Computational Data Analysis

Linear Classifiers and Basis Expansion

Lars Arvastson and Line Clemmensen

February 20, 2018

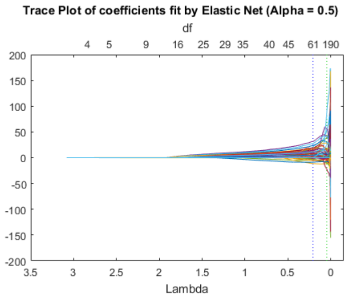
Todays Lecture

- Recap
- Linear Discriminant Analysis
- Logistic Regression
- Basis Expansion

Recap Lecture 3

- Curse of dimensionality
- Regularization
- Multiple testing

Recap Lecture 3




```
a = .5;

[B,FitInfo] =...
   lasso(X,Y,'alpha',a,'CV',5,'Standardize',true,'MCrep',1);

lassoPlot(B,FitInfo,'PlotType','CV');
lassoPlot(B,FitInfo,'PlotType','Lambda');
```

Linear discriminant analysis

Classification

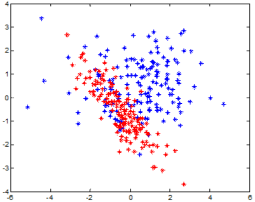
- Based on probability of class belonging
- Linear decision boundary

Linear Discriminant Analysis

- Classification from a probabilistic viewpoint
  - $P(G = k|X = x)$
  - Probability of class k, given observation x

Example:
$P(G = \text{red}|X = [0, -1])$ and
$P(G = \text{blue}|X = [0, -1])$

Predict that observation $X = [0, -1]$ belongs to the class with the **highest probability**



- We need a **stochastic model** for data to calculate probabilities
- Assume that data come from different **Gaussian distributions**
  - Different mean
  - Same correlation structure (just for simplicity)
- Data from different classes will overlap
- A straight line will be our decision boundary

Calculating class probabilities

$G(x)$ predicts class belonging for x,

$$G(x) = \arg\max_k \mathbf{P}(G = k | X = x).$$

Probablity given by **Bayes theorem**

$$\mathbf{P}(G = k | X = x) = \frac{f_k(x)\pi_k}{\sum_{\ell=1}^{k} f_\ell(x)\pi_\ell}$$

$f_\ell =$ distribution for class ℓ
$\pi_\ell =$ a priori probability for class ℓ (estimate or best guess)
Total probability, $\sum \pi_\ell = 1$.

Odds-rations

Look at log-**odds-ratio** for the two classes k and ℓ

$$\log \frac{\mathbf{P}(G=k|X=x)}{\mathbf{P}(G=\ell|X=x)} = \log \frac{f_k(x)\pi_k / \sum_i f_i \pi_i}{f_{\ell(x)}\pi_\ell / \sum_i f_i \pi_i}$$

$$= \log \frac{f_k(x)}{f_\ell(x)} + \log \frac{\pi_k}{\pi_\ell}$$

We must make an assumption about f.

Assume that data in each class follows a multivariate normal distribution,

$$f(x) = (2\pi)^{-p/2}|\Sigma_k|^{-1/2} e^{-\frac{1}{2}(x-\mu_k)^T \Sigma_k^{-1}(x-u_k)}$$

with a common covariance matrix $\Sigma_k = \Sigma$.

Linear decision boundary

$$\log \frac{\mathbf{P}(G=k|X=x)}{\mathbf{P}(G=\ell|X=x)}$$
$$= ...$$
$$= \log \frac{\pi_k}{\pi_\ell} - \frac{1}{2}(\mu_k+\mu_\ell)^T\Sigma^{-1}(\mu_k-\mu_\ell) + x^T\Sigma^{-1}(\mu_k-\mu_\ell)$$

Along the decision boundary we have $f_k\pi_k = f_\ell\pi_\ell$ (equal probability for both classes) and a log-odds-ratio $= \log 1 = 0$.

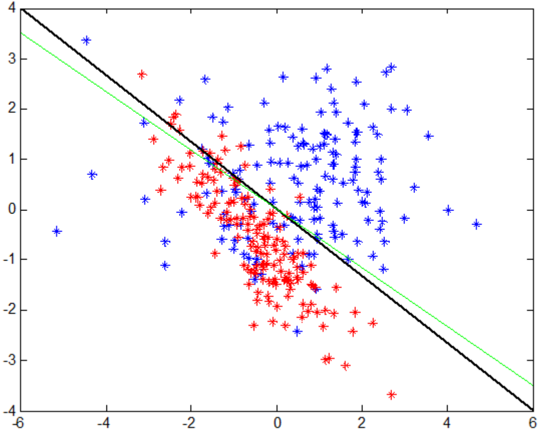
The decision boundary becomes

$$\log \frac{\pi_k}{\pi_\ell} - \frac{1}{2}(\mu_k+\mu_\ell)^T\Sigma^{-1}(\mu_k-\mu_\ell) + x^T\Sigma^{-1}(\mu_k-\mu_\ell) = 0$$

which is linear in x - in p dimensions a **hyper plane** like,

$$a + x^T b = 0$$

LDA result



LDA on the computer

The decision rule $G(x)$ assigns class with highest probability

$$G(x) = \arg\max_k \delta_k(x).$$

using discriminant functions ($P(G = k|X = x)$ with constants removed)

$$\delta_k(x) = \underline{x^T\Sigma^{-1}\mu_k} - \underline{\frac{1}{2}\mu_k^T\Sigma^{-1}\mu_k} + \underline{\log\pi_k}; \qquad k = 1, ..., K$$

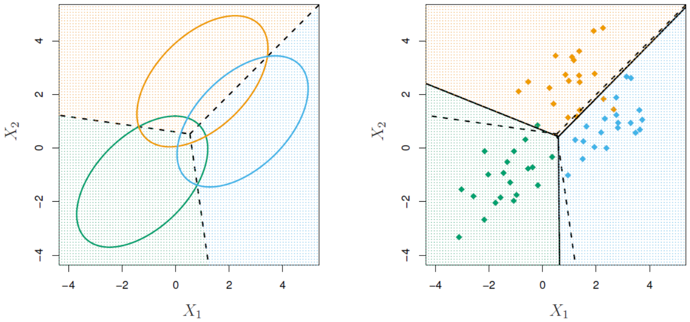
Use **plug-in estimates** for unknown parameters,

$$\hat{\pi}_k = N_k/N, \text{ where } N_k \text{ is number of class-k observations}$$

$$\hat{\mu}_k = \sum_{g_i=k} x_i/N_k$$

$$\hat{\Sigma} = \sum_{k=1}^{K}\sum_{g_i=k}(x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T/(N-K)$$

More than two classes



- One decision line for each pair of classes
- One discriminant function for each class
    - Assign class to highest probability.

Exercise - linear discriminant analysis

We have data with four different measures from flowers of three different species (FisherIris.csv). There are 50 observations of each species. Build a linear discriminant classifier for the three species.

Iris setosa

Iris versiocolor

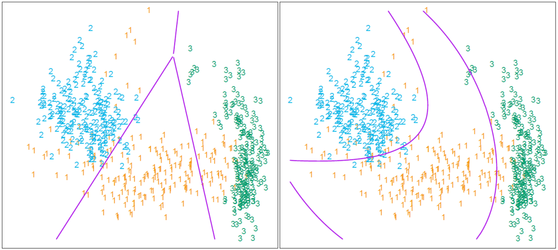
Iris verginica

- Calculate plug-in estimates $\hat{\pi}_k$, $\hat{\mu}_k$and $\hat{\Sigma}$.
- Calculate discriminant function δ_k.
- Predict class belongings for all observations in training data.
- Calculate confusion matrix for training data.

Quadratic discriminant analysis

Linear discriminant analysis assumes that the covariance structures are equal.

When we drop this restriction we get **quadratic discriminant analysis**, QDA, and the decision boundaries becomes non-linear.



Regularized discriminant analysis

It takes a lot of observations to estimate a large covariance matrix with precision. Three increasingly harsh regularizations are available

1. Make a compromise between LDA and QDA,

$$\hat{\Sigma}_k(\alpha) = \alpha\hat{\Sigma}_k + (1-\alpha)\hat{\Sigma}$$

2. Shrink the covariance towards its diagonal

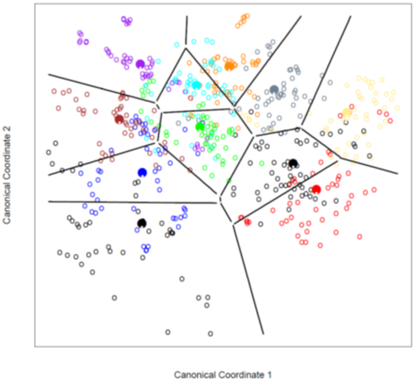
$$\hat{\Sigma}_k(\gamma) = \gamma\hat{\Sigma} + (1-\gamma)\,\mathrm{diag}(\hat{\Sigma})$$

3. Shrink the covariance towards a scalar covariance structure

$$\hat{\Sigma}_k(\gamma) = \gamma\hat{\Sigma} + (1-\gamma)\hat{\sigma}^2 I$$

Reduced rank discriminant analysis

Classification in a reduced subspace. In higher dimensional subspace, the decision boundaries are hyper-planes and can not be represented as lines. Hence, this technique is very **useful for illustrating class separation**.



We do the computations in the sub-space lecture.

LDA in Matlab

```
lda = fitcdiscr(X, y,'DiscrimType','Linear','Gamma',.5);

yhat = predict(lda,xnew);
```

Matlab supports shrinkage towards a common diagonal covariance matrix,

$$\hat{\Sigma}_k(\gamma) = \gamma\hat{\Sigma} + (1-\gamma)\,\mathrm{diag}(\hat{\Sigma})$$

Logistic regression

Linear classification

- Fewer assumptions than LDA
- More robust than LDA
- Just as easy!

LDA assumptions revisited

What made LDA linear?

- Equal covariance matrices
  - Unequal covariances lead to a quadratic discriminant analysis
- Classes have Gaussian distributions

Away with the assumptions

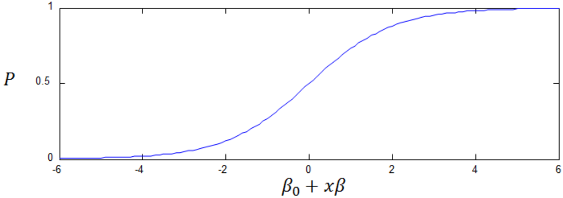
- Never mind about covariances and distributions!
- Optimize linear log-odds function directly
  - $\log \frac{P(G=red|X)}{P(G=blue|X)} = \beta_0 + x\beta$
- This is **logistic regression**
- What is a good choice of $\{\beta_0, \beta\}$?

Class probability

Derive expressions for the two class problem

- $P(G = \textcolor{red}{red} | X = x)$ = ?
- $P(G = \textcolor{blue}{blue} | X = x)$ = ?

when $\log \frac{P_r}{P_b} = \beta_0 + x\beta$



Likelihood

Combine this for all data points x_i

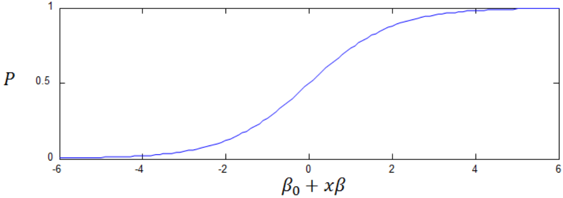
$$L(\beta_0, \beta) = \prod_{i=1}^{n} P(G = g_{x_i} | X = x_i)$$

Assuming independence, this is the **joint probability**

Logistic regression

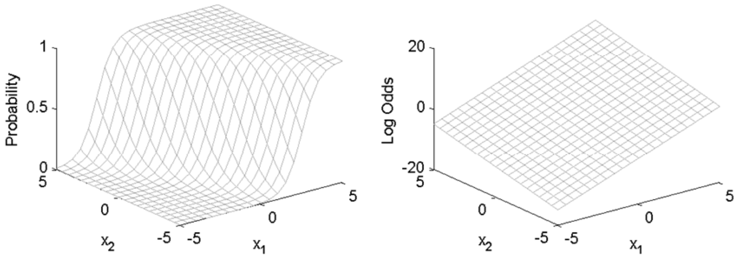
- Maximize the likelihood, L, wrt β_0 and β
  - $\arg\max_{\beta_0,\beta} L(\beta_0, \beta)$
  - Easier to maximize the log of $L(\beta_0, \beta)$
  - $l(\beta_0, \beta) = \log(L(\beta_0, \beta)) = \sum_i \mathbb{1}(x_i = \text{red})(\beta_0 + x_i\beta) - \log(1 + e^{\beta_0 + x_i\beta})$
- The approach is known as **maximum likelihood**
- The result is called **logistic regression**
- The maximization can be carried out using any method for numerical optimization
  - One algorithm uses an iteratively reweighted least squares solution

The Logistic Function

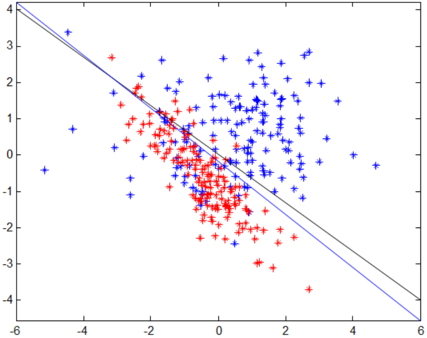


- $P(G = red|X = x) = \frac{e^{\beta_0 + x\beta}}{1+e^{\beta_0 + x\beta}}$
- Decision boundary: $P = 1/2 \rightarrow \beta_0 + x\beta = 0$
- Well inside $P \approx 1$, well outside $P \approx 0$
  - Outliers are handled gracefully
  - Logistic regression focuses on observations close to the boundary

The Logistic function in 2D



Logistic Regression vs LDA



Multiple Logistic Regression

K classes, $K = 1, 2, ..., K$

$\log \frac{P(G=1|X=x)}{P(G=K|X=x)} = \beta_{10} + x\beta_1$

$\log \frac{P(G=2|X=x)}{P(G=K|X=x)} = \beta_{20} + x\beta_2$

$\vdots$

$\log \frac{P(G=K-1|X=x)}{P(G=K|X=x)} = \beta_{(K-1)0} + x\beta_{K-1}$

Arbitrary which class we put in the denominator

Multiple logistic regression, cont'd

Since $P(G = K) = 1 - \sum_{i=1}^{K-1} P(G = i)$, we can show that

$$P(G = K|X = x) = \frac{1}{1 + \sum_{i=1}^{K-1} \exp(\beta_{i0} + x\beta_i)}$$

and then

$$P(G = k|X = x) = \frac{\exp(\beta_{k0} + x\beta_k)}{1 + \sum_{i=1}^{K-1} \exp(\beta_{i0} + x\beta_i)}$$

Hence, the class probabilities does not depend on the choice of denominator in the odds-ratios.

Why Logistic Regression?

- Statistics
  - Identify variables important for separating the classes
    - Biostatistics and epidemiology
- Classification
  - Predict class belonging of new observations
    - For example spam/email or diseased/healthy
- Risk prediction
  - Estimate probability (risk) for each class
    - Fraud detection in insurance claims

Interpreting the coefficients

- We have estimated β_0 and β
- What do they mean?
  - $\log \frac{P(G=red|X)}{P(G=blue|X)} = \beta_0 + x\beta$
  - They denote the log-odds contribution of each variable

Example: Model lung cancer (yes/no) as a function of smoking (number of cigarettes per day)

- $\beta = 0.02$
- A unit increase in smoking (one extra cigarette) means an increase in lung cancer risk (odds) of $\exp(0.02) \approx 1.02 = 2\%$

Regularized logistic regression

Few observations (low n) and high dimension (high p) data is a problem also for logistic regression.

One solution is an **elastic net** regularization of the likelihood,

$$[\beta, \beta_0] = \arg\max_{\beta_0,\beta} \{\log L(\beta, \beta_0) - P_{\lambda,\alpha}(\beta)\}$$
$$= \arg\max_{\beta_0,\beta} \left\{ \sum_{i=1}^{n} \left[y_i(\beta_0 + \beta^T x_i) - \log(1 + e^{1+\beta_0+\beta^T x_i}) \right] - P_{\lambda,\alpha}(\beta) \right\}$$

with

$$P_{\lambda,\alpha}(\beta) = \lambda \left(\frac{1}{2}(1-\alpha)||\beta||_2^2 + \alpha||\beta||_1 \right)$$

Use cross-validation for λ and α.

Why minus in front of $P_{\lambda,\alpha}(\beta)$?
Why is β_0 not regularized?

Logistic regression in Matlab

Standard logistic regression,

```
model = fitglm(X,Y,'linear','distr','binomial');

yhat = predict(model, Xnew);
```

and elastic net regularized version,

```
[B,FitInfo] = lassoglm(X,Y,'binomial','Alpha',0.5,'CV',10);

lassoPlot(B,FitInfo,'PlotType','CV');
```

Predictions can be calculated using `glmval`.

Properties

- Logistic regression is more robust than LDA
  - It relies on fewer assumptions
  - When is this a bad thing when compared to LDA?
- Logistic regression handles categorical variables better than LDA
- Observations far away from the boundary are down-weighted
  - You will have a look at how this works during the exercises
- Breaks down when classes are perfectly separable
- Easy to interpret and explain
- Surprisingly often hard to beat
- Can be combined with regularization of parameters ($n < p$)
- Can be generalized to multi-class problems

Basis Expansion

- General non-linear transforms
- Cubic splines

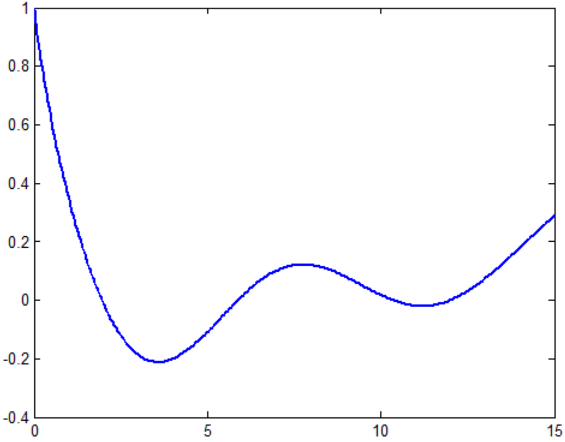
Basis expansion

- We are not limited to use our data as they are
- Linear models
  - Easy to interpret
  - First order Taylor expansion of non-linearities
  - Might be ok even for non-linear data if we have few observations
- Non-linear problem - transform data and use linear model
  - $h_m(X) = X_j^2$ and $h_m(X) = X_j X_k$
  - $h_m(X) = \log(X_j)$ or $h_m(X) = sqrt(X_j)$
  - $h_m(X) = \frac{X - m_X}{s_X}$ (always used when using regularization)
  - $h_m(X_{(i)}) = i$, sort data $X_{(1)} \leq X_{(2)} \leq ...$ and use the rank
  - Either replacing X with $h_m(X)$ or expanding $\{X, h_m(X)\}$

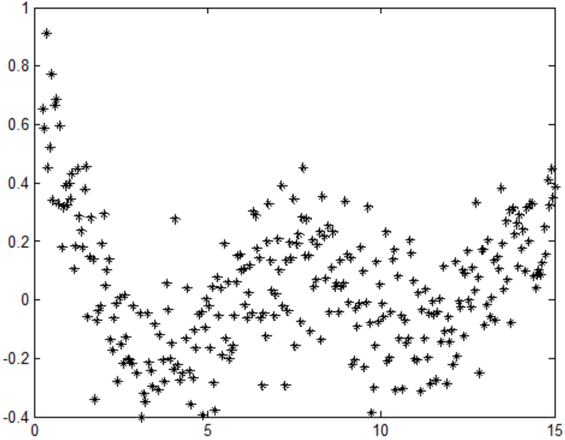
More advanced transforms

- Splines
  - We'll talk about that next
- Fourier/Wavelet transforms
  - Time series data/images
- Principal components
  - Projection along eigenvectors
  - We'll talk about that later in the course
- Moving averages
  - Possibly also delayed averages capturing time dynamics
  - Time series data

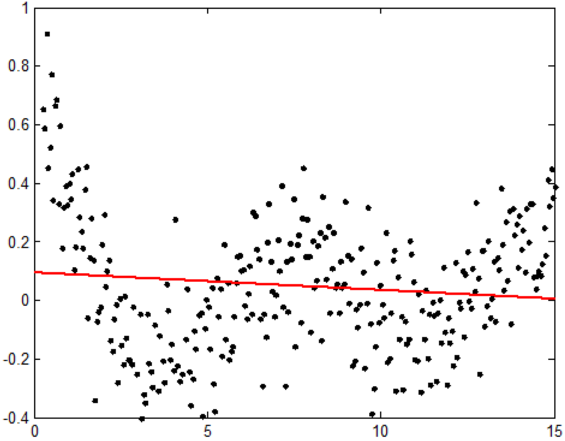
The Convenient Truth...



The Inconvenient Reality...



Ordinary Least Squares



Basis Expansions

Idea: replace variables (columns) of the data matrix, X, with transformations $h(X)$

The linear model

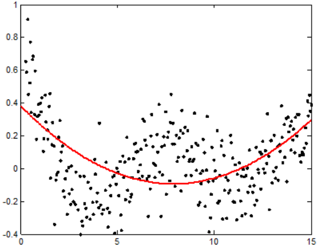
$$y = X\beta = \sum_{i=1}^{p} \beta_i x_i \rightarrow \sum_{i=1}^{M} \beta_i' h_i(X)$$

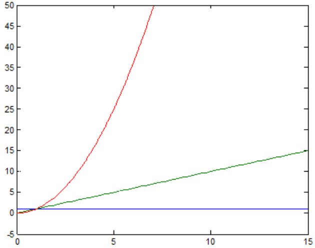
In this way we handle non-linear problems with our well known linear models

Basis Expansion

Example

```
X = [ones(n,1) x x.^2];
y = X*(X\y)
```

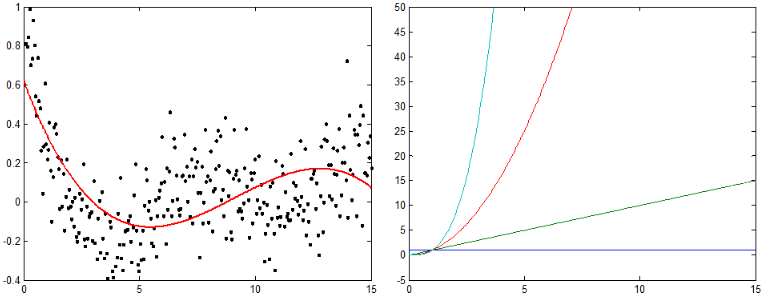




Basis Expansion

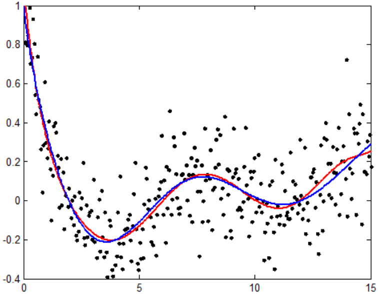
Third degree polynomial

```
beta = [0.6232, -0.3198, 0.0417, 0.0015]
```



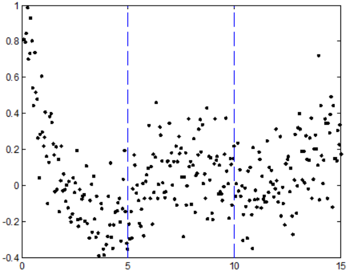
Basis Expansion

8th degree polynomial



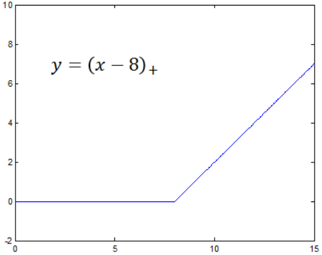
Piece-wise Basis Expansion

- To introduce flexibility while keeping the variance under control, we define **different basis functions for different intervals of x.**
- Example, divide the range of x in three parts



The hinge function

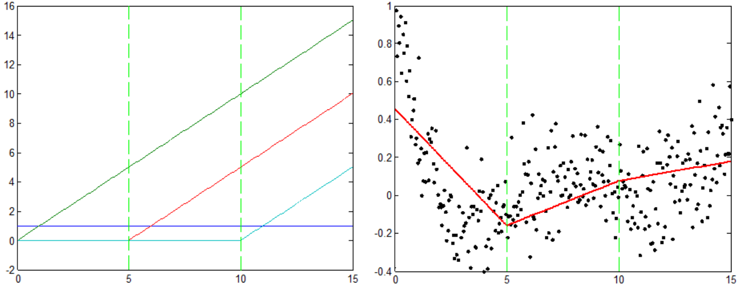
- Introducing the "**hinge**" function $y = (f(x))_+$
- Zero when $f(x)$ is less than zeros, otherwise $f(x)$
- In Matlab: e.g. `max(0,x)`
- Example
  `plot(x,max(0,x-8));`



Piece-wise Polynomials

Lets try three different linear functions in our three intervals

```
X = [ones(n,1) x max(0,x-5) max(0,x-10)];
```



Cubic Splines

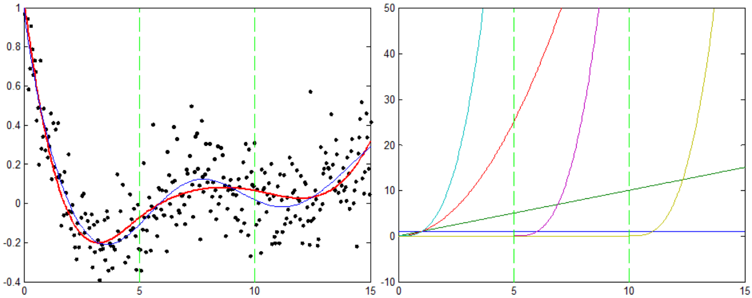
Splines are piece-wise polynomials

- The basis functions are

```
X = [ones(n,1) x x.^2 x.^3 max(0,x-5).^3 max(0,x-10).^3];
```

- Or in the proper way
  - $h_0(x) = 1$
  - $h_1(x) = x$
  - $h_2(x) = x^2$
  - $h_3(x) = x^3$
  - $h_4(x) = (x-5)_+^3$
  - $h_5(x) = (x-10)_+^3$
- Cubic splines have continuous first and second derivatives at the knots.
  - I.e a smooth function

Cubic Splines, cont'd



Notice that this non-linear function was obtained with a linear model

```
X = [ones(n,1) x x.^2 x.^3 max(0,x-5).^3 max(0,x-10).^3];
b = X\y;
```

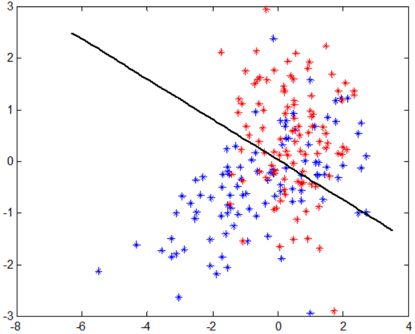
Spline approximation is

$$f(t) = b_0 + b_1 t + b_2 t^2 + b_3 t^3 + b_4 (t-5)_+^3 + b_5 (t-10)_+^3$$

```
t = 0:0.01:15;
T = [ones(n,1) t t.^2 t.^3 max(0,t-5).^3 max(0,t-10).^3];
plot(t,T*b,'r')
```

Application To Linear Discriminant Analysis

- Standard linear discriminant analysis, LDA
- Linear decision line



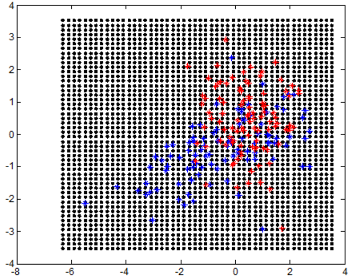
Application To Linear Discriminant Analysis

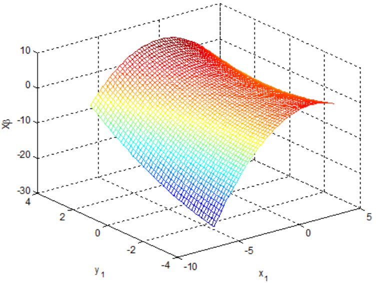
Let's try with basis expansions

```
load('synthetic_2D_ol_1.mat');
[n, p] = size(X);
Xe = [ X X(:,1).^2 X(:,1).*X(:,2) X(:,2).^2 ];

lda = fitcdiscr(Xe, y);
beta  = lda.Coeffs(1,2).Linear;
beta0 = lda.Coeffs(1,2).Const;

res=50;
[xx, yy] = meshgrid( linspace( -6, 3, 50), linspace( -3, 3, 50 ) );
xxe = [ xx(:), yy(:) xx(:).^2 xx(:).*yy(:) yy(:).^2 ];
f = xxe*beta + beta0;
ff = reshape(f, res, res);

figure(1)
plot(X(y==0,1), X(y==0,2), 'b*',X(y==1,1), X(y==1,2), 'r*');
hold on
[~, l1] = contour( xx,yy, ff, [0 0]);
set(l1, 'LineColor', 'k', 'LineStyle', '-', 'LineWidth', 2);
hold off
```

Application To Linear Discriminant Analysis

To plot the boundary we must classify a fine grid of points and find those that are on or near the boundary





Application To Linear Discriminant Analysis

Calculate the grid

```
load('synthetic_2D_ol_1.mat');
[n, p] = size(X);
Xe = [ X X(:,1).^2 X(:,1).*X(:,2) X(:,2).^2 ];

lda = fitcdiscr(Xe, y);
beta  = lda.Coeffs(1,2).Linear;
beta0 = lda.Coeffs(1,2).Const;

res=50;
[xx, yy] = meshgrid( linspace( -6, 3, 50), linspace( -3, 3, 50 ) );
xxe = [ xx(:), yy(:) xx(:).^2 xx(:).*yy(:) yy(:).^2 ];
f = xxe*beta + beta0;
ff = reshape(f, res, res);

figure(1)
plot(X(y==0,1), X(y==0,2), 'b*',X(y==1,1), X(y==1,2), 'r*');
hold on
[~, l1] = contour( xx,yy, ff, [0 0]);
set(l1, 'LineColor', 'k', 'LineStyle', '-', 'LineWidth', 2);
hold off
```

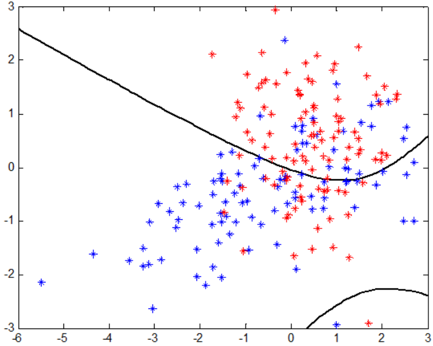
Application To Linear Discriminant Analysis

Contour plot, $x\beta^T + \beta_0 = 0$

```
load('synthetic_2D_ol_1.mat');
[n, p] = size(X);
Xe = [ X X(:,1).^2 X(:,1).*X(:,2) X(:,2).^2 ];

lda = fitcdiscr(Xe, y);
beta  = lda.Coeffs(1,2).Linear;
beta0 = lda.Coeffs(1,2).Const;

res=50;
[xx, yy] = meshgrid( linspace( -6, 3, 50), linspace( -3, 3, 50 ) );
xxe = [ xx(:), yy(:) xx(:).^2 xx(:).*yy(:) yy(:).^2 ];
f = xxe*beta + beta0;
ff = reshape(f, res, res);

figure(1)
plot(X(y==0,1), X(y==0,2), 'b*',X(y==1,1), X(y==1,2), 'r*');
hold on
[~, l1] = contour( xx,yy, ff, [0 0]);
set(l1, 'LineColor', 'k', 'LineStyle', '-', 'LineWidth', 2);
hold off
```

Application To Linear Discriminant Analysis

The decision boundary became non-linear in *x*, despite a linear classifier



Logistic regression with splines

The linear log-odds model is replaced with a flexible spline function

$$\log \frac{P(G=0|X=x)}{P(G=1|X=x)} = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 (x-2)^3_+ + \beta_5 (x-5)^3_+$$

- Non-linear in x, linear in β
  - Standard logistic regression problem after basis expansion
- Easy to interpret
- Gives probability for class belongings

Summary

- Linear methods are nice!
  - But natural processes are often non-linear
- Basis expansion opens for **non-linear modeling of data using linear methods**.
  - Data is getting more high-dimensional
  - Model selection is critical
- Splines proved flexibility with few parameters to tune

Questions?