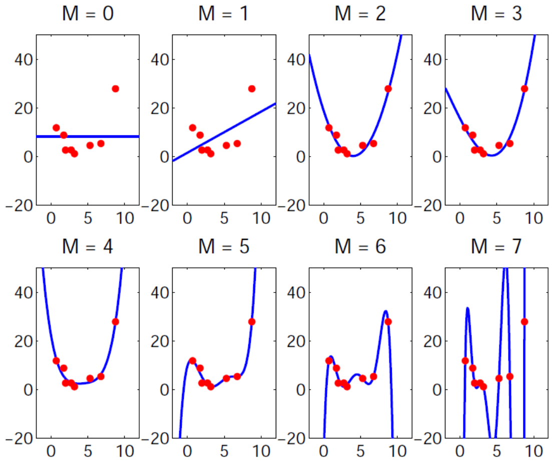
Smooth (C_2)

Convex

Quadratic

Model Comparison

- If models are optimized according to some objective, it is natural to compare them based on the value of that objective…
  - for least squares regression, compare the residual squared error of two models (with different regressors).
  - for ML estimates, compute the likelihood (or log likelihood) ratio, and compare to 1 (or zero).
  - for MAP estimates, common to compute the posterior ratio (a.k.a. the *Bayes factor*)
- **Problem**: evaluating the objective with the same data used to optimize the model leads to over-fitting! We really want to predict error on non-training data…



How do we avoid overfitting (i.e., concluding that M=7 is "best")?

Comparing models' predictive performance

Option 1: Include a penalty for number of parameters:

For an ML estimate: $\hat{\theta} = \arg\min_{\theta} p(\vec{d}\,|\,\theta)$

a. Compare Akaike information criterion (AIC) [Akaike, 1974]

$$E_{\text{AIC}}(\vec{d},\hat{\theta}) = 2\ \dim(\hat{\theta}) - 2\ln\left(p(\vec{d}\,|\,\hat{\theta})\right)$$

b. Compare Bayesian information criterion (BIC) [Schwartz, 1978]

$$E_{\text{BIC}}(\vec{d},\hat{\theta}) = \dim(\hat{\theta})\ \ln\left(\dim(\vec{d})\right) - 2\ln\left(p(\vec{d}\,|\,\hat{\theta})\right)$$

valid when $\dim(\vec{d}) >> \dim(\hat{\theta})$

Option 2: Cross-validation: partition data into two subsets, fit parameters to "training" subset, evaluate objective on "test" subset.

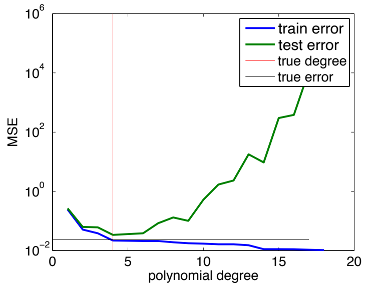
Cross-validation

A resampling method for estimating predictive error of a model. Widely used to identify/avoid over-fitting, and to provide a fair comparison of models.

(1) Randomly partition data into a “training” set, and a “test” set.
(2) Fit model to training set. Measure error on test set.
(3) Repeat (many times)

(4) Choose model that minimizes the average cross-validated (“test”) error

Using cross-validation to select the degree of a polynomial model:



Ridge regression

(a.k.a. Tikhonov regularization)

Ordinary least squares regression:

$$\arg\min_{\vec{\beta}} ||\vec{y} - X\vec{\beta}||^2$$

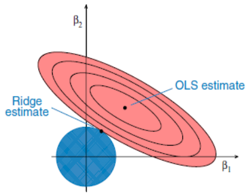
“Regularized” least squares regression:

$$\arg\min_{\vec{\beta}} ||\vec{y} - X\vec{\beta}||^2 + \lambda||\vec{\beta}||^2$$

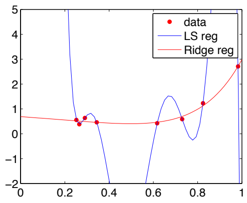
Equivalent formulation: MAP estimate, assuming Gaussian likelihood & prior!

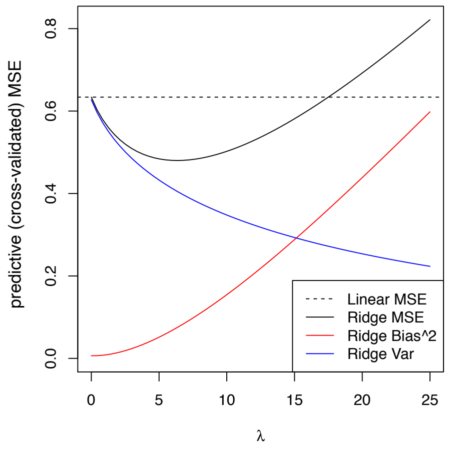
$$\hat{\beta}_{\text{ridge}} = (X^T X + \lambda I)^{-1} X^T \vec{y}$$

Choose lambda by cross-validation:

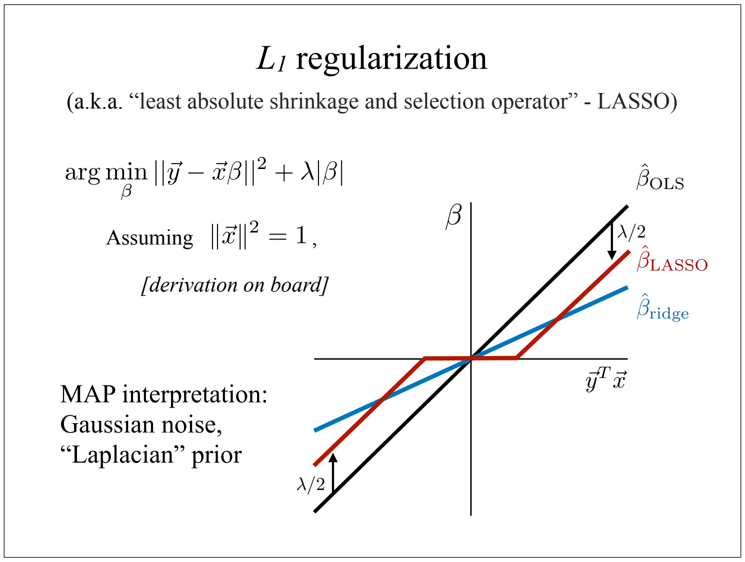


7th-order polynomial regression:





from http://www.stat.cmu.edu/~ryantibs/datamining/



L_1 regularization

(a.k.a. "least absolute shrinkage and selection operator" - LASSO)

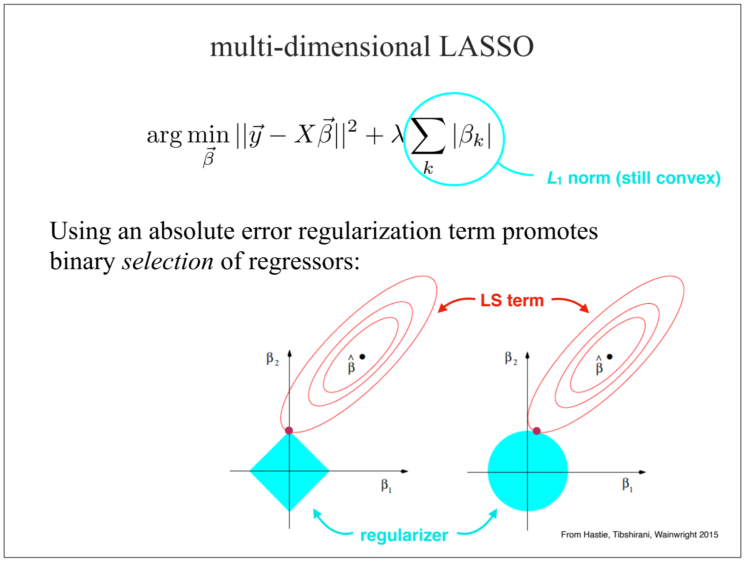
$$\arg\min_{\beta} ||\vec{y} - \vec{x}\beta||^2 + \lambda|\beta|$$

Assuming $\|\vec{x}\|^2 = 1$,

[derivation on board]

MAP interpretation:
Gaussian noise,
"Laplacian" prior

$\hat{\beta}_{\text{OLS}}$
β
$\lambda/2$
$\hat{\beta}_{\text{LASSO}}$
$\hat{\beta}_{\text{ridge}}$
$\vec{y}^T\vec{x}$
$\lambda/2$



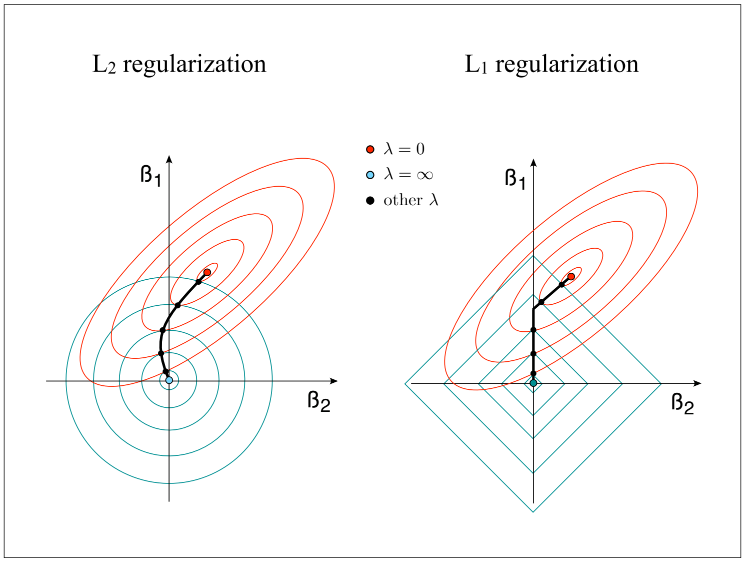
multi-dimensional LASSO

$$\arg\min_{\vec{\beta}} ||\vec{y} - X\vec{\beta}||^2 + \lambda\sum_k |\beta_k|$$

L_1 norm (still convex)

Using an absolute error regularization term promotes binary *selection* of regressors:

From Hastie, Tibshirani, Wainwright 2015



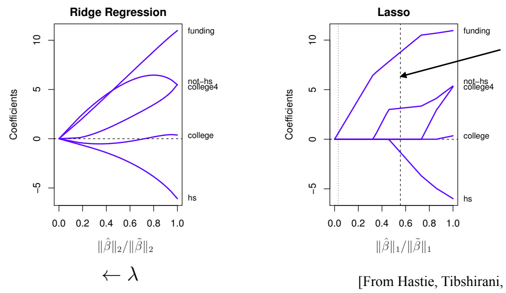
L_2 regularization

L_1 regularization

LASSO vs. ridge regression

Table 2.1 *Crime data: Crime rate and five predictors, for $N = 50$ U.S. cities.*

city	funding	hs	not-hs	college	college4	crime rate
1	40	74	11	31	20	478
2	32	72	11	43	18	494
3	57	70	18	16	16	643
4	31	71	11	25	19	341
5	67	72	9	29	24	773
⋮	⋮	⋮	⋮	⋮		
50	66	67	26	18	16	940

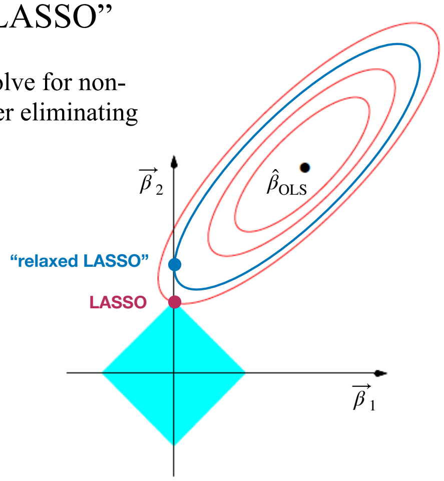


best (cross-validated) lambda

[From Hastie, Tibshirani, Wainwright 2015]

The “Relaxed LASSO”

To reduce bias, re-solve for non-zero coefficients after eliminating unused regressors



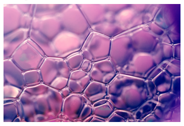
Clustering

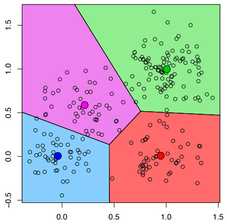
- K-Means (Lloyd, 1957)
- “Soft-assignment” version of K-means
  (a form of Expectation-Maximization - EM)
- In general, alternate between:
  1) Estimating cluster assignments
  2) Estimating cluster parameters
- Coordinate descent: converges to (possibly local) minimum
- Need to choose K (number of clusters) - cross-validation!

K-Means algorithm - alternate between two steps:

- Estimating cluster assignments: given class centers, assign each point to closest one.

Soap bubbles:

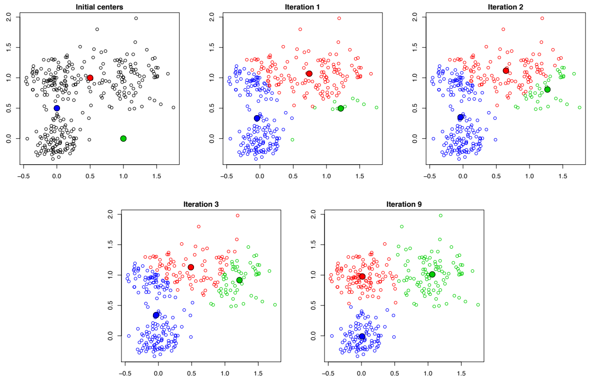




- Estimating cluster parameters: given assignments, re-estimate the centroid of each cluster.

K-means example

Here $X_i \in \mathbb{R}^2$, $n = 300$, and $K = 3$



Initial centers

Iteration 1

Iteration 2

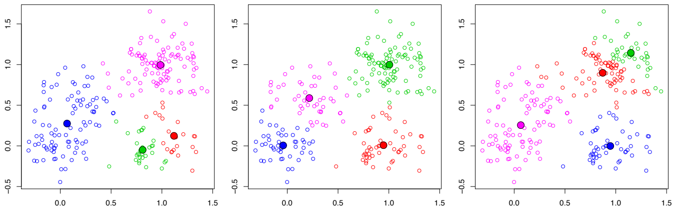
Iteration 3

Iteration 9

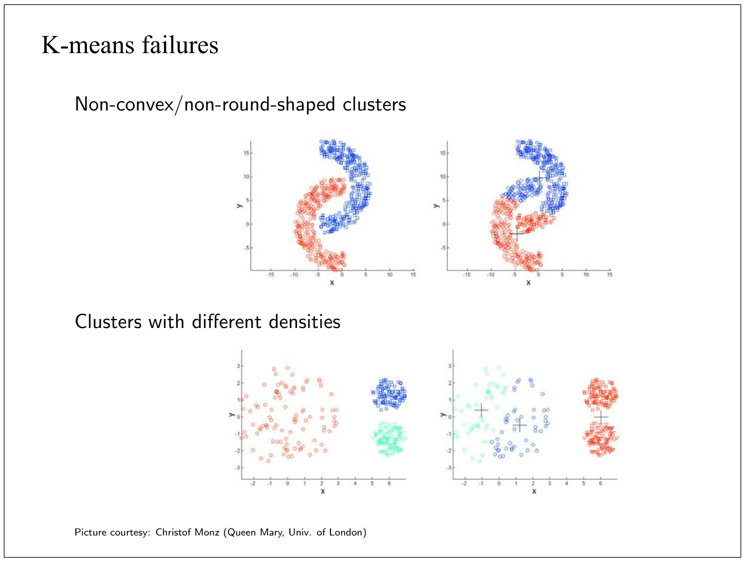
[from R. Tibshirani, 2013]

Warning: Initialization matters (due to local minima) …

Three solutions obtained with different random starting points:



[from R. Tibshirani, 2013]



K-means failures

Non-convex/non-round-shaped clusters

Clusters with different densities

Picture courtesy: Christof Monz (Queen Mary, Univ. of London)

ML for discrete mixture of Gaussians: soft K-means

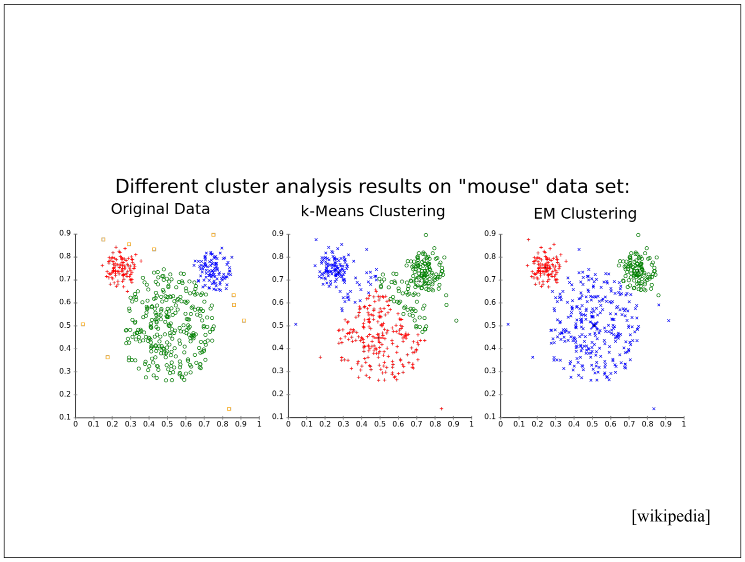
$$p(\vec{x}_n | a_{nk}, \vec{\mu}_k, \Lambda_k) \propto \sum_k \frac{a_{nk}}{\sqrt{|\Lambda_k|}} e^{-(\vec{x}_n - \vec{\mu}_k)^T \Lambda_k^{-1} (\vec{x}_n - \vec{\mu}_k)/2}$$

a_{nk} = assignment *probability*

$\{\vec{\mu}_k, \Lambda_k\}$ = mean/covariance of class *k*

Intuition: alternate between maximizing these two sets of variables (“coordinate descent”)

Essentially, a version of K-means with “soft” (i.e., continuous, as opposed to binary) assignments!



Different cluster analysis results on "mouse" data set:

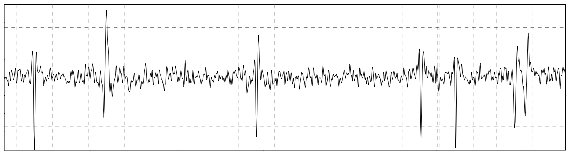
Original Data

k-Means Clustering

EM Clustering

[wikipedia]

Application to neural “spike sorting”

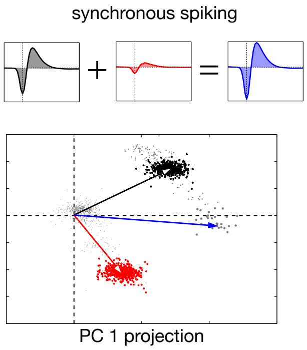


Standard solution:

1. Threshold to find segments containing spikes
2. Reduce dimensionality of segments using PCA
3. Identify spikes using clustering (e.g., K-means)

Note: Fails for overlapping spikes!

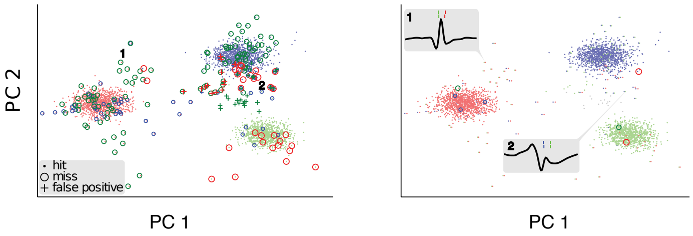
Failures of clustering for near-synchronous spikes



[Pillow et. al. 2013]

Simulated data [Quiroga et. al. 2004]

clustering (K-means) CBP



[Ekanadham et al, 2014]