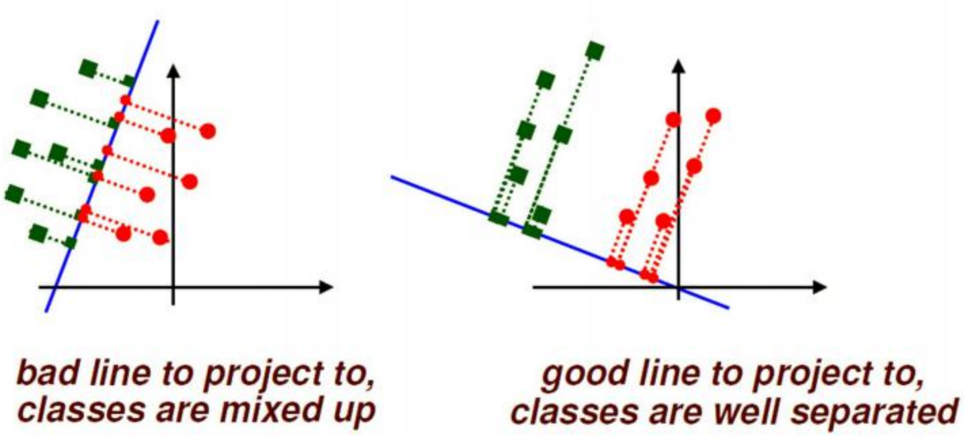
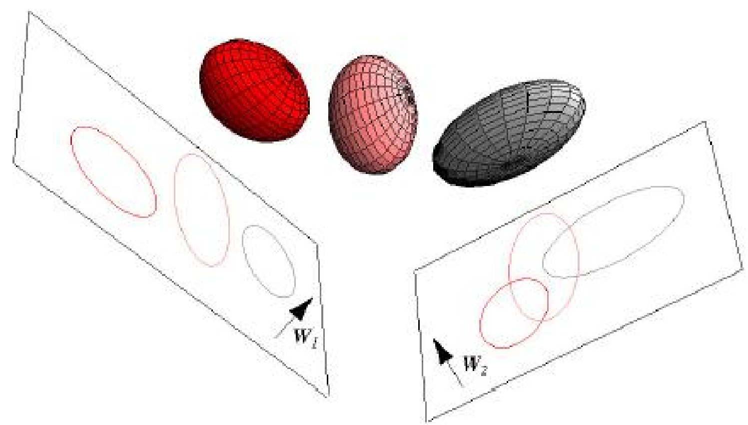
Rensselaer

Lecture 9: Linear Discriminant Functions (1)

Dr. Chengjiang Long
Computer Vision Researcher at Kitware Inc.
Adjunct Professor at RPI.
Email: **longc3@rpi.edu**

Recap Previous Lecture





Outline

- Generative vs Discriminant Approach
- Linear Discriminant Function and Decision Surface
- Linear Separability
- Learning with Gradient Decent and Netwon's Method

Outline

- **Generative vs Discriminant Approach**
- Linear Discriminant Function and Decision Surface
- Linear Separability
- Learning with Gradient Decent and Netwon's Method

Generative vs Discriminant Approach

- **Generative** approaches estimate the discriminant function by first estimating the probability distribution of the patterns belonging to each class.

- **Discriminant** approaches estimate the discriminant function explicitly, without assuming a probability distribution.

Generative Approach (two categories)

- More common to use a single discriminant function (*dichotomizer*) instead of two:

$$g(\mathbf{x}) = g_1(\mathbf{x}) - g_2(\mathbf{x})$$

Decide ω_1 if $g(\mathbf{x}) > 0$; otherwise decide ω_2

$$g(\mathbf{x}) = P(\omega_1 / \mathbf{x}) - P(\omega_2 / \mathbf{x})$$

$$g(\mathbf{x}) = \ln \frac{p(\mathbf{x} / \omega_1)}{p(\mathbf{x} / \omega_2)} + \ln \frac{P(\omega_1)}{P(\omega_2)}$$

If $g(\mathbf{x})=0$, then **x** lies on the decision boundary and can be assigned to either class.

Generative Approach

- Advantage

  – Prior information about the structure of the data is often most naturally specified through a generative model $P(X|Y)$

  For example, for male faces, we would expect to see heavier eyebrows, a more square jaw, etc.

- Disadvantages

  – The generative approach does not directly target the classification model $P(Y|X)$ since the goal of generative training is $P(X|Y)$

  – If the data x are complex, finding a suitable generative data model $P(X|Y)$ is a difficult task

  – Since each generative model is separately trained for each class, there is no competition amongst the models to explain the data

  – The decision boundary between the classes may have a simple form, even if the data distribution of each class is complex

Discriminant Approach (two categories)

- Specify parametric form of the discriminant function, for example, a linear discriminant:

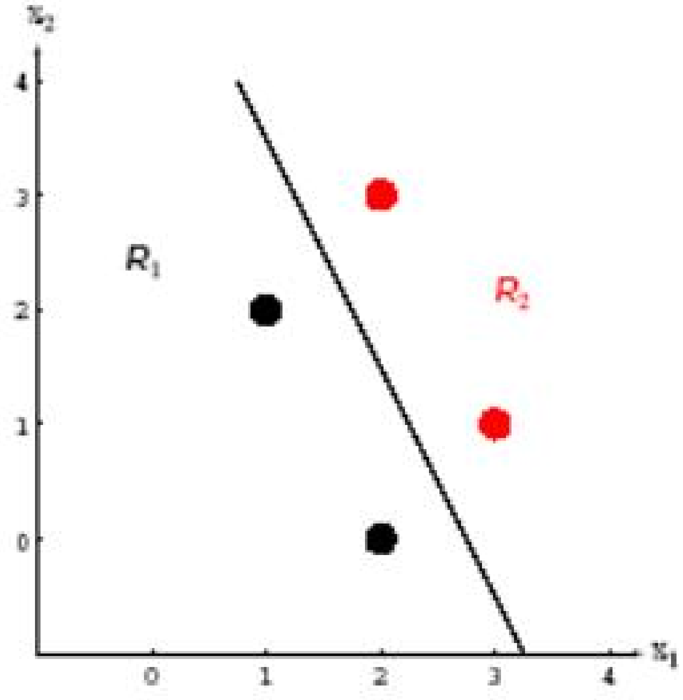
$$g(\mathbf{x}) = \mathbf{w}^t\mathbf{x} + w_0 = \sum_{i=1}^{d} w_i x_i + w_0$$

Decide w_1 if $g(\mathbf{x}) > 0$ and w_2 if $g(\mathbf{x}) < 0$

If $g(\mathbf{x})=0$, then **x** lies on the decision boundary and can be assigned to either class.

Discriminant Approach (cont'd)

- Find the "best" decision boundary (i.e., estimate **w** and w_0) using a set of training examples $\mathbf{x}_k$.



Discriminant Approach (cont'd)

- The solution can be found by minimizing an error function (e.g., "training error" or "empirical risk"):

class labels:

$$J(\mathbf{w}, w_0) = \frac{1}{n}\sum_{k=1}^{n}[z_k - g(\mathbf{x}_k)]^2 \qquad z_k = \begin{cases} +1 \ if \ \mathbf{x}_k \in \omega_1 \\ -1 \ if \ \mathbf{x}_k \in \omega_2 \end{cases}$$

correct class　　　predicted class

- "Learning" algorithms can be applied to find the solution.

Linear Discriminants (two categories)

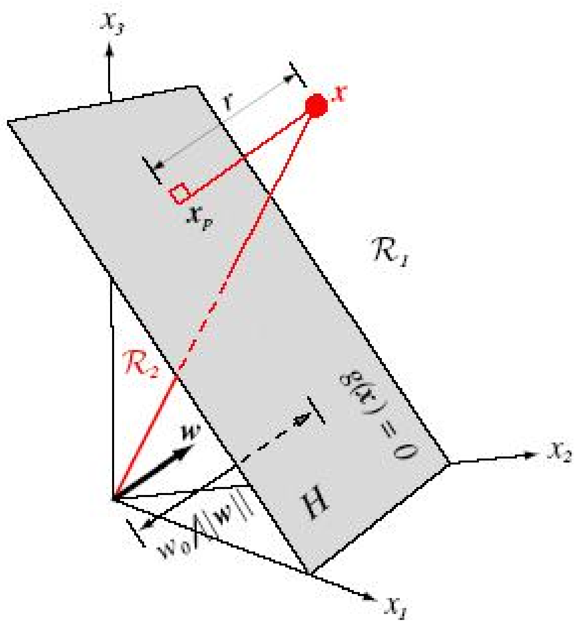
- A linear discriminant has the following form:

$$g(\mathbf{x}) = \mathbf{w}^t\mathbf{x} + w_0 = \sum_{i=1}^{d} w_i x_i + w_0$$

- The decision boundary $(g(\mathbf{x})=0)$, is a **hyperplane** where the orientation of the hyperplane is determined by **w** and its location by w_0.
  - **w** is the normal to the hyperplane
  - If $w_0=0$, the hyperplane passes through the origin

Geometric Interpretation of g(x)

- *g(***x***)* provides an algebraic measure of the distance of **x** from the hyperplane.



x can be expressed as follows:

$$\mathbf{x} = \mathbf{x}_p + r\frac{\mathbf{w}}{||\mathbf{w}||}$$

(direction of *r*)

Geometric Interpretation of g(x) (cont'd)

- Substitute **x** in *g(***x***)*:

$$g(\mathbf{x}) = \mathbf{w}^t\mathbf{x} + w_0 = \mathbf{w}^t(\mathbf{x}_p + r\frac{\mathbf{w}}{\|\mathbf{w}\|}) + w_0 =$$

$$= \mathbf{w}^t\mathbf{x}_p + r\frac{\mathbf{w}^t\mathbf{w}}{\|\mathbf{w}\|} + w_0 = r\|\mathbf{w}\|$$

Where

$$\mathbf{w}^t\mathbf{w} = \|\mathbf{w}\|^2$$

$$\mathbf{w}^t\mathbf{x}_p + w_0 = 0$$

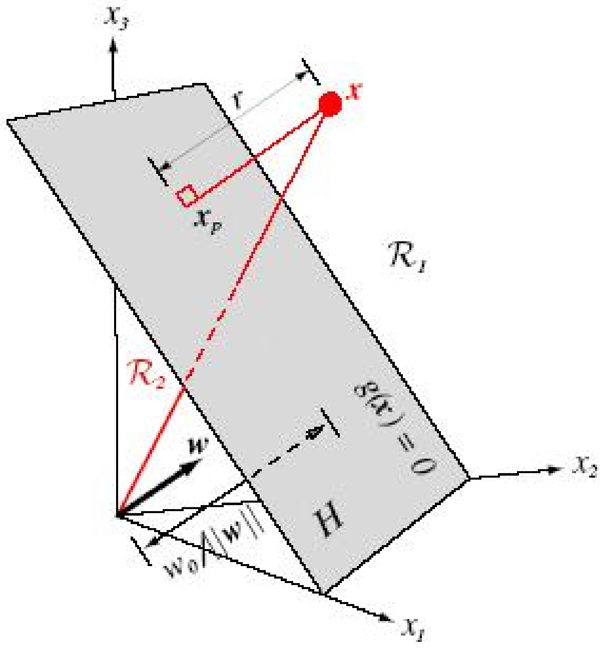
Geometric Interpretation of g(x) (cont'd)

- The distance of **x** from the hyperplane is given by:

$$r = \frac{g(\mathbf{x})}{\|\mathbf{w}\|}$$

Setting x=0, we get:

$$r = \frac{w_0}{\|\mathbf{w}\|}$$



Discriminative Approach

- Advantages
  - The discriminative approach directly addresses finding an accurate classifier $P(Y|X)$ based on modelling the decision boundary, as opposed to the class conditional data distribution
  - Whilst the data from each class may be distributed in a complex way, it could be that the decision boundary between them is relatively easy to model
- Disadvantages
  - Discriminative approaches are usually trained as “blackbox” classifiers, with little prior knowledge built used to describe how data for a given class is distributed
  - Domain knowledge is often more easily expressed using the generative framework

Outline

- Generative vs Discriminant Approach
- **Linear Discriminant Function and Decision Surface**
- Linear Separability
- Learning with Gradient Decent and Netwon's Method

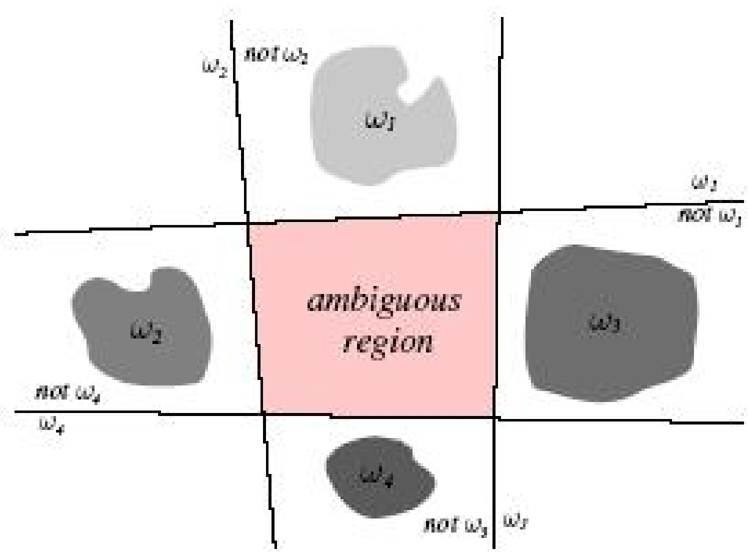
Linear Discriminant Functions: (Multi-category case)

- There are several ways to devise multi/ category classifiers using linear discriminant functions:

  (1): One against the rest

problem:

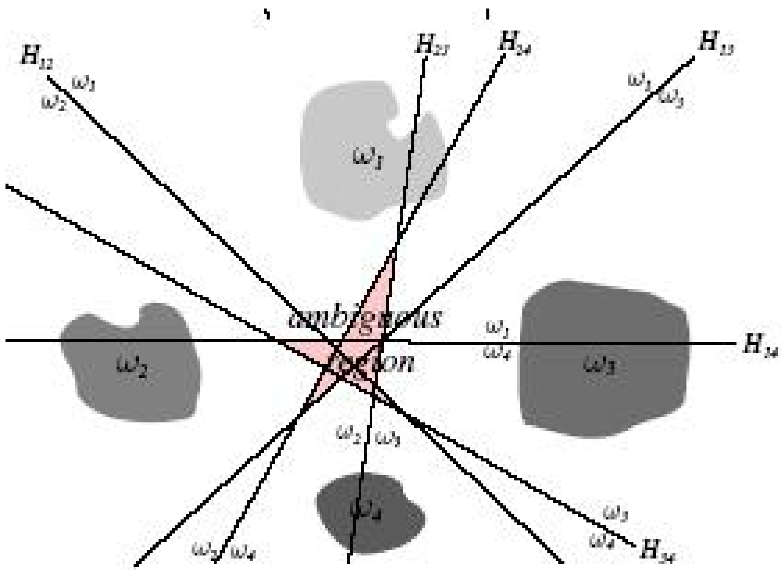
ambiguous regions



Linear Discriminant Functions: (Multi-category case) (cont'd)

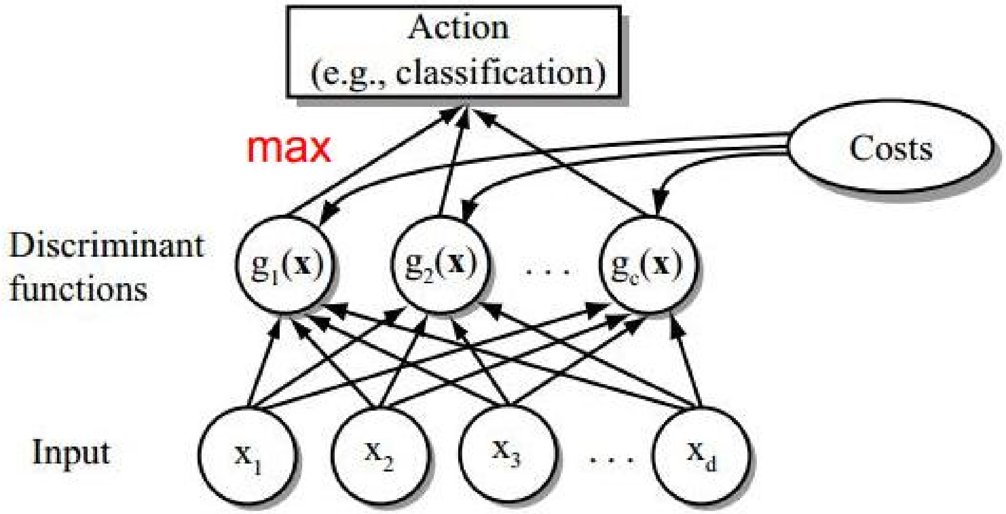
- There are several ways to devise multi–category classifiers using linear discriminant functions:
  (2): One against another (i.e., *c(c-1)/2* pairs of classes)

problem:
ambiguous regions



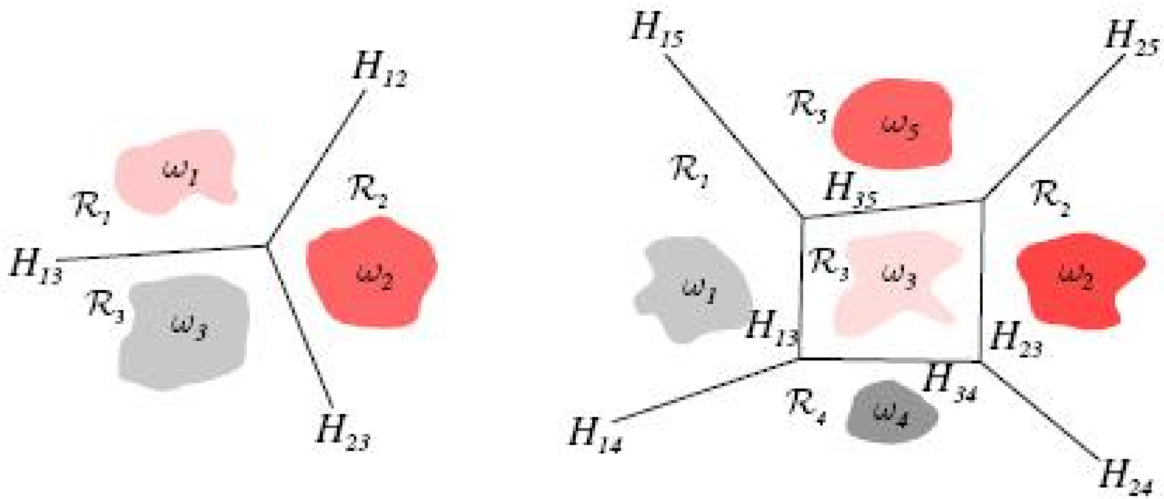
Linear Discriminant Functions: (Multi-category case) (cont'd)

- To avoid the problem of ambiguous regions:
  - Define c linear discriminant functions
  - Assign $\mathbf{x}$ to w_i if $g_i(\mathbf{x}) > g_j(\mathbf{x})$ for all $j \neq i$.
- The resulting classifier is called a linear machine



Linear Discriminant Functions: (Multi-category case) (cont'd)

- A linear machine divides the feature space in c convex decisions regions.
  - If **x** is in region R_i, the $g_i(\mathbf{x})$ is the largest.



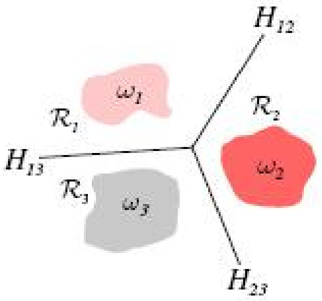
Note: although there are $c(c-1)/2$ pairs of regions, there are typically less decision boundaries

Linear Discriminant Functions: (Multi-category case) (cont'd)

- The decision boundary between adjacent regions R_i and R_j is a portion of the hyperplane H_{ij} given by:

$$g_i(\mathbf{x}) = g_j(\mathbf{x}) \quad or \quad g_i(\mathbf{x}) - g_j(\mathbf{x}) = 0$$

$$or \quad (\mathbf{w}_i - \mathbf{w}_j)^t \mathbf{x} + (w_{i0} - w_{j0}) = 0$$



- $(\mathbf{w}_i - \mathbf{w}_j)$ is normal to H_{ij} and the signed distance from $\mathbf{x}$ to H_{ij} is

$$r = \frac{g_i(\mathbf{x}) - g_j(\mathbf{x})}{\| \mathbf{w}_i - \mathbf{w}_j \|}$$

Higher Order Discriminant Functions

- Can produce more complicated decision boundaries than linear discriminant functions.

Quadratic discriminant: obtained by adding terms corresponding to products of pairs of components of x

$$g(\mathbf{x}) = w_0 + \sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d} \sum_{j=1}^{d} x_i x_j w_{ij}$$

Polynomial discriminant: obtained by adding terms such as $x_i x_j x_k w_{ijk}$.

Linear Discriminants

- Augmented feature/parameter space

$$g(\mathbf{x}) = \mathbf{w}^t\mathbf{x} + w_0 = \sum_{i=1}^{d} w_i x_i + x_0 w_0 = \sum_{i=0}^{d} w_i x_i = \boldsymbol{\alpha}^t\mathbf{y}$$

$$\mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ \cdots \\ w_d \end{bmatrix} \Rightarrow \boldsymbol{\alpha} = \begin{bmatrix} w_0 \\ w_1 \\ \cdots \\ w_d \end{bmatrix} \qquad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_d \end{bmatrix} \Rightarrow \mathbf{y} = \begin{bmatrix} x_0 \\ x_1 \\ \cdots \\ x_d \end{bmatrix}$$

Discriminant:

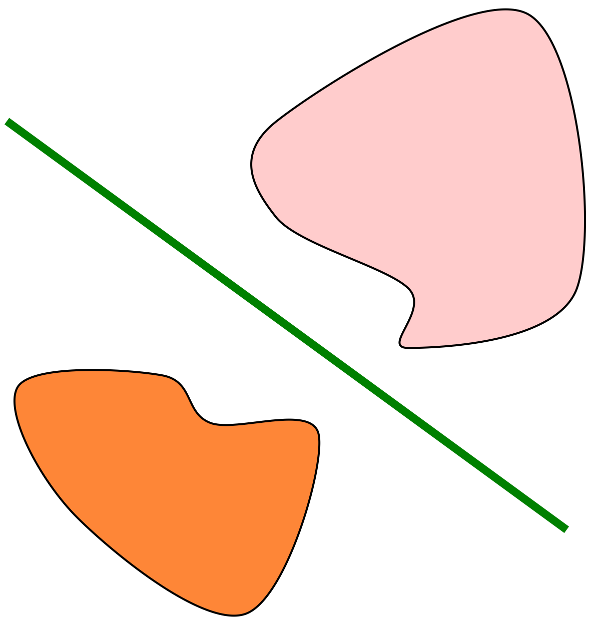
$$g(\mathbf{x}) = \boldsymbol{\alpha}^t\mathbf{y}$$

If	$\boldsymbol{\alpha}^t\mathbf{y}_i \geq 0$	assign $\mathbf{y}_i$ to ω_1
else if	$\boldsymbol{\alpha}^t\mathbf{y}_i < 0$	assign $\mathbf{y}_i$ to ω_2

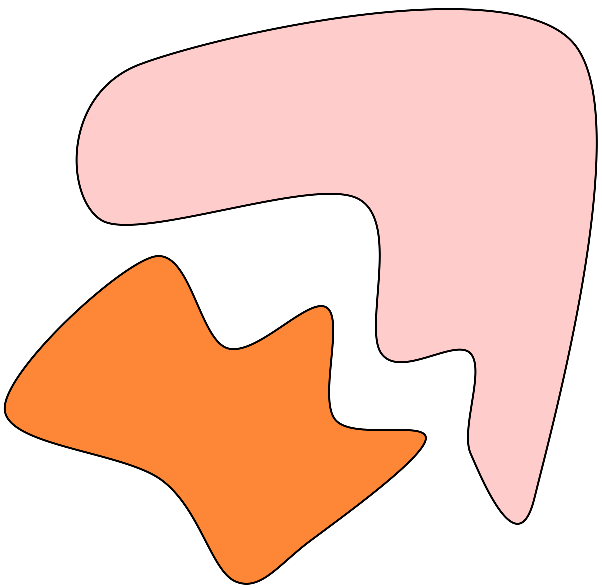
Outline

- Generative vs Discriminant Approach
- Linear Discriminant Function and Decision Surface
- **Linear Separability**
- Learning with Gradient Decent and Netwon's Method

The Two-Category Case

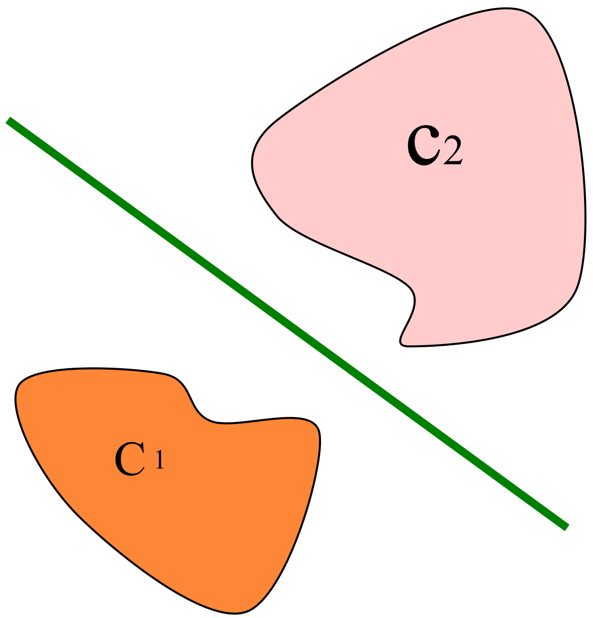


Linearly Separable



Not Linearly Separable

The Two-Category Case



How to find **a**?

Given a set of samples $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n$, some labeled c_1 and some labeled c_2,

If there exists a vector **a** such that

$$\mathbf{a}^T \mathbf{y}_i > 0 \quad \text{if } \mathbf{y}_i \text{ is labeled } c_1$$

$$\mathbf{a}^T \mathbf{y}_i < 0 \quad \text{if } \mathbf{y}_i \text{ is labeled } c_2$$

then the samples are said to be

Linearly Separable

Normalization

Withdrawing all labels of samples and replacing the ones labeled c_2 by their *negatives*, it is equivalent to find a vector $\mathbf{a}$ such that

$$\mathbf{a}^T \mathbf{y}_i > 0 \qquad \forall i$$

Given a set of samples $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n$, some labeled c_1 and some labeled c_2,

if there exists a vector $\mathbf{a}$ such that

$$\mathbf{a}^T \mathbf{y}_i > 0 \qquad \text{if } \mathbf{y}_i \text{ is labeled } c_1$$

$$\mathbf{a}^T \mathbf{y}_i < 0 \qquad \text{if } \mathbf{y}_i \text{ is labeled } c_2$$

then the samples are said to be

Linearly Separable

Generalized Discriminants

- First, map the data to a space of higher dimensionality.
  - Non–linearly separable → linearly separable

- This can be accomplished using special transformation functions (ϕ functions):
  - Map a point from a d–dimensional space to a point in a $\hat{d}$ –dimensional

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_d \end{bmatrix} \xrightarrow{\varphi} \begin{bmatrix} y_1(\mathbf{x}) \\ y_2(\mathbf{x}) \\ \cdots \\ y_{\hat{d}}(\mathbf{x}) \end{bmatrix}$$

Generalized Discriminants

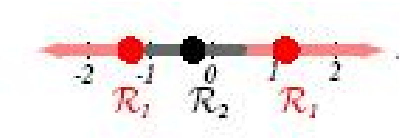
- A generalized discriminant is a linear discriminant in the $\hat{d}$−dimensional space:

$$g(\mathbf{x}) = \sum_{i=1}^{\hat{d}} a_i y_i(\mathbf{x}) \quad or \quad g(\mathbf{x}) = \boldsymbol{\alpha}^t \mathbf{y}$$

- Separates points in the $\hat{d}$ space by a hyperplane passing through the origin.

Example

- The corresponding decision regions R_1, R_2 in the d–space are **not** simply connected (not linearly separable).



- Consider the following mapping functions:

ϕ functions

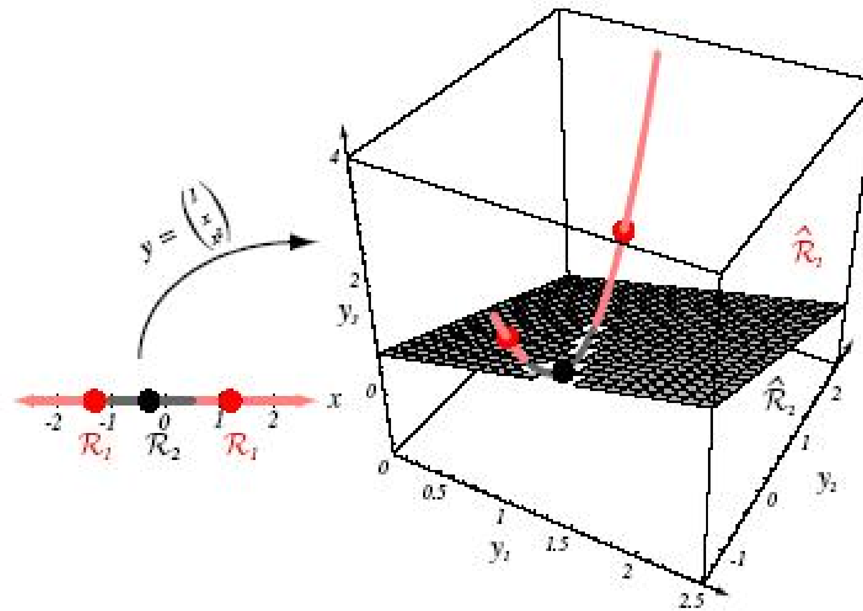
$$y = \begin{bmatrix} y_1(x) \\ y_2(x) \\ y_3(x) \end{bmatrix} = \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}$$

Discriminant:

$$\boldsymbol{\alpha} = \begin{bmatrix} -1 \\ 1 \\ 2 \end{bmatrix} \qquad g(x) = -1 + x + 2x^2$$

Example

g(**x**) maps a line in *d* space to a parabola in $\hat{d}$ space.



$$g(x) = -1 + x + 2x^2$$

The problem has now become linearly separable!

The plane $\boldsymbol{\alpha}^t y = 0$ divides the $\hat{d}$-space in two decision regions: $\hat{R}_1, \hat{R}_2$

Outline

- Generative vs Discriminant Approach
- Linear Discriminant Function and Decision Surface
- Linear Separability
- **Learning with Gradient Decent and Netwon's Method**

Learning Algorithms

- To design a learning algorithm, we face the following problems:

① Whether to stop？

② In what direction to proceed？

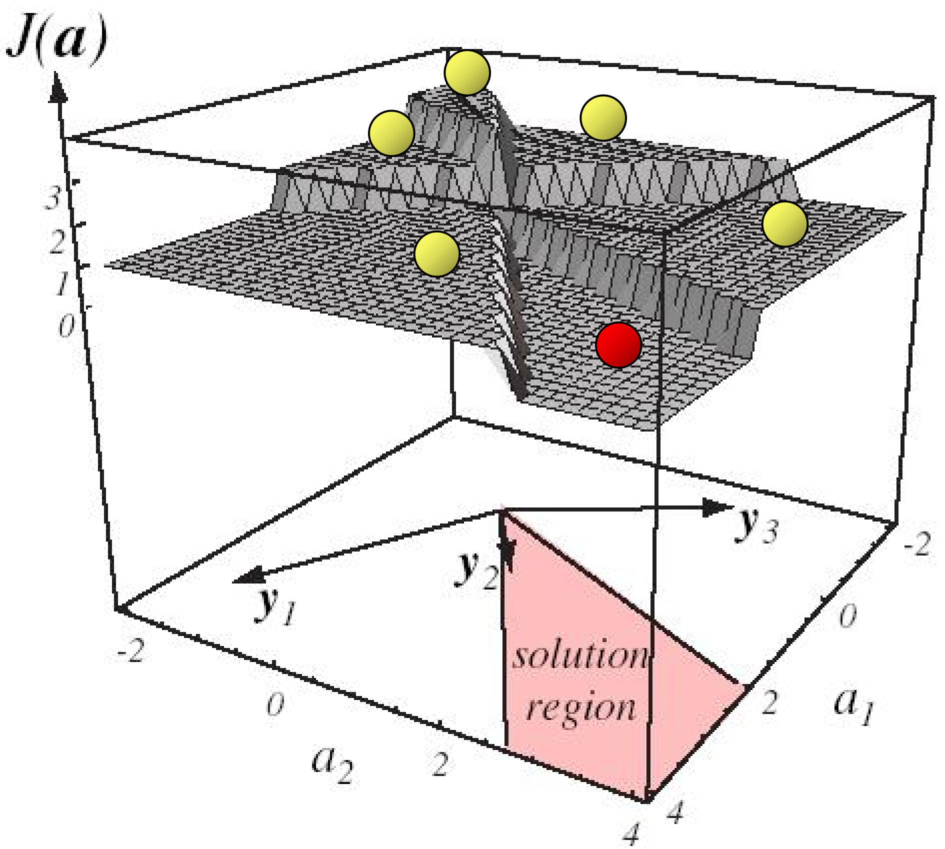
③ How long a step to take？

Is the criterion satisfactory?

Criterion Function

- To facilitate learning, we usually define a scalar *criterion function*.
- It usually represents the *penalty* or *cost* of a solution.
- Our goal is to *minimize* its value, i.e., *Function optimization*.

Criterion Functions: The Two-Category Case



$J(\mathbf{a})$

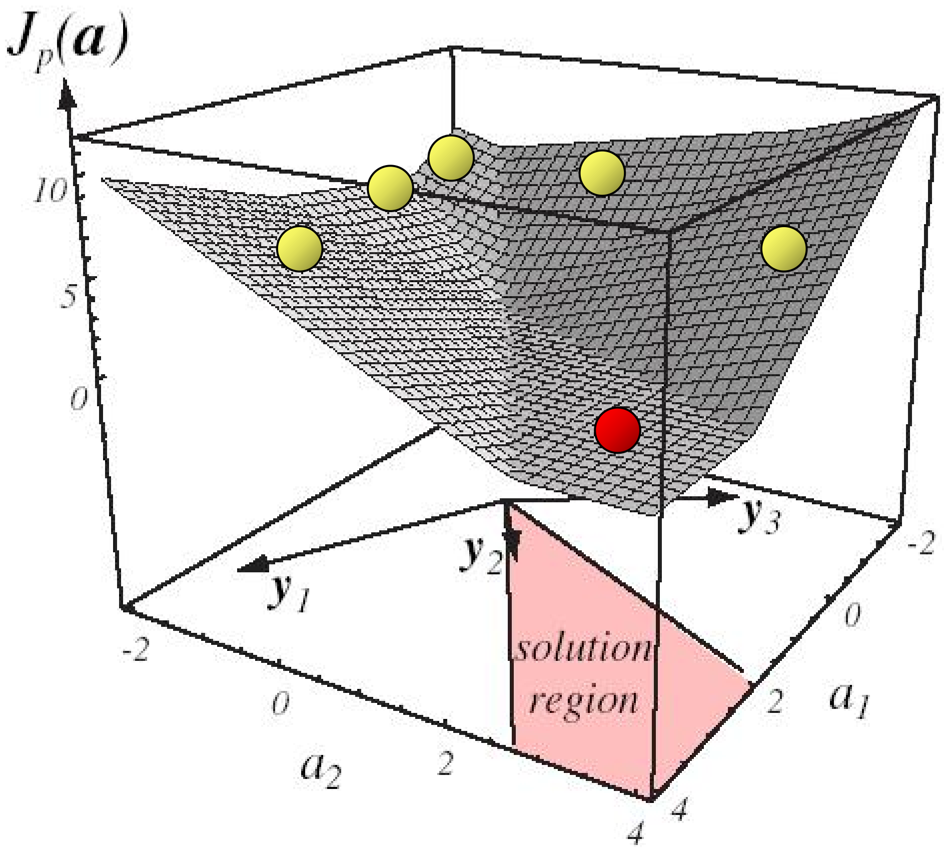
# of misclassified patterns

- solution state
- where to go?

Criterion Functions: The Two-Category Case

$$J_p(\mathbf{a}) = \sum_{\mathbf{y} \in \mathcal{Y}} (-\mathbf{a}^T \mathbf{y})$$

$\mathcal{Y}$: the set of misclassified patterns



Perceptron Criterion Function

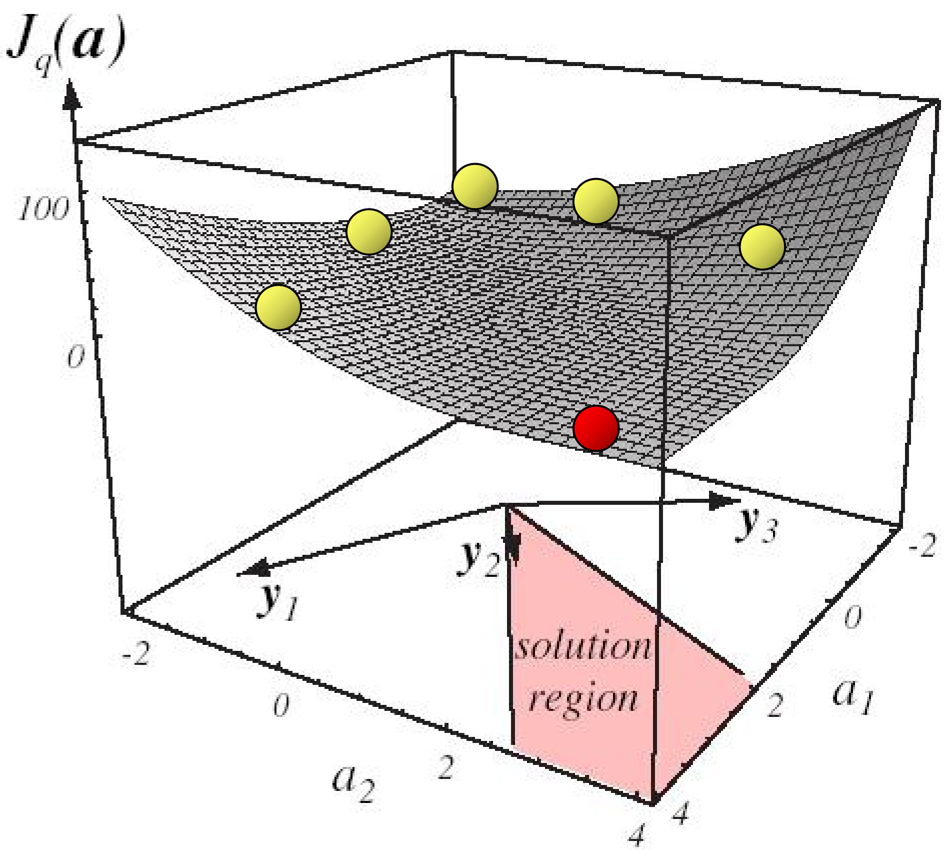
- solution state
- where to go?

What problem it has?

Criterion Functions: The Two-Category Case

$$J_q(\mathbf{a}) = \sum_{\mathbf{y} \in \mathcal{Y}} (\mathbf{a}^T \mathbf{y})^2$$

Y : the set of misclassified patterns



A Relative of Perceptron Criterion Function

- solution state
- where to go?

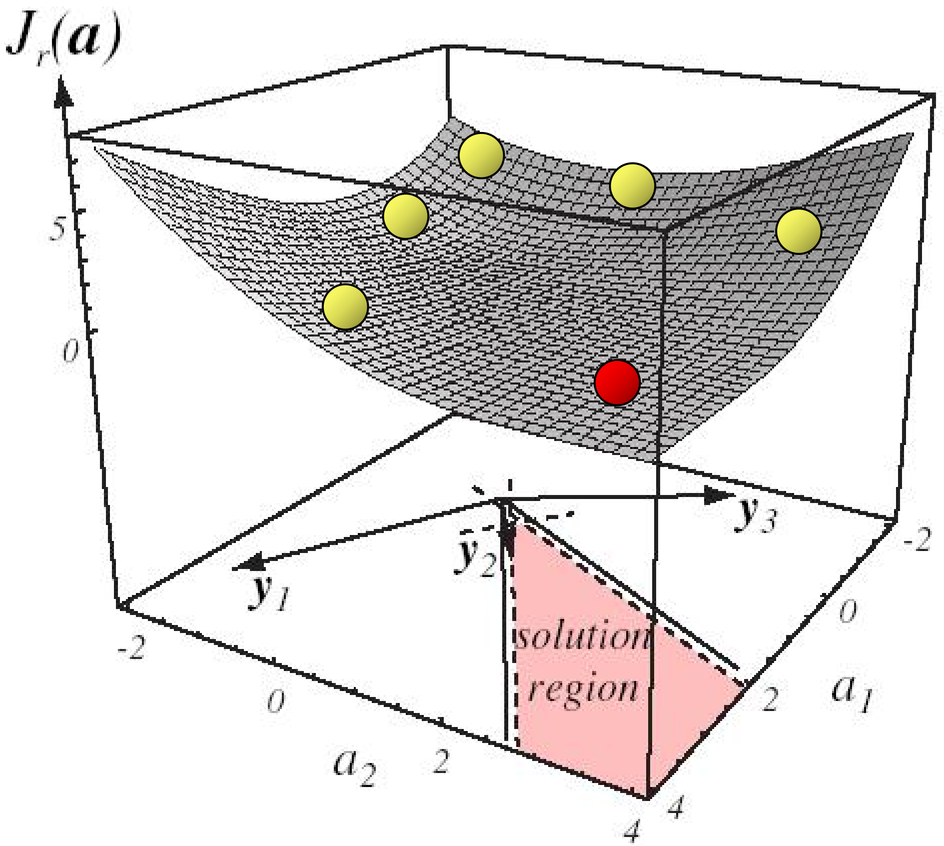
Is this criterion much better?

What problem it has?

Criterion Functions: The Two-Category Case

$$J_r(\mathbf{a}) = \sum_{\mathbf{y} \in \mathcal{Y}} \frac{(\mathbf{a}^T \mathbf{y} - b)^2}{\|\mathbf{y}\|^2}$$

$\mathcal{Y}$: the set of misclassified patterns



What is the difference with the previous one?

- solution state
- where to go?

Is this criterion good enough?

Are there others?

Learning: linearly separable case (two categories)

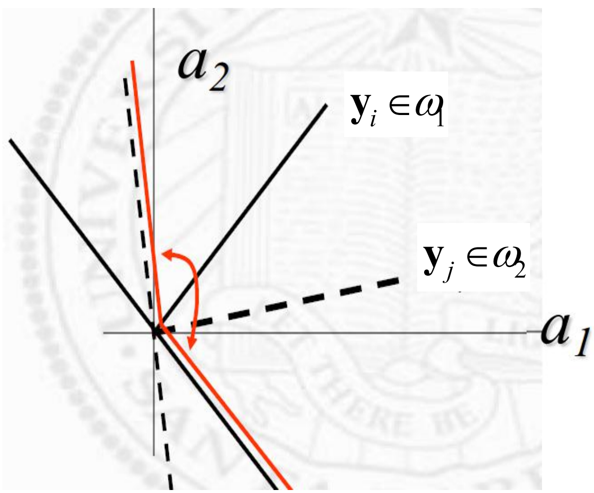
- Given a linear discriminant function

$$g(\mathbf{x})=\boldsymbol{\alpha}^t\mathbf{y}$$

The goal is to **"learn"** the parameters (weights) $\boldsymbol{\alpha}$ from a set of ***n*** labeled samples $\mathbf{y}_i$, where each $\mathbf{y}_i$ has a class label ω_1 or ω_2

Learning: effect of training examples

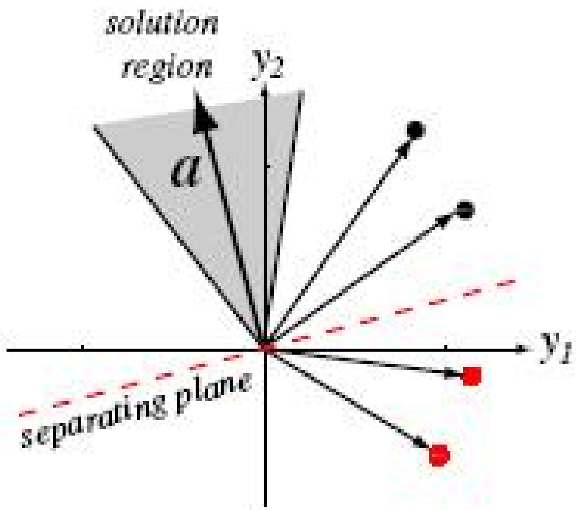
- Every training sample $\mathbf{y}_i$ places a constraint on the weight vector $\boldsymbol{\alpha}$; let's see how.
- **Case 1:** visualize solution in "parameter space":
  - $\boldsymbol{\alpha}^t\mathbf{y}=0$ defines a hyperplane in the parameter space with $\mathbf{y}$ being the normal vector.
  - Given n examples, the solution $\boldsymbol{\alpha}$ must lie on the intersection of n half spaces.



parameter space (α_1, α_2)

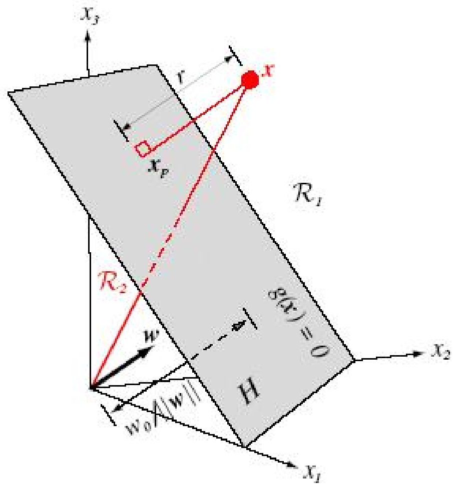
Learning: effect of training examples

- **Case 2**: visualize solution in "feature space":
  - $\alpha^t \mathbf{y}=0$ defines a hyperplane in the feature space with α being the normal vector.
  - Given *n* examples, the solution α must lie within a certain region.



Uniqueness of Solution

- Solution vector $\boldsymbol{\alpha}$ is usually not unique; we can impose certain constraints to enforce uniqueness, for example:
  - "Find unit-length weight vector that maximizes the minimum distance from the training examples to the separating plane"

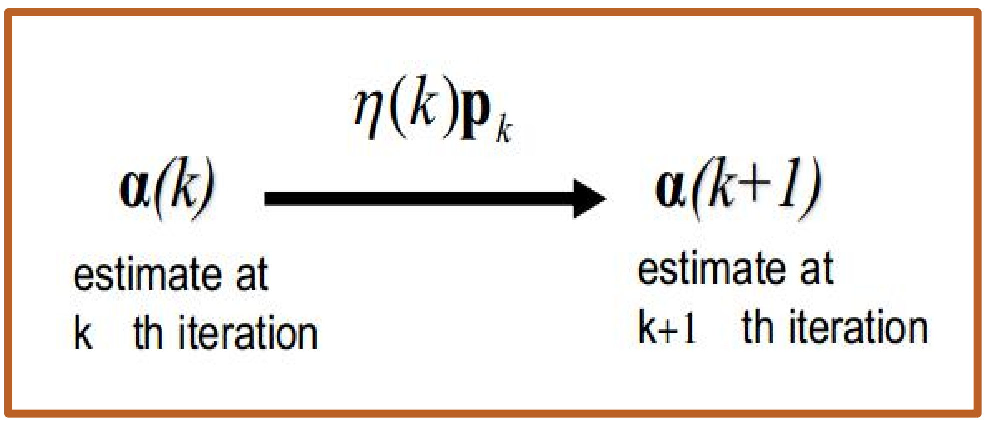


Learning Using Iterative Optimization

- Minimize an error function $J(\boldsymbol{\alpha})$ (e.g., classification error) with respect to $\boldsymbol{\alpha}$:
- Minimize $J(\boldsymbol{\alpha})$ iteratively: $\boldsymbol{\alpha}(k+1) = \boldsymbol{\alpha}(k) + \eta(k)\mathbf{p}_k$

learning rate

search direction



How should we choose $\mathbf{p}_k$?

Choosing p_k using Gradient Descent

$$\mathbf{p}_k = -\nabla J(\boldsymbol{\alpha}(k))$$

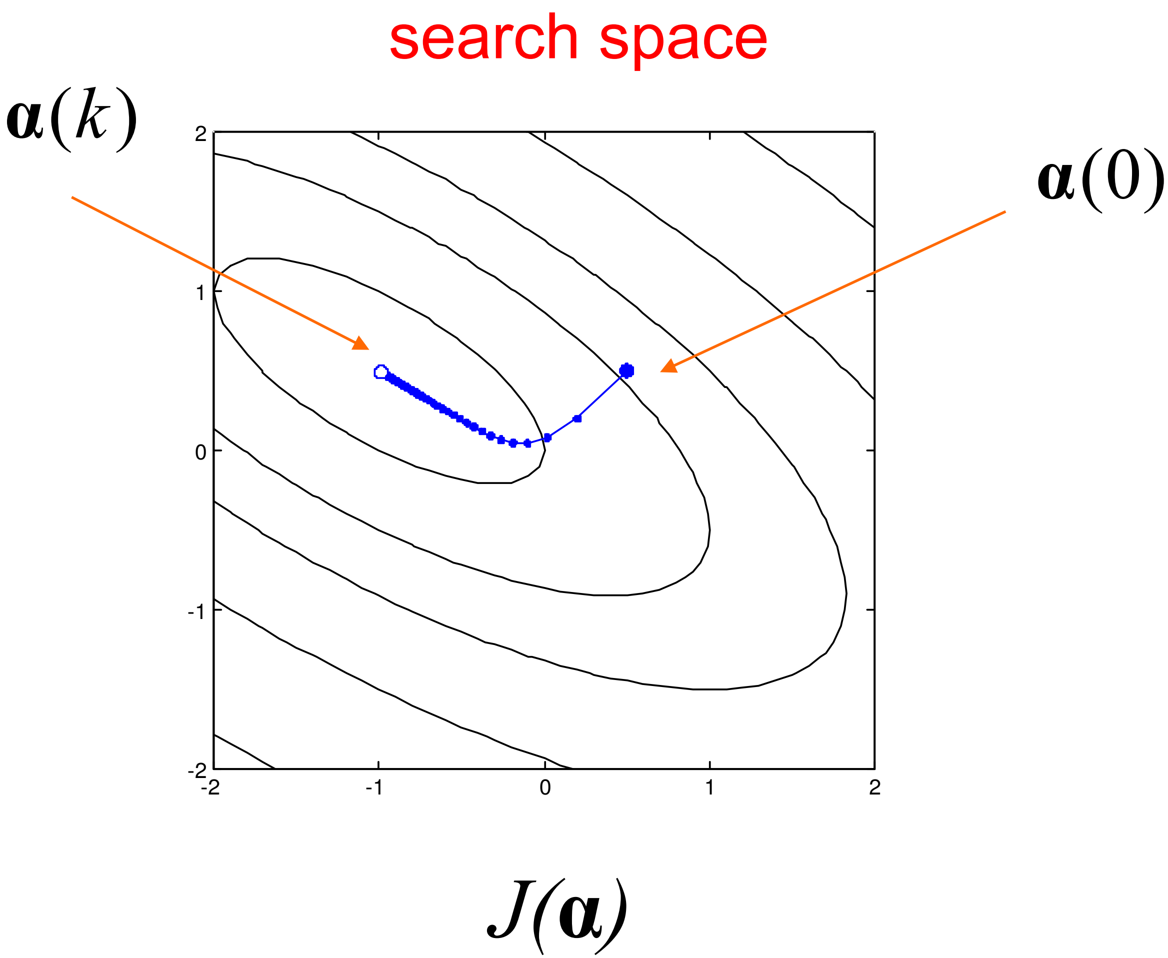
$$\mathbf{a}(k+1) = \mathbf{a}(k) - \eta(k)\nabla J(\mathbf{a}(k))$$

Algorithm 1 (Basic gradient descent)

1 **begin initialize** $\mathbf{a}$, criterion $\theta, \eta(\cdot), k = 0$
2   **do** $k \leftarrow k + 1$
3         $\mathbf{a} \leftarrow \mathbf{a} - \eta(k)\nabla J(\mathbf{a})$
4   **until** $\eta(k)\nabla J(\mathbf{a}) < \theta$
5 **return** $\mathbf{a}$
6 **end**

(a = α)

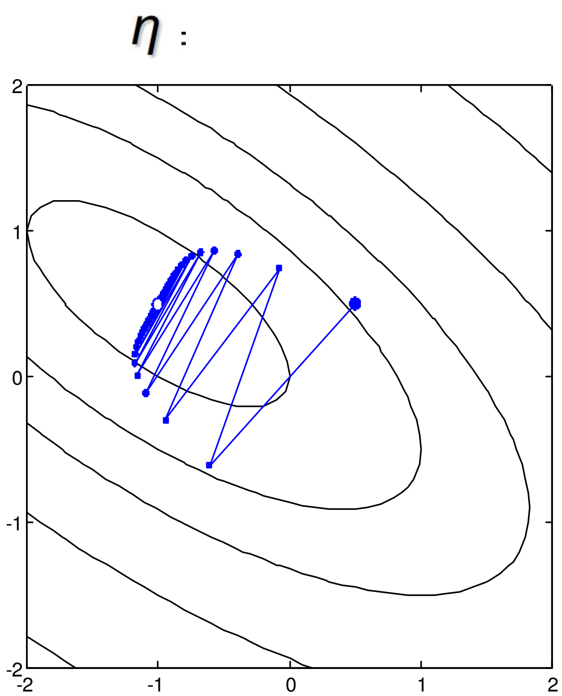
Gradient Decent (cont'd)



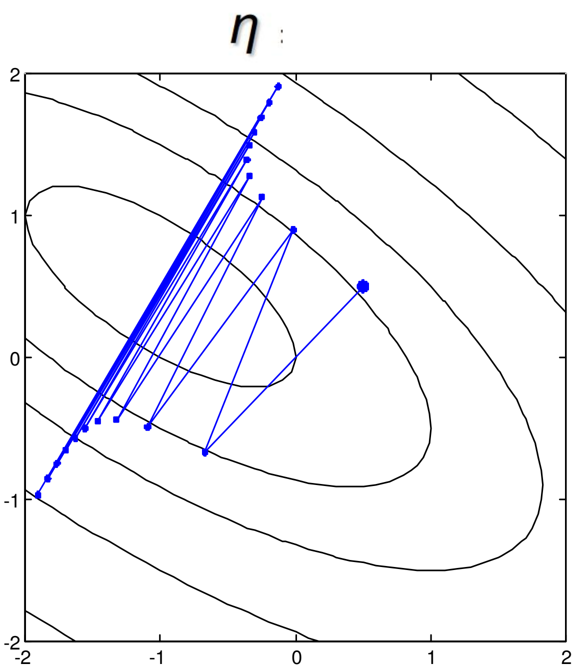
Gradient Decent (cont'd)

- What is the effect of the learning rate?

$J(\boldsymbol{\alpha})$



slow but converges to solution



fast by overshoots solution

Gradient Decent (cont'd)

- How to choose the learning rate $\eta(k)$?

Taylor series approximation (a = α)

$$J(\mathbf{a}) \simeq J(\mathbf{a}(k)) + \nabla J^t(\mathbf{a} - \mathbf{a}(k)) + \frac{1}{2}(\mathbf{a} - \mathbf{a}(k))^t\mathbf{H}\,(\mathbf{a} - \mathbf{a}(k))$$

Hessian (2nd derivatives)

Setting a=a(k+1) and using $\mathbf{a}(k+1) = \mathbf{a}(k) - \eta(k)\nabla J(\mathbf{a}(k))$

$$J(\mathbf{a}(k+1)) \simeq J(\mathbf{a}(k)) - \eta(k)\|\nabla J\|^2 + \frac{1}{2}\eta^2(k)\nabla J^t\mathbf{H}\nabla J$$

$$\eta(k) = \frac{\|\nabla J\|^2}{\nabla J^t\mathbf{H}\nabla J}$$

optimum learning rate

Choosing pk using Newton's Method

$$\mathbf{p}_k = -H^{-1}\nabla J(\boldsymbol{\alpha}(k))$$

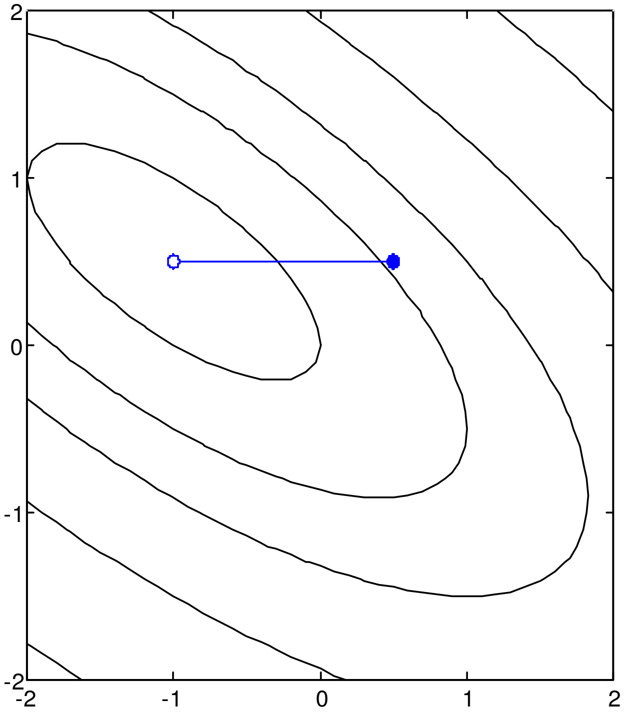
$$\mathbf{a}(k+1) = \mathbf{a}(k) - \mathbf{H}^{-1}\nabla J$$ requires inverting H

Algorithm 2 (Newton descent)

<u>**begin**</u> <u>**initialize**</u> $\mathbf{a}$, criterion θ
        <u>**do**</u>
            $\mathbf{a} \leftarrow \mathbf{a} - \mathbf{H}^{-1}\nabla J(\mathbf{a})$ (a = α)
        <u>**until**</u> $\mathbf{H}^{-1}\nabla J(\mathbf{a}) < \theta$
    <u>**return**</u> $\mathbf{a}$
<u>**end**</u>

Newton's Method (cont'd)

$J(\boldsymbol{\alpha})$

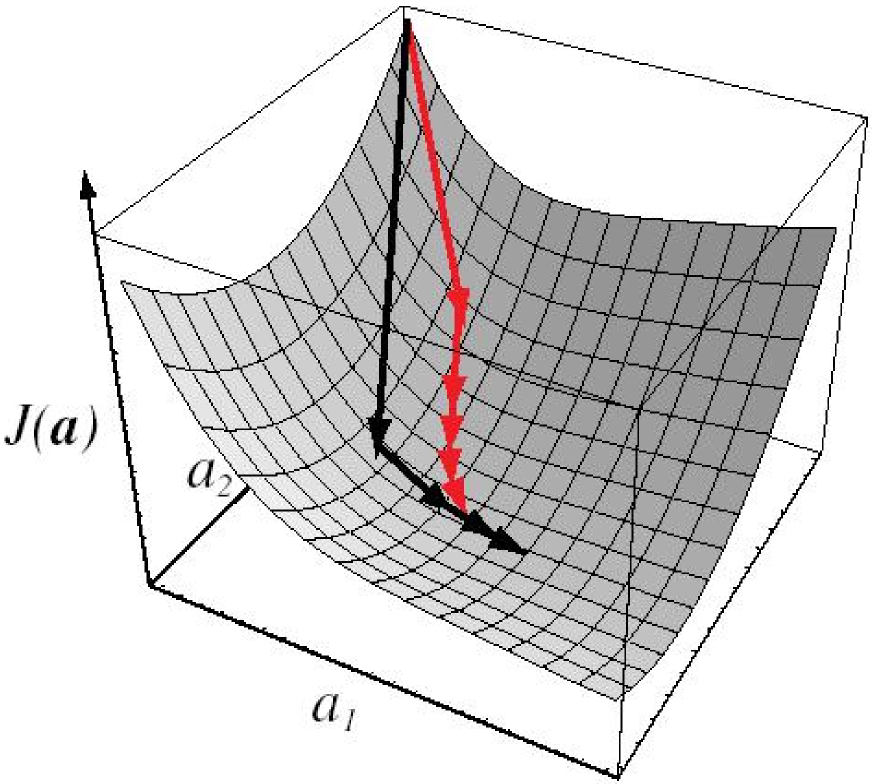


If $J(\alpha)$ is **quadratic**, Newton's method converges in **one step**!

Gradient decent vs Newton's method

Newton's method

Gradient Decent



Q & A