

# Statistical Learning Theory: Concepts, Learning Proofs and Kernels

Rodrigo Fernandes de Mello

Invited Professor at Télécom ParisTech

Associate Professor at Universidade de São Paulo, ICMC, Brazil

<http://www.icmc.usp.br/~mello>

[mello@icmc.usp.br](mailto:mello@icmc.usp.br)



- Objective:
  - Introduce the Theoretical Foundations on Supervised Learning
- Subjects:
  - What and How to define the Algorithm Bias?
  - How to formulate Supervised Learning?
  - How to prove a Supervised Algorithm learns?
    - The Empirical Risk Minimization Principle
    - Chernoff and Hoeffding's bounds
    - Vapnik-Chervonenkis Dimension
    - The Shattering coefficient as complexity measure of algorithm biases

- Subjects:
  - Support Vector Machines as the best supervised learning algorithms
    - Provided the best feature spaces
    - Large-margin bound
    - Algebraic formulation of the SVM classification problem
  - Analyzing Kernel Transformations
    - Linear and Polynomial kernel effects on spaces
    - Estimating the complexity of the resultant spaces

# About this course

- Main References:
  - Mello, R. F. and Ponti, M. A., Machine Learning: A Practical Approach on the Statistical Learning Theory, Springer, 2018
  - Ulrike von Luxburg and Bernhard Schölkopf, Statistical Learning Theory: Models, Concepts, and Results, 2008 [Base](#)
  - Vapnik, V. N.; Statistical Learning Theory, 1998, Wiley-Interscience, 1.º Edição. Vapnik, V. N.; The Nature of Statistical Learning Theory, 1999, Springer, 2.
  - Bernhard Schölkopf and Alexander J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond
  - C. Bishop, Pattern Recognition and Machine Learning, Springer-Verlag New York, 2006
  - Stephen Boyd and Lieven Vandenberghe, Convex Optimization, Cambridge University Press
  - Mokhtar S. Bazaraa, John J. Jarvis, Hanif D. Sherali, Linear Programming and Network Flows, 4th Edition, 2009, Wiley
  - Gilbert Strang, Introduction to Linear Algebra, Wellesley-Cambridge Press, 2016

- TPs
  - December 4<sup>th</sup> from 10:30hs to 12:30hs
  - January 22<sup>nd</sup> from 14hs to 19hs

# **Machine Learning Review**

# Machine Learning Review

- What is machine learning for you?
- How would you define ML?

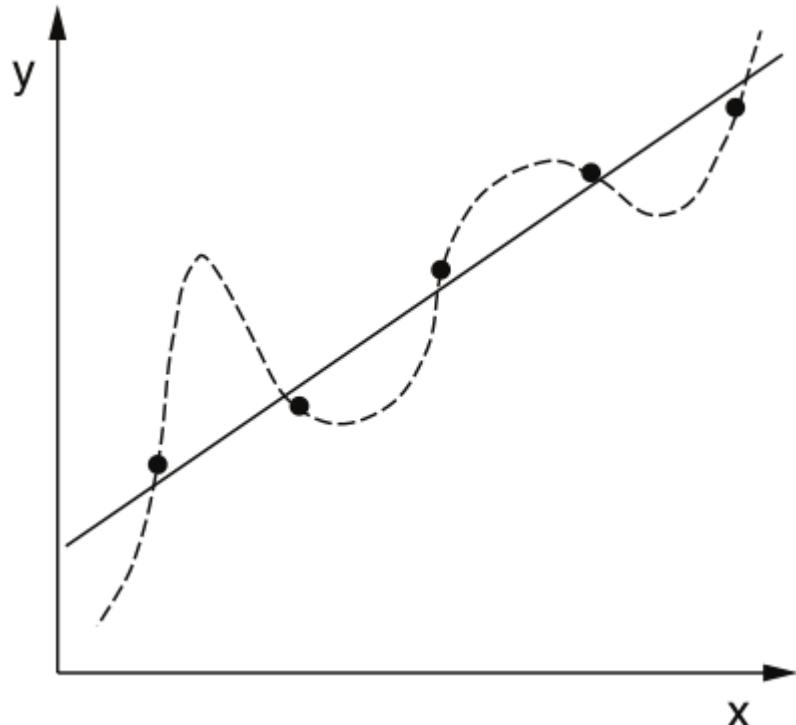
# Machine Learning Review

- There are two main types of learning:
  - Supervised
    - Ulrike von Luxburg and Bernhard Schoelkopf, Statistical Learning Theory: Models, Concepts, and Results, Handbook for the History of Logic, Vol. 10: Inductive Logic. Elsevier, 2011
    - D. H. Wolpert and William G. Macready, 1997. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1), 67–82
  - Non-supervised
    - Gunnar Carlsson and Facundo Memoli, Characterization, Stability and Convergence of Hierarchical Clustering Methods, *Journal of Machine Learning Research*, 2010

**Supervised Learning is strongly based on  
the Bias-Variance Dilemma**

# Bias-Variance Dilemma

- For instance, consider the examples (points) collected during an experiment. Then, take two different functions to model them

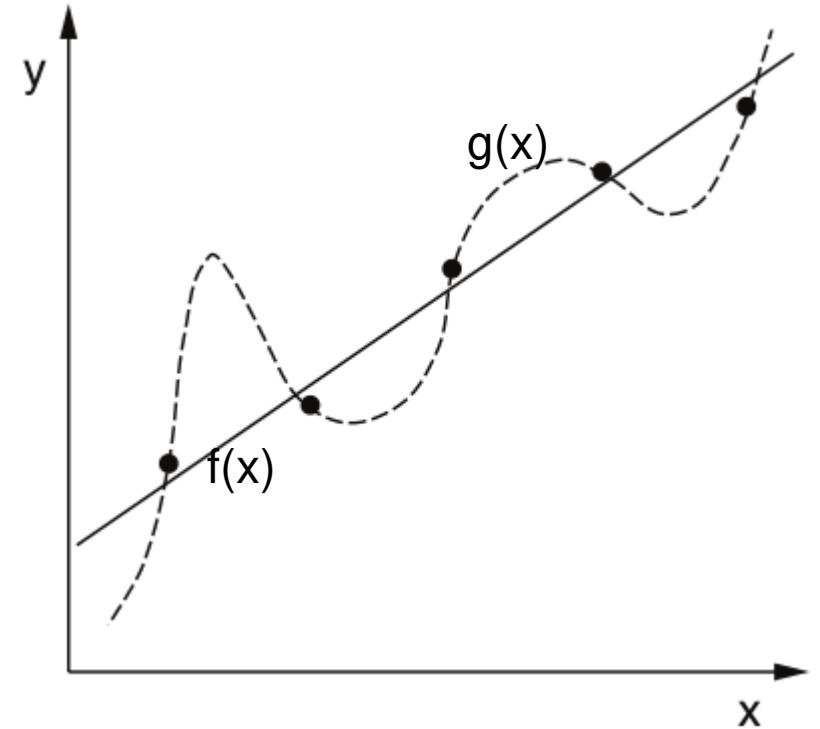


**Which function is the best?**

To answer it we need to assess their **Expected Risks**

# Bias-Variance Dilemma

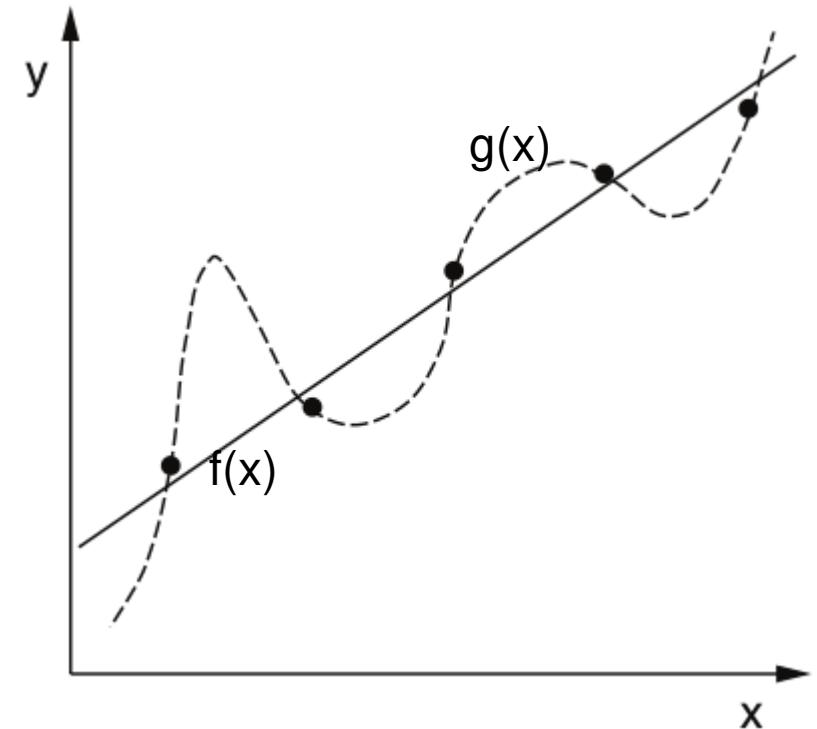
- But how to compute the Expected Risk if we only have one sample?
  - Consider that line  $f(x)$  has training error greater than zero
  - Consider the polynomial function  $g(x)$  has training error equals to zero



# Bias-Variance Dilemma

- But how to compute the Expected Risk if we only have one sample?

- Consider that line  $f(x)$  has training error greater than zero
- Consider the polynomial function  $g(x)$  has training error equals to zero
- Let  $g(x)$  represent an overfitted model
- Thus, as unseen examples are received,  $g(x)$  moves from no Training error to a great Expected error



# Bias-Variance Dilemma

- But how to compute the Expected Risk if we only have one sample?

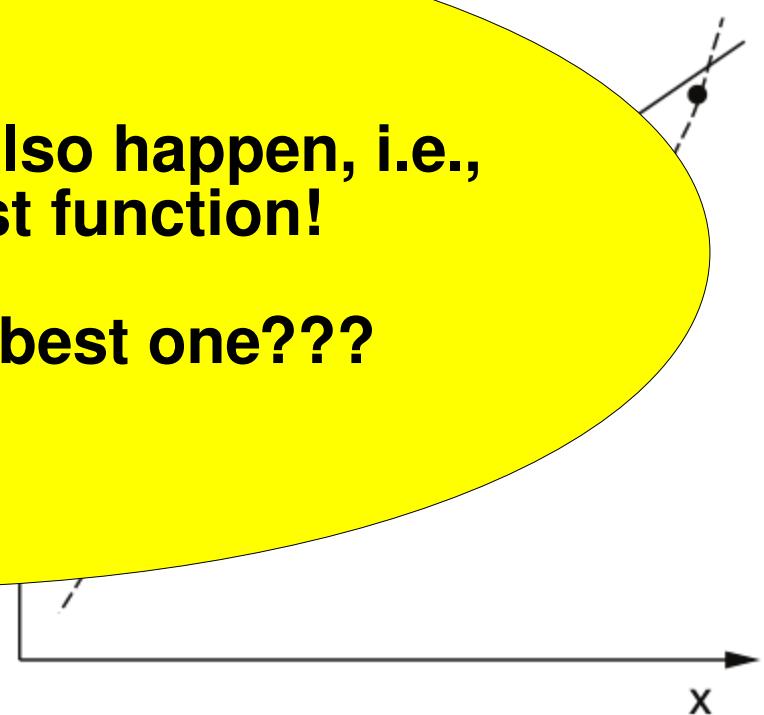
- Consider that line  $f(x)$  has training error greater than zero

- Consider  $g(x)$  has training error less than zero

But the opposite could also happen, i.e.,  
 $g(x)$  may be the best function!

So, how to select the best one???

- Let's consider two models  $f(x)$  and  $g(x)$ .  
Let's assume that  $f(x)$  is a linear model and  $g(x)$  is a quadratic model.  
Thus,  $f(x) = w_0 + w_1 x$  and  $g(x) = w_0 + w_1 x + w_2 x^2$ .  
Training error to  $f(x)$  is  $\sum_i (f(x_i) - y_i)^2$  and training error to  $g(x)$  is  $\sum_i (g(x_i) - y_i)^2$ .



# Bias-Variance Dilemma

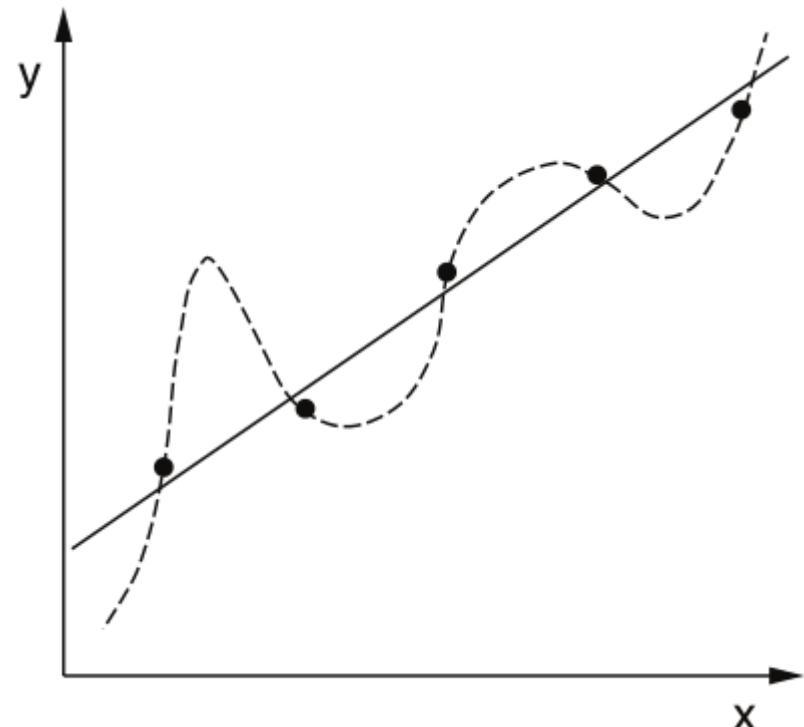
- Finally, which classifier should we choose?

1) The one with best fit with training data (the most complex one)?

2) Or the one that has greater Training error, however it was obtained using a simpler class of functions?

In Statistics, this is known as the Bias-Variance Dilemma

This Dilemma is central for supervised learning!

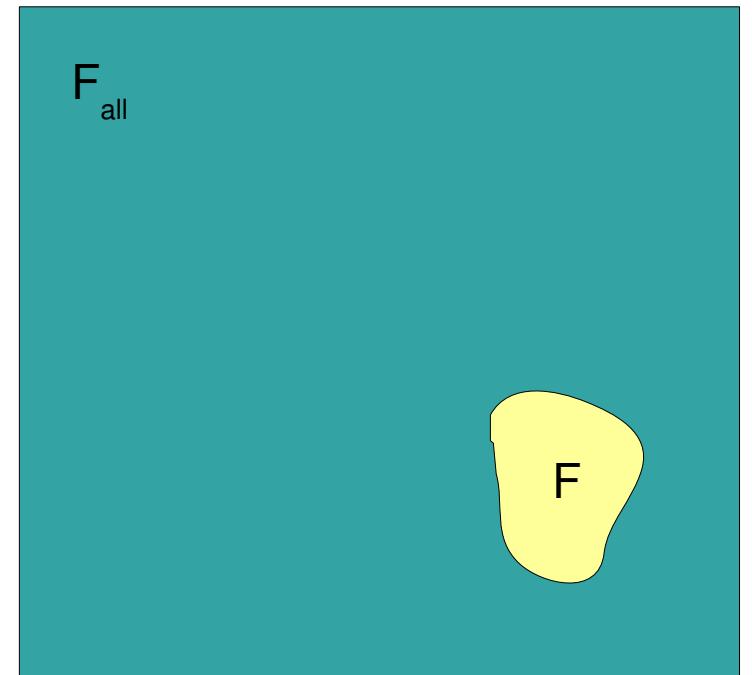


# Bias-Variance Dilemma

- Dilemma:
  - Bias:
    - If we assume a linear fit, only a linear classifier could be obtained
    - i.e., there is a strong bias imposed by ourselves
  - Variance:
    - If we fit a high-order polynomial function over training data, we can always have a perfect classifier for the sample
    - However this classifier is subject to greater fluctuations to unseen data

# Bias-Variance Dilemma

- This dichotomy associated to the Bias-Variance Dilemma is obvious when we consider the space of possible functions to build our classifier



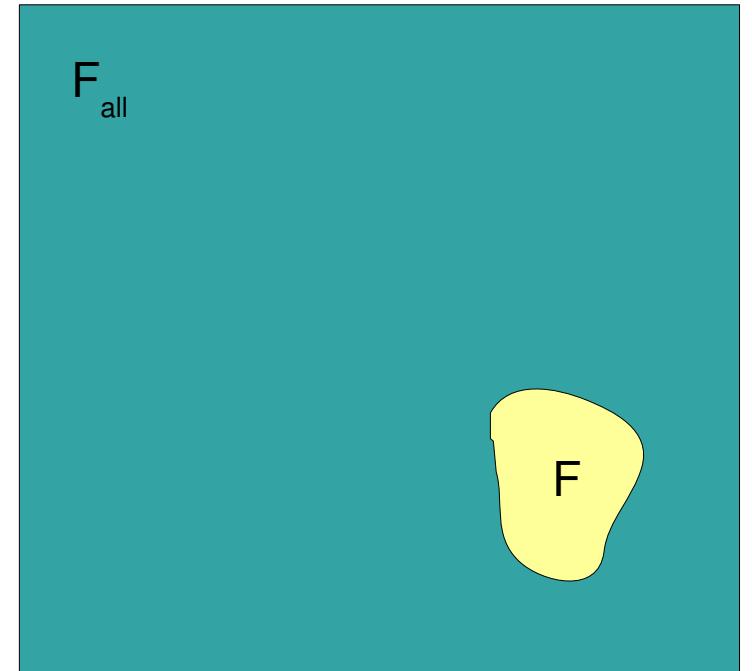
# Bias-Variance Dilemma

- This dichotomy associated to the Bias-Variance Dilemma is obvious when we consider the space of possible functions to build our classifier

Let space  $F_{\text{all}}$  contain all possible classification functions (or regression functions)

We could define a strong bias, i.e., a subspace  $F$  which contains only the linear functions to perform regression

This bias reduces the variance, i.e., it reduces the number of possible classifiers we can produce



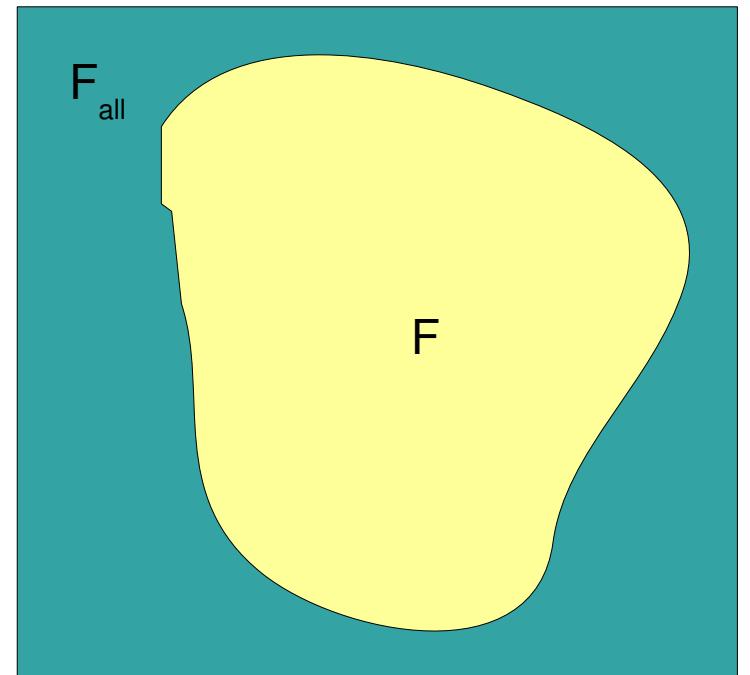
# Bias-Variance Dilemma

- This dichotomy associated to the Bias-Variance Dilemma is obvious when we consider the space of possible functions to build our classifier

**On the other hand, now see a small bias defined by subspace  $F$ , which contains many more functions to produce a classifier**

**In this case, variance is greater, i.e., the number of possible classifiers to select from and choose is huge!**

**What if this space contains a memory-based classifier?**



**To assess the quality of a Classifier,  
we need to define a Loss function**

- Pay attention:
  - We need some function to compute when classifications are missed
  - Every classification technique relies on a loss function:
    - Then they may “walk” on the descent gradient to reduce training error
  - But how can we guarantee a small training error is good?
    - This is the central question for the STL
    - This is basically the Bias-Variance Dilemma

- Pay attention:
  - We need some function to compute when classifications are missed
  - Every classification technique relies on a loss function:
    - Then they may “walk” on the descent gradient to reduce training error
  - But how can we guarantee a small training error is good?
    - This is the central question for the STL
    - This is basically the Bias-Variance Dilemma
    - **We should evaluate our classifier in terms of unseen examples**
      - **And compare the errors produced after training and testing!**

**Recall some important classification algorithms**

# Machine Learning Review

- Recall some important supervised learning algorithms:
  - Perceptron
  - Multilayer Perceptron
  - Naive Bayes
  - Decision Trees
    - C4.5, ID3
  - Support Vector Machines
- Do they work? Do they “learn” something?
  - First of all, what is learning for them?
  - Can we obtain some sort of formalization that ensures learning for them?

# Statistical Learning Theory

Universidade de São Paulo  
Instituto de Ciências Matemáticas e de Computação  
Departamento de Ciências de Computação

Rodrigo Fernandes de Mello

<http://www.icmc.usp.br/~mello>

[mello@icmc.usp.br](mailto:mello@icmc.usp.br)

# Statistical Learning Theory

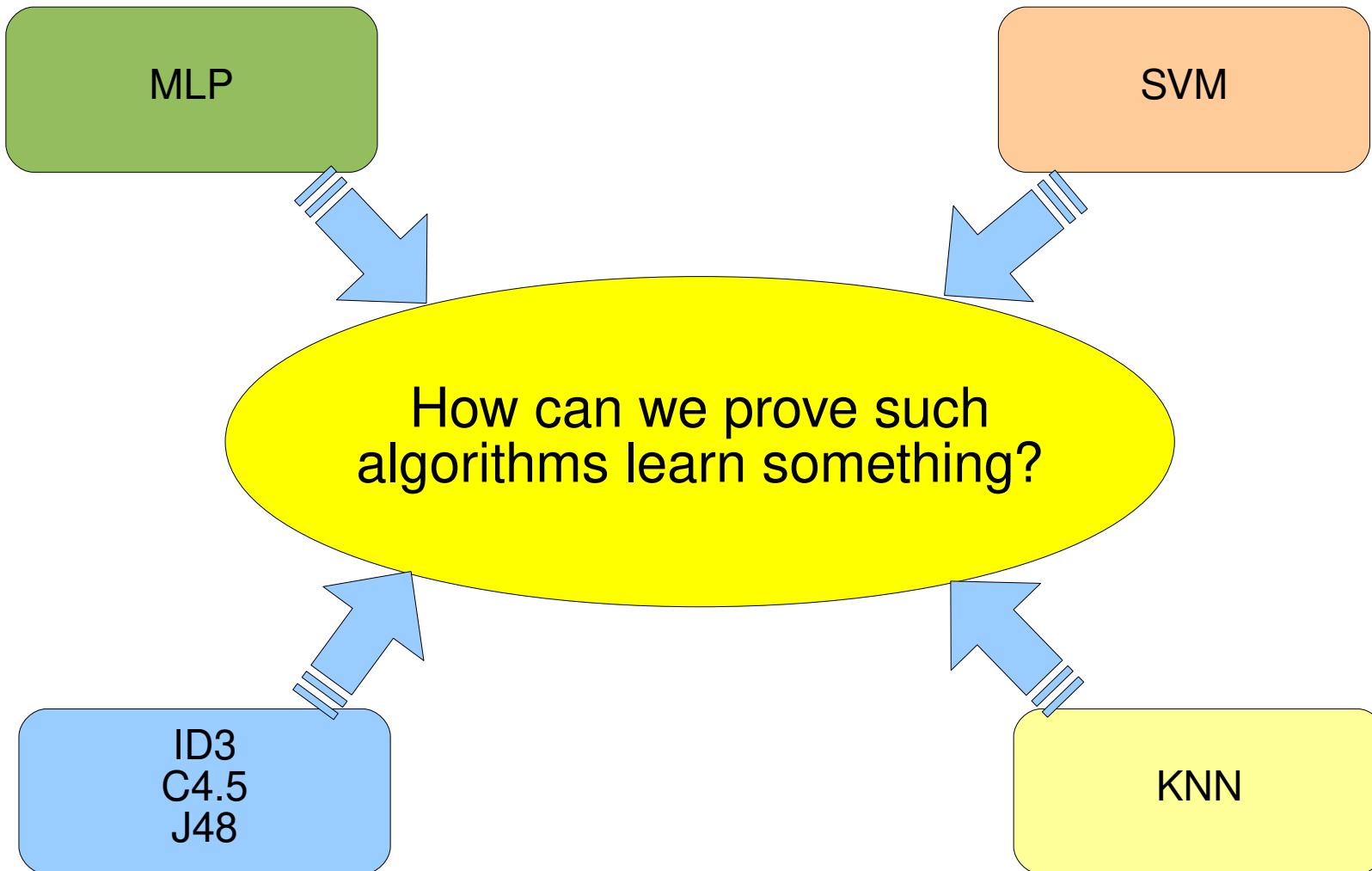
MLP

SVM

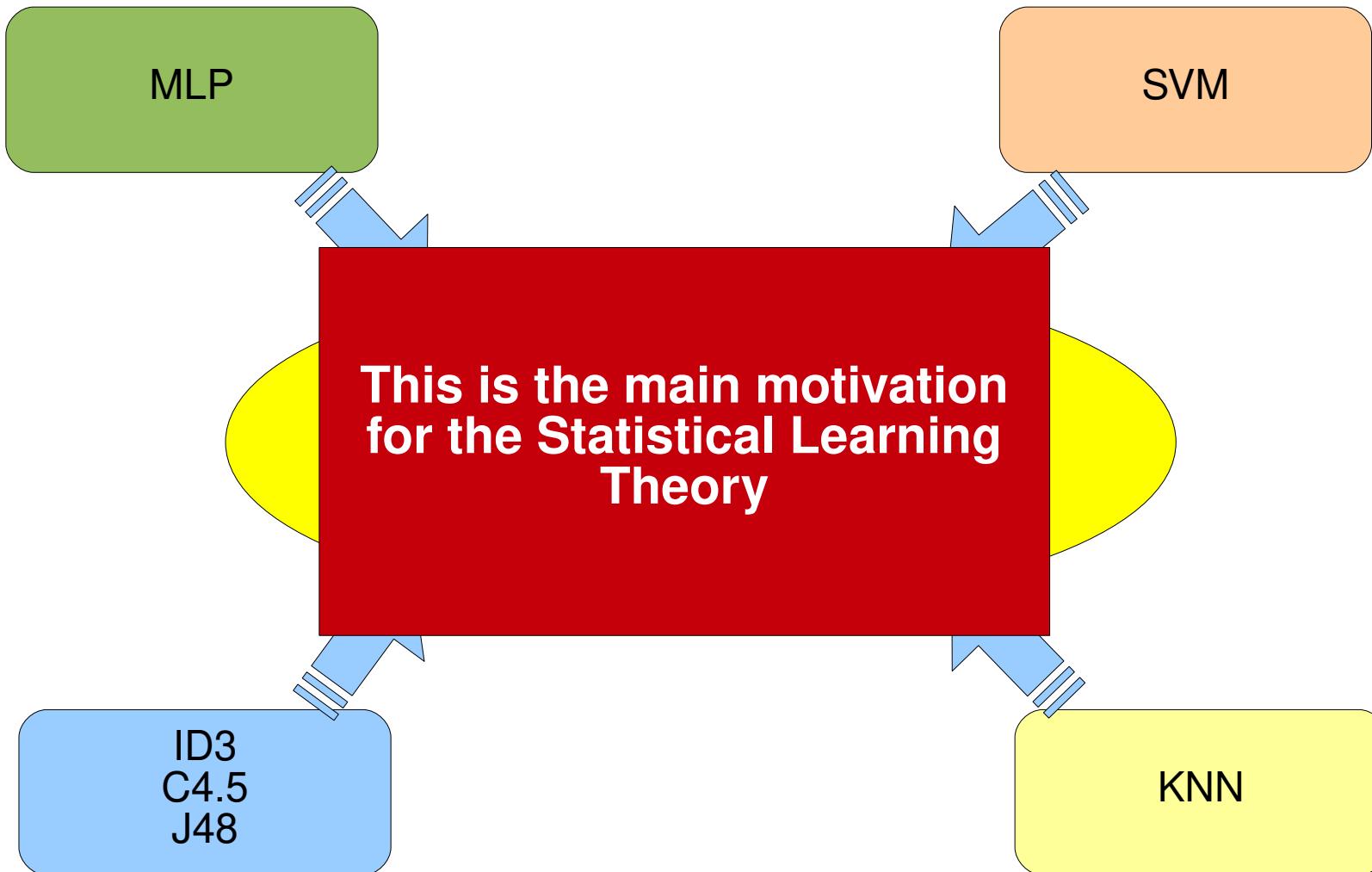
ID3  
C4.5  
J48

KNN

# Statistical Learning Theory



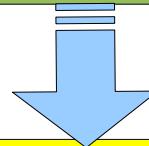
# Statistical Learning Theory



# Basic concepts

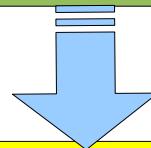
**Proposed by Vapnik et al.**

Proposed by Vapnik et al.

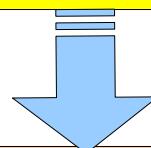


How did they define learning?

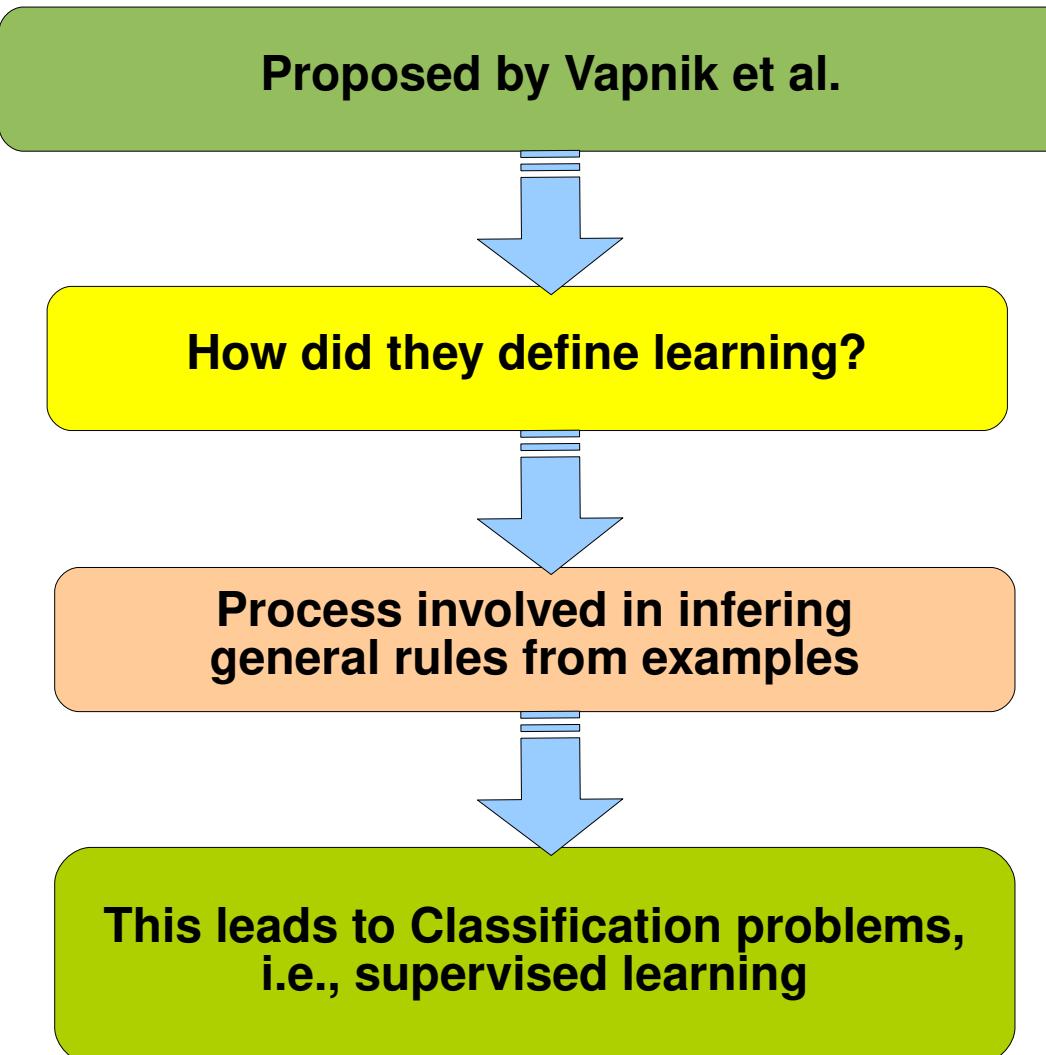
Proposed by Vapnik et al.



How did they define learning?



Process involved in inferring  
general rules from examples



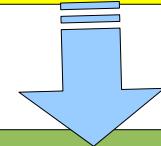
# Basic concepts



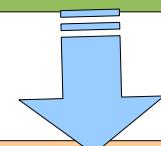
**No one needs to formally define a car  
Children learn by examples and labels**

**Which features can we use to classify an object as a car?**

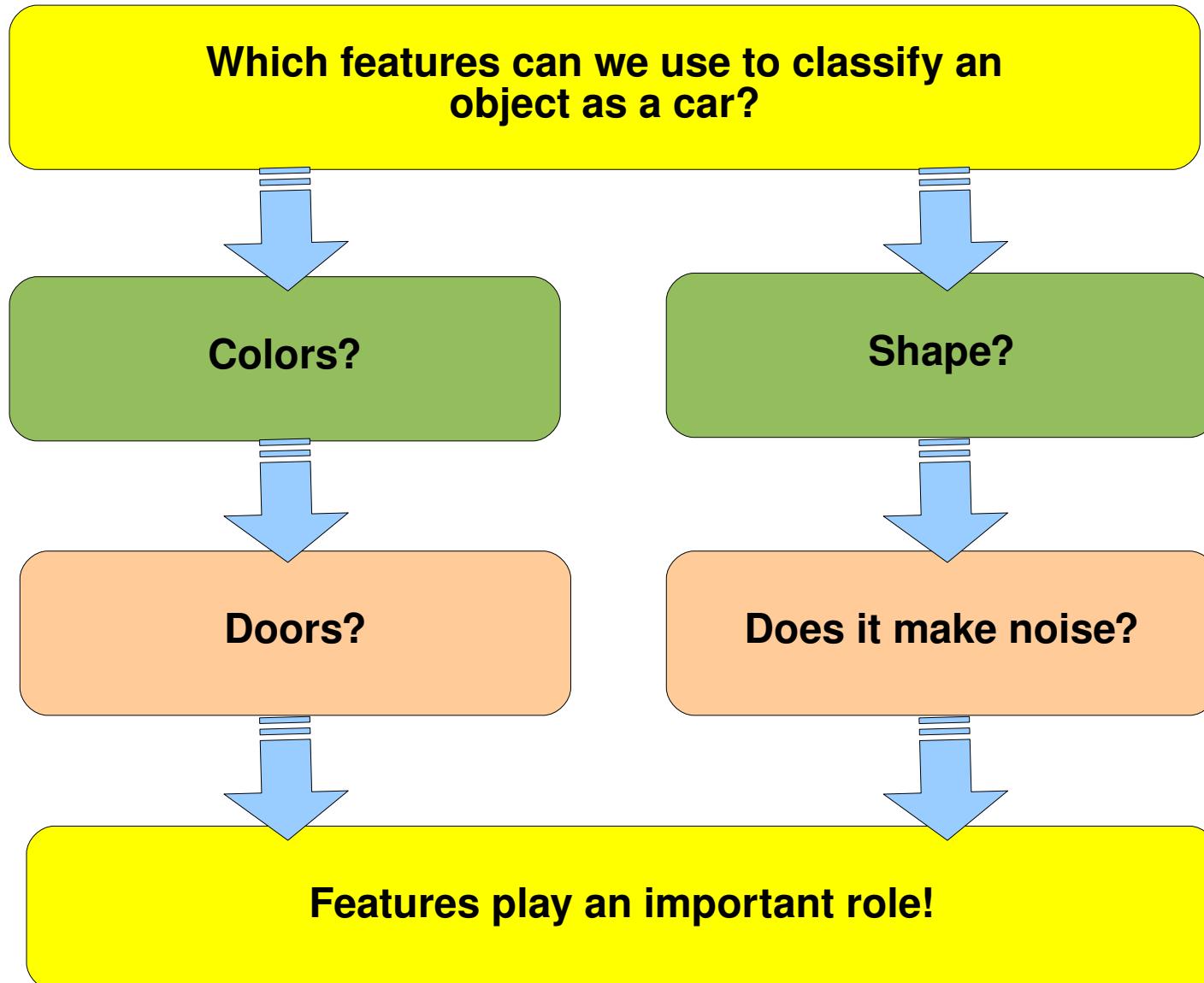
**Which features can we use to classify an object as a car?**



**Colors?**



**Doors?**

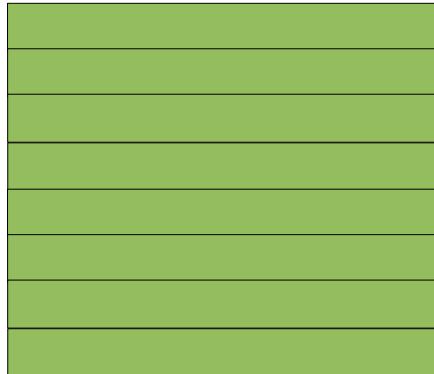


- As first step, we must define:
  - The input space
  - The output space
  - Necessary assumptions
  - The loss function
  - The risk of a classifier

- In **supervised learning**, we have:
  - Input space X
  - Output space (classes or labels) Y
    - We will take the binary classification {-1, +1}
- Therefore learning involves **estimating** the functional:

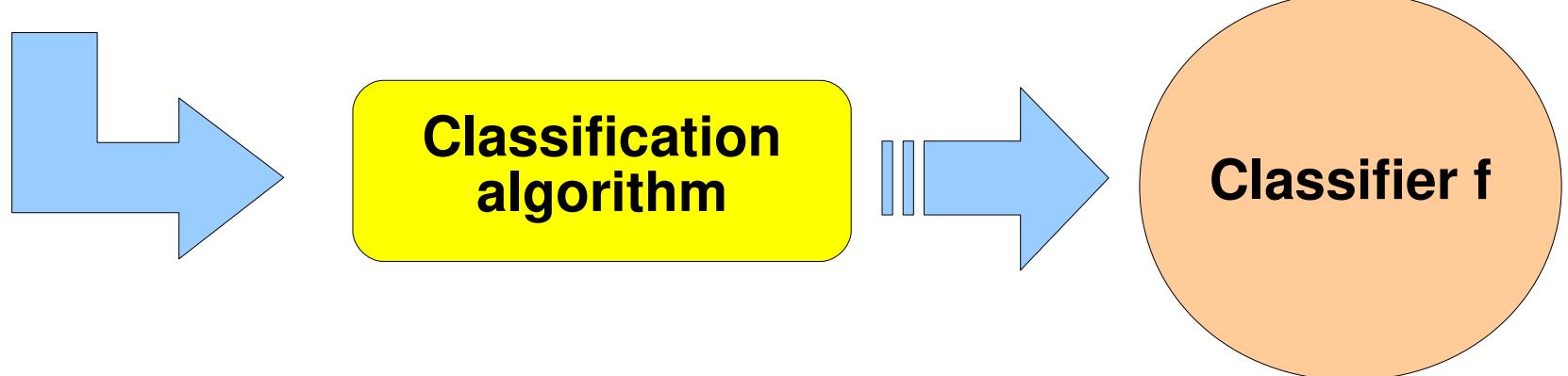
$$f : X \rightarrow Y$$

- Having f as the **classifier**
  - A classifier is different from the **Classification algorithm**

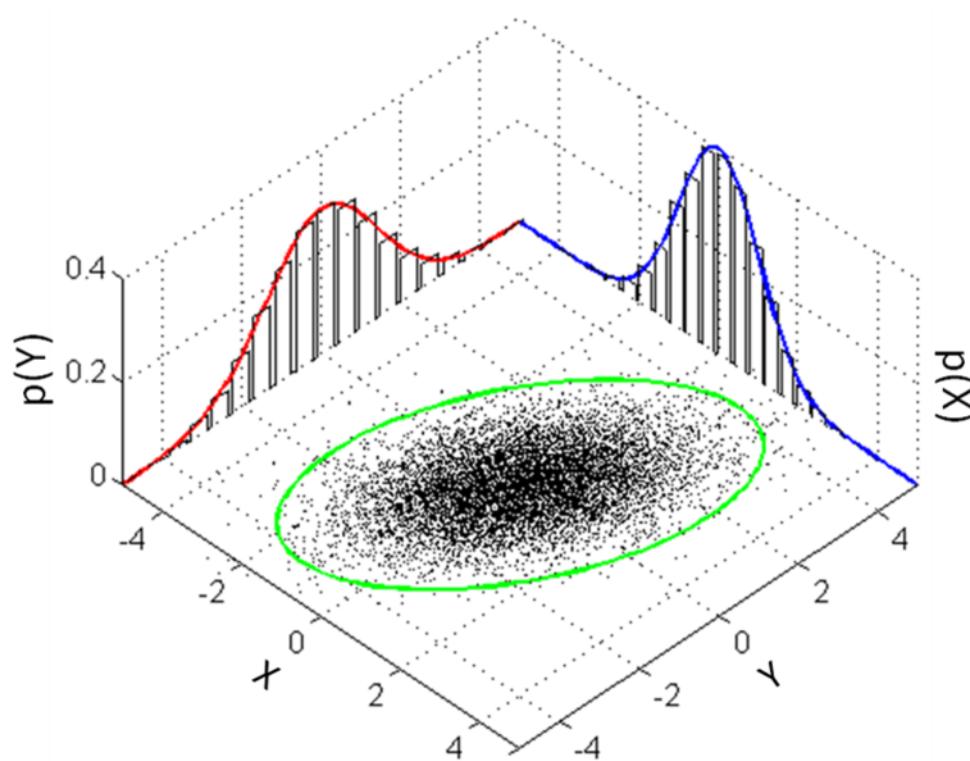


**Learning examples**

$$(X_1, Y_1), \dots, (X_n, Y_n) \in X \times Y$$



- SLT considers a **joint probability distribution** function  $P$  over  $X \times Y$ 
  - Let's exemplify such distribution
  - This distribution models the probability each pair  $X$  and  $Y$  is in a particular interval



- Example of the **joint probability distribution P**
  - Consider one throws a die
    - Let  $X=1$  if an **even** number occurs (i.e. 2, 4, or 6)
    - Let  $X=0$  otherwise

- Example of the **joint probability distribution P**
  - Consider one throws a die
    - Let  $X=1$  if an **even** number occurs (i.e. 2, 4, or 6)
    - Let  $X=0$  otherwise
  - In addition, let  $Y=1$  if the number is **prime** (i.e. 2, 3, or 5) and  $Y=0$ , otherwise
  - So, the joint probability for X and Y is given by:

$$P(X = 0, Y = 0) = P\{1\} = \frac{1}{6} \quad P(X = 1, Y = 0) = P\{4, 6\} = \frac{2}{6}$$
$$P(X = 0, Y = 1) = P\{3, 5\} = \frac{2}{6} \quad P(X = 1, Y = 1) = P\{2\} = \frac{1}{6}$$

- Example of the **joint probability distribution P**
  - Consider one die roll
    - Let  $X=1$  if the outcome is odd (i.e. 1, 3, or 5)
    - Let  $X=2$  if the outcome is even (i.e. 2, 4, or 6)
  - What if  $X = 0$ ?
    - Should we choose  $Y=0$  or  $Y=1$ ?
  - In addition, let  $Y=1$  if the outcome is even (i.e. 2, 4, or 6) and  $Y=0$ , otherwise.
  - So, the joint probability distribution  $P(X, Y)$  is given by:

$$\begin{aligned} P(X = 0, Y = 0) &= P\{1\} = \frac{1}{6} & P(X = 1, Y = 0) &= P\{4, 6\} = \frac{2}{6} \\ P(X = 0, Y = 1) &= P\{3, 5\} = \frac{2}{6} & P(X = 1, Y = 1) &= P\{2\} = \frac{1}{6} \end{aligned}$$

- Example of the **joint probability distribution P**
  - Consider one trial
    - Let  $X=1$ ,  $Y=0$  (e.g. 1, 2, 4, or 6)
    - Let  $X=1$ ,  $Y=1$  (e.g. 2, 3, or 5)
  - In addition, let  $X=0$ ,  $Y=0$ ,  
and  $Y=1$ ,  
 $X=0$ ,  $Y=1$
  - So, the joint probability distribution P is given by:

Try to illustrate the space  $P(X,Y)$

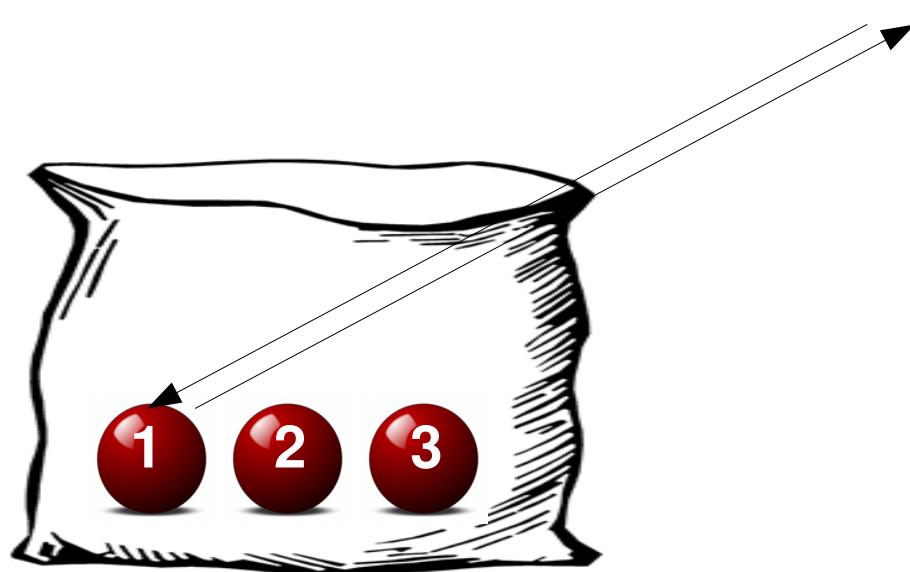
$$\begin{aligned} P(X = 0, Y = 0) &= P\{1\} = \frac{1}{6} & P(X = 1, Y = 0) &= P\{4, 6\} = \frac{2}{6} \\ P(X = 0, Y = 1) &= P\{3, 5\} = \frac{2}{6} & P(X = 1, Y = 1) &= P\{2\} = \frac{1}{6} \end{aligned}$$

- But what does it mean this **joint distribution P?**
  - In Statistics: it corresponds to the relationship between two random variables X and Y
  - In Machine Learning: it corresponds to the relationship between input space X and labels Y

- Vapnik et al. still made assumptions to ensure learning:
  - Examples are sampled in an independent manner
  - No assumption is made about  $P$
  - Labels can assume nondeterministic values
    - Due to noise and class overlapping
  - Distribution  $P$  is static
  - Distribution  $P$  is unknown at training

- SLT assumes examples are sampled in an independent manner:
  - Thus, the probability to obtain the first example with its label, i.e.,  $(X_1, Y_1)$ , does not influence in the next draw

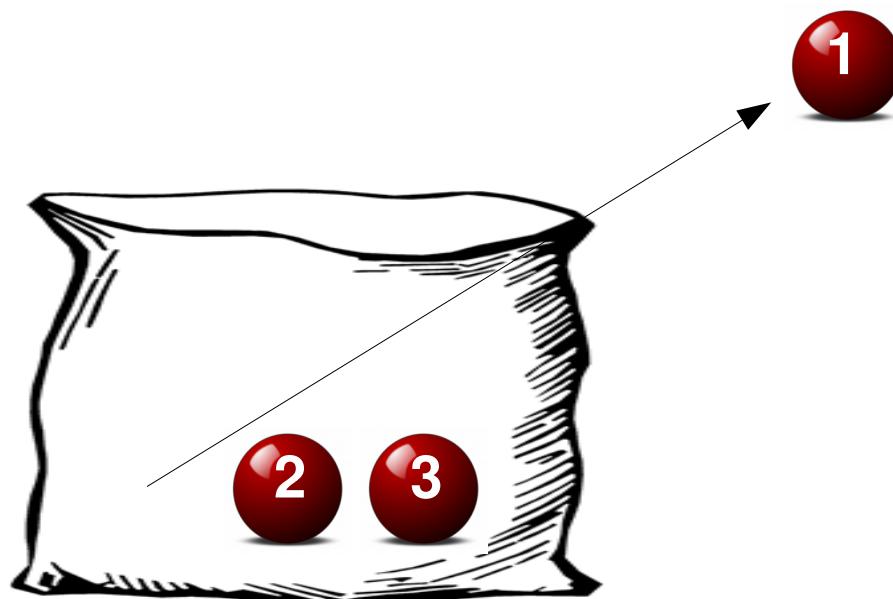
- SLT assumes examples are sampled in an independent manner:
  - **Draw with replacement**



**Probability to obtain ball with value 1 is always  $1/3$**

**If we first draw ball 1, there is no change in the probability of others nor even ball 1**

- SLT assumes examples are sampled in an independent manner:
  - **Draw without replacement**



**At first, the probability to draw ball 1 is  $1/3$ , afterwards it is zero!**

**Observe this draw has influenced in the probability**

**In this situation events are dependent!**

- **Independent sampling**

- For example, consider a training set we use to learn **handwritten characters**:
  - It is safe to assume a set of characters written by a large number of people is an independent sample in terms of the whole population of handwritten characters
- On the other hand, consider the **drug discovery** scenario:
  - This is a field in which researchers look for new useful compounds to tackle health issues
  - It is too way expensive to extract features of those compounds

- **Independent sampling**

- On the other hand, consider the **drug discovery** scenario:
  - As result, only some compounds are experimented
  - Those compounds are carefully selected based on some previous knowledge
  - In this scenario, we cannot assume that compounds are an independent sample from the population of chemicals, as they are **manually selected and not randomly**

- **Independent sampling**

- There are some areas that impose relaxations to this principle:
  - For example, there are researchers who predict time series considering observations are independent among each other

- Test data dependency:
  - Using autocorrelation function to check out the dependence among examples:
    - Normal distribution
    - Any other deterministic time series
  - Try to estimate the probability distribution of a time series along the time domain (using data windows)

- Vapnik et al. still made assumptions to ensure learning:
  - Examples are sampled in an independent manner
  - No assumption is made about  $P$
  - Labels can assume nondeterministic values
    - Due to noise and class overlapping
  - Distribution  $P$  is static
  - Distribution  $P$  is unknown at training

- **No assumption is made about P**
  - P can be any distribution!
  - Thus the SLT works in an agnostic way, what is different from the traditional Statistics which would assume P comes from some family of distributions:
    - What would reduce the objective to estimate such distribution parameters

- Vapnik et al. still made assumptions to ensure learning:
  - Examples are sampled in an independent manner
  - No assumption is made about  $P$
  - Labels can assume nondeterministic values
    - Due to noise and class overlapping
  - Distribution  $P$  is static
  - Distribution  $P$  is unknown at training

- **Labels can assume nondeterministic values:**
  - Two reasons for that:
    - First: Data can contain noise in labels, i.e., labels  $Y_i$  may be wrong:
      - This is an important assumption once people may label data incorrectly, such as in a spam detection tool
      - Of course we expect only a small portion of labels is wrong

- **Labels can assume nondeterministic values:**
  - Two reasons for that:
    - Second: Overlapping classes
      - For example, consider the problem of classifying gender according to people heights
      - Of course a person of height 1.80 meter can be from either one of genders, therefore we cannot associate a unique label Y to an input X = 1.80

- **Labels can assume nondeterministic values:**
  - For this theoretical framework, it does not matter why labels are nondeterministic. In practice what matters is to find the conditional probabilities:

$$\begin{aligned}P(Y = 1 | X = x) \\ P(Y = -1 | X = x)\end{aligned}$$

- In case of small noise perturbations in labels, the conditional probabilities will be close to 0 or 1
- In case of great noise, such probabilities will be closer to 0.5, jeopardizing learning results

- **Labels can assume nondeterministic values:**
  - The same problem happens when classifying people according to their heights, this occurs because there is a significant overlapping between classes
  - For example, let us consider:

$$P(Y = \text{"masculino"} | X = 1.70) = 0.6$$

- In average we will make an error of 40%
- Thus, learning becomes even more difficult when the conditional probability is close to 0.5, producing a classifier that will make a greater error

- Vapnik et al. still made assumptions to ensure learning:
  - Examples are sampled in an independent manner
  - No assumption is made about  $P$
  - Labels can assume nondeterministic values
    - Due to noise and class overlapping
  - **Distribution  $P$  is static**
  - Distribution  $P$  is unknown at training

- **Distribution P is static:**

- SLT does not assume the **parameter time**, therefore there is no modification of P along time
- This is not the case of problems such as:
  - Learning on time series
  - Concept drift in data streams

- Vapnik et al. still made assumptions to ensure learning:
  - Examples are sampled in an independent manner
  - No assumption is made about  $P$
  - Labels can assume nondeterministic values
    - Due to noise and class overlapping
  - Distribution  $P$  is static
  - **Distribution  $P$  is unknown at training**

- **Distribution P is unknown at training stage**
  - If we knew P, learning would be trivial
    - We would just solve a regression problem to find its parameters
  - Therefore we have an indirect access to P
    - This means that if we have a sufficiently large set of training examples, we can **estimate P**

- As seen before, the objective of **supervised learning** is to learn a functional:

$$f : X \rightarrow Y$$

- Considering:
  - An input space  $X$
  - An output space  $Y$
- We refer to this function  $f$  as **classifier**
- In order to estimate  $f$ , we need a measure to quantify **how good** is such function when used to classify unseen examples:
  - In this way we introduce the concept of **loss function**

- As seen before, the objective of **supervised learning** is to learn a functional:

$$f : X \rightarrow Y$$

- Considering:
  - Applying
  - Comparing
  - Optimizing
- With respect to:
  - What we want
- In order to get good results from unseen examples:
  - In this way we can define a loss function

This is an important concept for the SLT!

how  
unseen

loss function

- The simplest **loss function** is the **0-1-loss** also called classification error:
  - This function results in 0 if  $x$  is correctly classified as  $y$
  - Otherwise 1

$$l(x, y, f(x)) = \begin{cases} 1 & \text{se } f(x) \neq y \\ 0 & \text{caso contrário.} \end{cases}$$

- The simplest **loss function** is the **0-1-loss** also called classification error:
  - This function results in 0 if  $x$  is correctly classified as  $y$
  - Otherwise 1

$$l(x, y, f(x)) = \begin{cases} 1 & \text{se } f(x) \neq y \\ 0 & \text{caso contrário.} \end{cases}$$

- There are other problems, such as function regression, in which errors must be measured using real numbers. In such situations, one can use a loss function based on **squared errors**:

$$l(x, y, f(x)) = (y - f(x))^2$$

- **i.e., the loss function may vary according to the learning objective.** The only convention is that zero indicates perfect classification and greater values correspond to losses

# Risk or Expected Risk

- While the **loss function** measures the error of a function  $f$  for an individual input  $x$ , the **risk** or **expected risk** computes the average loss according to the **joint probability distribution  $P$** :

$$R(f) = E(l(x, y, f(x)))$$

- For all  $x$  and  $y$ 
  - Observe: even for  $x$  values we did not have access while training, i.e., even for unseen data

## Risk or Expected Risk

- In this manner, a **function f is better than another g** if and only if:

$$R(f) < R(g)$$

- Thus, the best classifier f is given by the smaller value for  $R(f)$ , i.e., the one that presents the lowest **risk**
  - Observe  $R(f)$  measures how good f fits the space  $P(X,Y)$  even for unseen data

# Risk or Expected Risk

- In this manner, a **function f is better than another g if and only if:**

$$R(f) < R(g)$$

- Thus, the classifier with the lower value for  $R(f)$  is better.

**This makes sense!**

i.e., the classifier with the lowest risk according to the joint probability distribution P

# Risk or Expected Risk

- In this manner, a **function f is better than another g** if and only if:

$$R(f) < R(g)$$

- Thus, the function  $f$  has a smaller value for  $R(f)$ .

**The problem now is that we cannot compute the expected risk once we assumed no previous knowledge about the joint probability distribution P!!!**

- We now have a **finite set of examples** to produce a good classifier  $f$ , i.e., we do not have the whole universe of examples
  - Thus, we cannot compute  $R(f)$  for a given classifier  $f$
  - However, we can employ the concept of **Empirical risk** or **training error** as follows:

$$R_{emp}(f) = \frac{1}{n} \sum_{i=1}^n l(x_i, y_i, f(x_i))$$

- We now have a **finite** set of possible classifiers  $f$ , i.e., we can consider the entire universe of possible classifiers  $f$ .

- The classifier  $f$  has to produce a good result on the training examples.

**Here we start the Principle of Empirical Risk Minimization**

**Such principle supports the selection of a classifier!**

**This is the most basic concept for SLT!**

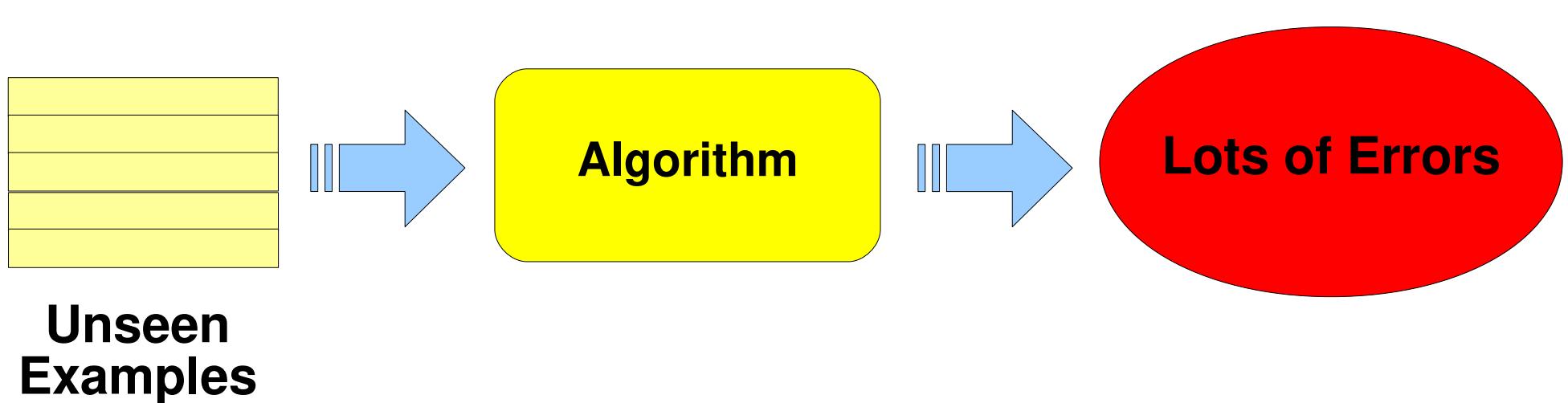
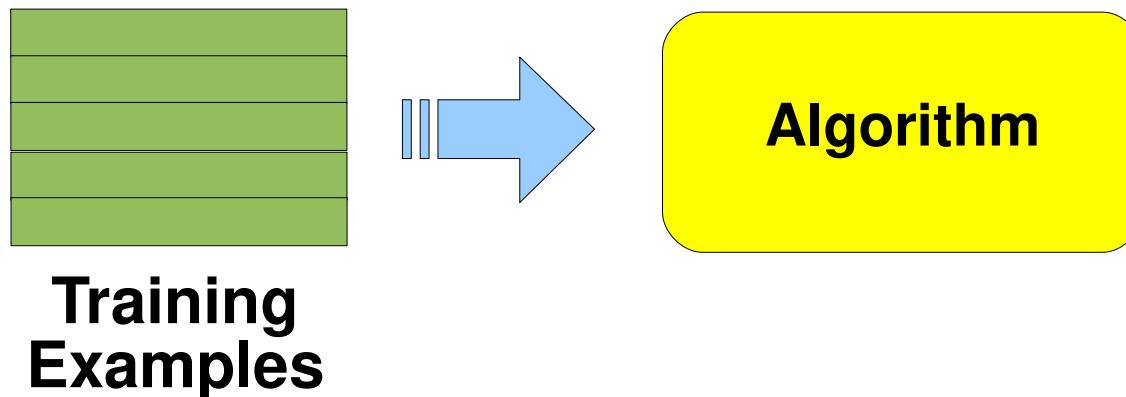
- We expect that a good classifier  $f$  produces a small **Empirical Risk** for the training set
  - Otherwise the classifier is not even capable of representing the training examples
  - However, are we sure that minimizing the Empirical Risk we obtain a good classifier?
    - A classifier may be too specific or specialized over the training set and produces a minimal as possible **Empirical Risk**, however its **Expected Risk** could be maximal

- We expect that a good classifier  $f$  produces a small **Empirical Risk** for the training set
  - Otherwise the classifier is not even capable of representing the training examples
  - However, we can always find a classifier with zero Empirical Risk we just need to overfit the training data

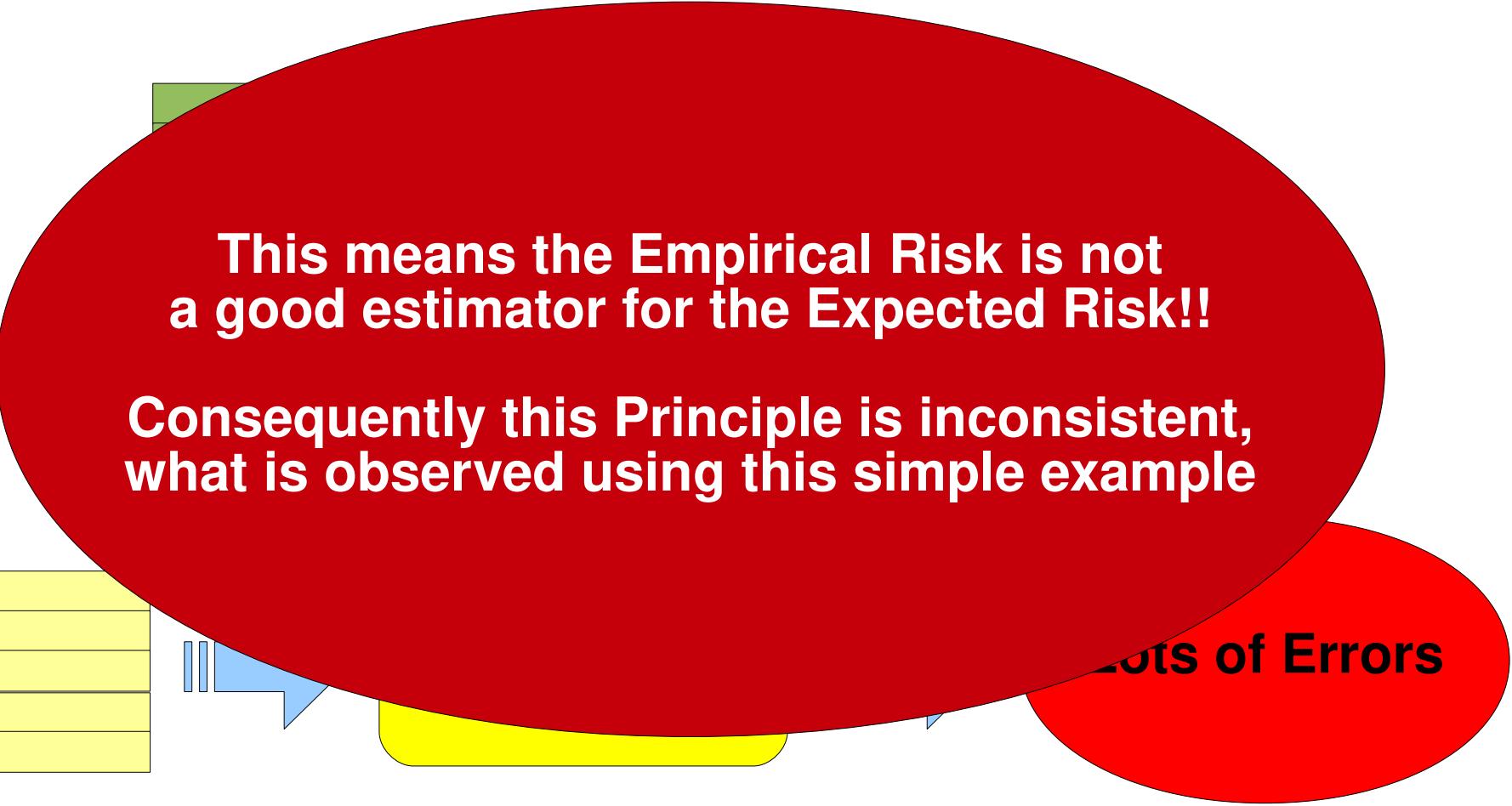
**This makes inconsistent  
the Principle of Empirical Risk Minimization!**

**Let's see another example...**

- A memory-based classifier



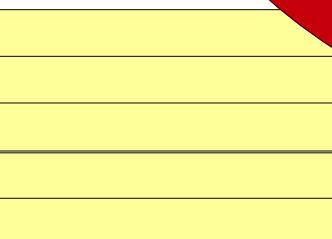
- A memory-based classifier



This means the Empirical Risk is not a good estimator for the Expected Risk!!

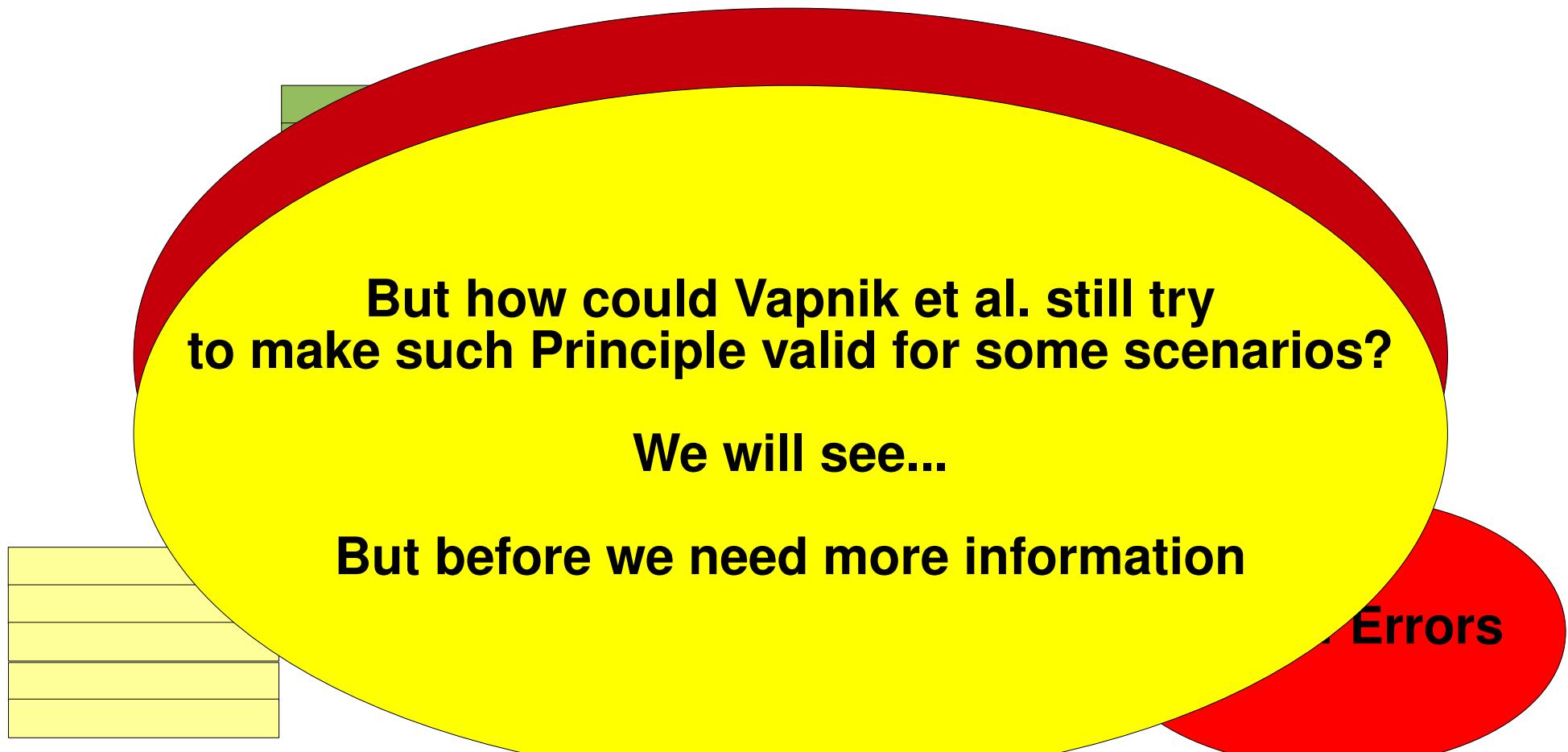
Consequently this Principle is inconsistent, what is observed using this simple example

Lots of Errors



Unseen  
Examples

- A memory-based classifier



Unseen  
Examples

- Then we have the concept of **Generalization** of a classifier  $f$  as follows:

$$|R(f_n) - R_{emp}(f_n)|$$

- In this manner, a classifier generalizes well when this difference is small, i.e., the **Empirical Risk** is close to the **Expected Risk**
- **A classifier with good generalization does not necessarily produce a small Empirical Risk nor even a small Expected Risk**

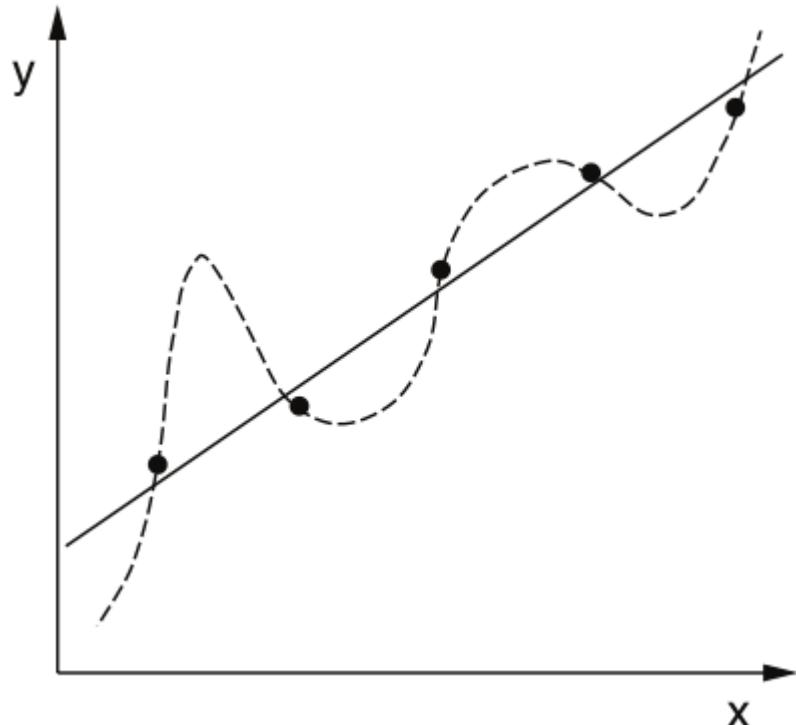
- Then we have the concept of **Generalization** of a classifier  $f$  as follows:

$$|R(f_n) - R_{emp}(f_n)|$$

- In this difference, the first term is the Expected Risk, which is the true risk of the classifier  $f$ , and the second term is the Empirical Risk, which is the risk on the training set. When this difference is small, it means that the classifier generalizes well.
- Therefore, a classifier with good generalization is the one whose Empirical Risk is a good estimator for the Expected Risk.

# Bias-Variance Dilemma

- For instance, consider the examples (points) collected during an experiment. Then, take two different functions to model them



**Which function is the best?**

To answer it we need to assess their **Expected Risks**

# Bias-Variance Dilemma

- For instance, consider the examples (points) collected during an experiment. Then, take two different functions to model them

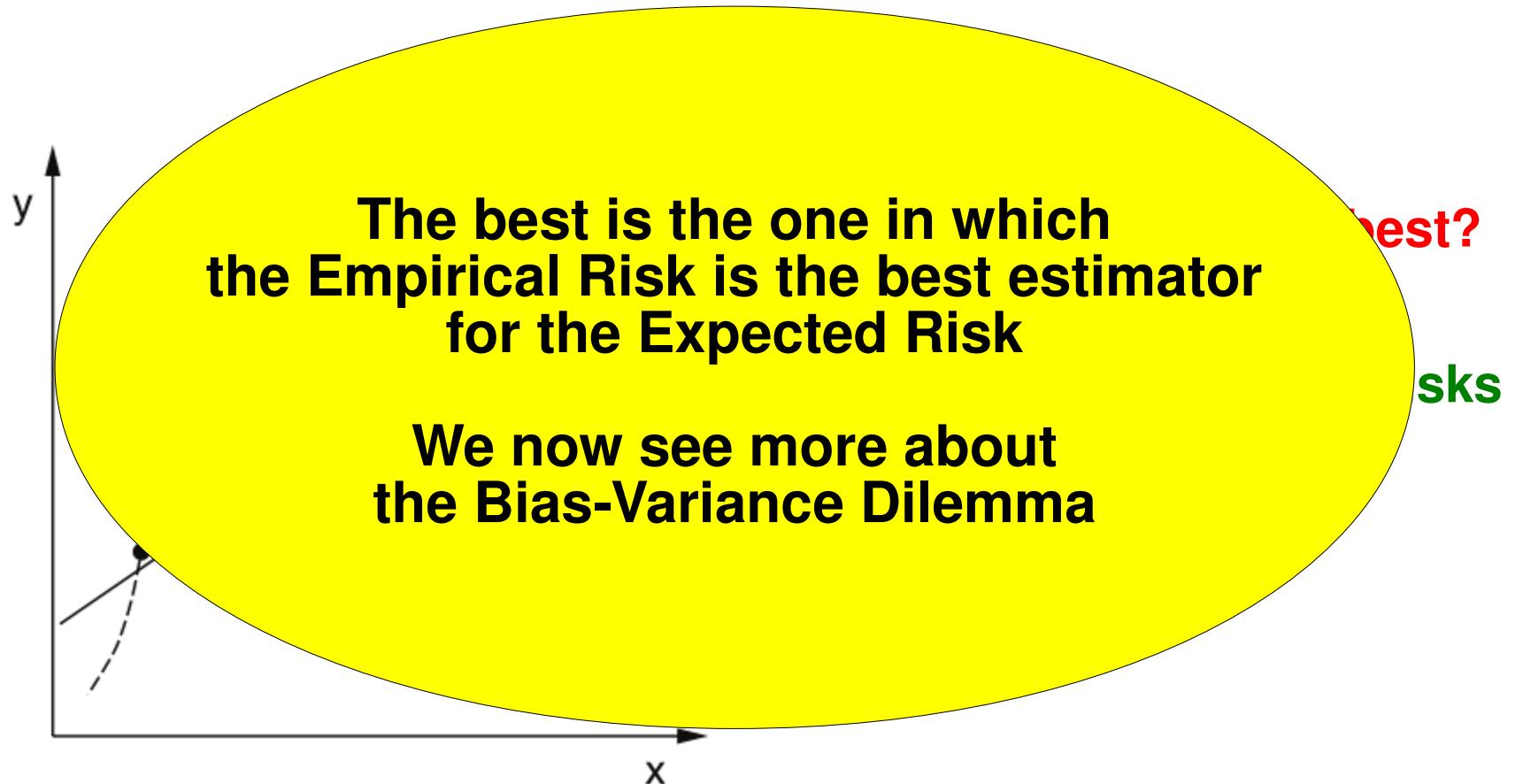
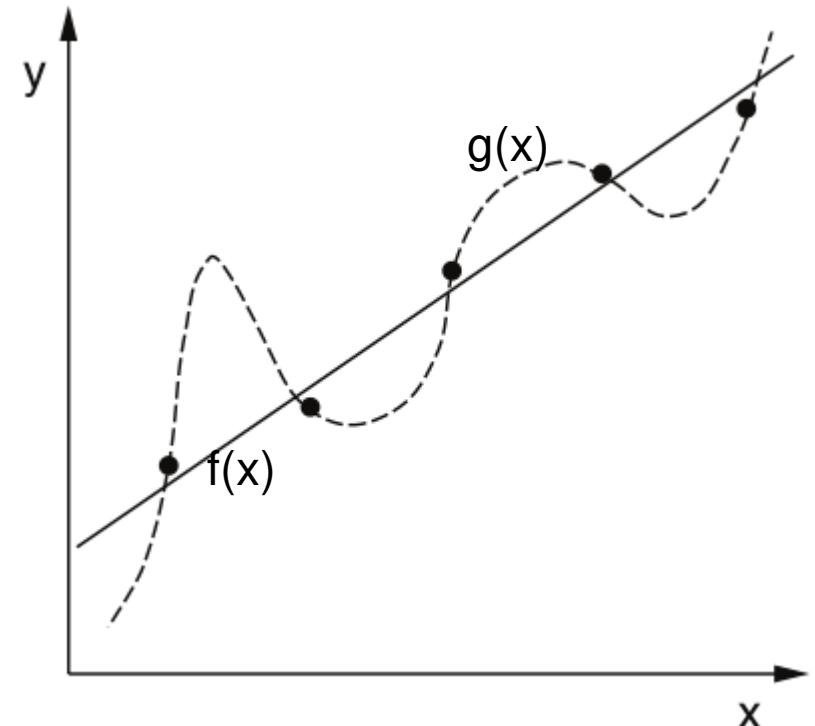


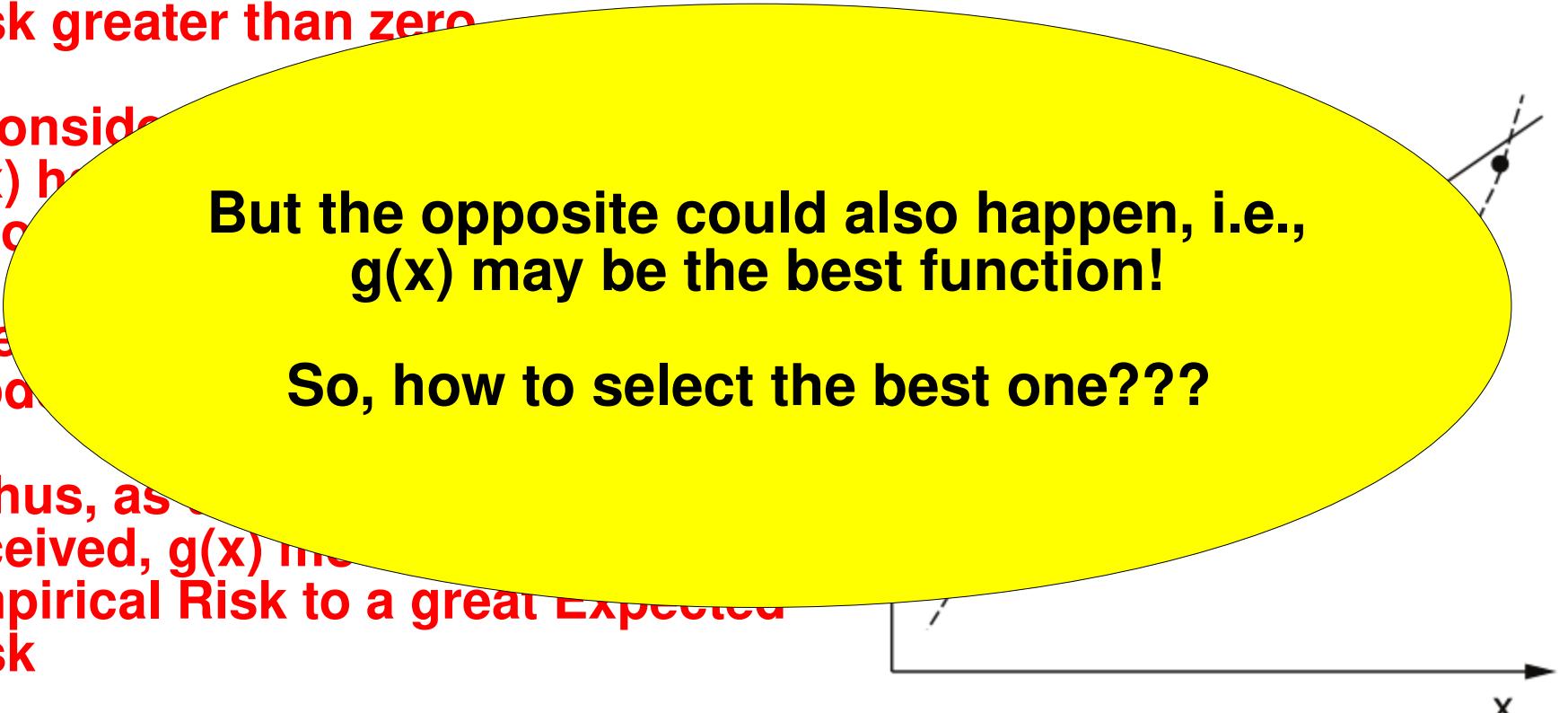
Figure from Luxburg and Scholkopf, Statistical Learning Theory: Models, Concepts, and Results

# Bias-Variance Dilemma

- But how to compute the Expected Risk if we only have one sample?
  - Consider that line  $f(x)$  has Empirical Risk greater than zero
  - Consider the polynomial function  $g(x)$  has Empirical Risk equals to zero
  - Let  $g(x)$  represent an overfitted model
  - Thus, as unseen examples are received,  $g(x)$  moves from no Empirical Risk to a great Expected Risk



# Bias-Variance Dilemma

- But how to compute the Expected Risk if we only have one sample?
    - Consider that line  $f(x)$  has Empirical Risk greater than zero
    - Consider that line  $g(x)$  has Empirical Risk less than zero
    - Let's say we have two models  $f(x)$  and  $g(x)$  which we received,  $f(x)$  incurs a great Empirical Risk to a great Expected Risk
- But the opposite could also happen, i.e.,  
 $g(x)$  may be the best function!
- So, how to select the best one???
- 

# Bias-Variance Dilemma

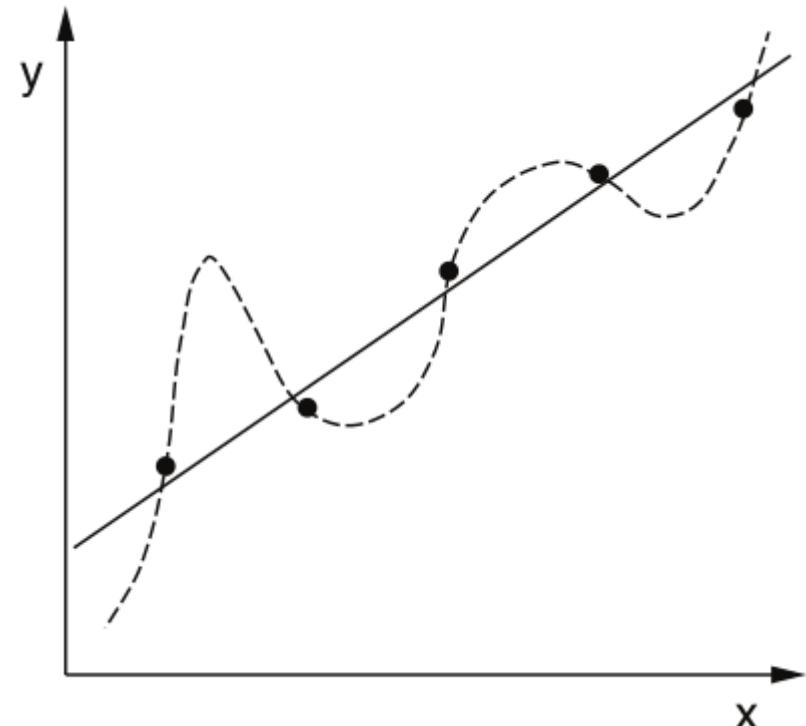
- Finally, which classifier should we choose?

1) The one with best fit with training data (the most complex one)?

2) Or the one that has greater Empirical Risk, however it was obtained using a simpler class of functions?

In Statistics, this is known as the Bias-Variance Dilemma

This Dilemma is addressed by SLT!



# Bias-Variance Dilemma

- Dilemma:
  - Bias:
    - If we assume a linear fit, only a linear classifier could be obtained
    - i.e., there is a strong bias imposed by ourselves
  - Variance:
    - If we fit a high-order polynomial function over training data, we can always have a perfect classifier for the sample
    - However this classifier is subject to greater fluctuations to unseen data

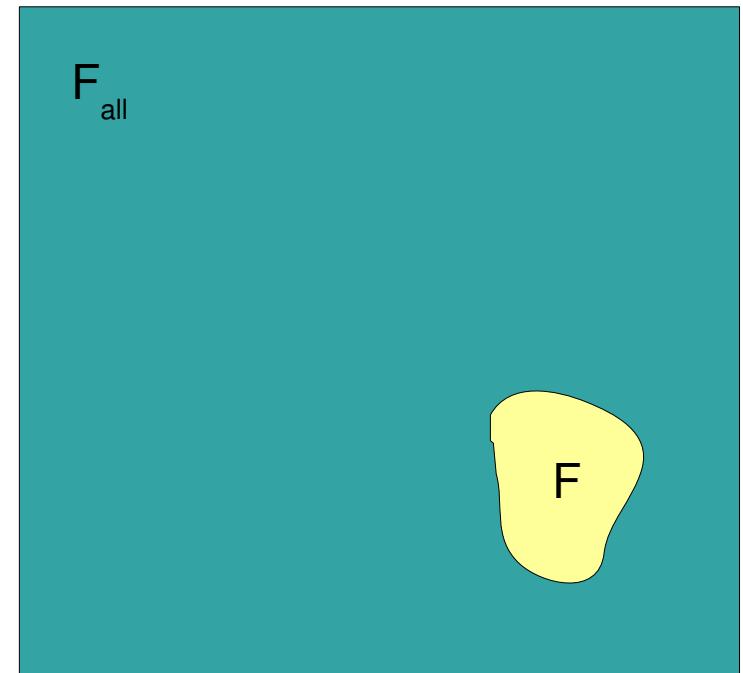
# Bias-Variance Dilemma

- This dichotomy associated to the Bias-Variance Dilemma is obvious when we consider the space of possible functions to build our classifier

Let space  $F_{\text{all}}$  contain all possible classification functions (or regression functions)

We could define a strong bias, i.e., a subspace  $F$  which contains only the linear functions to perform regression

This strong bias reduces the variance, i.e., it reduces the number of possible classifiers we can produce

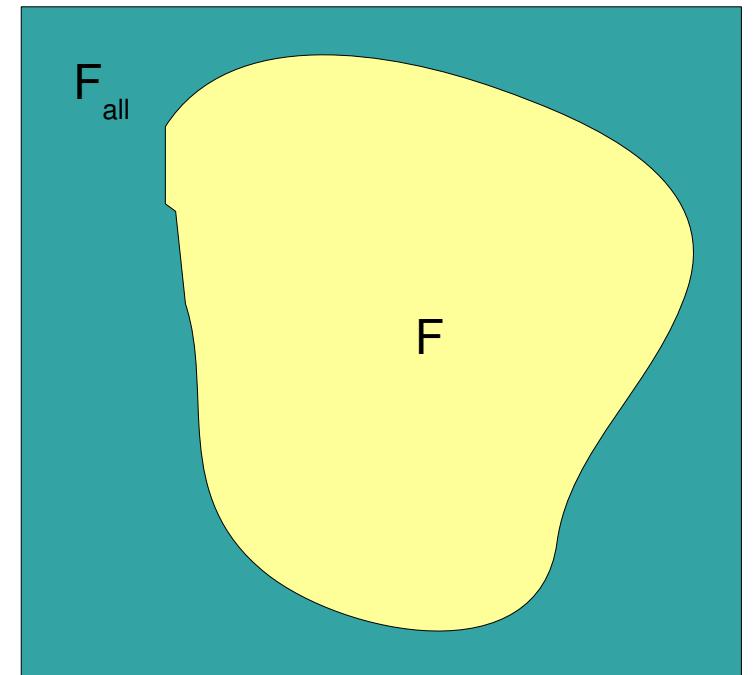


# Bias-Variance Dilemma

- This dichotomy associated to the Bias-Variance Dilemma is obvious when we consider the space of possible functions to build our classifier

**On the other hand, a weaker bias, i.e., subspace  $F$  contains many more functions to produce a classifier**

**In this case, variance is greater, i.e., the number of possible classifiers to select from and choose is huge!**



- Vapnik relates the concept of Generalization with Consistency
  - We know **Generalization** is defined as:
$$|R(f_n) - R_{emp}(f_n)|$$
  - **Consistency** is not a property of an individual function, but a property of a set of functions
    - As in Statistics, it allows us to study what happens in the limit of infinitely many sample points
  - This means a supervised learning algorithm should converge to the best classifier as possible, as the sample size increases

- Vapnik relates the concept of Generalization with Consistency
  - We know Generalization is a process of finding a classifier that generalizes well to new data.
  - After the Generalization,  
**Vapnik et al. relied on concepts to prove a classification algorithm starts with some classifier and tends to find the best one inside a subspace, as the sample size increases**
  - This means the algorithm should converge to the best classifier possible, as the sample size increases

- Vapnik relates the concept of Generalization with Consistency
  - We know Generalization is important

• Thus, there are two important cases considered by Vapnik et al.:

- To find a way to ensure that the Empirical Risk is close enough to the Expected Risk
- A supervised learning algorithm should converge to the best classifier inside its bias, as the sample size increases

- However, there are types of **Consistency** in the literature:
  - **Consistency with respect to subspace F**
    - A classification algorithm defines a bias, i.e., a subspace of functions it considers to produce a classifier  $f$
    - Such algorithm is consistent with respect to subspace  $F$  when it converges to the best classifier inside its subspace as the sample size increases

- However, there are types of **Consistency** in the literature:
  - **Consistency with respect to subspace F**
    - A classification algorithm defines a bias, i.e., a subspace of functions it considers to produce a classifier  $f$
    - Such algorithm is consistent with respect to subspace  $F$  when it converges to the best classifier inside its subspace as the sample size increases
  - **Bayes consistent**
    - Inside space  $F_{\text{all}}$  there is the ideal classifier (also known as  $f_{\text{Bayes}}$  or the Bayes classifier), which is the best of all
    - An algorithm is Bayes consistent when it converges to this ideal classifier as the sample size increases

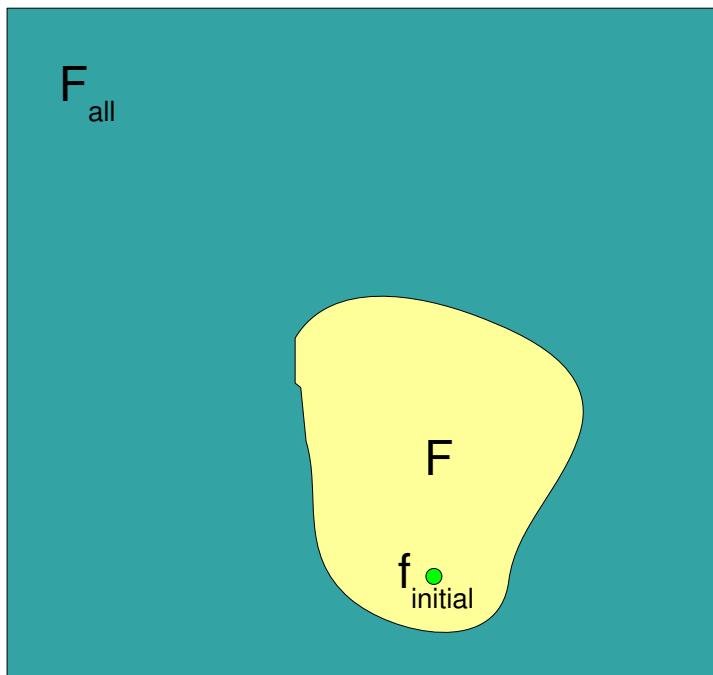
- However, there are types of **Consistency** in the literature:
  - **Consistency with respect to subspace F**
    - The best classifier inside subspace F is given by:

$$f_{\mathcal{F}} = \arg \min_{f \in \mathcal{F}} R(f)$$

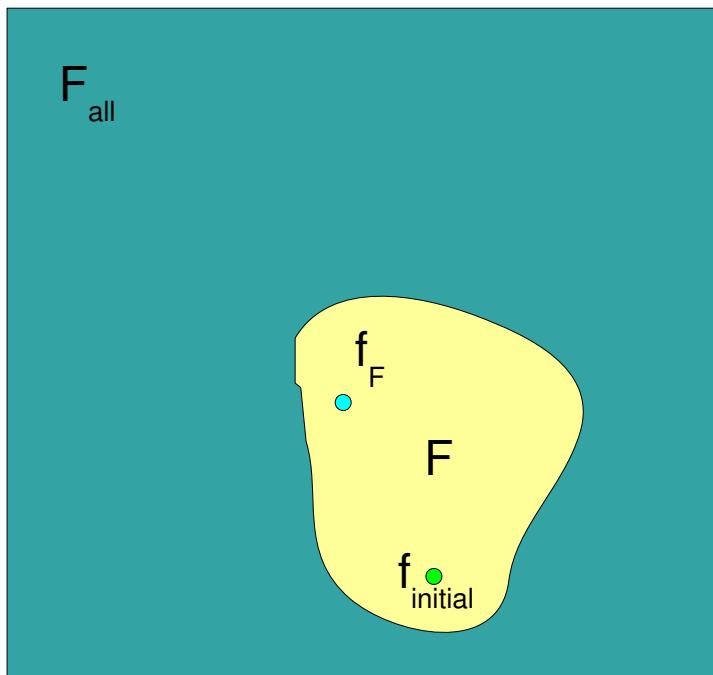
- **Bayes consistent**
  - The Bayes classifier is given by:

$$f_{Bayes} = \arg \min_{f \in \mathcal{F}_{all}} R(f)$$

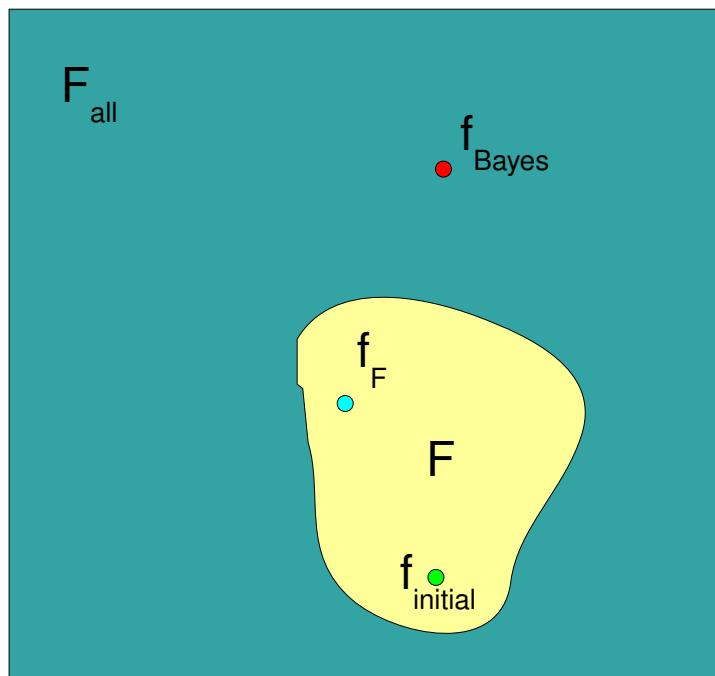
- Consistency with respect to subspace  $F$



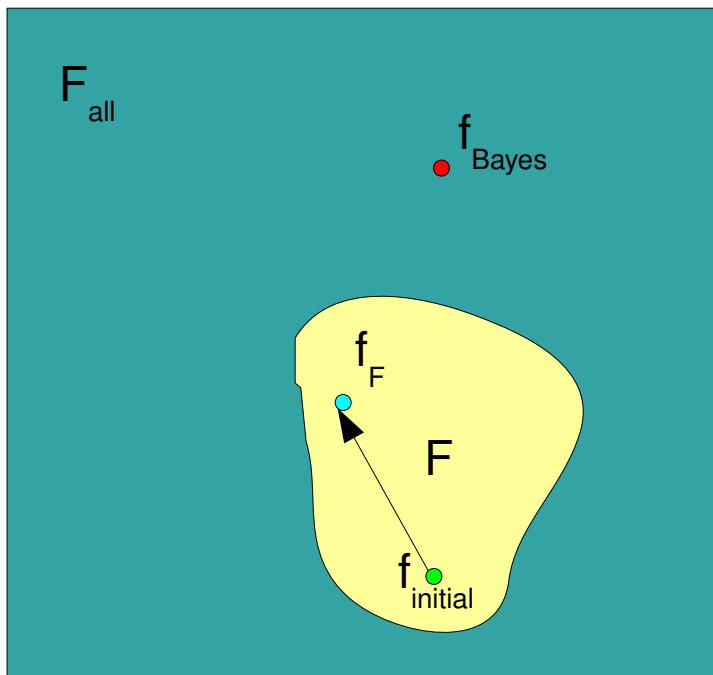
- Consistency with respect to subspace  $F$



- Consistency with respect to subspace  $F$



- Consistency with respect to subspace  $F$



- A classification algorithm is said to be **consistent with respect to Bayes considering samples are drawn independently from some probability distribution P:**
  - If the Expected Risk  $R(f_n)$  of a classifier  $f_n$  probabilistically converges to the Expected Risk  $R(f_{Bayes})$  of the Bayes classifier, as the sample size increases

$$P(R(f_n) - R(f_{Bayes}) > \varepsilon) \rightarrow 0 \text{ conforme } n \rightarrow \infty$$

- Observe this is valid for a particular distribution P, which is our joint probability distribution

- We say a classification algorithm is **consistent with respect to  $F$  considering samples are drawn independently from some probability distribution  $P$ :**
  - If the Expected Risk of a classifier  $f_n$ , i.e.,  $R(f_n)$ , probabilistically converges to the expected risk  $R(f_F)$  of the best classifier in  $F$ , as the sample size increases:

$$P(R(f_n) - R(f_F) > \varepsilon) \rightarrow 0 \text{ conforme } n \rightarrow \infty$$

- Observe this is valid for a particular distribution  $P$ , which is our joint probability distribution
  - **Why we did not use the absolute value?**

- We say a classification algorithm is **universally consistent with respect to F**:
  - If it is consistent with respect to subspace F
  - For all probability distributions P
- **This is a stronger notion of consistency:**
  - In supervised learning, the algorithm must be consistent for any joint probability distribution
    - It makes little sense that an algorithm is consistent for only a given distribution

- We say a classification algorithm is **universally consistent with respect to  $F$** :
  - If it is consistent for all  $F \in \mathcal{F}$
  - For all  $F \in \mathcal{F}$
- This is the consistency that matters to the SLT!

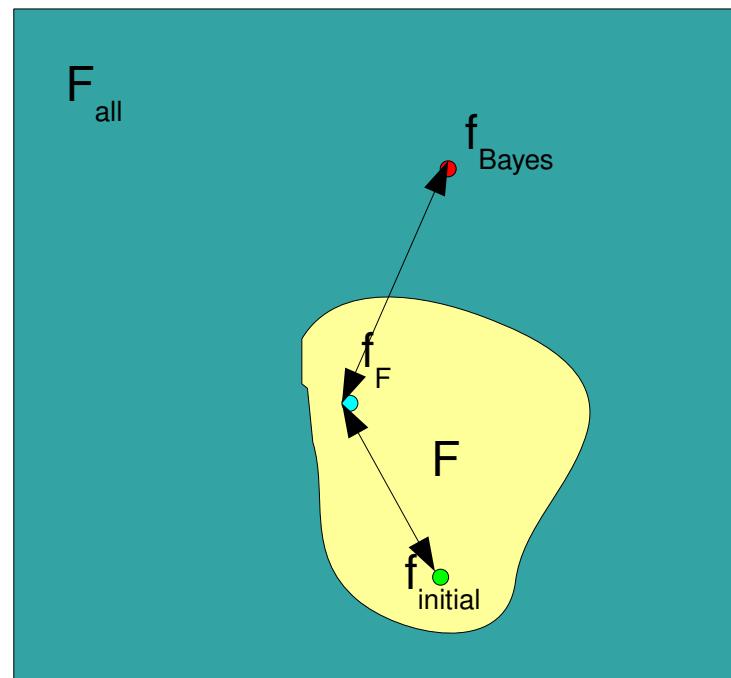
Recall we assumed we do not know the joint probability distribution  $P!!!$

ant

for

# Consistency versus Learning errors

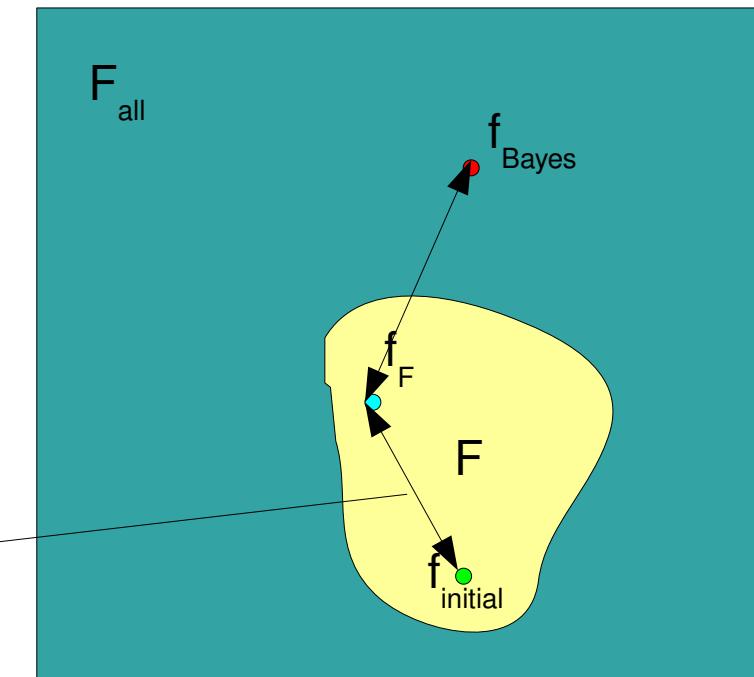
- Relationship among Consistency and Learning errors



# Consistency versus Learning errors

- Relationship among Consistency and Learning errors

**Estimation error:**  
how far our solution  
is from the best classifier in  $F$

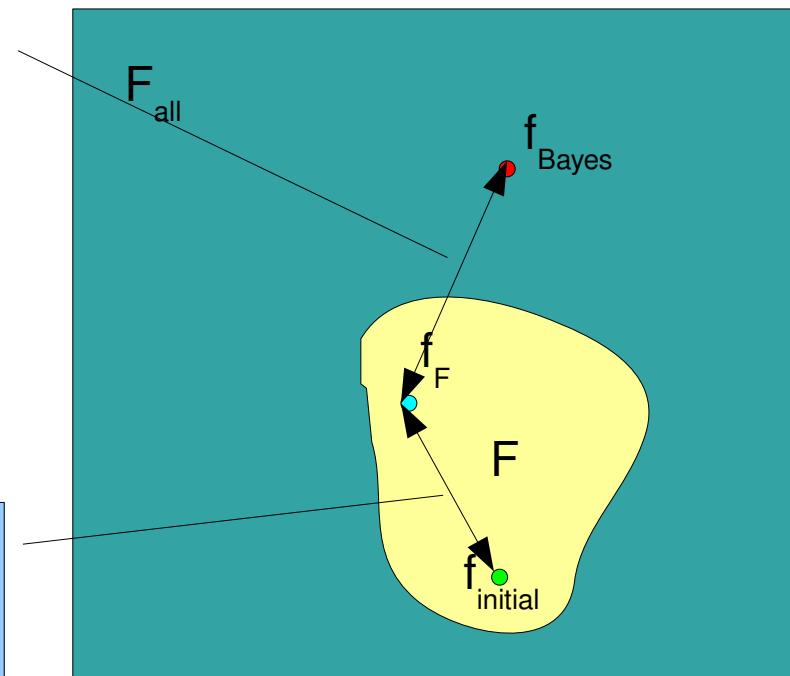


# Consistency versus Learning errors

- Relationship among Consistency and Learning errors

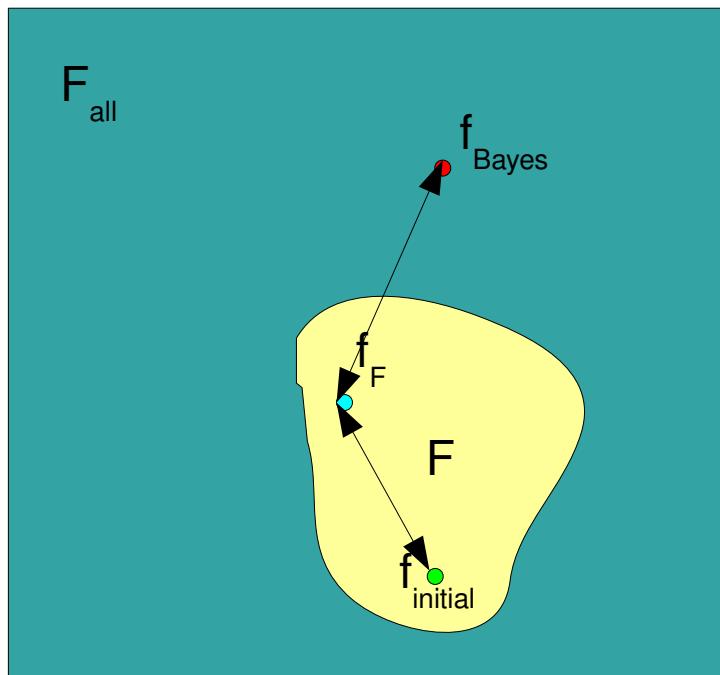
**Approximation error:**  
how far the best  
solution in  $F$  is  
from the best classifier at all

**Estimation error:**  
how far our solution is  
from the best classifier in  $F$



# Consistency versus Learning errors

- The total error is defined in terms of the approximation and estimation errors



$$R(f_n) - R(f_{Bayes}) = \underbrace{(R(f_n) - R(f_F))}_{\text{Estimation error}} + \underbrace{(R(f_F) - R(f_{Bayes}))}_{\text{Approximation error}}$$

# Consistency versus Learning errors

- Thus:
  - If we choose a too large subspace  $F$ , the approximation error tends to become small, however the estimation error will be too large
    - The estimation error tends to be too large, because this space contains more complex functions and, therefore, they tend to **overfit** data
  - The opposite happens when subspace  $F$  is too small

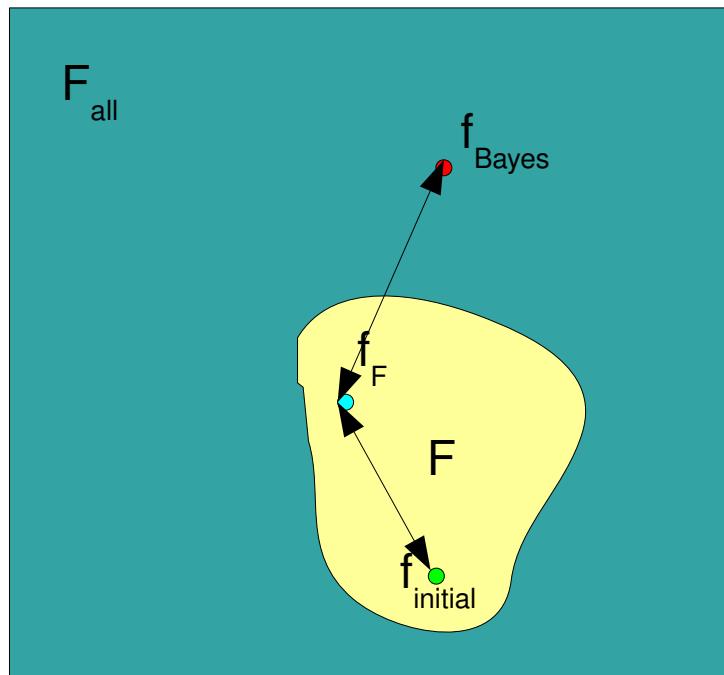
$$R(f_n) - R(f_{Bayes}) = \underbrace{(R(f_n) - R(f_F))}_{\text{Estimation error}} + \underbrace{(R(f_F) - R(f_{Bayes}))}_{\text{Approximation error}}$$

# Consistency versus Learning errors

- Estimation error:
  - It is the result of uncertainty present in training data
  - In Statistics, this error is also called **variance**
- Approximation error:
  - It is the result of the algorithm bias
  - In Statistics, this error is also called **bias**

# Consistency versus Learning errors

- Now we observe that by defining a subspace  $F$ , we define the balance between the estimation and approximation errors:
  - A stronger bias produces low variance or a small estimation error, however it leads to a large approximation error
  - A weaker bias produces great variance or a large estimation error, however it leads to a small approximation error



$$R(f_n) - R(f_{Bayes}) = \underbrace{(R(f_n) - R(f_F))}_{\text{Estimation error}} + \underbrace{(R(f_F) - R(f_{Bayes}))}_{\text{Approximation error}}$$

# Underfitting and Overfitting

- From this perspective we define:
  - **Underfitting** – If subspace  $F$  is small, then the estimation error is small, but the approximation error is large
  - **Overfitting** – If subspace  $F$  is large, then the estimation error is large, but the approximation error is small
- The best for supervised learning is the balance

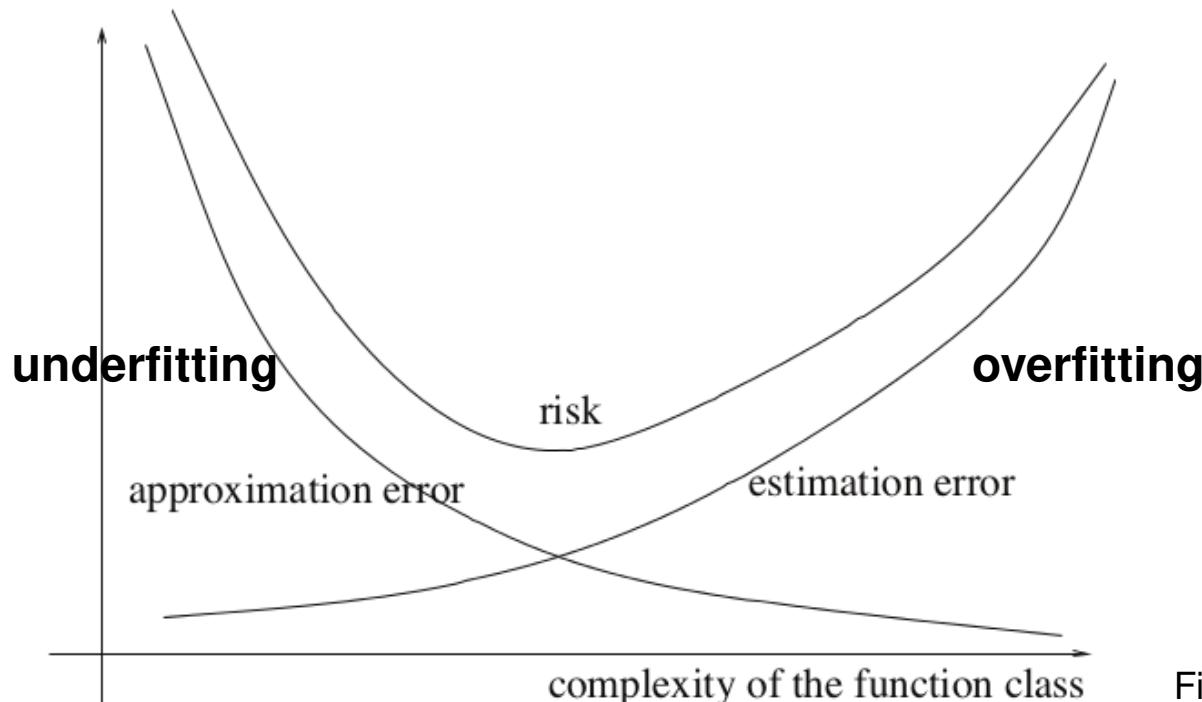
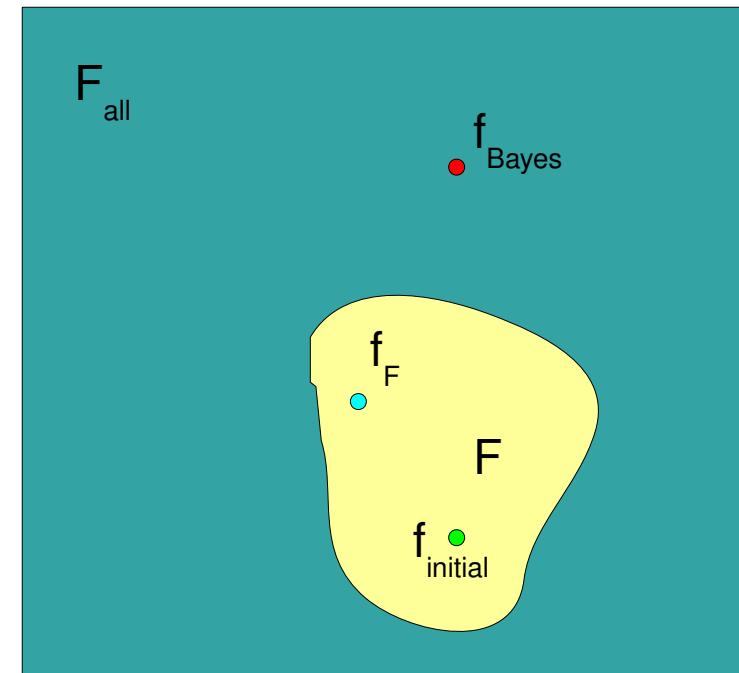


Figure from Luxburg and Scholkopf, Statistical Learning Theory: Models, Concepts, and Results

- Some observations about bias:

The bias of an algorithm is influenced by:

- 1) Hypothesis bias – it defines the representation language of hypothesis or models. For example: if-then rules, decision trees, artificial neural networks, etc.
- 2) Preference bias – it defines the conditions under which the algorithm prefers one classifier over another
- 3) Search bias – heuristics or search criterion to look for more solutions in subspace  $F$



## Starting the formalization of the SLT

- To reach the best classifier, we need the Expected Risk:

$$R(f) < R(g)$$

- Observe that all **Consistency** definitions we saw rely on the Expected Risk of a classifier
  - But there is no way to compute it! Only if we know  $P(X, Y)$

# Starting the formalization of the SLT

- To reach the best classifier, we need the Expected Risk:

$$R(f) < R(g)$$

- Observe that all **Consistency** definitions we saw rely on the Expected Risk of a classifier
  - But there is no way to compute it! Only if we know  $P(X, Y)$
  - However, the Empirical Risk is our main estimator for the Expected Risk
    - But how to rely on the Empirical Risk once there might have overfitting on training data? E.g. memory-based classifier.
    - From now on we will see how Vapnik et al. used the Empirical Risk as a way to verify the **universal consistency for classification algorithms with respect to subspace F**

# Starting the formalization of the SLT

- To reach the best classifier, we need the Expected Risk:

$$R(f) < R(g)$$

- Observe that all we have seen so far relied on the Expected Risk
- But there is a problem with the Expected Risk
- However, Vapnik et al. came up with the Principle of Empirical Risk Minimization
- But how to avoid overfitting on the training data? A classifier might have zero risk on the training data but still be bad.
- From now on we will focus on the Empirical Risk as a way to verify the universal consistency for classification algorithms with respect to subspace  $F$

**This is the way Vapnik et al. came up  
with the Principle of  
Empirical Risk Minimization**

# Empirical Risk Minimization Principle

- The problem consists in finding a classifier  $f$  that minimizes the Expected Risk:

$$R(f) = E(l(x, y, f(x)))$$

- However:
  - There is no way to compute  $R(f)$ , because we do not know the joint probability distribution  $P$
  - Therefore, we could estimate the Expected Risk using the Empirical Risk and minimize it to find  $f$ :

$$R_{emp}(f) = \frac{1}{n} \sum_{i=1}^n l(x_i, y_i, f(x_i))$$

# Empirical Risk Minimization Principle

- Thus, by assuming:
  - A function subspace  $F$  and
  - A loss function
- Vapnik et al. propose the estimation of the Expected Risk using the Empirical Risk and, in this way, we look for a classifier  $f_n$  such as:

$$f_n = \operatorname{argmin}_{f \in F} R_{emp}(f)$$

- Vapnik used the **Law of Large Number** from Statistics as main motivation for that...

# Empirical Risk Minimization Principle

- Vapnik observed it was possible to use the **Law of Large Numbers**:
  - A Theorem from Statistics
  - According to this law, assuming data are identically and independently sampled from any probability distribution  $P$ , the average of a sample converges to the expected value as the sample size increases:

$$\frac{1}{n} \sum_{i=1}^n \xi_i \rightarrow E(\xi) \quad \text{para } n \rightarrow \infty$$

# Empirical Risk Minimization Principle

- Then Vapnik assumed the Empirical Risk of some classifier  $f$  converges to the Expected Risk of  $f$  as the sample size tends to infinity:

$$R_{emp}(f) = \frac{1}{n} \sum_{i=1}^n l(x_i, y_i, f(x_i)) \rightarrow E(l(x, y, f(x)))$$

Para  $n \rightarrow \infty$

# Empirical Risk Minimization Principle

- Chernoff (1952) proposed an inequality, extended by Hoeffding (1963), that characterizes how well the empirical average approximates to the expected value:

$$P\left(\left|\frac{1}{n} \sum_{i=1}^n \xi_i - E(\xi)\right| \geq \epsilon\right) \leq 2 \exp(-2n\epsilon^2)$$

Sendo  $\xi_i$  valores aleatórios no intervalo  $[0, 1]$

- According to this theorem:
  - The probability of the sampling average deviates more than epsilon from the expected value is limited by a small quantity defined on the right side of the inequality
  - Observe that as n increases, the right-side term gets smaller

# Empirical Risk Minimization Principle

- In this manner, we can apply Hoeffding inequality:

$$P\left(\left|\frac{1}{n} \sum_{i=1}^n \xi_i - E(\xi)\right| \geq \epsilon\right) \leq 2 \exp(-2n\epsilon^2)$$

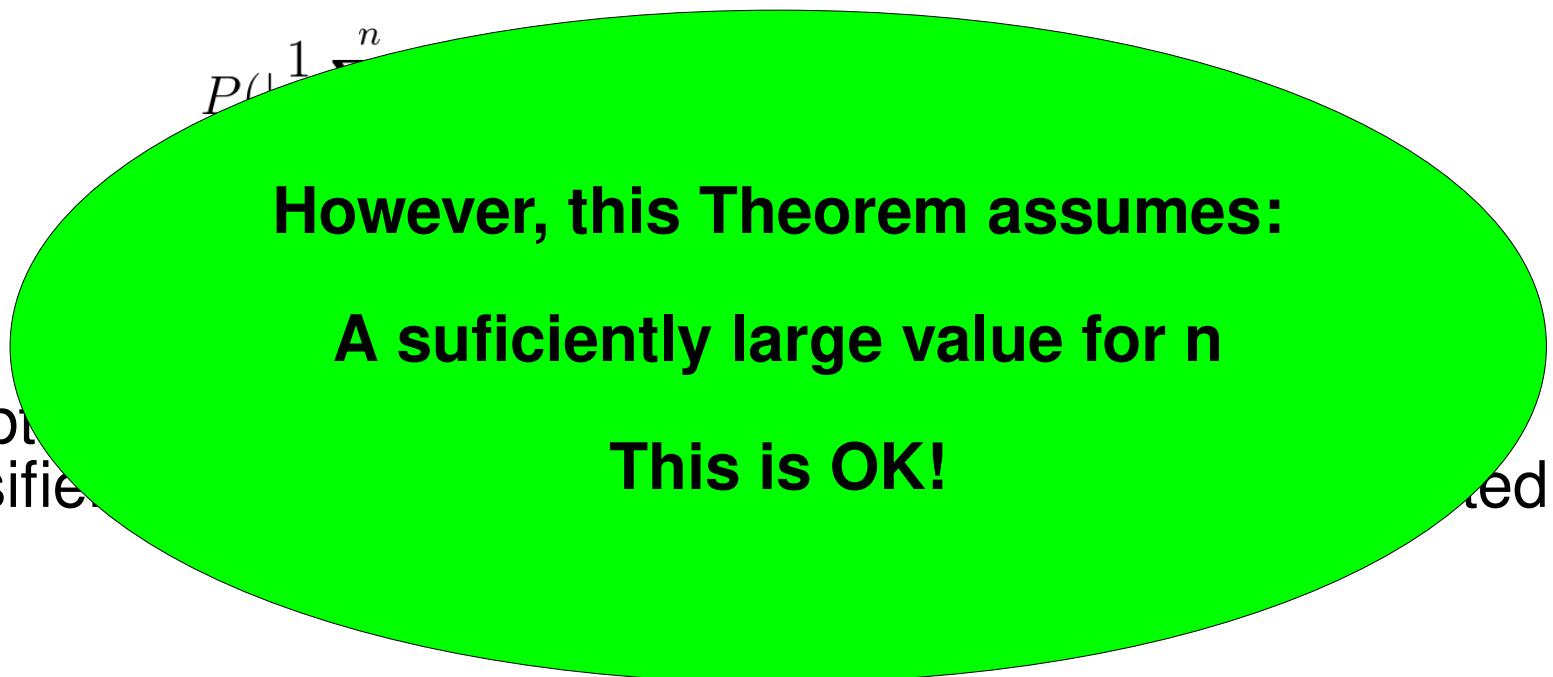
Sendo  $\xi_i$  valores aleatórios no intervalo  $[0, 1]$

- To obtain a threshold which defines how much, for a given classifier  $f$ , the Empirical Risk approximates to the Expected Risk:

$$P(|R_{emp}(f) - R(f)| \geq \epsilon) \leq 2 \exp(-2n\epsilon^2)$$

# Empirical Risk Minimization Principle

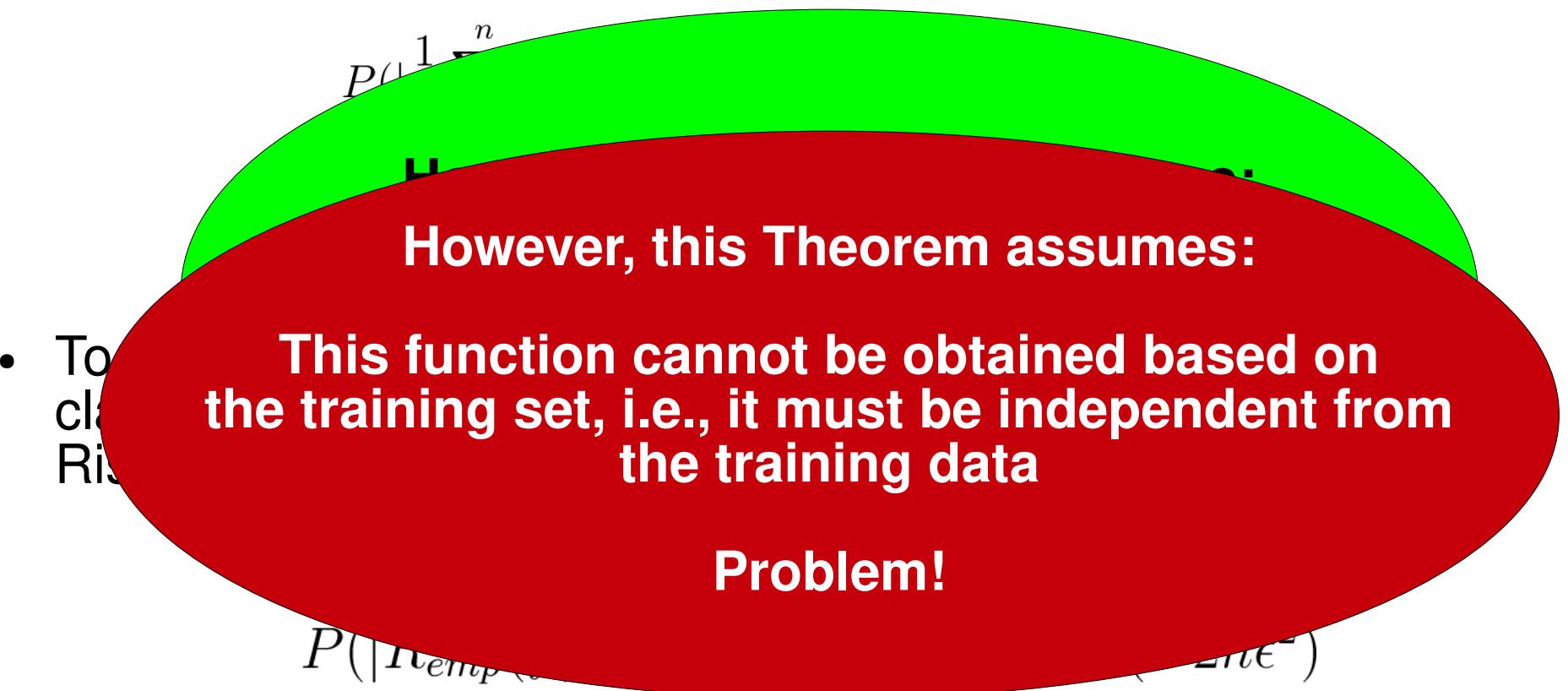
- In this manner, we can apply Hoeffding inequality:



$$P(|R_{emp}(f) - R(f)| \geq \epsilon) \leq 2 \exp(-2n\epsilon^2)$$

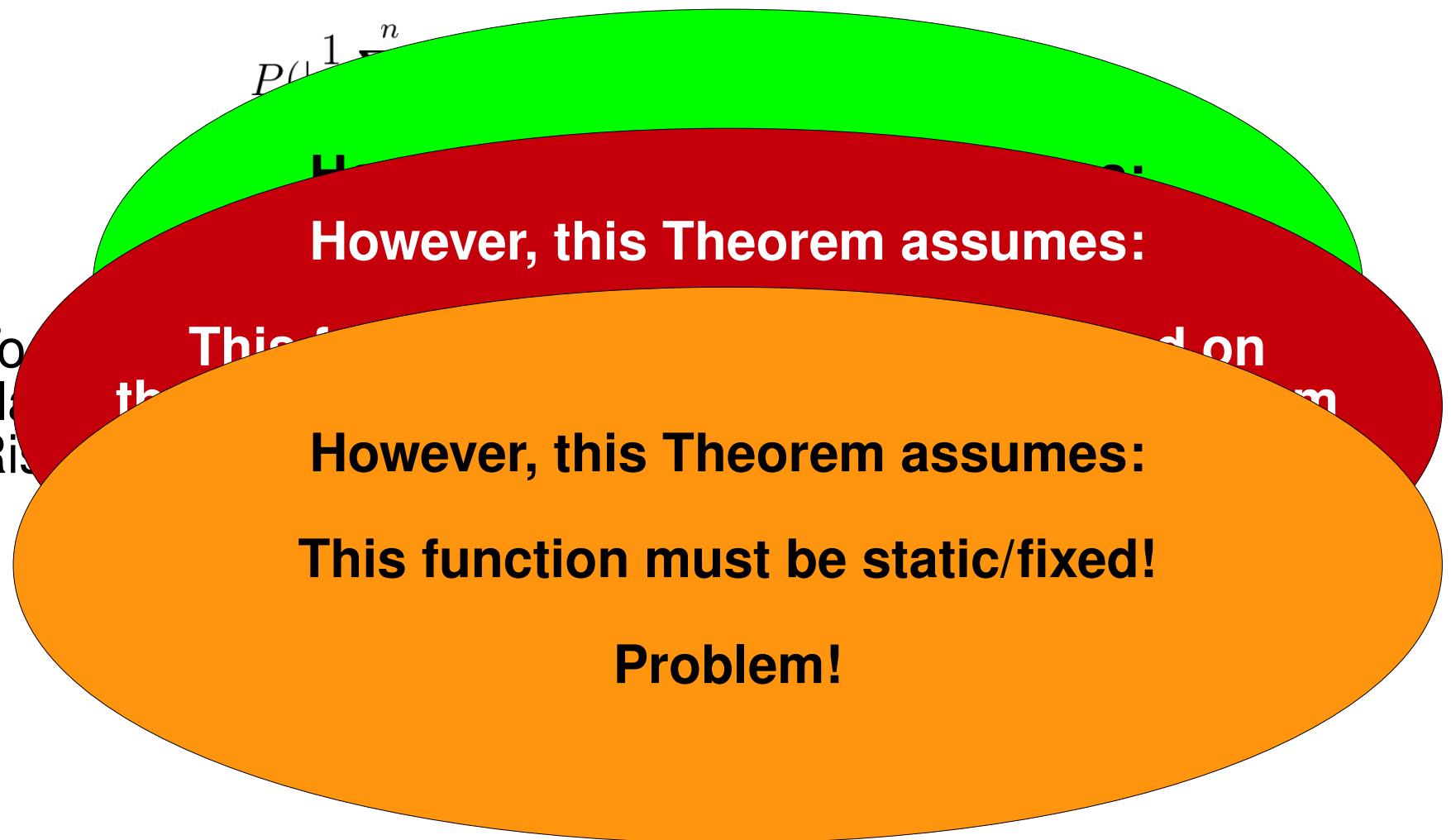
# Empirical Risk Minimization Principle

- In this manner, we can apply Hoeffding inequality:



# Empirical Risk Minimization Principle

- In this manner, we can apply Hoeffding inequality:



# Empirical Risk Minimization Principle

- However, there are constraints to employ the Chernoff bound:
  - **It is only valid for a fixed function or classifier  $f$**
  - **The classifier must be selected independently of the training set**
  - However, in supervised learning:
    - Function  $f$  is obtained based on the training set
    - We start with an initial function and attempt to converge to the best classifier inside the bias (subspace) of the Classification Algorithm
    - At first this makes improper the use of the Law of Large Numbers to prove the Empirical Risk can be used to estimate the Expected Risk

# Empirical Risk Minimization Principle

- However, there are constraints to employ the Chernoff bound:

- **It is only valid for a fixed function or classifier  $f$**
- **The classifier  $f$  must be independent of the training set**
- However

**What would lead to the inconsistency  
of the ERM Principle**

**But Vapnik did not give up...**

- A few years later, Vapnik and Chervonenkis developed a new method to estimate the generalization error.

## Inconsistency related to ERM Principle

- For example, consider the space of examples versus class labels which is produced using a deterministic function:

$$Y = \begin{cases} -1 & \text{if } X < 0.5 \\ 1 & \text{if } X \geq 0.5 \end{cases}$$

- Let a classifier  $f$  that produces zero error for the training set, because it simply memorizes all answers  $Y$  for every input sample  $X$  as follows:

$$f_n(X) = \begin{cases} Y_i & \text{if } X = X_i \text{ for some } i = 1, \dots, n \\ 1 & \text{otherwise.} \end{cases}$$

## Inconsistency related to ERM Principle

- The Empirical Risk of this classifier  $f$  is zero!
  - However, for unseen examples, it always assigns a fixed label, let's say 1
  - Consider that half of examples will have such class, then it will hit in 50% of cases and miss in the other 50%

## Inconsistency related to ERM Principle

- The Empirical Risk of this classifier  $f$  is zero!
  - However, for unseen examples, it always assigns a fixed label, let's say 1
  - Consider that half of examples will have such class, then it will hit in 50% of cases and miss in the other 50%
  - In this scenario, the Empirical Risk of  $f$  will not approximate the Expected Risk of  $f$ 
    - i.e., the ERM Principle is inconsistent for this scenario
      - **Assuming the classifier was obtained based on the training set**

## Inconsistency related to ERM Principle

- This is an extreme situation of **overfitting**
  - i.e., the classifier  $f$  is optimal to the training set, but it miserably fails for unseen examples
    - Observe that if we have 2 options, 50% is basically obtained by “guessing” either class -1 or +1 for unseen examples

## Inconsistency related to ERM Principle

- This is an extreme situation of **overfitting**
  - i.e., the classifier  $f$  is optimal to the training set, but it miserably fails for unseen examples
    - Observe that if we have 2 options, 50% is basically obtained by “guessing” either class -1 or +1 for unseen examples
- However, we only have the Empirical Risk to work on and estimate the Expected Risk:
  - Then how could we, in some way, recall the ERM Principle and make it work as a good estimator for the Expected Risk?

# Inconsistency related to ERM Principle

- Then how could we, in some way, recall the ERM Principle and make it work as a good estimator for the Expected Risk?
  - Yes, but in order to make it we need to evaluate the class of functions or subspace  $F$  contained in  $F_{\text{all}}$ 
    - Given  $F$  is the bias of our learning algorithm

# Inconsistency related to ERM Principle

- Then how could we, in some way, recall the ERM Principle and make it work as a good estimator for the Expected Risk?
  - Yes, but in order to make it we need to evaluate the class of functions or subspace  $F$  contained in  $F_{\text{all}}$ 
    - Given  $F$  is the bias of our learning algorithm
  - Actually, the ERM Principle must be consistent for every function in  $F$ , because any one of them may be selected as classifier

# Inconsistency related to ERM Principle

- Then how could we, in some way, recall the ERM Principle and make it work as a good estimator for the Expected Risk?
  - Yes, but in order to make it we need to evaluate the class of functions or subspace  $F$  contained in  $F_{\text{all}}$ 
    - Given  $F$  is the bias of our learning algorithm
  - Actually, the ERM Principle must be consistent for every function in  $F$ , because any one of them may be selected as classifier
  - If we consider a subspace  $F$  that contains a memory-based function, then this principle will NEVER work!
  - Therefore, we need to restrict subspace  $F$  used to estimate our classifier  $f$ 
    - Otherwise we cannot ensure learning
    - Fortunately we have approaches to restrict subspace  $F$

## Bringing Consistency to the ERM Principle

- The conditions to make the ERM Principle consistent involve the **restriction of the space of admissible functions**
  - i.e., the reduction of subspace  $F$  which contains the available functions or classifiers we will consider

# Bringing Consistency to the ERM Principle

- The conditions to make the ERM Principle consistent involve the **restriction of the space of admissible functions**
  - i.e., the reduction of subspace  $F$  which contains the available functions or classifiers we will consider
- So we can consider the **worst-case scenario for all available classifiers  $f$  in subspace  $F$** 
  - This because any of those classifiers can be selected by our learning algorithm
  - **Thus we verify if, for all functions in subspace  $F$ , the Empirical Risk converges to the Expected Risk as the sample size tends to infinity**

# Bringing Consistency to the ERM Principle

- The conditions to make the ERM Principle consistent involve the **restriction of the space of admissible functions**
  - i.e., the reduction of subspace  $F$  which contains the available functions or classifiers we will consider
- So we can **consistently classify**
  - Then it converges for the worst function in  $F$ , then it will converge for the other functions in  $F$
  - Then we can consistently estimate the error on the sample

## Bringing Consistency to the ERM Principle

- An approach to ensure convergence for every function  $f$  in subspace  $F$  is by using the **uniform convergence over  $F$** :
  - Thus, as  $n$  (size of the training set) increases, the difference below must decrease
- Considering every  $f$  in subspace  $F$
- As result, for a given  $n$ , the following inequality will be valid for a small epsilon:

$$|R_{emp}(f) - R(f)| \leq \epsilon \text{ para toda função } f \in F, F \subset F_{all}$$

# Bringing Consistency to the ERM Principle

- This means the distance between the Empirical and the Expected Risk curves, for all functions  $f$  in subspace  $F$ , will never be greater than a given epsilon

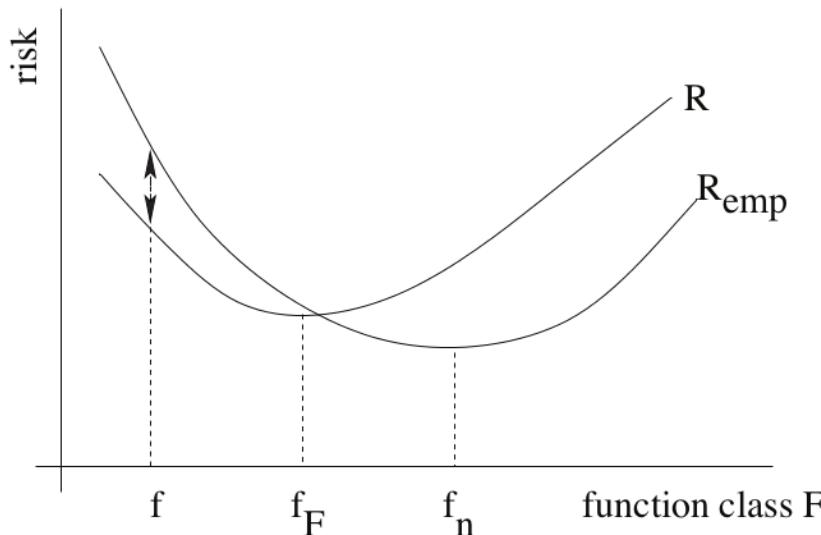


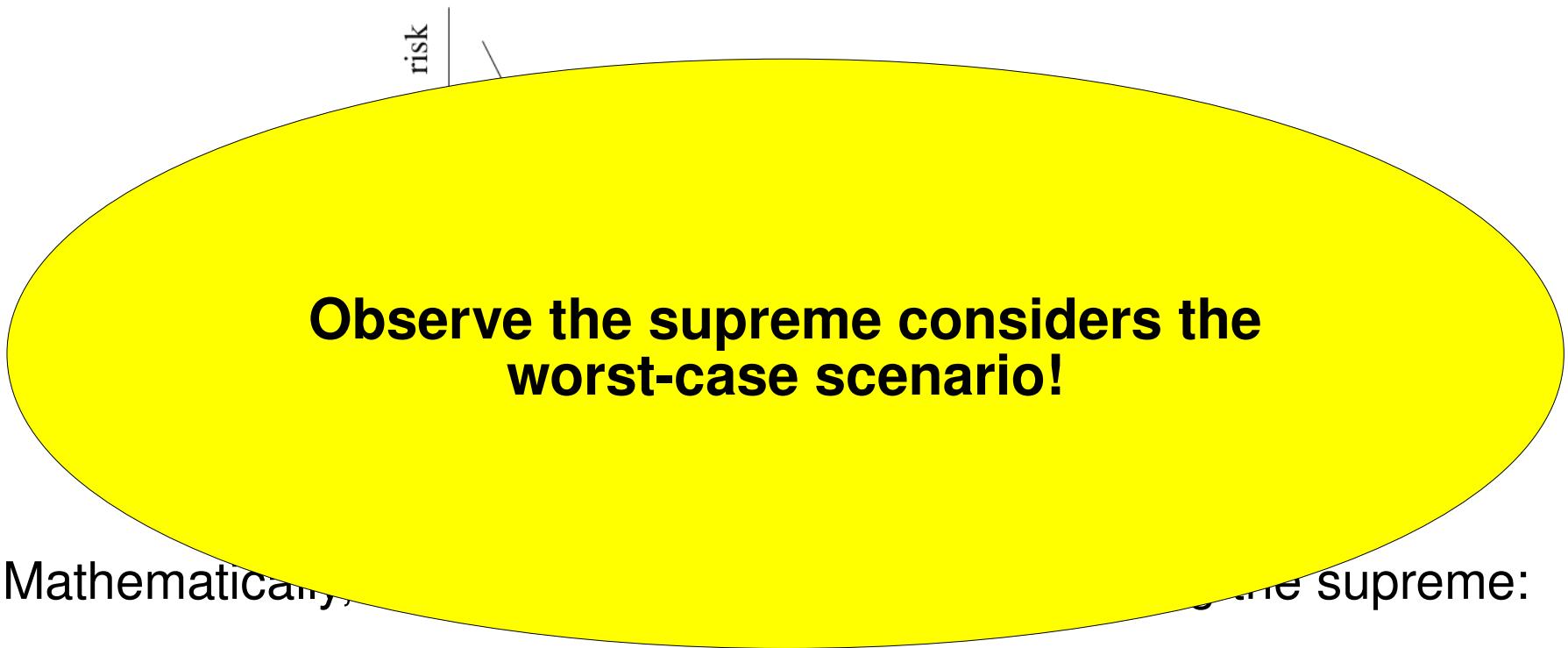
Figure obtained from Luxburg and Scholkopf, Statistical Learning Theory: Models, Concepts, and Results

- Mathematically, we can represent such concept using the supreme:

$$\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| \leq \varepsilon.$$

# Bringing Consistency to the ERM Principle

- This means the distance between the Empirical and the Expected Risk curves, for all functions  $f$  in subspace  $F$ , will never be greater than a given epsilon



$$\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| \leq \varepsilon.$$

# Bringing Consistency to the ERM Principle

- Thus, the inequality below expresses all differences between the Empirical and the Expected Risks, for every function  $f$  in  $\mathcal{F}$ , which is limited by the supreme (right-side term):

$$|R(f) - R_{\text{emp}}(f)| \leq \sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)|$$

- Then Vapnik applied probability on both sides:

$$P(|R(f_n) - R_{\text{emp}}(f_n)| \geq \varepsilon) \leq P(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| \geq \varepsilon)$$

## Bringing Consistency to the ERM Principle

- On the right side of the inequality below, we have exactly the situation in which the **Law of Large Numbers deals with, i.e., considering a set of fixed functions** (all classifiers in  $\mathcal{F}$ ):

$$P(|R(f_n) - R_{\text{emp}}(f_n)| \geq \varepsilon) \leq P\left(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| \geq \varepsilon\right)$$

## Bringing Consistency to the ERM Principle

- On the right side of the inequality below, we have exactly the situation in which the **Law of Large Numbers deals with, i.e., considering a set of fixed functions** (all classifiers in  $\mathcal{F}$ ):

$$P(|R(f_n) - R_{\text{emp}}(f_n)| \geq \varepsilon) \leq P(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| \geq \varepsilon)$$

- Therefore, we can say that the Law of Large Numbers is uniformly valid over a set of functions (or subspace  $\mathcal{F}$ ) for every epsilon greater than zero (remember epsilon is a distance) as follows:

$$P(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| \geq \varepsilon) \rightarrow 0 \text{ as } n \rightarrow \infty$$

- In which this probability is an upper limit for every function  $f$  in  $\mathcal{F}$

## Bringing Consistency to the ERM Principle

- Then we can use the inequality:

$$P(|R(f_n) - R_{\text{emp}}(f_n)| \geq \varepsilon) \leq P(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| \geq \varepsilon)$$

- **To show that the Law of Large Numbers is valid for a class of functions  $\mathcal{F}$ , i.e., for a subspace  $\mathcal{F}$  with a restricted set of functions**
  - Thus, the ERM Principle is consistent with respect to subspace  $\mathcal{F}$
  - For that, consider the distance between the Expected Risk of any classifier in  $\mathcal{F}$  versus the best in  $\mathcal{F}$

$$|R(f_n) - R(f_{\mathcal{F}})|$$

# Bringing Consistency to the ERM Principle

- It is obvious that:

$$|R(f_n) - R(f_F)| \geq 0$$

- Because the Expected Risk of the best classifier in subspace F is smaller or equal to any other classifier in F, thus we can write:

$$|R(f_n) - R(f_F)| = R(f_n) - R(f_F)$$

- From that, Vapnik added terms related to the Empirical Risks as follows:

$$R(f_n) - R(f_F) =$$

$$R(f_n) - R_{emp}(f_n) + R_{emp}(f_n) - R_{emp}(f_F) + R_{emp}(f_F) - R(f_F)$$

## Bringing Consistency to the ERM Principle

- Thus, Vapnik conclude that:

$$R(f_n) - R(f_F) =$$

$$R(f_n) - R_{emp}(f_n) + R_{emp}(f_n) - R_{emp}(f_F) + R_{emp}(f_F) - R(f_F)$$

- has an upper limit as follows:

$$R(f_n) - R_{emp}(f_n) + R_{emp}(f_n) - R_{emp}(f_F)$$

$$+ R_{emp}(f_F) - R(f_F) \leq \boxed{R(f_n) - R_{emp}(f_n) + R_{emp}(f_F) - R(f_F)}$$

## Bringing Consistency to the ERM Principle

- Thus, Vapnik conclude that:

$$R(f_n) - R(f_F) =$$

$$R(f_n) - R_{emp}(f_n) + R_{emp}(f_n) - R_{emp}(f_F) + R_{emp}(f_F) - R(f_F)$$

- has an upper limit as follows:

$$R(f_n) - R_{emp}(f_n) + R_{emp}(f_n) - R_{emp}(f_F)$$

$$+ R_{emp}(f_F) - R(f_F) \leq R(f_n) \xrightarrow{R_{emp}(f_n) + R_{emp}(f_F) - R(f_F)}$$

- Because, by definition, the Empirical Risk of the best classifier in  $F$  is greater than (or equal to itself or another equivalent) any other in  $F$ , as  $n$  increases, i.e.:

$$R_{emp}(f_n) - R_{emp}(f_F) \leq 0$$

By definition  $R_{emp}(f_n) \leq R_{emp}(f_F)$  because  $f_n$  was obtained (optimized) based on the training set

# Bringing Consistency to the ERM Principle

- Thus, Vapnik conjectured:

$$R(f_n) - R_{emp}(f_n) \leq R(f_F) - R_{emp}(f_F)$$

To see, remember what happens if space  $F$  contains the memory-based classifier

- has an upper bound:

$$R(f_n) - R_{emp}(f_n) + R_{emp}(f_n) - R_{emp}(f_F) + R_{emp}(f_F) - R(f_F) \leq R(f_n) \rightarrow R_{emp}(f_n) + R_{emp}(f_F) - R(f_F)$$

- Because, by definition, the Empirical Risk of the best classifier in  $F$  is greater than (or equal to itself or another equivalent) any other in  $F$ , as  $n$  increases, i.e.:

$$R_{emp}(f_n) - R_{emp}(f_F) \leq 0$$

By definition  $R_{emp}(f_n) \leq R_{emp}(f_F)$  because  $f_n$  was obtained (optimized) based on the training set

## Bringing Consistency to the ERM Principle

- Vapnik still worked on the inequality:

$$R(f_n) - R_{emp}(f_n) + R_{emp}(f_n) - R_{emp}(f_F) \\ + R_{emp}(f_F) - R(f_F) \leq \boxed{R(f_n) - R_{emp}(f_n)} + \boxed{R_{emp}(f_F) - R(f_F)}$$

- Observe its right side contains the distances between the Empirical and Expected Risks for  $f_n$  and for  $f_F$
- **Then Vapnik considered the worst function  $f_n$  in  $F$** , i.e., the one with maximum distance between the Empirical and the Expected Risks, then he obtained another bound for the previous inequality:

$$R(f_n) - R_{emp}(f_n) + R_{emp}(f_F) - R(f_F) \leq 2 \sup_{f \in F} |R(f) - R_{emp}(f)|$$

# Bringing Consistency to the ERM Principle

- Vapnik still worked on the inequality:

$$R(f_n)$$

+

**Observe term:**

$$2 \sup |R(f) - R_{\text{emp}}(f)|$$

$$R(f_F)$$

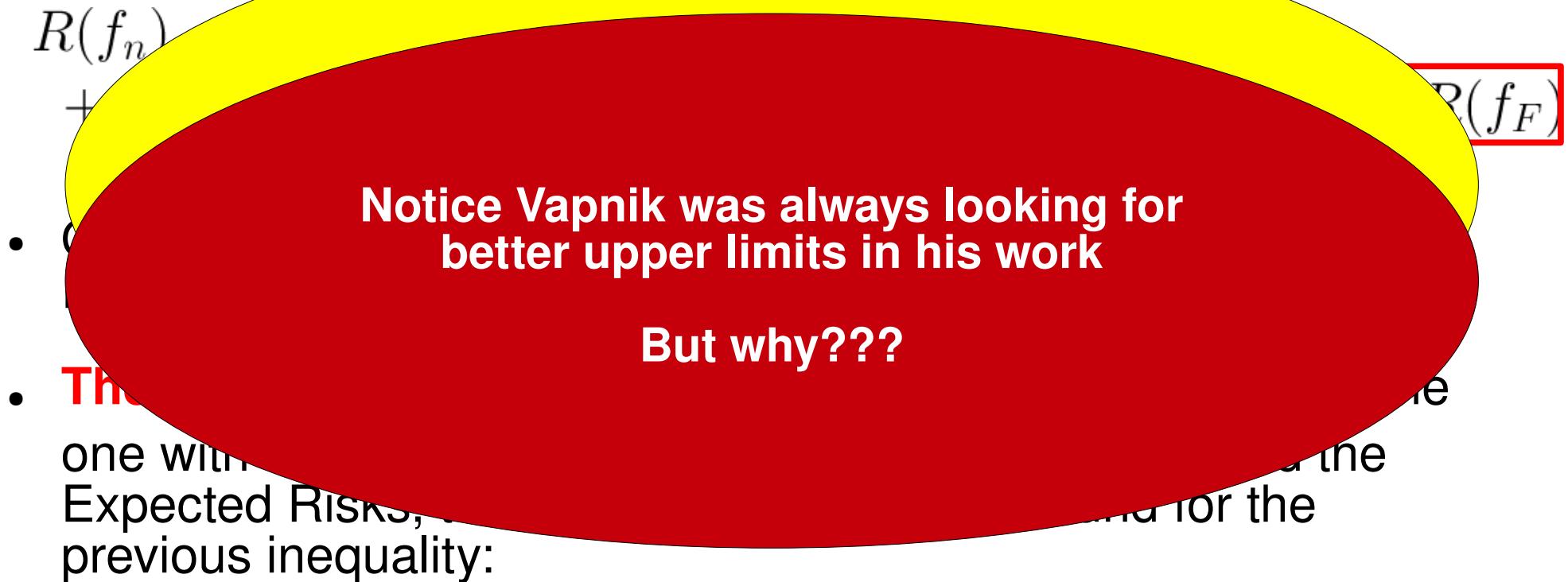
- Observe that  $R(f_F)$  emerges as an upper limit to the worst function in subspace  $F$

- Then Vapnik observed that if  $f_n$  is the function in  $F$  which achieves the maximum distance between the Empirical and the Expected Risks, i.e., the one with maximum distance between the Empirical and the Expected Risks, then he obtained another bound for the previous inequality:

$$R(f_n) - R_{\text{emp}}(f_n) + R_{\text{emp}}(f_F) - R(f_F) \leq 2 \sup_{f \in F} |R(f) - R_{\text{emp}}(f)|$$

# Bringing Consistency to the ERM Principle

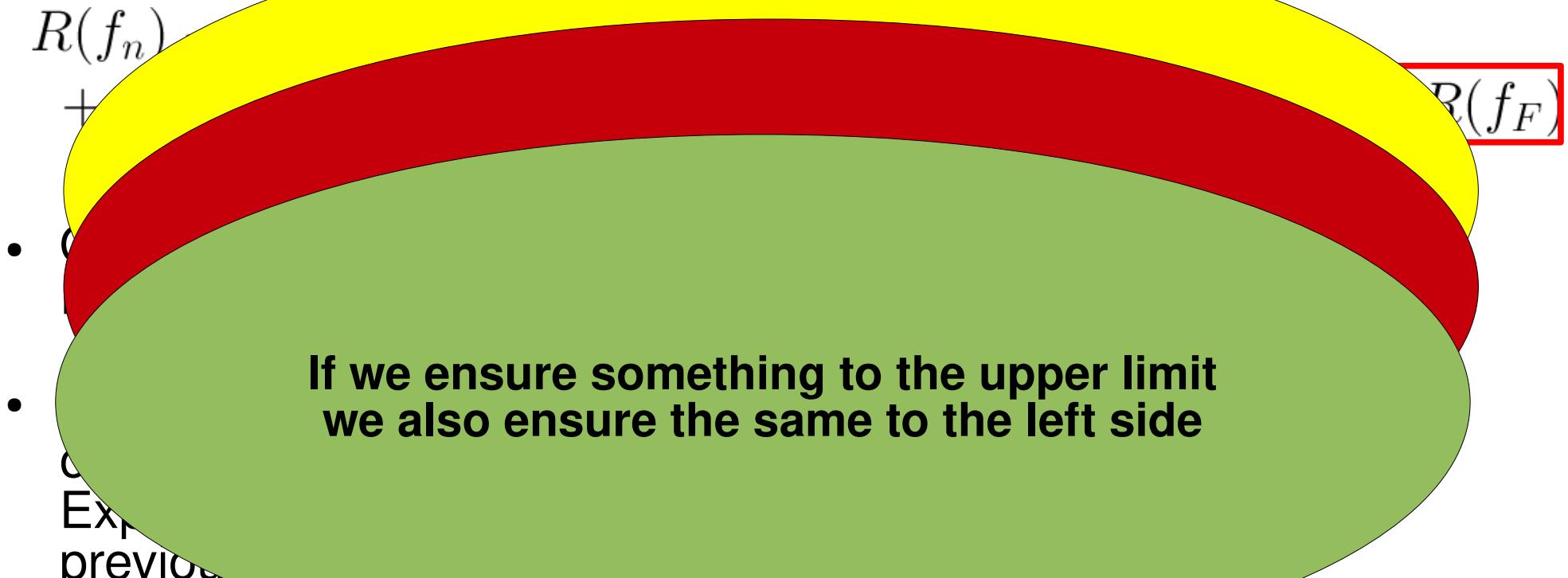
- Vapnik still worked on the inequality:



$$R(f_n) - R_{emp}(f_n) + R_{emp}(f_F) - R(f_F) \leq 2 \sup_{f \in F} |R(f) - R_{emp}(f)|$$

# Bringing Consistency to the ERM Principle

- Vapnik still worked on the inequality:



$$R(f_n) - R_{emp}(f_n) + R_{emp}(f_F) - R(f_F) \leq 2 \sup_{f \in F} |R(f) - R_{emp}(f)|$$

# Bringing Consistency to the ERM Principle

- Vapnik went back to the **Estimation error**:

$$|R(f_n) - R(f_{\mathcal{F}})|$$

- and used:

$$R(f_n) - R_{emp}(f_n) + R_{emp}(f_F) - R(f_F) \leq 2 \sup_{f \in F} |R(f) - R_{emp}(f)|$$

- To produce:

$$P(|R(f_n) - R(f_{\mathcal{F}})| \geq \varepsilon) \leq P(\sup_{f \in \mathcal{F}} |R(f) - R_{emp}(f)| \geq \varepsilon/2)$$

# Bringing Consistency to the ERM Principle

- Vapnik went back to the **Estimation error**:

$$|R(f_n) - R(f_{\mathcal{F}})|$$

- and used

$$R(f_n)$$

$$(f)|$$

**In this manner the right side term, which only considers the worst function in  $\mathcal{F}$  is enough to bound the distance between the worst and the best classifier in  $\mathcal{F}$**

$$\mathbb{E}$$

$$/2)$$

## Bringing Consistency to the ERM Principle

- In this manner, the supreme (i.e., the situation for the worst classifier) of differences between the Empirical and Expected Risks, for every function  $f$  in  $\mathcal{F}$ , converges to zero as the sample size tends to infinity:

$$P(|R(f_n) - R(f_{\mathcal{F}})| \geq \varepsilon) \leq P(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| \geq \varepsilon/2)$$

## Bringing Consistency to the ERM Principle

- In this manner, the supreme (i.e., the situation for the worst classifier) of differences between the Empirical and Expected Risks, for every function  $f$  in  $\mathcal{F}$ , converges to zero as the sample size tends to infinity:

$$P(|R(f_n) - R(f_{\mathcal{F}})| \geq \varepsilon) \leq P(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| \geq \varepsilon/2)$$

- This condition is enough to ensure consistency to the ERM Principle
- **Important Conclusions:**
  - We need some bias, i.e., we need to select some subspace  $\mathcal{F}$  to make this principle valid
  - This condition of uniform convergence depends on the definition of a subspace of functions
    - Without a bias there is no learning guarantee

# Bringing Consistency to the ERM Principle

- In this manner, the supreme (i.e., the situation for the worst classifier) of differences between the Empirical and Expected Risks, for every function  $f$  in  $\mathcal{F}$ , converges to zero as the sample size tends to infinity:

$$P(|R(f_n) - R(f_{\mathcal{F}})| \geq \varepsilon) \leq P(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| \geq \varepsilon/2)$$

- This condition is enough to guarantee consistency to the ERM Principle
- Important Question:**
  - We need to make sure that space  $\mathcal{F}$  to make sure that the empirical risk minimizer is in the subspace
  - This condition is not always satisfied, e.g.: the memory-based classifier
  - Without a bias term, we cannot guarantee

# Bringing Consistency to the ERM Principle

- In this manner, the supreme (i.e., the situation for the worst classifier) of differences between the Empirical and Expected Risks, for every function  $f$  in  $\mathcal{F}$ , converges to zero as the sample size tends to infinity:

$$P(|R(f_n) - R(f_{\mathcal{F}})| \geq \varepsilon) \leq P(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| \geq \varepsilon/2)$$

- This condition is enough to guarantee consistency to the ERM Principle
  - Important Condition:**
    - We need to make sure that  $\mathcal{F}$  is a closed subspace of the function space.
    - This condition is crucial for the definition of the supremum risk.
    - Without a closed subspace, the supremum risk may not be well-defined.
- Observe  $|R(f) - R_{\text{emp}}(f)|$  will never converge!

# Bringing Consistency to the ERM Principle

- **Important Conclusions:**

- The greater subspace  $F$  is, greater the supreme of differences is:

$$\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)|$$

- Consequently, the greater  $F$  is, the more difficult is to satisfy the Law of Large Numbers
  - Putting in simple words:
    - As subspace  $F$  increases in size, it is more difficult to ensure consistency for the ERM Principle
    - Smaller subspaces are easier to provide consistency to the ERM Principle

# Bringing Consistency to the ERM Principle

- **Important Conclusions:**

- The greater subspace  $F$  is, greater the supreme of differences is:

$$\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)|$$

- Consequently, the greater  $F$  is, the more difficult is to satisfy the Law of Large Numbers
  - Putting in simple words:

- As subspace  $F$  increases in size, it is more difficult to ensure consistency for the ERM Principle
- Smaller subspaces are easier to provide consistency to the ERM Principle

- **What is the relevance of providing consistency to the ERM Principle?**

- **If consistent, the Empirical Risk tends to the Expected Risk, i.e., we have a good estimator for the Expected Risk and therefore we can find the best classifier in the algorithm bias (or subspace  $F$ )!**

# Bringing Consistency to the ERM Principle

- **Important Conclusions:**
  - The greater subspace  $F$  is, greater the supreme of differences is:
  - Consequently, the Empirical Risk is a good estimator for the Expected Risk by the Law of Large Numbers.
  - Consistency of the Empirical Risk implies consistency of the classifier.
- **What is the ERM Principle?**
  - **If consistent, the Empirical Risk tends to the Expected Risk, i.e., we have a good estimator for the Expected Risk and therefore we can find the best classifier in the algorithm bias (or subspace  $F$ )!**

# Bringing Consistency to the ERM Principle

- **Important Conclusions:**

- The greater subspace  $F$  is, greater the supreme of differences is:  
$$\sup_{f \in F} |f(x) - f(x')| \leq \epsilon$$
- Consequently, the generalization error is bounded by the Law of Large Numbers:  
$$E_R(\hat{f}) \leq E_R(f^*) + \epsilon$$

**It does not mean there is no learning when this Principle is inconsistent!**

**However there is no guarantee according to it!**

-  
Prin-

- If consistency is guaranteed, we have a good estimator for the Expected Risk, i.e., we have a good estimator for the Expected Risk and therefore we can find the best classifier in the algorithm bias (or subspace  $F$ )!

## Properties of subspace F to ensure uniform convergence

- Now we know under which conditions the ERM Principle is consistent:
  - Besides all the theoretical stuff we have seen already:
    - In practice, they do not help that much!

## Properties of subspace F to ensure uniform convergence

- Now we know under which conditions the ERM Principle is consistent:
  - Besides all the theoretical stuff we have seen already:
    - In practice, they do not help that much!
  - **In order to get practical results, we need to define the properties of subspace F to ensure uniform convergence to the ERM Principle**
    - i.e., what properties subspace F must hold so the Empirical Risk is a good estimator to the Expected Risk as the sample size tends to infinity?

## Properties of subspace F to ensure uniform convergence

- However, let us get back to the Law of Large Numbers whose upper bound is given by Chernoff:

$$P\left(\left| \frac{1}{n} \sum_{i=1}^n \xi_i - E(\xi) \right| \geq \epsilon\right) \leq 2 \exp(-2n\epsilon^2)$$

Sendo  $\xi_i$  valores aleatórios no intervalo  $[0, 1]$

## Properties of subspace F to ensure uniform convergence

- However, let us get back to the Law of Large Numbers whose upper bound is given by Chernoff:

$$P\left(\left|\frac{1}{n} \sum_{i=1}^n \xi_i - E(\xi)\right| \geq \epsilon\right) \leq 2 \exp(-2n\epsilon^2)$$

Sendo  $\xi_i$  valores aleatórios no intervalo  $[0, 1]$

- Consider subspace  $F$  is finite and contains the following functions  $F = \{f_1, \dots, f_m\}$ . Each one of those functions respect the Law of Large Numbers having the Chernoff's bound:

$$P(|R(f_i) - R_{\text{emp}}(f_i)| \geq \varepsilon) \leq 2 \exp(-2n\varepsilon^2)$$

- This is true for every individual function in  $F$

## Properties of subspace F to ensure uniform convergence

- However, let us get back to the Law of Large Numbers whose upper bound is given by Chernoff:

$$P\left(\left|\frac{1}{n} \sum_{i=1}^n f(x_i) - \mathbb{E}[f]\right| > 2n\epsilon^2\right)$$

- Consider the function space  $F$  that respects the Law of Large Numbers, considering none of them was obtained using the training set

**Functions in this subspace respect the Law of Large Numbers, considering none of them was obtained using the training set**

- This is true for every individual function in  $F$

## Properties of subspace F to ensure uniform convergence

- However, we need something that is valid for all functions in F and not only for each individual function
- That is why Vapnik rewrote:

$$P(|R(f_i) - R_{\text{emp}}(f_i)| \geq \varepsilon) \leq 2 \exp(-2n\varepsilon^2)$$

- as follows:

$$\begin{aligned} P(\sup_{f \in F} |R(f) - R_{\text{emp}}(f)| \geq \epsilon) &= \\ P(|R(f_1) - R_{\text{emp}}(f_1)| \geq \epsilon \text{ or } \dots \text{ or } |R(f_m) - R_{\text{emp}}(f_m)| \geq \epsilon) \end{aligned}$$

## Properties of subspace F to ensure uniform convergence

- However, we need something that is valid for all functions in F and not only for each individual function
- That is why Vapnik rewrote:

$$P(|R(f_i) - R_{\text{emp}}(f_i)| \geq \varepsilon) \leq 2 \exp(-2n\varepsilon^2)$$

- as follows:

$$\begin{aligned} P(\sup_{f \in F} |R(f) - R_{\text{emp}}(f)| \geq \epsilon) &= \\ P(|R(f_1) - R_{\text{emp}}(f_1)| \geq \epsilon \text{ or } \dots \text{ or } |R(f_m) - R_{\text{emp}}(f_m)| \geq \epsilon) \end{aligned}$$

- and afterwards he found an upper bound in terms of probabilities:

$$\begin{aligned} P(|R(f_1) - R_{\text{emp}}(f_1)| \geq \epsilon \text{ or } \dots \text{ or } |R(f_m) - R_{\text{emp}}(f_m)| \geq \epsilon) \\ \leq \sum_{i=1}^m P(|R(f_i) - R_{\text{emp}}(f_i)| \geq \epsilon) \end{aligned}$$

## Properties of subspace F to ensure uniform convergence

- Finally we obtain another bound using Chernoff's for the whole set of functions in subspace F as follows:

$$\sum_{i=1}^m P(|R(f_i) - R_{emp}(f_i)| \geq \epsilon) \leq 2m \exp(-2n\epsilon^2)$$

- Then we found a bound for the uniform convergence considering all **m** functions in the **finite** subspace F as follows:

$$P(\sup_{f \in F} |R(f) - R_{emp}(f)| \geq \epsilon) \leq 2m \exp(-2n\epsilon^2)$$

## Properties of subspace F to ensure uniform convergence

- Finally we obtain another bound using Chernoff's for the whole set of functions in subspace F as follows:

$$\sum_{i=1}^m P(|R(f_i) - R_{emp}(f_i)| \geq \epsilon) \leq 2m \exp(-2n\epsilon^2)$$

- Then we found a bound for the uniform convergence considering all **m** functions in the **finite** subspace F as follows:

$$P(\sup_{f \in F} |R(f) - R_{emp}(f)| \geq \epsilon) \leq 2m \exp(-2n\epsilon^2)$$

- Observe if subspace F is finite, **m** becomes a constant and the uniform convergence still occurs as the sample size tends to infinity
- In this manner we prove the ERM Principle is consistent over a finite subspace of functions
  - But what happens when subspace F is infinite?**

## Properties of subspace F to ensure uniform convergence

- Finally we obtain another bound using Chernoff's for the whole set of functions in subspace F as follows:

$$\sum_{i=1}^m P(|R(f_i) - R_{emp}(f_i)| \geq \epsilon) \leq 2m \exp(-2n\epsilon^2)$$

- Then we found a bound for the whole set of functions in F by considering all m functions.

**Vapnik had another challenge!  
How to work on infinite spaces?**

- infinite dimensional function space
  - In this note we considered a finite subspace
- **But what happens when subspace F is infinite?**

## Properties of subspace F to ensure uniform convergence

- However, before that, Vapnik solved the problem of computing the Expected Risk using a **ghost sample**
  - In that situation he used the **Symmetrization lemma**
    - The main objective was to transform the term below to only use samples, because we cannot compute the Expected Risk!

$$\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)|$$

## Properties of subspace F to ensure uniform convergence

- However, before that, Vapnik solved the problem of computing the Expected Risk using a **ghost sample**
  - In that situation he used the **Symmetrization lemma**
    - The main objective was to transform the term below to only use samples, because we cannot compute the Expected Risk!

$$\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)|$$

- Then Vapnik considered a sample A and another A' (ghost) of labeled instances.
  - He assumed samples were independently drawn
  - Observe sample A' is not really used, it is a hypothetical sample (if we obtain one more sample), i.e., it is the one called **ghost sample**

## Properties of subspace F to ensure uniform convergence

- Having the **ghost sample**  $A'$ , Vapnik defined the following bound:

$$P\left(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| > \epsilon\right) \leq 2P\left(\sup_{f \in \mathcal{F}} |R_{\text{emp}}(f) - R'_{\text{emp}}(f)| > \epsilon/2\right)$$

- Each sample on the right side has size  $n$ , that is why he came up with constant 2 in this formulation

## Properties of subspace F to ensure uniform convergence

- Having the **ghost sample** A', Vapnik defined the following bound:

$$P\left(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| > \epsilon\right) \leq 2P\left(\sup_{f \in \mathcal{F}} |R_{\text{emp}}(f) - R'_{\text{emp}}(f)| > \epsilon/2\right)$$

- Each sample on the right side has size n, that is why he came up with constant 2 in this formulation
- This lemma is also referred to as the **Symmetrization lemma**
  - Besides we do not show all proofs in here, it is reasonable that Empirical Risks for two different and independent samples are close as n tends to infinity
    - Thus those Empirical Risk should also be close to the Expected Risk
  - In this manner, Vapnik worked out to remove term R(f) which could not be computed, as we only have samples available

## Properties of subspace $F$ to ensure uniform convergence

- Now we see how the **Symmetrization lemma** is useful:
  - Remember Vapnik proved the convergence for the ERM Principle considering a **finite subspace**  $F$
  - He still had to prove the same for **infinite subspaces**
    - For that, he needed the **Symmetrization lemma**

## Properties of subspace F to ensure uniform convergence

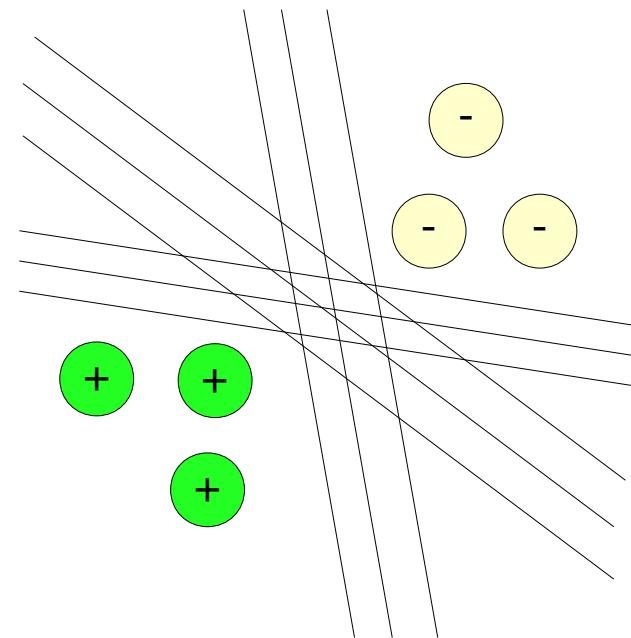
- Now we see how the **Symmetrization lemma** is useful:
  - Remember Vapnik proved the convergence for the ERM Principle considering a **finite subspace** F
  - He still had to prove the same for **infinite subspaces**
    - For that, he needed the **Symmetrization lemma**
- It is important to observe that besides subspace F contains infinite functions, the way they classify a training set with n instances is finite, for example:
  - Suppose n input instances  $X_1, \dots, X_n$
  - Let only two available class labels:  $\{-1, +1\}$
  - In this scenario, every function f in F can provide at most  $2^n$  different results

## Properties of subspace F to ensure uniform convergence

- Even though subspace F is infinite, each function  $f$  in F can only provide at most  $2^n$  different results over a training set with  $n$  instances
  - This means there are infinite functions which provide the same classification result, for example:

For example, consider subspace F is composed of all linear functions

There are infinite linear functions that provide the same classification, therefore, they are taken as similar



## Properties of subspace $F$ to ensure uniform convergence

- Thus, supreme performs only over a **finite set of functions in  $F$** :

$$\sup_{f \in \mathcal{F}} |R_{\text{emp}}(f) - R'_{\text{emp}}(f)|$$

- **In this situation, two functions  $f$  and  $g$ , both in  $F$ , produce the same Empirical Risk  $R_{\text{emp}}(f) = R_{\text{emp}}(g)$ , when they produce similar classifications**

## Properties of subspace F to ensure uniform convergence

- Thus, supreme performs only over a **finite set of functions in F**:

$$\sup_{f \in \mathcal{F}} |R_{\text{emp}}(f) - R'_{\text{emp}}(f)|$$

- In this situation, two functions  $f$  and  $g$ , both in  $F$ , produce the same Empirical Risk  $R_{\text{emp}}(f) = R_{\text{emp}}(g)$ , when they produce similar classifications
- Then, as we have two samples we will have  $2^n$  different ways of classifying each one, and  $2^{2n}$  different ways of classifying both together, since both have  $n$  instances when combined:
  - i.e., there will be at most  $2^{2n}$  different functions in  $F$  considering both samples

## Properties of subspace F to ensure uniform convergence

- Then Vapnik used the previous results to derive the first **capacity measure** for a subspace F (or class of functions)
  - For that, let  $Z_n = \{(X_1, Y_1), \dots, (X_n, Y_n)\}$  be a set of n instances
  - Let  $|F_{Z_n}|$  be the cardinality of subspace F for set  $Z_n$ , i.e., the number of functions that produce different classification results for  $Z_n$
  - Let us define the maximal number of functions that can be distinguished, i.e., produce different classification results, using the following:

$$\mathcal{N}(\mathcal{F}, n) = \max\{|\mathcal{F}_{Z_n}| \mid X_1, \dots, X_n \in \mathcal{X}\}$$

## Properties of subspace F to ensure uniform convergence

- Term  $\mathcal{N}(\mathcal{F}, n)$  is referred to as **shattering coefficient** for subspace  $\mathcal{F}$  with respect to a set with  $n$  instances
- If  $\mathcal{N}(\mathcal{F}, n) = 2^n$ , this means that exists **at least** one sample with  $n$  examples which can be classified in all possible manners, given two class labels  $\{-1, +1\}$

## Properties of subspace F to ensure uniform convergence

- Term  $\mathcal{N}(\mathcal{F}, n)$  is referred to as **shattering coefficient** for subspace  $\mathcal{F}$  with respect to a set with  $n$  instances
- If  $\mathcal{N}(\mathcal{F}, n) = 2^n$ , this means that exists **at least** one sample with  $n$  examples which can be classified in all possible manners, given two class labels  $\{-1, +1\}$ 
  - In this situation we say that subspace  $\mathcal{F}$  is capable of **shattering**  $n$  points in all possible ways
  - Observe this concept considers the situation in which there is **at least one sample** with  $n$  elements that can be classified in all possible ways (because there is a **max** in the formulation of the previous slide)
    - This does not imply that every sample will be shattered (classified) in all possible ways!

## Properties of subspace $F$ to ensure uniform convergence

- The **shattering coefficient** is a **capacity measure** for the class of functions  $F$ 
  - By capacity we mean it allows to measure the size which refers to the complexity of the subspace
  - Observe a greater subspace  $F$  tends to have a greater shattering coefficient

## Properties of subspace $\mathcal{F}$ to ensure uniform convergence

- The **shattering coefficient** is a **capacity measure** for the class of functions  $\mathcal{F}$ 
  - By capacity we mean it allows to measure the size which refers to the complexity of the subspace
  - Observe a greater subspace  $\mathcal{F}$  tends to have a greater shattering coefficient
- But how Vapnik used the shattering coefficient to produce an upper bound for the **Symmetrization lemma**?

$$P\left(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| > \epsilon\right) \leq 2P\left(\sup_{f \in \mathcal{F}} |R_{\text{emp}}(f) - R'_{\text{emp}}(f)| > \epsilon/2\right)$$

## Properties of subspace $F$ to ensure uniform convergence

- But how Vapnik used the shattering coefficient to produce an upper bound for the **Symmetrization lemma**?
  - He considered that there are  $2n$  instances, therefore there is a set  $Z_{2n}$ , in which the first  $n$  examples are from the first sample and the following  $n$  are from the **ghost sample**
  - As first step, Vapnik substituted the supreme over  $F$  by the supreme over  $F_{z_{2n}}$ 
    - i.e., the maximal number of functions  $\mathcal{N}(\mathcal{F}, n) \leq 2^{2n}$  that produce different classifications for the two samples together is given by  $2^{2n}$

## Properties of subspace F to ensure uniform convergence

- From the following:

$$P\left(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| > \epsilon\right) \leq 2P\left(\sup_{f \in \mathcal{F}} |R_{\text{emp}}(f) - R'_{\text{emp}}(f)| > \epsilon/2\right)$$

- Vapnik obtained:

$$\begin{aligned} 2P\left(\sup_{f \in F} |R_{\text{emp}}(f) - R'_{\text{emp}}(f)| \geq \epsilon/2\right) = \\ 2P\left(\sup_{f \in F_{Z_{2n}}} |R_{\text{emp}}(f) - R'_{\text{emp}}(f)| \geq \epsilon/2\right) \end{aligned}$$

## Properties of subspace F to ensure uniform convergence

- From the following:

$$P\left(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| > \epsilon\right) \leq 2P\left(\sup_{f \in \mathcal{F}} |R_{\text{emp}}(f) - R'_{\text{emp}}(f)| > \epsilon/2\right)$$

- Vapnik obtained:

$$2P\left(\sup_{f \in F} |R_{\text{emp}}(f) - R'_{\text{emp}}(f)| \geq \epsilon/2\right) =$$

$$2P\left(\sup_{f \in F_{Z_{2n}}} |R_{\text{emp}}(f) - R'_{\text{emp}}(f)| \geq \epsilon/2\right)$$

- From that, as  $F_{z_{2n}}$  contains at most  $\mathcal{N}(F, 2n)$  different functions, he could employ the Chernoff bound as follows:

$$2P\left(\sup_{f \in F_{Z_{2n}}} |R_{\text{emp}}(f) - R'_{\text{emp}}(f)| \geq \epsilon/2\right) \leq 2\mathcal{N}(F, 2n) \exp(-n\epsilon^2/4)$$

## Properties of subspace F to ensure uniform convergence

- From the following:

$$P\left(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| > \epsilon\right) \leq 2P\left(\sup_{f \in \mathcal{F}} |R_{\text{emp}}(f) - R'_{\text{emp}}(f)| > \epsilon/2\right)$$

- Vapnik o

**In place of m**

$$\geq \epsilon/2) =$$

$$2P\left(\sup_{f \in F_{Z_{2n}}} |R_{\text{emp}}(f) - R'_{\text{emp}}(f)| \geq \epsilon/2\right)$$

- From that, as  $F_{z_{2n}}$  contains at most  $\mathcal{N}(\mathcal{F}, 2n)$  different functions, he could employ the Chernoff bound as follows:

$$2P\left(\sup_{f \in F_{Z_{2n}}} |R_{\text{emp}}(f) - R'_{\text{emp}}(f)| \geq \epsilon/2\right) \leq 2\underline{\mathcal{N}(F, 2n)} \exp(-n\epsilon^2/4)$$

# Properties of subspace F to ensure uniform convergence

- **First conclusion:**

- Consider the shattering coefficient is significantly smaller than  $2^{2n}$ , i.e.,  $\mathcal{N}(\mathcal{F}, 2n) \leq (2n)^k$  for a constant k
  - This means the shattering coefficient grows in a polynomial way, then plugging this in the Chernoff bound, we have:

$$\begin{aligned} 2\mathcal{N}(\mathcal{F}, 2n) \exp(-n\varepsilon^2/4) &= 2 \cdot (2n)^k \cdot \exp(-n\varepsilon^2/4) = \\ &= 2 \exp(k \cdot \log(2n) - n\varepsilon^2/4) \end{aligned}$$

- Here we see that as n tends to infinity, the whole expression converges to zero
- **i.e., the ERM Principle is consistent with respect to F when the shattering coefficient grows polynomially**

## Properties of subspace $\mathcal{F}$ to ensure uniform convergence

- **Second conclusion:**

- Consider the situation we have  $\mathcal{F}_{\text{all}}$ , i.e., taking into account the full space (**no bias**) which contains every possible function
  - Of course in this situation we can classify the set of instances in all possible ways, i.e.:

$$\mathcal{N}(\mathcal{F}, 2n) = 2^{2n}$$

- Thus, by substituting in the Chernoff bound we have:

$$2\mathcal{N}(\mathcal{F}, 2n) \exp(-n\varepsilon^2/4) = 2 \cdot 2^{2n} \exp(-n\varepsilon^2/4) = 2 \exp(n(2 \log(2) - \varepsilon^2/4))$$

- As epsilon assumes a very small value, soon we see that as  $n$  increases, this expression does not goes to zero

# Properties of subspace F to ensure uniform convergence

- **Second conclusion:**

- **Therefore we cannot conclude the consistency is valid for the ERM Principle when the subspace is  $F_{\text{all}}$**

- On another hand, **we cannot either conclude** the ERM Principle is inconsistent for  $F_{\text{all}}$
  - This happens due to the right-side term of the inequality below provides a sufficient condition for consistency but not a necessary condition

$$P\left(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| > \epsilon\right) \leq 2\mathcal{N}(\mathcal{F}, 2n) \exp(-n\epsilon^2/4)$$

- **However, later on Vapnik and Chervonenkis proved the following condition is necessary to ensure the consistency of the ERM Principle:**

$$\frac{\log \mathcal{N}(F, n)}{n} \rightarrow 0$$

- Finally:
  - If  $N(F, n)$  is a polynomial, then the condition goes to zero
  - If we consider a space without bias, i.e.,  $F_{\text{all}}$ , we will have  $N(F, n) = 2^n$ , thus:

$$\frac{\log N(F, n)}{n} = \frac{\log 2^n}{n} = \frac{n}{n} = 1$$

- **In this situation, the ERM Principle is not consistent for  $F_{\text{all}}$**

# Generalization Bounds

- We can rewrite the inequality below in another way:

$$P\left(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| > \epsilon\right) \leq 2\mathcal{N}(\mathcal{F}, 2n) \exp(-n\epsilon^2/4)$$

- Instead of fixing epsilon and computing the probability the Empirical Risk deviates from the Expected Risk, we can specify the probability delta for which the following bound is true:
  - For that we set the right-side term equal to delta, which is greater than zero, as follows:

$$P\left(\sup_{f \in F} |R(f) - R_{\text{emp}}(f)| > \epsilon\right) \leq \delta$$

- And then we solve for epsilon...

$$2\mathcal{N}(F, 2n) \exp(-n\epsilon^2/4) = \delta \quad \text{para } \delta > 0$$

- Solving for epsilon:

$$2\mathcal{N}(F, 2n) \exp(-n\epsilon^2/4) = \delta$$

$$\exp(-n\epsilon^2/4) = \frac{\delta}{2\mathcal{N}(F, 2n)}$$

$$\log(\exp(-n\epsilon^2/4)) = \log(\delta) - \log(2\mathcal{N}(F, 2n))$$

$$-n\epsilon^2/4 = \log(\delta) - \log(2\mathcal{N}(F, 2n))$$

$$\epsilon^2 = -\frac{4}{n} (\log(\delta) - \log(2\mathcal{N}(F, 2n)))$$

$$\epsilon = \sqrt{\frac{4}{n} (\log(2\mathcal{N}(F, 2n)) - \log(\delta))}$$

# Generalization Bounds

- As we substituted delta as presented next:

$$P\left(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| > \epsilon\right) \leq 2\mathcal{N}(\mathcal{F}, 2n) \exp(-n\epsilon^2/4)$$

$$P\left(\sup_{f \in F} |R(f) - R_{\text{emp}}(f)| > \epsilon\right) \leq \delta$$

- So, being upper limited by delta we have:

$$\sup_{f \in F} |R(f) - R_{\text{emp}}(f)| > \epsilon$$

$$\sup_{f \in F} |R(f) - R_{\text{emp}}(f)| > \sqrt{\frac{4}{n} (\log(2\mathcal{N}(F, 2n)) - \log(\delta))}$$

# Generalization Bounds

- So, being upper limited by delta we have:

$$\sup_{f \in F} |R(f) - R_{emp}(f)| > \epsilon$$

$$\sup_{f \in F} |R(f) - R_{emp}(f)| > \sqrt{\frac{4}{n} (\log(2\mathcal{N}(F, 2n)) - \log(\delta))}$$

- But delta means when the divergence between risks is above epsilon
  - What about when such divergence is below or equal?
    - This is the hit chance

# Generalization Bounds

- So, being upper limited by delta we have:

$$\sup_{f \in F} |R(f) - R_{emp}(f)| > \epsilon$$

$$\sup_{f \in F} |R(f) - R_{emp}(f)| > \sqrt{\frac{4}{n} (\log(2\mathcal{N}(F, 2n)) - \log(\delta))}$$

- The hit chance is represented by 1-delta, this is:

$$\sup_{f \in F} |R(f) - R_{emp}(f)| \leq \epsilon$$

$$\sup_{f \in F} |R(f) - R_{emp}(f)| \leq \sqrt{\frac{4}{n} (\log(2\mathcal{N}(F, 2n)) - \log(\delta))}$$

# Generalization Bounds

- So, being upper limited by delta we have:

$$\sup_{f \in F} |R(f) - R_{emp}(f)| > \epsilon$$

$$\sup_{f \in F} |R(f) - R_{emp}(f)| > \sqrt{\frac{4}{n}} (1-\delta)$$

- The hit chance is represented by

$$\sup_{f \in F} |R(f) - R_{emp}(f)| \leq \epsilon$$

$$\sup_{f \in F} |R(f) - R_{emp}(f)| \leq \sqrt{\frac{4}{n} (\log(2\mathcal{N}(F, 2n)) - \log(\delta))}$$

Observe the  
change in the  
Operator!

1-delta is the complement!

# Generalization Bounds

- Then we have:

$$\sup_{f \in F} |R(f) - R_{emp}(f)| \leq \epsilon$$

$$\sup_{f \in F} |R(f) - R_{emp}(f)| \leq \sqrt{\frac{4}{n} (\log(2\mathcal{N}(F, 2n)) - \log(\delta))}$$

- Assuming (remember the memory-based classifier) the empirical risk is less than the expected one we can rewrite as follows:

$$R(f) - R_{emp}(f) \leq \sqrt{\frac{4}{n} (\log(2\mathcal{N}(F, 2n)) - \log(\delta))}$$

for the worst classifier  $f \in F$

# Generalization Bounds

- As we substituted delta as presented next:

$$P\left(\sup_{f \in \mathcal{F}} |R(f) - R_{\text{emp}}(f)| > \epsilon\right) \leq 2\mathcal{N}(\mathcal{F}, 2n) \exp(-n\epsilon^2/4)$$
$$P\left(\sup_{f \in F} |R(f) - R_{\text{emp}}(f)| > \epsilon\right) \leq \delta$$

- Finally we obtain another upper limit for the hit chance:

$$R(f) \leq R_{\text{emp}}(f) + \sqrt{\frac{4}{n} (\log(2\mathcal{N}(F, 2n)) - \log(\delta))}$$

for the worst classifier  $f \in F$

# Generalization Bounds

- Now it is obvious from where Vapnik and Chervonenkis obtained:

$$\frac{\log \mathcal{N}(F, n)}{n} \rightarrow 0$$

- Because we found out:

$$R(f) \leq R_{\text{emp}}(f) + \sqrt{\frac{4}{n} (\log(2\mathcal{N}(\mathcal{F}, n)) - \log(\delta))}$$

# Generalization Bounds

- Let us intuitively analyze the inequality:

$$R(f) \leq R_{\text{emp}}(f) + \sqrt{\frac{4}{n} (\log(2\mathcal{N}(\mathcal{F}, n)) - \log(\delta))}$$

- If the Empirical Risk and the term inside the squared root are small, then there is a great probability of the Expected Risk to be small
  - If:
    - we use a small subspace  $\mathcal{F}$ , i.e., with few functions that produce different classifications
    - and this subspace is still capable of representing the training set
  - Then we can conclude that there is a great probability of learning a concept

# Generalization Bounds

- Let us intuitively analyze the inequality:

$$R(f) \leq R_{\text{emp}}(f) + \sqrt{\frac{4}{n} (\log(2\mathcal{N}(\mathcal{F}, n)) - \log(\delta))}$$

- i.e., there is a great probability of learning when:
  - There is bias, i.e., we define a subspace  $\mathcal{F}$
  - This subspace is capable of representing the training set (avoiding underfitting)

# Generalization Bounds

- On the other hand what happens if our problem is too complex?

$$R(f) \leq R_{\text{emp}}(f) + \sqrt{\frac{4}{n} (\log(2\mathcal{N}(\mathcal{F}, n)) - \log(\delta))}$$

- We need a larger subspace  $\mathcal{F}$  to model the training set
  - However the larger  $\mathcal{F}$  is we tend to the full space  $\mathcal{F}_{\text{all}}$  which is capable of representing any classification
  - However, this large space makes  $\mathcal{N}(\mathcal{F}, 2n) \rightarrow 2^{2n}$  what does not allow us to define the upper bound, therefore we cannot ensure the Empirical Risk is a good estimator for the Expected Risk

# Generalization Bounds

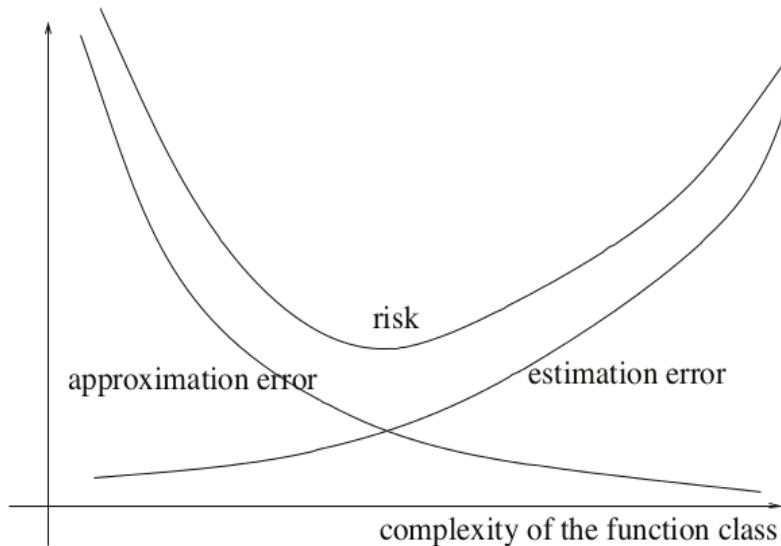
- On the other hand what happens if our problem is too complex?

$$R(f) \leq R_{\text{emp}}(f) + \sqrt{\frac{4}{n} (\log(2\mathcal{N}(\mathcal{F}, n)) - \log(\delta))}$$

- However, even for a very complex problem we could define some bias, i.e., a subspace  $\mathcal{F}$ 
  - Therefore the ERM Principle could still be consistent

# Generalization Bounds

- Summarizing:
  - The definition of some bias is essential to make the ERM Principle consistent
    - What leads us to ensure learning



- Observe the importance in assessing different techniques (**under different biases**) to approach the same problem:
  - Many researchers employ ensembles
- The formulation in the previous slide is very similar to the Tikonov regularization, which is commonly used in the Signal Processing domain

- The shattering coefficient motivated Vapnik and Chervonenkis to propose another capacity measure for set of functions (subspaces)

- Vapnik and Chervonenkis proposed a **capacity measure** to characterize the **growth of the shattering coefficient**:
  - VC (Vapnik-Chervonenkis) Dimension

- Vapnik and Chervonenkis proposed a **capacity measure** to characterize the **growth of the shattering coefficient**:
  - VC (Vapnik-Chervonenkis) Dimension
- Vapnik and Chervonenkis assumed that:
  - A sample with  $n$  instances  $Z_n$  is **shattered** by a class of functions  $F$  if such class can produce all possible classifications for this sample, i.e., the cardinality is  $|F_{Z_n}| = 2^n$
  - Then, the VC Dimension of  $F$  is defined as the greatest number  $n$  such that there exists one sample with  $n$  instances that can be **shattered** by  $F$

$$VC(F) = \max(n \in \mathbb{Z}^+ \mid |F_{Z_n}| = 2^n \text{ para algum } Z_n)$$

- If this maximum does not exist, the VC Dimension is defined as infinite

- In this way, once we know the **VC Dimension for the class of functions in  $F$  is finite**:
  - We know the shattering coefficient grows polynomially as the sample size goes to infinity
  - This implies consistency for the ERM Principle
    - What finally ensures learning! Great!

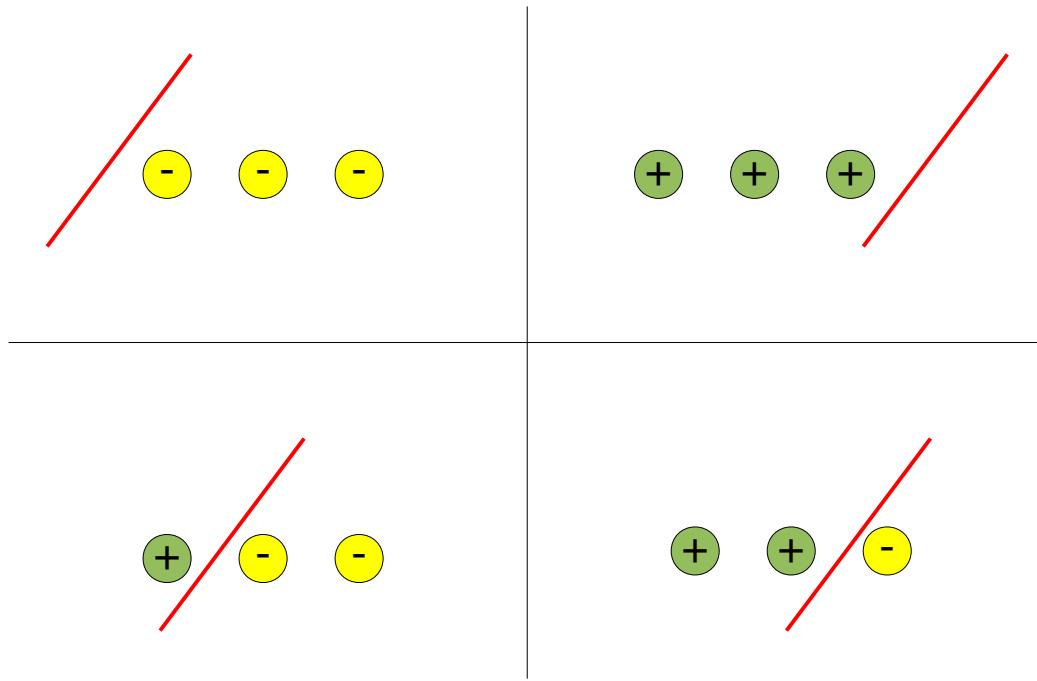
- In this way, once we know the **VC Dimension for the class of functions in F is finite**:
  - We know the shattering coefficient grows polynomially as the sample size goes to infinity
  - This implies consistency for the ERM Principle
    - What finally ensures learning! Great!
- If the **VC dimension is infinite** then there is at least one sample which can be shattered by functions in F in all possible ways, i.e.,  $2^n$  different classifications
  - In this form, as previously seen, the ERM Principle would not be consistent
    - Therefore there is no learning guarantee

# VC Dimension

- As an example, consider 3 points in a two-dimensional plane



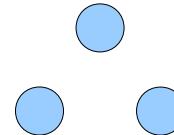
- Suppose  $F$  contains linear functions:



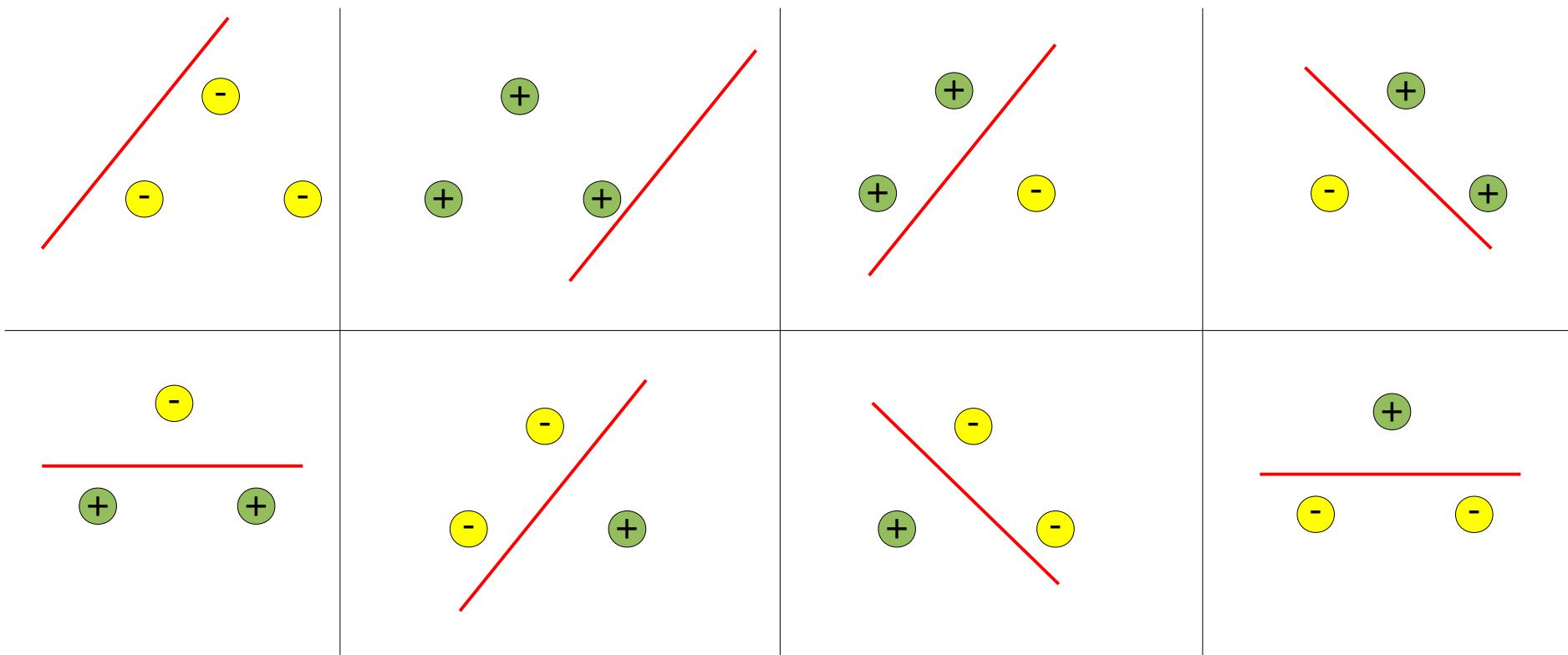
- We could shatter this sample in 4 different ways
  - But is there any other 3-point sample that we could shatter in more ways?

# VC Dimension

- Suppose we have the 3 points in different setting:



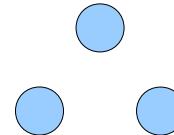
- Again consider  $F$  contains all linear functions:



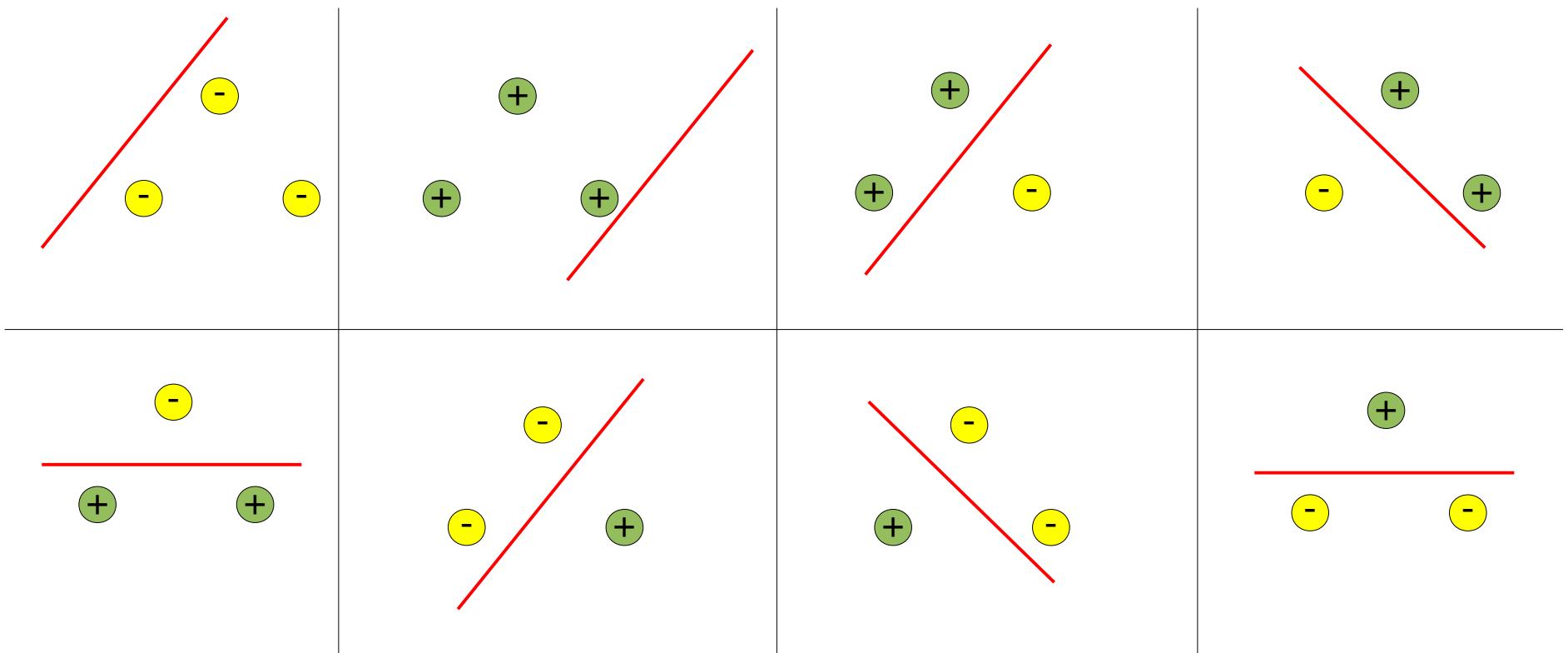
- Observe  $F$  was capable of shattering this sample in all  $2^n$  possible ways, what take us to the fact that  $F$  has a VC dimension at least equal to 3
  - Because there is at least one sample with 3 instances that can be shattered in all possible ways

# VC Dimension

- Suppose we have the 3 points in different setting:



- Again consider  $F$  contains all linear functions:



- Observe  $F$  was capable of shattering this sample in all  $2^n$  possible ways, what take us to the fact that  $F$  has a VC dimension at least equal to 3
  - Because there is at least one sample with 3 instances that can be shattered in all possible ways

# VC Dimension

- In this manner, for a classification problem where the input space is in  $R^2$  and  $F$  contains all linear functions,  $F$  is capable of shattering samples in at least:
  - $2^3 = 8$  ways, then we know the VC Dimension is equal or greater than 3, i.e.,  $VC \geq 3$ 
    - The question is: Is there any other sample with more points ( $n > 3$  with input space in  $R^2$ ) which can be shattered in  $2^n$  ways?
    - Consequently we still do not know the correct VC Dimension, thus we say its minimal value is 3

# VC Dimension

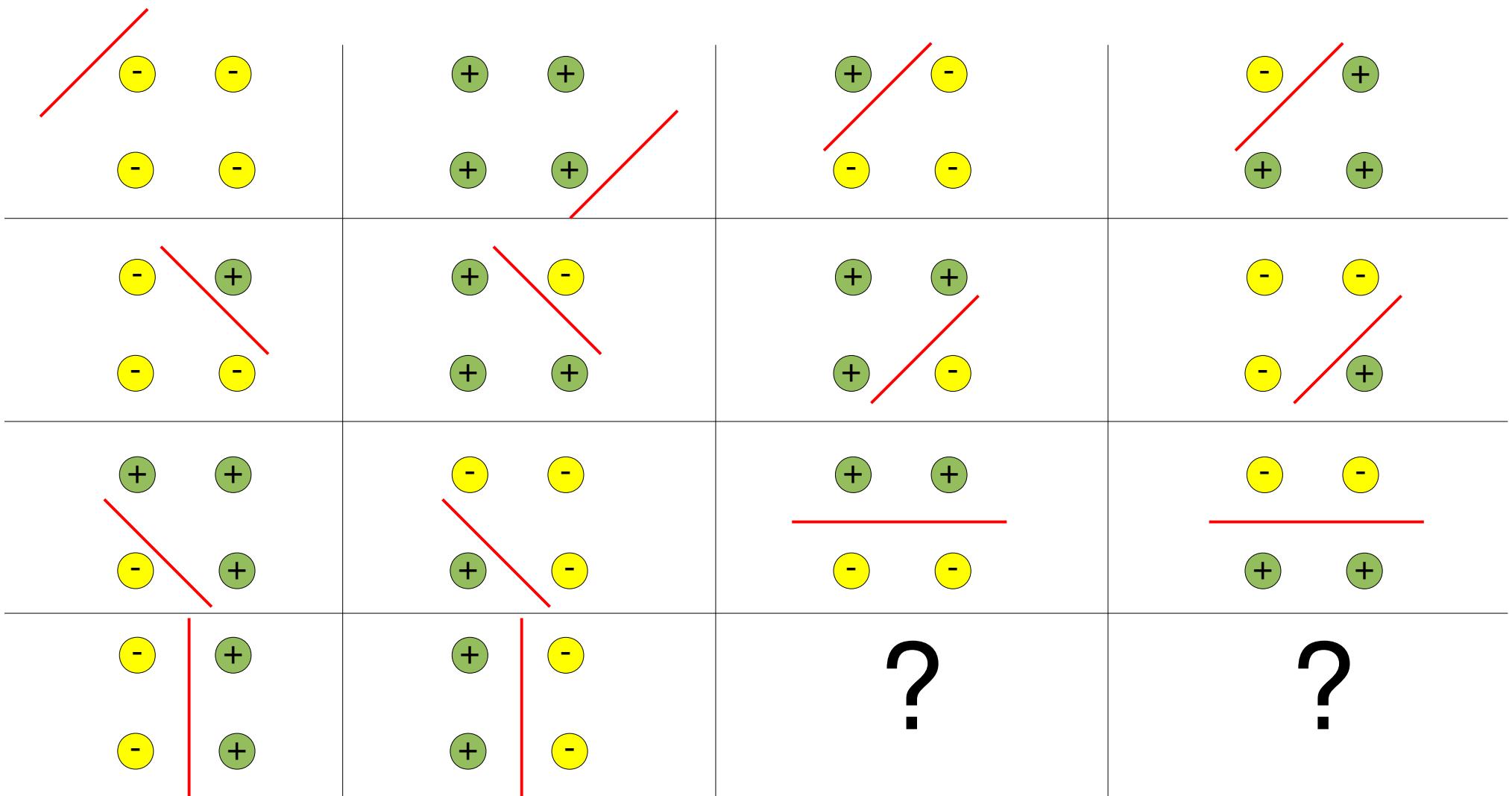
- In this manner, for a classification problem where the input space is in  $R^2$  and  $F$  contains all linear functions,  $F$  is capable of shattering samples in at least:
  - $2^3 = 8$  ways, then we know the VC Dimension is equal or greater than 3, i.e.,  $VC \geq 3$ 
    - The question is: Is there any other sample with more points ( $n > 3$  with input space in  $R^2$ ) which can be shattered in  $2^n$  ways?
    - Consequently we still do not know the correct VC Dimension, thus we say its minimal value is 3
- As the VC Dimension is given by the maximum value for the problem, if there is one sample with more points that can be shattered in all ways is enough to increase the value of the VC Dimension
  - But what about if we try to shatter more points in  $R^2$  using the same linear functions?
    - Maybe the VC Dimension is still greater for  $F$ !!!

# VC Dimension

- Let's consider:



- Let  $F$  contain all possible linear functions:



- There are ways to prove the VC Dimension without verifying all possible sample settings, but they are a bit complex!
- In order to make it we need to prove there is no 4-point sample in  $\mathbb{R}^2$  that can be shattered in all possible ways
  - **Not easy...**

- There are ways to prove the VC Dimension without verifying all possible sample settings, but they are a bit complex!
- In order to make it we need to prove there is no 4-point sample in  $\mathbb{R}^2$  that can be shattered in all possible ways
  - **Not easy...**
- **However, there is still an important proof about linear hyperplanes:**
  - If  $F$  contains all possible  $(n-1)$ -linear hyperplanes to classify points in  $\mathbb{R}^n$ , its VC Dimension is equal to  $n+1$

- There are ways to prove the VC Dimension without verifying all possible sample settings. Just consider the  $\text{VC Dim}(F) \leq \text{VC Dim}(\text{Hypothesis Space})$ ! It's not that complex!
- In order to prove the VC Dimension of a hypothesis space  $F$ , we can choose a sample in  $\mathbb{R}^2$  that ...

**Observe this conclusion:**

- From the bias of the classification algorithm, we can find the VC Dimension of subspace  $F$  and conclude about the consistency of the ERM Principle, and finally, conclude whether learning is guaranteed!

**Remember: The opposite is not guaranteed, i.e., it might occur learning even though the ERM Principle is inconsistent**

# Margin bounds

- Finally we will see one more **capacity measure** for subspace F
  - Consider the situation in which we use linear functions to separate points in input space  $\mathbb{R}^2$
  - Let a labeled sample that can be **perfectly shattered**
  - The **margin of a classifier**  $f$  is the shortest distance from any point to the line

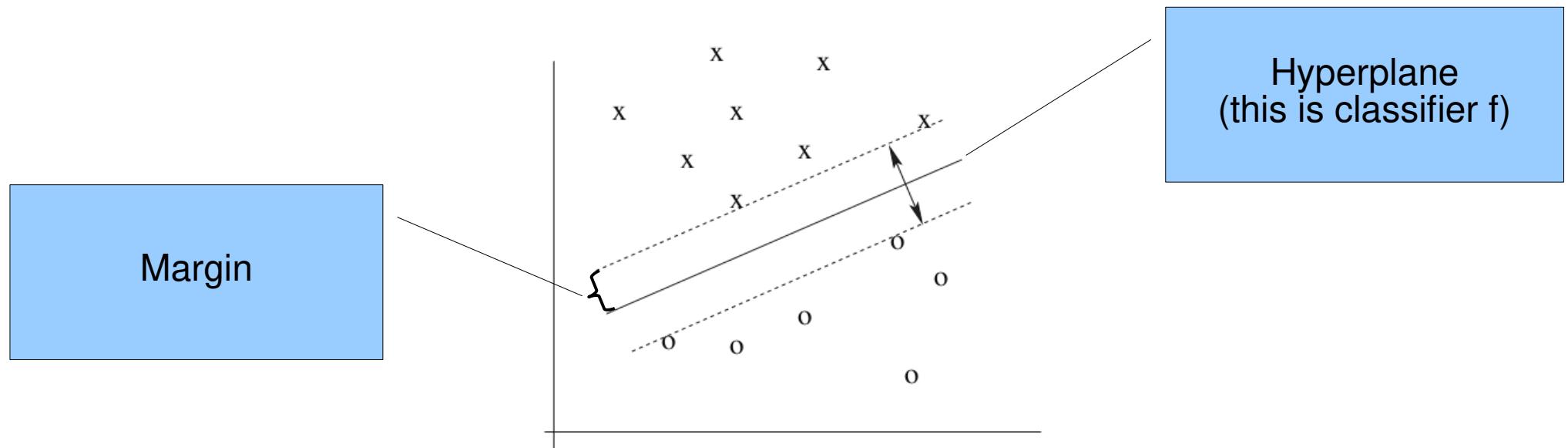


Figure obtained from Luxburg and Scholkopf, Statistical Learning Theory: Models, Concepts, and Results

# Margin bounds

- There is a proof the VC Dimension of a class of linear functions  $F_p$  that presents at least margin  $p$  is bounded by the ratio of radius  $R^p$  of the **smallest sphere enclosing all data points**:

$$VC(F_p) \leq \min \left( d, \frac{4R^2}{p^2} \right) + 1$$

- In which  $d$  is the input space dimension  $\mathbb{R}^d$

# Margin bounds

- There is a proof the VC Dimension of a class of linear functions  $F_p$  that presents at least margin  $p$  is bounded by the ratio of radius  $R^p$  of the **smallest sphere enclosing all data points**:

$$VC(F_p) \leq \min \left( d, \frac{4R^2}{p^2} \right) + 1$$

- In which  $d$  is the input space dimension  $\mathbb{R}^d$
- Consequently, as proven by Vapnik, the greater margin  $p$  is, smaller is the VC Dimension
  - Therefore the margin can be used as a capacity measure to a class of functions → **This is indeed the main motivation for the Support Vector Machines (SVMs)**

- Vapnik still proved the margin bound using the ERM Principle:

$$R(f) \leq v(f) + \sqrt{\frac{c}{n} \left( \frac{R^2}{\rho^2} \log(n)^2 + \log(1/\delta) \right)}$$

- in which  $v(f)$  is a fraction of training samples that present margin of at least equal to  $p$
- In this way the **margin maximization** provides consistency for the ERM Principle

# Conclusions

- SLT allows to verify under what conditions learning occurs
- Observe:
  - **Without bias, the ERM Principle is not consistent**
    - i.e., we need some bias to learn!

# Conclusions

- SLT allows to verify under what conditions learning occurs
- Observe:
  - **Without bias, the ERM Principle is not consistent**
    - i.e., we need some bias to learn!
  - Computing the VC Dimension for a set of functions  $F$ , we verify the consistency of the ERM Principle

# Conclusions

- SLT allows to verify under what conditions learning occurs
- Observe:
  - **Without bias, the ERM Principle is not consistent**
    - i.e., we need some bias to learn!
  - Computing the VC Dimension for a set of functions  $F$ , we verify the consistency of the ERM Principle
  - We can also investigate algorithms that adapt their biases
    - This would allow us to better explore space  $F_{\text{all}}$  in attempt to define a good subspace  $F$  for our problem → **avoiding over and underfitting**
      - Even better if it contains  $f_{\text{Bayes}}$

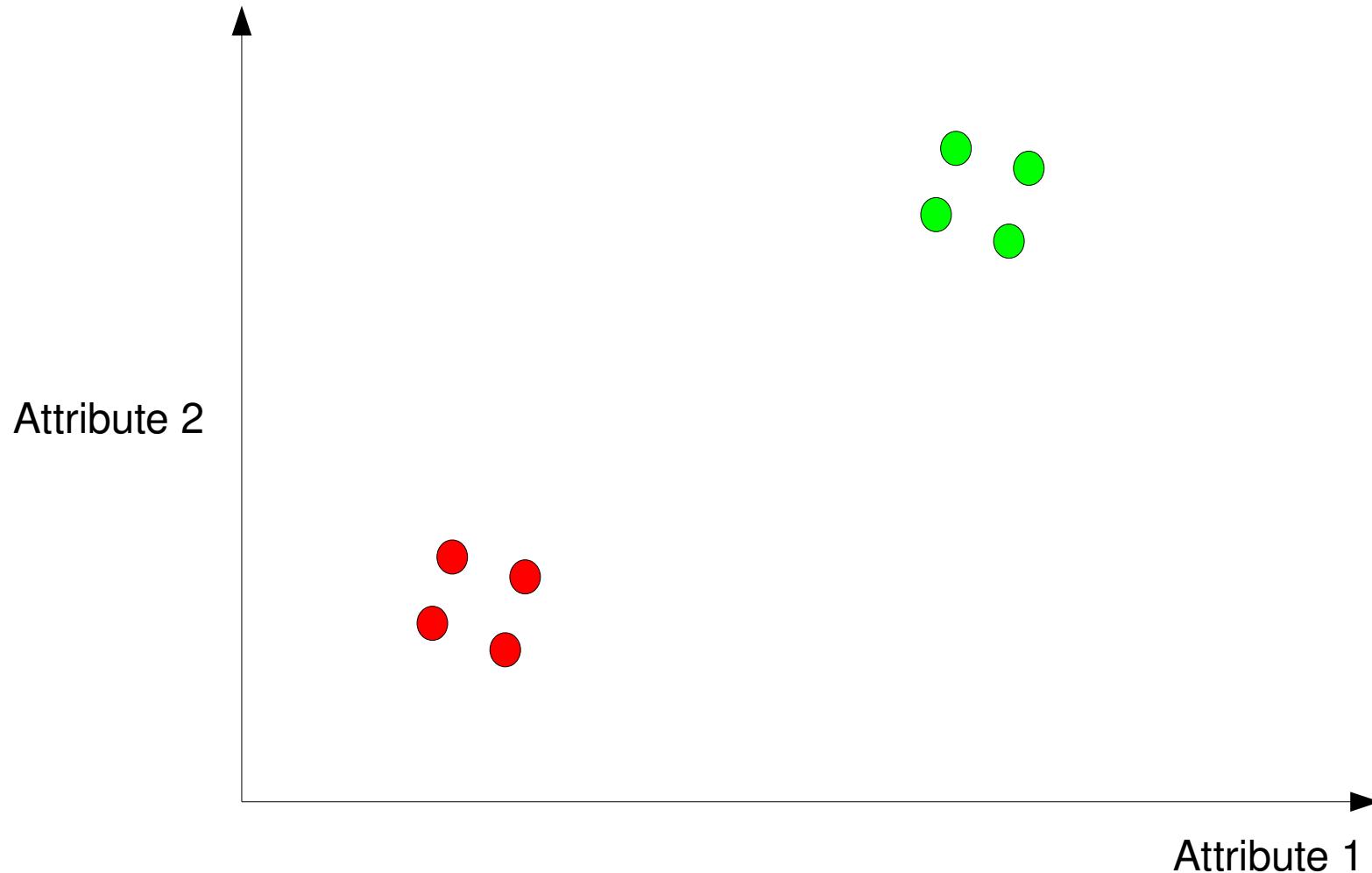
# Conclusions

- SLT allows to verify under what conditions learning occurs
- Observe:
  - **Without bias, the ERM Principle is not consistent**
    - i.e., we need some bias to learn!
  - Computing the VC Dimension for a set of functions  $F$ , we verify the consistency of the ERM Principle
  - We can also investigate algorithms that adapt their biases
    - This would allow us to better explore space  $F_{\text{all}}$  in attempt to define a good subspace  $F$  for our problem → **avoiding over and underfitting**
      - Even better if it contains  $f_{\text{Bayes}}$
- Besides other studies before Vladimir Vapnik, he was the main person involved in the development of the SLT
  - With the help of Alexey Chervonenkis

# **Assessing Supervised Learning Algorithms: The Distance-Weighted Nearest Neighbors as an example**

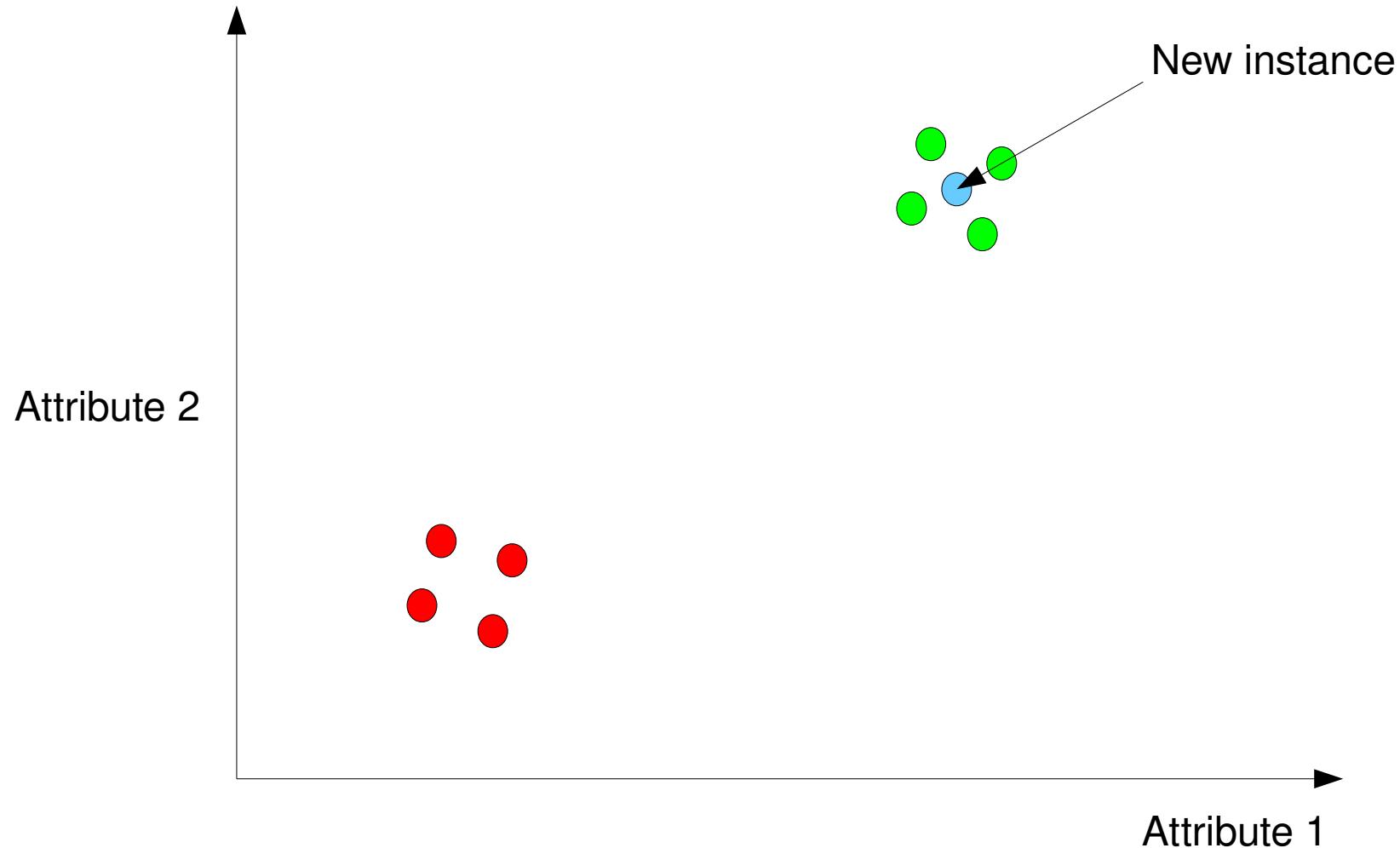
# Distance-Weighted Nearest Neighbors

- Based on the same principles as the k-Nearest Neighbors



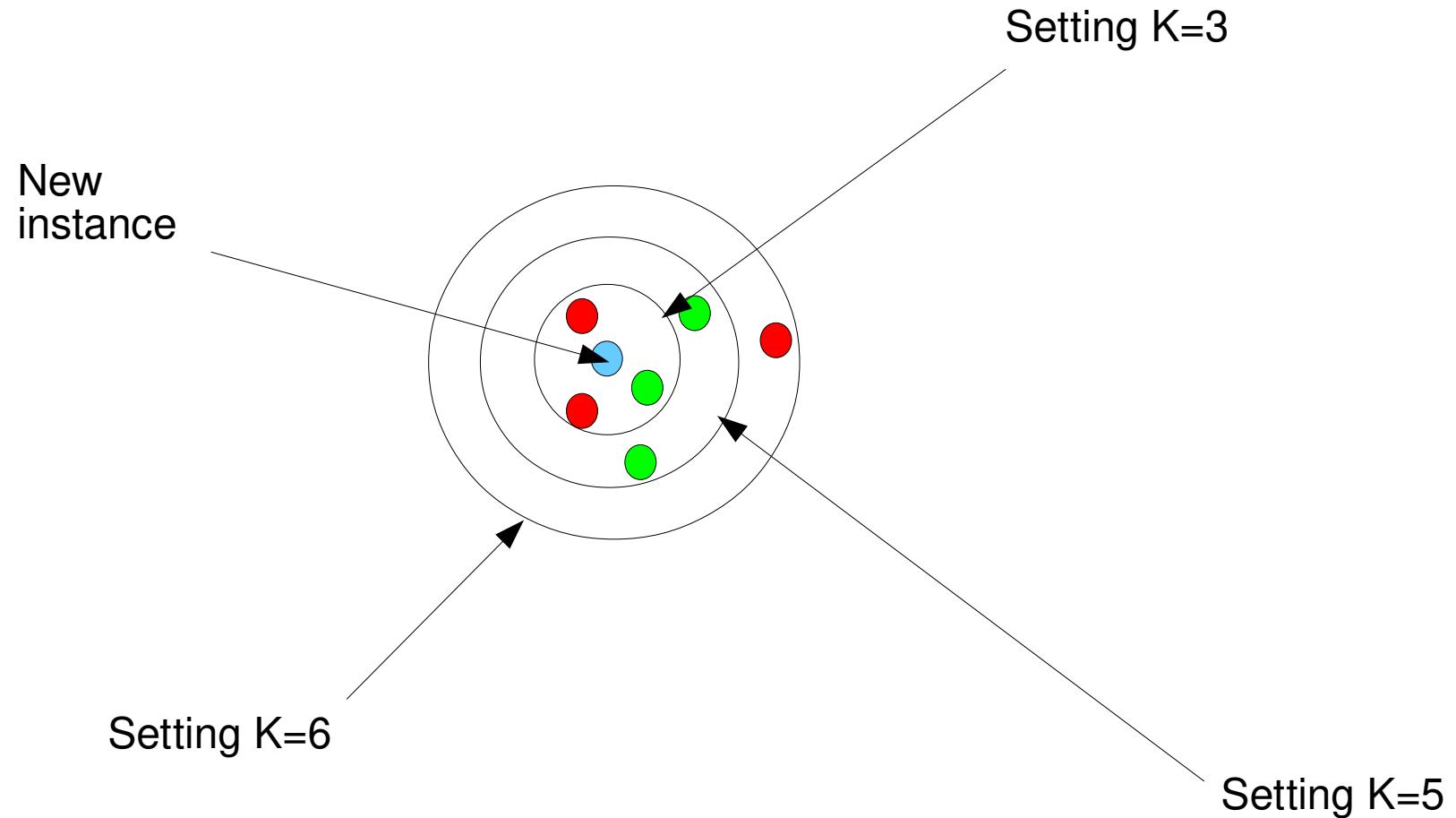
# Distance-Weighted Nearest Neighbors

- Based on the same principles as the k-Nearest Neighbors



# Distance-Weighted Nearest Neighbors

- Based on the same principles as the k-Nearest Neighbors



# Distance-Weighted Nearest Neighbors

- It is based on Radial functions centered at the new instance a.k.a. query point
- Classification output:

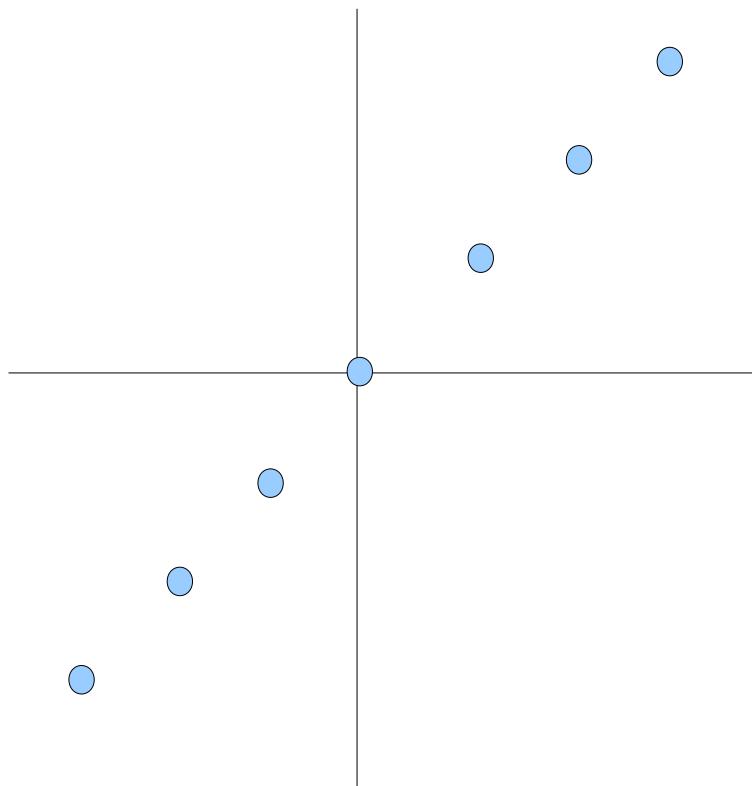
$$f(\mathbf{x}) = \frac{\sum_{i=1}^n w_i y_i}{\sum_{i=1}^n w_i}$$

- Given the weight function:

$$w_i = \exp - \frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{2\sigma^2}$$

# Distance-Weighted Nearest Neighbors

- After implementing, test it on this simple example of an identity function:



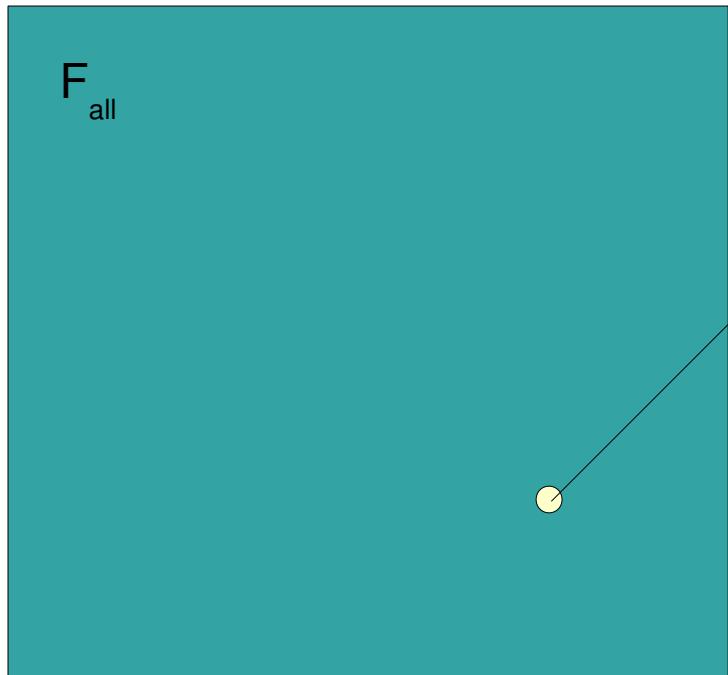
Two main questions:

- What happens if sigma is too big?
- What happens if sigma is too small?

So, how can we define the best value for sigma?

# Distance-Weighted Nearest Neighbors

- When sigma tends to infinity

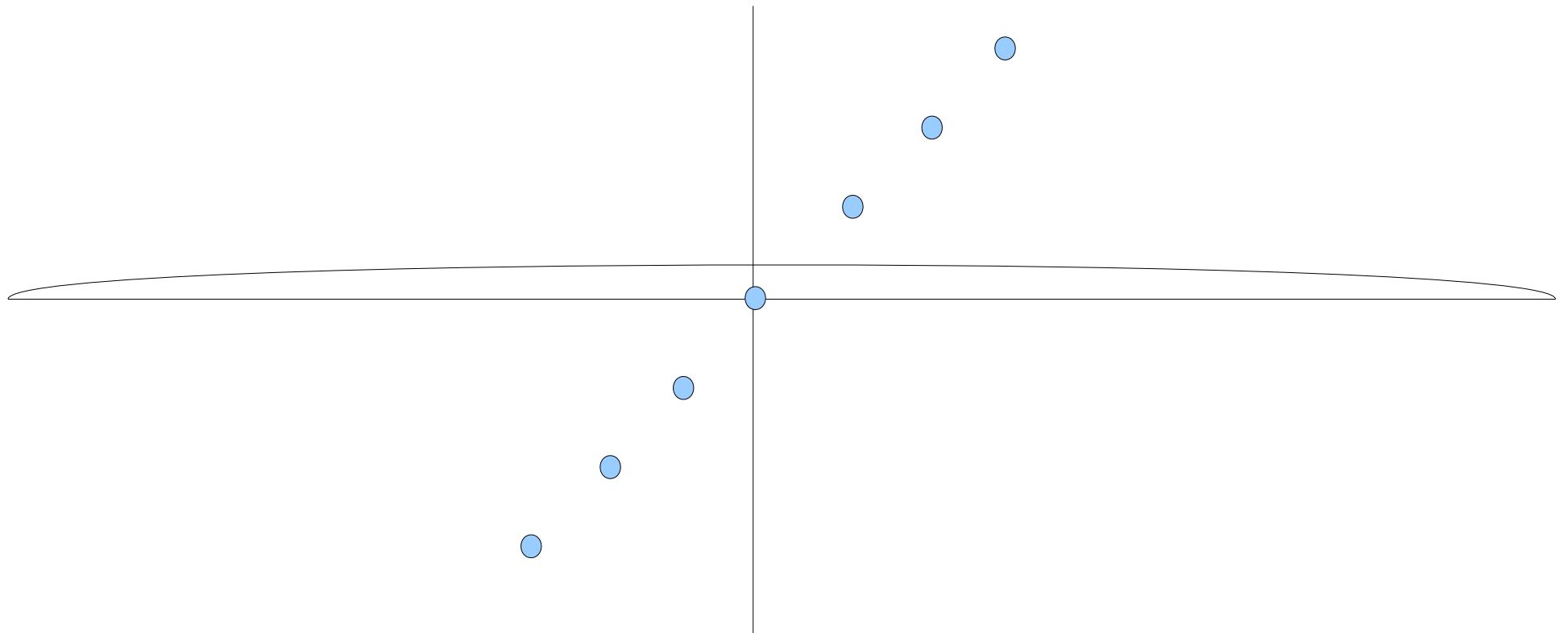


The space of admissible functions (bias) will contain a single function

In this case, the average function

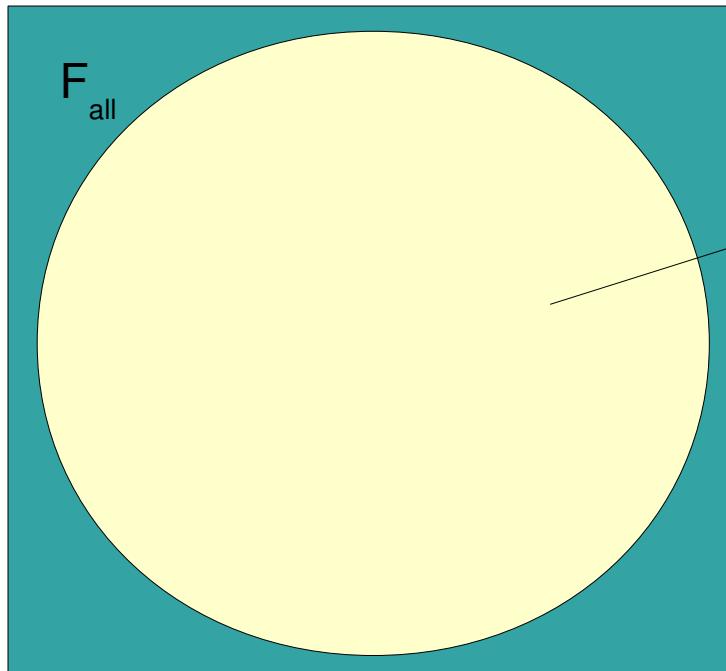
# Distance-Weighted Nearest Neighbors

- When sigma tends to infinity



# Distance-Weighted Nearest Neighbors

- When sigma tends to zero



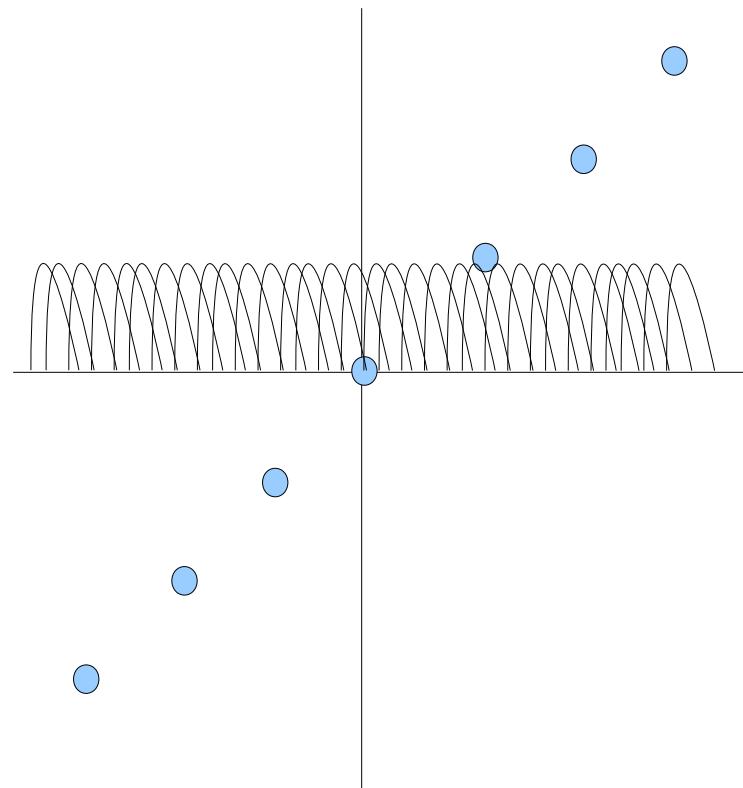
The space of admissible functions (bias) will tend to the whole space

What is the problem with that?

It will most probably contain at least one memory-based classifier

# Distance-Weighted Nearest Neighbors

- When sigma tends to zero



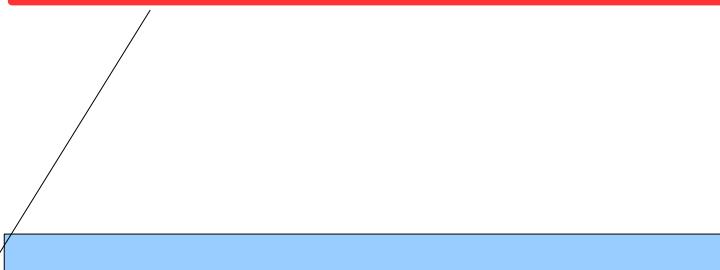
**Basic Concepts on Kernels  
so that we can understand  
Supervised Learning algorithms**

# Basic Concepts on Kernels

- Kernels
  - Kernels can be basically seen as similarity measures on a given data space
  - Important question:
    - Data may not be separable in a given input space
    - Could we modify such input space, i.e., translate it into another space, in order to make classes separable?

# Basic Concepts on Kernels

- Kernels
  - Kernels can be basically seen as similarity measures on a given data space
  - Important question:
    - Data may not be separable in a given input space
    - Could we modify such input space, i.e., translate it into another space, in order to make classes separable?



This is the main motivation for Kernels when talking about Classification algorithms

# Basic Concepts on Kernels

- Consider we have an input space labeled under two classes:

$$(x_1, y_1), \dots, (x_m, y_m) \in \mathcal{X} \times \{\pm 1\}$$

- We must **estimate a binary classifier**  $f$  capable of separating them according to classes
- To meet this requirement we need a **similarity measure**
  - In this way, we will know when examples are similar and thus mapped to the same label
  - The similarity measure to consider is a main concern for Machine Learning

# Basic Concepts on Kernels

- For instance, let a similarity measure be given by:

$$k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$$

- So given two instances  $x$  and  $x'$ , function  $k$  returns a real number which corresponds to the similarity between both
- Function  $k$  is called a kernel

# Basic Concepts on Kernels

- An example of similarity measure is the **dot product** or **scalar product**
  - For instance, consider two vectors:

$$\mathbf{x}, \mathbf{x}' \in \mathbb{R}^N$$

- Remember the **dot product** is given by:

$$\langle \mathbf{x}, \mathbf{x}' \rangle = \sum_{i=1}^N [\mathbf{x}]_i [\mathbf{x}']_i$$

- Or, in another form:

$$\langle \mathbf{x}, \mathbf{x}' \rangle = [\mathbf{x}]_1 [\mathbf{x}']_1 + \dots + [\mathbf{x}]_m [\mathbf{x}']_m$$

# Basic Concepts on Kernels

- We can interpret the **dot product** of two vectors by considering the magnitude of both with respect to their angle
  - If the angle is 90 degrees, then:

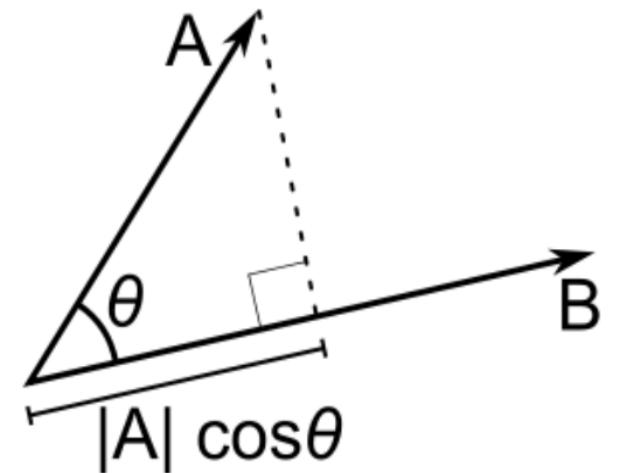
$$\langle \mathbf{x}, \mathbf{x}' \rangle = 0$$

- In another extreme, if  $\mathbf{x} = \mathbf{x}'$ , then:  $\|\mathbf{x}\|^2$
- Dot product can also be expressed using the cosine of the angle in form:

$$\langle \mathbf{x}, \mathbf{x}' \rangle = \|\mathbf{x}\| \|\mathbf{x}'\| \cos \theta$$

- In which  $\|\mathbf{x}\|$  is the length of  $\mathbf{x}$

$$\|\mathbf{x}\| = \sqrt{\langle \mathbf{x}, \mathbf{x} \rangle}$$



# Basic Concepts on Kernels

- In order to use the **dot product** as similarity measure we need to represent examples or objects in  $X$  as vectors in some **scalar product space  $H$** 
  - For that we use a function  $\Phi$  called **map** in form:

$$\Phi : \mathcal{X} \rightarrow \mathcal{H}$$

$$\Phi(x) = \mathbf{x}, \quad x \in \mathcal{X} \text{ e } \mathbf{x} \in \mathcal{H}$$

- Even having examples in a **scalar product space**, i.e., we can transform them using a mapping function to another **scalar product space**

# Basic Concepts on Kernels

- This space  $H$  is also called **feature space**
- After mapping every instance  $x$  into a vector  $\mathbf{x}$  in  $H$ , we can:
  - Compute the similarity between  $\mathbf{x}$  and  $\mathbf{x}'$  using the **dot product**:

$$k(x, x') = \langle \mathbf{x}, \mathbf{x}' \rangle = \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle$$

- This allows to study examples in a **geometric point of view**, using **linear algebra** and **analytic geometry**
- The freedom to choose different mapping functions  $\Phi$  allows us to evaluate different similarity measures

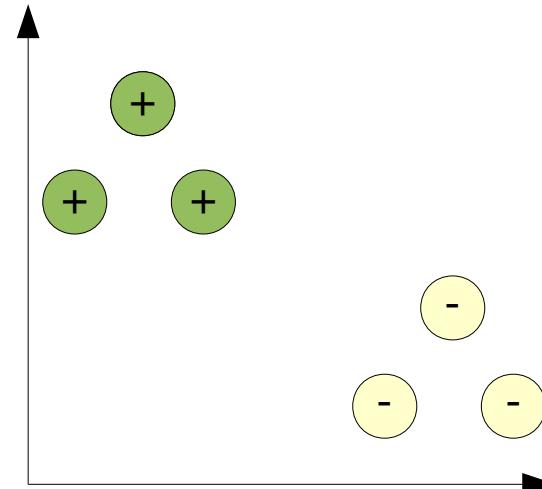
# **First Kernel-Based Supervised Learning Algorithm**

# First Kernel-Based Classification Algorithm

- We can better understand how a kernel works by studying a simple classification algorithm
  - Let training data be already in **feature space H**
  - Firstly we compute the averages for every class:

$$c_+ = \frac{1}{m_+} \sum_{i|y_i=+1} \mathbf{x}_i$$

$$c_- = \frac{1}{m_-} \sum_{i|y_i=-1} \mathbf{x}_i$$



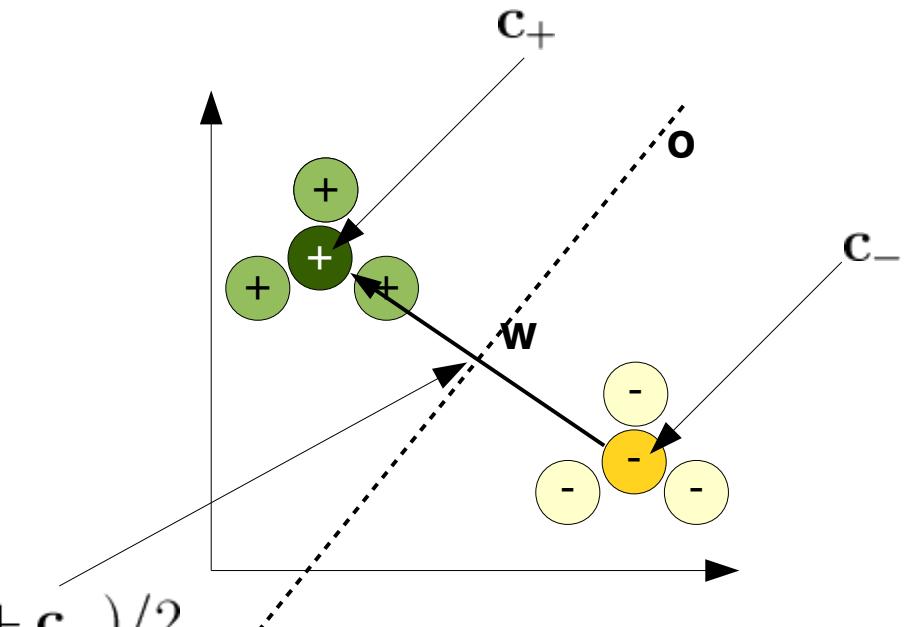
- In which  $m_+$  and  $m_-$  correspond to the number of examples in each class

# First Kernel-Based Classification Algorithm

- First kernel-based classification algorithm:
  - Now let us find the vector  $\mathbf{w} = \mathbf{c}_+ - \mathbf{c}_-$  between class averages
  - Now we compute vector  $\mathbf{o}$  which is orthogonal to  $\mathbf{w}$  and which passes through the central point  $\mathbf{c}$  of  $\mathbf{w}$ 
    - This vector  $\mathbf{o}$  defines a hiperplane to separate both classes

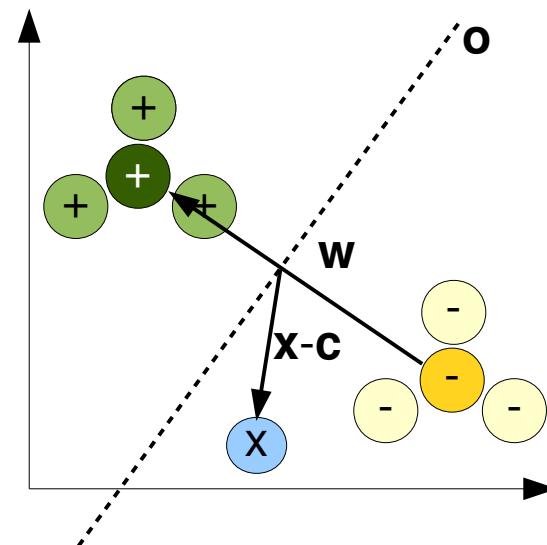
$$\mathbf{c}_+ = \frac{1}{m_+} \sum_{i|y_i=+1} \mathbf{x}_i$$
$$\mathbf{c}_- = \frac{1}{m_-} \sum_{i|y_i=-1} \mathbf{x}_i$$

$$\mathbf{c} = (\mathbf{c}_+ + \mathbf{c}_-)/2$$



# First Kernel-Based Classification Algorithm

- First kernel-based classification algorithm:
  - Now we analyze a new instance represented by vector  $\mathbf{x}$
  - We find the label of  $\mathbf{x}$  by verifying for which of the sides of  $\circ$  the vector  $\mathbf{x} - \mathbf{c}$  has an angle smaller than  $\pi/2$  with vector  $\mathbf{w}$

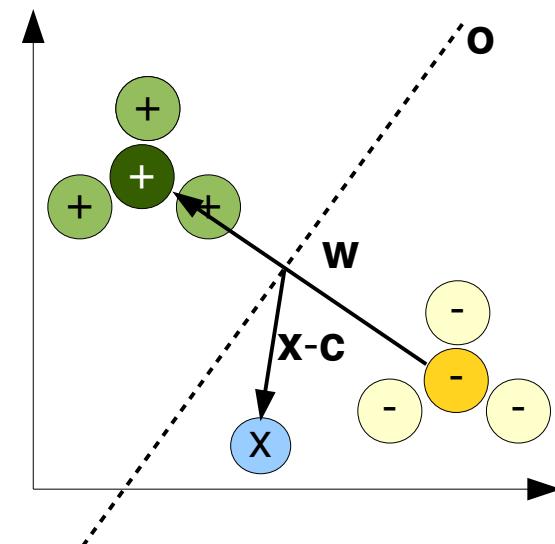


# First Kernel-Based Classification Algorithm

- First kernel-based classification algorithm:
  - What takes us to the **dot product**:

$$y = \operatorname{sgn} < (\mathbf{x} - \mathbf{c}), \mathbf{w} >$$

- If  $y > 0$  then vectors  $\mathbf{w}$  and  $\mathbf{x}-\mathbf{c}$  “pull” to the same side
- If  $y < 0$  then vectors  $\mathbf{w}$  and  $\mathbf{x}-\mathbf{c}$  “pull” to opposite sides
- If  $y = 0$ , they are orthogonal



# First Kernel-Based Classification Algorithm

- By solving the **dot product** we have:

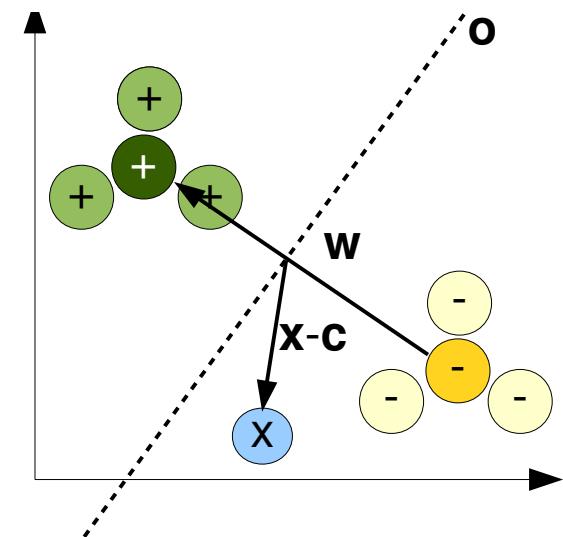
$$y = \text{sgn} < (\mathbf{x} - \mathbf{c}), \mathbf{w} >$$

$$< (\mathbf{x} - \mathbf{c}), \mathbf{w} > = < (\mathbf{x} - \mathbf{c}), (\mathbf{c}_+ - \mathbf{c}_-) > =$$

$$< (\mathbf{x} - \frac{1}{2}(\mathbf{c}_+ + \mathbf{c}_-)), (\mathbf{c}_+ - \mathbf{c}_-) > =$$

$$< \mathbf{x}, \mathbf{c}_+ > - < \mathbf{x}, \mathbf{c}_- > - \frac{1}{2}(< \mathbf{c}_+, \mathbf{c}_+ > - < \mathbf{c}_+, \mathbf{c}_- > + < \mathbf{c}_-, \mathbf{c}_+ > - < \mathbf{c}_-, \mathbf{c}_- >) =$$

$$< \mathbf{x}, \mathbf{c}_+ > - < \mathbf{x}, \mathbf{c}_- > + \frac{1}{2}(< \mathbf{c}_-, \mathbf{c}_- > - < \mathbf{c}_+, \mathbf{c}_+ >)$$



# First Kernel-Based Classification Algorithm

- Finally we have:

$$y = \text{sgn}(\langle \mathbf{x}, \mathbf{c}_+ \rangle - \langle \mathbf{x}, \mathbf{c}_- \rangle + b)$$

$$b = \frac{1}{2}(\langle \mathbf{c}_-, \mathbf{c}_- \rangle - \langle \mathbf{c}_+, \mathbf{c}_+ \rangle)$$

# First Kernel-Based Classification Algorithm

- So we can rewrite:

$$y = \text{sgn}(\langle \mathbf{x}, \mathbf{c}_+ \rangle - \langle \mathbf{x}, \mathbf{c}_- \rangle + b)$$

$$b = \frac{1}{2}(\langle \mathbf{c}_-, \mathbf{c}_- \rangle - \langle \mathbf{c}_+, \mathbf{c}_+ \rangle)$$

- In terms of a **kernel function** which considers every training vector  $\mathbf{x}_i$  as well as every test vector  $\mathbf{x}$  to be classified, assuming no set is empty, i.e., there are positive and negative training examples:

$$\begin{aligned} y &= \text{sgn} \left( \frac{1}{m_+} \sum_{i|y_i=+1} \langle \mathbf{x}, \mathbf{x}_i \rangle - \frac{1}{m_-} \sum_{i|y_i=-1} \langle \mathbf{x}, \mathbf{x}_i \rangle + b \right) \\ &= \text{sgn} \left( \frac{1}{m_+} \sum_{i|y_i=+1} k(x, x_i) - \frac{1}{m_-} \sum_{i|y_i=-1} k(x, x_i) + b \right) \end{aligned}$$

$$b = \frac{1}{2} \left( \frac{1}{m_-^2} \sum_{(i,j)|y_i=y_j=-1} k(x_i, x_j) - \frac{1}{m_+^2} \sum_{(i,j)|y_i=y_j=+1} k(x_i, x_j) \right)$$

# First Kernel-Based Classification Algorithm

- As homework:
  - Implement this first kernel-based classification algorithm
    - Employ two Normal distributions in a 2-dimensional space. Each one under a different class  $\{-1, +1\}$

# **Hyperplane-based classifiers: An intuitive view**

# Hyperplane-based classifiers

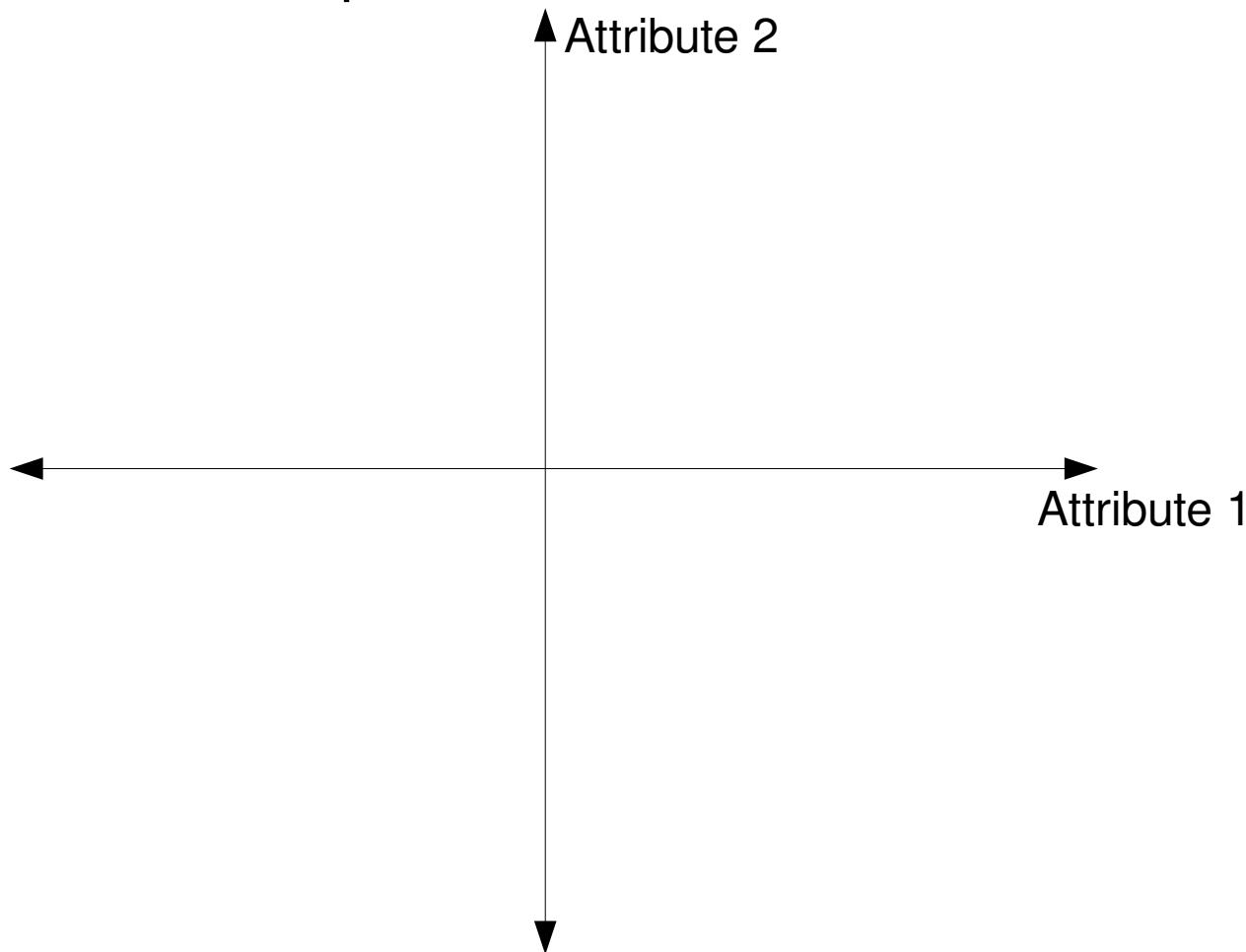
- Let us design a more-effective hyperplane-based learning algorithm in the **dot product space**
  - Motivations:
    - We consider there is a unique and optimal hyperplane
      - Distinguished by the maximum margin of separation
    - Consider two 2D Normal Distributions which are linearly separable:
      - Assume the iid sampling of those distributions to produce the training set
      - Assume they are fixed distributions
    - Estimating the hyperplane using a subset of training vectors will produce a fair-enough generalized solution
      - Compare the maximum-margin hyperplane versus the one produced by MLP

# Hyperplane-based classifiers

- Let us understand how to find this optimal hyperplane
  - Consider a 3-dimensional space
  - x-axis corresponds to attribute 1
  - y-axis corresponds to attribute 2
  - Z-axis corresponds to the output produced by the classification algorithm

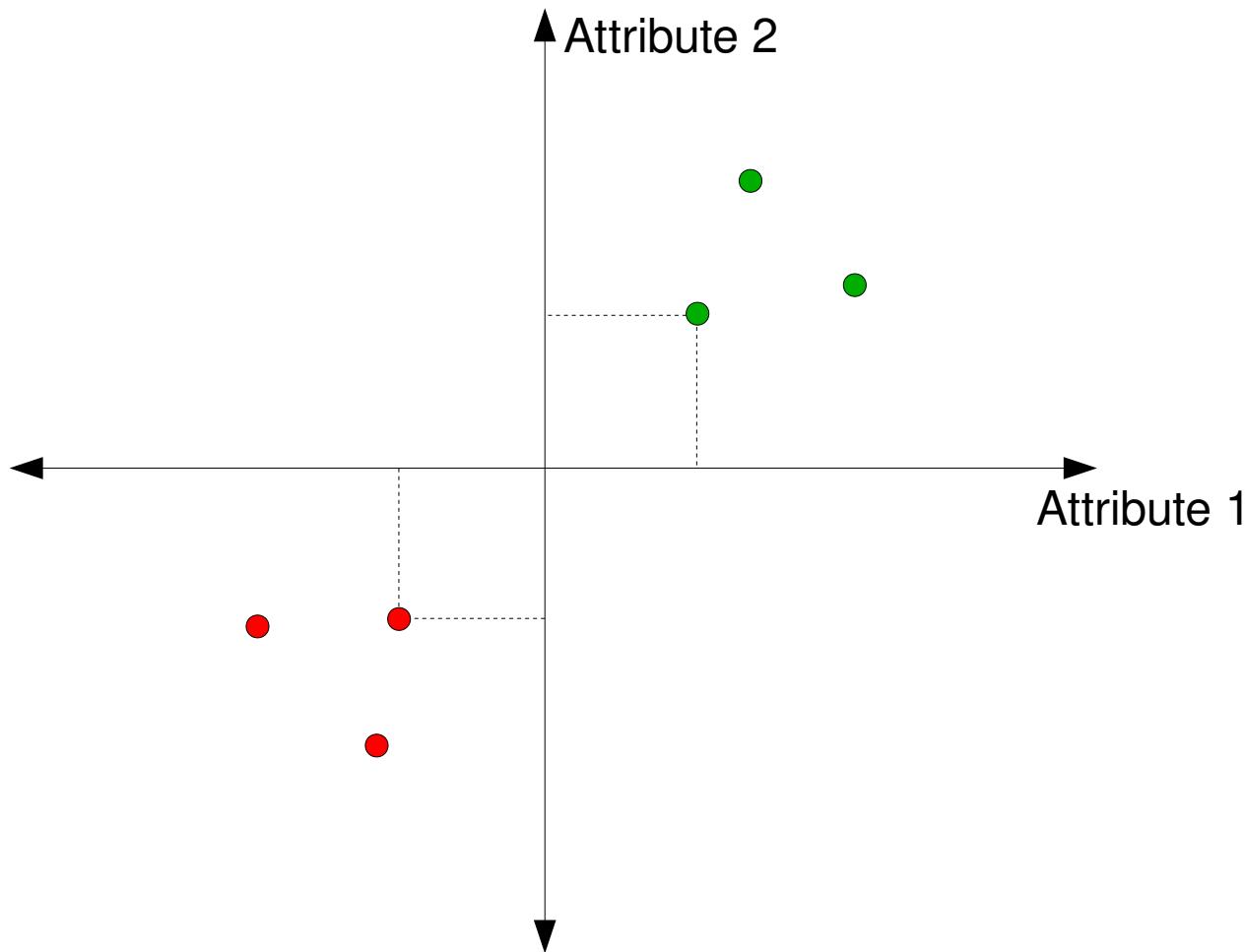
# Hyperplane-based classifiers

- Let us understand how to find this optimal hyperplane
  - We will draw only two dimensions to simplify understanding
    - Consider this space



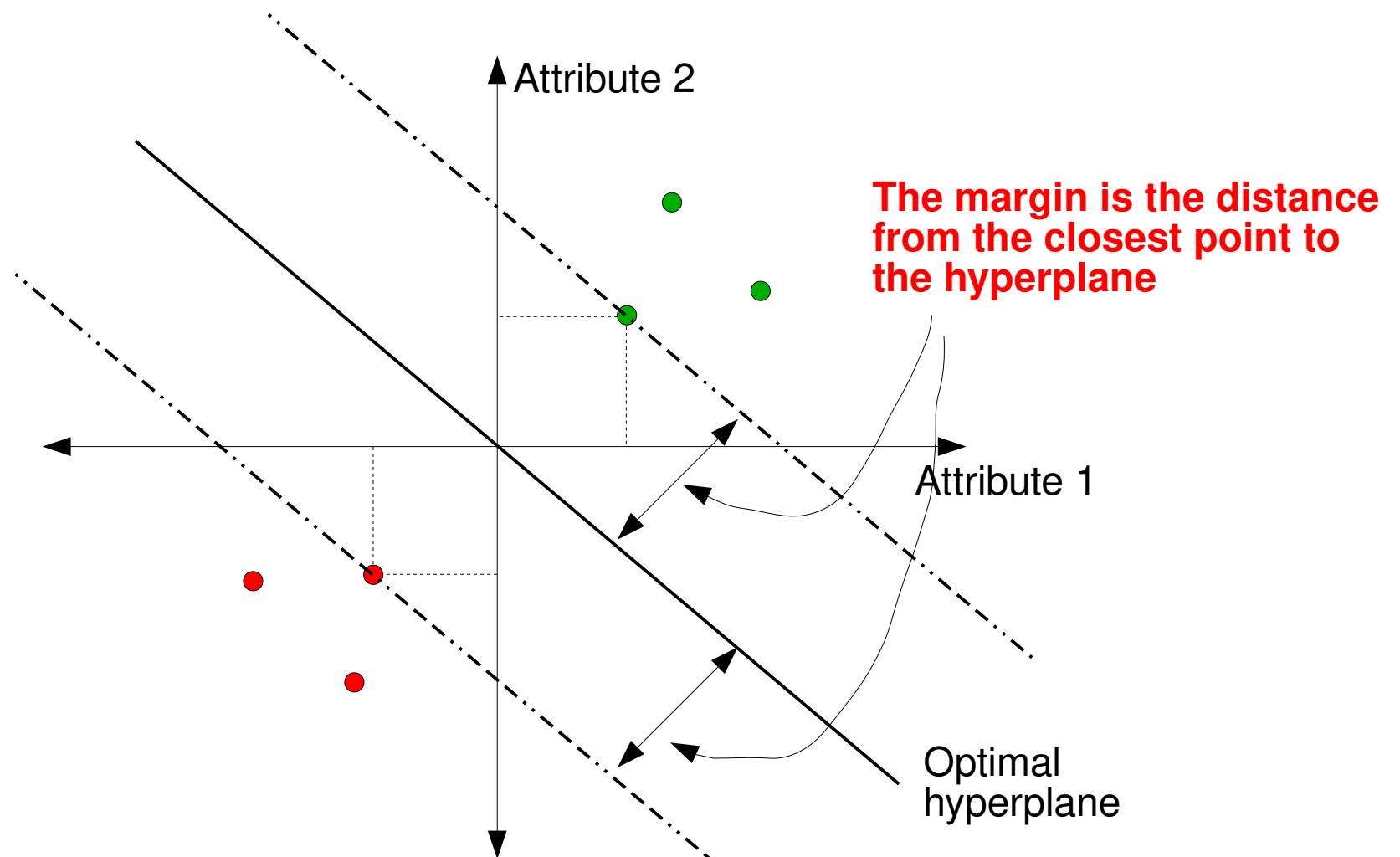
# Hyperplane-based classifiers

- Let us understand how to find this optimal hyperplane
  - Now plot the input examples (each color means one class)



# Hyperplane-based classifiers

- Let us understand how to find this optimal hyperplane
  - Our objective is to find the optimal hyperplane, i.e., the one with maximal margin



# Hyperplane-based classifiers

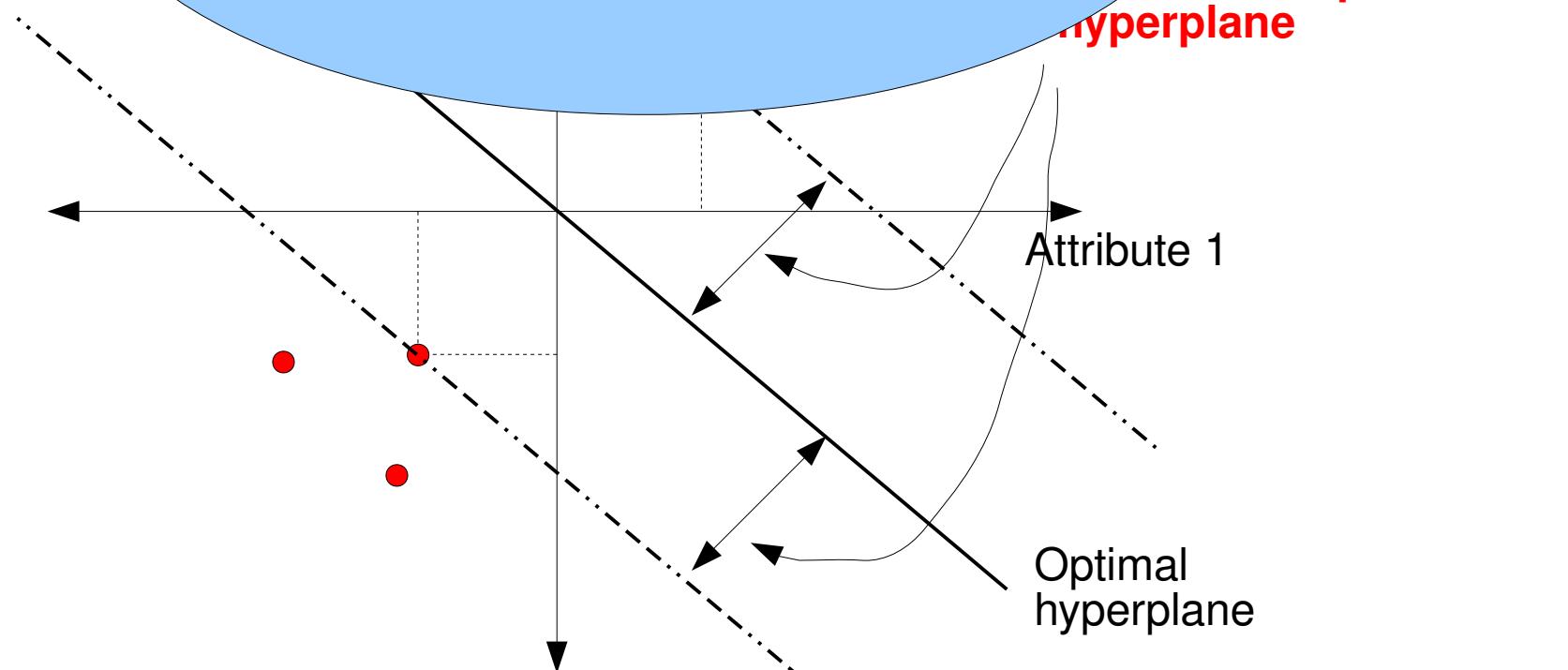
- Let us understand how to find this optimal hyperplane
  - Our objective is to find the optimal hyperplane, i.e., the one with maximum margin.

Why is it optimal?

Intuition:

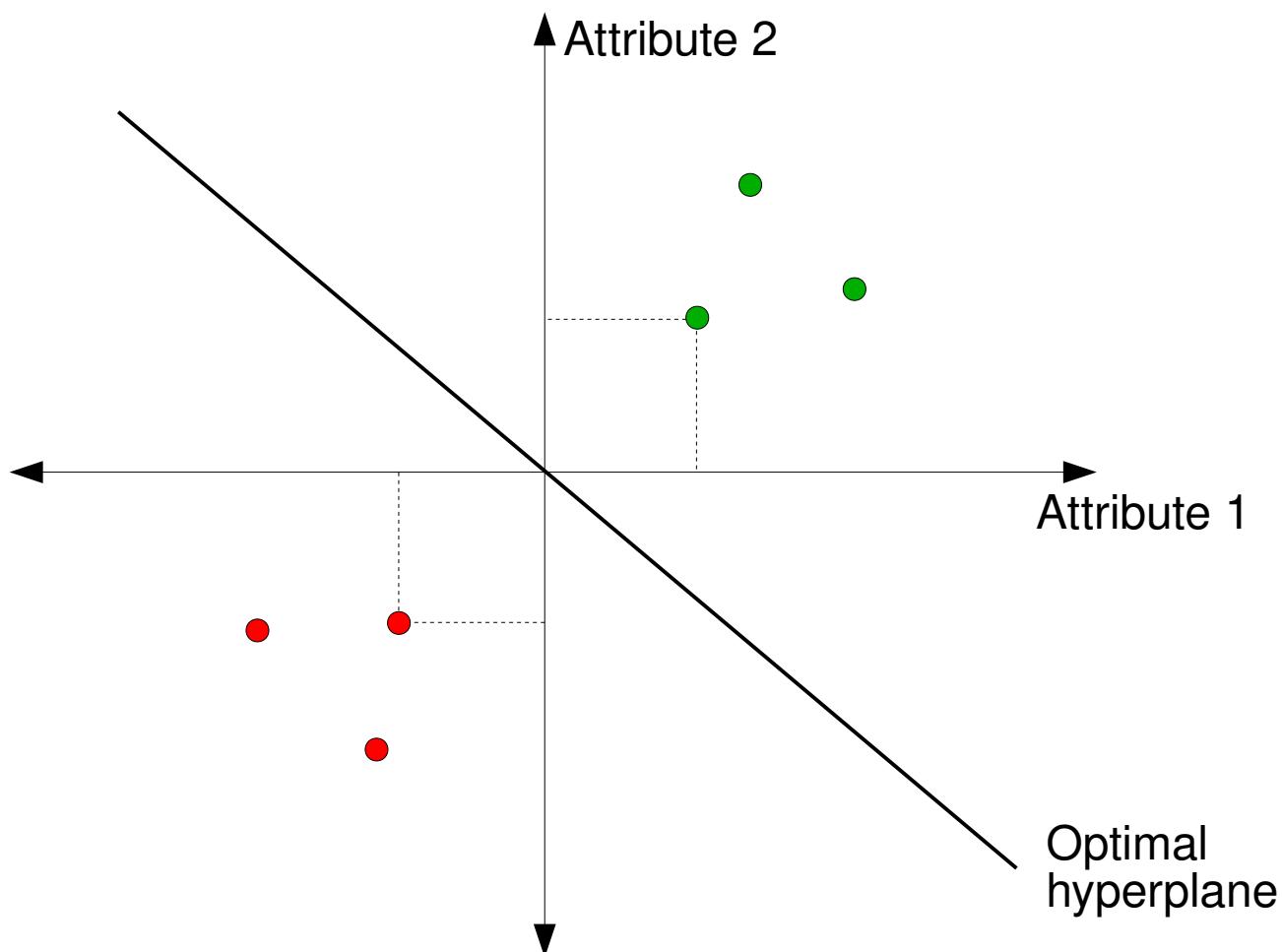
Because new inputs associated to different labels most probably fall in different sides of this hyperplane

**Margin is the distance from the closest point to the hyperplane**



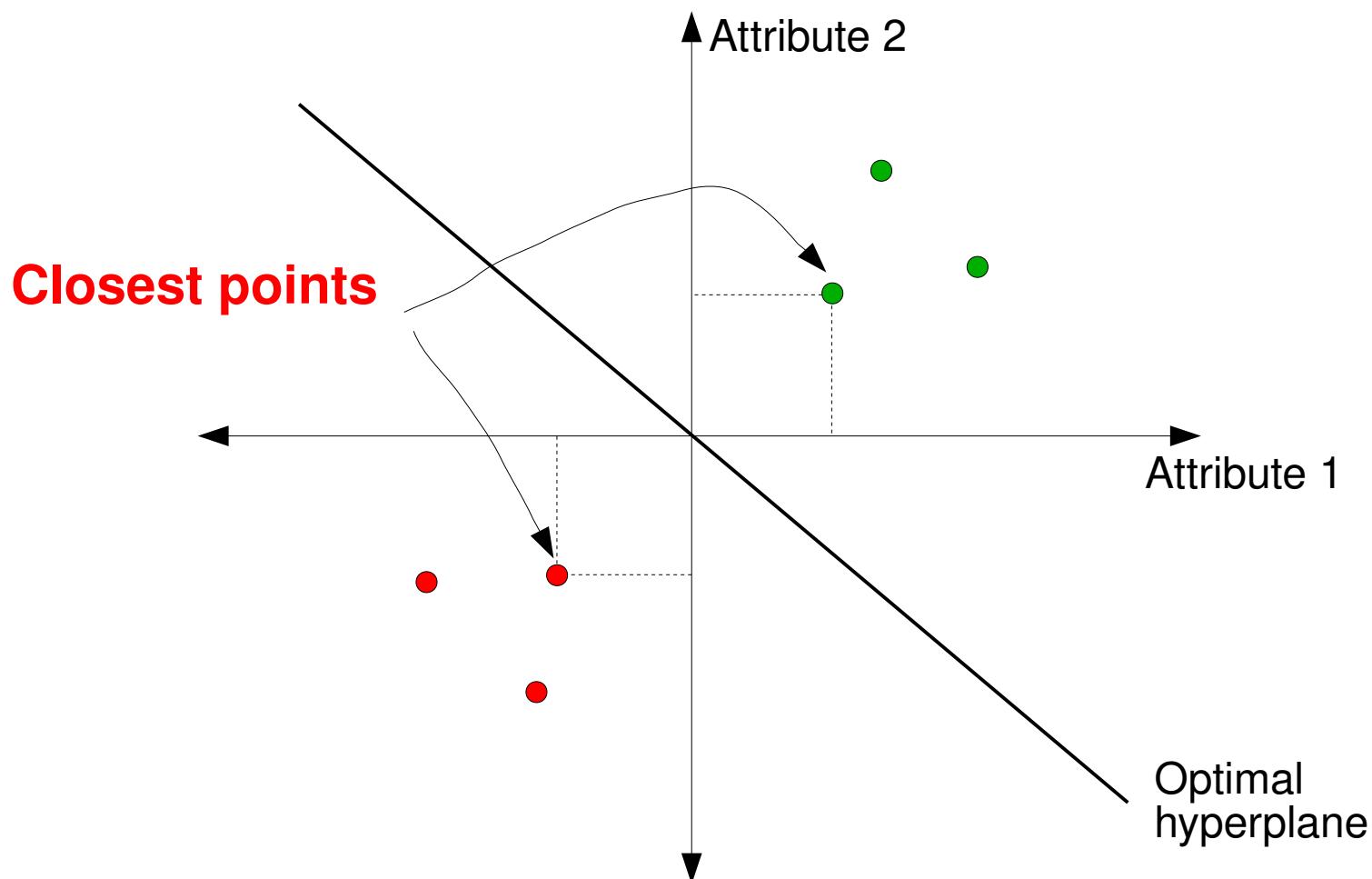
# Hyperplane-based classifiers

- Let us understand how to find this optimal hyperplane
  - This problem is solved so the point(s) closest to the hyperplane satisfy  $| \langle \mathbf{w}, \mathbf{x}_i \rangle + b | = 1$



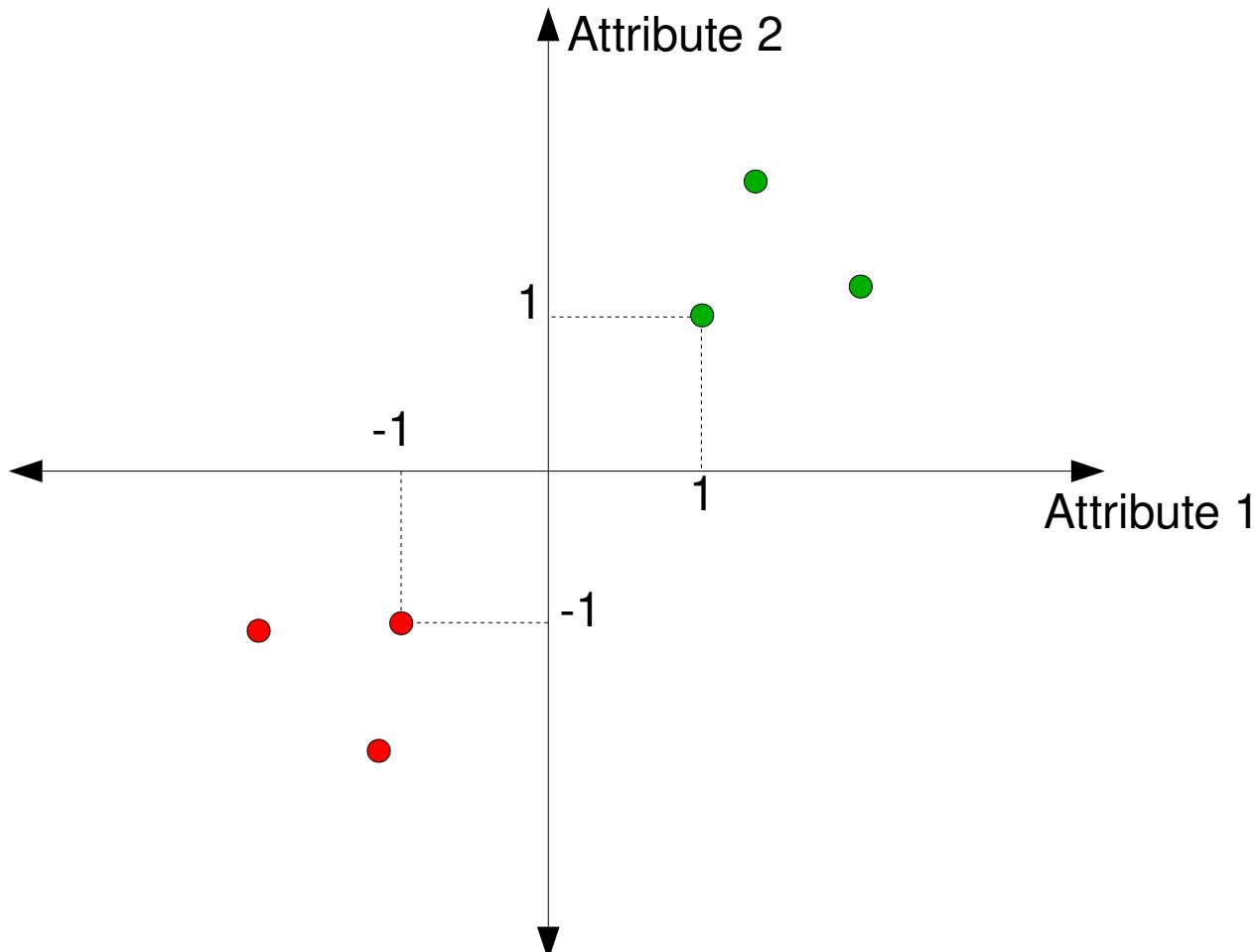
# Hyperplane-based classifiers

- Let us understand how to find this optimal hyperplane
  - This problem is solved so the point(s) closest to the hyperplane satisfy  $| \langle \mathbf{w}, \mathbf{x}_i \rangle + b | = 1$



# Hyperplane-based classifiers

- Let us understand how to find this optimal hyperplane
  - Now suppose we have values for points
    - How can we find  $w$  and  $b$  ???

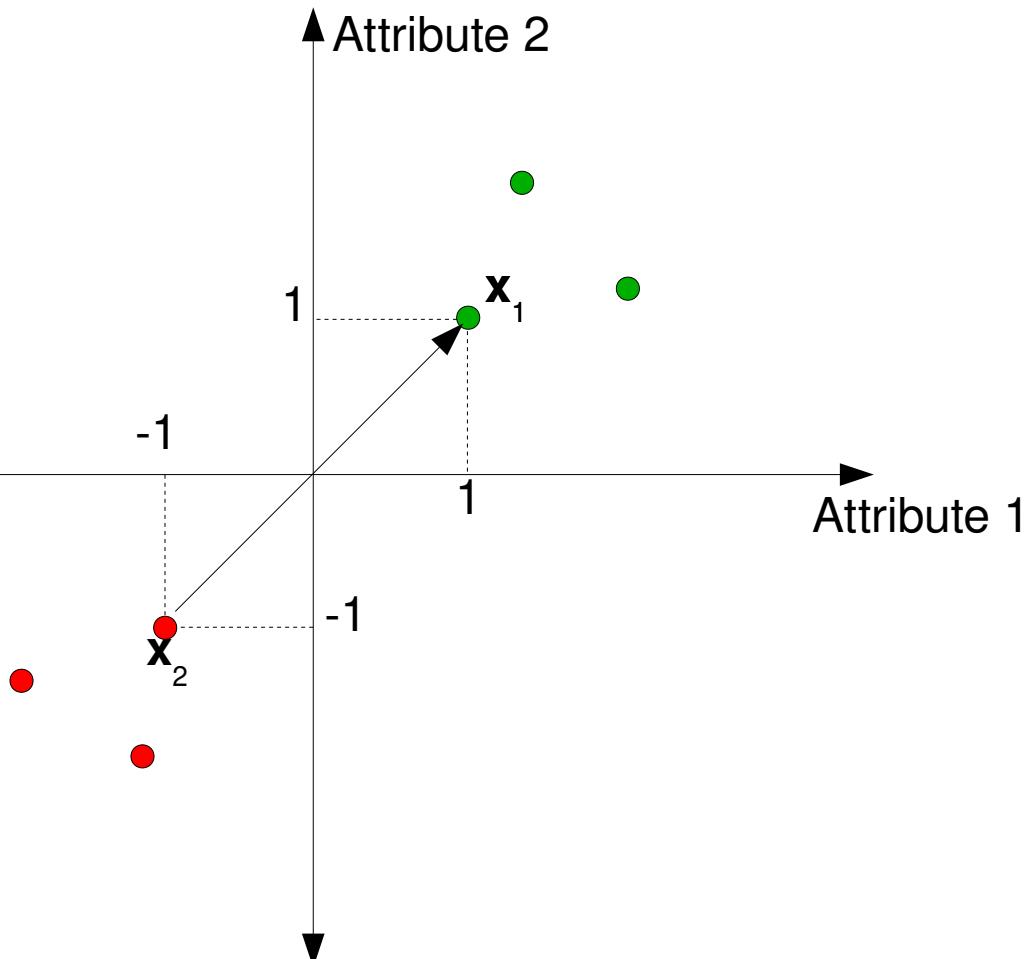


# Hyperplane-based classifiers

- Let us understand how to find this optimal hyperplane
  - Now suppose we have values for points
    - How can we find  $w$  and  $b$  ???

First we find the direction for  $w$  by finding the difference vector between the closest points

$$w = x_1 - x_2$$
$$w = \begin{bmatrix} 1 \\ 1 \end{bmatrix} - \begin{bmatrix} -1 \\ -1 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$$



# Hyperplane-based classifiers

- Let us understand how to find this optimal hyperplane
  - Now suppose we have values for points
    - How can we find  $w$  and  $b$  ???

Now we find  $b$ ...

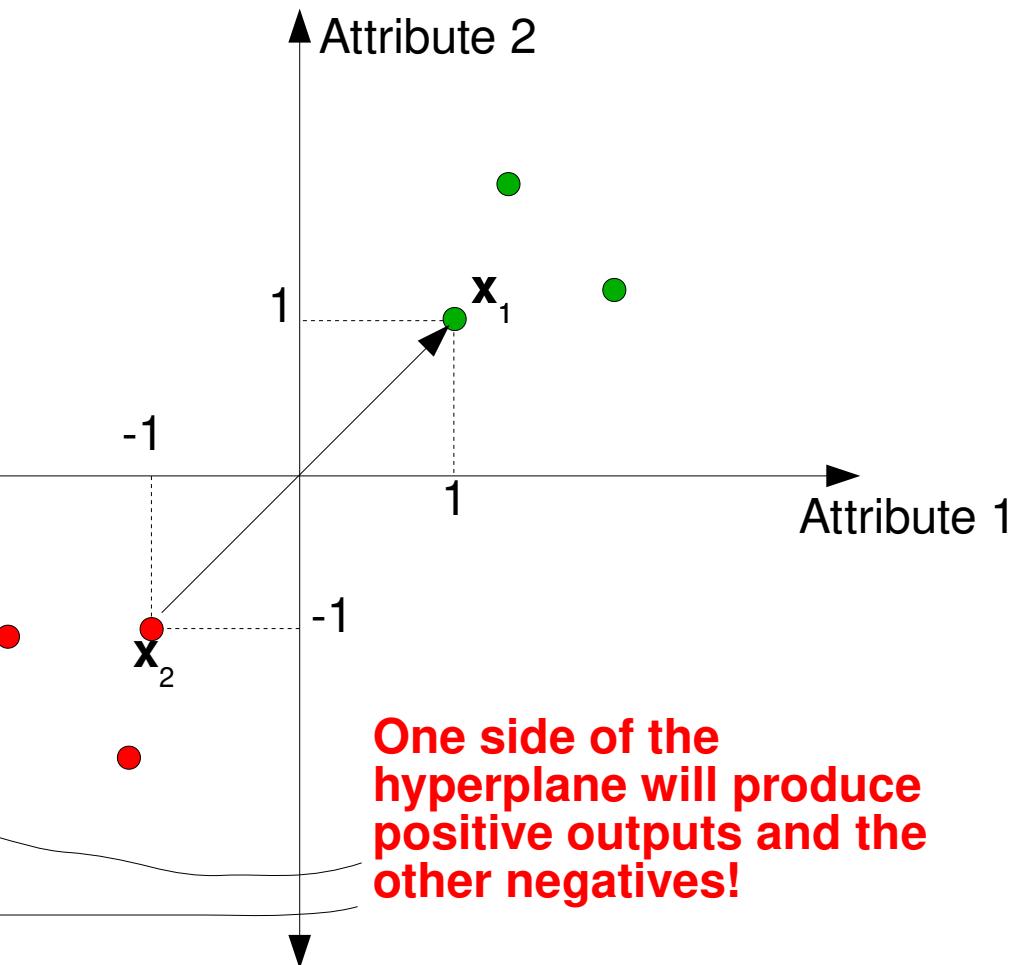
Remember the constraint mentioned earlier:

$$\begin{aligned} |\langle \mathbf{w}, \mathbf{x}_1 \rangle + b| &= 1 \\ |\langle \mathbf{w}, \mathbf{x}_2 \rangle + b| &= 1 \end{aligned}$$

Then we apply  $w$  on them and find  $b$ :

$$\begin{bmatrix} 2 \\ 2 \\ 2 \\ 2 \end{bmatrix} \mathbf{x}_1 + b = +1$$

$$\begin{bmatrix} 2 \\ 2 \\ 2 \\ 2 \end{bmatrix} \mathbf{x}_2 + b = -1$$



# Hyperplane-based classifiers

- Let us understand how to find this optimal hyperplane
  - Now suppose we have values for points
    - How can we find  $w$  and  $b$  ???

Solving for  $b$  we have:

$$\begin{bmatrix} 2 \\ 2 \end{bmatrix} [1 \ 1] + b = +1$$

$$\begin{bmatrix} 2 \\ 2 \end{bmatrix} [-1 \ -1] + b = -1$$

$$2 + 2 + b = +1 \Rightarrow b = -3$$

$$-2 + (-2) + b = -1 \Rightarrow b = +3$$

Well, how can we have two values for  $b$ ??? Is it possible to find a single value for  $b$  or rewrite this constraint in another way??

# Hyperplane-based classifiers

- Let us understand how to find this optimal hyperplane
  - What about if we rewrite the constraints in terms of outputs?

From:

$$\langle \mathbf{w}, \mathbf{x}_1 \rangle + b = +1$$

$$\langle \mathbf{w}, \mathbf{x}_2 \rangle + b = -1$$

We could multiply both sides by the expected output  $y_i$

$$y_1(\langle \mathbf{w}, \mathbf{x}_1 \rangle + b) = +1y_1$$

$$y_2(\langle \mathbf{w}, \mathbf{x}_2 \rangle + b) = -1y_2$$

What is the same as

$$y_1(\langle \mathbf{w}, \mathbf{x}_1 \rangle + b) = 1$$

$$y_2(\langle \mathbf{w}, \mathbf{x}_2 \rangle + b) = 1$$

# Hyperplane-based classifiers

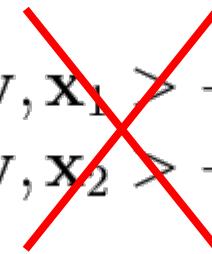
- Let us understand how to find this optimal hyperplane
  - However the equations below only hold for the two closest points of the hiperplane (one at each side)

$$y_1(\langle \mathbf{w}, \mathbf{x}_1 \rangle + b) = 1$$
$$y_2(\langle \mathbf{w}, \mathbf{x}_2 \rangle + b) = 1$$

- Try to apply vector  $\mathbf{w}$  and  $b$  we found there is no equality when considering all points in space!
  - Then, what do do???

# Hyperplane-based classifiers

- Let us understand how to find this optimal hyperplane
  - However the equations below only hold for the two closest points of the hiperplane (one at each side)

$$\begin{aligned}y_1(\langle \mathbf{w}, \mathbf{x}_1 \rangle + b) &= 1 \\y_2(\langle \mathbf{w}, \mathbf{x}_2 \rangle + b) &= 1\end{aligned}$$


- Try to apply vector  $\mathbf{w}$  and  $b$  we found there is no equality when considering all points in space!
  - Then, what do do???
  - We can rewrite as an equality, which now holds!

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$$

# Hyperplane-based classifiers

- Let us understand how to find this optimal hyperplane
  - Verify if the inequality constraint holds for  $\mathbf{w}$  and  $b=3$ :

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$$

- Testing for  $\mathbf{x}_1$ 
$$+ 1 \left( \begin{bmatrix} 2 \\ 2 \end{bmatrix} [1 \ 1] + 3 \right) \geq ? 1, \text{ yes, it is } 7$$

- Testing for  $\mathbf{x}_2$ 
$$- 1 \left( \begin{bmatrix} 2 \\ 2 \end{bmatrix} [-1 \ -1] + 3 \right) \geq ? 1, \text{ yes, it is } 1$$

# Hyperplane-based classifiers

- Let us understand how to find this optimal hyperplane
  - Verify if the inequality constraint holds for  $\mathbf{w}$  and  $b=-3$ :

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$$

- Testing for  $\mathbf{x}_1$

$$+ 1 \left( \begin{bmatrix} 2 \\ 2 \end{bmatrix} [1 \ 1] - 3 \right) \geq ? 1, \text{ yes, it is } 1$$

- Testing for  $\mathbf{x}_2$

$$- 1 \left( \begin{bmatrix} 2 \\ 2 \end{bmatrix} [-1 \ -1] - 3 \right) \geq ? 1, \text{ yes, it is } 7$$

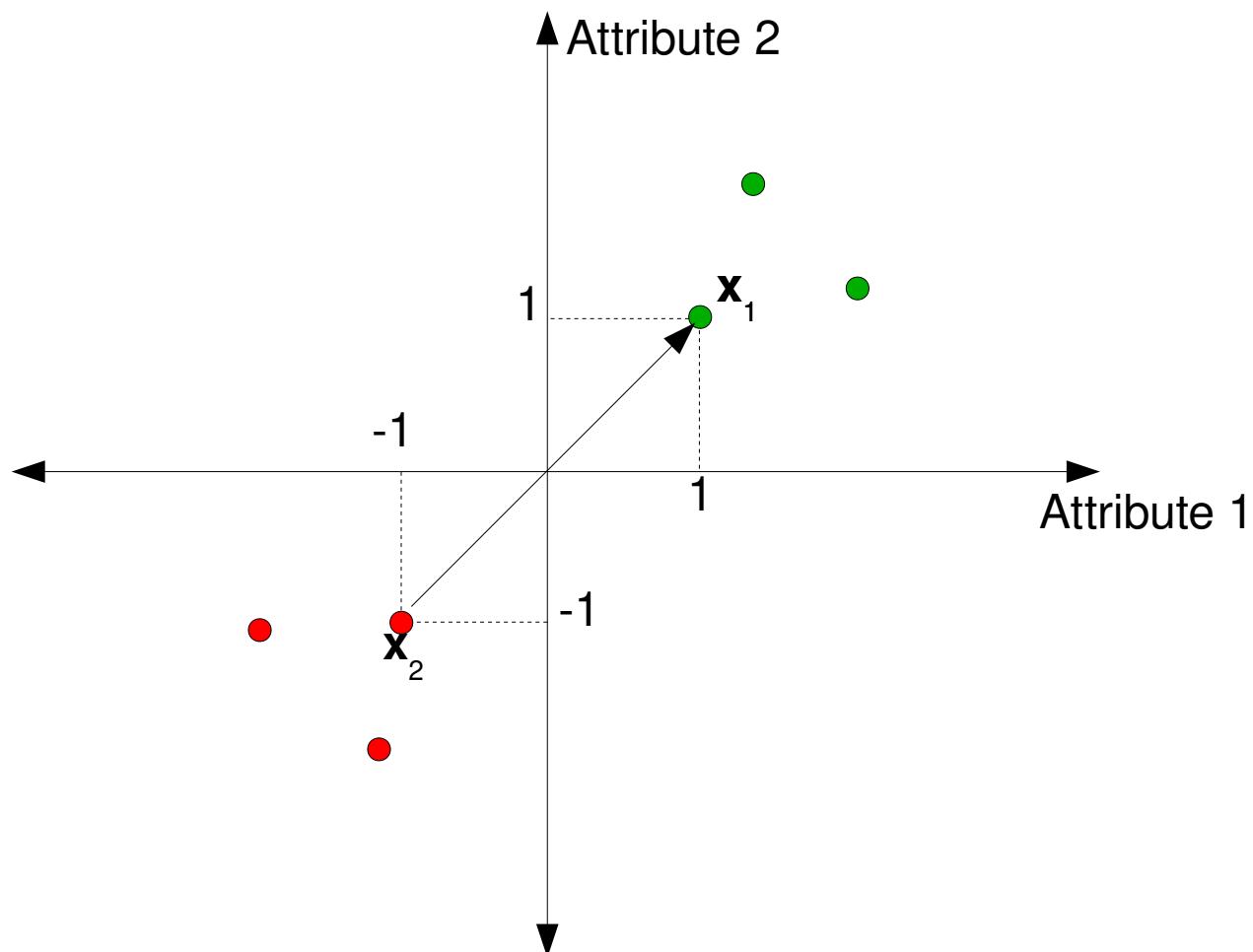
# Hyperplane-based classifiers

- Let us understand how to find this optimal hyperplane

So, now we have:

$$\mathbf{w} = \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \quad b = \pm 3$$

What does this mean?



# Hyperplane-based classifiers

- Let us understand how to find this optimal hyperplane

So, now we have:

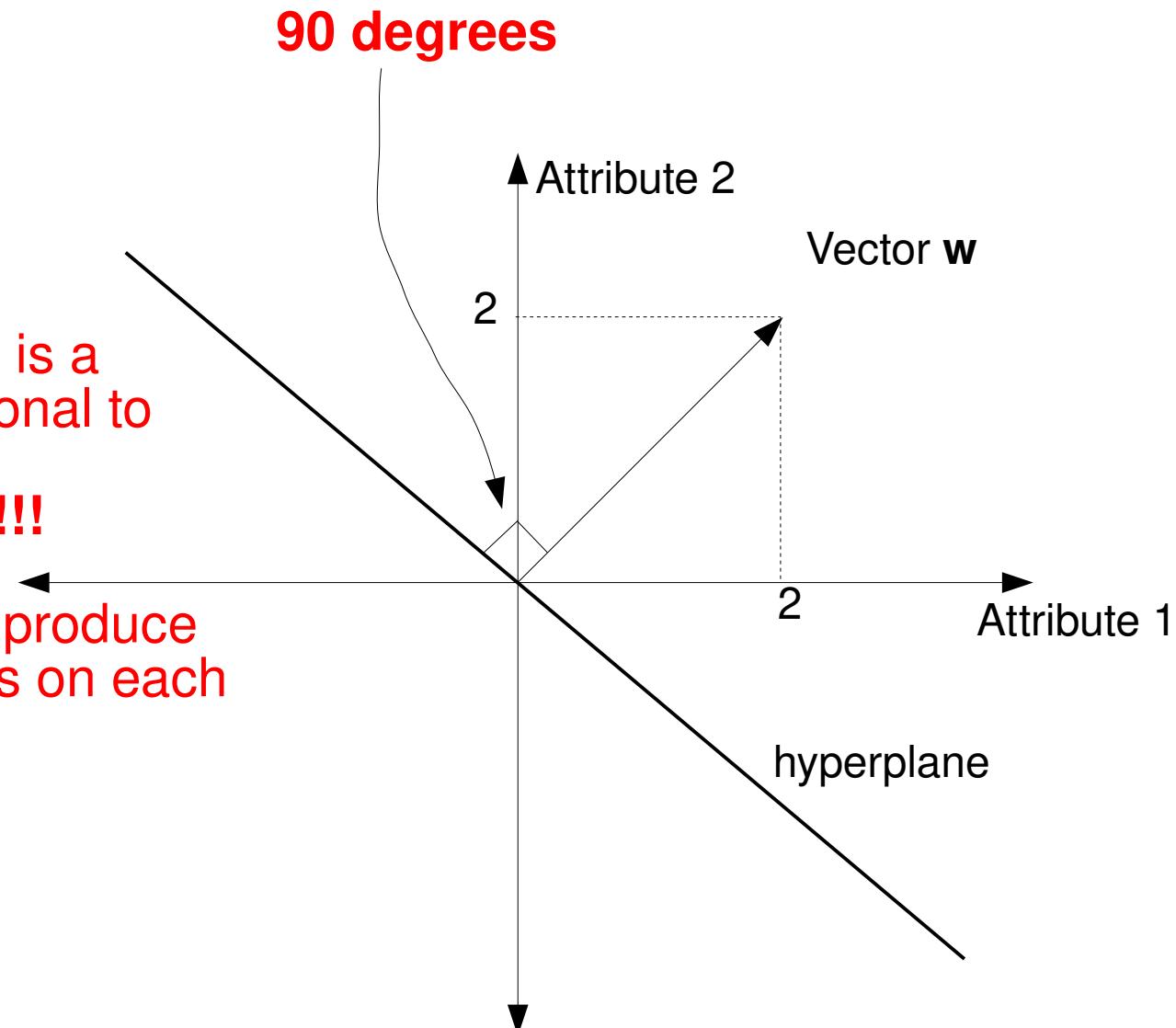
$$\mathbf{w} = \begin{bmatrix} 2 \\ 2 \end{bmatrix}, \quad b = \pm 3$$

What does this mean?

- First of all it means there is a hyperplane which is orthogonal to vector  $\mathbf{w}$

**Observe the angle!!!**

- Term  $b$  is a correction to produce positive and negative values on each side of this hyperplane



# Hyperplane-based classifiers

- Now we do have a hyperplane
  - We have values for  $w$  and  $b$ !!!
  - The question is: are there other values for  $w$  and  $b$  which satisfy the inequality constraint???
    - Yes! Infinite possibilities!
    - For example:

$$\mathbf{w} = \begin{bmatrix} 10 \\ 10 \end{bmatrix}, \quad b = \pm 19$$

# Hyperplane-based classifiers

- Now we do have a hyperplane
  - However, if we plug them we will have outputs very far from 1

$$\mathbf{w} = \begin{bmatrix} 10 \\ 10 \end{bmatrix}, \quad b = \pm 19$$

$$+ 1 \left( \begin{bmatrix} 10 \\ 10 \end{bmatrix} [1 \ 1] + 19 \right) \geq 1, \text{ yes, but it is } 39$$

$$- 1 \left( \begin{bmatrix} 10 \\ 10 \end{bmatrix} [-1 \ -1] + 19 \right) \geq 1, \text{ yes, it is } 1$$

- Observe when greater values for  $\mathbf{w}$  and  $b$  are used, we get outputs very far from 1

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$$

# Hyperplane-based classifiers

- Now we do have a hyperplane
  - So what is the closest solution to produce values equal to 1???

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$$

- Let's guess:

$$\mathbf{w} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}, \quad b = 0$$

$$+ 1 \left( \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} [1 \ 1] + 0 \right) \geq 1, \text{ yes, it is 1}$$

$$- 1 \left( \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} [-1 \ -1] + 0 \right) \geq 1, \text{ yes, it is 1}$$

- This is close enough! It produced exactly 1 as output

# Hyperplane-based classifiers

- So which is the best solution?
  - It is the one that minimizes  $\mathbf{w}$  and  $b$  subject to the inequality constraints
  - This is called the optimal hyperplane:

$$\min_{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}} \tau(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|^2$$

subject to  $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$  for all  $i = 1, \dots, m.$

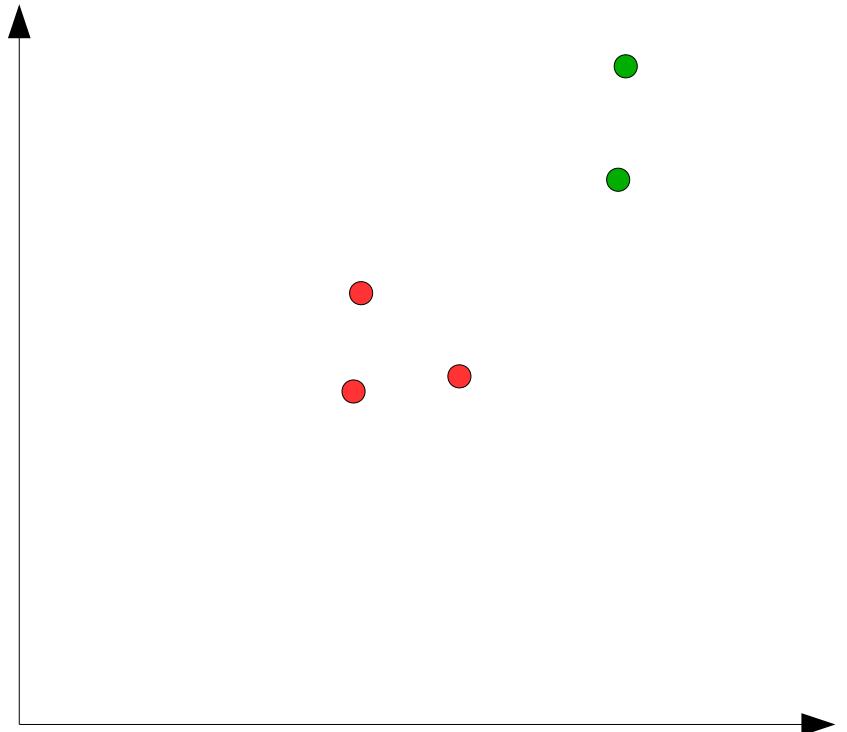
- But how to solve this optimization problem?

# **Hyperplane-based classifiers: An Algebraic view**

# Hyperplane-based classifiers: an algebraic view

- Another **MUCH BETTER** approach considers Linear Algebra

Consider positive (green) and negative (red) examples in  $\mathbb{R}^2$

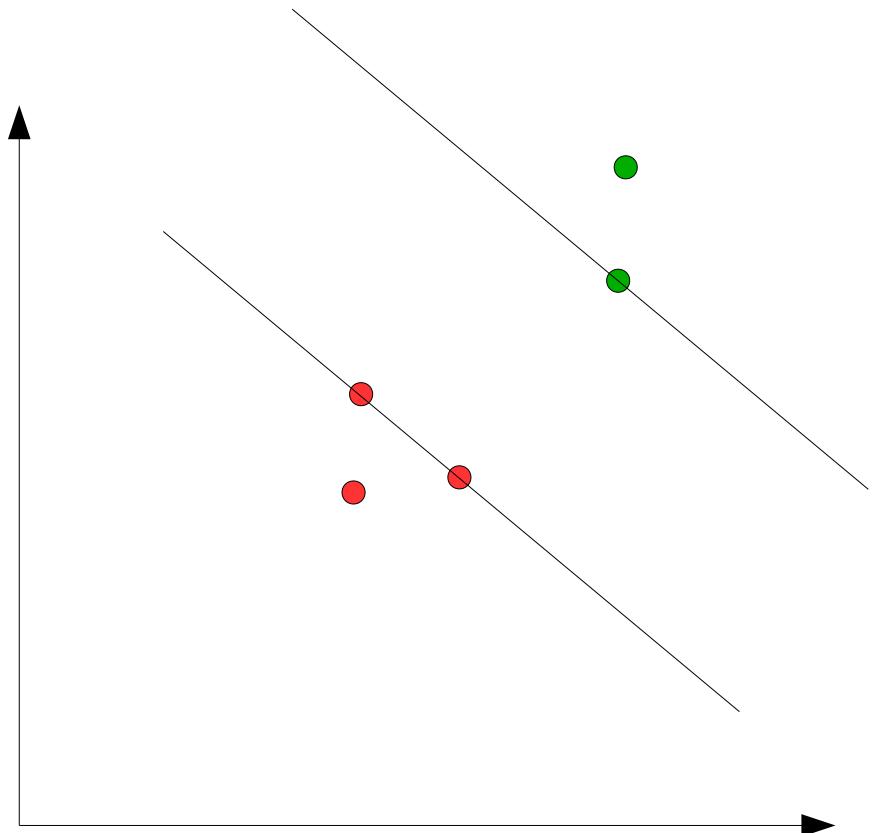


# Hyperplane-based classifiers: an algebraic view

- Another **MUCH BETTER** approach considers Linear Algebra

Consider positive (green) and negative (red) examples in  $\mathbb{R}^2$

Now let two linear hyperplanes which define the “sidewalks” for a “street”



# Hyperplane-based classifiers: an algebraic view

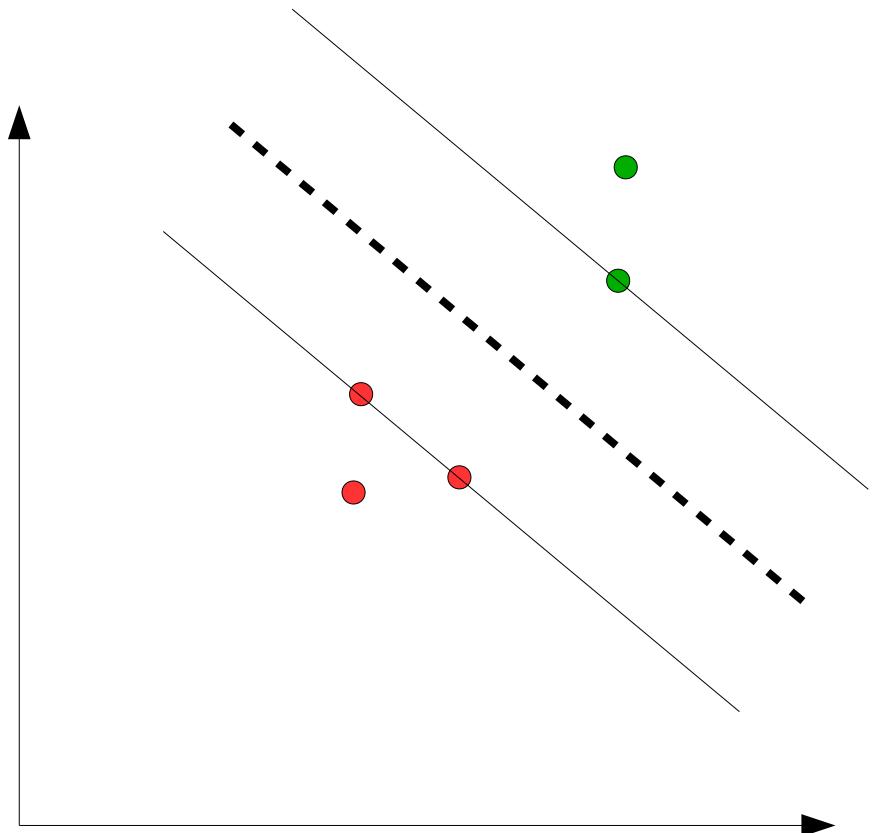
- Another **MUCH BETTER** approach considers Linear Algebra

Consider positive (green) and negative (red) examples in  $\mathbb{R}^2$

Now let two linear hyperplanes which define the “sidewalks” for a “street”

Observe that we can divide the “street” in half

Then we can formulate this problem algebraically..

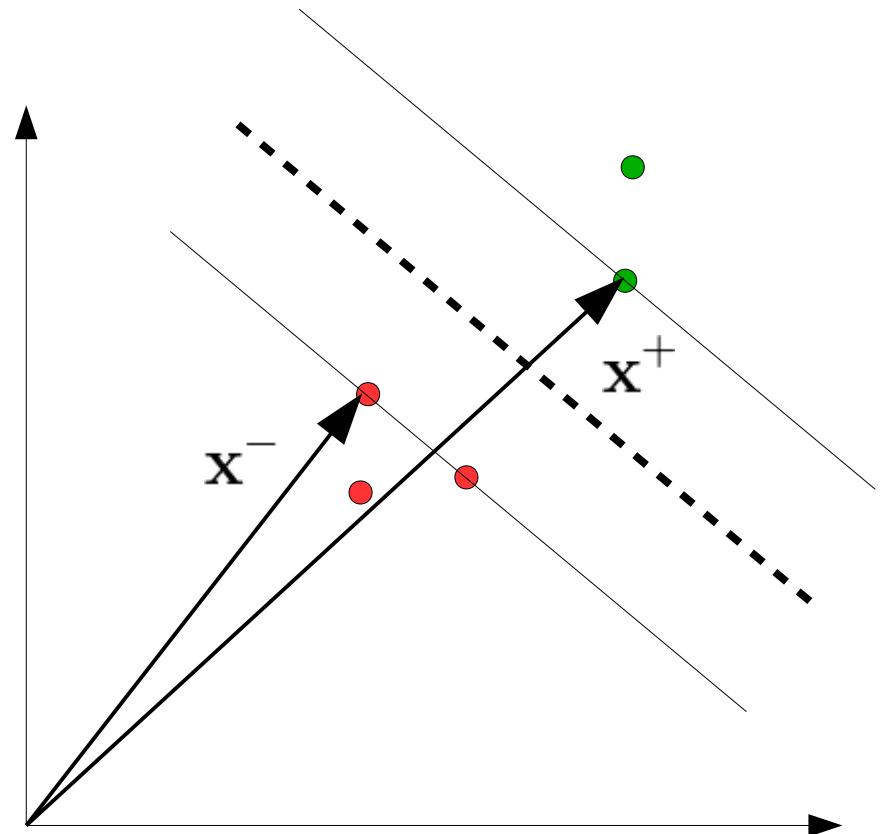


# Hyperplane-based classifiers: an algebraic view

- Another **MUCH BETTER** approach considers Linear Algebra

Formulating this problem algebraically...

We have two support vectors plotted, one for each class



# Hyperplane-based classifiers: an algebraic view

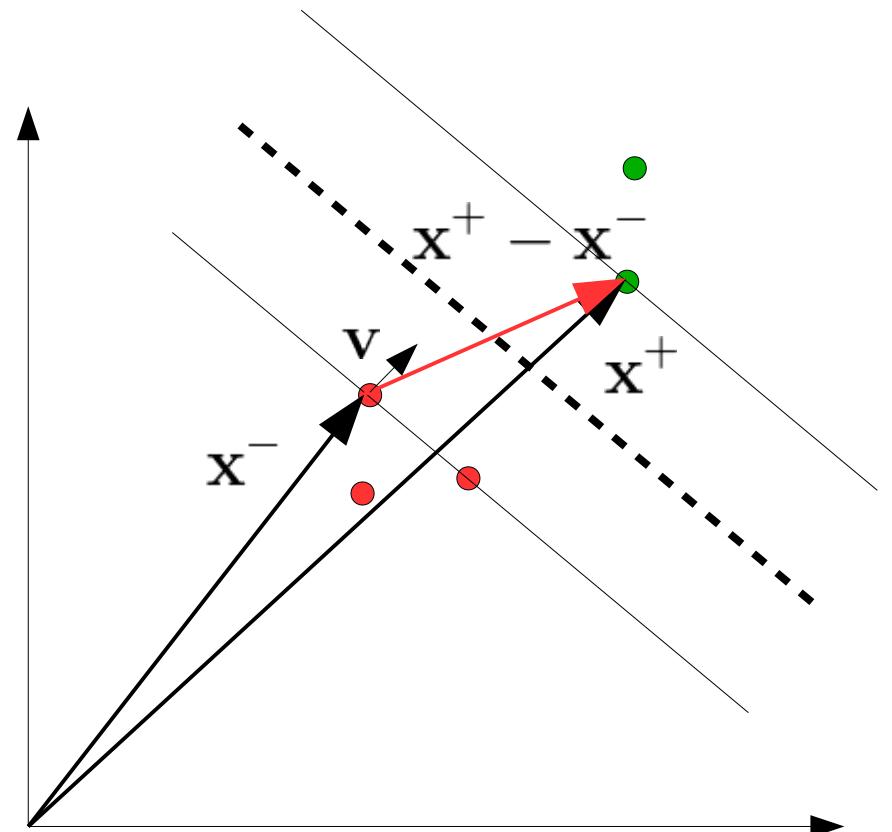
- Another **MUCH BETTER** approach considers Linear Algebra

Formulating this problem algebraically...

We have two support vectors plotted, one for each class

The difference vector projected over the unitary vector  $v$  orthogonal to both support vectors hyperplanes defines the “street” width

$$\text{Proj}_v(x^+ - x^-) = \langle x^+ - x^-, v \rangle$$



# Hyperplane-based classifiers: an algebraic view

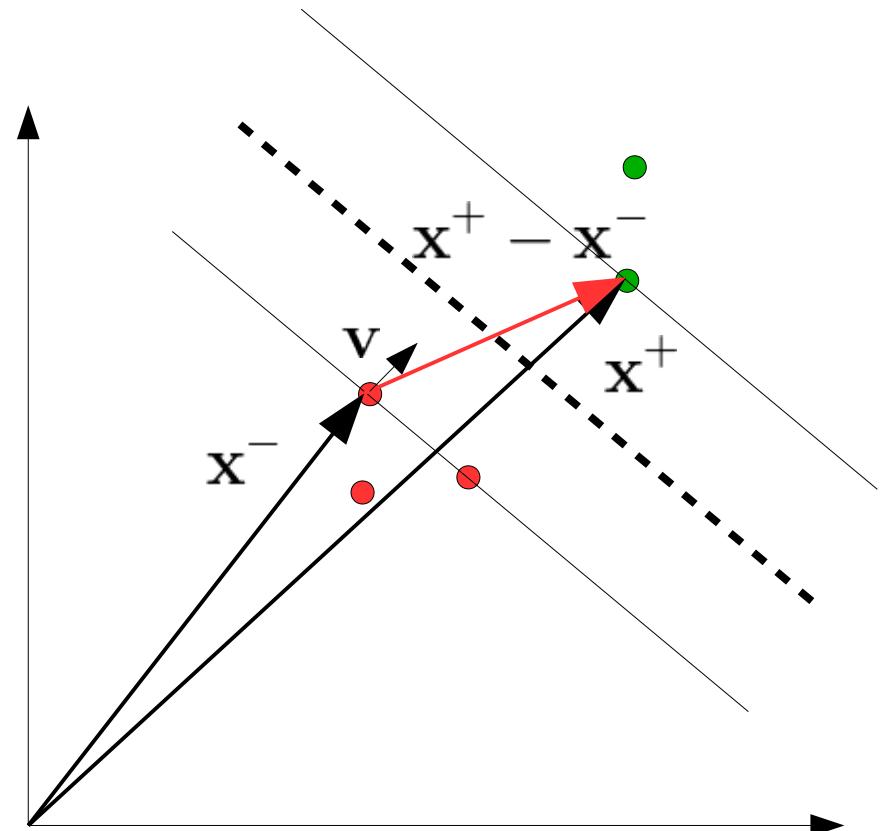
- Another **MUCH BETTER** approach considers Linear Algebra

Formulating this problem algebraically...

We have two support vectors plotted, one for each class

The **difference vector** projected over the **unitary vector  $v$**  orthogonal to both support vectors hyperplanes defines the “street” width

$$\text{Proj}_v(x^+ - x^-) = \langle x^+ - x^-, v \rangle$$



So, the optimization is basically interested in maximizing this projection!

# Hyperplane-based classifiers: an algebraic view

- Another **MUCH BETTER** approach considers Linear Algebra

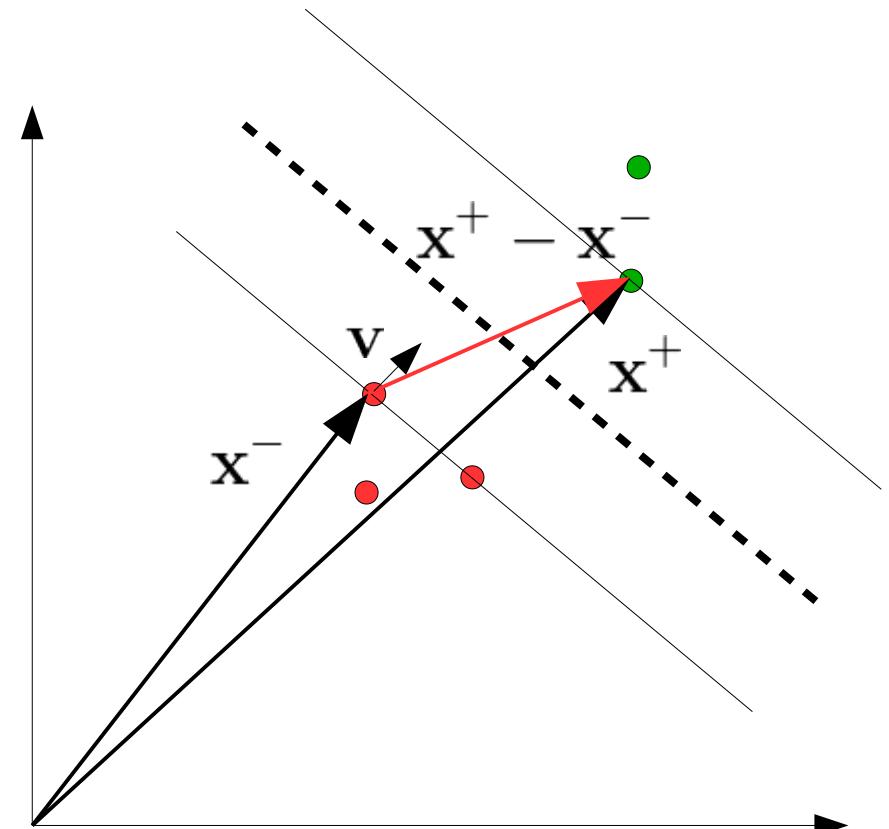
Formulating this problem algebraically...

We have two support vectors plotted, one for each class

The **difference vector** projected over the **unitary vector  $v$**  orthogonal to both support vectors hyperplanes defines the “street” width

$$\text{Proj}_v(x^+ - x^-) = \langle x^+ - x^-, v \rangle$$

But, we do not have  $v$ , however we can define it using a vector  $w$



$$\text{Proj}_v(x^+ - x^-) = \langle x^+ - x^-, \frac{w}{\|w\|} \rangle, \quad v = \frac{w}{\|w\|}$$

# Hyperplane-based classifiers: an algebraic view

- Another **MUCH BETTER** approach considers Linear Algebra

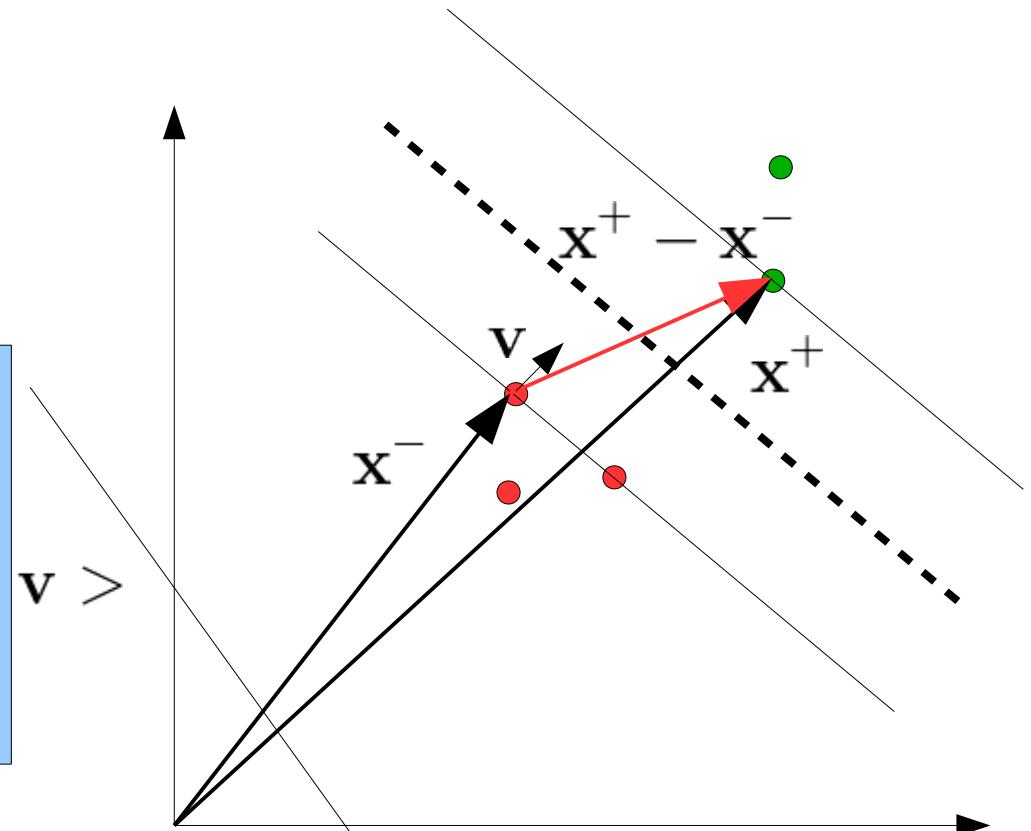
Formulating this problem algebraically...

We have two support vectors plotted, one for each class

The **difference vector** projected over the **unitary vector  $v$**  orthogonal to both support vectors

Projecting  $x^+ - x^-$  onto  $v$

But, we can define  $w$  using a vector  $v$



$$\text{Proj}_v(x^+ - x^-) = \langle x^+ - x^-, \frac{w}{\|w\|} \rangle, \quad v = \frac{w}{\|w\|}$$

# Hyperplane-based classifiers: an algebraic view

- Another **MUCH BETTER** approach considers Linear Algebra

Formulating this problem algebraically...

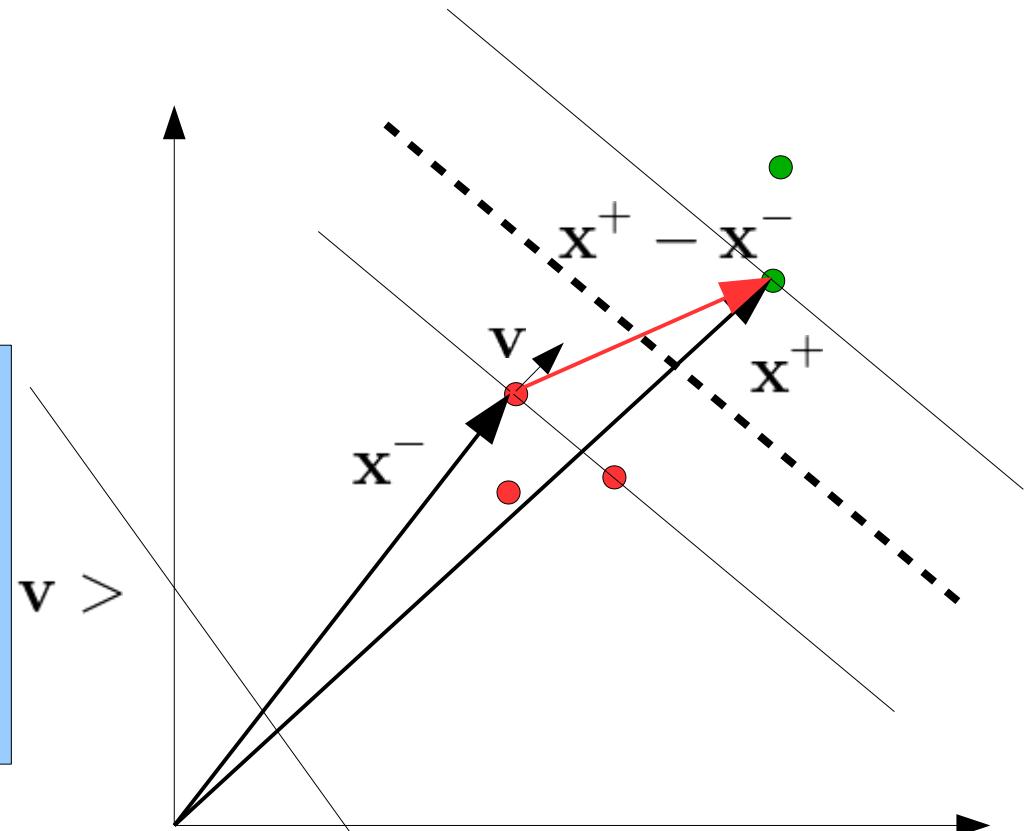
We have two support vectors plotted, one for each class

The **difference vector** projected over the **unitary vector  $v$**  orthogonal to both support vectors

But how would you define  $w$ ?

Projecting  $w$  onto  $v$

But, we can define  $w$  using a vector  $v$



$$\text{Proj}_v(x^+ - x^-) = \langle x^+ - x^-, \frac{w}{\|w\|} \rangle, \quad v = \frac{w}{\|w\|}$$

# Hyperplane-based classifiers: an algebraic view

- Another **MUCH BETTER** approach considers Linear Algebra
  - So, we have:

$$\begin{aligned}\text{Proj}_{\mathbf{v}}(\mathbf{x}^+ - \mathbf{x}^-) &= \langle \mathbf{x}^+ - \mathbf{x}^-, \frac{\mathbf{w}}{\|\mathbf{w}\|} \rangle, \quad \mathbf{v} = \frac{\mathbf{w}}{\|\mathbf{w}\|} \\ &= \langle \mathbf{x}^+, \frac{\mathbf{w}}{\|\mathbf{w}\|} \rangle - \langle \mathbf{x}^-, \frac{\mathbf{w}}{\|\mathbf{w}\|} \rangle\end{aligned}$$

# Hyperplane-based classifiers: an algebraic view

- Another **MUCH BETTER** approach considers Linear Algebra
  - So, we have:

$$\begin{aligned}\text{Proj}_{\mathbf{v}}(\mathbf{x}^+ - \mathbf{x}^-) &= \langle \mathbf{x}^+ - \mathbf{x}^-, \frac{\mathbf{w}}{\|\mathbf{w}\|} \rangle, \quad \mathbf{v} = \frac{\mathbf{w}}{\|\mathbf{w}\|} \\ &= \langle \mathbf{x}^+, \frac{\mathbf{w}}{\|\mathbf{w}\|} \rangle - \langle \mathbf{x}^-, \frac{\mathbf{w}}{\|\mathbf{w}\|} \rangle\end{aligned}$$

- Now, remember the closest positive point(s) must be classified as +1 while the closest negative point(s) as -1
  - So, remember the constraint for our problem:

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) - 1 = 0$$

# Hyperplane-based classifiers: an algebraic view

- Another **MUCH BETTER** approach considers Linear Algebra
  - So, we have:

$$\begin{aligned}\text{Proj}_{\mathbf{v}}(\mathbf{x}^+ - \mathbf{x}^-) &= \langle \mathbf{x}^+ - \mathbf{x}^-, \frac{\mathbf{w}}{\|\mathbf{w}\|} \rangle, \quad \mathbf{v} = \frac{\mathbf{w}}{\|\mathbf{w}\|} \\ &= \langle \mathbf{x}^+, \frac{\mathbf{w}}{\|\mathbf{w}\|} \rangle - \langle \mathbf{x}^-, \frac{\mathbf{w}}{\|\mathbf{w}\|} \rangle\end{aligned}$$

- Now, remember the closest positive point(s) must be classified as +1 while the closest negative point(s) as -1
  - So, remember the constraint for our problem:

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) - 1 = 0$$

Simplest formulation for the constraints  
(only for the closest points)

# Hyperplane-based classifiers: an algebraic view

- Another **MUCH BETTER** approach considers Linear Algebra
  - So, we have:

$$\begin{aligned}\text{Proj}_{\mathbf{v}}(\mathbf{x}^+ - \mathbf{x}^-) &= \langle \mathbf{x}^+ - \mathbf{x}^-, \frac{\mathbf{w}}{\|\mathbf{w}\|} \rangle, \quad \mathbf{v} = \frac{\mathbf{w}}{\|\mathbf{w}\|} \\ &= \langle \mathbf{x}^+, \frac{\mathbf{w}}{\|\mathbf{w}\|} \rangle - \langle \mathbf{x}^-, \frac{\mathbf{w}}{\|\mathbf{w}\|} \rangle\end{aligned}$$

- Now, remember the closest positive point(s) must be classified as +1 while the closest negative point(s) as -1
  - So, remember the constraint for our problem:

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) - 1 = 0$$

- Substituting for the positive and negative closest points:

$$\text{Proj}_{\mathbf{v}}(\mathbf{x}^+ - \mathbf{x}^-) = \frac{1 - b}{\|\mathbf{w}\|} - \frac{-1 - b}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|}$$

# Hyperplane-based classifiers: an algebraic view

- Another approach is by considering linear algebra
  - So, we have:

**Observe:**

$$+1(\langle \mathbf{w}, \mathbf{x}_+ \rangle + b) - 1 = 0$$
$$\langle \mathbf{w}, \mathbf{x}_+ \rangle = 1 - b$$

$$-1(\langle \mathbf{w}, \mathbf{x}_+ \rangle + b) - 1 = 0$$
$$-\langle \mathbf{w}, \mathbf{x}_+ \rangle = 1 + b$$
$$\langle \mathbf{w}, \mathbf{x}_+ \rangle = -1 - b$$

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) - 1 = 0$$

- Substituting for the positive and negative closest points:

$$\text{Proj}_{\mathbf{v}}(\mathbf{x}^+ - \mathbf{x}^-) = \frac{1 - b}{\|\mathbf{w}\|} - \frac{-1 - b}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|}$$

# Hyperplane-based classifiers: an algebraic view

- Remember the objective is to maximize the “street” width which is the same as:

$$\text{Maximize } \text{Proj}_{\mathbf{v}}(\mathbf{x}^+ - \mathbf{x}^-)$$

$$\text{Maximize } \frac{2}{\|\mathbf{w}\|}$$

but subject to the constraints:  $y_i(\langle \mathbf{x}_i, \mathbf{w} \rangle + b) \geq 1$

# Hyperplane-based classifiers: an algebraic view

- Instead of maximizing, we can also see this problem as a minimization problem in form:

$$\begin{aligned} & \text{Maximize } \frac{2}{\|\mathbf{w}\|} \\ & \text{Subject to: } y_i(\langle \mathbf{x}_i, \mathbf{w} \rangle + b) \geq 1 \end{aligned}$$

Mathematical convenience

$$\begin{aligned} & \text{Minimize } \frac{1}{2} \|\mathbf{w}\|^2 \\ & \text{Subject to: } y_i(\langle \mathbf{x}_i, \mathbf{w} \rangle + b) \geq 1 \end{aligned}$$

See more about this formulation at: [https://www.youtube.com/watch?v=\\_PwhiWxHK8o](https://www.youtube.com/watch?v=_PwhiWxHK8o)

**We already have the Classification problem formalized**

**Now, we need to solve it**

**That depends on remembering:  
Lagrange Multipliers and Karush-Kuhn-Tucker  
conditions**

# Lagrange multipliers

- Lagrange Multipliers
  - Named after Joseph Louis Lagrange
  - Strategy to find local maxima and minima subject to equality constraints

Solve problems in form:

maximize  $f(x, y)$   
subject to  $g(x, y) = c$ .

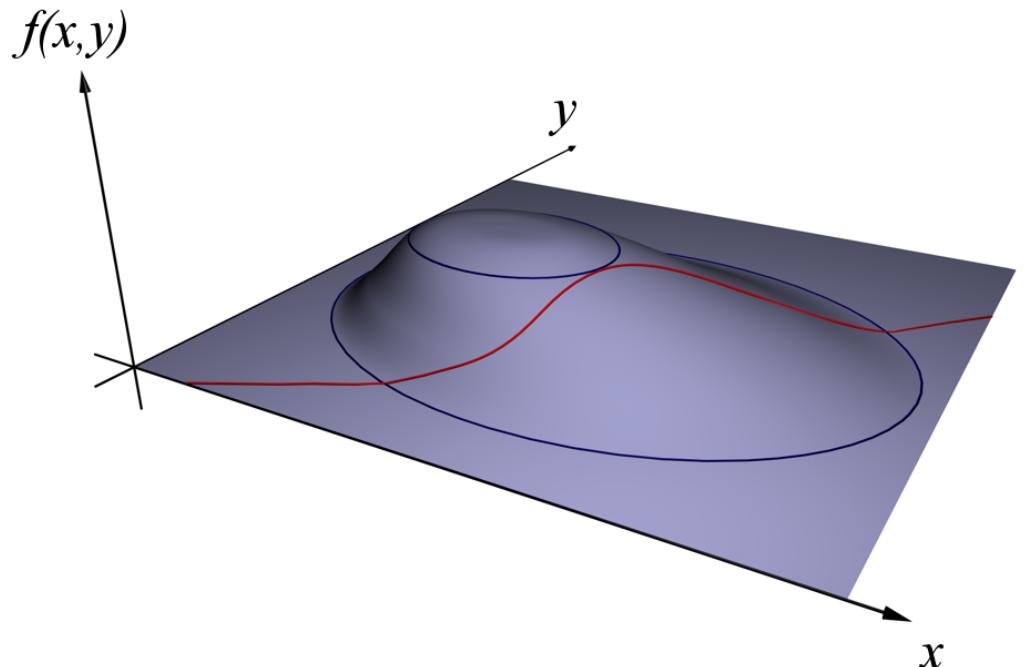


Figure obtained at [http://en.wikipedia.org/wiki/Lagrange\\_multiplier](http://en.wikipedia.org/wiki/Lagrange_multiplier)

# Lagrange multipliers

- Lagrange Multipliers
  - The great discovery was:  $\nabla f = -\lambda \nabla g$
  - At some point  $x,y$  for the **objective function**  $f$  and for the **constraint function**  $g$

Gradient vectors (GVs) are therefore parallel at some point

Lambda is the Lagrange multiplier, which is necessary due to besides GVs are parallel they can pull to different sides and have different magnitudes!

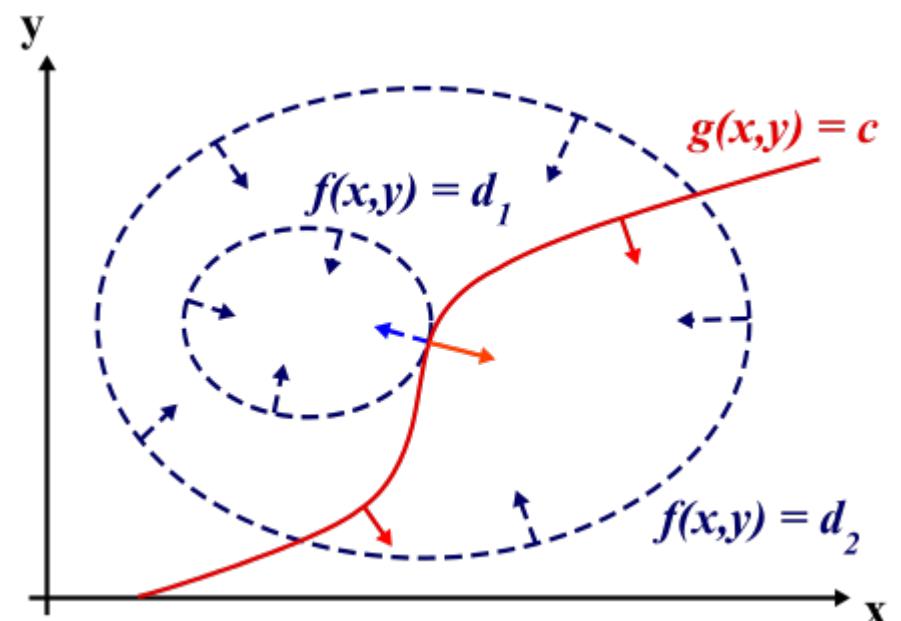


Figure obtained at [http://en.wikipedia.org/wiki/Lagrange\\_multiplier](http://en.wikipedia.org/wiki/Lagrange_multiplier)

# Lagrange multipliers

- Lagrange Multipliers
  - Intuitive notion
    - We will use the demonstration code for Mathematica available at:
      - <http://demonstrations.wolfram.com/ConstrainedOptimization/>
    - Video is lagrange.ogv

# Lagrange multipliers

- Lagrange Multipliers
  - Now suppose a problem (if the conditions below are satisfied we can apply Lagrange Multipliers):
  - Are  $f$  and  $g$  differentiable?
    - Yes
  - Does it consider one or more equality constraints?
    - Yes
  - The gradient of  $g$  is different of zero vector?
    - Yes

$$\begin{aligned} & \text{maximize } f(x, y) = x + y \\ & \text{subject to } x^2 + y^2 = 1 \end{aligned}$$

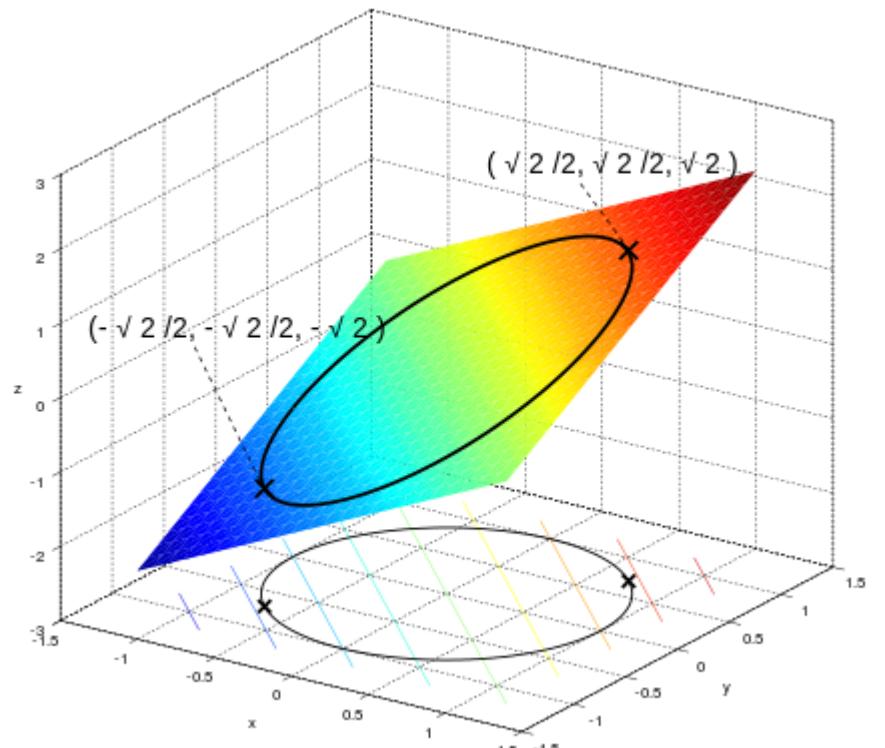


Figure obtained at [http://en.wikipedia.org/wiki/Lagrange\\_multiplier](http://en.wikipedia.org/wiki/Lagrange_multiplier)

# Lagrange multipliers

- Lagrange Multipliers
  - Now suppose a problem:

maximize  $f(x, y) = x + y$

subject to  $x^2 + y^2 = 1$

- So we can solve the system of equations:  $\begin{cases} \nabla f = -\lambda \nabla g \\ g(x, y) = 0 \end{cases}$
- And find:

• or: 
$$\begin{cases} (1, 1) = -\lambda (2x, 2y) \\ x^2 + y^2 - 1 = 0 \end{cases}$$

$$\begin{cases} 1 = -\lambda 2x \\ 1 = -\lambda 2y \\ x^2 + y^2 - 1 = 0 \end{cases}$$

# Lagrange multipliers

- Lagrange Multipliers

- Solving this system of equations:

$$\begin{cases} 1 = -\lambda 2x \\ 1 = -\lambda 2y \\ x^2 + y^2 - 1 = 0 \end{cases}$$

- We find:

$$\begin{cases} \lambda 2x + 1 = 0 \Rightarrow x = -\frac{1}{2\lambda} \\ \lambda 2y + 1 = 0 \Rightarrow y = -\frac{1}{2\lambda} \\ x^2 + y^2 - 1 = 0 \Rightarrow \left(-\frac{1}{2\lambda}\right)^2 + \left(-\frac{1}{2\lambda}\right)^2 - 1 = 0 \text{ for } \lambda \neq 0 \end{cases}$$

- And solving for Lambda:

$$\lambda = \mp 1/\sqrt{2}$$

# Lagrange multipliers

- Lagrange Multipliers
  - Now we can find x and y by plugging:

$$\lambda = \mp 1/\sqrt{2}$$

- Then we have:

$$\text{for } \lambda = -\frac{1}{\sqrt{2}}$$

$$x = y = -\frac{1}{2\lambda} = \left(-\frac{1}{2}\right) \cdot \left(-\frac{\sqrt{2}}{1}\right) = \frac{\sqrt{2}}{2}$$

$$\text{and for } \lambda = \frac{1}{\sqrt{2}}$$

$$x = y = -\frac{1}{2\lambda} = -\frac{1}{2} \frac{\sqrt{2}}{1} = -\frac{\sqrt{2}}{2}$$

# Lagrange multipliers

- Lagrange Multipliers
  - As a consequence we found two points of interest:
$$(\sqrt{2}/2, \sqrt{2}/2) \text{ and } (-\sqrt{2}/2, -\sqrt{2}/2)$$
  - As the last step we plug these x,y values into f and find maxima and/or minima points:

$$f(\sqrt{2}/2, \sqrt{2}/2) = \sqrt{2} \text{ and } f(-\sqrt{2}/2, -\sqrt{2}/2) = -\sqrt{2}$$

**Global Maximum  
subject to the  
constraint!**

**Global Minimum  
subject to the  
constraint!**

# Lagrange multipliers

- Lagrange Multipliers
  - Plotting the problem and solutions

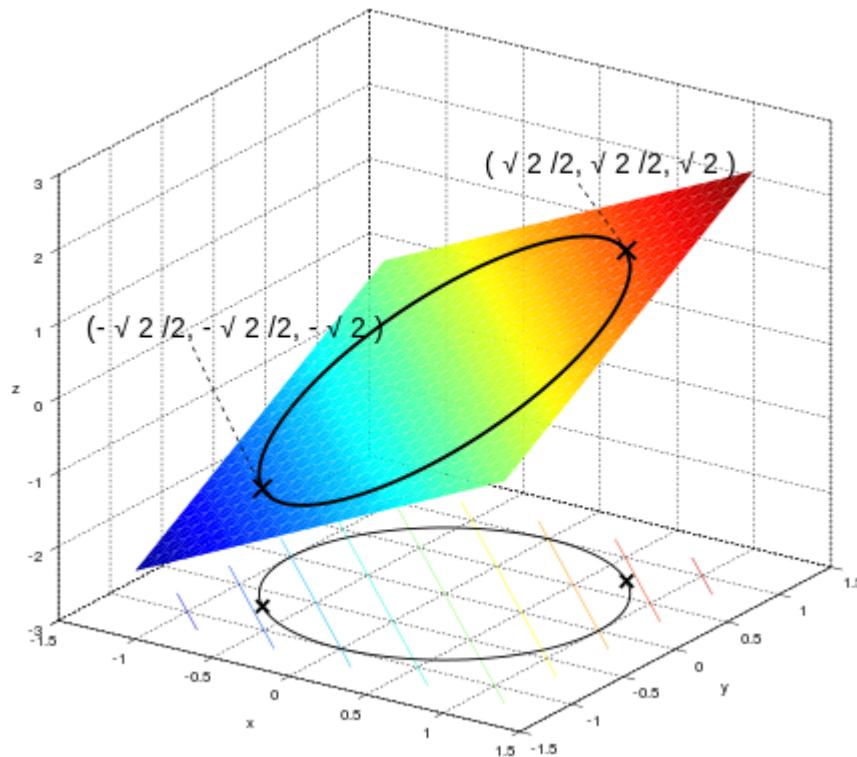


Figure obtained at [http://en.wikipedia.org/wiki/Lagrange\\_multiplier](http://en.wikipedia.org/wiki/Lagrange_multiplier)

# Lagrange multipliers

- Lagrange Multipliers

- In this **second example** we will build the solution in the traditional way
- Consider the problem:

$$\begin{aligned} & \text{maximize } f(x, y) = x^2y \\ & \text{subject to: } g(x, y) = x^2 + y^2 = 3. \end{aligned}$$

- The traditional formulation considers the **Lagrange function** or the **Lagrangian** and its **partial derivatives** with respect to x, y and lambda:

$$\Lambda(x, y, \lambda) = f(x, y) + \lambda(g(x, y) - 3) = x^2y + \lambda(x^2 + y^2 - 3).$$

- Which is the same as solving the system:  $\begin{cases} \nabla f = -\lambda \nabla g \\ g = 0 \end{cases}$

# Lagrange multipliers

- Lagrange Multipliers
  - Looking at the second problem and its solutions:

$$\Lambda(x, y, \lambda) = f(x, y) + \lambda(g(x, y) - 3) = x^2y + \lambda(x^2 + y^2 - 3).$$

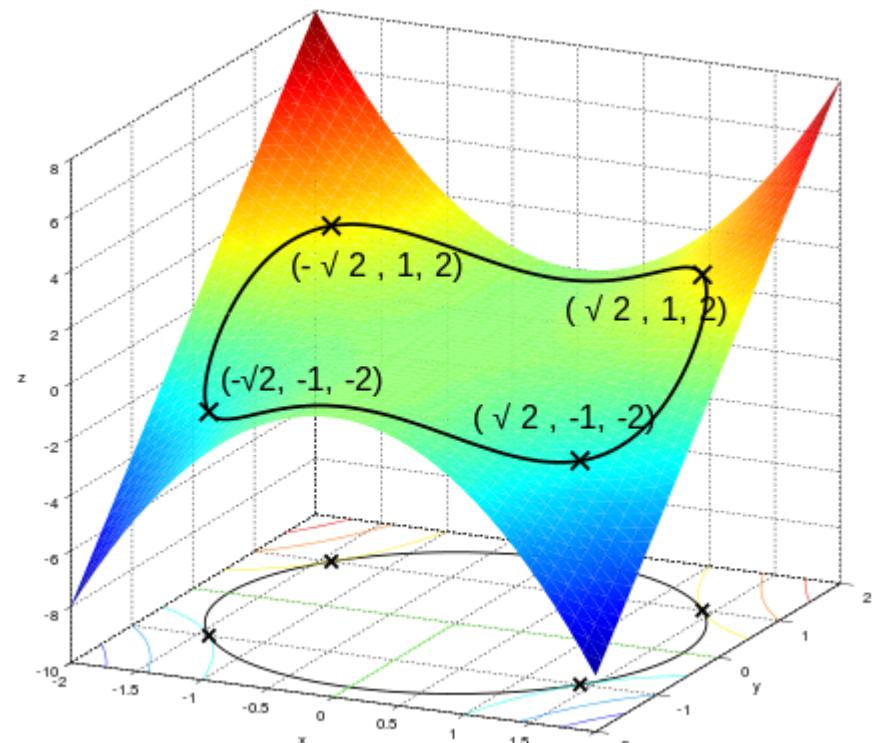


Figure obtained at [http://en.wikipedia.org/wiki/Lagrange\\_multiplier](http://en.wikipedia.org/wiki/Lagrange_multiplier)

# Lagrange multipliers

- Lagrange Multipliers
  - Solving the Lagrangian function:

$$\Lambda(x, y, \lambda) = f(x, y) + \lambda(g(x, y) - 3) = x^2y + \lambda(x^2 + y^2 - 3).$$

- We have the following system of equations:

$$\begin{cases} \frac{\partial \Lambda(x,y,\lambda)}{\partial x} = 2xy + 2x\lambda = 0 \\ \frac{\partial \Lambda(x,y,\lambda)}{\partial y} = x^2 + 2y\lambda = 0 \\ \frac{\partial \Lambda(x,y,\lambda)}{\partial \lambda} = x^2 + y^2 - 3 = 0 \end{cases}$$

- Equation (1) implies:  $x = 0$  or  $\lambda = -y$

# Lagrange multipliers

- Lagrange Multipliers

- Then if  $x=0$

$y = \pm\sqrt{3}$  by applying Equation (3)  
and  $\lambda = 0$  by Equation(2)

- Then if  $\lambda = -y$

And substituting into Equation (2) we have  $x^2 - 2y^2 = 0$ . Then  $x^2 = 2y^2$ .  
Substituting into Equation (3) and solving for  $y$  we have  $y = \pm 1$

- So we have six critical points to analyze:

$$(\sqrt{2}, 1); \quad (-\sqrt{2}, 1); \quad (\sqrt{2}, -1); \quad (-\sqrt{2}, -1); \quad (0, \sqrt{3}); \quad (0, -\sqrt{3}).$$

# Lagrange multipliers

- Lagrange Multipliers
  - So the next step is to evaluate  $f$  for those six points:

$$f(\pm\sqrt{2}, 1) = 2; \quad f(\pm\sqrt{2}, -1) = -2; \quad f(0, \pm\sqrt{3}) = 0.$$

**Global  
Maximum  
subject to the  
constraint!**

**Global  
Minimum  
subject to the  
constraint!**

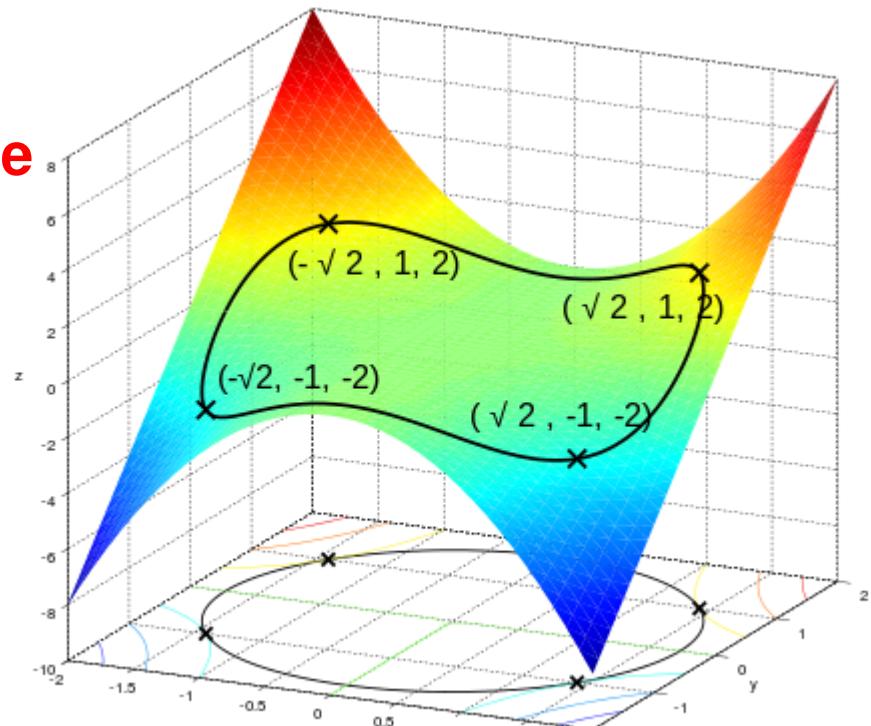


Figure obtained at [http://en.wikipedia.org/wiki/Lagrange\\_multiplier](http://en.wikipedia.org/wiki/Lagrange_multiplier)

# Lagrange multipliers

- Exercises using Lagrange Multipliers:
  - Problem 1 – Solve the following problem using  $\nabla f = -\lambda \nabla g$ 
$$\begin{aligned} & \text{maximize } x^2y \\ & \text{subject to } x + y = 5 \end{aligned}$$
  - Problem 2 – Solve the same problem using the Lagrange function

$$\begin{aligned} & \text{maximize } x^2y \\ & \text{subject to } x + y = 5 \end{aligned}$$

# Lagrange multipliers

- Exercises using Lagrange Multipliers:

- Problem 3

$$\begin{aligned} & \text{maximize } f(x, y) = x^2 - y^2 \\ & \text{subject to } g(x, y) = y - x^2 = 0. \end{aligned}$$

- Problem 4

Let the price of a product be given by  $f(x, y) = x + 2y$ .

Find the maximum and minimum price given  $x^2 + y^2 = 1000$ .

Watch this video: <https://www.youtube.com/watch?v=KVwgFvmHnBM>

**But Lagrange Multipliers are not enough  
because we have inequality constraints!**

# Karush-Kuhn-Tucker conditions

- After learning/remembering a bit about Lagrange Multipliers
  - Let's see something about the KKT conditions
    - KKT defines the **necessary conditions** for a solution in linear/nonlinear programming to be optimal
    - It is more general than Lagrange Multipliers as it allows **inequality constraints**
    - It is **usually not solved directly**, consequently there are several **optimization algorithms to numerically** solve the KKT system of equations

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions
  - Used to solve problems in the two **standard forms**:

Maximize  $f(x)$   
subject to  $g_i(x) \geq 0, h_j(x) = 0$

**OR**

Minimize  $f(x)$   
subject to  $g_i(x) \geq 0, h_j(x) = 0$

- Where:
  - f is the objective function
  - And we have **inequality and equality constraints**

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions:
  - There are four KKT Conditions, we will see them while solving a problem:
    - **First condition (Primal Feasibility):** it assumes there is an optimal solution  $x^*$  for which all constraints are satisfied:

$$\begin{aligned}g_i(x^*) &\geq 0 \\h_j(x^*) &= 0\end{aligned}$$

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions:
  - There are four KKT Conditions, we will see them while solving a problem:
    - **Second condition (Stationarity)**: there is at least one stationary point, basically there is at least a point in which the gradient of  $f$  minus constraints are parallel
      - No direction improves objective and it is feasible

$$\nabla f(x^*) - \sum_{i=1}^m \mu_i^* \nabla g_i(x^*) - \sum_{j=1}^l \lambda_j^* \nabla h_j(x^*) = 0$$

**KKT Multipliers**

**Stars mean optimal values!**

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions:
  - There are four KKT Conditions, we will see them while solving a problem:
    - **Third condition (Complementary slackness):** KKT multipliers times inequality constraints are equal to zero

$$\mu_i g_i(x^*) = 0, \text{ for all } i = 1, \dots, m.$$

– **Only for inequality constraints!**

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions:
  - There are four KKT Conditions, we will see them while solving a problem:
    - **Fourth condition (Dual Feasibility):** All KKT multipliers must be positive for inequality constraints

$$\mu_i \geq 0, \text{ for all } i = 1, \dots, m$$

– **Only for inequality constraints!**

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions:

- Let's solve a problem:

$$\text{Minimize } f(x_1, x_2) = 4x_1^2 + 2x_2^2$$

$$\text{s.t. } 3x_1 + x_2 = 8$$

$$2x_1 + 4x_2 \leq 15$$

- Consider someone tells you the second constraint is **non-binding**

- A binding constraint is the one that the solution is dependent on. So, if you change it, the optimal solution will have to change.
    - The non-binding constraint does not affect the optimal solution and can be discarded without changing it.

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions:
  - Let's solve a problem:

$$\begin{aligned} \text{Minimize } & f(x_1, x_2) = 4x_1^2 + 2x_2^2 \\ \text{s.t. } & 3x_1 + x_2 = 8 \\ & \underline{2x_1 + 4x_2 \leq 15} \end{aligned}$$

We drop the second constraint because it is non-binding so  $\mu=0$

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions:
  - Let's solve a problem:

$$\text{Minimize } f(x_1, x_2) = 4x_1^2 + 2x_2^2$$

$$\text{s.t. } 3x_1 + x_2 = 8$$

$$\underline{-2x_1 + 4x_2 \leq 15}$$

We drop the second constraint  
because it is non-bidding so  $\mu=0$

- Anyway we should write this problem in the standard form:

$$\text{Minimize } f(x_1, x_2) = 4x_1^2 + 2x_2^2$$

$$\text{s.t. } 3x_1 + x_2 = 8$$

$$-2x_1 - 4x_2 \geq -15$$

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions:

- Let's solve this problem:

$$\begin{aligned} \text{Minimize } & f(x_1, x_2) = 4x_1^2 + 2x_2^2 \\ \text{s.t. } & 3x_1 + x_2 = 8 \\ & -2x_1 - 4x_2 \geq -15 \end{aligned}$$

- So, we have the KKT conditions:

- (1) all constraints have to be satisfied:

$$3x_1 + x_2 = 8$$

- The inequality constraint is non-binding, so we do not need to worry about it!

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions:

- Let's solve this problem:

$$\begin{aligned} \text{Minimize } f(x_1, x_2) &= 4x_1^2 + 2x_2^2 \\ \text{s.t. } 3x_1 + x_2 &= 8 \\ -2x_1 - 4x_2 &\geq -15 \end{aligned}$$

- So, we have the KKT conditions:
    - (2) stationarity:

$$\nabla f(x^*) - \sum_{i=1}^m \mu_i^* \nabla g_i(x^*) - \sum_{j=1}^l \lambda_j^* \nabla h_j(x^*) = 0$$
$$\begin{bmatrix} 8x_1 \\ 4x_2 \end{bmatrix} - \lambda_1 \begin{bmatrix} 3 \\ 1 \end{bmatrix} - \mu_1 \begin{bmatrix} -2 \\ -4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad \text{as } \mu_1 = 0, \text{ we have:}$$
$$\begin{bmatrix} 8x_1 \\ 4x_2 \end{bmatrix} - \lambda_1 \begin{bmatrix} 3 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions:
  - Let's solve a problem:
    - (2) stationarity: We solve it using the matrix form:

$$AX = B$$

$$\begin{bmatrix} 3 & 1 & 0 \\ 8 & 0 & -3 \\ 0 & 4 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \lambda_1 \end{bmatrix} = \begin{bmatrix} 8 \\ 0 \\ 0 \end{bmatrix}$$

- Then we solve it to obtain:

$$X = A^{-1}B \qquad X = \begin{bmatrix} 2.182 \\ 1.455 \\ 5.818 \end{bmatrix}$$

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions
  - Further conditions are only necessary for inequalities, what is not the case here
    - (3) Complementary slackness
    - (4) Dual feasibility
  - Consequently we found a feasible solution for this problem considering the KKT conditions

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions
  - Solve the problem:

$$\text{Minimize } f(x_1, x_2, x_3) = x_1^2 + 2x_2^2 + 3x_3^2$$

$$\text{Subject to } g_1(x_1, x_2, x_3) = -5x_1 + x_2 + 3x_3 \leq -3$$

$$g_2(x_1, x_2, x_3) = 2x_1 + x_2 + 2x_3 \geq 6$$

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions
  - Solve the problem:

$$\text{Minimize } f(x_1, x_2, x_3) = x_1^2 + 2x_2^2 + 3x_3^2$$

$$\text{Subject to } g_1(x_1, x_2, x_3) = -5x_1 + x_2 + 3x_3 \leq -3$$

$$g_2(x_1, x_2, x_3) = 2x_1 + x_2 + 2x_3 \geq 6$$

- First of all, write it in the **standard form**:

$$\text{Minimize } f(x_1, x_2, x_3) = x_1^2 + 2x_2^2 + 3x_3^2$$

$$\text{Subject to } g_1(x_1, x_2, x_3) = 5x_1 - x_2 - 3x_3 \geq 3$$

$$g_2(x_1, x_2, x_3) = 2x_1 + x_2 + 2x_3 \geq 6$$

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions
  - Now solve the problem:

$$\text{Minimize } f(x_1, x_2, x_3) = x_1^2 + 2x_2^2 + 3x_3^2$$

$$\begin{aligned}\text{Subject to } g_1(x_1, x_2, x_3) &= 5x_1 - x_2 - 3x_3 \geq 3 \\ g_2(x_1, x_2, x_3) &= 2x_1 + x_2 + 2x_3 \geq 6\end{aligned}$$

- As condition (1) we must remember all constraints must be satisfied → let's see what happens in this case
- As condition (2) we write the Lagrangian:

$$\nabla f(x^*) - \sum_{i=1}^m \mu^* g_i(x^*) - \sum_{j=1}^l \lambda^* h_j(x^*) = 0$$

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions
  - Now we solve the Lagrangian:

$$\left\{ \begin{array}{l} \frac{\partial f}{\partial x_1} - \mu_1 \frac{\partial g_1}{\partial x_1} - \mu_2 \frac{\partial g_2}{\partial x_1} = 0 \\ \frac{\partial f}{\partial x_2} - \mu_1 \frac{\partial g_1}{\partial x_2} - \mu_2 \frac{\partial g_2}{\partial x_2} = 0 \\ \frac{\partial f}{\partial x_3} - \mu_1 \frac{\partial g_1}{\partial x_3} - \mu_2 \frac{\partial g_2}{\partial x_3} = 0 \\ g_1 = 3 \\ g_2 = 6 \end{array} \right.$$

- Then we have:

$$\left\{ \begin{array}{l} 2x_1 - \mu_1(5) - \mu_2(2) = 0 \\ 4x_2 - \mu_1(-1) - \mu_2(1) = 0 \\ 6x_3 - \mu_1(-3) - \mu_2(2) = 0 \\ 5x_1 - x_2 - 3x_3 = 3 \\ 2x_1 + x_2 + 2x_3 = 6 \end{array} \right.$$

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions
  - Then we solve the linear system:

$$AX = B$$

- Finding:

$$x_1 = 1.450, \quad x_2 = 0.8, \quad x_3 = 1.150, \quad \mu_1 = -0.5, \quad \mu_2 = 2.70$$

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions
  - Then we solve the linear system:

$$AX = B$$

- Finding:

$$x_1 = 1.450, \quad x_2 = 0.8, \quad x_3 = 1.150, \quad \mu_1 = -0.5, \quad \mu_2 = 2.70$$

Now we see the Fourth KKT condition is violated!

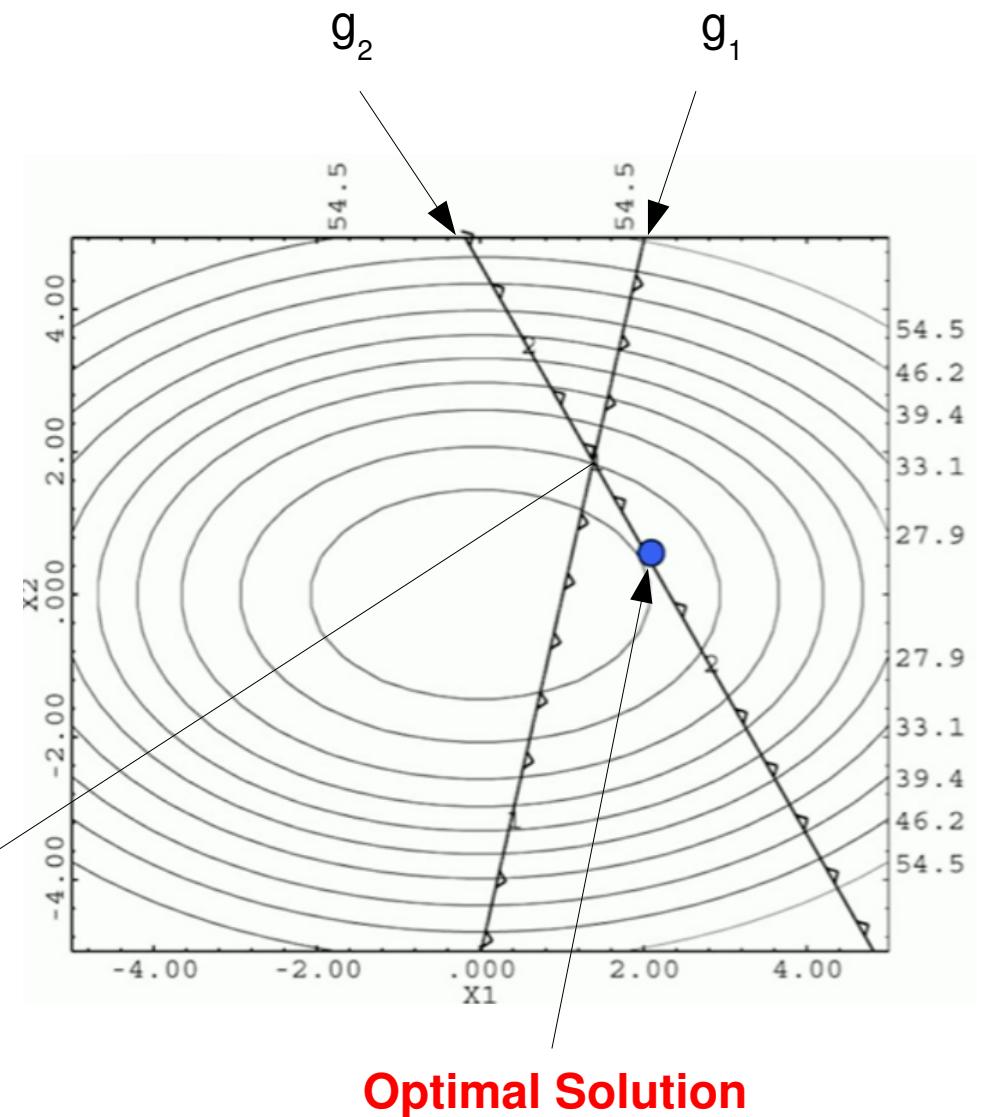
This means this is not an optimal solution!!!

What can we do???

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions
  - This happens because constraint  $g_1$  is not active at the optimal solution
  - So we can make  $\mu_1=0$ , dropping this non-binding constraint and solving the problem again

If optimal solution were here, both constraints would be active and therefore binding



# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions
  - Now solve this problem for  $\mu_1=0$  as homework

# Karush-Kuhn-Tucker conditions

- Curiosities:
  - The KKT conditions were originally named after Harold W. Kuhn and Albert W. Tucker
    - They were the first to publish the conditions in 1951
  - However, later other researchers discovered that those necessary conditions had already been stated by William Karush in his master's thesis in 1939

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions
  - As homework solve the **problem**:

$$\text{Minimize } f(x_1, x_2) = x_1^2 + x_2$$

$$\text{Subject to } g_1(x_1, x_2) = x_1^2 + x_2^2 - 9 \leq 0$$

$$g_2(x_1, x_2) = x_1 + x_2 - 1 \leq 0$$

- Do not forget to write it in the **standard form** as follows:

$$\text{Minimize } f(x_1, x_2) = x_1^2 + x_2$$

$$\text{Subject to } g_1(x_1, x_2) = -x_1^2 - x_2^2 + 9 \geq 0$$

$$g_2(x_1, x_2) = -x_1 - x_2 + 1 \geq 0$$

# Karush-Kuhn-Tucker conditions

- Karush-Kuhn-Tucker conditions
  - I suggest some videos I used to prepare this material:
    - <https://www.youtube.com/watch?v=eaKPzb11qFw&list=WL&index=3>
    - <https://www.youtube.com/watch?v=JTTiELgMyuM&list=WL&index=2>
    - <https://www.youtube.com/watch?v=AQWy73cHoIU&index=4&list=WL>
  - They are part of the course Optimization Techniques in Engineering available at:
    - <http://apmonitor.com/me575/index.php/Main/KuhnTucker>
      - By John D. Hedengren and Alan R. Parkinson

# **Getting back to find the optimal hyperplane**

# Getting back to find the optimal hyperplane

- So what is the best solution?
  - It is the one that minimizes  $\mathbf{w}$  and  $b$  subject to the inequality constraints
  - This is called the optimal hyperplane:

$$\min_{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}} \tau(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|^2$$

subject to  $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$  for all  $i = 1, \dots, m.$

- But how to solve this optimization problem?
  - Lagrange Multipliers and KKT Conditions! **BUT WHY???**

# Getting back to find the optimal hyperplane

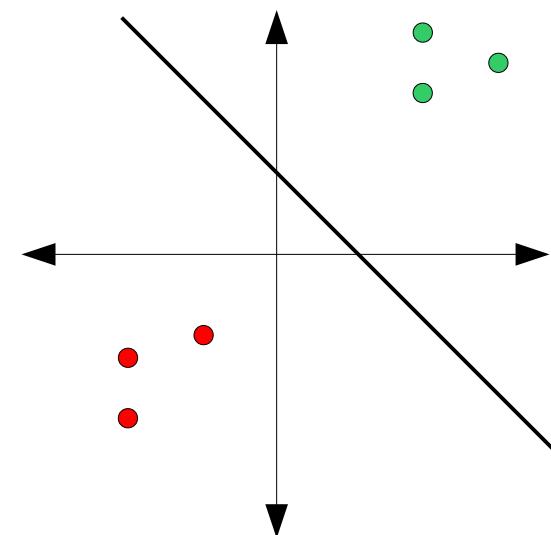
- So we need to guarantee the KKT Conditions:
  - (1) all constraints have to be satisfied
    - Something we assume
  - (2) stationarity

$$\Lambda(\mathbf{w}, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^m \alpha_i (y_i (\langle \mathbf{x}_i, \mathbf{w} \rangle + b) - 1)$$

# Getting back to find the optimal hyperplane

- So we need to guarantee the KKT Conditions:
  - (3) Complementary slackness: KKT multipliers times inequality constraints are equal to zero
    - Either because the multiplier make the constraint equals zero or the constraint is non-binding
      - This indeed happens when attempting to find the optimal hyperplane for linearly separable data!

The points closest to the hyperplane will have alpha different from zero (they are called the **support vectors**), others will have alpha = 0



# Getting back to find the optimal hyperplane

- So we need to guarantee the KKT Conditions:
  - (4) Dual feasibility
    - All KKT multipliers must be positive for inequality constraints, this is

$$\alpha_i \geq 0, \text{ for all } i = 1, \dots, m$$

- If all KKT Conditions are satisfied after we solve the problem, then we have an optimal solution

# Getting back to find the optimal hyperplane

- To solve this optimization problem we will solve for the Lagrangian:

$$\Lambda(\mathbf{w}, b) = \frac{1}{2} \|\mathbf{w}\|^2 - C \sum_{i=1}^m \max(1 - \mathbf{w}^\top \mathbf{x}_i - b, 0)$$

- With the following constraints:

Now we will see Lagrange multipliers are also useful to find dual forms of optimization problems

$$\frac{\partial \Lambda}{\partial b} = - \sum_{i=1}^m \alpha_i y_i = 0$$

# Getting back to find the optimal hyperplane

- To solve this optimization problem we will solve for the Lagrangian:

$$\Lambda(\mathbf{w}, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^m \alpha_i (y_i (\langle \mathbf{x}_i, \mathbf{w} \rangle + b) - 1)$$

- What requires the derivation in the direction of variables, leading us to:

$$\frac{\partial \Lambda}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i = 0$$

$$\text{Thus we have: } \mathbf{w} = \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i$$

$$\frac{\partial \Lambda}{\partial b} = - \sum_{i=1}^m \alpha_i y_i = 0$$

# Getting back to find the optimal hyperplane

- Opening the Lagrangian terms:

$$\Lambda(\mathbf{w}, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^m \alpha_i y_i \langle \mathbf{x}_i, \mathbf{w} \rangle - b \sum_{i=1}^m \alpha_i y_i + \sum_{i=1}^m \alpha_i$$

- Now, we substitute the derivations we found in order to reach minimum for  $\mathbf{w}$  and  $b$

- (1) Substituting:

$$-\sum_{i=1}^m \alpha_i y_i = 0$$

- We have:

$$\Lambda(\mathbf{w}, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^m \alpha_i y_i \langle \mathbf{x}_i, \mathbf{w} \rangle + \sum_{i=1}^m \alpha_i$$

# Getting back to find the optimal hyperplane

- Opening the Lagrangian terms:

$$\Lambda(\mathbf{w}, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^m \alpha_i y_i \langle \mathbf{x}_i, \mathbf{w} \rangle + \sum_{i=1}^m \alpha_i$$

- Now, we substitute the derivations we found in order to reach minimum for  $\mathbf{w}$  and  $b$

- (2) Substituting:

$$\mathbf{w} = \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i$$

- We have:

$$\begin{aligned} \Lambda(\mathbf{w}, b, \alpha) &= \frac{1}{2} \left\langle \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i, \sum_{j=1}^m \alpha_j y_j \mathbf{x}_j \right\rangle - \sum_{i=1}^m \alpha_i y_i \left\langle \mathbf{x}_i, \sum_{j=1}^m \alpha_j y_j \mathbf{x}_j \right\rangle \\ &\quad + \sum_{i=1}^m \alpha_i \end{aligned}$$

# Getting back to find the optimal hyperplane

- We now simplify the Lagrangian:

$$\begin{aligned}\Lambda(\mathbf{w}, b, \alpha) = & \frac{1}{2} \left\langle \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i, \sum_{j=1}^m \alpha_j y_j \mathbf{x}_j \right\rangle - \sum_{i=1}^m \alpha_i y_i \left\langle \mathbf{x}_i, \sum_{j=1}^m \alpha_j y_j \mathbf{x}_j \right\rangle \\ & + \sum_{i=1}^m \alpha_i\end{aligned}$$

- To find:

$$\begin{aligned}\Lambda(\mathbf{w}, b, \alpha) = & \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle - \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle \\ & + \sum_{i=1}^m \alpha_i\end{aligned}$$

# Getting back to find the optimal hyperplane

- Finally from:

$$\begin{aligned}\Lambda(\mathbf{w}, b, \alpha) &= \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle - \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \\ &\quad + \sum_{i=1}^m \alpha_i\end{aligned}$$

- We find:

$$\Lambda(\mathbf{w}, b, \alpha) = -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^m \alpha_i$$

# Getting back to find the optimal hyperplane

- So now we have the Lagrangian to minimize  $\mathbf{w}$  and  $b$  but represented in terms of multipliers alpha!!!
  - Observe that we simplified the Lagrangian
  - Solving this problem we find the optimal hyperplane

$$\Lambda(\mathbf{w}, b, \alpha) = -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^m \alpha_i$$

- However, we cannot forget this formulation must respect the KKT Conditions, so we have a solution only if:

$$\alpha_i \geq 0, \text{ for all } i = 1, \dots, m$$

# Getting back to find the optimal hyperplane

- We now rewrite the formulation:

$$\Lambda(\mathbf{w}, b, \alpha) = -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^m \alpha_i$$

- In terms of alpha, because this is the variable to work with:
  - As the function name was not representative anymore

$$W(\alpha) = -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^m \alpha_i$$

# Getting back to find the optimal hyperplane

- After all this problem reformulation, we leave the minimization and go for the maximization of:

$$W(\alpha) = -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^m \alpha_i$$
$$\alpha_i \geq 0, \text{ for all } i = 1, \dots, m$$

- With respect to  $\alpha$ 
  - i.e., by adapting alpha

# Getting back to find the optimal hyperplane

- After all this problem reformulation, we leave the minimization and go for the maximization of:

$$W(\alpha) = \frac{1}{2} \sum_{i=1}^m \alpha_i - \sum_{i=1}^m \alpha_i$$

- With now:
    - i.e.
- In fact we rewrote the same Primal problem in its Dual form!

# Getting back to find the optimal hyperplane

- By observing the **Second KKT Condition** (stationarity) the following must hold:

$$-\sum_{i=1}^m \alpha_i y_i = 0$$

- So, we conclude we have another constraint which is:

Signal does not matter  
(multiply both sides by -1)

You typically find this term  
without the minus sign

$$-\sum_{i=1}^m \alpha_i y_i = 0$$

# Getting back to find the optimal hyperplane

- Finally, putting all pieces together we have the problem in its **dual form**:

$$\underset{\alpha \in \mathbb{R}^m}{\text{Maximize}} \quad W(\alpha) = -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^m \alpha_i$$

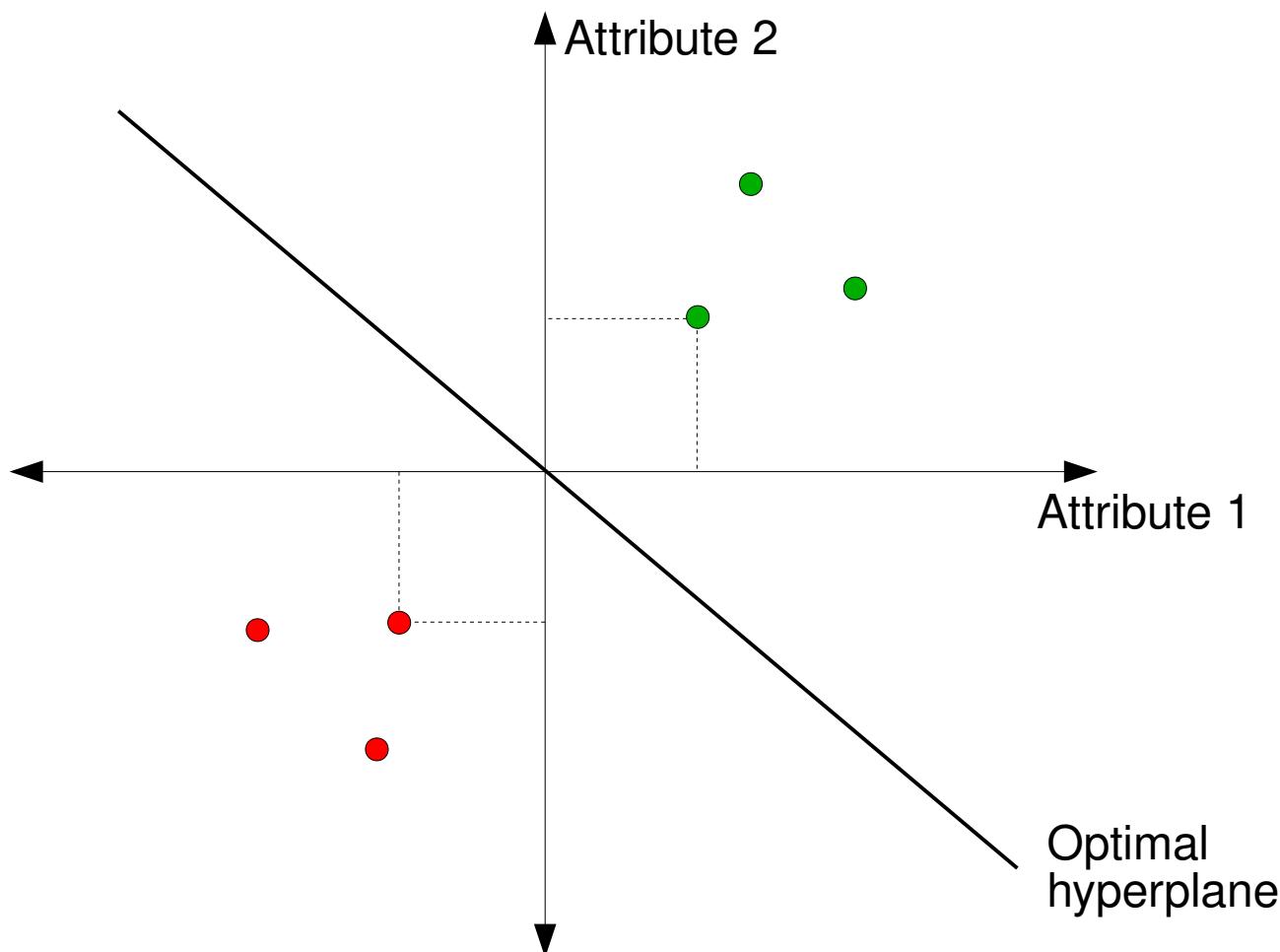
Subject to:  $\alpha_i \geq 0$ , for  $i = 1, \dots, m$ , and  $\sum_{i=1}^m \alpha_i y_i = 0$

- This is the problem people solve in practice!
- Well, what is the difference from the **primal** and the **dual** forms?

# Getting back to find the optimal hyperplane

- Remember every point lying on the hyperplane holds:

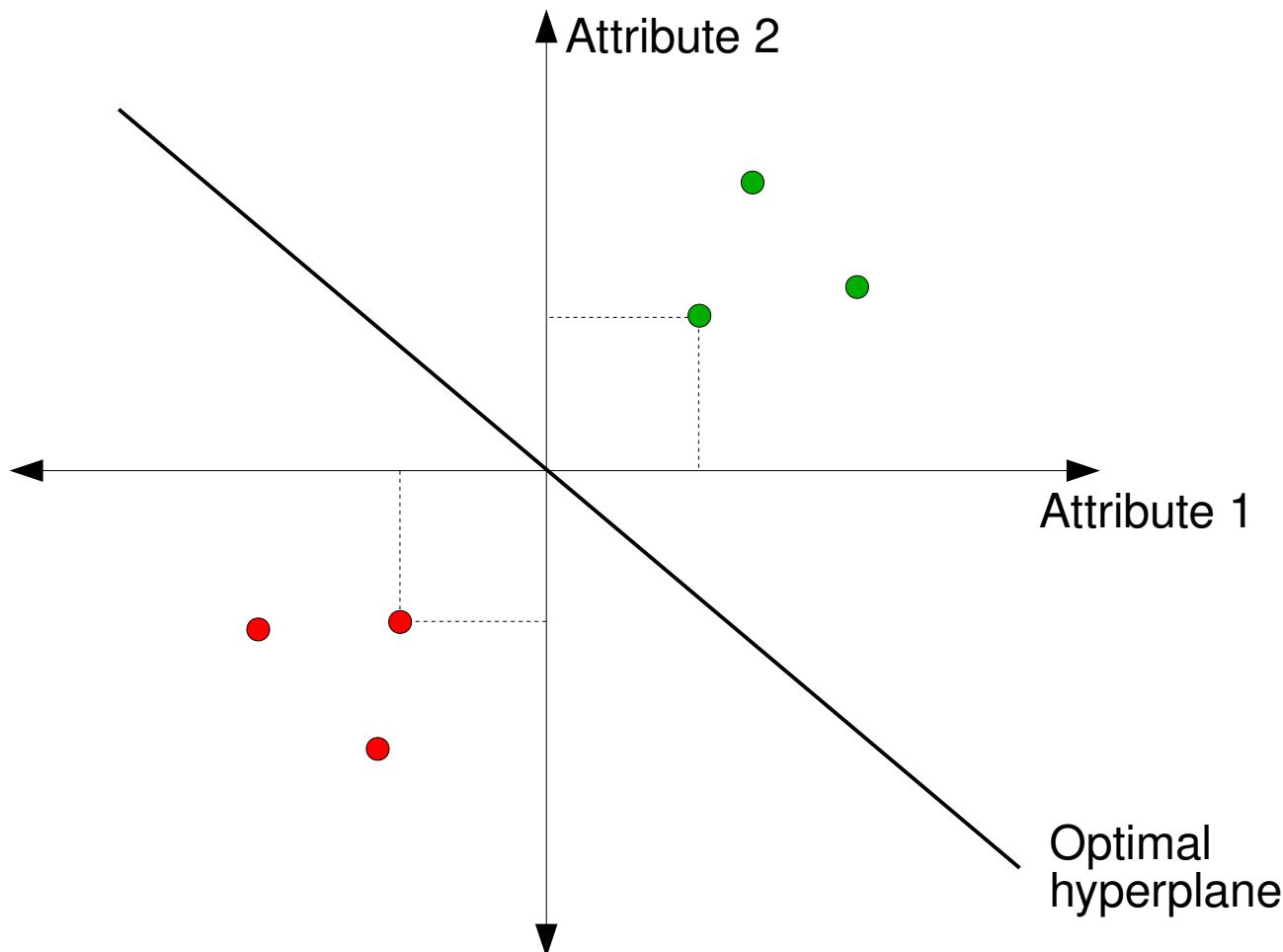
$$f(\mathbf{x}_i) = \langle \mathbf{w}, \mathbf{x}_i \rangle + b = 0$$



# Getting back to find the optimal hyperplane

- If the point lies on one of the sides we have either:

$$f(\mathbf{x}_i) = \langle \mathbf{w}, \mathbf{x}_i \rangle + b > 0 \text{ or } f(\mathbf{x}_i) = \langle \mathbf{w}, \mathbf{x}_i \rangle + b < 0$$



# Getting back to find the optimal hyperplane

- Finally we need to solve the problem:

$$\underset{\alpha \in \mathbb{R}^m}{\text{Maximize}} \quad W(\alpha) = -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^m \alpha_i$$

Subject to:  $\alpha_i \geq 0$ , for  $i = 1, \dots, m$ , and  $\sum_{i=1}^m \alpha_i y_i = 0$

- And after obtaining the hyperplane, the classification is simply by applying:

$$f(\mathbf{x}) = \text{sgn} (\langle \mathbf{w}, \mathbf{x} \rangle + b)$$

$$f(\mathbf{x}) = \text{sgn} \left( \sum_{i=1}^m \alpha_i y_i \langle \mathbf{x}, \mathbf{x}_i \rangle + b \right)$$

- And b is found using:  $\frac{\partial \Lambda}{\partial \alpha_i} = 0$  this is, using constraints.

# Getting back to find the optimal hyperplane

- Let's find  $b$ 
  - To find its value we need to solve the system:

$$\begin{cases} \langle \mathbf{x}_+, \mathbf{w} \rangle + b = +1 \\ \langle \mathbf{x}_-, \mathbf{w} \rangle + b = -1 \end{cases}$$

- So, we sum both equations to obtain:

$$\langle \mathbf{x}_+, \mathbf{w} \rangle + \langle \mathbf{x}_-, \mathbf{w} \rangle + 2b = +1 - 1$$

$$\langle \mathbf{x}_+ + \mathbf{x}_-, \mathbf{w} \rangle + 2b = 0$$

$$b = -\frac{1}{2} \langle \mathbf{x}_+ + \mathbf{x}_-, \mathbf{w} \rangle$$

# Getting back to find the optimal hyperplane

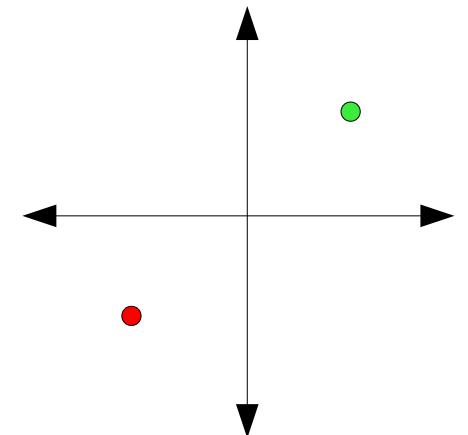
- Now, what about solving this optimization problem?
  - Suppose the simplest scenario at all

X  
(1,1)  
(-1,-1)

Y  
+1  
-1

Test the Lagrangian using values for both alphas:

0.1, 0.3, 0.5, 0.7, 1.0, 5, 10



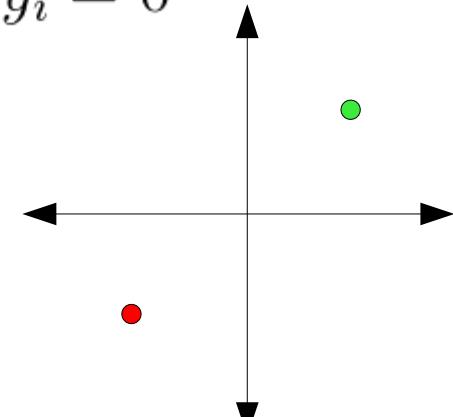
# Getting back to find the optimal hyperplane

- Now, what about solving this optimization problem?
  - Suppose the simplest scenario at all

$$\underset{\alpha \in \mathbb{R}^m}{\text{Maximize}} \quad W(\alpha) = -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^m \alpha_i$$

Subject to:  $\alpha_i \geq 0$ , for  $i = 1, \dots, m$ , and  $\sum_{i=1}^m \alpha_i y_i = 0$

```
x = NULL
x = rbind(c(+1, +1))
x = rbind(x, c(-1, -1))
y = c(+1, -1)
alphas = c(0.1, 0.1)
W_of_alpha = 0
for (i in 1:2) {
  for (j in 1:2) {
    W_of_alpha = W_of_alpha + alphas[i]*alphas[j]*y[i]*y[j]*x[i,] %*% x[j,]
  }
}
result = -1/2*W_of_alpha + sum(alphas)
print(result)
```



# Getting back to find the optimal hyperplane

- Now, what about solving this optimization problem?
  - Suppose the simplest scenario at all

Maximize  
 $\alpha \in \mathbb{R}^m$

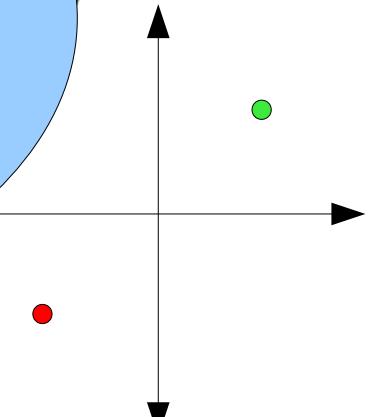
Subject to

```
x = rbind(x,rb1)
x = rbind(x,rb2)
y = c(+1,-1)
alphas = c(0.1,0.1)
W_of_alpha = 0
for (i in 1:2) {
  for (j in 1:2) {
    W_of_alpha = W_of_alpha + alphas[i]*alphas[j]*y[i]*y[j]*x[i]%%x[j]
  }
}
result = -1/2*W_of_alpha + sum(alphas)
print(result)
```

Now try with:

alphas = c(0.3, 0.3)  
alphas = c(0.5, 0.5)  
and  
alphas = c(1,1)

$$y_i \cdot w \geq 1 - \sum_{i=1}^m \alpha_i$$



# Getting back to find the optimal hyperplane

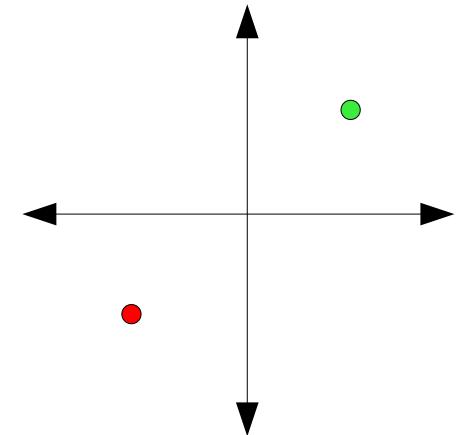
- Now, let's plug the values we found for alpha on:

$$f(\mathbf{x}) = \operatorname{sgn} (\langle \mathbf{w}, \mathbf{x} \rangle + b)$$

$$f(\mathbf{x}) = \operatorname{sgn} \left( \sum_{i=1}^m \alpha_i y_i \langle \mathbf{x}, \mathbf{x}_i \rangle + b \right)$$

- And try to classify unseen points:

(+3,+3)  
(-3, -3)



**This is just the start!**

# Getting back to find the optimal hyperplane

- Now we implement our very first SVM code

**Where kernels are plugged into this SVM formulation?**

# Where should we plug kernels into?

- Finally we need to solve the problem:

$$\underset{\alpha \in \mathbb{R}^m}{\text{Maximize}} \quad W(\alpha) = -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^m \alpha_i$$

Subject to:  $\alpha_i \geq 0$ , for  $i = 1, \dots, m$ , and  $\sum_{i=1}^m \alpha_i y_i = 0$

- And after obtaining the hyperplane, the classification is simply by applying:

$$f(\mathbf{x}) = \text{sgn} (\langle \mathbf{w}, \mathbf{x} \rangle + b)$$
$$f(\mathbf{x}) = \text{sgn} \left( \sum_{i=1}^m \alpha_i y_i \langle \mathbf{x}, \mathbf{x}_i \rangle + b \right)$$

# Where should we plug kernels into?

- Finally we need to solve the problem:

$$\underset{\alpha \in \mathbb{R}^m}{\text{Maximize}} \quad W(\alpha) = -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j k(x_i, x_j) + \sum_{i=1}^m \alpha_i$$

Subject to:  $\alpha_i \geq 0$ , for  $i = 1, \dots, m$ , and  $\sum_{i=1}^m \alpha_i y_i = 0$

- And after obtaining the hyperplane, the classification is simply by applying:

$$\begin{aligned} f(x) &= \text{sgn}(\langle \mathbf{w}, \mathbf{x} \rangle + b) = \text{sgn}\left(\sum_{i=1}^m \alpha_i y_i \langle \mathbf{x}, \mathbf{x}_i \rangle + b\right) \\ &= \text{sgn}\left(\sum_{i=1}^m \alpha_i y_i k(x, x_i) + b\right) \end{aligned}$$

Observe they were substituted by terms which are not feature vectors anymore! i.e., they need to be mapped into the features space

# Where should we plug kernels into?

- Observe they were substituted by terms which are not **feature vectors** anymore!
  - i.e., they need to be mapped into the features space
  - This is:

$$k(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle$$

- In which:

$$\begin{aligned}\mathbf{x}_i &= \Phi(x_i) \\ \mathbf{x}_j &= \Phi(x_j)\end{aligned}$$

- They were mapped into the features space

# Where should we plug kernels into?

- Common kernels:
  - Linear:  $k(x, x') = \langle \Phi(x), \Phi(x') \rangle = \langle \mathbf{x}, \mathbf{x}' \rangle$
  - Polynomial:  $k(x, x') = \langle \Phi(x), \Phi(x') \rangle^d = \langle \mathbf{x}, \mathbf{x}' \rangle^d$
  - Gaussian:  $k(x, x') = \langle \Phi(x), \Phi(x') \rangle = \exp\left(-\frac{\|\mathbf{x}-\mathbf{x}'\|^2}{2\sigma^2}\right)$
  - Sigmoidal:  $k(x, x') = \langle \Phi(x), \Phi(x') \rangle = \tanh(\kappa \langle \mathbf{x}, \mathbf{x}' \rangle + \Theta)$
- For suitable choices of:  $d \in \mathbb{N}$  and  $\sigma, \kappa, \Theta \in \mathbb{R}$

# Where should we plug kernels into?

- Homework:
  - Try to solve this optimization problem using any technique you know, for example:
    - Genetic Algorithms
    - Particle Swarm Optimization
    - Etc.

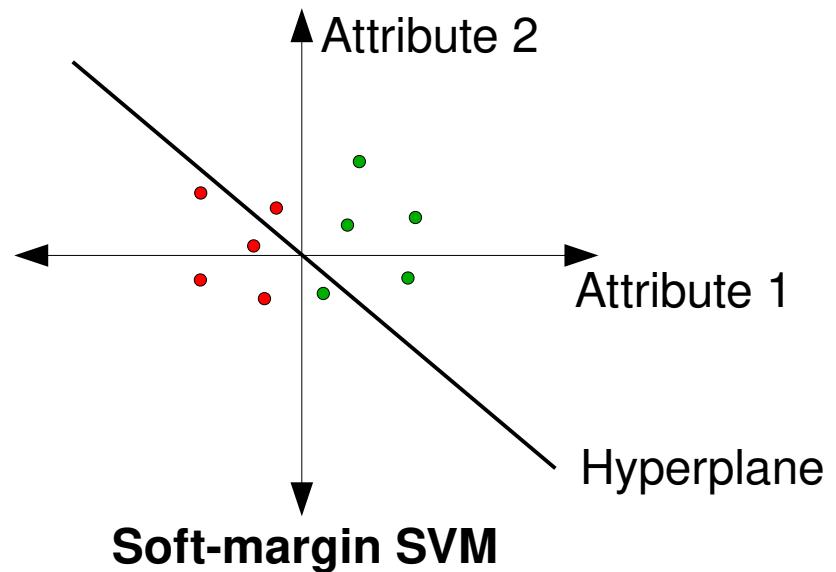
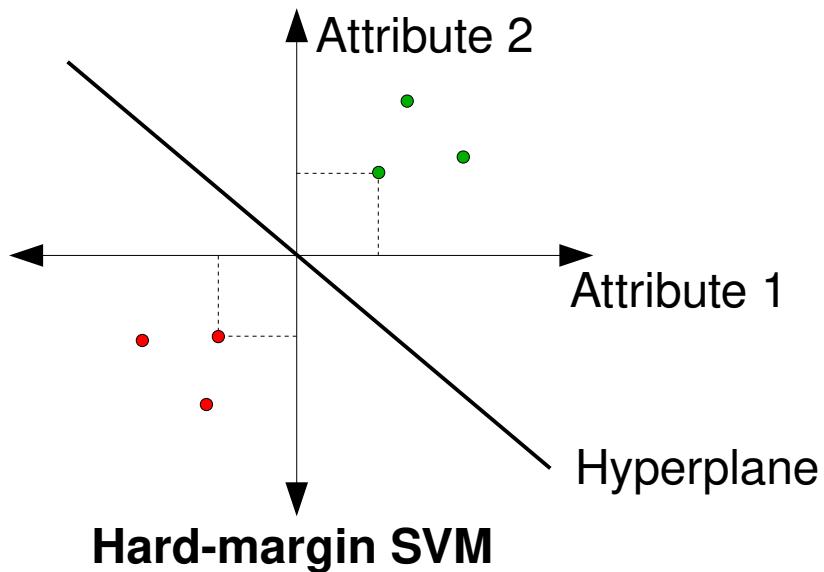
**We have seen the hard-margin SVM.  
What about the soft margin?**

- Using the hard-margin SVM:
  - There is no mix between classes {+1, -1}
  - This is not common in practical problems
- The soft-margin SVM introduces **slack variables** to relax the assumption of perfect separation between classes
  - Suppose slack variables in form:
$$\xi_i \geq 0 \text{ for all } i = 1, \dots, m$$
  - In order to relax the constraints as follows:

$$y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i \text{ for all } i = 1, \dots, m$$

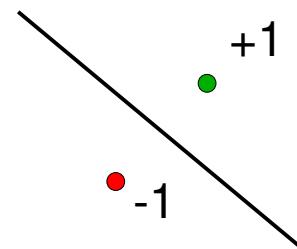
# Soft-margin SVM

- We can compare hard and soft-margin SVMs as follows:

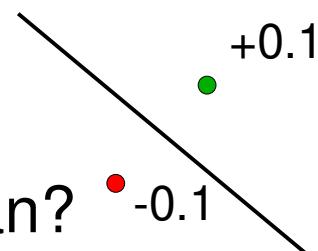


- But what is the effect of **slack variables**?

- What is the effect of **slack variables**?
  - The hyperplane does not have to provide outputs +1 and -1 to the closest points

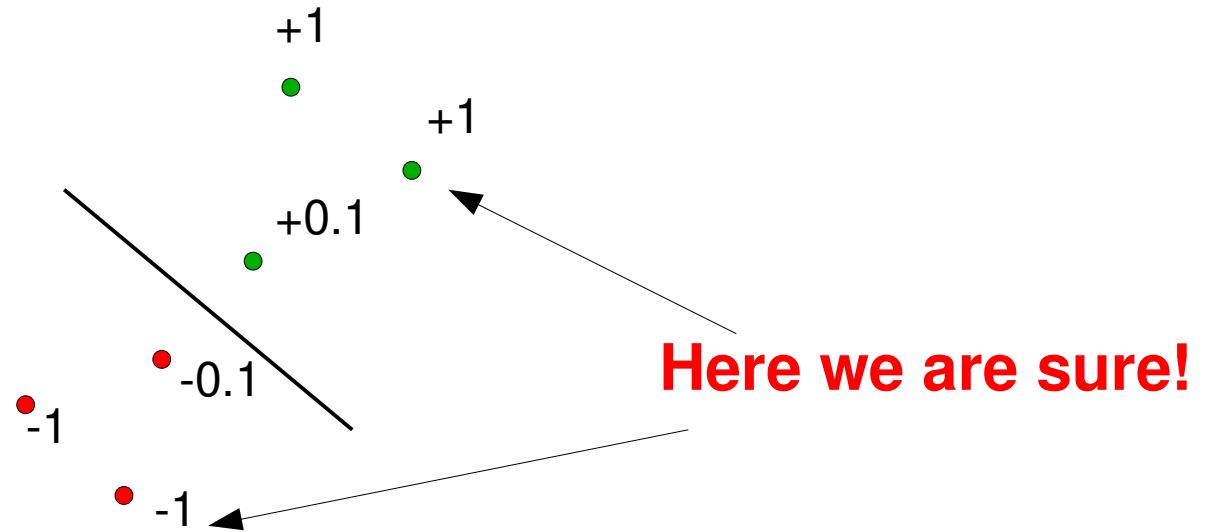


- It may provide smaller values such as:



- What does this mean?

- The effect of **slack variables**?
  - It works as a relaxation such as we are only sure about +1 and -1 after some distance from the hyperplane



- Consequently, as we want to be more sure about where the hyperplane should shatter the space, thus we need to use as less as possible those slack variables
  - i.e., if their values are greater than needed, more points are in this doubt or not-so-sure area

- Thus, the original optimization problem:

$$\min_{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}} \tau(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|^2$$

subject to  $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$  for all  $i = 1, \dots, m$ .

- Is rewritten in form:

$$\underset{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}, \xi \in \mathbb{R}^m}{\text{Minimize}} \quad \tau(\mathbf{w}, \xi) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \xi_i$$

subject to  $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i$  for all  $i = 1, \dots, m$   
 $\xi_i \geq 0$  for all  $i = 1, \dots, m$

- Where constant  $C > 0$  defines the trade-off between margin maximization and the classification relaxation

- Homework:
  - Rewrite this problem using KKT Conditions and obtain a maximization problem as before (for the hard-margin SVM)
  - Try to solve the maximization problem using any technique you know (as before)
    - Also try to apply kernels
      - Linear
      - Polynomial
      - Radial
      - Sigmoidal

- You should find the following dual form:

$$\begin{aligned} \underset{\alpha_i}{\text{Maximize}} \quad & W(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C \quad \text{for all } i = 1, \dots, m \\ & \sum_{i=1}^m \alpha_i y_i = 0 \end{aligned}$$

- And the classification function:

$$\begin{aligned} f(\mathbf{x}) &= \operatorname{sgn} (\langle \mathbf{w}, \mathbf{x} \rangle + b) \\ f(\mathbf{x}) &= \operatorname{sgn} \left( \sum_{i=1}^m \alpha_i y_i \langle \mathbf{x}, \mathbf{x}_i \rangle + b \right) \end{aligned}$$

# Soft-margin SVM

- Finding the **dual form** for the **soft-margin SVM problem**:

$$\underset{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}, \xi \in \mathbb{R}^m}{\text{Minimize}} \quad \tau(\mathbf{w}, \xi) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \xi_i$$

subject to  $y_i(\mathbf{w}, \mathbf{x}_i > +b) \geq 1 - \xi_i$  for all  $i = 1, \dots, m$   
 $\xi_i \geq 0$  for all  $i = 1, \dots, m$

- First of all we verify the KKT Conditions:

- (1) all constraints have to be satisfied
  - Something we assume
- (2) stationarity

**ξ must also compose the constraints!!! because every ξ<sub>i</sub> must be greater or equal zero!**

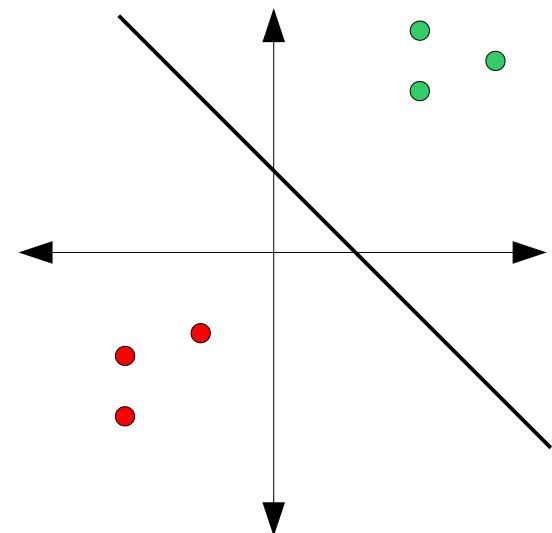
$$\Lambda(\mathbf{w}, b, \xi, \alpha, \lambda) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \xi_i$$

$$- \sum_{i=1}^m \alpha_i (y_i(\mathbf{w}, \mathbf{x}_i) - 1 + \xi_i) - \sum_{i=1}^m \lambda_i \xi_i$$

# Soft-margin SVM

- So we need to guarantee the KKT Conditions:
  - (3) Complementary slackness: KKT multipliers times inequality constraints are equal to zero
    - Either because the multiplier make the constraint equals zero or the constraint is non-binding
      - This indeed happens when attempting to find the optimal hyperplane for linearly separable data!

The points closest to the hyperplane will have alpha different from zero (they are called the **support vectors**), others will have alpha = 0



- So we need to guarantee the KKT Conditions:
  - (4) Dual feasibility
    - All KKT multipliers must be positive for inequality constraints, this is:
$$\alpha_i \geq 0, \text{ for all } i = 1, \dots, m$$
$$\lambda_i \geq 0, \text{ for all } i = 1, \dots, m$$
- If all KKT Conditions are satisfied after we solve the problem, then we have an optimal solution

# Soft-margin SVM

- Thus we solve the Lagrangian:

$$\begin{aligned}\Lambda(\mathbf{w}, b, \xi, \alpha, \lambda) = & \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \xi_i \\ & - \sum_{i=1}^m \alpha_i (y_i (\langle \mathbf{x}_i, \mathbf{w} \rangle + b) - 1 + \xi_i) - \sum_{i=1}^m \lambda_i \xi_i\end{aligned}$$

- As follows:

$$\frac{\partial \Lambda}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i = 0$$

$$\frac{\partial \Lambda}{\partial b} = \sum_{i=1}^m \alpha_i y_i = 0$$

- And in the direction of slack variables:

$$\frac{\partial \Lambda}{\partial \xi_i} = C - \alpha_i - \lambda_i = 0, \quad \text{for all } i = 1, \dots, m$$

- From the Lagrangian in the direction of slack variables we find:

$$\frac{\partial \Lambda}{\partial \xi_i} = C - \alpha_i - \lambda_i = 0, \text{ for all } i = 1, \dots, m$$

- And consequently:

$$\alpha_i = C - \lambda_i, \text{ for all } i = 1, \dots, m$$

- As (due to the Fourth KKT Condition):

$$\lambda_i \geq 0$$

- We finally have that:

$$0 \leq \alpha_i \leq C$$

# Soft-margin SVM

- Substituting  $\mathbf{w}$  in:

$$\Lambda(\mathbf{w}, b, \xi, \alpha, \lambda) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \xi_i - \sum_{i=1}^m \alpha_i (y_i (\langle \mathbf{x}_i, \mathbf{w} \rangle + b) - 1 + \xi_i) - \sum_{i=1}^m \lambda_i \xi_i$$

- We have:

$$\begin{aligned} W(\alpha, \lambda) &= \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + C \sum_{i=1}^m \xi_i \\ &\quad - \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle - b \sum_{i=1}^m \alpha_i y_i \\ &\quad + \sum_{i=1}^m \alpha_i - \sum_{i=1}^m \alpha_i \xi_i - \sum_{i=1}^m (C - \alpha_i) \xi_i \end{aligned}$$

This is zero!

- So we finally have:

$$W(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle$$

- Given:

$$0 \leq \alpha_i \leq C, \text{ for all } i = 1, \dots, m \text{ and}$$

$$\sum_{i=1}^m \alpha_i y_i = 0$$

- Thus we need to maximize this Lagrangian in terms of  $\alpha$ , respecting the constraints

# Soft-margin SVM

- So we have to solve the following problem:

$$\begin{aligned} \underset{\alpha \in \mathbb{R}^m}{\text{Maximize}} \quad & W(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \\ \text{subject to} \quad & 0 \leq \alpha_i \leq C, \quad \text{for all } i = 1, \dots, m \quad \text{and} \\ & \sum_{i=1}^m \alpha_i y_i = 0 \end{aligned}$$

- And the classification function is:

$$\begin{aligned} f(\mathbf{x}) &= \operatorname{sgn} (\langle \mathbf{w}, \mathbf{x} \rangle + b) \\ f(\mathbf{x}) &= \operatorname{sgn} \left( \sum_{i=1}^m \alpha_i y_i \langle \mathbf{x}, \mathbf{x}_i \rangle + b \right) \end{aligned}$$

- $b$  is found as before...

- And  $b$  is found after solving the system we must guarantee:

$$\begin{cases} \langle \mathbf{x}_+, \mathbf{w} \rangle + b = +1 - \xi_+ \\ \langle \mathbf{x}_-, \mathbf{w} \rangle + b = -1 - \xi_- \end{cases}$$

- By summing up both equations we have:

$$\langle \mathbf{x}_+ + \mathbf{x}_-, \mathbf{w} \rangle + 2b = -\xi_+ - \xi_-$$

- Thus we have:

$$b = -\frac{1}{2}(\xi_+ + \xi_-) - \frac{1}{2} \langle \mathbf{x}_+ + \mathbf{x}_-, \mathbf{w} \rangle$$

# Soft-margin SVM

- Now, implement it as you wish and try it!

- Lagrange Multipliers
  - Intuitive notion
    - We will use the demonstration code for Mathematica available at:
      - <http://demonstrations.wolfram.com/ConstrainedOptimization/>
    - Video is lagrange.ogv

- Karush-Kuhn-Tucker conditions
  - I suggest some videos I used to prepare this material:
    - <https://www.youtube.com/watch?v=eaKPzb11qFw&list=WL&index=3>
    - <https://www.youtube.com/watch?v=JTTiELgMyuM&list=WL&index=2>
    - <https://www.youtube.com/watch?v=AQWy73cHoIU&index=4&list=WL>
  - They are part of the course Optimization Techniques in Engineering available at:
    - <http://apmonitor.com/me575/index.php/Main/KuhnTucker>
      - By John D. Hedengren and Alan R. Parkinson

# **Designing the Support Vector Machine Optimization Algorithm**

- Due to the lack of quality in previous optimization algorithms:
  - We will now talk a little more about optimization
- Topics:
  - What is an optimization problem?
  - Main types of optimization problems
    - Solving linear problems
      - Examples
      - An overview about Primal versus Dual forms
      - Graphical interpretation of Primal and Dual forms
      - Using Lagrange multipliers to build Dual forms
    - Solving nonlinear but convex problems

# What is an Optimization Problem?

- An optimization problem has the following form:

Minimize  $f_0(x)$

subject to  $f_i(x) \leq b_i \quad i = 1, \dots, m$

- Having:

Vector  $x = (x_1, \dots, x_n)$  contains the optimization variables,  
i.e., the ones we should modify to optimize the problem.

Function  $f_0 : \mathbb{R}^n \rightarrow \mathbb{R}$  is the objective function.

Functions  $f_i : \mathbb{R}^n \rightarrow \mathbb{R}$  are the constraints.

Constants  $b_1, \dots, b_m$  are the constraint bounds.

# What is an Optimization Problem?

- Given this problem, a vector  $x^*$  is said **optimal** or the **problem solution** if it produces the minimal value for the objective function given all possible values that satisfy all constraints
  - This is, for every vector  $z$  in which:

$$f_1(z) \leq b_1, \dots, f_m(z) \leq b_m$$

- We have:
$$f_0(z) \geq f_0(x^*)$$
- Observation:** an optimization problem could also be written in maximization form

# Main Types of Optimization Problems

- Given this definition of an optimization problem, the main types of optimization problems we have are:
  - Linear optimization problems
  - Nonlinear optimization problems
    - Convex problems
    - Quasi-convex problems
    - Non-convex problems

# Main Types of Optimization Problems

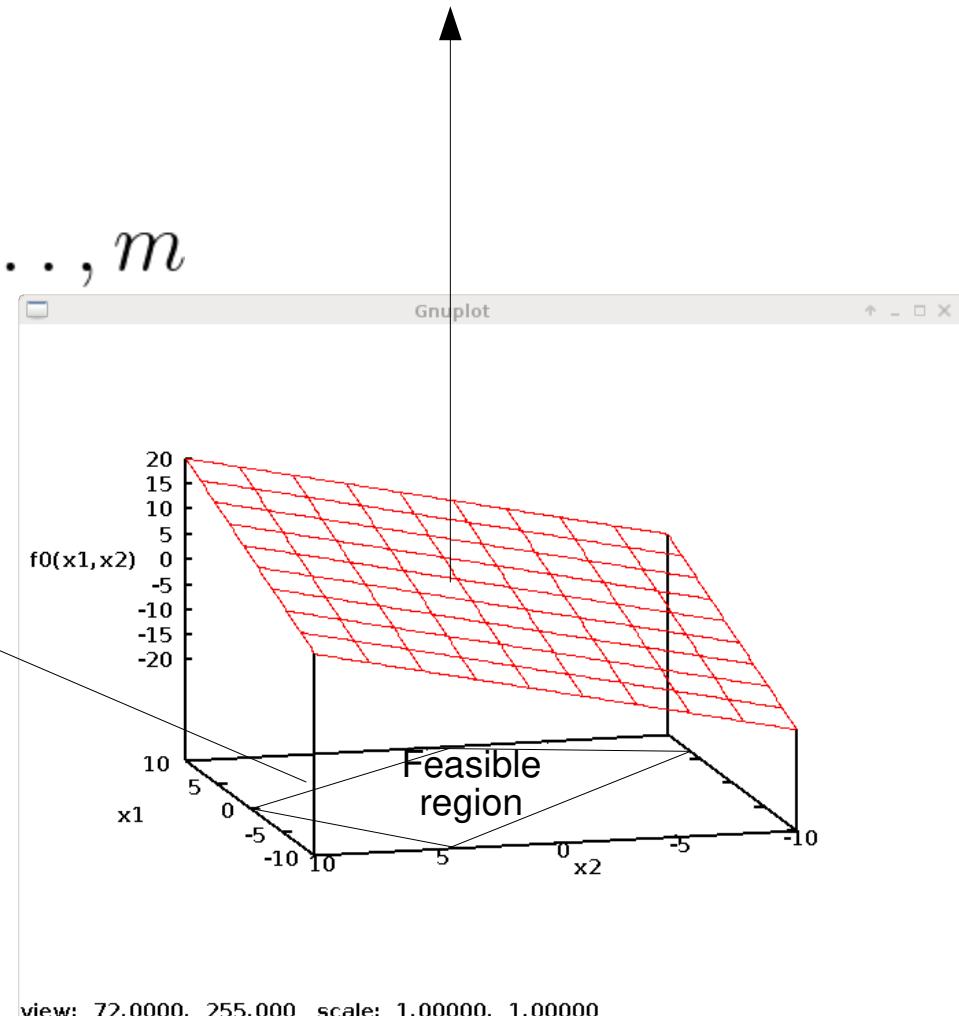
- Linear optimization problems
  - The objective function is **linear**
  - All constraints are **linear**

Minimize  $f_0(x)$

subject to  $f_i(x) \leq b_i \quad i = 1, \dots, m$

Constraints are linear

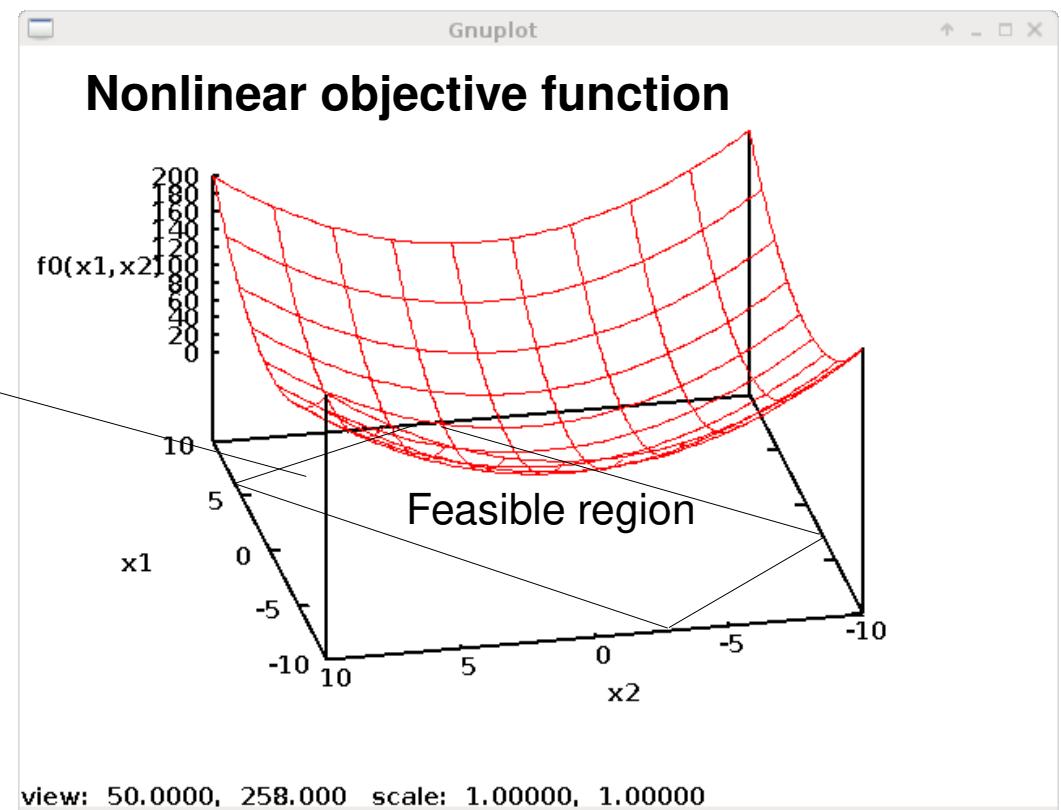
Objective function is linear



# Main Types of Optimization Problems

- Nonlinear optimization problems
  - One of the functions is **nonlinear**
    - Either the objective or any of the constraints

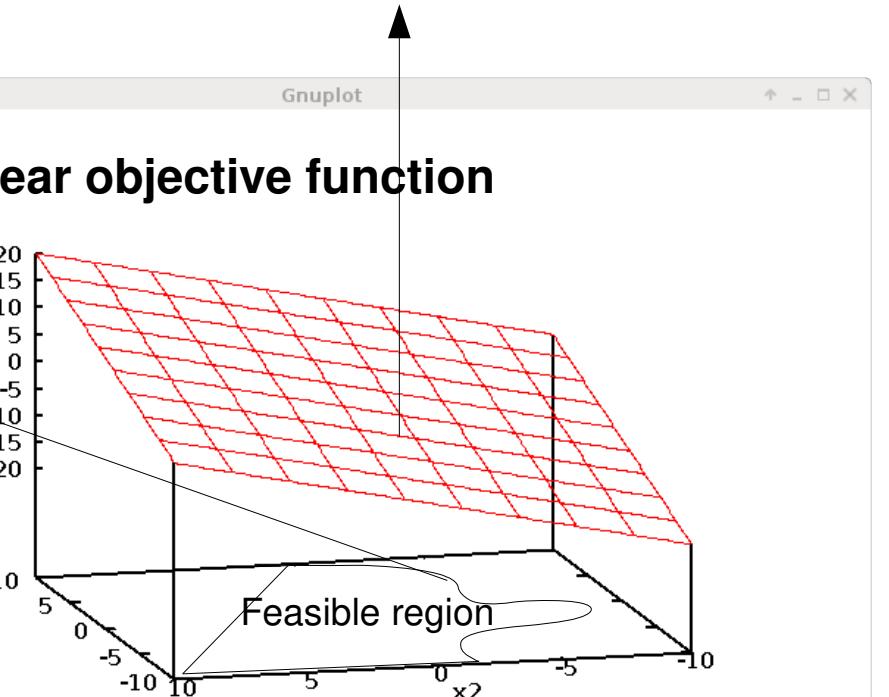
Linear  
Constraints



# Main Types of Optimization Problems

- Nonlinear optimization problems
  - One of the functions is **nonlinear**
    - Either the objective or any of the constraints

Objective function is linear



One of the constraints is nonlinear

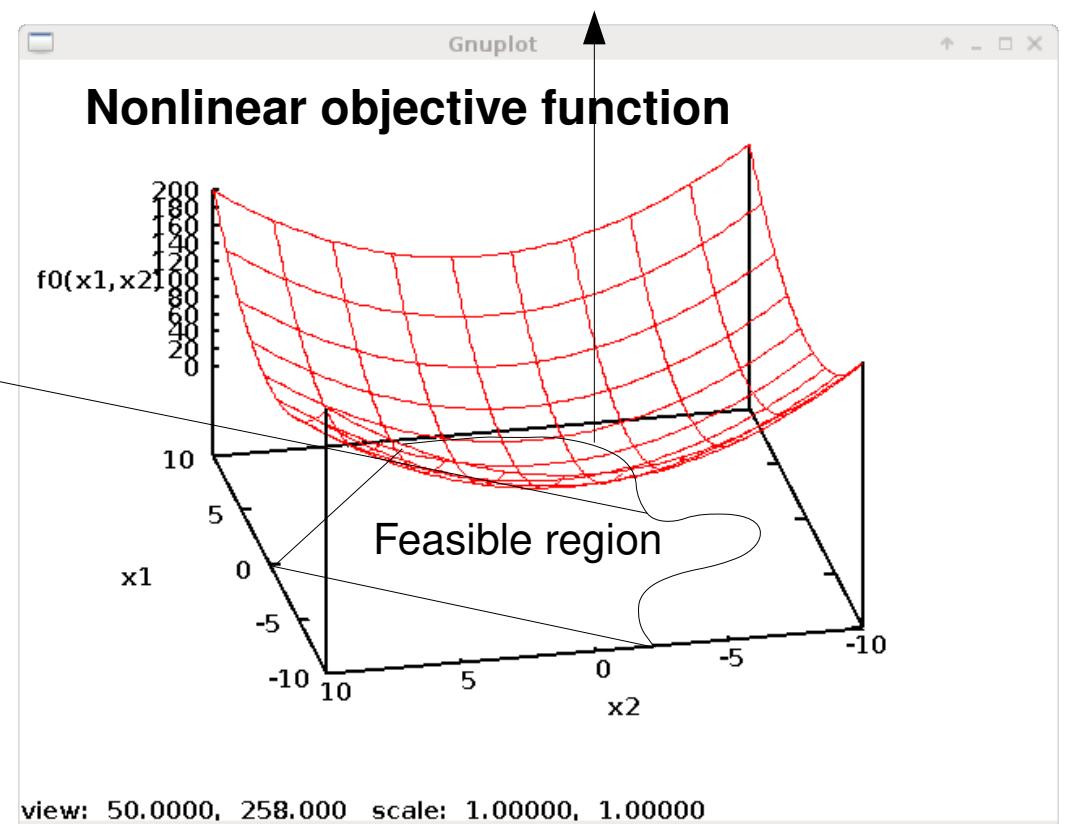


# Main Types of Optimization Problems

- Nonlinear optimization problems
  - One of the functions is **nonlinear**
    - Either the objective or any of the constraints

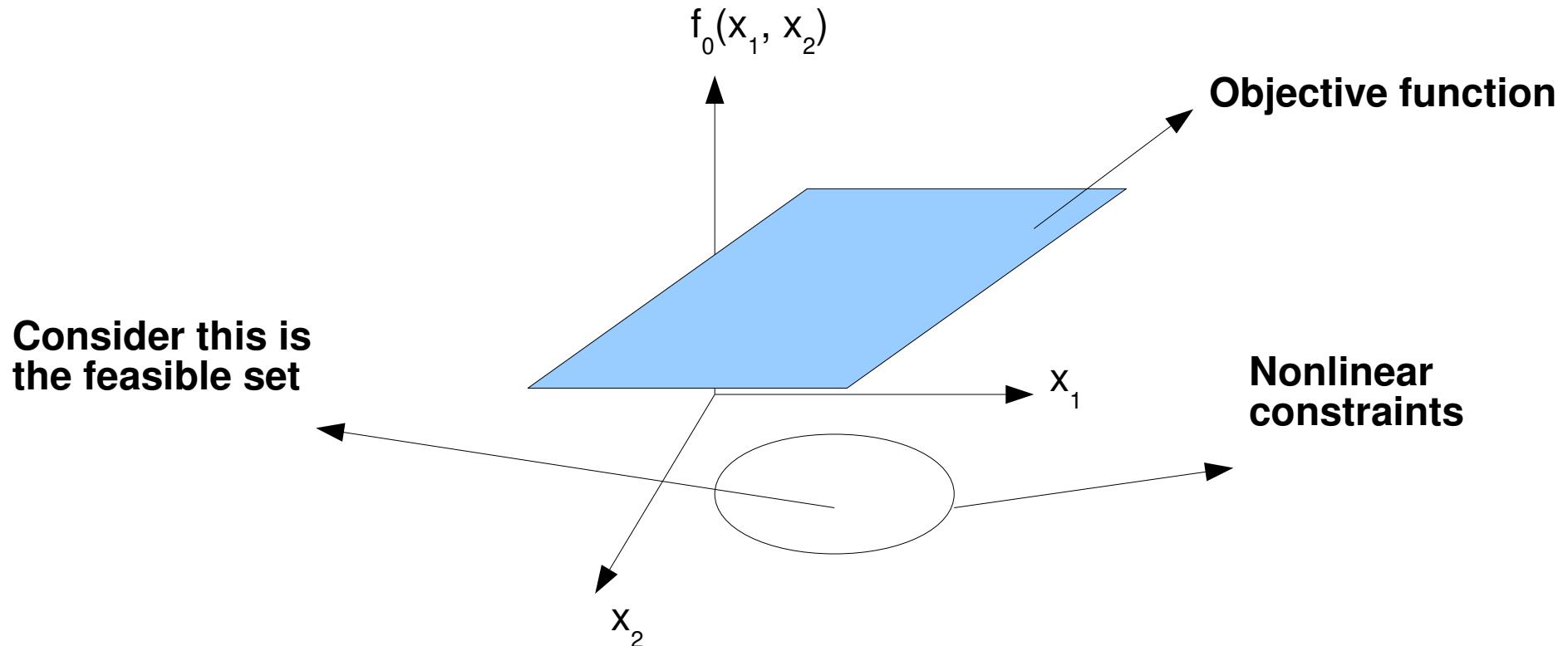
Objective function is nonlinear

One of the constraints is nonlinear



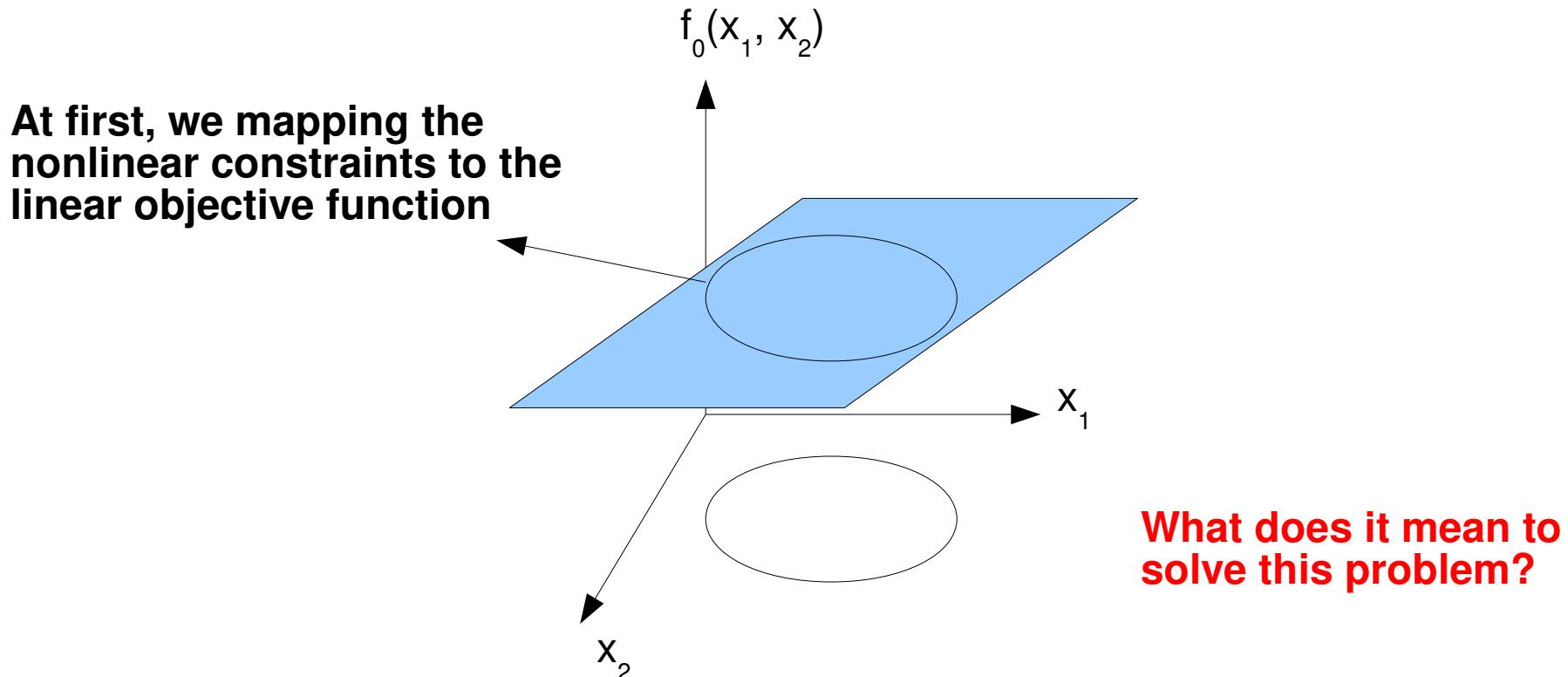
# Main Types of Optimization Problems

- Besides **nonlinearity**, consider the following situation:
  - Where:
    - The **feasible set** is **convex**
    - and the objective function is **linear**
      - This problem is still easy to solve



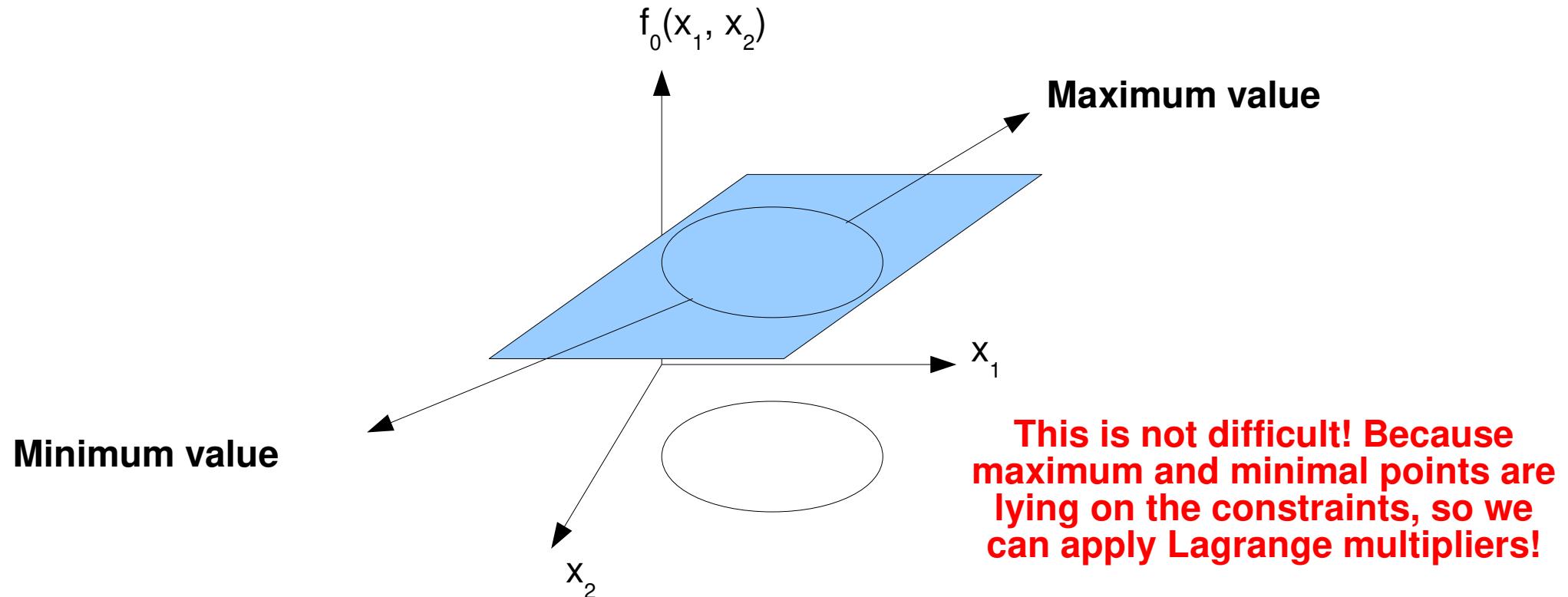
# Main Types of Optimization Problems

- Besides **nonlinearity**, consider the following situation:
  - Where:
    - The **feasible set** is **convex**
    - and the objective function is **linear**
      - This problem is still easy to solve



# Main Types of Optimization Problems

- Besides **nonlinearity**, consider the following situation:
  - Where:
    - The **feasible set** is **convex**
    - and the objective function is **linear**
      - This problem is still easy to solve

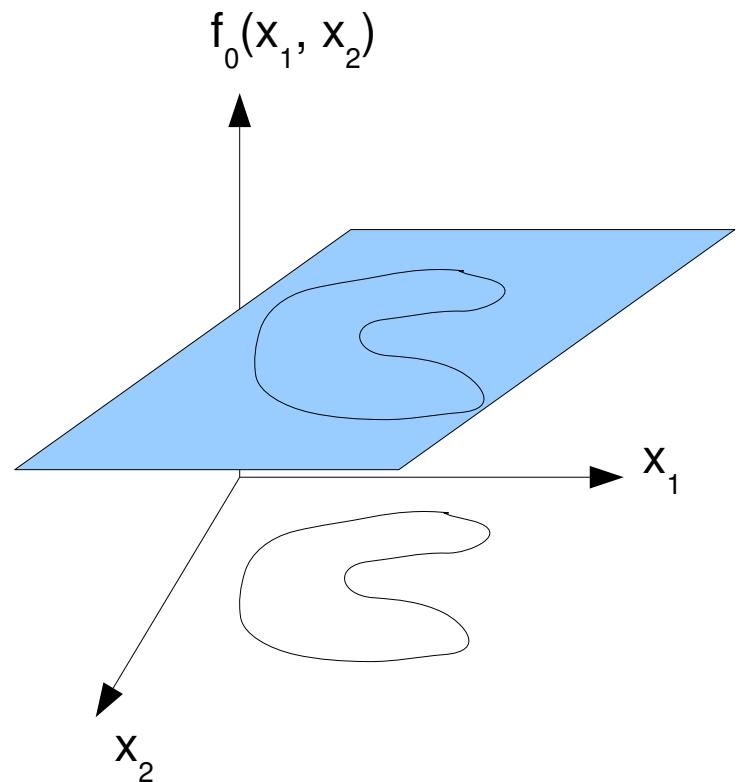


# Main Types of Optimization Problems

- Besides **nonlinearity**, consider the following situation:
  - Where:
    - The **feasible set** is **convex**
    - and the objective function is **nonlinear but convex**
      - We need more advanced methods:
        - Is a point lying on the constraint enough for you?
          - If so, you can use **Lagrange multipliers**
        - If you need a point inside the feasible region
          - Well, that requires more advanced techniques such as the **Interior Point Method**

# Main Types of Optimization Problems

- Besides **nonlinearity**, consider the following situation:
  - Where:
    - The **feasible set** is **not convex**
    - This is harder to solve specially when the objective function is nonlinear
    - It is common to approximate the nonlinear constraints using linear functions
      - Creating a convex approximation for the feasible set



# Main Types of Optimization Problems

- In our case, we are much more interested in:
  - **Linear problems**
    - We will start remembering how to solve such problems
  - **Nonlinear but convex problems**
    - In our case the typical problem for SVMs considers a convex (quadratic) objective function and linear constraints
    - So we need to learn how to find solutions for such problems

# **Solving Linear Optimization Problems**

# Solving Linear Optimization Problems

- Consider you have a **Linear objective function** you are trying to **maximize** or **minimize** subject to linear constraints
  - First steps:
    - Graph constraints
    - Compute the function at the vertices

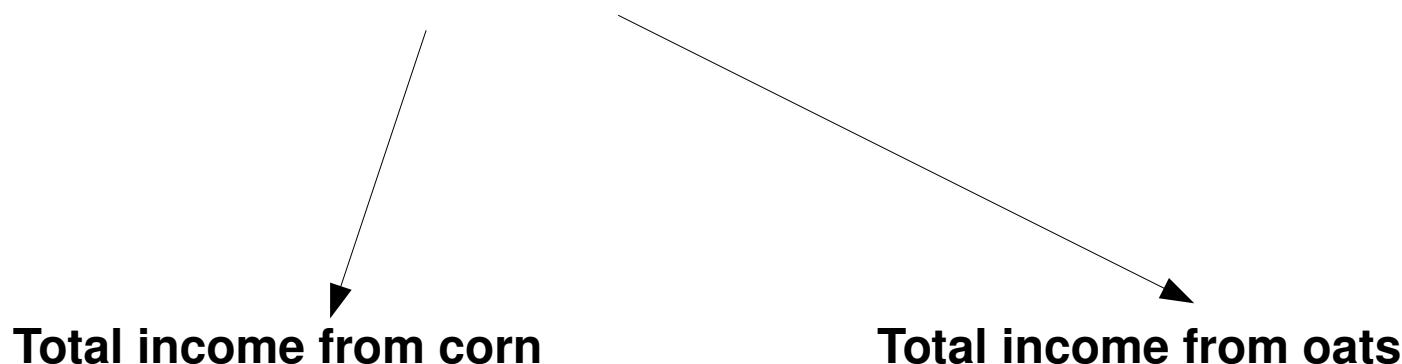
# Solving Linear Optimization Problems

- First example:
  - **Problem:** Suppose you have 240 acres of land and you can make:
    - \$40/acre if you grow corn and
    - \$30/acre if you grow oats
  - **Suppose:**
    - Corn takes 2 hours of labor per acre
    - Oats requires 1 hour per acre
    - You have only 320 hours of labor available
  - **Question:**
    - How many acres of each should you grow to maximize profit?

# Solving Linear Optimization Problems

- First example:
  - First we define:
    - X as the number of acres of corn we plant
    - Y as the number of acres of oats we plant
  - We also need to define the profit:

$$P = 40X + 30Y$$



# Solving Linear Optimization Problems

- First example:
  - First we define:
    - X as the number of acres of corn we plant
    - Y as the number of acres of oats we plant
  - We also need to define the profit:
$$P = 40X + 30Y$$
  - Now we need to define the constraints
$$X \geq 0, Y \geq 0$$



Because you cannot grow a  
negative number of acres!!!

# Solving Linear Optimization Problems

- First example:
  - First we define:
    - X as the number of acres of corn we plant
    - Y as the number of acres of oats we plant
  - We also need to define the profit:
$$P = 40X + 30Y$$
  - Now we need to define the constraints
$$X \geq 0, Y \geq 0$$
$$X + Y \leq 240$$



**Because you cannot grow more than the available land!!!**

# Solving Linear Optimization Problems

- First example:
  - First we define:
    - X as the number of acres of corn we plant
    - Y as the number of acres of oats we plant
  - We also need to define the profit:

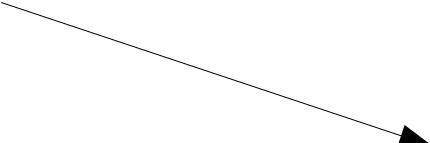
$$P = 40X + 30Y$$

- Now we need to define the constraints

$$X \geq 0, Y \geq 0$$

$$X + Y \leq 240$$

$$2X + Y \leq 320$$



This is the individual cost (in hours) to plant corn and oats. It cannot be over your available time! So, this is also a constraint!

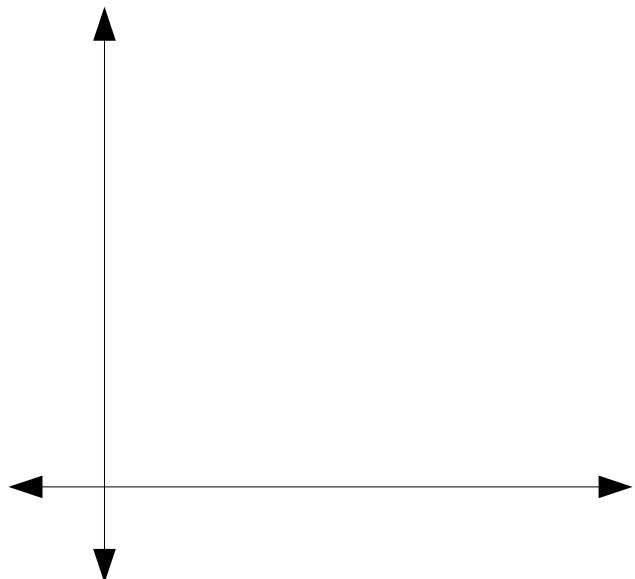
# Solving Linear Optimization Problems

- First example:
  - First we define:
    - X as the number of acres of corn we plant
    - Y as the number of acres of oats we plant
  - We also need to define the profit:
$$P = 40X + 30Y$$
  - Now we need to define the constraints
$$X \geq 0, Y \geq 0$$
$$X + Y \leq 240$$
$$2X + Y \leq 320$$
  - After having the problem formulated, we can graph it!!!

# Solving Linear Optimization Problems

- First example:
  - As next step we graph the constraints!
    - Let's graph the following constraint first:

$$X + Y \leq 240$$



# Solving Linear Optimization Problems

- First example:
  - As next step we graph the constraints!
    - Let's graph the following constraint first:

$$X + Y \leq 240$$

Let's find the coordinates when  
X = 0 and Y = 0 for the equality

So we have:

For X = 0

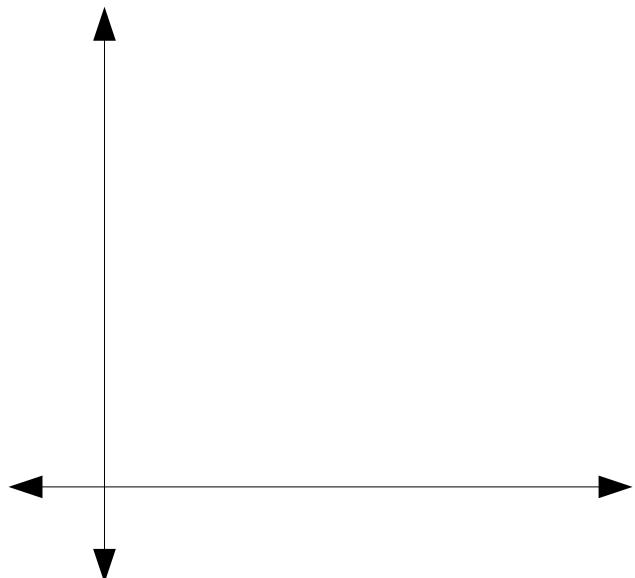
$$0 + Y = 240$$

$$Y = 240$$

For Y = 0

$$X + 0 = 240$$

$$X = 240$$



# Solving Linear Optimization Problems

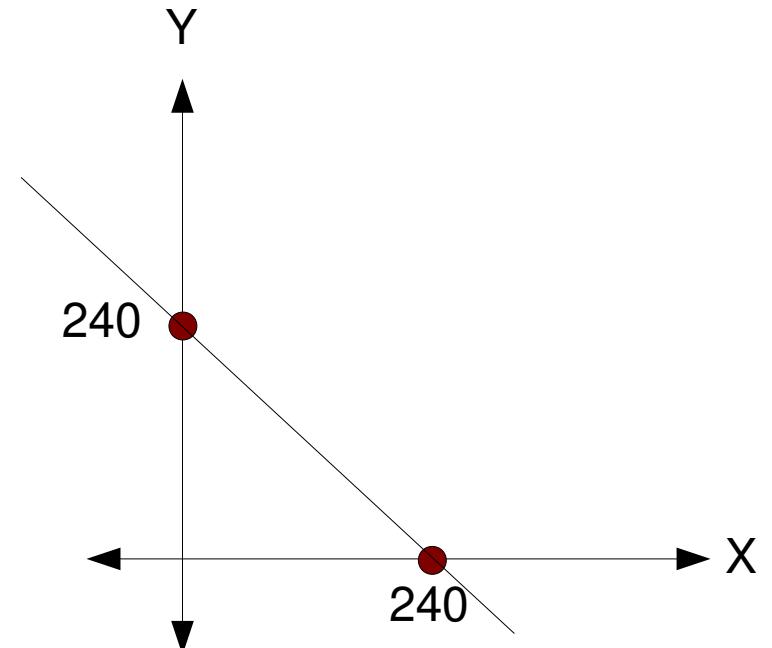
- First example:
  - As next step we graph the constraints!
    - Let's graph the following constraint first:

$$X + Y \leq 240$$

So we have the coordinates:

(0, 240)  
(240, 0)

For the equality, but we need to solve for the inequality!



# Solving Linear Optimization Problems

- First example:
  - As next step we graph the constraints!
    - Let's graph the following constraint first:

$$X + Y \leq 240$$

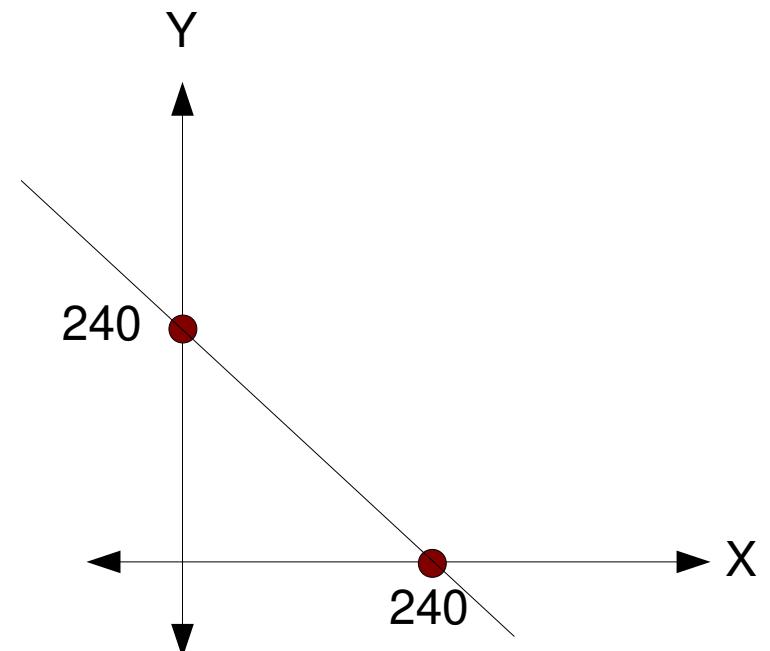
So we have the coordinates:

(0, 240)  
(240, 0)

For the equality, but we need to solve for the inequality!

So, we test for some X and Y pair such as (0,0)

$$0 + 0 \leq 240 \quad (\text{Yes, it is!})$$



# Solving Linear Optimization Problems

- First example:
  - As next step we graph the constraints!
    - Let's graph the following constraint first:

$$X + Y \leq 240$$

So we have the coordinates:

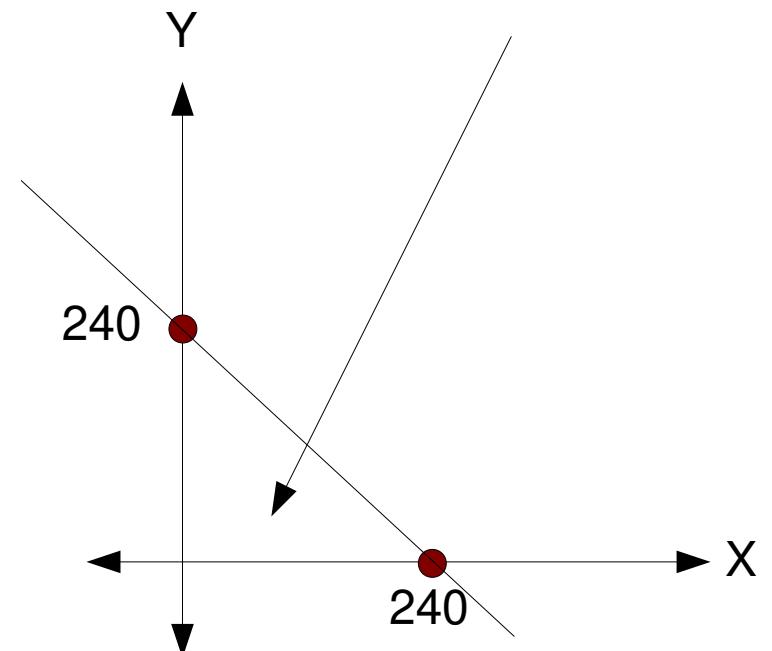
(0, 240)  
(240, 0)

For the equality, but we need to solve for the inequality!

So, we test for some X and Y pair such as (0,0)

$$0 + 0 \leq 240 \quad (\text{Yes, it is!})$$

So, this constraint makes the feasible region below the line!



# Solving Linear Optimization Problems

- First example:
  - As next step we graph the constraints!
    - Now we graph the second constraint:
$$2X + Y \leq 320$$

We solve it for the equality

So we have:

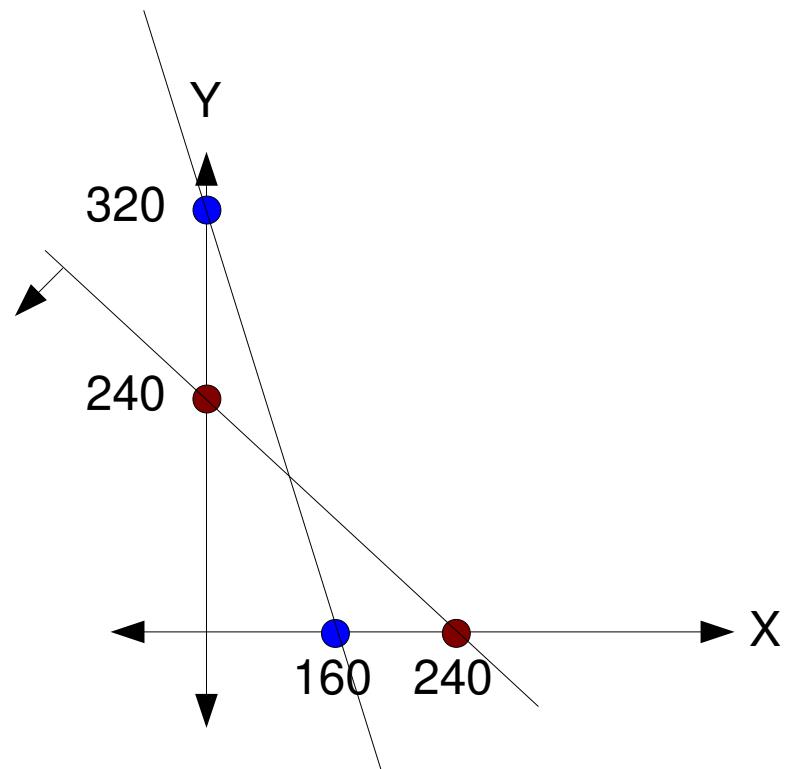
For  $X = 0$

$$\begin{aligned}2(0) + Y &= 320 \\Y &= 320\end{aligned}$$

For  $Y = 0$

$$\begin{aligned}2X + 0 &= 320 \\X &= 160\end{aligned}$$

Having the coordinates:  $(0, 320), (160, 0)$



# Solving Linear Optimization Problems

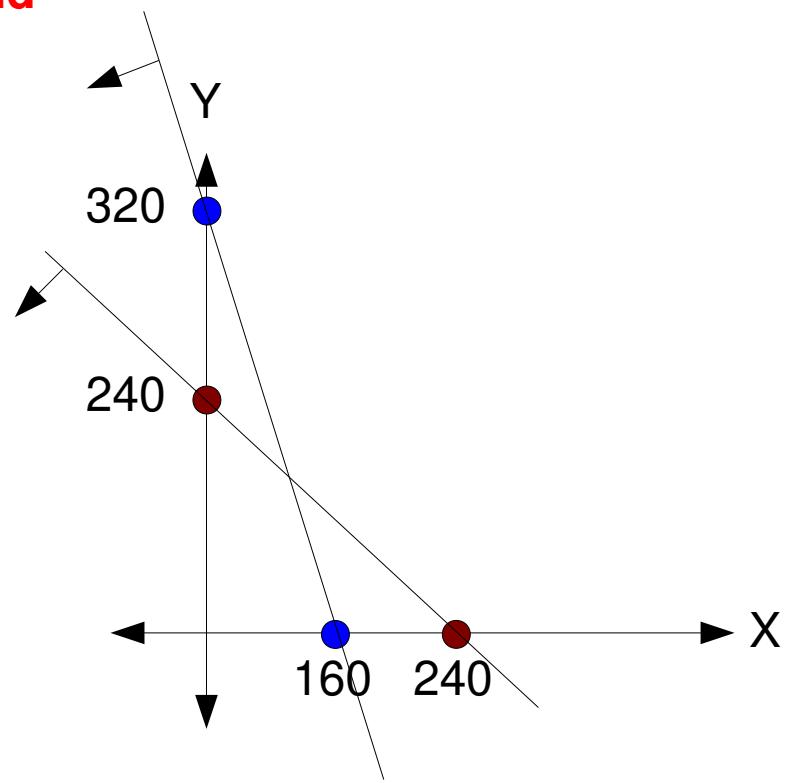
- First example:
  - As next step we graph the constraints!
    - Now we graph the second constraint:
$$2X + Y \leq 320$$

Then we find the feasible region for this second constraint

For example, solve for (0, 0)

$$2(0) + 0 \leq 320 \text{ (Yes, it is!)}$$

Also under the line!

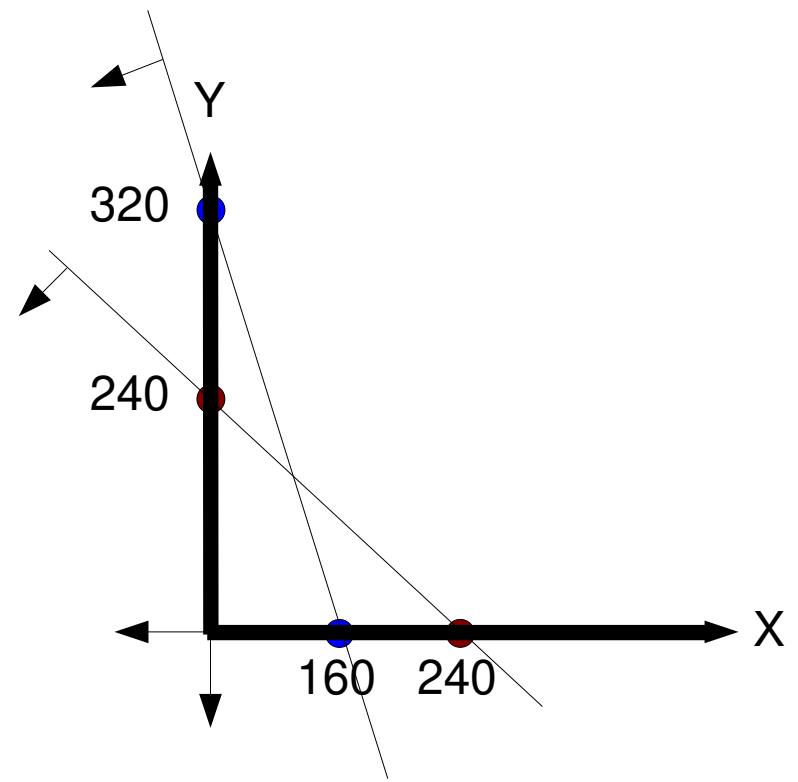


# Solving Linear Optimization Problems

- First example:
  - As next step we graph the constraints!
    - We still have two other constraints:
      - $X \geq 0$  and  $Y \geq 0$

Thus we can only consider positive values  
for X and Y

This encloses the overall feasible region  
for this problem!



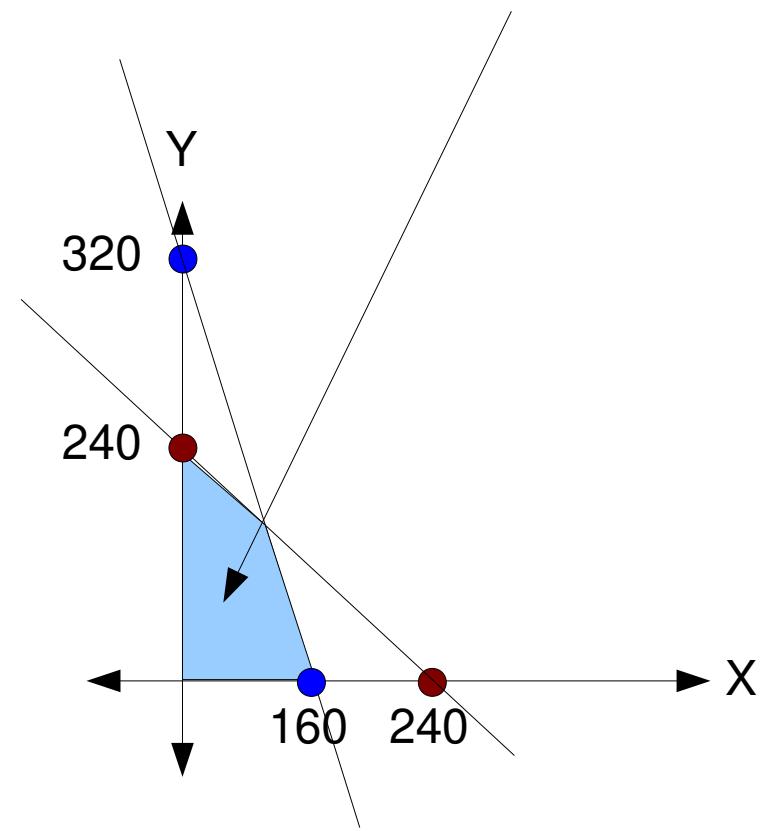
# Solving Linear Optimization Problems

- First example:
  - As next step we graph the constraints!
    - We still have two other constraints:
      - $X \geq 0$  and  $Y \geq 0$

Feasible region contains the overlap of all constraints!!!

Thus we can only consider positive values for X and Y

This encloses the overall feasible region for this problem!



# Solving Linear Optimization Problems

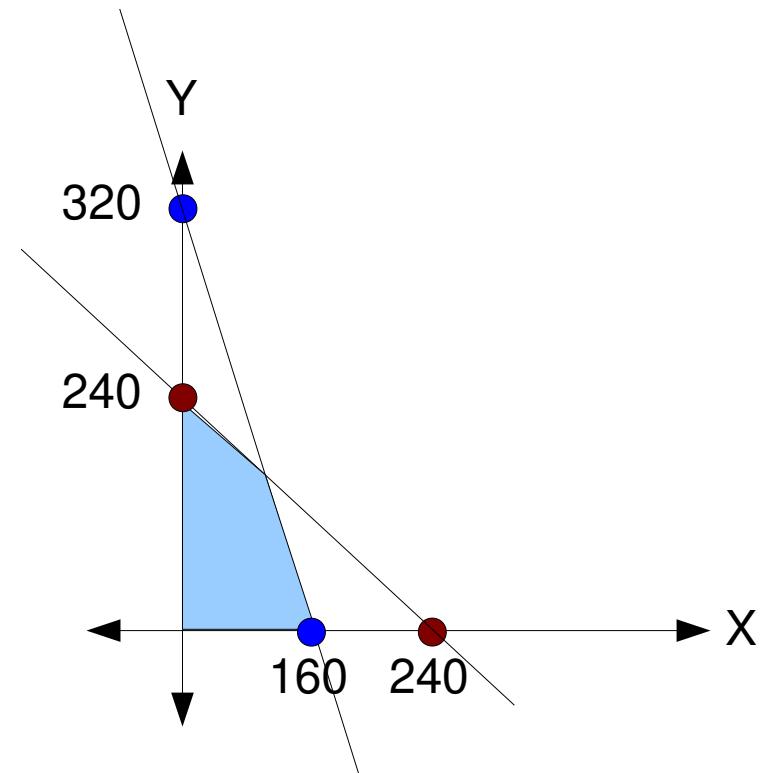
- First example:
  - Now we get back to the objective function:

$$P = 40X + 30Y$$

This objective function is a surface that requires a third axis P to be drawn

- A bit hard to draw :(

Imagine this third axis and the feasible region mapped on the surface!



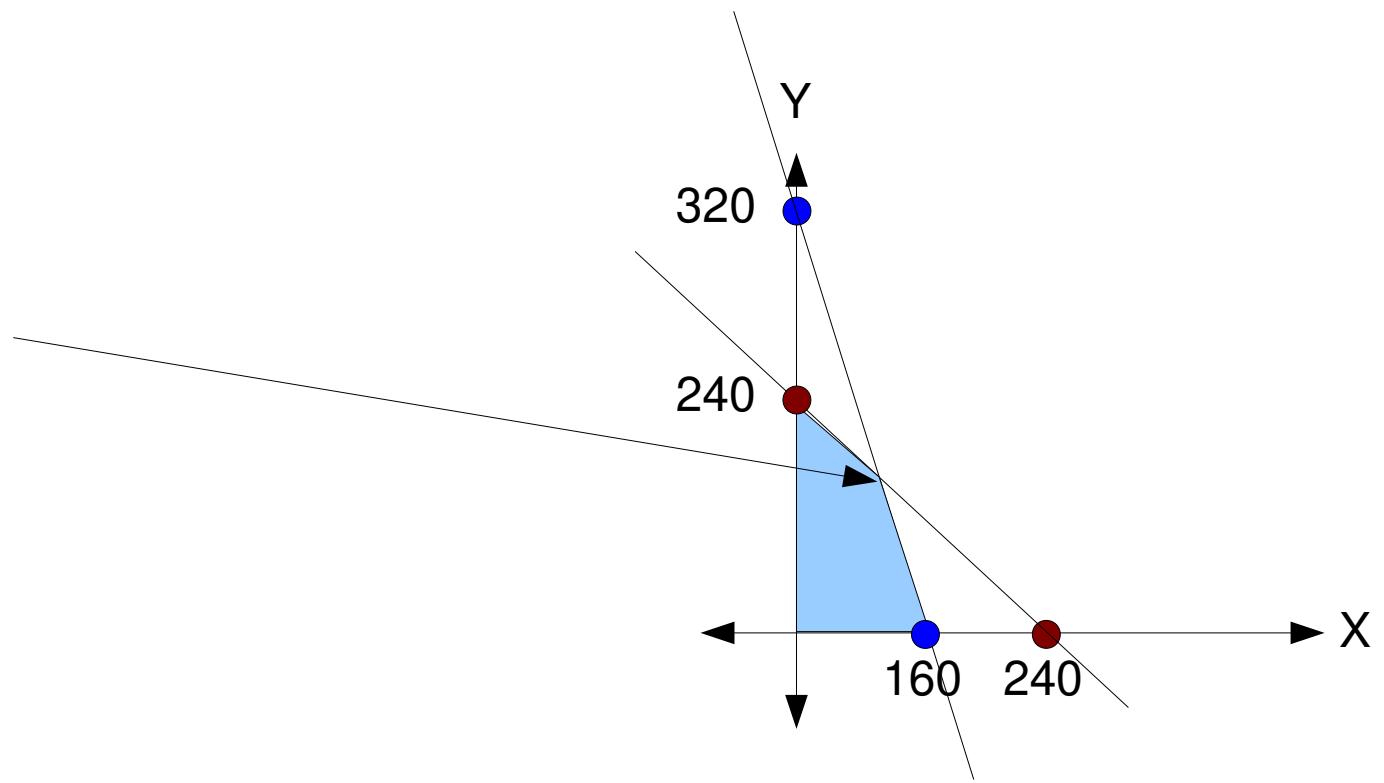
# Solving Linear Optimization Problems

- First example:
  - To solve this problem we only need to compute the objective function on specific points (**boundary points**)!

The points are:

(0,0)  
(160, 0)  
(0, 240)

And one more!!!



# Solving Linear Optimization Problems

- First example:
  - To solve this problem we only need to compute the objective function on specific points (**boundary points**)!

The points are:

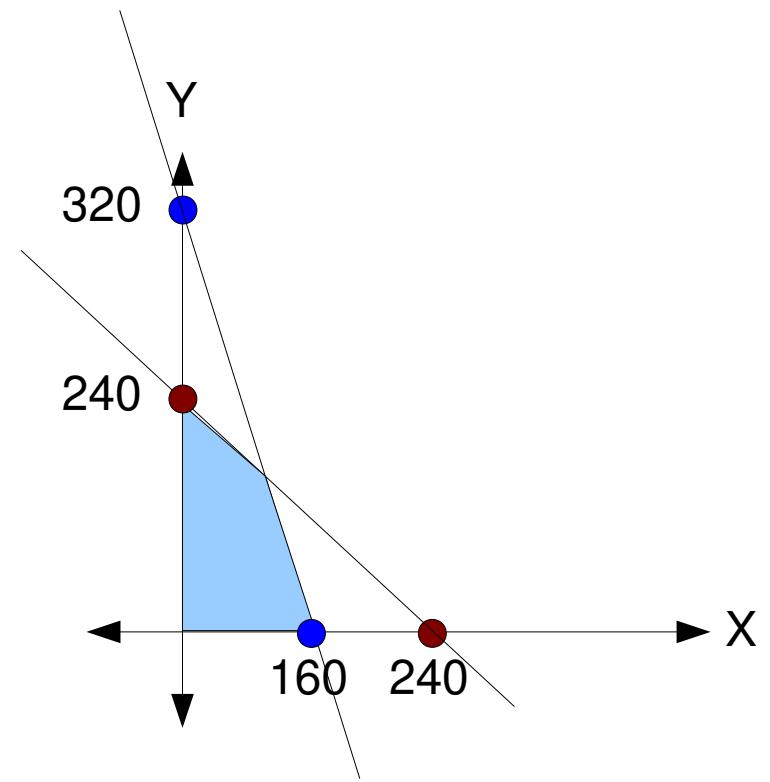
(0,0)  
(160, 0)  
(0, 240)

And one more!!!

Which comes from the equality of both constraints:

$$2X + Y = 320 \\ X + Y = 240$$

$$X = 320 - 240 = 80$$



# Solving Linear Optimization Problems

- First example:
  - To solve this problem we only need to compute the objective function on specific points (**boundary points**)!

The points are:

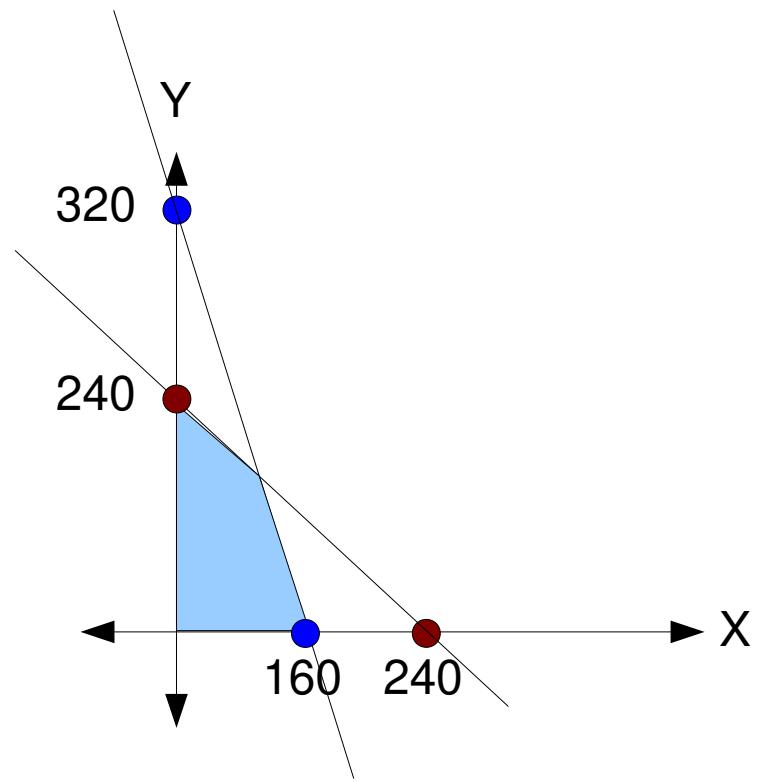
(0,0)  
(160, 0)  
(0, 240)

And one more!!!

Plugging  $X = 80$  we have:

$$Y = 240 - 80 \\ Y = 160$$

So the point is (80, 160)



# Solving Linear Optimization Problems

- First example:
  - To solve this problem we only need to compute the objective function on specific points (**boundary points**)!

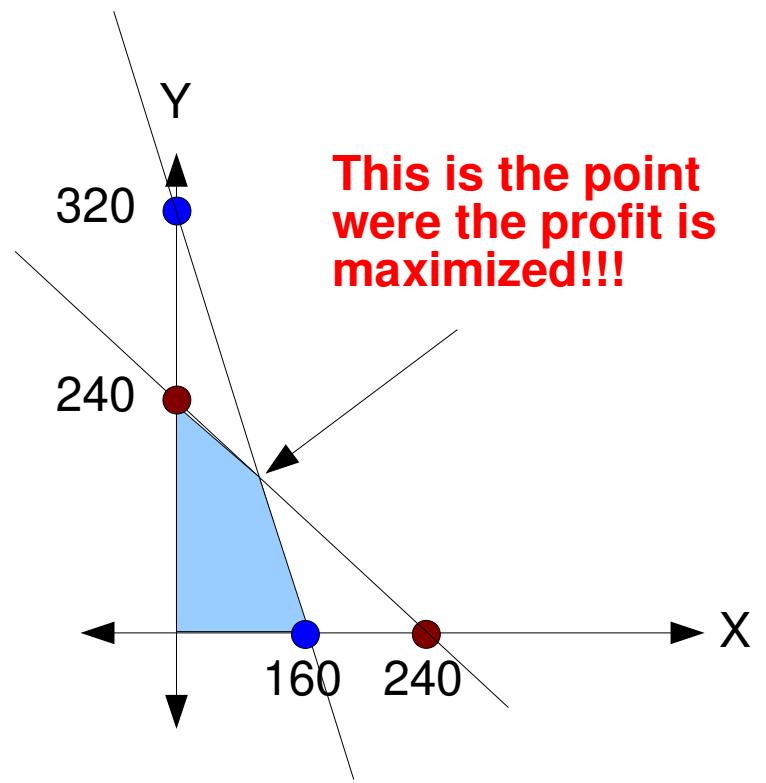
Applying P on points we have:

$$P(0,0) = 0$$

$$P(160, 0) = 6400$$

$$P(0, 240) = 7200$$

$$P(80, 160) = 8000$$



# Solving Linear Optimization Problems

- First example:
  - To solve this problem we only need to compute the objective function on specific points (**boundary points**)!

Applying P.O.

P(0,0)

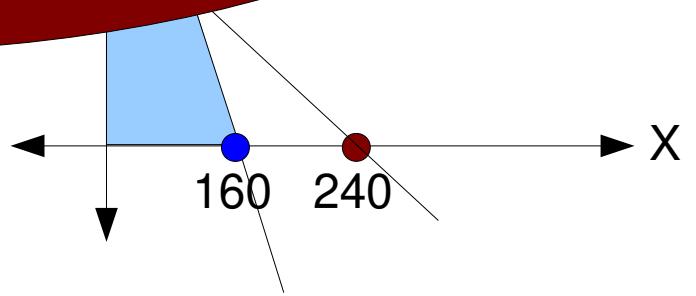
P(1,0)

P(0,1)

P(2,0)

We should plant 80 acres  
of corn and 160 acres of oats!

This is the solution for this  
Linear Problem!



**What about solving another Linear  
Optimization Problem**

# Solving Linear Optimization Problems

- Suppose a rancher who wants to mix two types of food for his cattle:
  - Brand X
    - 15 grams of protein
    - 10 grams of fat
    - Costs 80 cents per unit
  - Brand Y
    - 20 grams of protein
    - 5 grams of fat
    - Costs 50 cents per unit
- Consider **each serving** is required to have **at least**:
  - 60 grams of protein
  - 30 grams of fat
- **Question:** How much of each type of food should be used to minimize the cost to the rancher?

# Solving Linear Optimization Problems

- We will refer to:
  - X as the number of units of Brand X
  - Y as the number of units of Brand Y
- So we define the Cost to be minimized as:
$$C = 0.8 X + 0.5 Y$$

# Solving Linear Optimization Problems

- We will refer to:
  - X as the number of units of Brand X
  - Y as the number of units of Brand Y
- So we define the Cost to be minimized as:
$$C = 0.8 X + 0.5 Y$$
- What about the constraints?
  - First of all the number of units MUST be greater than or equal to zero, so:
$$X \geq 0 \text{ and } Y \geq 0$$
  - As there is no sense in mixing a negative number of units

# Solving Linear Optimization Problems

- We will refer to:
  - X as the number of units of Brand X
  - Y as the number of units of Brand Y
- So we define the Cost to be minimized as:
$$C = 0.8 X + 0.5 Y$$
- What about the constraints?
$$X \geq 0 \text{ and } Y \geq 0$$
- Now we need to check it out the grams of protein and fat necessary
  - First the protein!

# Solving Linear Optimization Problems

- First the protein!

$$15X + 20Y \geq 60$$

- This inequality is justified because cattle requires at least 60 grams of protein per serving

- Now fat:

$$10X + 5Y \geq 30$$

- This inequality is justified because cattle requires at least 30 grams of fat per serving

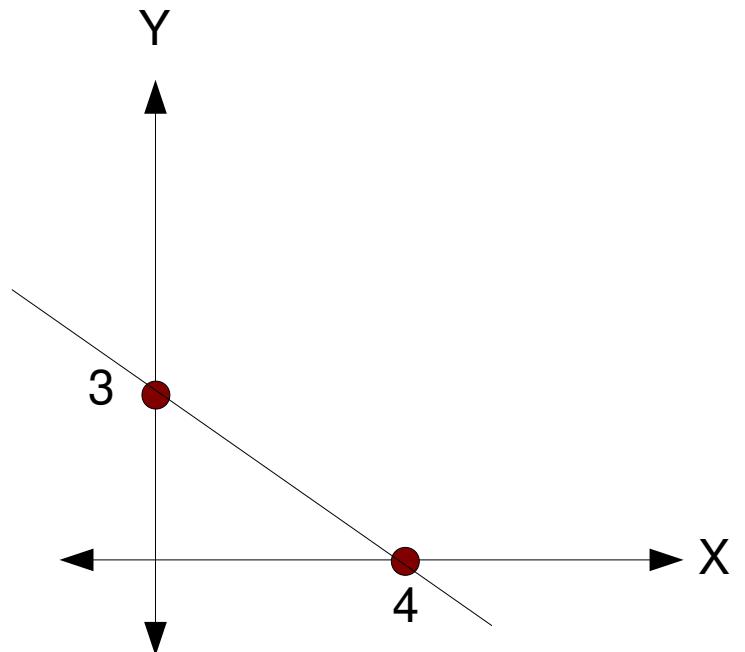
- Yes! Now we have all constraints!

- Then we will graph the feasible region for this problem!

# Solving Linear Optimization Problems

- Second example:
  - As next step we graph the constraints!
    - Firstly we graph the following constraint:
$$15X + 20Y \geq 60$$

We can find the intercepts which will be...



# Solving Linear Optimization Problems

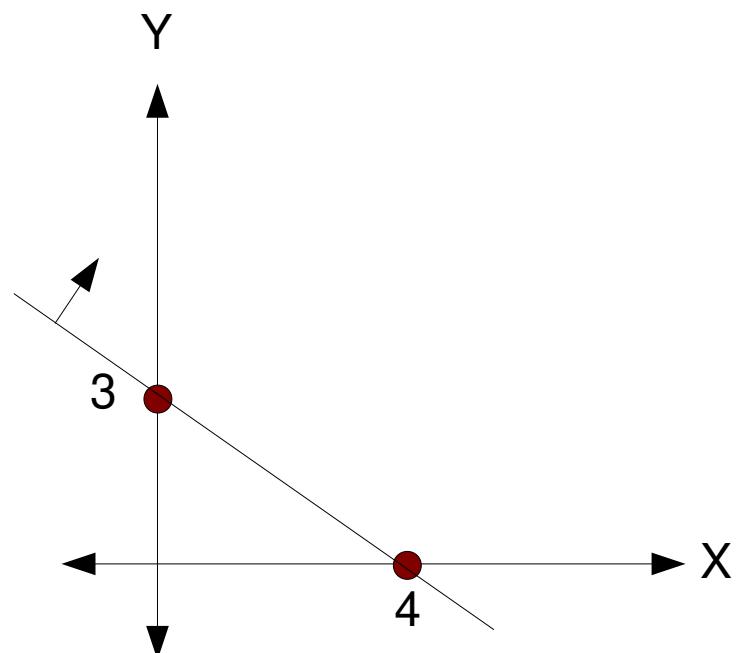
- Second example:
  - As next step we graph the constraints!
    - Firstly we graph the following constraint:
$$15 X + 20 Y \geq 60$$

Now we plug some value to understand which is the side of this line to be considered as part of the feasible region

Plugging (0,0) we get:

$$15 (0) + 20 (0) \geq 60 \quad (\text{No way!})$$

So the above side is part of the feasible region!

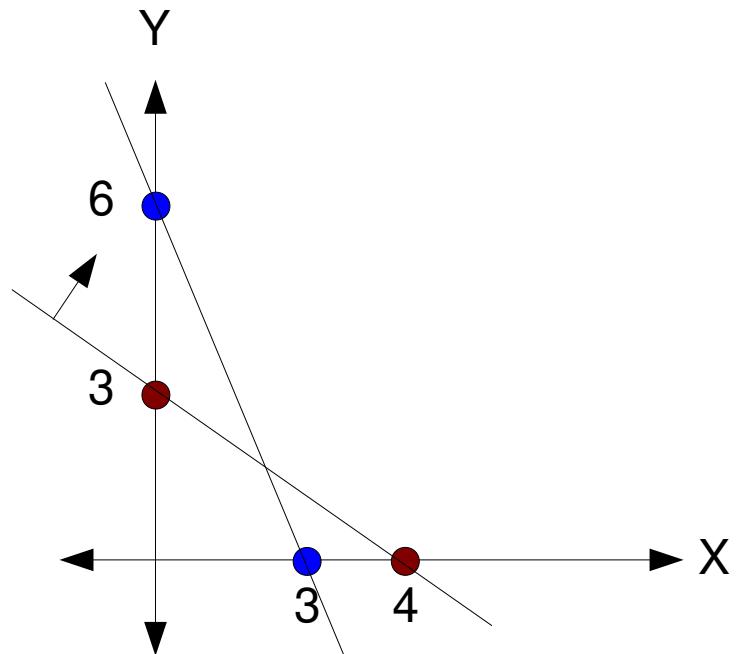


# Solving Linear Optimization Problems

- Second example:
  - As next step we graph the constraints!
  - Then we graph the following constraint:

$$10 X + 5 Y \geq 30$$

We can find the intercepts which will be...



# Solving Linear Optimization Problems

- Second example:
  - As next step we graph the constraints!
    - Then we graph the following constraint:

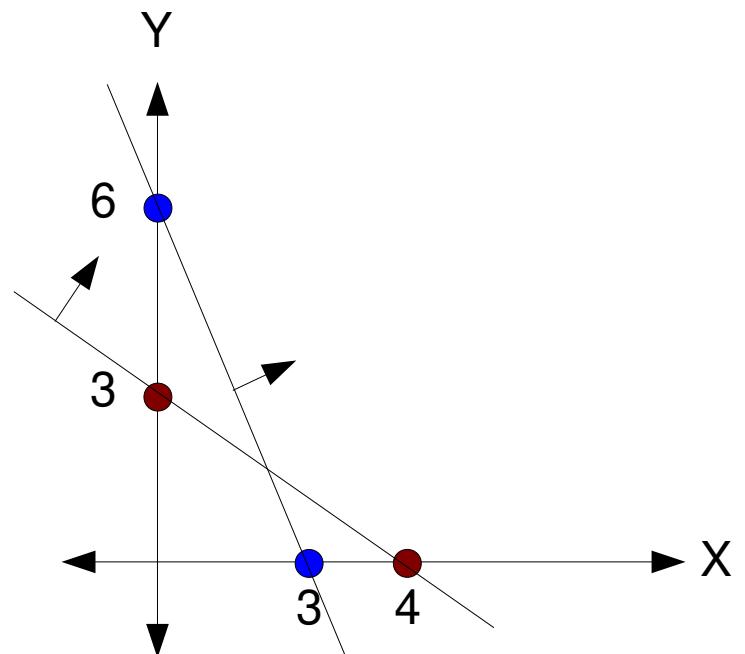
$$10 X + 5 Y \geq 30$$

Then we need to test which side is part of the feasible region

Plugging  $(0, 0)$  we obtain:

$10 (0) + 5 (0) \geq 30$  (No way!)

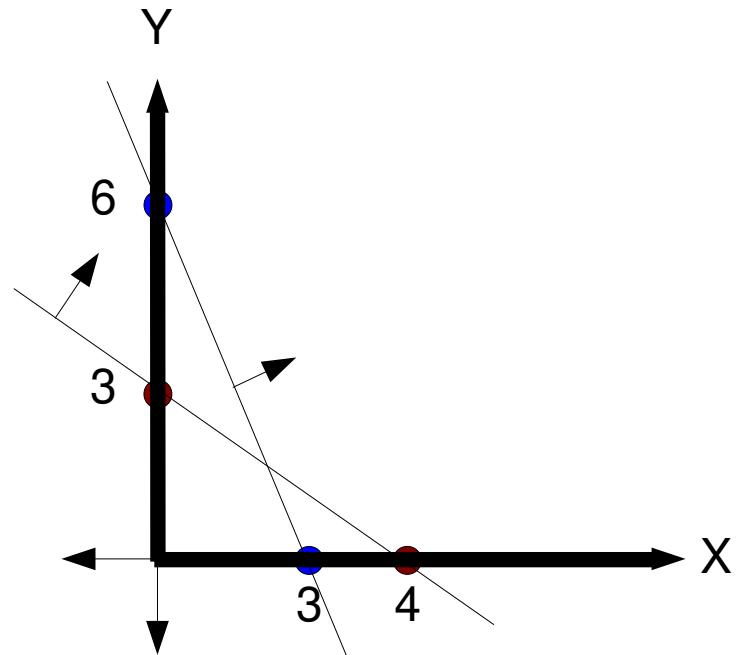
So we also must consider the region above!



# Solving Linear Optimization Problems

- Second example:
  - As next step we graph the constraints!
    - Remember  $X \geq 0$  and  $Y \geq 0$

So now we can observe all the overlappings among constraints and found out the feasible region!



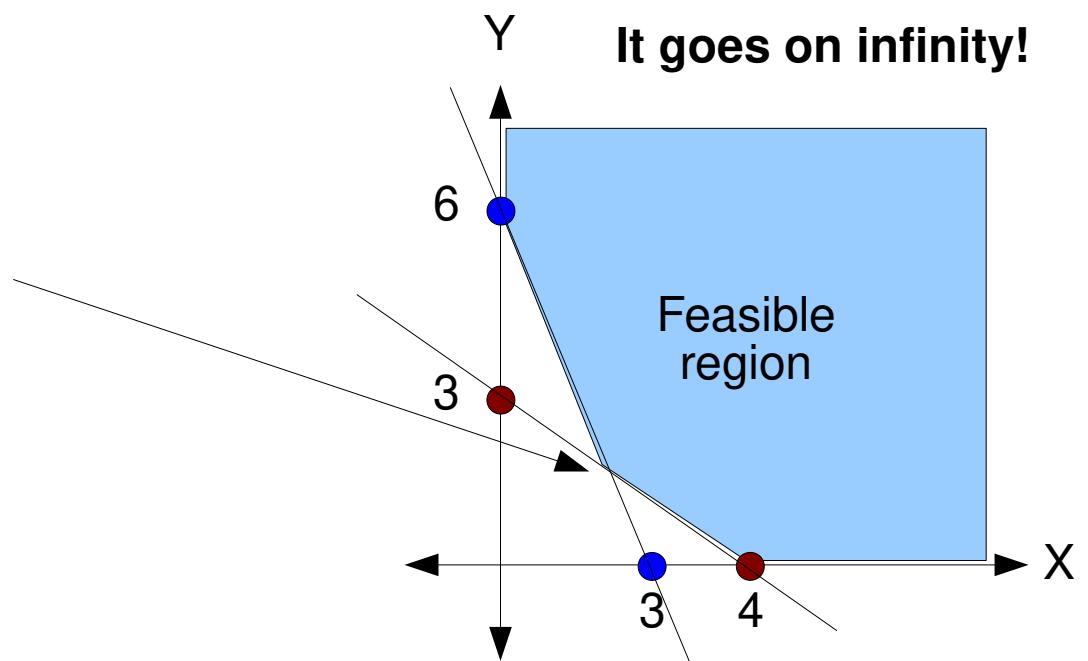
# Solving Linear Optimization Problems

- Second example:
  - So we have the feasible region

Now we need to evaluate the main points for our problem which are:

(4,0)  
(0,6)

And an additional one...



# Solving Linear Optimization Problems

- Second example:
  - So we have the feasible region

Now we need to evaluate the main points for our problem which are:

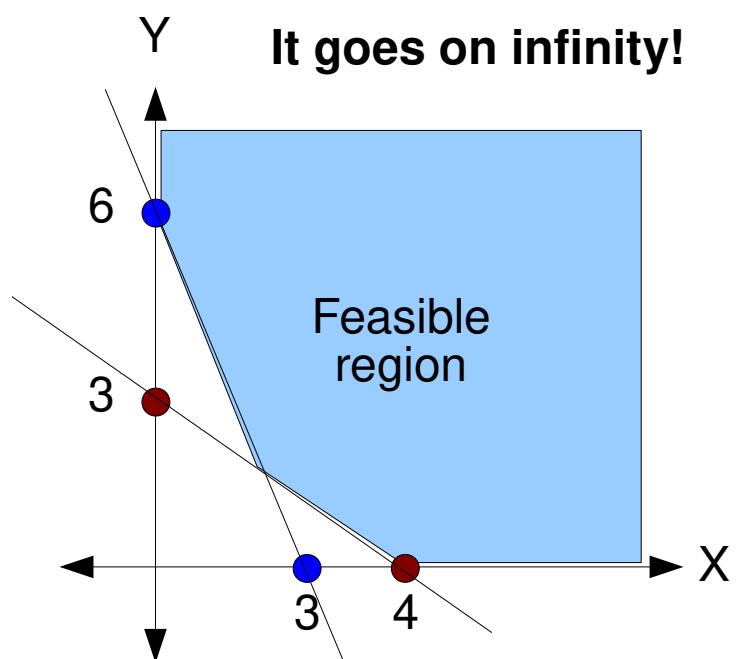
(4,0)  
(0,6)

And an additional one...

$$10X + 5Y = 30$$

$$15X + 20Y = 60$$

X = 2.4 and Y = 1.2 so we have (2.4, 1.2)



# Solving Linear Optimization Problems

- Second example:
  - Now we get back to the objective function (cost function) and solve it for the points we found!

Applying:

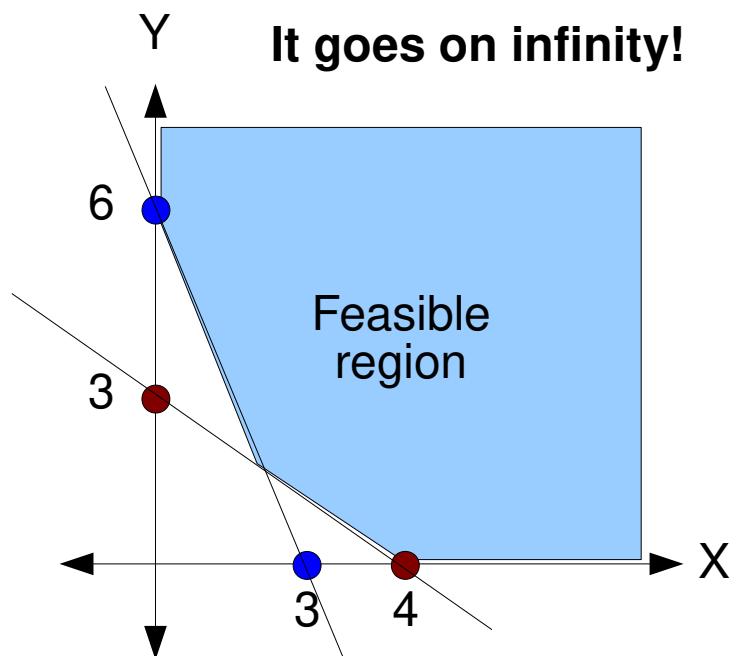
$$C(4,0) = 3.2$$

$$C(0,6) = 3$$

$$C(2.4, 1.2) = 2.52$$

So the minimal cost is at (2.4, 1.2) which is \$2.52 per serving!

So we should use 2.4 units of Brand X and 1.2 units of Brand Y to get the minimal cost!



# Solving Linear Optimization Problems

- Now, try to solve the following problem:
  - **Problem 1:** A refinery produces both gasoline and fuel oil and sells gasoline for \$1/litre and fuel oil for \$0.90/liter. The refinery can produce at most 600,000 litres a day, but must produce at least 2 litres of fuel oil for every litre of gasoline. At least 150,000 litres of fuel oil must be produced each day to meet current demands. How much of each type of fuel should be produced in order to maximize daily profits?
  - **Solution:** [https://www.youtube.com/watch?v=8AA\\_81xI3ik&index=30&list=WL](https://www.youtube.com/watch?v=8AA_81xI3ik&index=30&list=WL)

# Solving Linear Optimization Problems

- Now, try to solve the following problem:
  - **Problem 2:** Solve the problem:

Maximize  $C = x + y$   
subject to:

$$y \geq 0$$

$$x \geq 0$$

$$4x + 2y \leq 8$$

$$2x - y \leq 0$$

- **Solution:** <https://www.youtube.com/watch?v=z6UIXIXZ4yQ&list=WL&index=31>

# Solving Linear Optimization Problems

- Now, try to solve the following problem:
  - Problem 3:** Solve the problem:

Minimize  $C = 4x + 2y$

subject to:

$$-3x + 2y \leq 6$$

$$3x + y \leq 3$$

$$y \geq 0$$

- Solution:** <https://www.youtube.com/watch?v=l7brzjFhYEU&list=WL&index=28>

# **Primal and Dual Forms of Linear Problems**

# Primal and Dual Forms of Linear Problems

- After understanding how to solve linear optimization problems, we will now translate them to equivalent forms
  - We refer to the optimization problem we need to solve as **primal**
  - We can translate or rewrite it on another problem form:
    - Which is called the **dual problem** or **dual form**

# Primal and Dual Forms of Linear Problems

- After understanding how to solve linear optimization problems, we will now translate them to equivalent forms
  - We refer to the optimization problem we need to solve as **primal**
  - We can translate or rewrite it on another problem form:
    - Which is called the **dual problem** or **dual form**
- Why do we do that?
  - That **may** help us to **simplify the problem**
  - For instance, instead of having a two-variable problem we could formulate it in terms of only one variable

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

$$\text{Maximize } 3x + 6y + 2z$$

$$\text{subject to: } 3x + 4y + z \leq 2$$

$$x + 2y + 3z = 10$$

$$y \geq 0$$

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

$$\text{Maximize } 3x + 6y + 2z$$

$$\text{subject to: } 3x + 4y + z \leq 2$$

$$x + 2y + 3z = 10$$

$$y \geq 0$$

- We start building the dual form as follows:

If the primal problem is maximize then the dual is minimize (and vice-versa)

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

Maximize  $3x + 6y + 2z$

subject to:  $3x + 4y + z \leq 2$

$x + 2y + 3z = 10$

$y \geq 0$

- We start building the dual form as follows:

Minimize

2

10

**Constraint bound are part of the objective function**

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

$$\text{Maximize } 3x + 6y + 2z$$

$$\text{subject to: } 3x + 4y + z \leq 2$$

$$x + 2y + 3z = 10$$

$$y \geq 0$$

- We start building the dual form as follows:

$$\text{Minimize } 2\lambda_1 + 10\lambda_2$$

**Then we need other variables and we always sum the terms**

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

$$\text{Maximize } 3x + 6y + 2z$$

$$\text{subject to: } 3x + 4y + z \leq 2$$

$$x + 2y + 3z = 10$$

$$y \geq 0$$

- We start building the dual form as follows:

$$\text{Minimize } 2\lambda_1 + 10\lambda_2$$

**Now we need to build the constraints**

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

$$\begin{aligned} & \text{Maximize } 3x + 6y + 2z \\ \text{subject to: } & \boxed{3x + 4y + z \leq 2} \\ & \boxed{x + 2y + 3z = 10} \\ & y \geq 0 \end{aligned}$$

- We start building the dual form as follows:

$$\begin{aligned} & \text{Minimize } 2\lambda_1 + 10\lambda_2 \\ \text{subject to: } & \begin{matrix} 3 & 1 \\ 4 & 2 \\ 1 & 3 \end{matrix} \end{aligned}$$

**Take all primal constraints as columns**

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

$$\begin{aligned} & \text{Maximize } 3x + 6y + 2z \\ \text{subject to: } & \boxed{3x + 4y + z \leq 2} \\ & \boxed{x + 2y + 3z = 10} \\ & y \geq 0 \end{aligned}$$

- We start building the dual form as follows:

$$\begin{aligned} & \text{Minimize } 2\lambda_1 + 10\lambda_2 \\ \text{subject to: } & \begin{array}{ll} 3\lambda_1 & 1\lambda_2 \\ 4\lambda_1 & 2\lambda_2 \\ 1\lambda_1 & 3\lambda_2 \end{array} \end{aligned}$$

The first column is related to the first new variable  
and the second with the other variable

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

$$\begin{array}{ll} \text{Maximize} & 3x + 6y + 2z \\ \text{subject to:} & 3x + 4y + z \leq 2 \\ & x + 2y + 3z = 10 \\ & y \geq 0 \end{array}$$

- We start building the dual form as follows:

$$\begin{array}{llll} \text{Minimize} & 2\lambda_1 & + & 10\lambda_2 \\ \text{subject to:} & 3\lambda_1 & + & 1\lambda_2 & 3 \\ & 4\lambda_1 & + & 2\lambda_2 & 6 \\ & 1\lambda_1 & + & 3\lambda_2 & 2 \end{array}$$

The terms of the objective function will be constraint bounds

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

$$\text{Maximize } 3x + 6y + 2z$$

$$\begin{aligned}\text{subject to: } & 3x + 4y + z \leq 2 \\ & x + 2y + 3z = 10\end{aligned}$$

$$y \geq 0$$

- We start building the dual form as follows:

$$\text{Minimize } 2\lambda_1 + 10\lambda_2$$

$$\text{subject to: } 3\lambda_1 + 1\lambda_2 \quad 3$$

$$4\lambda_1 + 2\lambda_2 \quad 6$$

$$1\lambda_1 + 3\lambda_2 \quad 2$$

As we have two constraints in the primal form, we will have two variables in the dual form

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

$$\begin{aligned} & \text{Maximize } 3x + 6y + 2z \\ & \text{subject to: } 3x + 4y + z \leq 2 \\ & \quad x + 2y + 3z = 10 \\ & \quad y \geq 0 \end{aligned}$$

- We start building the dual form as follows:

$$\begin{aligned} & \text{Minimize } 2\lambda_1 + 10\lambda_2 \\ & \text{subject to: } \begin{array}{lll} 3\lambda_1 + 1\lambda_2 & & 3 \\ 4\lambda_1 + 2\lambda_2 & & 6 \\ 1\lambda_1 + 3\lambda_2 & & 2 \end{array} \end{aligned}$$

We need to verify the operators we have in those constraints to define constraints for the two dual variables

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

$$\begin{aligned} & \text{Maximize } 3x + 6y + 2z \\ & \text{subject to: } 3x + 4y + z \leq 2 \\ & \quad x + 2y + 3z = 10 \\ & \quad y \geq 0 \end{aligned}$$

- We start building the dual form as follows:

$$\begin{aligned} & \text{Minimize } 2\lambda_1 + 10\lambda_2 \\ & \text{subject to: } \begin{array}{lll} 3\lambda_1 + 1\lambda_2 & & 3 \\ 4\lambda_1 + 2\lambda_2 & & 6 \\ 1\lambda_1 + 3\lambda_2 & & 2 \end{array} \end{aligned}$$

The first constraint corresponds to  $\lambda_1$ , and the second to  $\lambda_2$

# Primal and Dual Forms of Linear Problems

- Let's understand how those operators will define bounds for  $\lambda_1$  and  $\lambda_2$ :

Maximize

Minimize

$\leq$

variable must assume  
non-negative values

$=$

Any

variable must assume  
non-negative values

$\geq$

Any

$=$

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

$$\text{Maximize } 3x + 6y + 2z$$

$$\text{subject to: } 3x + 4y + z \leq 2$$

$$x + 2y + 3z = 10$$

$$y \geq 0$$

- We start building the dual form as follows:

$$\text{Minimize } 2\lambda_1 + 10\lambda_2$$

$$\text{subject to: } 3\lambda_1 + 1\lambda_2 \quad 3$$

$$4\lambda_1 + 2\lambda_2 \quad 6$$

$$1\lambda_1 + 3\lambda_2 \quad 2$$

The first operator “if less than or equal”, therefore  $\lambda_1$  will be bounded as follows... let's look at the table

# Primal and Dual Forms of Linear Problems

- Let's understand how those operators will define bounds for  $\lambda_1$  and  $\lambda_2$ :

Maximize

Minimize

$\leq$

variable must assume  
non-negative values

$=$

Any

variable must assume  
non-negative values

$\geq$

Any

$=$

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

$$\begin{aligned} & \text{Maximize } 3x + 6y + 2z \\ & \text{subject to: } 3x + 4y + z \leq 2 \\ & \quad x + 2y + 3z = 10 \\ & \quad y \geq 0 \end{aligned}$$

- We start building the dual form as follows:

$$\begin{aligned} & \text{Minimize} && 2\lambda_1 + 10\lambda_2 \\ & \text{subject to:} && \begin{array}{lll} 3\lambda_1 + 1\lambda_2 & & 3 \\ 4\lambda_1 + 2\lambda_2 & & 6 \\ 1\lambda_1 + 3\lambda_2 & & 2 \\ \lambda_1 \geq 0 & & \end{array} \end{aligned}$$

$\lambda_1$  must assume non-negative values

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

$$\text{Maximize } 3x + 6y + 2z$$

$$\text{subject to: } 3x + 4y + z \leq 2$$

$$x + 2y + 3z = 10$$

$$y \geq 0$$

- We start building the dual form as follows:

$$\text{Minimize } 2\lambda_1 + 10\lambda_2$$

$$\text{subject to: } 3\lambda_1 + 1\lambda_2 \quad 3$$

$$4\lambda_1 + 2\lambda_2 \quad 6$$

$$1\lambda_1 + 3\lambda_2 \quad 2$$

$$\lambda_1 \geq 0$$

What about the second variable?

# Primal and Dual Forms of Linear Problems

- Let's understand how those operators will define bounds for  $\lambda_1$  and  $\lambda_2$ :

Maximize

Minimize

$\leq$

variable must assume  
non-negative values

$=$

Any

variable must assume  
non-negative values

$\geq$

Any

$=$

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

$$\text{Maximize } 3x + 6y + 2z$$

$$\text{subject to: } 3x + 4y + z \leq 2$$

$$x + 2y + 3z = 10$$

$$y \geq 0$$

- We start building the dual form as follows:

$$\text{Minimize } 2\lambda_1 + 10\lambda_2$$

$$\text{subject to: } 3\lambda_1 + 1\lambda_2 \quad 3$$

$$4\lambda_1 + 2\lambda_2 \quad 6$$

$$1\lambda_1 + 3\lambda_2 \quad 2$$

$$\lambda_1 \geq 0$$

**It can assume any value... That is why we do not define bounds for  $\lambda_2$**

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

Maximize

Now we need to define the constraint operators!

How would you define them?

- We start building the **dual** form using the following **primal**:

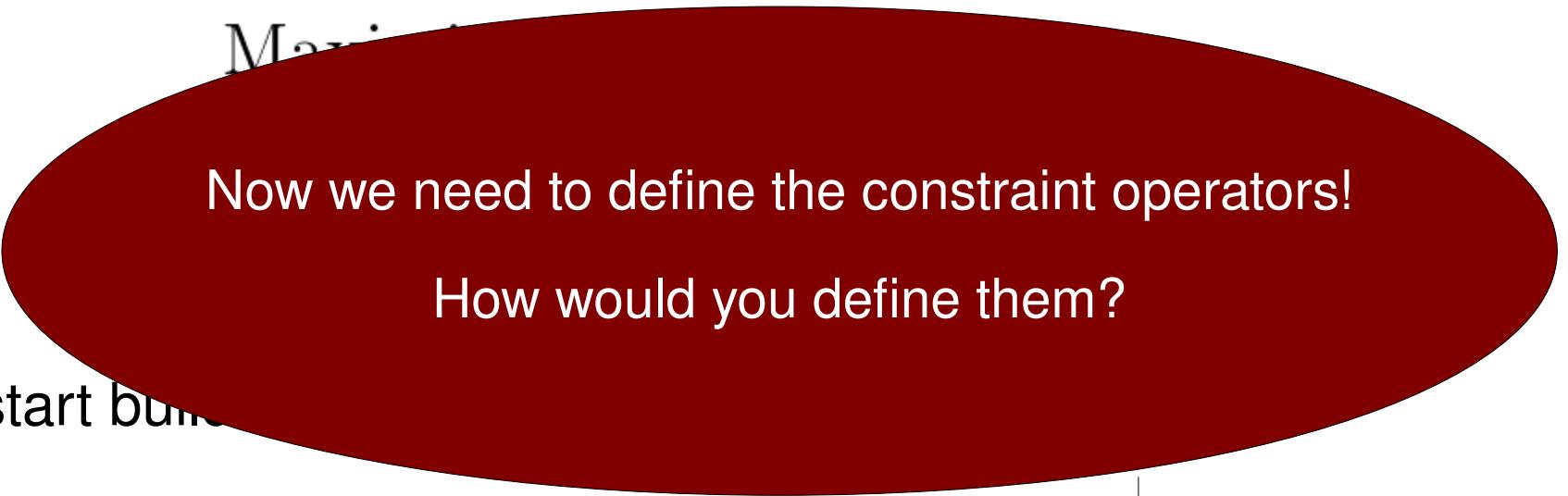
Minimize  $2 \lambda_1 + 10 \lambda_2$

subject to:  $3\lambda_1 + 1\lambda_2$

$$4\lambda_1 + 2\lambda_2$$

$$1\lambda_1 + 3\lambda_2$$

$$\lambda_1 \geq 0$$



A vertical arrow points from the text "How would you define them?" down to this table.

3
6
2

**It can assume any value... That is why we do not define bounds for  $\lambda_2$**

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

$$\begin{aligned} & \text{Maximize } 3x + 6y + 2z \\ & \text{subject to: } 3x + 4y + z \leq 2 \\ & \quad x + 2y + 3z = 10 \\ & \quad y \geq 0 \end{aligned}$$

- We start building the dual form as follows:

$$\begin{aligned} & \text{Minimize} && 2\lambda_1 + 10\lambda_2 \\ & \text{subject to:} && \begin{array}{lll} 3\lambda_1 + 1\lambda_2 & & 3 \\ 4\lambda_1 + 2\lambda_2 & & 6 \\ 1\lambda_1 + 3\lambda_2 & & 2 \\ \lambda_1 \geq 0 & & \end{array} \end{aligned}$$

**First of all, in the primal problem, variables x and z do not have bounds!!! What does that mean? Back to our table...**

# Primal and Dual Forms of Linear Problems

- Let's understand how those operators define bounds

Maximize

Minimize

$\leq$

variable must assume  
non-negative values

$=$

Any

variable must assume  
non-negative values

$\geq$

Any

$=$

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

$$\begin{aligned} & \text{Maximize } 3x + 6y + 2z \\ & \text{subject to: } 3x + 4y + z \leq 2 \\ & \quad x + 2y + 3z = 10 \\ & \quad y \geq 0 \end{aligned}$$

- We start building the dual form as follows:

$$\begin{aligned} & \text{Minimize } 2\lambda_1 + 10\lambda_2 \\ & \text{subject to: } \begin{array}{rcl} 3\lambda_1 + 1\lambda_2 & = & 3 \\ 4\lambda_1 + 2\lambda_2 & = & 6 \\ 1\lambda_1 + 3\lambda_2 & = & 2 \end{array} \\ & \quad \lambda_1 \geq 0 \end{aligned}$$

Observe each constraint correspond to one variable in the primal form

So the corresponding constraints will consider operator “equal to”

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

$$\text{Maximize } 3x + 6y + 2z$$

$$\text{subject to: } 3x + 4y + z \leq 2$$

$$x + 2y + 3z = 10$$

$$y \geq 0$$

- We start building the dual form as follows:

$$\text{Minimize } 2\lambda_1 + 10\lambda_2$$

$$\text{subject to: } 3\lambda_1 + 1\lambda_2 = 3$$

$$4\lambda_1 + 2\lambda_2 = 6$$

$$1\lambda_1 + 3\lambda_2 = 2$$

$$\lambda_1 \geq 0$$

So, what about the constraint associated to  $y$  ??? Back to our table...

# Primal and Dual Forms of Linear Problems

- Let's understand how those operators define bounds

Maximize

Minimize

$\leq$

variable must assume  
non-negative values

$=$

Any

variable must assume  
non-negative values

$\geq$

Any

$=$

# Primal and Dual Forms of Linear Problems

- We will start building the **dual** form using the following **primal**:

$$\text{Maximize } 3x + 6y + 2z$$

$$\text{subject to: } 3x + 4y + z \leq 2$$

$$x + 2y + 3z = 10$$

$$y \geq 0$$

- We start building the dual form as follows:

$$\text{Minimize } 2\lambda_1 + 10\lambda_2$$

$$\text{subject to: } 3\lambda_1 + 1\lambda_2 = 3$$

$$4\lambda_1 + 2\lambda_2 \geq 6$$

$$1\lambda_1 + 3\lambda_2 = 2$$

$$\lambda_1 \geq 0$$

The constraint must be greater or equal to the bound

# Primal and Dual Forms of Linear Problems

- From the **primal** form:

$$\begin{aligned} & \text{Maximize } 3x + 6y + 2z \\ & \text{subject to: } 3x + 4y + z \leq 2 \\ & \quad x + 2y + 3z = 10 \\ & \quad y \geq 0 \end{aligned}$$

- We have finally found the **dual** form:

$$\begin{aligned} & \text{Minimize } 2\lambda_1 + 10\lambda_2 \\ & \text{subject to: } 3\lambda_1 + \lambda_2 = 3 \\ & \quad 4\lambda_1 + 2\lambda_2 \geq 6 \\ & \quad \lambda_1 + 3\lambda_2 = 2 \\ & \quad \lambda_1 \geq 0 \end{aligned}$$

# Primal and Dual Forms of Linear Problems

- From the **primal** form:

$$\text{Maximize } 3x + 6y + 2z$$

$$\text{subject to: } 3x + 4y + z \leq 2$$

- We have

The whole idea behind this transformation  
is to find equivalent problems!

$$x\wedge_1 + z\wedge_2 \geq 6$$

$$\lambda_1 + 3\lambda_2 = 2$$

$$\lambda_1 \geq 0$$

# Primal and Dual Forms of Linear Problems

- Now, if the **primal** form is

$$\text{Minimize } 2\lambda_1 + 10\lambda_2$$

$$\text{subject to: } 3\lambda_1 + \lambda_2 = 3$$

$$4\lambda_1 + 2\lambda_2 \geq 6$$

$$\lambda_1 + 3\lambda_2 = 2$$

$$\lambda_1 \geq 0$$

- What is the **dual** form?

- Solve it!

- Observe we will get a form with more variables

# Primal and Dual Forms of Linear Problems

- Exercise:
  - Find the **dual** form for the **primal** problem:

Maximize  $3x + 2y$

Subject to:

$$x - 4y = 4$$

$$3x - 2y \leq 1$$

$$5x - 8y \leq -7$$

$$x \geq 0$$

- Afterwards confirm your solution with:

<https://www.youtube.com/watch?v=KMmgF3ZaBRE&index=36&list=WL>

# **Graphical interpretation of Primal and Dual forms**

# Graphical Interpretation

- Consider the problem in **primal** form:

Maximize  $6x + 4y$

Subject to:

$$x + y \leq 2$$

$$2x - y \leq 2$$

$$x \geq 0, y \geq 0$$

- First of all find the **dual** form as exercise...

# Graphical Interpretation

- Consider the problem in **primal** form:

$$\text{Maximize } 6x + 4y$$

Subject to:

$$x + y \leq 2$$

$$2x - y \leq 2$$

$$x \geq 0, y \geq 0$$

- The **dual** form is:

$$\text{Minimize } 2k + 2z$$

Subject to:

$$1k + 2z \geq 6$$

$$1k - 1z \geq 4$$

$$k \geq 0, z \geq 0$$

# Graphical Interpretation

- Consider the problem in **primal** form:

$$\text{Maximize } 6x + 4y$$

Subject to:

$$x + 2y \leq 2$$

- The

Let's graph both!

$$1k + 2z \geq 6$$

$$1k - 1z \geq 4$$

$$k \geq 0, z \geq 0$$

# Graphical Interpretation

- Plotting the **primal** form:

Maximize  $6x + 4y$

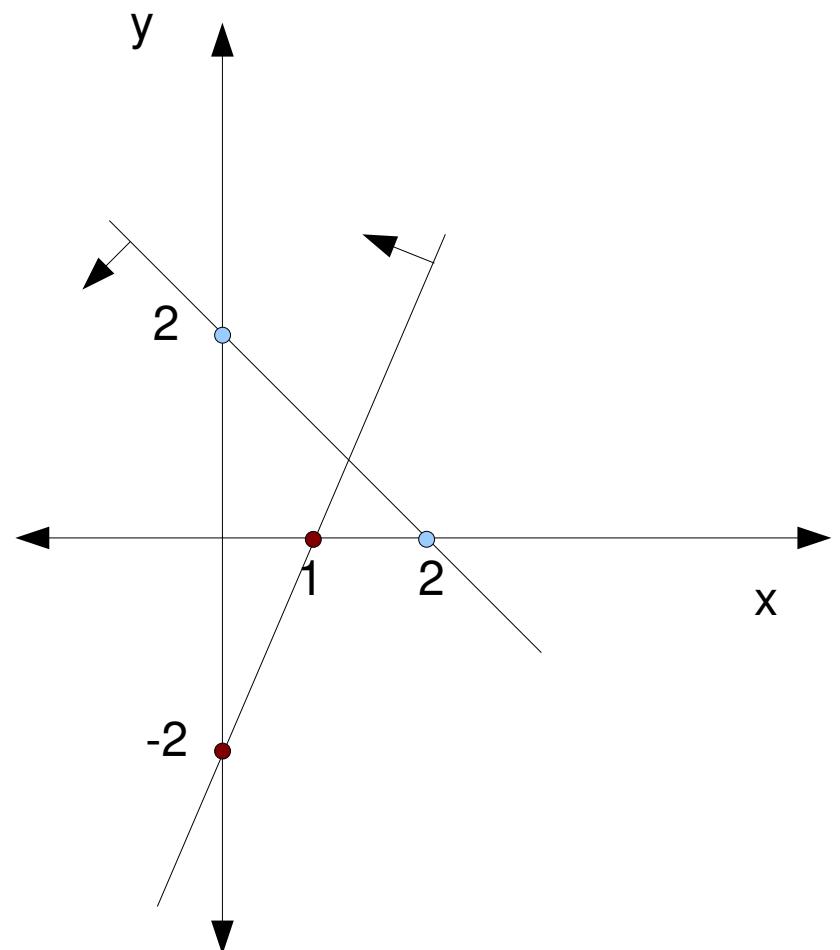
Subject to:

$$x + y \leq 2$$

$$2x - y \leq 2$$

$$x \geq 0, y \geq 0$$

Solving the two constraints for equality we have...



# Graphical Interpretation

- Plotting the **primal** form:

Maximize  $6x + 4y$

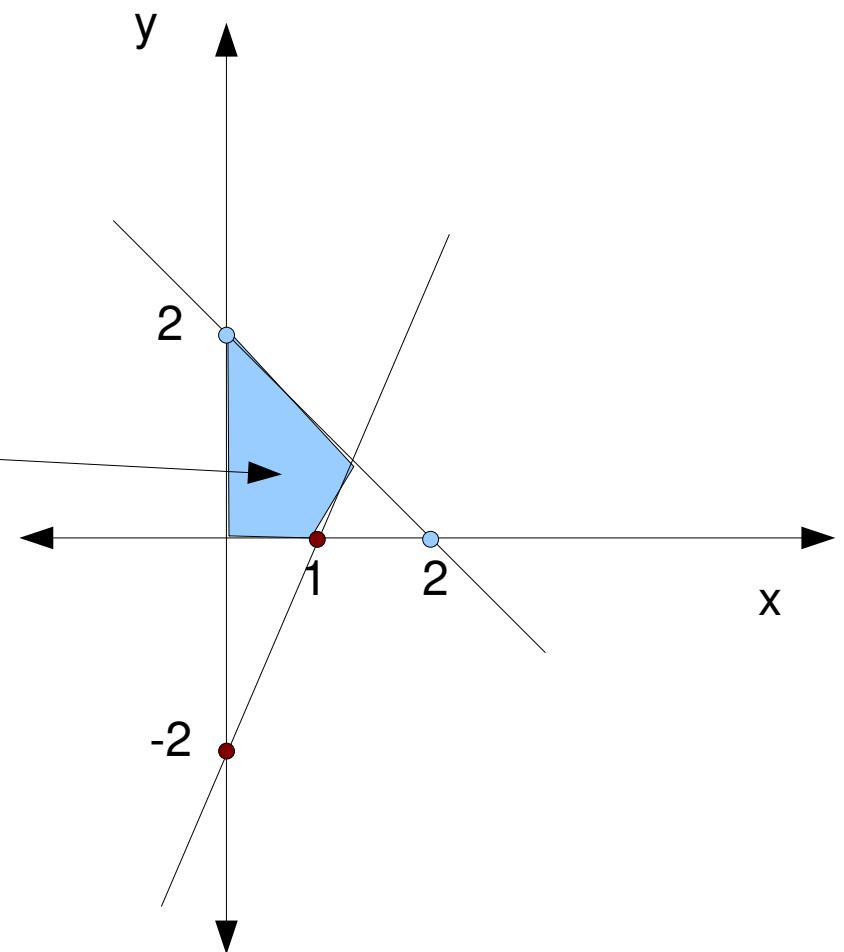
Subject to:

$$x + y \leq 2$$

$$2x - y \leq 2$$

$$x \geq 0, y \geq 0$$

**Feasible region**



# Graphical Interpretation

- Plotting the **primal** form:

Maximize  $6x + 4y$

Subject to:

$$x + y \leq 2$$

$$2x - y \leq 2$$

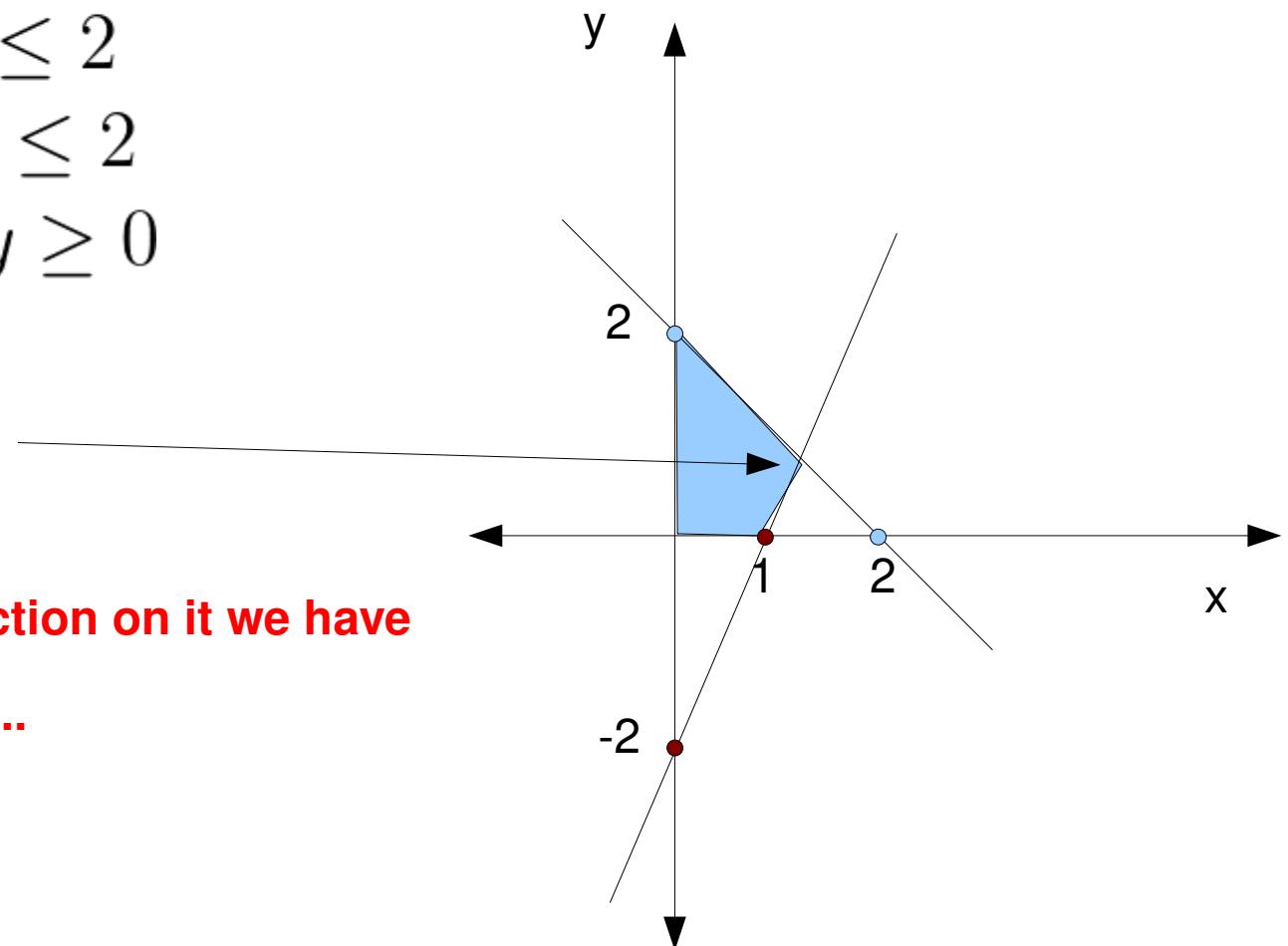
$$x \geq 0, y \geq 0$$

We will verify this corner

Which is at  $(4/3, 2/3)$

Applying the objective function on it we have

$$6(4/3) + 4(2/3) = 10.66\dots$$



# Graphical Interpretation

- Plotting the **dual form**:

Minimize  $2k + 2z$

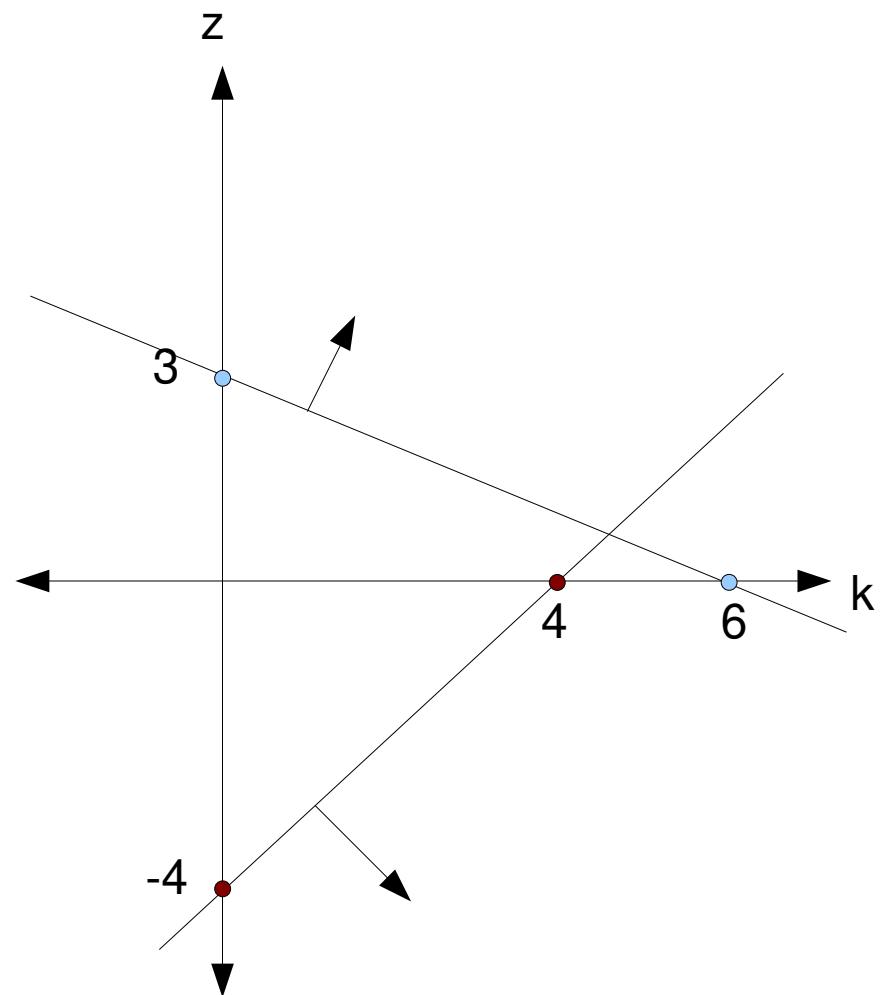
Subject to:

$$1k + 2z \geq 6$$

$$1k - 1z \geq 4$$

$$k \geq 0, z \geq 0$$

Solving the two constraints for equality we have...



# Graphical Interpretation

- Plotting the **dual form**:

Minimize  $2k + 2z$

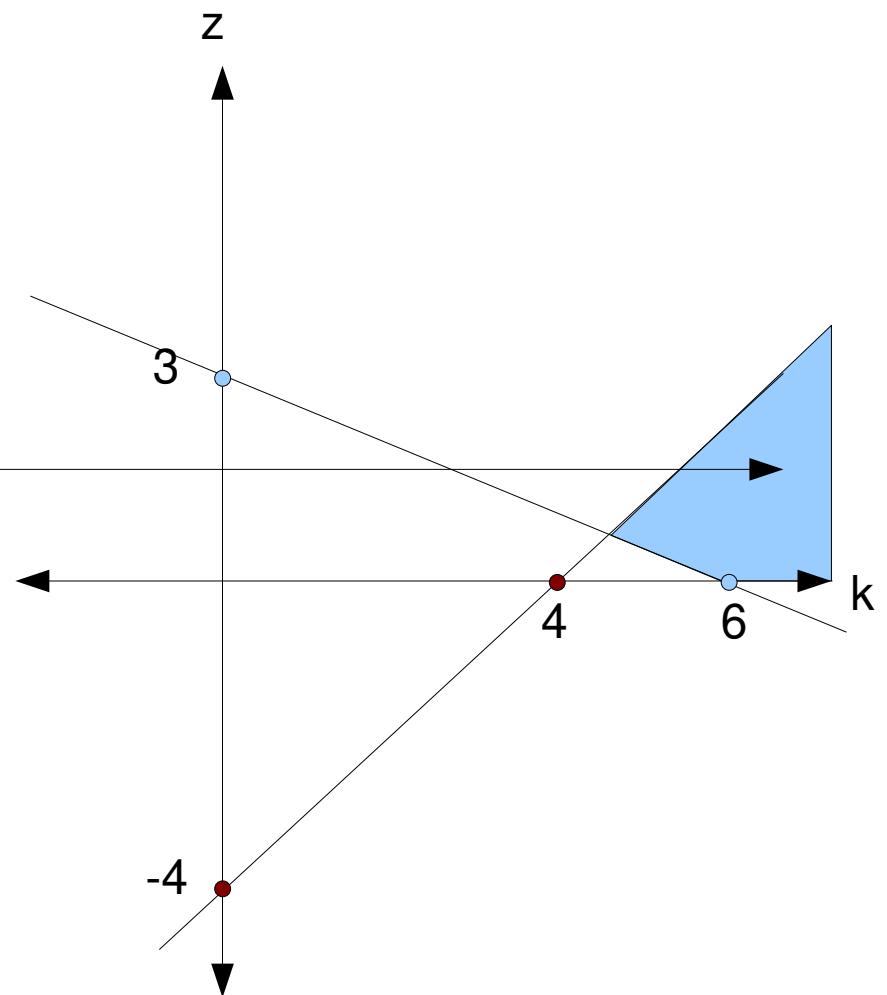
Subject to:

$$1k + 2z \geq 6$$

$$1k - 1z \geq 4$$

$$k \geq 0, z \geq 0$$

**Feasible region**



# Graphical Interpretation

- Plotting the **dual form**:

Minimize  $2k + 2z$

Subject to:

$$1k + 2z \geq 6$$

$$1k - 1z \geq 4$$

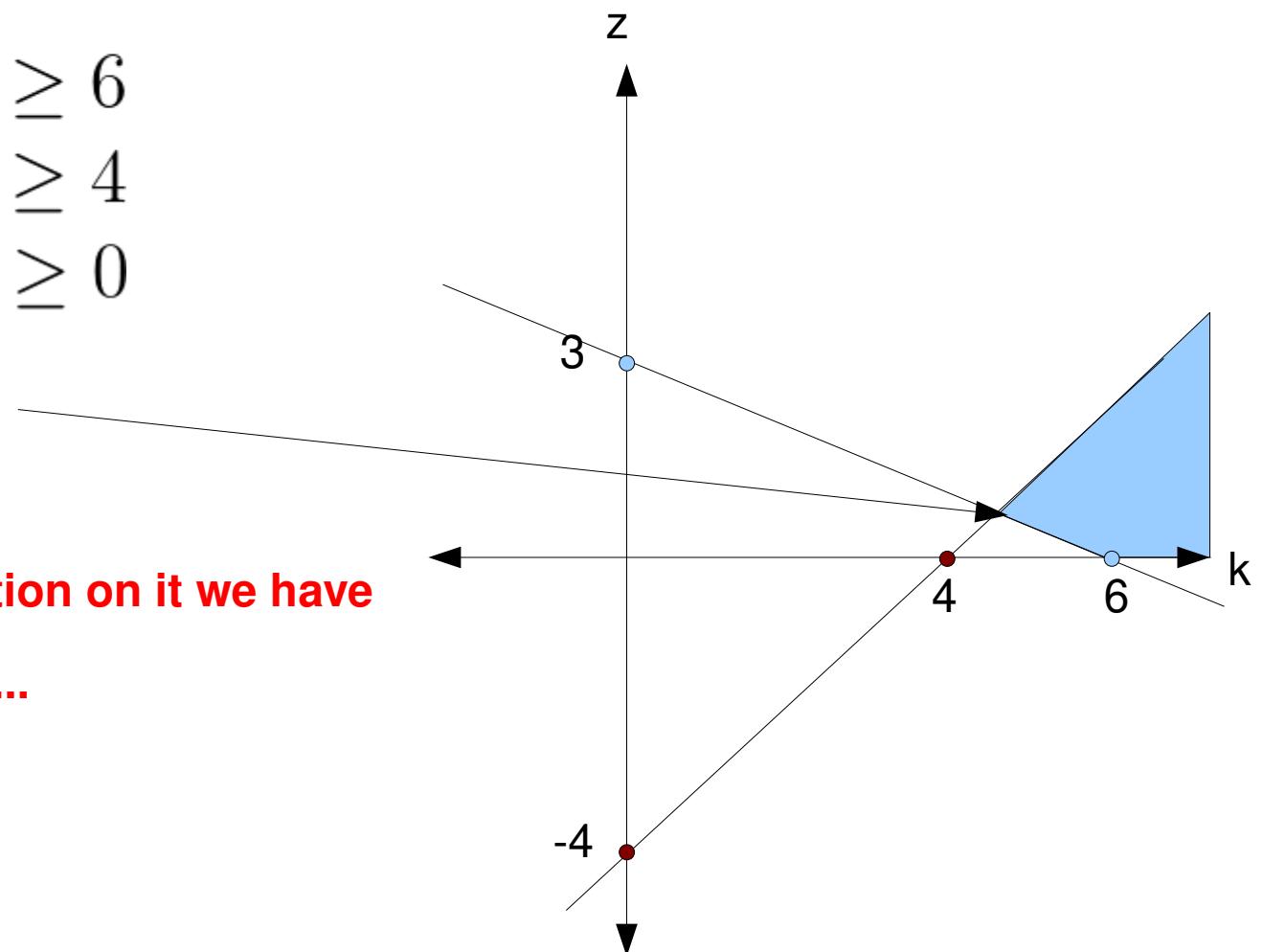
$$k \geq 0, z \geq 0$$

We will verify this corner

Which is at  $(14/3, 2/3)$

Applying the objective function on it we have

$$2(14/3) + 2(2/3) = 10.66\dots$$



# Graphical Interpretation

- Plotting the **dual form**:

Minimize  $2k + 2z$

Subject to:

$1k$

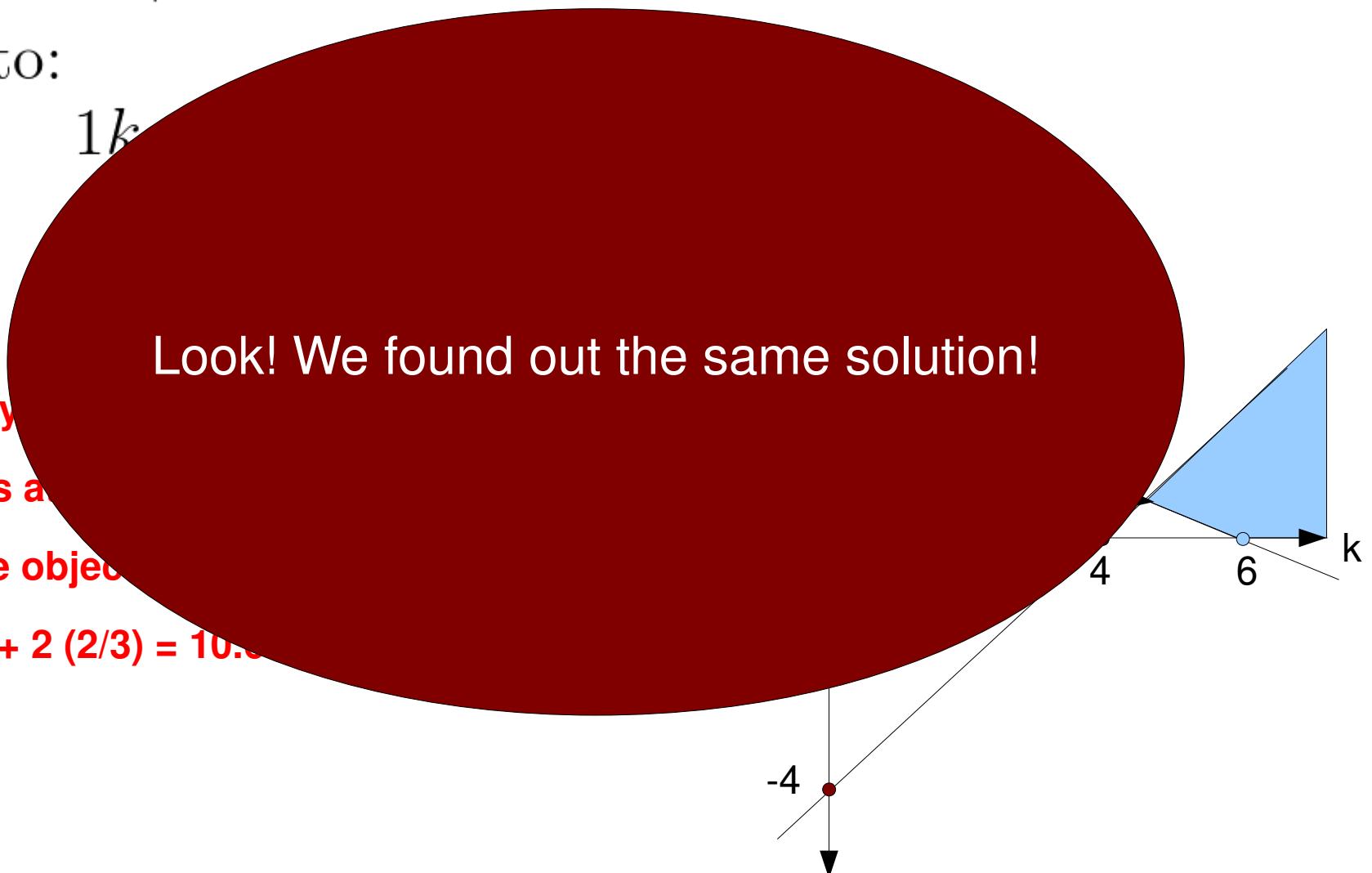
Look! We found out the same solution!

We will verify

Which is a

Applying the objec

$$2(14/3) + 2(2/3) = 10.67$$



# Graphical Interpretation

- Observe one solution come from one side (in terms of the feasible set) while the other comes from the other side
  - This is what changes from primal to dual when it comes to constraints
- Besides we did not plot the third axis (to see the objective function):
  - We would notice a primal maximum is translated to a dual minimum and vice-versa
- Another important point:
  - Observe **graph axes are different from primal to dual**

# **Some properties of the Primal and Dual forms for Linear problems**

# Properties for Primal/Dual Linear Problems

- Consider the primal and the dual forms:

Maximize  $F(x)$

Subject to:  $g(x) \leq b$

Minimize  $H(\lambda)$

Subject to:  $q(x) = c, \lambda \geq 0$

- The following properties are hold for linear problems:

- $F(x) \leq H(\lambda)$

- if the feasible set for  $x$  is not empty and  $x^*$  is the solution for the primal problem, then there is  $\lambda^*$  for the dual problem in which:

$$F(x^*) = H(\lambda^*)$$

Dual produces the same as Primal (Strong Duality)

# **Using Lagrange to Build the Dual Form for Linear Problems**

# Primal and Dual Forms of Linear Problems

- The approach we have seen before is actually a practical result from Lagrange for building the dual form using the primal
  - Now we will see a bit about the math involved

# Primal and Dual Forms of Linear Problems

- So how to build the **dual** form?
  - For example, consider the **primal** problem:

Maximize  $F(x)$   
subject to  $g(x) \leq b$

# Primal and Dual Forms of Linear Problems

- So how to build the **dual** form?
  - For example, consider the **primal** problem:

Maximize  $F(x)$   
subject to  $g(x) \leq b$

- We can build the associated **dual** problem by solving the Lagrangian:

Minimize  $L(x, \lambda) = F(x) - \lambda[g(x) - b]$   
subject to:

$$\begin{aligned}\frac{\partial L}{\partial x} &= 0 \\ \lambda &\geq 0 \\ x &\in D\end{aligned}$$

# Primal and Dual Forms of Linear Problems

- To exercise this formal framework, we will just apply it:
  - Build the **dual form** for the problem:

$$\begin{aligned} & \text{Maximize } c^T x \\ & \text{subject to } Ax \leq b \end{aligned}$$

# Primal and Dual Forms of Linear Problems

- To exercise this formal framework, we will just apply it:
  - Build the **dual form** for the problem:

$$\begin{aligned} & \text{Maximize } c^T x \\ & \text{subject to } Ax \leq b \end{aligned}$$

- First, we have to build the Lagragian:

$$\begin{aligned} & \text{Minimize } L(x, \lambda) = c^T x - \lambda(Ax - b) \\ & \text{subject to} \end{aligned}$$

$$\begin{aligned} \frac{\partial L}{\partial x} &= 0 \\ \lambda &\geq 0 \end{aligned}$$

$x \in D$  i.e., in the domain of the original  
(primal) objective function.

# Primal and Dual Forms of Linear Problems

- To exercise this formal framework, we will just apply it:
  - Now we find the gradient:

$$\frac{\partial L}{\partial x} = \frac{\partial}{\partial x} c^T x - \lambda(Ax - b) = c - A^T \lambda = 0$$

Thus:  $A^T \lambda = c$

- Now we can use this equation to substitute terms in the Lagrangian

# Primal and Dual Forms of Linear Problems

- To exercise this formal framework, we will just apply it:
  - Proceeding with the substitution we have:

$$L(x, \lambda) = c^T x - \lambda(Ax - b)$$

We can solve as follows:

$$L(x, \lambda) = c^T x - A^T \lambda x + \lambda b$$

As we found:  $A^T \lambda = c$

Substituting we have:

$$L(x, \lambda) = c^T x - c^T x + \lambda b$$

# Primal and Dual Forms of Linear Problems

- To exercise this formal framework, we will just apply it:
  - Proceeding with the substitution we have the **dual** form:

From:  $L(x, \lambda) = c^T x - c^T x + \lambda b$

We finally have the dual form:

Minimize  $\lambda b$   
subject to:

$$A^T \lambda = c$$

$$\lambda \geq 0$$

**Observe: Now the problem is no more in terms of x but in terms of lambda.**

# Primal and Dual Forms of Linear Problems

- Another exercise:
  - So what happens if we consider the **primal** problem as?

Minimize  $\lambda b$   
subject to:

$$\begin{aligned} A^T \lambda &= c \\ \lambda &\geq 0 \end{aligned}$$

- Try to solve it!

# Primal and Dual Forms of Linear Problems

- Another exercise:
  - First of all, we will build the Lagrangian for:

Minimize  $\lambda b$   
subject to:

$$A^T \lambda = c$$
$$\lambda \geq 0$$

- Which is:

$$L = \lambda b - x(A^T \lambda - c)$$

# Primal and Dual Forms of Linear Problems

- Another exercise:
  - Now we solve the Lagrangian:

$$L(\lambda, x) = \lambda b - x(A^T \lambda - c)$$

- Thus we find:

$$\frac{\partial L}{\partial \lambda} = b - A^T x = 0$$

# Primal and Dual Forms of Linear Problems

- Another exercise:
  - Now we substitute the terms in the Lagrangian:

$$L(\lambda, x) = \lambda b - x(A^T \lambda - c)$$

$$L(\lambda, x) = \lambda b - A^T x \lambda + c^T x$$

- Applying the derivative we found:

$$\frac{\partial L}{\partial \lambda} = b - A^T x = 0$$

- We then have:

$$L(\lambda, x) = \lambda b - \lambda b + c^T x = c^T x$$

# Primal and Dual Forms of Linear Problems

- Another exercise:
  - Consequently our problem is:

$$\text{Maximize } c^T x$$

- But what about the constraints?

# Primal and Dual Forms of Linear Problems

- Another exercise:
  - Consequently our problem is:

Maximize  $c^T x$

- But what about the constraints?
  - First of all remember the table we have seen before!

**Let's see the table...**

# Primal and Dual Forms of Linear Problems

- Table:

Maximize	Minimize
$\leq$	variable must assume non-negative values
$=$	Any
variable must assume non-negative values	$\geq$
Any	$=$

# Primal and Dual Forms of Linear Problems

- Another exercise:
  - Consequently our problem is:

$$\text{Maximize } c^T x$$

- But what about the constraints?
  - Now remember the constraints of the **primal** form:

$$\begin{aligned} \text{Minimize } & \lambda b \\ \text{subject to: } & \end{aligned}$$

$$\begin{aligned} A^T \lambda &= c \\ \lambda &\geq 0 \end{aligned}$$

Observe this!

So the first constraint  
will also have this  
signal!

# Primal and Dual Forms of Linear Problems

- Table:

Maximize

Minimize

$\leq$

variable must assume  
non-negative values

$=$

Any

variable must assume  
non-negative values

$\geq$

Any

$=$

# Primal and Dual Forms of Linear Problems

- Another exercise:
  - Consequently our problem is:

$$\text{Maximize } c^T x$$

- But what about the constraints?
  - So we will have the following constraint:
    - From the form:

- To:  $\frac{\partial L}{\partial \lambda} = b - A^T x = 0$

$$b - A^T x \geq 0$$

# Primal and Dual Forms of Linear Problems

- Another exercise:
  - Consequently our problem is:

$$\text{Maximize } c^T x$$

- But what about the constraints?
  - So we will have the following constraint:
    - Finally we have:

$$A^T x \leq b$$

- Observe we went back to the first problem form!

**Now we see both forms side to side**

- First problem form:

$$\begin{aligned} & \text{Maximize } c^T x \\ & \text{subject to } Ax \leq b \end{aligned}$$

- Second problem form:

$$\begin{aligned} & \text{Minimize } \lambda b \\ & \text{subject to:} \\ & \quad A^T \lambda = c \\ & \quad \lambda \geq 0 \end{aligned}$$

- What do you notice?

- First problem form:

Maximize  $c^T x$

- Second

The Lagrange multiplier of the first becomes the main variable for the second and vice-versa

$$\lambda \geq 0$$

- What do you notice?

- Look at the Lagrangians:

Maximize  $c^T x$   
subject to  $Ax \leq b$

$$L(x, \lambda) = c^T x - \lambda(Ax - b)$$

Minimize  $\lambda b$   
subject to:

$$L(\lambda, x) = \lambda b - x(A^T \lambda - c)$$

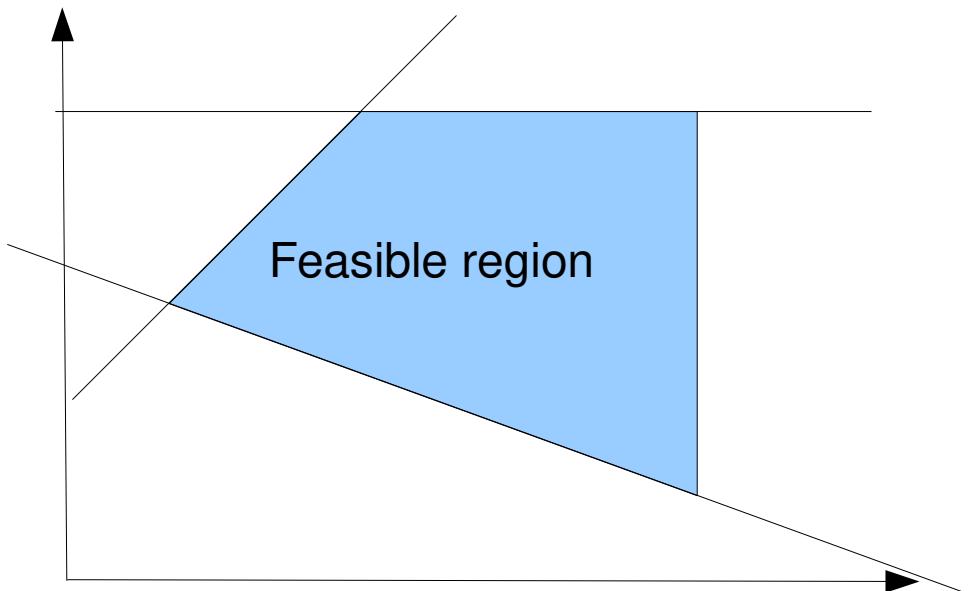
$$\begin{aligned}A^T \lambda &= c \\ \lambda &\geq 0\end{aligned}$$

# **Simplex: Algorithm to Solve Linear Optimization Problems**

- After understanding how to manually solve linear optimization problems, we will now solve them algorithmically
  - There is a method for solving **Linear Optimization Problems** which is called the **Simplex Method**
  - It solves linear problems in the form of **maximization** and constraints of the form  $\leq$  (or minimization and constraints of the form  $\geq$ )

Intuitively we can see the Simplex Method as an approach to visit each corner defined by constraints

Then it computes the objective function for every corner and decides on which is the solution



- We will consider a problem to understand how Simplex works:
  - Suppose a company manufactures different electronic components for computers
  - Component A requires 2 hours of fabrication and 1 hour of assembly
  - Component B requires 3 hours of fabrication and 1 hour of assembly
  - Component C requires 2 hours of fabrication and 2 hours of assembly
  - The company has up to 1,000 hours available for fabrication and 800 hours of assembly time a week
  - If the profit on each component A, B and C is respectively \$7, \$8 and \$10, how many of each component should be produced to maximize profit?

- We will start by defining the variables:
  - $X_1$  will be the number of components of type A
  - $X_2$  will be the number of components of type B
  - $X_3$  will be the number of components of type C

- We will start by defining the variables:
  - $X_1$  will be the number of components of type A
  - $X_2$  will be the number of components of type B
  - $X_3$  will be the number of components of type C
- Then we need to have the objective function to be maximized:

$$\text{Maximize } P = 7x_1 + 8x_2 + 10x_3$$

- We will start by defining the variables:
  - $X_1$  will be the number of components of type A
  - $X_2$  will be the number of components of type B
  - $X_3$  will be the number of components of type C
- Then we need to have the objective function to be maximized:

$$\text{Maximize } P = 7x_1 + 8x_2 + 10x_3$$

**Now we need the constraints!**

- We will start by defining the variables:
  - $X_1$  will be the number of components of type A
  - $X_2$  will be the number of components of type B
  - $X_3$  will be the number of components of type C
- Then we need to have the objective function to be maximized:

Maximize  $P = 7x_1 + 8x_2 + 10x_3$   
subject to:

$$x_1, x_2, x_3 \geq 0$$

**First of all, we cannot fabricate negative numbers of A, B and C**

- Then we need to have the objective function to be maximized:

Maximize  $P = 7x_1 + 8x_2 + 10x_3$   
subject to:

$$x_1, x_2, x_3 \geq 0$$

**Now we need to think about other constraints**

- Then we need to have the objective function to be maximized:

Maximize  $P = 7x_1 + 8x_2 + 10x_3$   
subject to:

$$x_1, x_2, x_3 \geq 0$$

**Fabrication constraint**

$$2x_1 + 3x_2 + 2x_3 \leq 1000$$

**Assembly constraint**

$$x_1 + x_2 + 2x_3 \leq 800$$

- Finally we have built the **linear optimization problem** to be solved:

Maximize  $P = 7x_1 + 8x_2 + 10x_3$

subject to:

$$2x_1 + 3x_2 + 2x_3 \leq 1000$$

$$x_1 + x_2 + 2x_3 \leq 800$$

$$x_1, x_2, x_3 \geq 0$$

- To apply Simplex we will first make the objective function

Maximize  $P = 7x_1 + 8x_2 + 10x_3$   
subject to:

$$\begin{aligned}2x_1 + 3x_2 + 2x_3 &\leq 1000 \\x_1 + x_2 + 2x_3 &\leq 800 \\x_1, x_2, x_3 &\geq 0\end{aligned}$$

- equals zero:

$$-7x_1 - 8x_2 - 10x_3 + P = 0$$

- Next we take the inequalities

Maximize  $P = 7x_1 + 8x_2 + 10x_3$   
subject to:

$$\begin{aligned}2x_1 + 3x_2 + 2x_3 &\leq 1000 \\x_1 + x_2 + 2x_3 &\leq 800 \\x_1, x_2, x_3 &\geq 0\end{aligned}$$

- And introduce **slack** variables (the idea is that this picks up the slack in inequality and converts them to equalities):

$$\begin{aligned}-7x_1 - 8x_2 - 10x_3 + P &= 0 \\2x_1 + 3x_2 + 2x_3 + S_1 &= 1000 \\x_1 + x_2 + 2x_3 + S_2 &= 800\end{aligned}$$

- From this:

$$-7x_1 - 8x_2 - 10x_3 + P = 0$$

$$2x_1 + 3x_2 + 2x_3 + S_1 = 1000$$

$$x_1 + x_2 + 2x_3 + S_2 = 800$$

- We have to build the **Simplex Tableau** by first labeling our variables as columns

$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
-------	-------	-------	-------	-------	-----	---------

**Then we build the Tableau with the constraints first**

- From this:

First the constraints

$$\left\{ \begin{array}{l} -7x_1 - 8x_2 - 10x_3 + P = 0 \\ 2x_1 + 3x_2 + 2x_3 + S_1 = 1000 \\ x_1 + x_2 + 2x_3 + S_2 = 800 \end{array} \right.$$

- We have to build the **Simplex Tableau** by first labeling our variables as columns

$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
2	3	2	1	0	0	1000
1	1	2	0	1	0	800
-7	-8	-10	0	0	1	0

The objective function is the last!

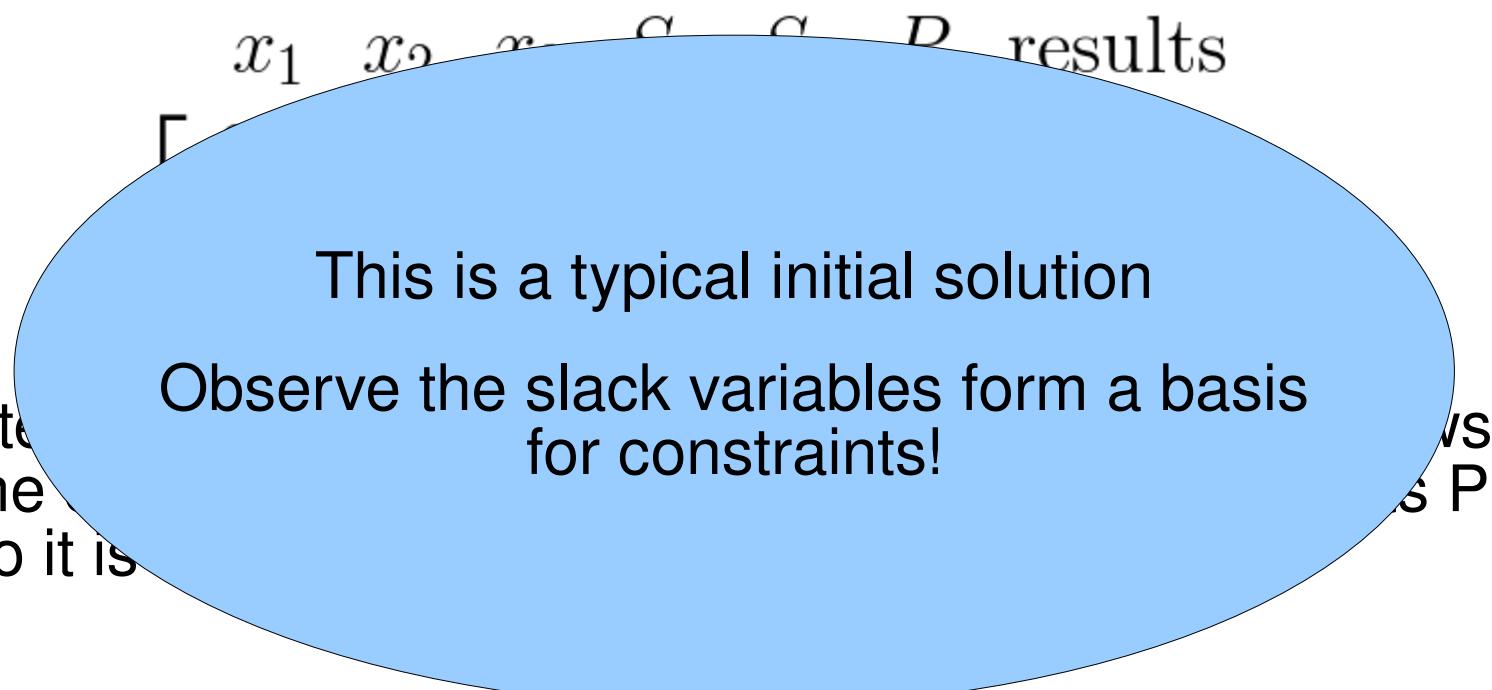
- Now we will start the Simplex steps:

$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
2	3	2	1	0	0	1000
1	1	2	0	1	0	800
-7	-8	-10	0	0	1	0

- This step is called **row reduction**, at first we label the rows with the correspondent slack variable and the last row as P (due to it is associated to the profit)

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	2	3	2	1	0	0	1000
$S_2$	1	1	2	0	1	0	800
$P$	-7	-8	-10	0	0	1	0

- Now we will start the Simplex steps:



- This step starts with the feasible region (due to it is closed)

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	2	3	2	1	0	0	1000
$S_2$	1	1	2	0	1	0	800
$P$	-7	-8	-10	0	0	1	0

- Now we will start the Simplex steps:

**It is typical because:**

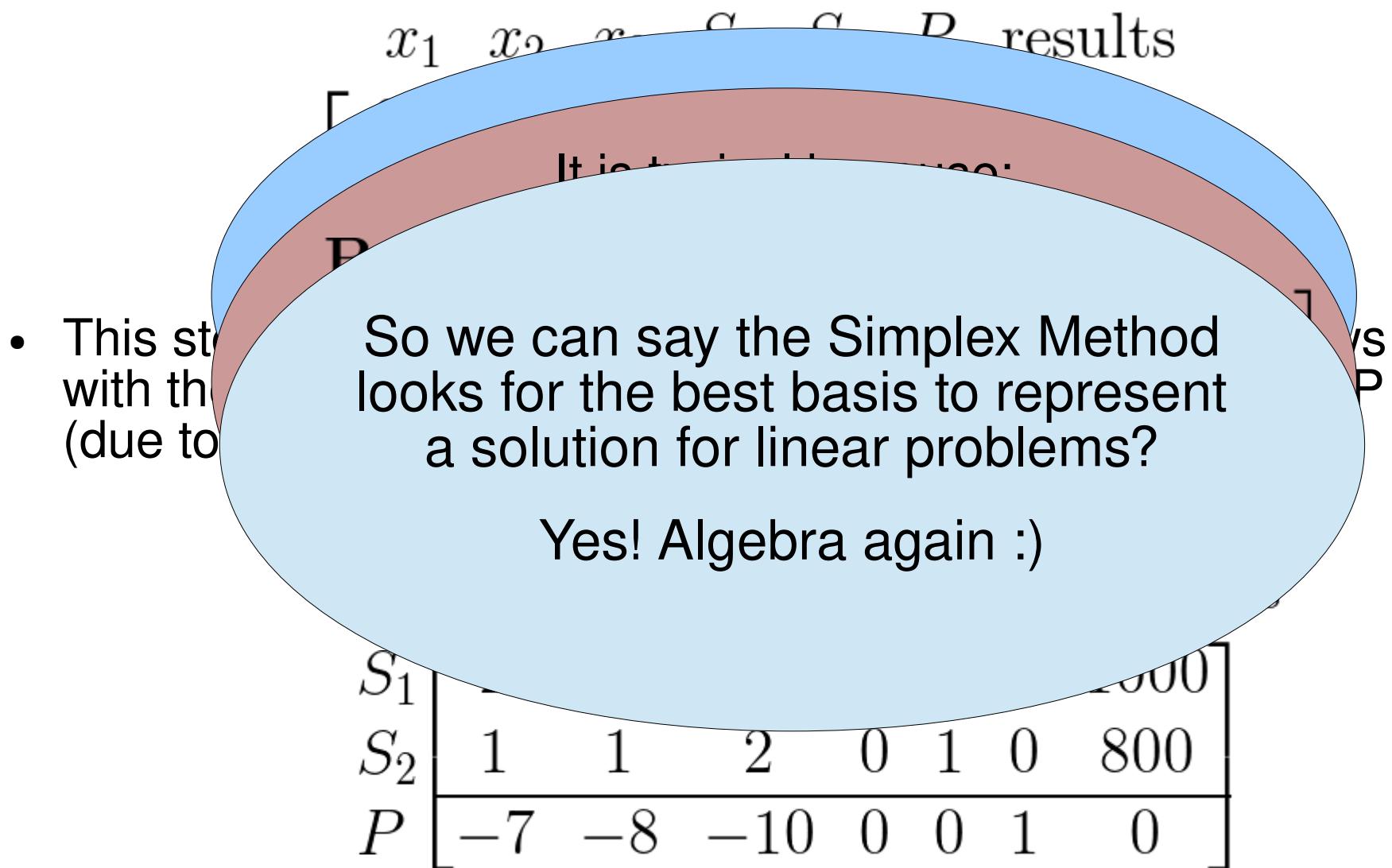
$$Bx = b$$

$$\mathbf{x} = \mathbf{B}^{-1}\mathbf{b} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1000 \\ 800 \end{bmatrix} = \begin{bmatrix} 1000 \\ 800 \end{bmatrix}$$

Since every element in  $\mathbf{x}$  is greater than or equal to zero and the result vector  $\mathbf{b}$  is positive

$$\begin{array}{ccccccc} S_1 & \left[ \begin{matrix} 2 & 5 & 2 & 1 & 0 & 0 & 1000 \end{matrix} \right] \\ S_2 & \left[ \begin{matrix} 1 & 1 & 2 & 0 & 1 & 0 & 800 \end{matrix} \right] \\ P & \left[ \begin{matrix} -7 & -8 & -10 & 0 & 0 & 1 & 0 \end{matrix} \right] \end{array}$$

- Now we will start the Simplex steps:



- The **row reduction** requires us to find the most negative term in the last row

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	2	3	2	1	0	0	1000
$S_2$	1	1	2	0	1	0	800
$P$	-7	-8	-10	0	0	1	0

- In this case it is -10
  - This will be called the **pivot column**
  - We will also need to find which of the constraint rows will be the **pivot row**

- The **row reduction** requires us to find the most negative term in the last row

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	2	3	2	1	0	0	1000
$S_2$	1	1	2	0	1	0	800
$P$	-7	-8	-10	0	0	1	0

- We thus divide the result by corresponding term in the **pivot column** as follows:

- The **row reduction** requires us to find the most negative term in the last row

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	2	3	2	1	0	0	1000
$S_2$	1	1	2	0	1	0	800
$P$	-7	-8	-10	0	0	1	0

- We thus divide the result by corresponding term in the **pivot column** as follows:
  - $1000 / 2 = 500$

- The **row reduction** requires us to find the most negative term in the last row

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	2	3	2	1	0	0	1000
$S_2$	1	1	2	0	1	0	800
$P$	-7	-8	-10	0	0	1	0

- We thus divide the result by corresponding term in the **pivot column** as follows:
  - $800 / 2 = 400$

- The **row reduction** requires us to find the most negative term in the last row

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	2	3	2	1	0	0	1000
$S_2$	1	1	2	0	1	0	800
$P$	-7	-8	-10	0	0	1	0

- We thus divide the result by corresponding term in the **pivot column**, then we select the row that produces the smallest value which is the second:
  - $1000 / 2 = 500$
  - $800 / 2 = 400$
- Thus the second row is the **pivot row**

- Now we found out our pivot, which is the number at the pivot row and pivot column position
  - Thus we will operate on this number to make it equal to one

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	2	3	2	1	0	0	1000
$S_2$	1	1	2	0	1	0	800
$P$	-7	-8	-10	0	0	1	0

- Now we found out our pivot, which is the number at the pivot row and pivot column position
  - Thus we will operate on this number to make it equal to one

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	2	3	2	1	0	0	1000
$S_2$	1	1	2	0	1	0	800
$P$	-7	-8	-10	0	0	1	0

This tells us that  $x_3$  is relevant to compose the next basis

- Now we found out our pivot, which is the number at the pivot row and pivot column position
  - Thus we will operate on this number to make it equal to one

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	2	3	2	1	0	0	1000
$S_2$	1	1	2	0	1	0	800
$P$	-7	-8	-10	0	0	1	0

and  $s_2$  is the less significative variable in the current basis

This tells us that  $x_3$  is relevant to compose the next basis

- Now we found out our pivot, which is the number at the pivot row and pivot column position

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	2	3	2	1	0	0	1000
$S_2$	1	1	2	0	1	0	800
$P$	-7	-8	-10	0	0	1	0

- Thus we will operate on this number to make it equal to one
  - We will make row operations, in this case:  
 $\frac{1}{2} R_2 \rightarrow R_2$
  - To find...

- To find:

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	2	3	2	1	0	0	1000
$S_2$	$\frac{1}{2}$	$\frac{1}{2}$	1	0	$\frac{1}{2}$	0	400
$P$	-7	-8	-10	0	0	1	0

- Now we need to make everything above and below our pivot equals to zero

- Thus we will apply other row operations:

$$-2 R_2 + R_1 \rightarrow R_1$$

$$10 R_2 + R_3 \rightarrow R_3$$

- To find:

Why that?

$S_2$	2						
$P$	-7	-8	-10	0	0	1	0

- Now we need to make everything above and below our pivot equals to zero

- Thus we will apply other row operations:

$$-2 R_2 + R_1 \rightarrow R_1$$

$$10 R_2 + R_3 \rightarrow R_3$$

- To find:

Again, to form a basis and ensure there is an inverse for such basis

$$P \left[ \begin{array}{c|cc} & 1 & -1 \\ \hline & 0 & 0 \end{array} \right]$$

- Now we need to make everything above and below our pivot equals to zero

- Thus we will apply other row operations:

$$-2 R_2 + R_1 \rightarrow R_1$$

$$10 R_2 + R_3 \rightarrow R_3$$

- Then we find:

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	1	2	0	1	-1	0	200
$S_2$	$\frac{1}{2}$	$\frac{1}{2}$	1	0	$\frac{1}{2}$	0	400
$P$	-2	-3	0	0	5	1	4000

- After using this pivot, we change its row name to the column name as follows:

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	1	2	0	1	-1	0	200
$x_3$	$\frac{1}{2}$	$\frac{1}{2}$	1	0	$\frac{1}{2}$	0	400
$P$	-2	-3	0	0	5	1	4000

- Then we find:

Yes, now we have a new basis!

- After using this information, we can change the column name as follows:

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	1	2	0	1	-1	0	200
$x_3$	$\frac{1}{2}$	$\frac{1}{2}$	1	0	$\frac{1}{2}$	0	400
$P$	-2	-3	0	0	5	1	4000

- Then we find:

This new basis is:

$$\mathbf{x} = \begin{bmatrix} S_1 \\ x_3 \end{bmatrix} = \mathbf{B}^{-1}\mathbf{b} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 200 \\ 400 \end{bmatrix} = \begin{bmatrix} 200 \\ 400 \end{bmatrix}$$

- After name  $\alpha$

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	1	2	0	1	-1	0	200
$x_3$	$\frac{1}{2}$	$\frac{1}{2}$	1	0	$\frac{1}{2}$	0	400
$P$	-2	-3	0	0	5	1	4000

- Now look again at the last row and if there is another negative number, we need to execute this pivot operation again

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	1	2	0	1	-1	0	200
$x_3$	$\frac{1}{2}$	$\frac{1}{2}$	1	0	$\frac{1}{2}$	0	400
$P$	-2	-3	0	0	5	1	4000

- So we select the most negative number to be the pivot column

- Now look again at the last row and if there is another negative number, we need to execute this pivot operation again

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	1	2	0	1	-1	0	200
$x_3$	$\frac{1}{2}$	$\frac{1}{2}$	1	0	$\frac{1}{2}$	0	400
$P$	-2	-3	0	0	5	1	4000

- So we select the most negative number to be the pivot column
  - Then divide again results by numbers to find the pivot row

- We now proceed with the pivot operation to make our pivot number equals to one

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	1	2	0	1	-1	0	200
$x_3$	$\frac{1}{2}$	$\frac{1}{2}$	1	0	$\frac{1}{2}$	0	400
$P$	-2	-3	0	0	5	1	4000

- Thus we apply the following row operation:

$$\frac{1}{2} R_1 \rightarrow R_1$$

- To obtain:

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	$\frac{1}{2}$	1	0	$\frac{1}{2}$	$-\frac{1}{2}$	0	100
$x_3$	$\frac{1}{2}$	$\frac{1}{2}$	1	0	$\frac{1}{2}$	0	400
$P$	-2	-3	0	0	5	1	4000

- Now we make the column terms equal to zero by applying:

$$-\frac{1}{2} R_1 + R_2 \rightarrow R_2$$

$$3 R_1 + R_3 \rightarrow R_3$$

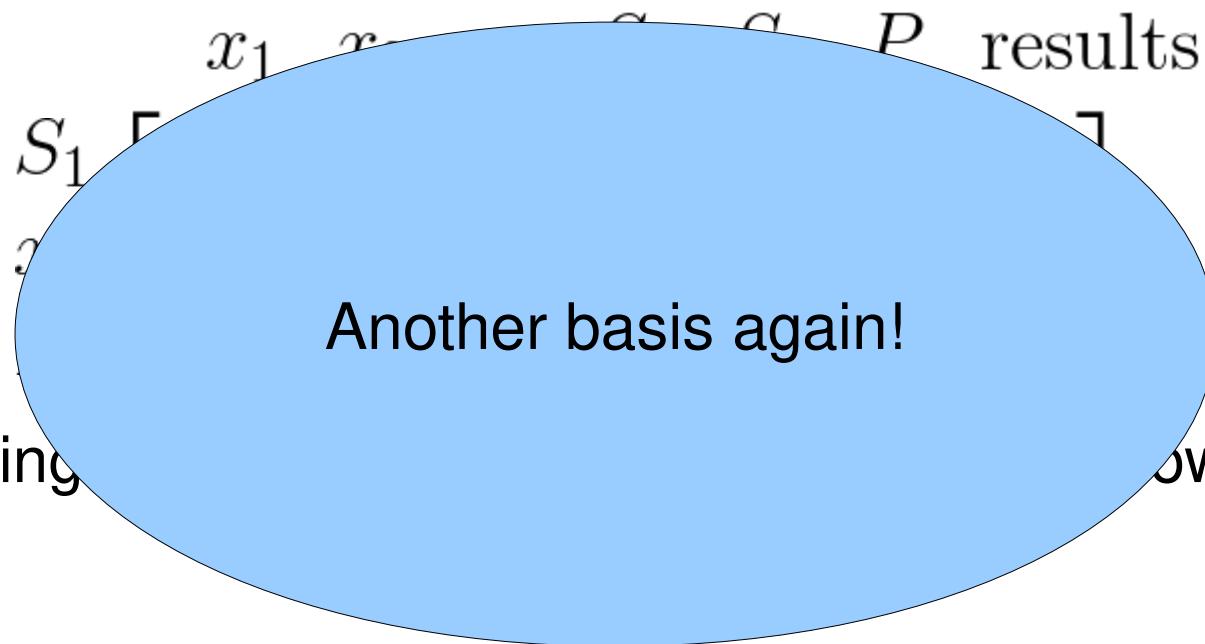
- And find:

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$S_1$	$\frac{1}{2}$	(1) 0	$\frac{1}{2}$	$-\frac{1}{2}$	0	100	
$x_3$	$\frac{1}{4}$	0 1	$-\frac{1}{4}$	$\frac{3}{4}$	0	350	
$P$	$-\frac{1}{2}$	0 0	$\frac{3}{2}$	$\frac{7}{2}$	1	4300	

- After solving for this pivot number we change the row name to obtain:

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$x_2$	$\frac{1}{2}$	(1) 0	$\frac{1}{2}$	$-\frac{1}{2}$	0	100	
$x_3$	$\frac{1}{4}$	0 1	$-\frac{1}{4}$	$\frac{3}{4}$	0	350	
$P$	$-\frac{1}{2}$	0 0	$\frac{3}{2}$	$\frac{7}{2}$	1	4300	

- And find:



- After solving we obtain:

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$x_2$	$\frac{1}{2}$	(1) 0	$\frac{1}{2}$	$-\frac{1}{2}$	0	100	
$x_3$	$\frac{1}{4}$	0	1	$-\frac{1}{4}$	$\frac{3}{4}$	0	350
$P$	$-\frac{1}{2}$	0	0	$\frac{3}{2}$	$\frac{7}{2}$	1	4300

- But we still have a negative number in the last row, so we need to apply the pivot operation again

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$x_2$	$\frac{1}{2}$	1	0	$\frac{1}{2}$	$-\frac{1}{2}$	0	100
$x_3$	$\frac{1}{4}$	0	1	$-\frac{1}{4}$	$\frac{3}{4}$	0	350
$P$	$-\frac{1}{2}$	0	0	$\frac{3}{2}$	$\frac{7}{2}$	1	4300

- So we need to find the pivot row
  - Which is clearly the first row

- But we still have a negative number in the last row, so we need to apply the pivot operation again

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$x_2$	$\frac{1}{2}$	1	0	$\frac{1}{2}$	$-\frac{1}{2}$	0	100
$x_3$	$\frac{1}{4}$	0	1	$-\frac{1}{4}$	$\frac{3}{4}$	0	350
$P$	$-\frac{1}{2}$	0	0	$\frac{3}{2}$	$\frac{7}{2}$	1	4300

- Now we need to make the pivot number equals to one

$$2 R_1 \rightarrow R_1$$

- But we still have a negative number in the last row, so we need to apply the pivot operation again

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$x_2$	1	2	0	1	-1	0	200
$x_3$	$\frac{1}{4}$	0	1	$-\frac{1}{4}$	$\frac{3}{4}$	0	350
$P$	$-\frac{1}{2}$	0	0	$\frac{3}{2}$	$\frac{7}{2}$	1	4300

- Now we need to make other numbers in this column equal to zero

$$-\frac{1}{4} R_1 + R_2 \rightarrow R_2$$

$$\frac{1}{2} R_1 + R_3 \rightarrow R_3$$

- But we still have a negative number in the last row, so we need to apply the pivot operation again

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$x_2$	1	2	0	1	-1	0	200
$x_3$	0	$-\frac{1}{2}$	1	$-\frac{1}{2}$	1	0	300
$P$	0	1	0	2	3	1	4400

- After finishing this pivot operation we change the row name to the column name as follows

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$x_1$	1	2	0	1	-1	0	200
$x_3$	0	$-\frac{1}{2}$	1	$-\frac{1}{2}$	1	0	300
$P$	0	1	0	2	3	1	4400

- But we still have a negative number in the last row, so we need to apply the pivot operation again

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$x_2$	1	2	0	-1	1	0	200

- After applying the pivot operation, we get the following tableau:
- We are now finished.**  
**But we still need to interpret the results!**

$x_1$	1	2	0	-1	1	0	200
$x_3$	0	$-\frac{1}{2}$	1	$-\frac{1}{2}$	1	0	300
$P$	0	1	0	2	3	1	4400

- Interpreting the final result:

	$x_1$	$x_2$	$x_3$	$S_1$	$S_2$	$P$	results
$x_1$	1	2	0	1	-1	0	200
$x_3$	0	$-\frac{1}{2}$	1	$-\frac{1}{2}$	1	0	300
$P$	0	1	0	2	3	1	4400

- The maximum profit is obtained setting the **row variables** equal to the values at the **right**

$$x_1 = 200, x_3 = 300$$

- The profit value is 4400
- Variables which are not in row names are set equal to zero

$$x_2 = 0, S_1 = 0, S_2 = 0$$

- Finally we will build:
  - 200 components of type A
  - Zero components of type B
  - 300 components of type C
  - Having zero slack fabrication time ( $S_1$ )
  - Having zero slack assembly time ( $S_2$ )
  - And having a profit equals to \$4,400

- Material based on:

- <https://www.youtube.com/watch?v=gRgsT9BB5-8&list=WL&index=25>
- <https://www.youtube.com/watch?v=yL7JByLlfrw&list=WL&index=40>
- <https://www.youtube.com/watch?v=vVzjXpwW2xI&index=23&list=WL>
- <https://www.youtube.com/watch?v=lPm46c1pfvQ&list=WL&index=26>
- <https://www.youtube.com/watch?v=WeK4JjNLSgw&index=41&list=WL>

**There is more...**

# What about the Karush-Kuhn-Tucker conditions?

- They are necessary conditions to ensure optimality for differentiable nonlinear programming problems that satisfy certain regularity conditions
- So, consider the **Primal** problem:

$$\begin{aligned} \text{Minimize: } & \mathbf{c}\mathbf{x} \\ \text{Subject to: } & \mathbf{A}\mathbf{x} \geq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned}$$

- And its **Dual** form:

$$\begin{aligned} \text{Maximize: } & \mathbf{b}\mathbf{w} \\ \text{Subject to: } & \mathbf{w}\mathbf{A} \leq \mathbf{c} \\ & \mathbf{w} \geq \mathbf{0} \end{aligned}$$

# What about the Karush-Kuhn-Tucker conditions?

- The Dual form was found using the following (**easier to apply**) Table (Bazaraa, Jarvis and Sherali, Linear Programming and Network Flows, 4<sup>th</sup> edition, John Wiley & Sons, page 264):

**Table 6.1 Relationships Between Primal and Dual Problems**

MINIMIZATION PROBLEM		MAXIMIZATION PROBLEM		Constraints
Variables	$\geq 0$	$\leq 0$	$\geq$	
Constraints	$\leq 0$	$\geq 0$	$\leq$	
	Unrestricted	$\geq 0$	$=$	
	$\geq 0$	$\leq 0$	$\geq 0$	
Variables	$\geq$	$\leq$	$\leq 0$	Variable s
	$\leq$	$\geq$	$\geq 0$	
	$=$	$\geq 0$	Unrestricted	

## Primal Problem

Minimize:  $\mathbf{c}\mathbf{x}$

Subject to:  $\mathbf{Ax} \geq \mathbf{b}$   
 $\mathbf{x} \geq 0$

## Dual Form

Maximize:  $\mathbf{bw}$

Subject to:  $\mathbf{wA} \leq \mathbf{c}$   
 $\mathbf{w} \geq 0$

# What about the Karush-Kuhn-Tucker conditions?

- Considering the constraints, we have the KKT conditions:

$$\mathbf{Ax} \geq \mathbf{b}, \quad \mathbf{x} \geq \mathbf{0}$$

$$\mathbf{wA} + \mathbf{s} = \mathbf{c}, \quad \mathbf{w} \geq \mathbf{0}, \quad \mathbf{s} \geq \mathbf{0}$$

$$\mathbf{w}(\mathbf{Ax} - \mathbf{b}) = \mathbf{0}, \quad \mathbf{sx} = \mathbf{0}$$

# What about the Karush-Kuhn-Tucker conditions?

- Considering the constraints, we have the KKT conditions:

$$\mathbf{Ax} \geq \mathbf{b}, \quad \mathbf{x} \geq \mathbf{0}$$

$$\mathbf{wA} + \mathbf{s} = \mathbf{c}, \quad \mathbf{w} \geq \mathbf{0}, \quad \mathbf{s} \geq \mathbf{0}$$

$$\mathbf{w}(\mathbf{Ax} - \mathbf{b}) = \mathbf{0}, \quad \mathbf{sx} = \mathbf{0}$$

This first constraint or condition is typically referred to as **primal feasibility**

# What about the Karush-Kuhn-Tucker conditions?

- Considering the constraints, we have the KKT conditions:

$$\mathbf{Ax} \geq \mathbf{b}, \quad \mathbf{x} \geq \mathbf{0}$$

$$\mathbf{wA} + \mathbf{s} = \mathbf{c}, \quad \mathbf{w} \geq \mathbf{0}, \quad \mathbf{s} \geq \mathbf{0}$$

$$\mathbf{w}(\mathbf{Ax} - \mathbf{b}) = \mathbf{0}, \quad \mathbf{sx} = \mathbf{0}$$

This second condition is typically referred to as **dual feasibility**

# What about the Karush-Kuhn-Tucker conditions?

- Considering the constraints, we have the KKT conditions:

$$\mathbf{Ax} \geq \mathbf{b}, \mathbf{x} \geq \mathbf{0}$$

$$\mathbf{wA} + \mathbf{s} = \mathbf{c}, \mathbf{w} \geq \mathbf{0}, \mathbf{s} \geq \mathbf{0}$$

$$\mathbf{w}(\mathbf{Ax} - \mathbf{b}) = \mathbf{0}, \mathbf{sx} = \mathbf{0}$$

This third condition is typically referred to as **complementary slackness**

# What about the Karush-Kuhn-Tucker conditions?

- Considering the constraints, we have the KKT conditions:

$$\mathbf{Ax} \geq \mathbf{b}, \quad \mathbf{x} \geq \mathbf{0}$$

$$\mathbf{wA} + \mathbf{s} = \mathbf{c}, \quad \mathbf{w} \geq \mathbf{0}, \quad \mathbf{s} \geq \mathbf{0}$$

$$\mathbf{w}(\mathbf{Ax} - \mathbf{b}) = \mathbf{0}, \quad \mathbf{sx} = \mathbf{0}$$

This term  $\mathbf{sx}$  is used to measure the gap between the primal and dual solutions

# What about the Karush-Kuhn-Tucker conditions?

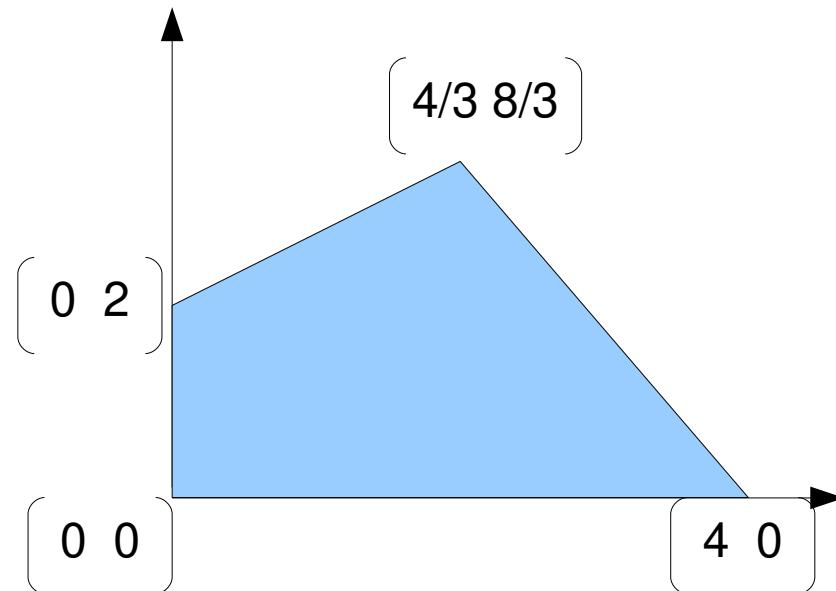
- We will now study an example:
  - Consider the linear programming problem:

Minimize:  $-x_1 - 3x_2$

Subject to:  $x_1 - 2x_2 \geq -4$

$-x_1 - x_2 \geq -4$

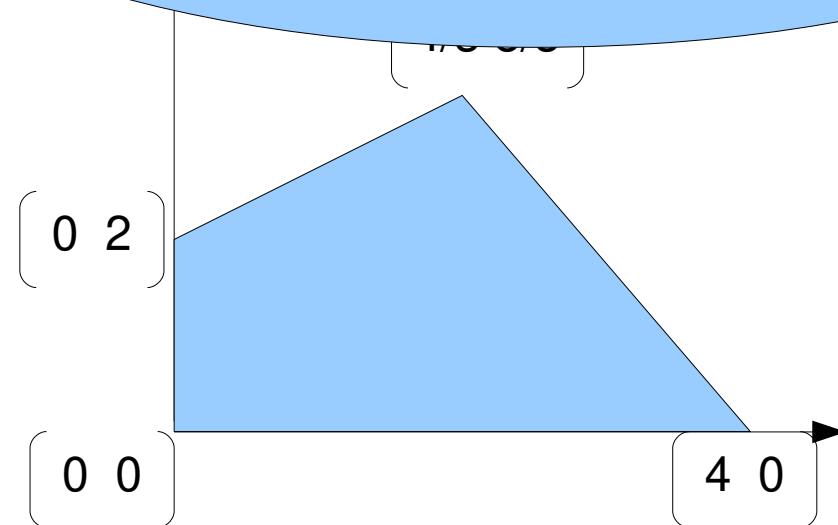
$x_1, x_2 \geq 0$



# What about the Karush-Kuhn-Tucker conditions?

- We will now study an example:
  - Consider the linear programming problem:

We will now verify if a point is optimal using the KKT conditions



# What about the Karush-Kuhn-Tucker conditions?

- Verifying the point  $\mathbf{x} = (0 \ 0)$

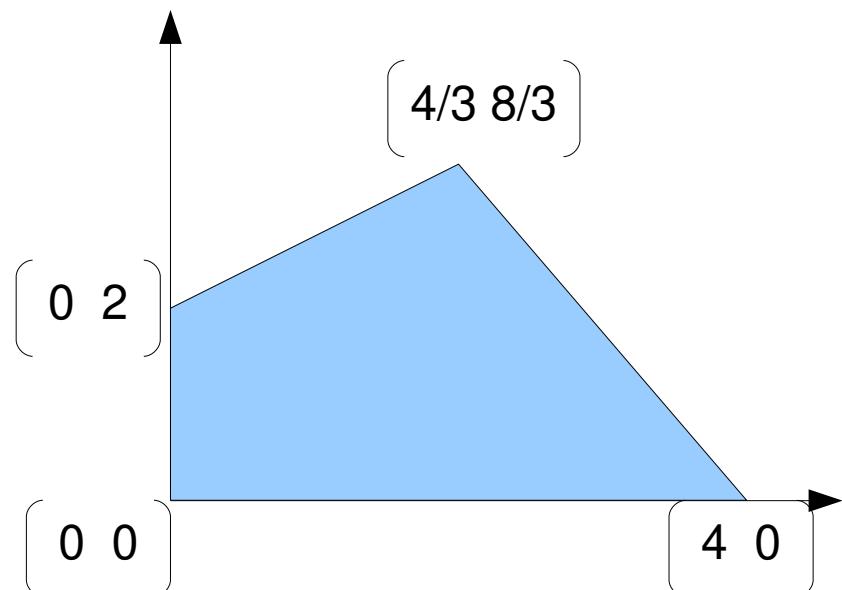
$$\mathbf{Ax} \geq \mathbf{b}, \mathbf{x} \geq \mathbf{0}$$

$$\mathbf{wA} + \mathbf{s} = \mathbf{c}, \mathbf{w} \geq \mathbf{0}, \mathbf{s} \geq \mathbf{0}$$

$$\mathbf{w}(\mathbf{Ax} - \mathbf{b}) = \mathbf{0}, \mathbf{sx} = \mathbf{0}$$

Minimize:  $-x_1 - 3x_2$   
Subject to:  $x_1 - 2x_2 \geq -4$   
 $-x_1 - x_2 \geq -4$   
 $x_1, x_2 \geq 0$

- We see the first condition is satisfied as  $\mathbf{A}(0 \ 0) \geq \mathbf{b}$



# What about the Karush-Kuhn-Tucker conditions?

- Verifying the point  $\mathbf{x} = (0 \ 0)$

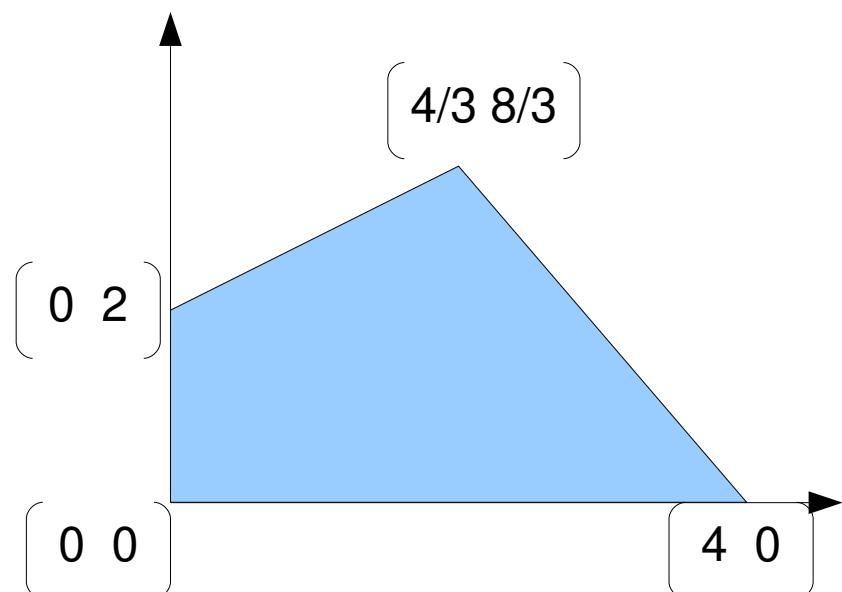
$$\mathbf{Ax} \geq \mathbf{b}, \mathbf{x} \geq \mathbf{0}$$

$$\mathbf{wA} + \mathbf{s} = \mathbf{c}, \mathbf{w} \geq \mathbf{0}, \mathbf{s} \geq \mathbf{0}$$

$$\mathbf{w}(\mathbf{Ax} - \mathbf{b}) = 0, \mathbf{s}\mathbf{x} = 0$$

- We geometrically observe the first two constraints are non-binding
- So (according to the third condition) we will certainly have vector  
 $\mathbf{w} = (0 \ 0)$
- Then from the second condition we will have  $\mathbf{s} = \mathbf{c}$  and so  
 $\mathbf{s} = (-1 \ -3)$   
violates the nonnegativity of  $\mathbf{s}$
- Therefore the point  $\mathbf{x}=(0 \ 0)$  is not an optimal point

Minimize:  $-x_1 - 3x_2$   
Subject to:  $x_1 - 2x_2 \geq -4$   
 $-x_1 - x_2 \geq -4$   
 $x_1, x_2 \geq 0$



# What about the Karush-Kuhn-Tucker conditions?

- Verifying the point  $\mathbf{x} = (0 \ 0)$

$$\mathbf{Ax} \geq \mathbf{b}, \mathbf{x} \geq \mathbf{0}$$

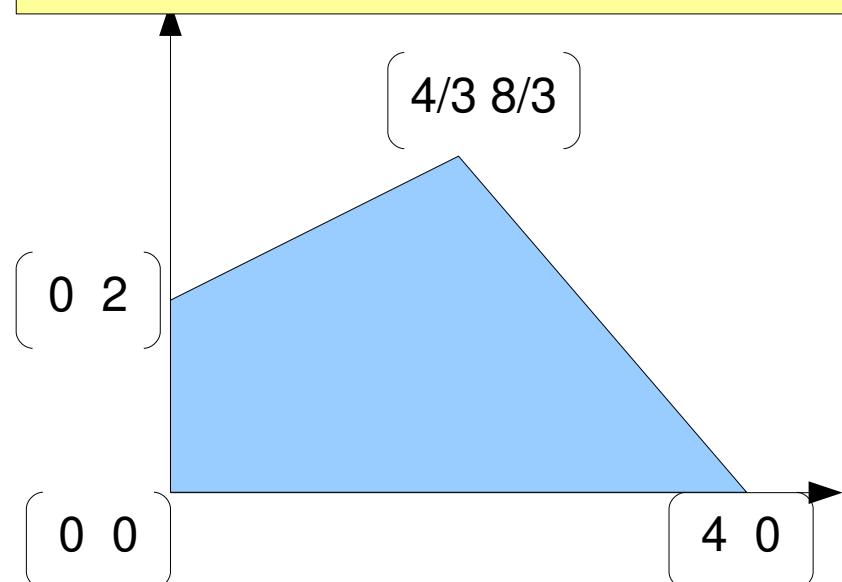
$$\mathbf{wA} + \mathbf{s} = \mathbf{c}, \mathbf{w} \geq \mathbf{0}, \mathbf{s} \geq \mathbf{0}$$

$$\mathbf{w}(\mathbf{Ax} - \mathbf{b}) = \mathbf{0}, \mathbf{sx} = \mathbf{0}$$

- We geometrically observe the first two constraints are non-binding
- So (according to the third condition) we will certainly have vector  
 $\mathbf{w} = (0 \ 0)$
- Then from the second condition we will have  $\mathbf{s} = \mathbf{c}$  and so  
 $\mathbf{s} = (-1 \ -3)$   
violates the nonnegativity of  $\mathbf{s}$
- Therefore the point  $\mathbf{x}=(0 \ 0)$  is not an optimal point

This is very important!

Observe vector  $\mathbf{w}$ , i.e.,  
the Lagrange multipliers, tell  
us which constraints are active!



# What about the Karush-Kuhn-Tucker conditions?

- Try to verify the point  $(4/3 \ 8/3)$ 
  - Observe  $\mathbf{w}$  has no element equals to zero, but why???

$$\mathbf{A}\mathbf{x} \geq \mathbf{b}, \mathbf{x} \geq \mathbf{0}$$

$$\mathbf{w}\mathbf{A} + \mathbf{s} = \mathbf{c}, \mathbf{w} \geq \mathbf{0}, \mathbf{s} \geq \mathbf{0}$$

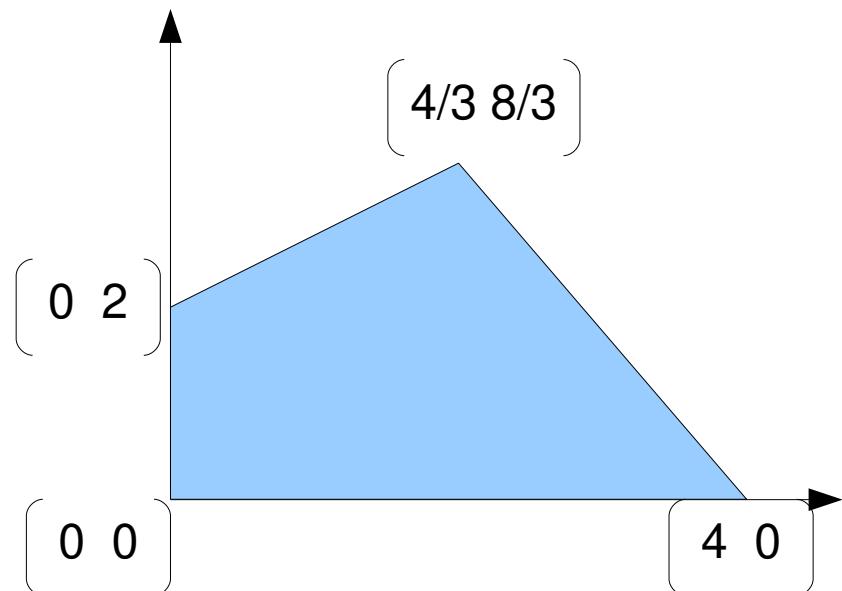
$$\mathbf{w}(\mathbf{A}\mathbf{x} - \mathbf{b}) = \mathbf{0}, \mathbf{s}\mathbf{x} = \mathbf{0}$$

$$\text{Minimize: } -x_1 - 3x_2$$

$$\text{Subject to: } x_1 - 2x_2 \geq -4$$

$$-x_1 - x_2 \geq -4$$

$$x_1, x_2 \geq 0$$



# What about the Karush-Kuhn-Tucker conditions?

- Try to verify the point  $(4/3 \ 8/3)$ 
  - Observe  $\mathbf{w}$  has no element equals to zero, but why???
    - Because both constraints are binding
  - As we have  $x_1 > 0$  and  $x_2 > 0$  then  $s_1$  and  $s_2$  must be equal to zero ( $s_1 = s_2 = 0$ ) in order to satisfy the complementary slackness, then:

$$\begin{aligned} \mathbf{wA} + \mathbf{s} &= \mathbf{c} \\ \mathbf{wA} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} &= \mathbf{c} \\ \begin{cases} w_1 - w_2 = -1 \\ -2w_1 - w_2 = -3 \end{cases} \\ w_1 = \frac{2}{3}, \quad w_2 = \frac{5}{3} \end{aligned}$$

# What about the Karush-Kuhn-Tucker conditions?

- Try to verify the point  $(4/3 \ 8/3)$

- Observe  $w$  has

- Because

- As

- to

but why???

So, having:

$$s_1 = s_2 = 0$$

$$\frac{4}{3}, \frac{8}{3}$$

$$x_1 = \frac{2}{3}, x_2 = \frac{5}{3}$$

$$w_1 = \frac{1}{3}, w_2 = \frac{2}{3}$$

All KKT conditions are satisfied.

Therefore the point  $(4/3 \ 8/3)$  is the optimal point

$\omega_1$

3

3

**There is another way to interpret the KKT conditions...**

# What about the Karush-Kuhn-Tucker conditions?

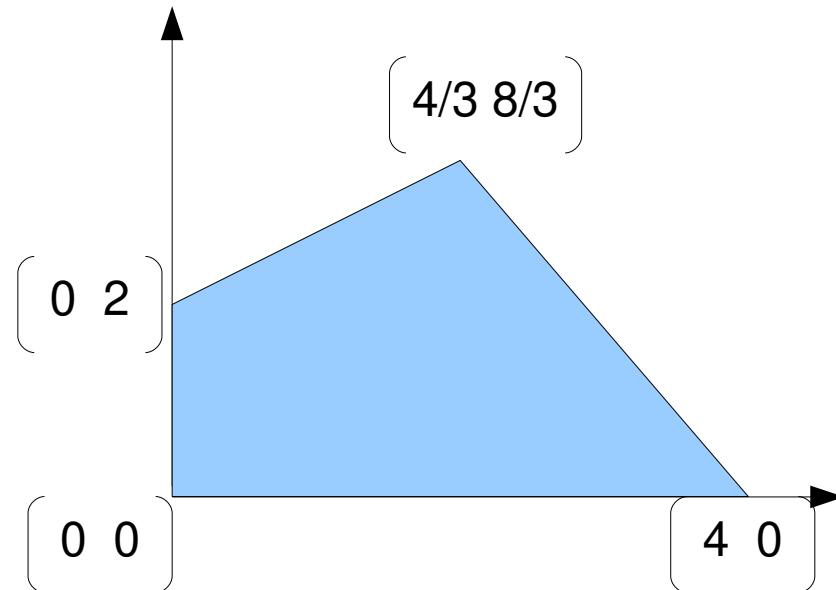
- Getting back to the linear programming problem:

Minimize:  $-x_1 - 3x_2$

Subject to:  $x_1 - 2x_2 \geq -4$

$-x_1 - x_2 \geq -4$

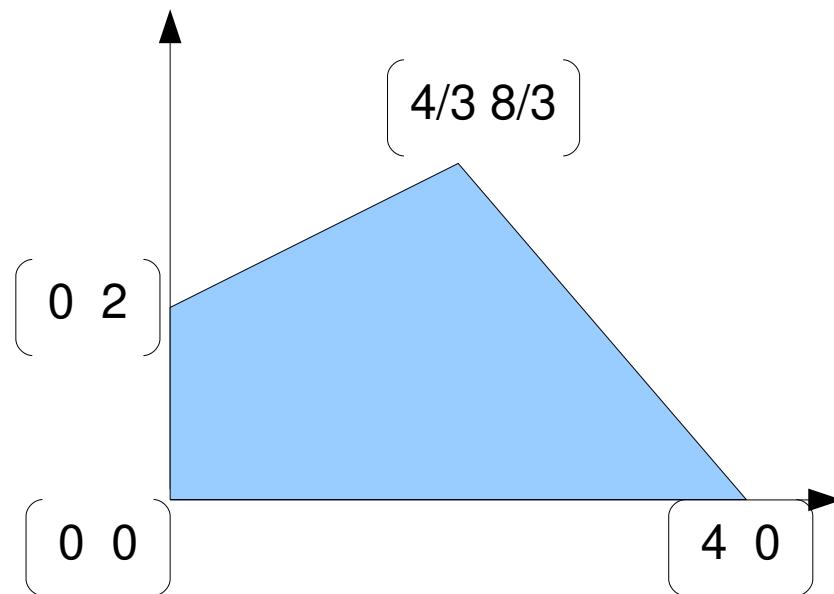
$x_1, x_2 \geq 0$



# What about the Karush-Kuhn-Tucker conditions?

- Getting back to the linear programming problem:

Now plot the gradients for the objective function and constraints



# What about the Karush-Kuhn-Tucker conditions?

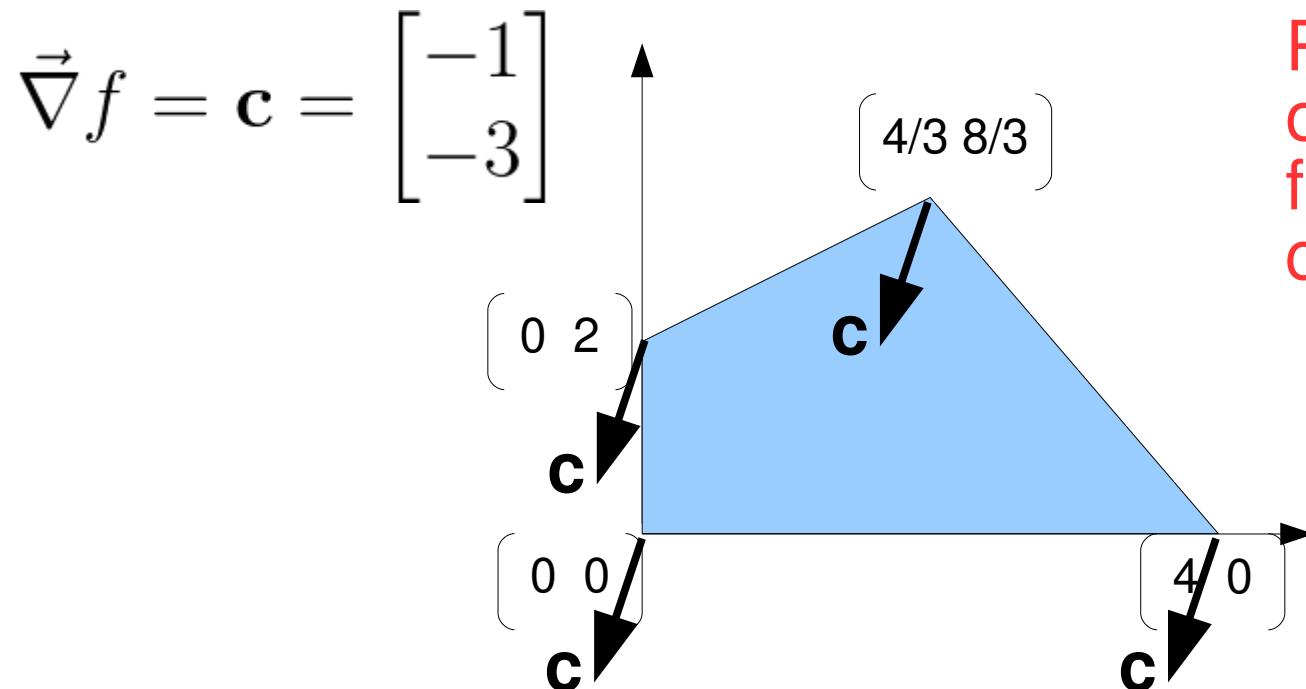
- Getting back to the linear programming problem:

Minimize:  $-x_1 - 3x_2$

Subject to:  $x_1 - 2x_2 \geq -4$

$-x_1 - x_2 \geq -4$

$x_1, x_2 \geq 0$



Plotting the gradient  
of the objective  
function at every  
candidate point

# What about the Karush-Kuhn-Tucker conditions?

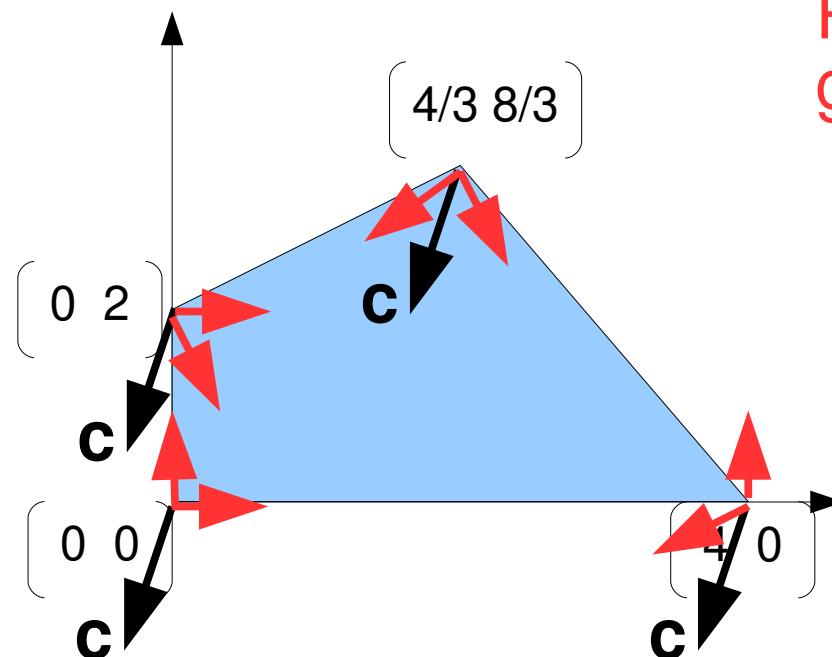
- Getting back to the linear programming problem:

Minimize:  $-x_1 - 3x_2$

Subject to:  $x_1 - 2x_2 \geq -4$

$-x_1 - x_2 \geq -4$

$x_1, x_2 \geq 0$



Plotting constraint  
gradients:

$$\mathbf{a}^1 = (1, -2)$$

$$\mathbf{a}^2 = (-1, -1)$$

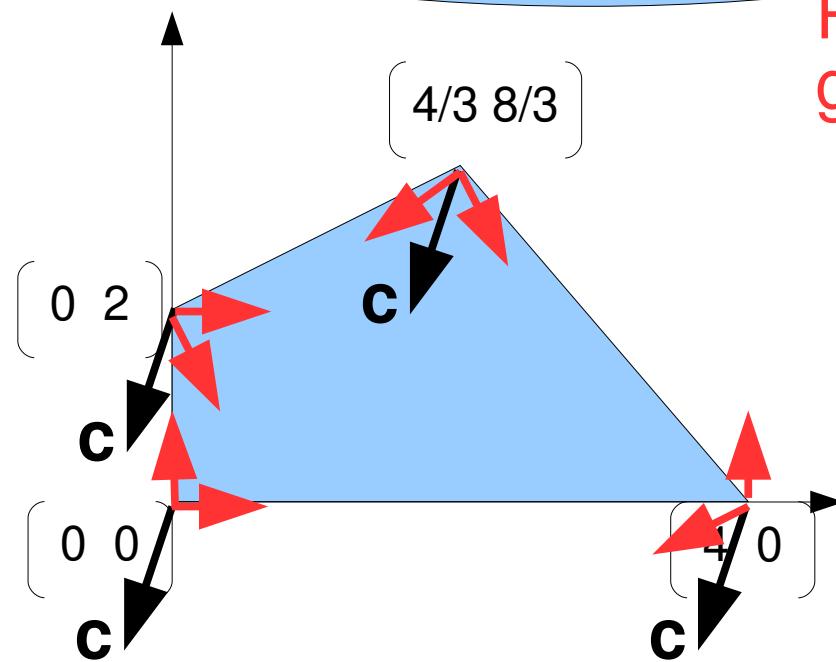
$$\mathbf{e}_1 = (1, 0)$$

$$\mathbf{e}_2 = (0, 1)$$

# What about the Karush-Kuhn-Tucker conditions?

- Getting back to the linear programming problem:

Observe the cones!



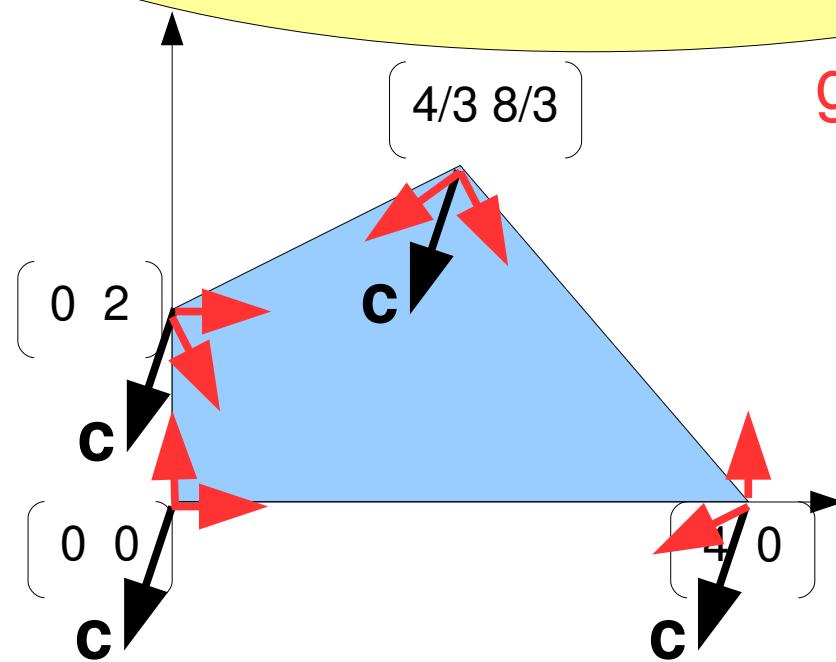
Plotting constraint  
gradients:

$$\begin{aligned}\mathbf{a}^1 &= (1, -2) \\ \mathbf{a}^2 &= (-1, -1) \\ \mathbf{e}_1 &= (1, 0) \\ \mathbf{e}_2 &= (0, 1)\end{aligned}$$

# What about the Karush-Kuhn-Tucker conditions?

- Getting back to the linear programming problem:

For this **minimization problem**, the gradient of the objective function must be inside the cone to have an optimal solution



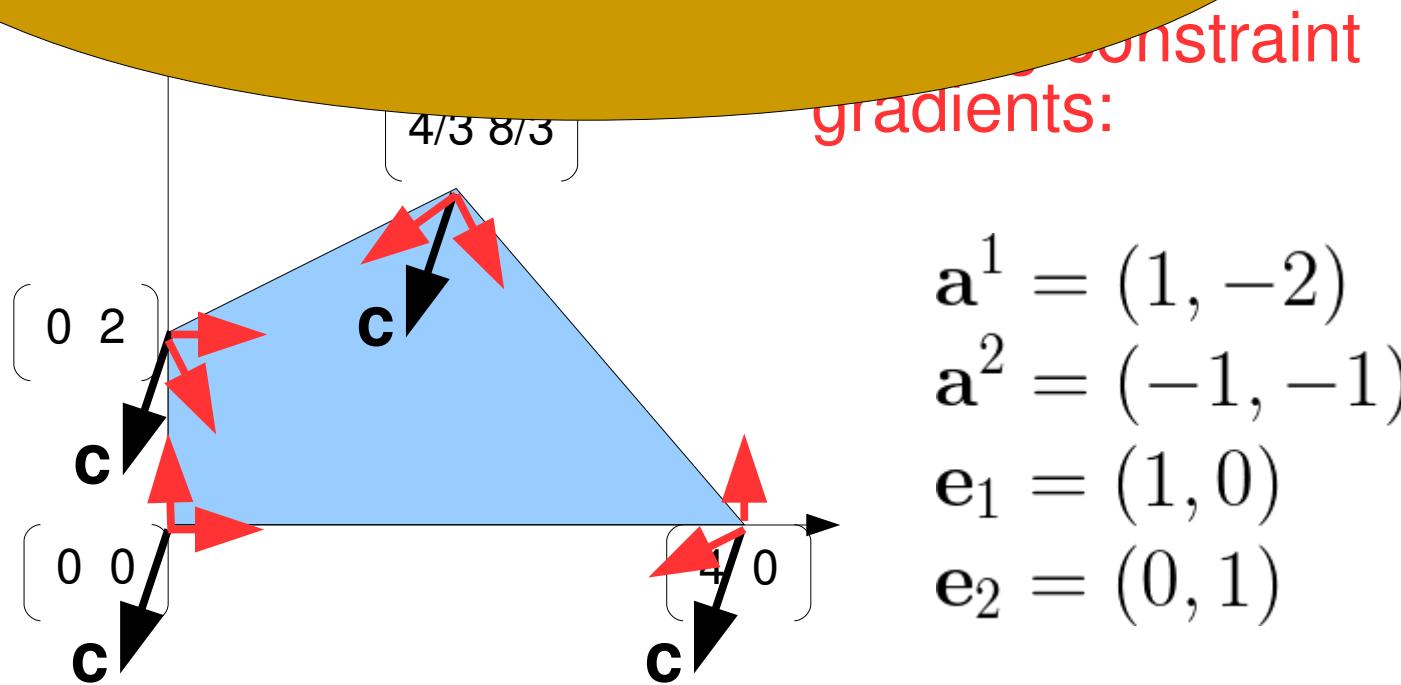
Feasible constraint gradients:

$$\begin{aligned} \mathbf{a}^1 &= (1, -2) \\ \mathbf{a}^2 &= (-1, -1) \\ \mathbf{e}_1 &= (1, 0) \\ \mathbf{e}_2 &= (0, 1) \end{aligned}$$

# What about the Karush-Kuhn-Tucker conditions?

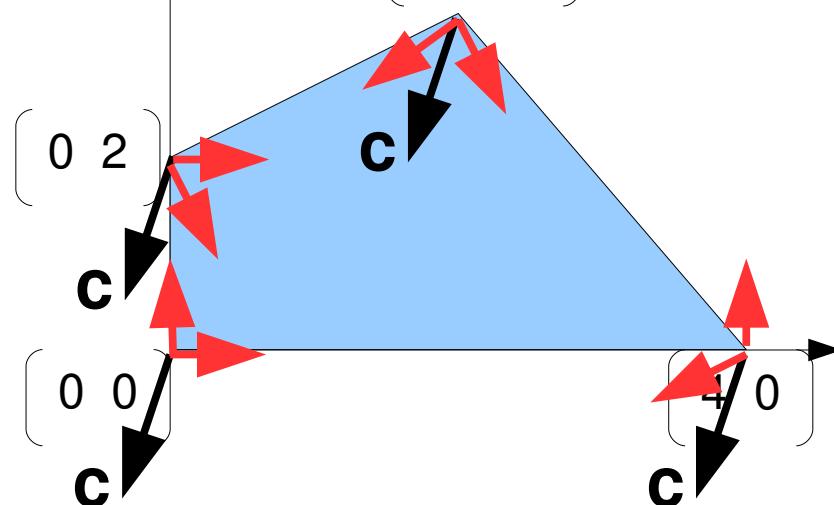
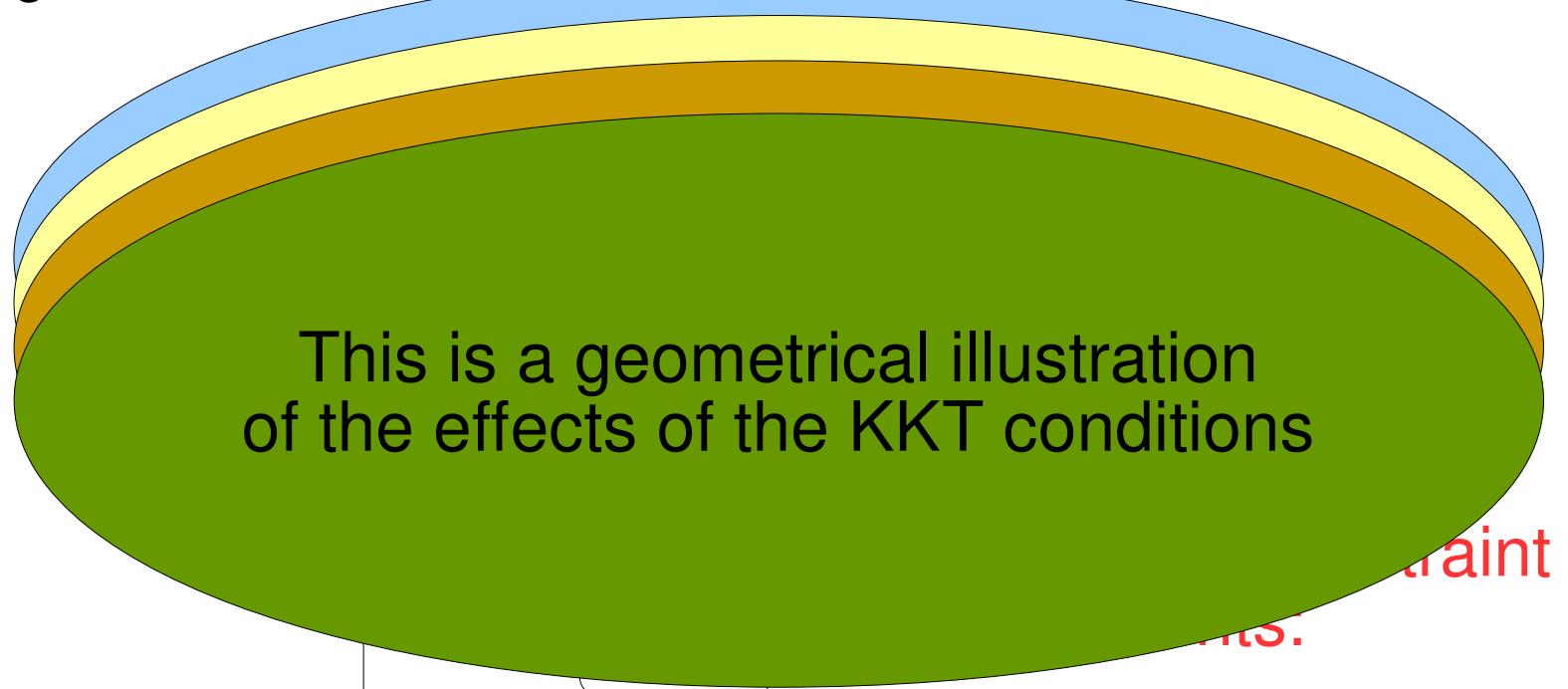
- Getting back to the linear programming problem:

For a **maximization problem**, the **negative** of the gradient of the objective function must be inside the cone to have an optimal solution



# What about the Karush-Kuhn-Tucker conditions?

- Getting back to the linear programming problem:



$$\begin{aligned}\mathbf{a}^1 &= (1, -2) \\ \mathbf{a}^2 &= (-1, -1) \\ \mathbf{e}_1 &= (1, 0) \\ \mathbf{e}_2 &= (0, 1)\end{aligned}$$

# What about the Karush-Kuhn-Tucker conditions?

- To better understand the KKT conditions, study more about the **Farkas' lemma**
  - Bazaraa, Jarvis and Sherali, Linear Programming and Network Flows, 4<sup>th</sup> edition, John Wiley & Sons
- As exercise, try to solve this previous problem using the Simplex Tableau
  - Observe:
    - You will have a basis
    - You will have an inverse matrix for such basis
    - You will find  $w$  in the Tableau
    - Values for  $w$  will be different from zero, meaning both constraints are binding

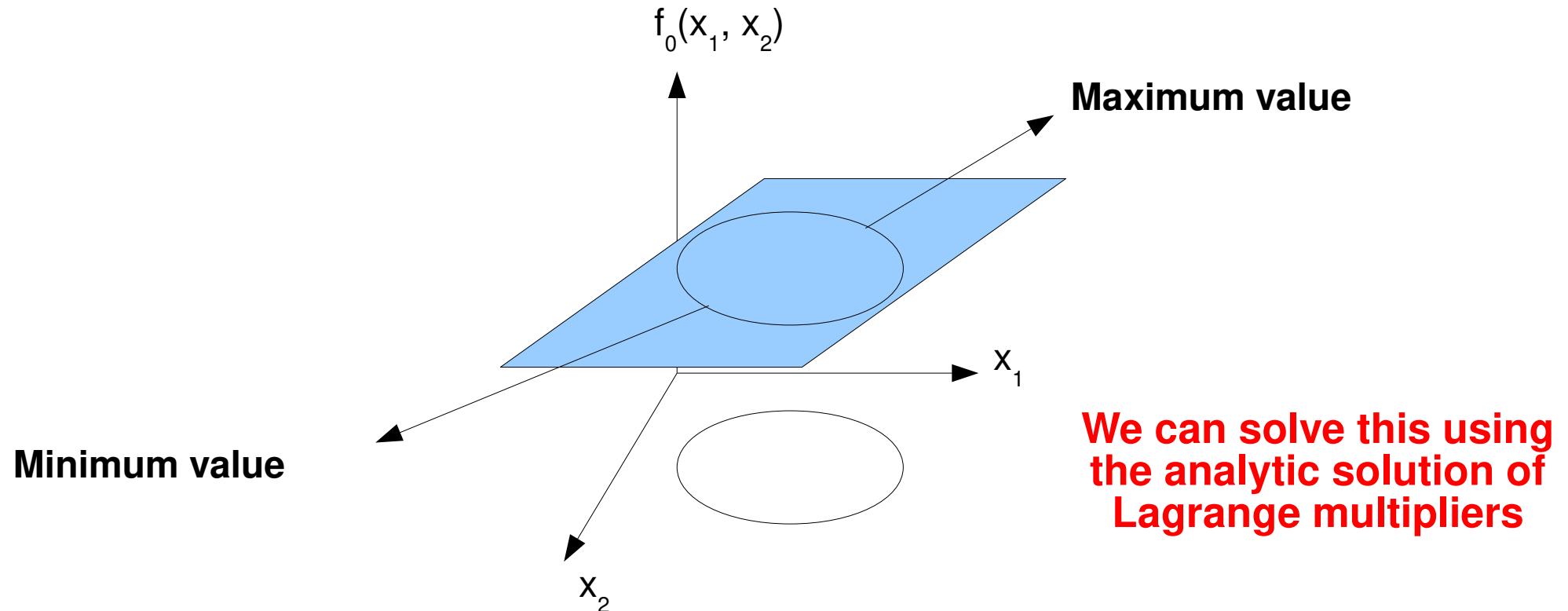
# **How to solve Convex Optimization Problems?**

# Convex Optimization Problems

- For **linear optimization** problems we can **graph constraints** to solve manually or algorithmically approach them using the **Simplex** method
- For convex problems we have some situations:
  - Convex constraint(s) and Linear objective function

# Convex Optimization Problems

- For **linear optimization** problems we can **graph constraints** to solve manually or algorithmically approach them using the **Simplex** method
- For convex problems we have some situations:
  - Convex constraint(s) and Linear objective function

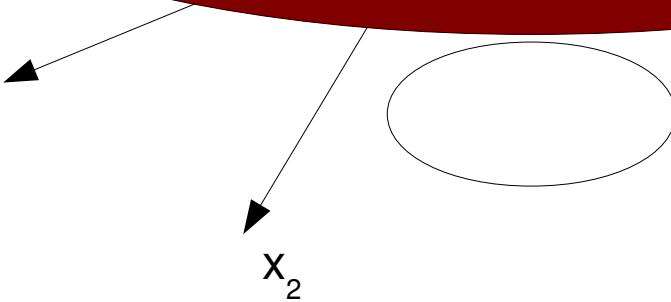


# Convex Optimization Problems

- For **linear optimization** problems we can **graph constraints** to solve manually or algorithmically approach them using the **Simplex** method
- For convex problems we have some situations:
  - Convex constraint function

Let's recall this situation

Minimum value



We can solve this using  
the analytic solution of  
Lagrange multipliers

# Convex const and Linear Objective Function

- Lagrange Multipliers
  - Now suppose a problem (if the conditions below are satisfied we can apply this approach):
  - Are f and g differentiable?
    - Yes
  - Does it consider one or more equality constraints?
    - Yes
  - The gradient of g is different of zero vector?
    - Yes

$$\begin{aligned} & \text{maximize } f(x, y) = x + y \\ & \text{subject to } x^2 + y^2 = 1 \end{aligned}$$

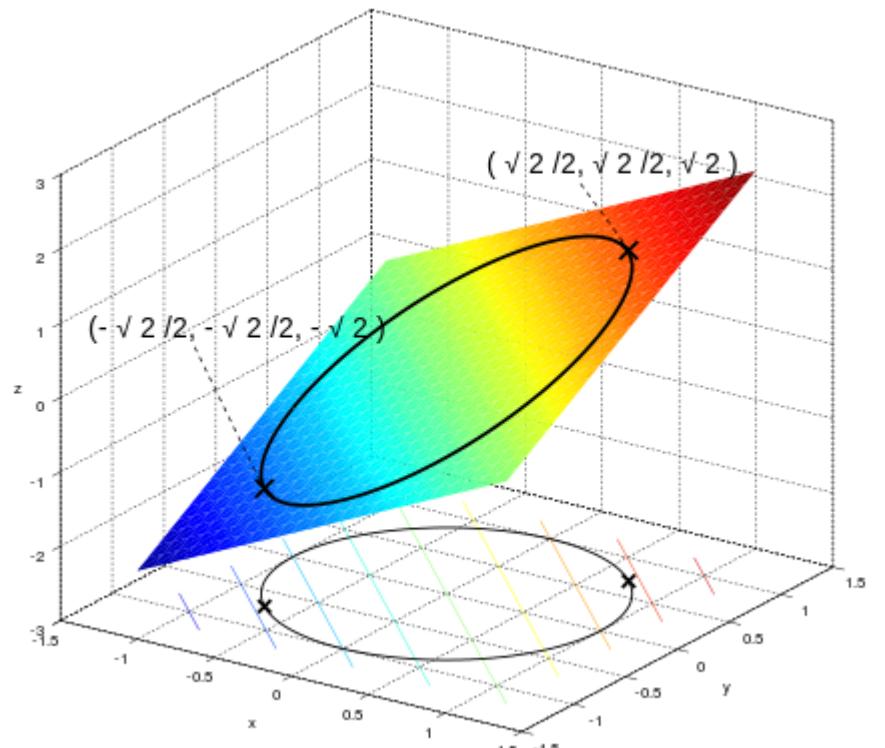


Figure obtained at [http://en.wikipedia.org/wiki/Lagrange\\_multiplier](http://en.wikipedia.org/wiki/Lagrange_multiplier)

# Convex const and Linear Objective Function

- Lagrange Multipliers
  - Now suppose a problem:

$$\begin{aligned} & \text{maximize } f(x, y) = x + y \\ & \text{subject to } x^2 + y^2 = 1 \end{aligned}$$

- So we can solve the system of equations:  $\begin{cases} \nabla f = -\lambda \nabla g \\ g(x, y) = 0 \end{cases}$
- And find:

$$\begin{cases} (1, 1) = -\lambda (2x, 2y) \\ x^2 + y^2 - 1 = 0 \end{cases}$$

- Which is the same as:

$$\begin{cases} 1 = -\lambda 2x \\ 1 = -\lambda 2y \\ x^2 + y^2 - 1 = 0 \end{cases}$$

# Convex const and Linear Objective Function

- Lagrange Multipliers
  - Solving this system of equations:

$$\begin{cases} 1 = -\lambda 2x \\ 1 = -\lambda 2y \\ x^2 + y^2 - 1 = 0 \end{cases}$$

- We find:

$$\begin{cases} \lambda 2x + 1 = 0 \Rightarrow x = -\frac{1}{2\lambda} \\ \lambda 2y + 1 = 0 \Rightarrow y = -\frac{1}{2\lambda} \\ x^2 + y^2 - 1 = 0 \Rightarrow \left(-\frac{1}{2\lambda}\right)^2 + \left(-\frac{1}{2\lambda}\right)^2 - 1 = 0 \text{ for } \lambda \neq 0 \end{cases}$$

- And solving for Lambda:

$$\lambda = \mp 1/\sqrt{2}$$

# Convex const and Linear Objective Function

- Lagrange Multipliers
  - Now we can find x and y by plugging:

$$\lambda = \mp 1/\sqrt{2}$$

- Then we have:

$$\text{for } \lambda = -\frac{1}{\sqrt{2}}$$

$$x = y = -\frac{1}{2\lambda} = \left(-\frac{1}{2}\right) \cdot \left(-\frac{\sqrt{2}}{1}\right) = \frac{\sqrt{2}}{2}$$

$$\text{and for } \lambda = \frac{1}{\sqrt{2}}$$

$$x = y = -\frac{1}{2\lambda} = -\frac{1}{2} \frac{\sqrt{2}}{1} = -\frac{\sqrt{2}}{2}$$

# Convex const and Linear Objective Function

- Lagrange Multipliers
  - As a consequence we found two points of interest:
$$(\sqrt{2}/2, \sqrt{2}/2) \text{ and } (-\sqrt{2}/2, -\sqrt{2}/2)$$
  - As the last step we plug these x,y values into f and find maxima and/or minima points

$$f(\sqrt{2}/2, \sqrt{2}/2) = \sqrt{2} \text{ and } f(-\sqrt{2}/2, -\sqrt{2}/2) = -\sqrt{2}$$

**Global Maximum  
subject to the  
constraint!**

**Global Minimum  
subject to the  
constraint!**

# Convex const and Linear Objective Function

- Lagrange Multipliers
  - Graphing the problem!

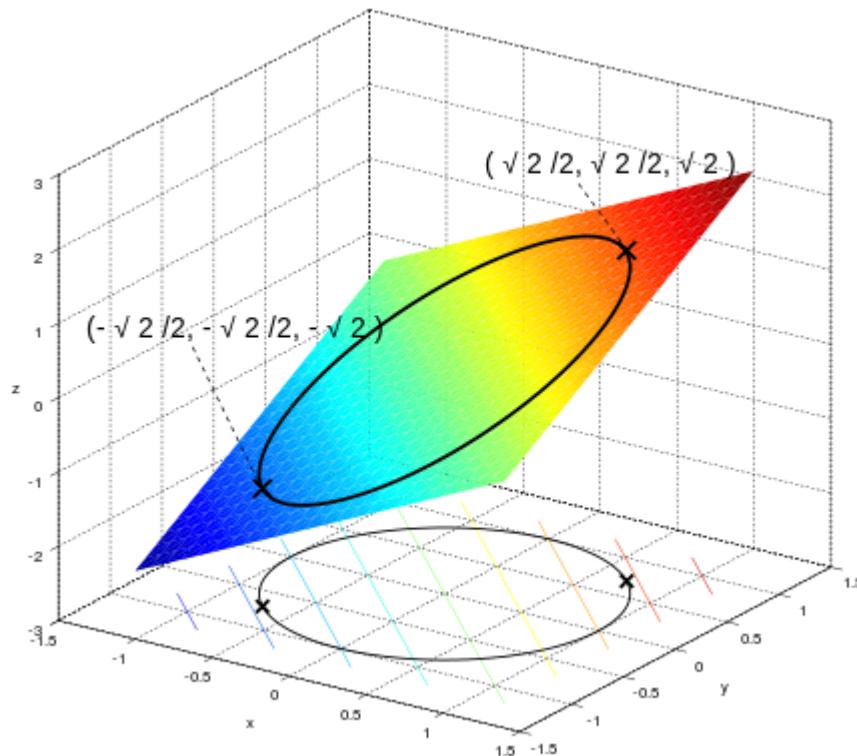


Figure obtained at [http://en.wikipedia.org/wiki/Lagrange\\_multiplier](http://en.wikipedia.org/wiki/Lagrange_multiplier)

# Convex Optimization Problems

- For **linear optimization** problems we can **graph constraints** to solve manually or algorithmically approach them using the **Simplex** method
- For convex problems we have some situations:
  - Convex constraint(s) and Linear objective function
  - Convex constraint(s) and Convex objective function

# Convex const and Convex Objective Function

- Remember the **primal** form for the SVM optimization

$$\min_{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}} \tau(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|^2$$

subject to  $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$  for all  $i = 1, \dots, m$ .

- Looking at:
  - The objective function, we see a convex function
  - The constraints, we see linear functions

# Convex const and Convex Objective Function

- Remember the **primal** form for the SVM optimization

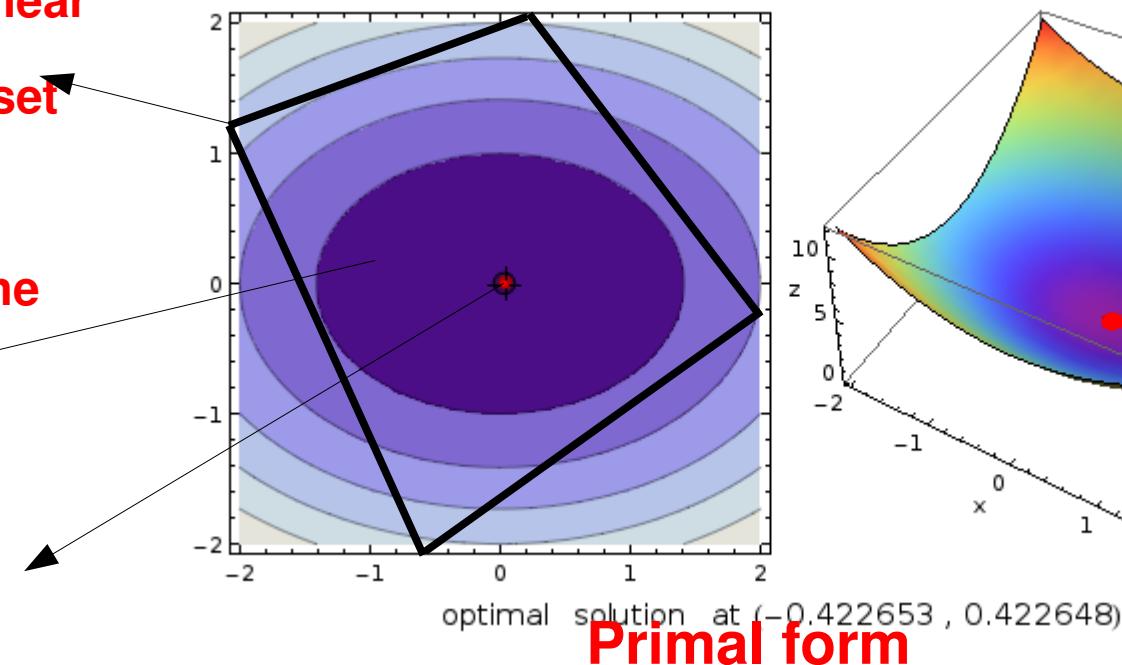
$$\min_{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}} \tau(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|^2$$

subject to  $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1$  for all  $i = 1, \dots, m$ .

Suppose these are the linear constraints  
They form a convex set

Suppose the interior is the feasible region

Here is the minimum

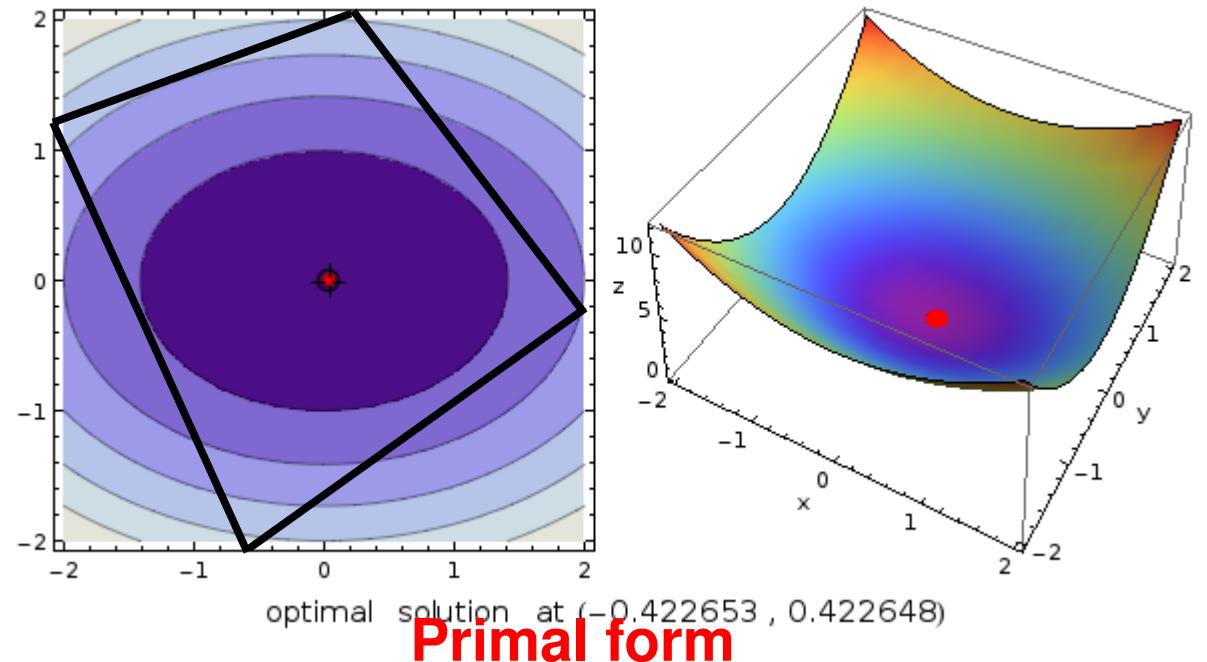


# Convex const and Convex Objective Function

- If we solve this problem using **Lagrange multipliers** and make appropriate substitutions we actually find the **dual** form as seen previously
  - Thus the problem will be converted into a **maximization** form
  - The objective function of the dual form will be **concave**

What do we get after analytically solving the Lagragian?

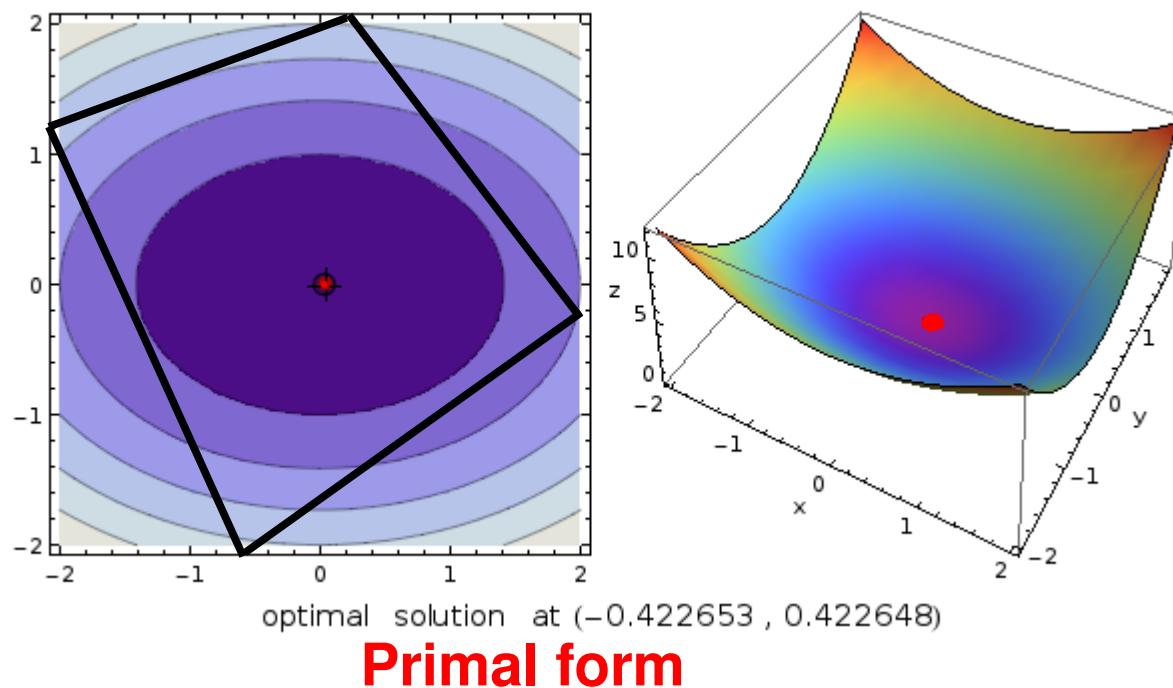
We will get points in which the gradients of constraints and the objective function are parallel



# Convex const and Convex Objective Function

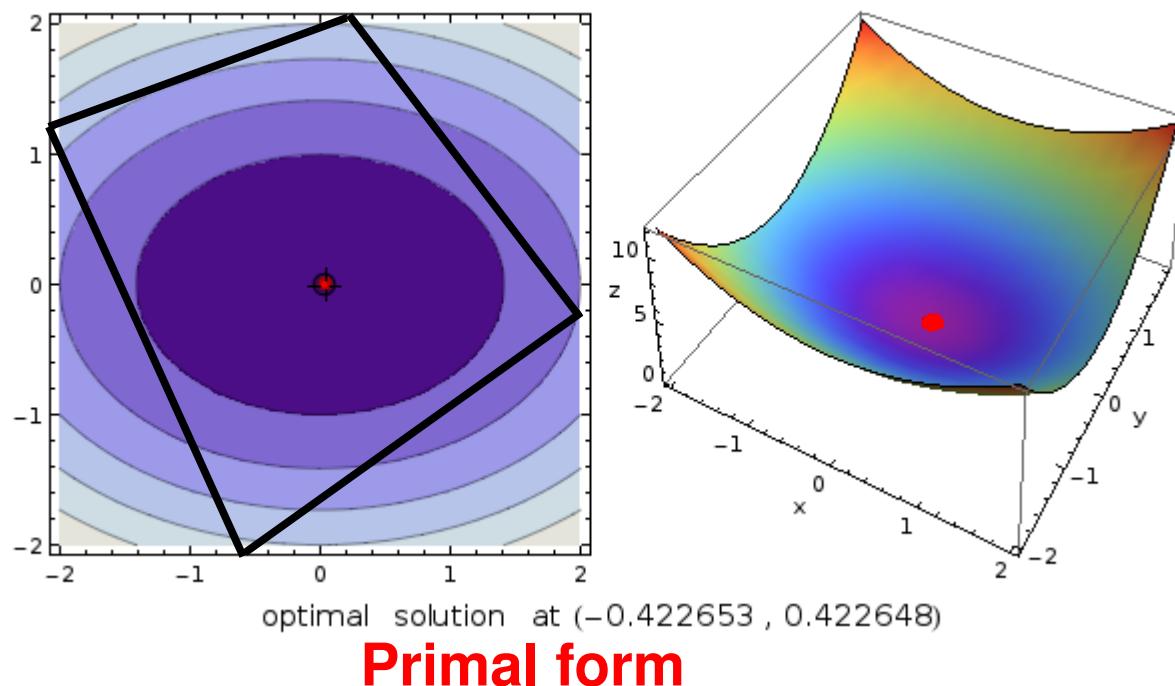
- So, solving this problem are we going to find the maximum for the dual or the minimum for the primal?

**No, in fact we will find points laying on the constraints**



# Convex const and Convex Objective Function

- So, how could we solve this problem?
  - We must **walk inside the feasible region** looking for the maximum or the minimum
    - **Primal form:** looking for the minimum
    - **Dual form:** looking for the maximum

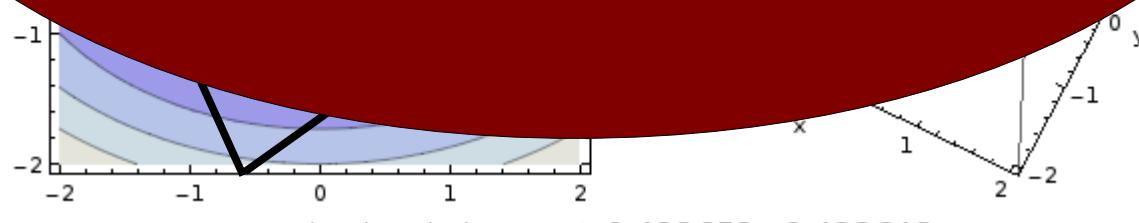


# Convex const and Convex Objective Function

- So, how could we solve this problem?
  - We must **walk inside the feasible region** looking for the maximum or the minimum.
  - Primal form
  - Dual form

**Interior Point Methods were designed  
to solve this type of problem!**

We will see that soon, but first...



**Primal form**

# **Primal and Dual Forms of Convex Problems**

# Convex Optimization Problems

- Remember the **primal** form of the SVM optimization problem:

$$\begin{aligned} \min_{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}} \quad & \tau(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|^2 \\ \text{subject to} \quad & y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 \quad \text{for all } i = 1, \dots, m. \end{aligned}$$

- How can we get the **dual form**? Let's remember...

# Convex Optimization Problems

- Remember the **primal** form of the SVM optimization problem:

$$\begin{aligned} \min_{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}} \quad & \tau(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|^2 \\ \text{subject to} \quad & y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 \quad \text{for all } i = 1, \dots, m. \end{aligned}$$

- How can we get the **dual form**? Let's remember...
  - Build the Lagrangian:

$$\Lambda(\mathbf{w}, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^m \alpha_i (y_i (\langle \mathbf{x}_i, \mathbf{w} \rangle + b) - 1)$$

# Convex Optimization Problems

- Solve the Lagragian:

$$\Lambda(\mathbf{w}, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^m \alpha_i (y_i (\langle \mathbf{x}_i, \mathbf{w} \rangle + b) - 1)$$

$$\frac{\partial \Lambda}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i = 0$$

$$\text{Thus we have: } \mathbf{w} = \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i$$

$$\frac{\partial \Lambda}{\partial b} = - \sum_{i=1}^m \alpha_i y_i = 0$$

# Convex Optimization Problems

- Open the Lagragian terms of:

$$\Lambda(\mathbf{w}, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^m \alpha_i (y_i (\langle \mathbf{x}_i, \mathbf{w} \rangle + b) - 1)$$

- In form:

$$\Lambda(\mathbf{w}, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^m \alpha_i y_i \langle \mathbf{x}_i, \mathbf{w} \rangle - b \sum_{i=1}^m \alpha_i y_i + \sum_{i=1}^m \alpha_i$$

- And make the appropriate substitutions, in this case:

$$-\sum_{i=1}^m \alpha_i y_i = 0$$

# Convex Optimization Problems

- Thus, from:

$$\Lambda(\mathbf{w}, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^m \alpha_i y_i \langle \mathbf{x}_i, \mathbf{w} \rangle - b \sum_{i=1}^m \alpha_i y_i + \sum_{i=1}^m \alpha_i$$

- We get:

$$\Lambda(\mathbf{w}, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^m \alpha_i y_i \langle \mathbf{x}_i, \mathbf{w} \rangle + \sum_{i=1}^m \alpha_i$$

- Carrying on with substitutions, we have:

$$\mathbf{w} = \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i$$

# Convex Optimization Problems

- Thus, from:

$$\Lambda(\mathbf{w}, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^m \alpha_i y_i \langle \mathbf{x}_i, \mathbf{w} \rangle + \sum_{i=1}^m \alpha_i$$

- We get:

$$\begin{aligned} \Lambda(\mathbf{w}, b, \alpha) &= \frac{1}{2} \left\langle \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i, \sum_{j=1}^m \alpha_j y_j \mathbf{x}_j \right\rangle - \sum_{i=1}^m \alpha_i y_i \left\langle \mathbf{x}_i, \sum_{j=1}^m \alpha_j y_j \mathbf{x}_j \right\rangle \\ &\quad + \sum_{i=1}^m \alpha_i \end{aligned}$$

- Now we proceed with simplifications...

# Convex Optimization Problems

- Thus, from:

$$\begin{aligned}\Lambda(\mathbf{w}, b, \alpha) = & \frac{1}{2} \left\langle \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i, \sum_{j=1}^m \alpha_j y_j \mathbf{x}_j \right\rangle - \sum_{i=1}^m \alpha_i y_i \left\langle \mathbf{x}_i, \sum_{j=1}^m \alpha_j y_j \mathbf{x}_j \right\rangle \\ & + \sum_{i=1}^m \alpha_i\end{aligned}$$

- We get:

$$\begin{aligned}\Lambda(\mathbf{w}, b, \alpha) = & \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle - \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle \\ & + \sum_{i=1}^m \alpha_i\end{aligned}$$

# Convex Optimization Problems

- And finally:

$$\Lambda(\mathbf{w}, b, \alpha) = -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^m \alpha_i$$

- Now we need to maximize this **dual** form to solve the problem

$$\underset{\alpha \in \mathbb{R}^m}{\text{Maximize}} \quad W(\alpha) = -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^m \alpha_i$$

Subject to:  $\alpha_i \geq 0$ , for  $i = 1, \dots, m$ , and  $\sum_{i=1}^m \alpha_i y_i = 0$

# Convex Optimization Problems

- And finally:

- Now

**This is the new problem or the dual problem**

$$\underset{\alpha \in \mathbb{R}^m}{\text{Maximize}} \quad W(\alpha) = -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j < \mathbf{x}_i, \mathbf{x}_j > + \sum_{i=1}^m \alpha_i$$

$$\text{Subject to: } \alpha_i \geq 0, \text{ for } i = 1, \dots, m, \text{ and } \sum_{i=1}^m \alpha_i y_i = 0$$

# Convex Optimization Problems

- How can we interpret this problem?

$$\underset{\alpha \in \mathbb{R}^m}{\text{Maximize}} \quad W(\alpha) = -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^m \alpha_i$$

Subject to:  $\alpha_i \geq 0$ , for  $i = 1, \dots, m$ , and  $\sum_{i=1}^m \alpha_i y_i = 0$

# Convex Optimization Problems

- How can we interpret this problem?

$$\underset{\alpha \in \mathbb{R}^m}{\text{Maximize}} \quad W(\alpha) = -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^m \alpha_i$$

Subject to:  $\alpha_i \geq 0$ , for  $i = 1, \dots, m$ , and  $\sum_{i=1}^m \alpha_i y_i = 0$

- As advantage we only have alphas instead of two other variables
  - So we simplified the problem

# Convex Optimization Problems

- How can we interpret this problem?

$$\underset{\alpha \in \mathbb{R}^m}{\text{Maximize}} \quad W(\alpha) = -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^m \alpha_i$$

Subject to:  $\alpha_i \geq 0$ , for  $i = 1, \dots, m$ , and  $\sum_{i=1}^m \alpha_i y_i = 0$

- As advantage we only have alphas instead of two other variables

- So we simplified the problem
- This term must form a concave function

# Convex Optimization Problems

- How can we interpret this problem?

$$\underset{\alpha \in \mathbb{R}^m}{\text{Maximize}} \quad W(\alpha) = -\frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + \sum_{i=1}^m \alpha_i$$

Subject to:  $\alpha_i \geq 0$ , for  $i = 1, \dots, m$ , and  $\sum_{i=1}^m \alpha_i y_i = 0$

- As advantage we only have alphas instead of two other variables
  - So we simplified the problem
  - This term must form a concave function
    - But how to guarantee that? Well, we will see later on...

# Convex Optimization Problems

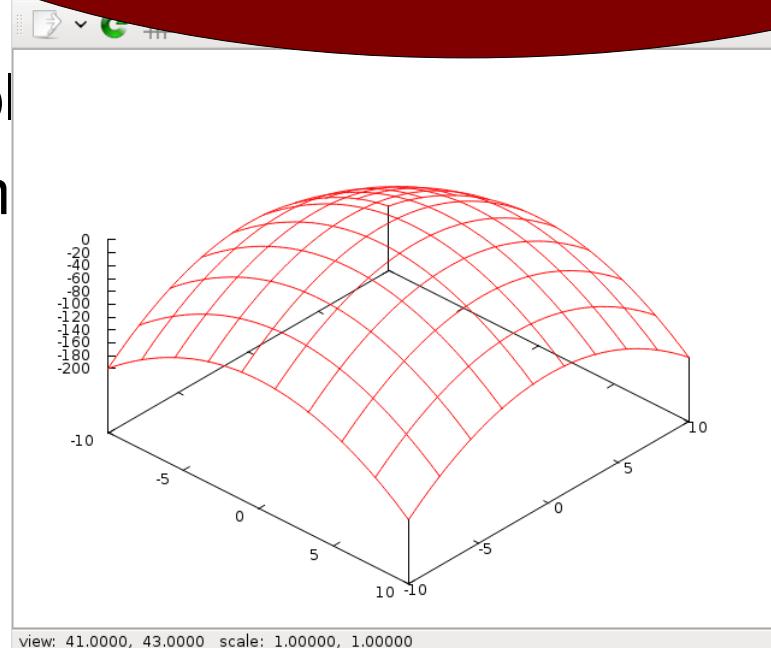
- How can we interpret this?
- Maximize  $\alpha^T b$   
Subject to  $\sum_{i=1}^m \alpha_i = 1$

Now it makes sense to maximize this function!

- As advantages of convex variables

- So we simply add this term
- This term means

  - But how



will see later on...

# **Interior Point Methods**

## **Algorithms to Solve Convex Optimization Problems**

# Interior Point Methods

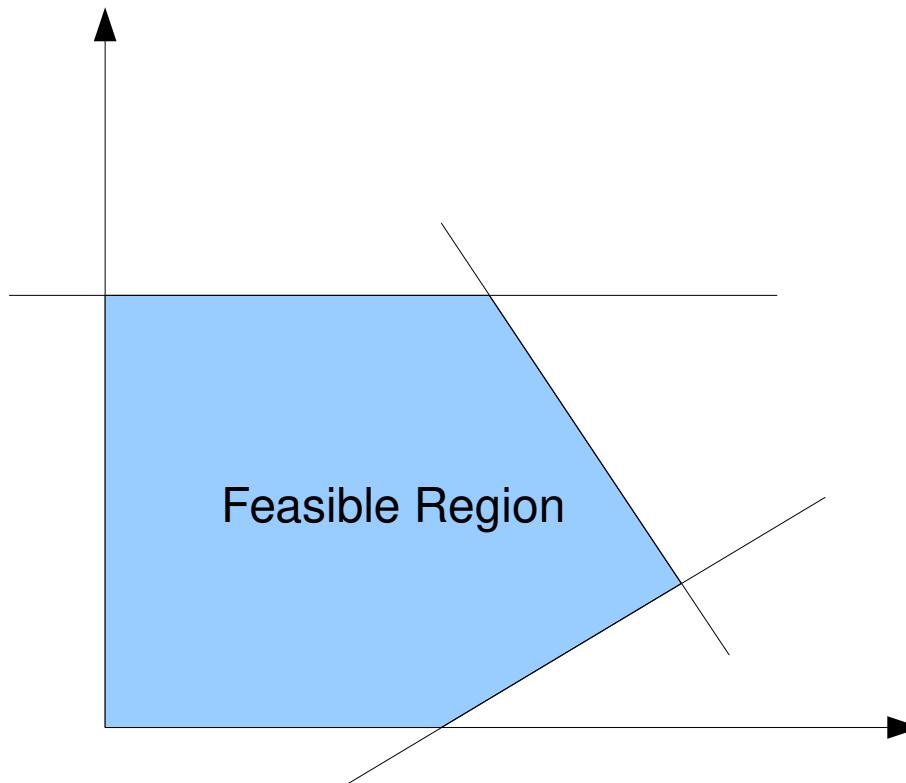
- In the late 1940's, almost at the same time Dantzig proposed the Simplex method, other researchers proposed interior point methods
  - Among those researchers were:
    - Von Neumann (1947)
    - Hoffman et al. (1953)
    - Frisch (1955)
  - IPM would transverse across the interior of the feasible region in attempt to avoid the combinatorial complexities of vertex-following algorithms (such as Simplex)

# Interior Point Methods

- However, IPM required expensive computational steps and suffered of numerical instabilities
  - Discouraging the adoption of IPM in practical scenarios
- Then, in 1984, Karmarkar presented a novel interior point method which motivated the adoption of such type of approach in practical scenarios
  - Gill et al. then showed a formal relationship between the new interior point method and the classical logarithmic barrier method

# Interior Point Methods

- While Simplex verifies extreme points of the feasible region, IPM moves through the interior points of the feasible region



- There are three major types of IPMs:
  - The **potential reduction algorithm** which most closely embodies the proposal by Karmarkar
  - The **affine scaling algorithm** which is probably the simplest to implement
  - **Path-following algorithms** which arguably combine excellent behavior in theory and practice
- In here, we will approach an interior point method of the third type called **primal-dual path following algorithm**
  - This is the mostly used algorithm for large scale implementations

- First of all consider the primal and dual forms for the same problem:

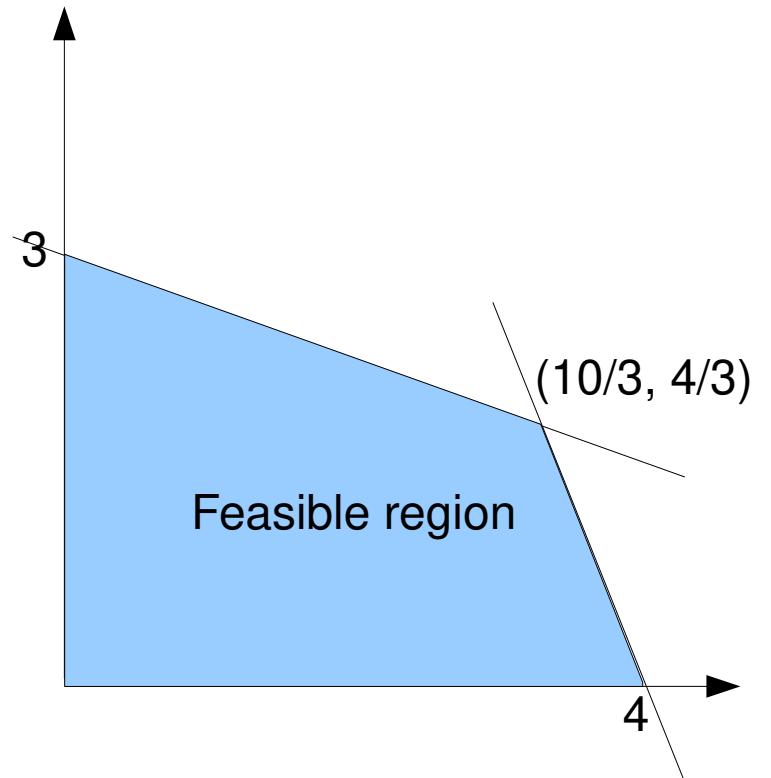
## Primal

$$\text{Maximize } z_p = 2x_1 + 3x_2$$

$$2x_1 + x_2 \leq 8$$

$$\text{Subject to: } x_1 + 2x_2 \leq 6$$

$$x_1, x_2 \geq 0$$



- First of all consider the primal and dual forms for the same problem:

## Primal

$$\text{Maximize } z_p = 2x_1 + 3x_2$$

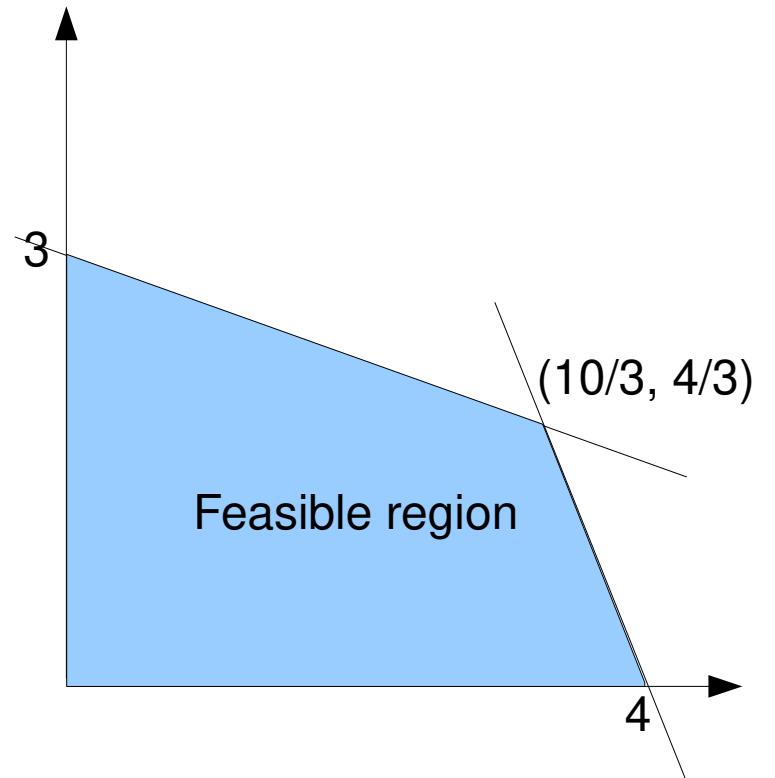
$$2x_1 + x_2 \leq 8$$

$$\text{Subject to: } x_1 + 2x_2 \leq 6$$

$$x_1, x_2 \geq 0$$

Here we can use Simplex, because the problem is linear!

What would Simplex do?



- First of all consider the primal and dual forms for the same problem:

## Primal

$$\text{Maximize } z_p = 2x_1 + 3x_2$$

$$2x_1 + x_2 \leq 8$$

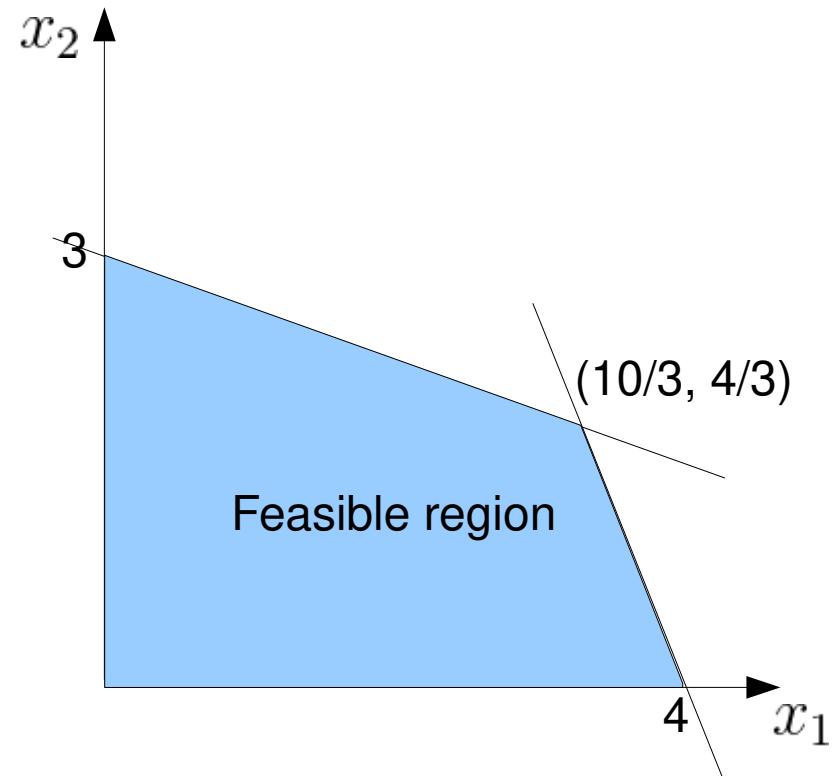
$$\text{Subject to: } x_1 + 2x_2 \leq 6$$

$$x_1, x_2 \geq 0$$

$$2(4) + 3(0) = 8$$

$$2(0) + 3(3) = 9$$

$$2(10/3) + 3(4/3) = 10.66\dots$$



- First of all consider the primal and dual forms for the same problem:

## Dual

Minimize  $z_d = 8\pi_1 + 6\pi_2$

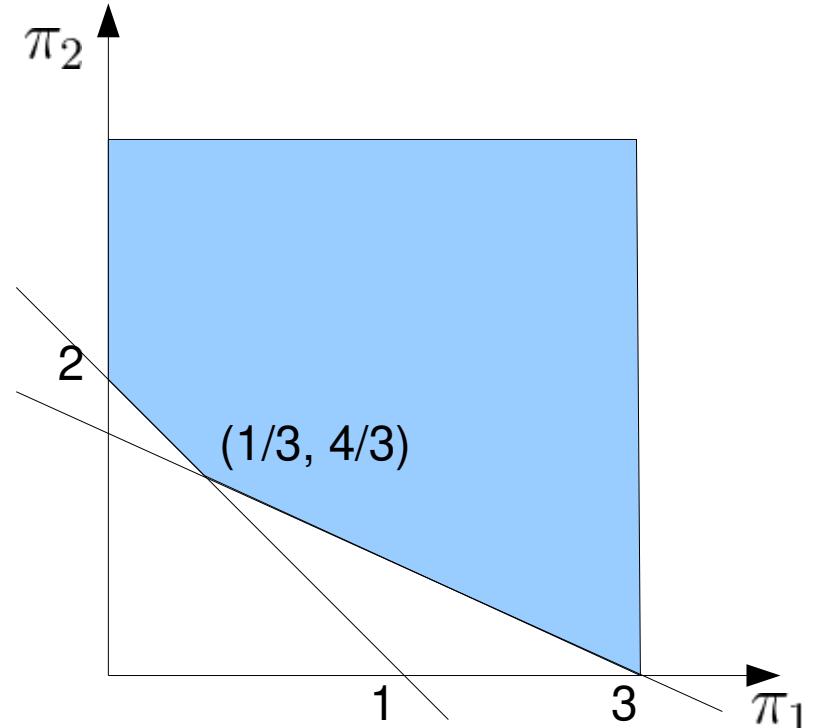
Subject to:

$$2\pi_1 + \pi_2 \geq 2$$

$$\pi_1 + 2\pi_2 \geq 3$$

$$\pi_1, \pi_2 \geq 0$$

We can also use linear methods, i.e., verifying the possible vertices



- First of all consider the primal and dual forms for the same problem:

## Dual

Minimize  $z_d = 8\pi_1 + 6\pi_2$

Subject to:

$$2\pi_1 + \pi_2 \geq 2$$

$$\pi_1 + 2\pi_2 \geq 3$$

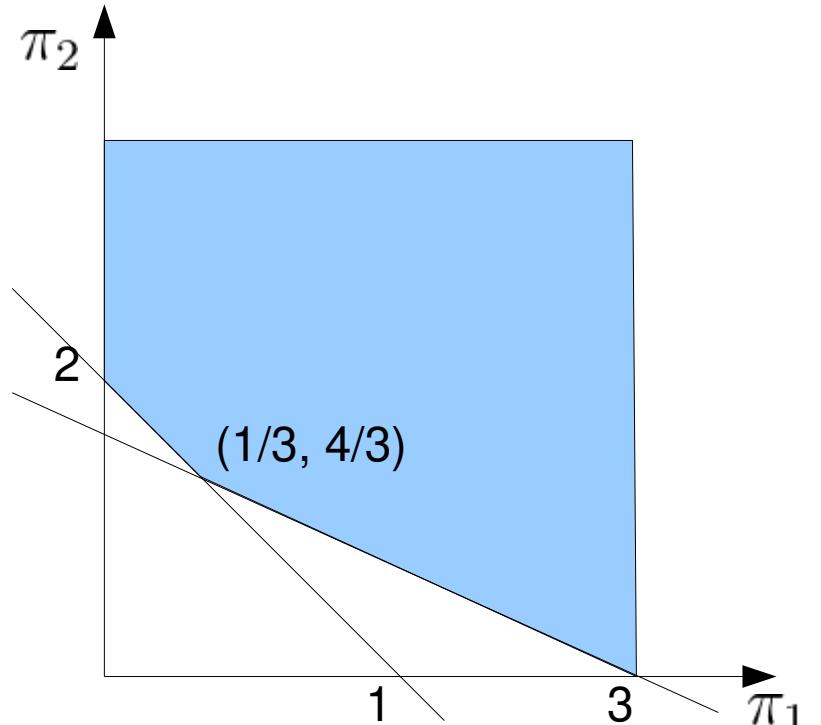
$$\pi_1, \pi_2 \geq 0$$

**8 (3) + 6 (0) = 24**

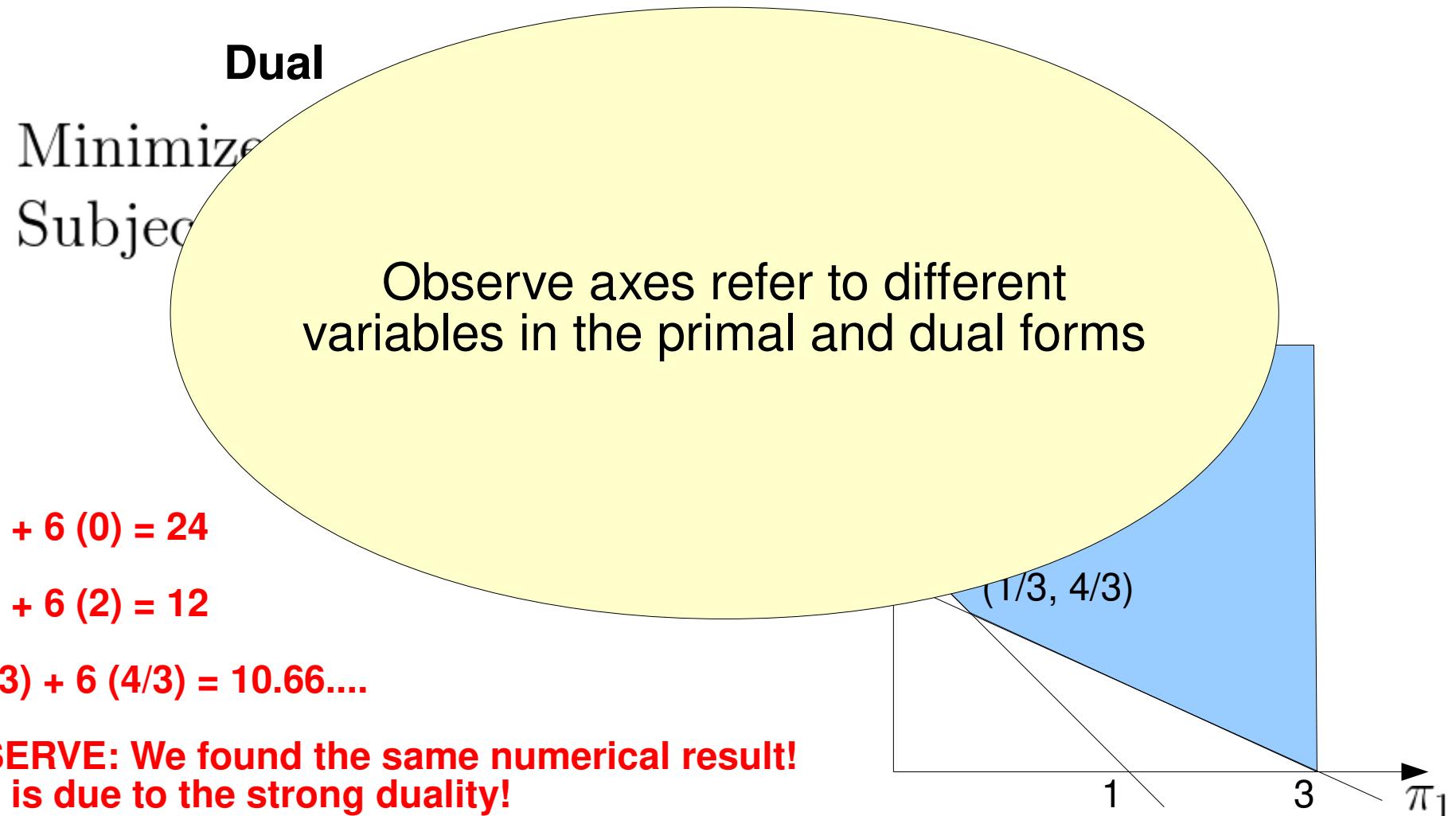
**8 (0) + 6 (2) = 12**

**8 (1/3) + 6 (4/3) = 10.66....**

**OBserve: We found the same numerical result!  
This is due to the strong duality!**



- First of all consider the primal and dual forms for the same problem:



# **Now we start the Primal-Dual Path Following Interior Point Method**

Based on the book chapter:

Jensen, P. A.; Bard, J. F. (2002) Operations Research Models and Methods. Chapter: LP Methods.S4 - Interior Point Methods. Wiley. ISBN: 978-0-471-38004-7

- As first step we have the primal and the dual forms of our problem as follows:

## Primal

$$\begin{aligned} \text{Maximize } z_p &= 2x_1 + 3x_2 \\ &2x_1 + x_2 \leq 8 \end{aligned}$$

$$\begin{aligned} \text{Subject to: } x_1 + 2x_2 &\leq 6 \\ x_1, x_2 &\geq 0 \end{aligned}$$

## Dual

$$\begin{aligned} \text{Minimize } z_d &= 8\pi_1 + 6\pi_2 \\ \text{Subject to: } & \end{aligned}$$

$$\begin{aligned} 2\pi_1 + \pi_2 &\geq 2 \\ \pi_1 + 2\pi_2 &\geq 3 \\ \pi_1, \pi_2 &\geq 0 \end{aligned}$$

The first step is to rewrite both forms using slack variables similarly to the Simplex method!

# Interior Point Methods

- After rewriting this problem, we have the inequality bounds defined in terms of slack variables as follows:

## Primal

$$\text{Maximize } z_p = 2x_1 + 3x_2$$

Subject to:

$$2x_1 + x_2 + \boxed{x_3} = 8$$

$$x_1 + 2x_2 + \boxed{x_4} = 6$$

$$x_1, x_2, x_3, x_4 \geq 0$$

## Dual

$$\text{Minimize } z_d = 8\pi_1 + 6\pi_2$$

Subject to:

$$2\pi_1 + \pi_2 - \boxed{z_1} = 2$$

$$\pi_1 + 2\pi_2 - \boxed{z_2} = 3$$

$$\pi_1 - \boxed{z_3} = 0$$

$$\pi_2 - \boxed{z_4} = 0$$

$$z_1, z_2, z_3, z_4 \geq 0$$

Observe the slack variables!

What do they do?

- After rewriting this problem, we have the inequality bounds defined in terms of slack variables as follows:

## Primal

$$\text{Maximize } z_p = 2x_1 + 3x_2$$

Subject to:

$$2x_1 + x_2 + \boxed{x_3} = 8$$

$$x_1 + 2x_2 + \boxed{x_4} = 6$$

$$x_1, x_2, x_3, x_4 \geq 0$$

## Dual

$$\text{Minimize } z_d = 8\pi_1 + 6\pi_2$$

Subject to:

$$2\pi_1 + \pi_2 - \boxed{z_1} = 2$$

$$\pi_1 + 2\pi_2 - \boxed{z_2} = 3$$

$$\pi_1 - \boxed{z_3} = 0$$

$$\pi_2 - \boxed{z_4} = 0$$

$$z_1, z_2, z_3, z_4 \geq 0$$

Observe z variables are subtracted, but why? Because if we subtract the greatest possible value we will end up having an equality where there was an inequality of the type greater or equal to

- After rewriting this problem, we have the inequality bounds defined in terms of slack variables as follows:

## Primal

$$\text{Maximize } z_p = 2x_1 + 3x_2$$

Subject to:

$$2x_1 + x_2 + \boxed{x_3} = 8$$

$$x_1 + 2x_2 + \boxed{x_4} = 6$$

$$x_1, x_2, x_3, x_4 \geq 0$$

## Dual

$$\text{Minimize } z_d = 8\pi_1 + 6\pi_2$$

Subject to:

$$2\pi_1 + \pi_2 - \boxed{z_1} = 2$$

$$\pi_1 + 2\pi_2 - \boxed{z_2} = 3$$

$$\pi_1 - \boxed{z_3} = 0$$

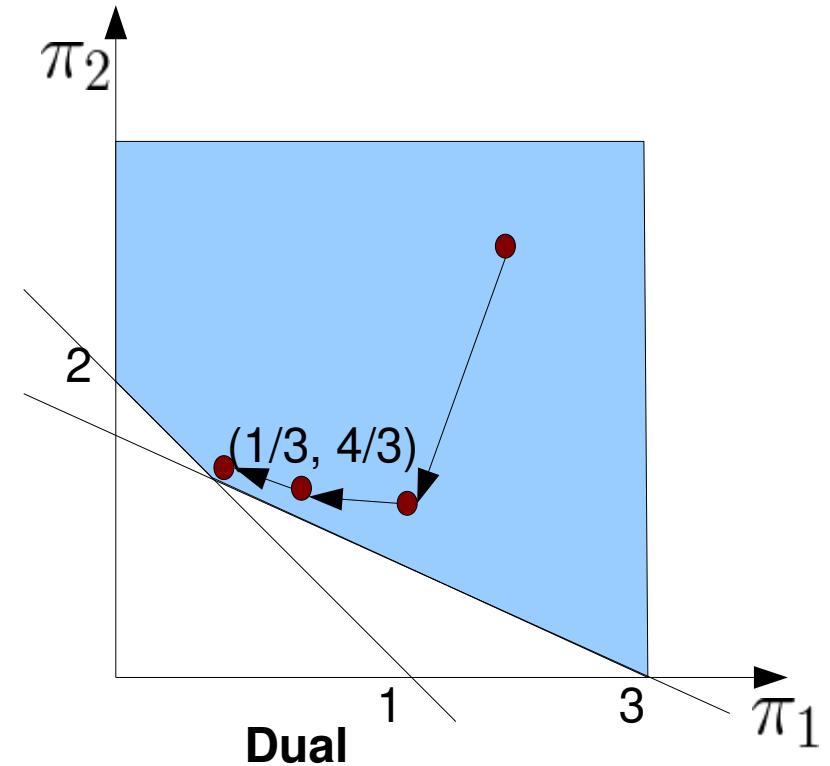
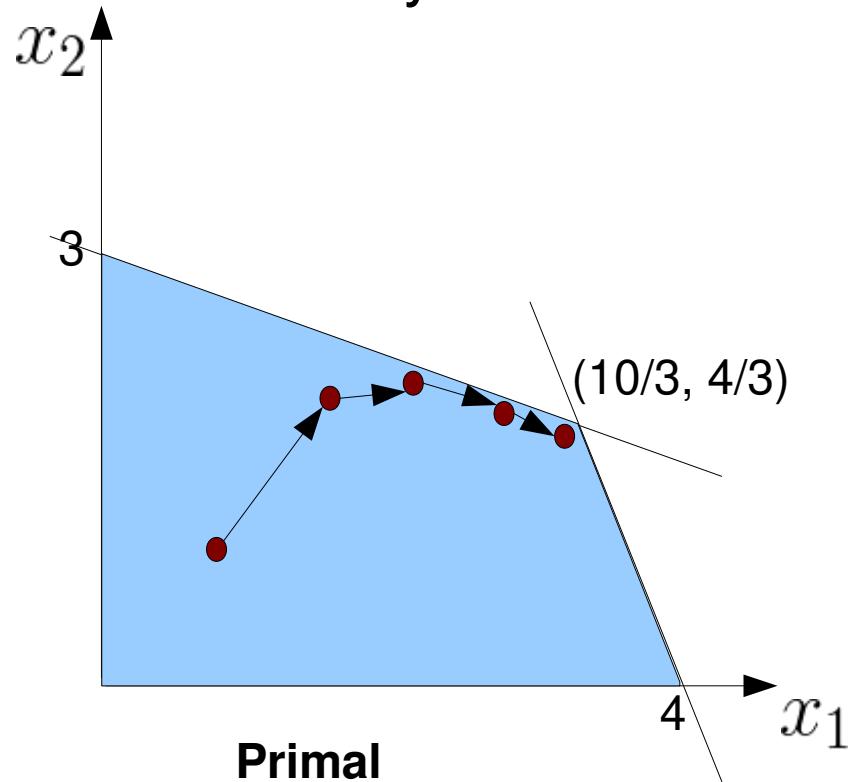
$$\pi_2 - \boxed{z_4} = 0$$

$$z_1, z_2, z_3, z_4 \geq 0$$

The opposite occurs for  $x_3$  and  $x_4$ , they are summed up because the inequalities were of the type less than or equal to

# Interior Point Methods

- Instead of walking on the boundaries as using the **Simplex method**, the IPM walks inside the feasible region
  - In this case the best solutions is in the boundary, because the problem is linear
    - So IPM goes towards it but never touches the boundary
      - Why? Barriers! We will see that...



# Interior Point Methods

- Before coming back and solve our primal-dual problem, we will first see a **generic formulation** for the primal-dual path following algorithm

# Interior Point Methods

- Before coming back and solve our primal-dual problem, we will first see a **generic formulation** for the primal-dual path following algorithm
- So, consider the **generic problem** in its primal and dual forms after plugging the slack variables  $z$

## Primal

$$\text{Maximize } z_p = \mathbf{c}\mathbf{x}$$

$$\begin{aligned} \text{Subject to: } & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0 \end{aligned}$$

## Dual

$$\text{Minimize } z_d = \boldsymbol{\pi}\mathbf{b}$$

$$\begin{aligned} \text{Subject to: } & \boldsymbol{\pi}\mathbf{A} - \mathbf{z} = \mathbf{c} \\ & \mathbf{z} \geq 0 \end{aligned}$$

- Before coming back and solve our primal-dual problem, we will first see a **generic formulation** for the primal-dual path following algorithm
- So, consider the **generic problem** in its primal and dual forms after plugging the slack variables  $z$

## Primal

$$\text{Maximize } z_p = \mathbf{c}\mathbf{x}$$

$$\begin{aligned} \text{Subject to: } & \mathbf{A}\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0 \end{aligned}$$

## Dual

$$\text{Minimize } z_d = \boldsymbol{\pi}\mathbf{b}$$

$$\begin{aligned} \text{Subject to: } & \boldsymbol{\pi}\mathbf{A} - \mathbf{z} = \mathbf{c} \\ & \mathbf{z} \geq 0 \end{aligned}$$

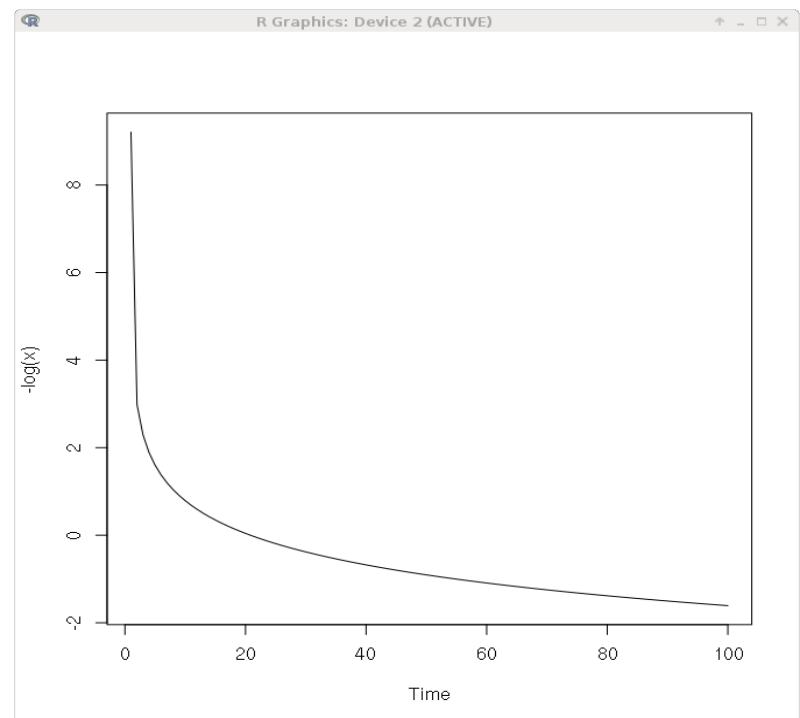
**After having the problem in primal and dual forms using slack variables, we then include the barriers**

- What are the barriers?
  - They are a way to guarantee we strictly move using the interior points defined by inequalities
  - So, we never reach a boundary
    - There are risks in reaching the boundary, because we might cross it
      - Once we cross it, we do not hold constraints anymore

# Interior Point Methods

- Fiacco and McCormick (1968) studied the barrier approach and designed it using logarithmic functions to avoid reaching a boundary
  - For example, suppose we are solving a **minimization** problem and there is a boundary as close as we get of value zero for the feasible set

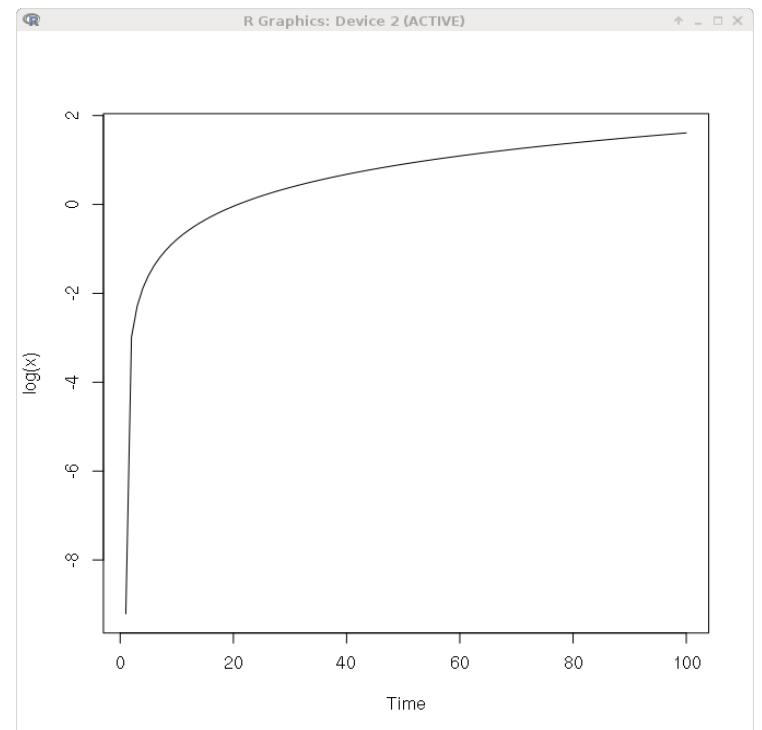
So, we will add this term to jeopardize the minimization in attempt to avoid the boundary



# Interior Point Methods

- Fiacco and McCormick (1968) studied the barrier approach and designed it using logarithmic functions to avoid reaching a boundary
  - For example, suppose we are solving a **maximization** problem and there is a boundary as close as we get of value zero for the feasible set

So, we will add this term to jeopardize the maximization in attempt to avoid the boundary



- So now we have the primal and the dual problems in terms of **barriers** as follows:

## Primal

$$\text{Maximize } \mathbf{c}\mathbf{x} + \mu \sum_{j=1}^n \log x_j$$

Subject to:  $\mathbf{A}\mathbf{x} = \mathbf{b}$

## Dual

$$\text{Minimize } \pi\mathbf{b} - \mu \sum_{j=1}^n \log z_j$$

Subject to:  $\pi\mathbf{A} - \mathbf{z} = \mathbf{c}$

**Parameter  $\mu$  is required to be positive to control the magnitude of the barrier term!**

**This parameter  $\mu$  should start with a value and decrease along iterations, what may allows us to get close enough to boundary points without crossing it.**

**Now we have the problem set!**

**Thus we need to solve it.**

# Interior Point Methods

- Once the problem is set we need to solve it
  - In order to solve it we use Lagrange multipliers (**we have equality constraints**)

**Primal**

$$\text{Maximize } \mathbf{c}\mathbf{x} + \mu \sum_{j=1}^n \log x_j$$

Subject to:  $\mathbf{A}\mathbf{x} = \mathbf{b}$

**Dual**

$$\text{Minimize } \pi\mathbf{b} - \mu \sum_{j=1}^n \log z_j$$

Subject to:  $\pi\mathbf{A} - \mathbf{z} = \mathbf{c}$

# Interior Point Methods

- Once the problem is set we need to solve it
  - In order to solve it we use Lagrange multipliers (**we have equality constraints**)

**Primal**

$$\text{Maximize } \mathbf{c}\mathbf{x} + \mu \sum_{j=1}^n \log x_j$$

Subject to:  $\mathbf{A}\mathbf{x} = \mathbf{b}$

**Dual**

$$\text{Minimize } \pi\mathbf{b} - \mu \sum_{j=1}^n \log z_j$$

Subject to:  $\pi\mathbf{A} - \mathbf{z} = \mathbf{c}$

**So, from this constrained problem version we get an unconstrained version as follows...**

# Interior Point Methods

- The unconstrained version is built using Lagrange multipliers as follows:

From:

Primal	Dual
Maximize $\mathbf{c}\mathbf{x} + \mu \sum_{j=1}^n \log x_j$	Minimize $\pi\mathbf{b} - \mu \sum_{j=1}^n \log z_j$
Subject to: $\mathbf{A}\mathbf{x} = \mathbf{b}$	Subject to: $\pi\mathbf{A} - \mathbf{z} = \mathbf{c}$

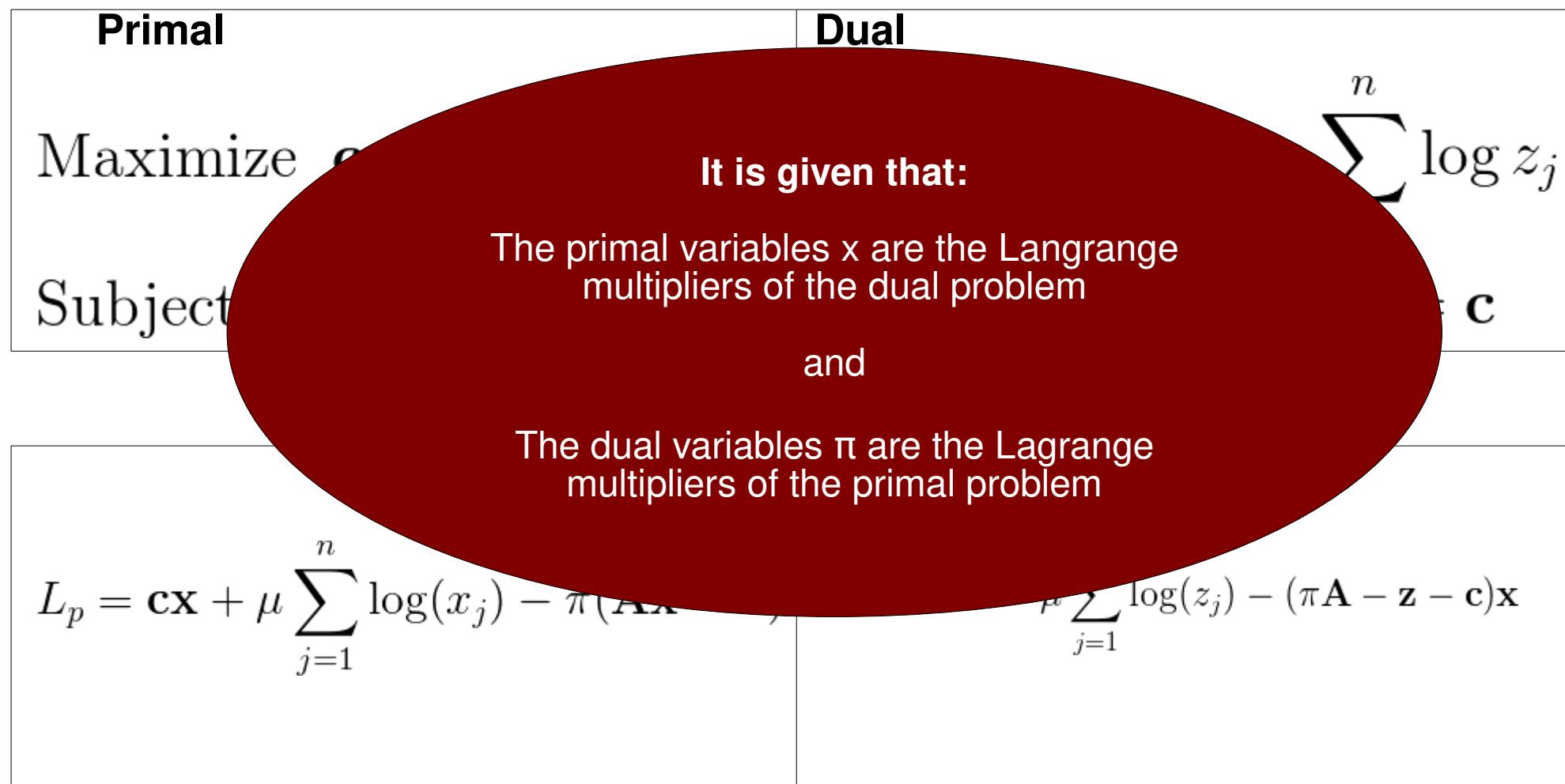
We get:

$L_p = \mathbf{c}\mathbf{x} + \mu \sum_{j=1}^n \log(x_j) - \pi(\mathbf{A}\mathbf{x} - \mathbf{b})$	$L_D = \pi\mathbf{b} - \mu \sum_{j=1}^n \log(z_j) - (\pi\mathbf{A} - \mathbf{z} - \mathbf{c})\mathbf{x}$
--	--

# Interior Point Methods

- The unconstrained version is built using Lagrange multipliers as follows:

From:



# Interior Point Methods

- To find the solution, we derive the Lagrangian according to variables to get stable points:

## Primal

$$L_p = \mathbf{c}\mathbf{x} + \mu \sum_{j=1}^n \log(x_j) - \pi(\mathbf{A}\mathbf{x} - \mathbf{b})$$

## Dual

$$L_D = \pi\mathbf{b} - \mu \sum_{j=1}^n \log(z_j) - (\pi\mathbf{A} - \mathbf{z} - \mathbf{c})\mathbf{x}$$

$$\frac{\partial L_p}{\partial x_j} = 0 \text{ and } \frac{\partial L_p}{\partial \pi_i} = 0$$

$$\frac{\partial L_D}{\partial z_j} = 0, \quad \frac{\partial L_D}{\partial \pi_i} = 0 \text{ and } \frac{\partial L_D}{\partial x_j} = 0$$

# Interior Point Methods

- To find the solution, we derive the Lagrangian according to variables to get stable points:

**Primal**

$$L_p = \mathbf{c}\mathbf{x} + \mu \sum_{j=1}^n l_j$$

**Dual**

$$(\pi\mathbf{A} - \mathbf{z} - \mathbf{c})\mathbf{x}$$

Thus, we have...

$$\frac{\partial L_p}{\partial x_j}$$

$$\frac{\partial L_D}{\partial x_j} = 0$$

# Interior Point Methods

- To find the solution, we derive the Lagrangian according to variables to get stable points:

## Primal

$$L_p = \mathbf{c}\mathbf{x} + \mu \sum_{j=1}^n \log(x_j) - \pi(\mathbf{A}\mathbf{x} - \mathbf{b})$$

## Dual

$$L_D = \pi\mathbf{b} - \mu \sum_{j=1}^n \log(z_j) - (\pi\mathbf{A} - \mathbf{z} - \mathbf{c})\mathbf{x}$$

$$\frac{\partial L_p}{\partial x_j} = 0$$

$$c_j - \sum_{i=1}^m a_{ij}\pi_j + \frac{\mu}{x_j} = 0$$

$$\frac{\partial L_D}{\partial z_j} = 0$$

$$-\frac{\mu}{z_j} + x_j = 0$$

$$z_j x_j = \mu, \quad j = 1, \dots, n$$

# Interior Point Methods

- To find the solution, we derive the Lagrangian according to variables to get stable points:

Primal	Dual
$L_p = \mathbf{c}\mathbf{x} - \mu \sum_{j=1}^n \log(x_j)$	$\pi\mathbf{b} - \mu \sum_{j=1}^n \log(z_j) - (\pi\mathbf{A} - \mathbf{z} - \mathbf{c})\mathbf{x}$

$$\frac{\partial L_p}{\partial x_j} = 0$$
$$c_j - \sum_{i=1}^m a_{ij}\pi_i + \frac{\mu}{x_j} = 0$$

$$\frac{\partial L_D}{\partial z_j} = 0$$
$$-\frac{\mu}{z_j} + x_j = 0$$
$$z_j x_j = \mu, \quad j = 1, \dots, n$$

# Interior Point Methods

$z_j x_j = \mu$  So we must guarantee:

$$\text{Therefore: } c_j - \sum_{i=1}^m a_{ij}\pi_j + \frac{\mu}{x_j} = 0$$

$$\text{we substitute } \mu \text{ as follows: } c_j - \sum_{i=1}^m a_{ij}\pi_j + \frac{z_j x_j}{x_j} = 0$$

$$\text{what makes it equal to: } c_j - \sum_{i=1}^m a_{ij}\pi_j + z_j = 0$$

according to

$$(z_j) - (\pi \mathbf{A} - \mathbf{z} - \mathbf{c})\mathbf{x}$$

$$c_j - \sum_{i=1}^m a_{ij}\pi_j + \frac{\mu}{x_j} = 0$$

$$\begin{aligned} \frac{\partial L_D}{\partial z_j} &= 0 \\ -\frac{\mu}{z_j} + x_j &= 0 \\ z_j x_j &= \mu, \quad j = 1, \dots, n \end{aligned}$$

# Interior Point Methods

- To find the solution, we solve the system of equations according to the variables  $x$ ,  $\pi$ ,  $z$ .

These are called the  $\mu$ -complementary slackness

They must agree in both problem forms

Primal

$$L_p = \mathbf{c}\mathbf{x}$$

$$\lambda - \mathbf{z} - \mathbf{c})\mathbf{x}$$

$$\frac{\partial L_p}{\partial x_j} = 0$$

$$c_j - \sum_{i=1}^m a_{ij}\pi_j + \frac{\mu}{x_j} = 0$$

$$c_j - \sum_{i=1}^m a_{ij}\pi_j + z_j = 0$$

$$\frac{\partial L_D}{\partial z_j} = 0$$

$$-\frac{\mu}{z_j} + x_j = 0$$

$$z_j x_j = \mu, \quad j = 1, \dots, n$$

# Interior Point Methods

- To find the solution we solve the system of equations according to the variables

Primal

$$L_p = \mathbf{c}\mathbf{x}$$

We intend to make  $\mu$  as close as possible to zero after a number of iterations.

So, the difference or **gap** in between the Primal and the Dual solutions is given

by 
$$\text{Gap} = \sum_{j=1}^n z_j x_j$$

$$\frac{\partial L_p}{\partial x_j} = 0$$

$$c_j - \sum_{i=1}^m a_{ij} \pi_j + \frac{\mu}{x_j} = 0$$

$$c_j - \sum_{i=1}^m a_{ij} \pi_j + z_j = 0$$

$$\frac{\partial L_D}{\partial z_j} = 0$$

$$-\frac{\mu}{z_j} + x_j = 0$$

$$z_j x_j = \mu, \quad j = 1, \dots, n$$

# Interior Point Methods

- To find the solution we have to minimize the barrier function according to the variables

Primal

$$L_p = \mathbf{c}x$$

We intend to make  $\mu$  as close as possible to zero after a number of iterations.

So, the difference or **gap** in between the Primal and the Dual solutions is given

$$- \mathbf{z} - \mathbf{c})x$$

So why don't we set  $\mu=0$  since the beginning?

Because we will not give any relevance to barrier and, consequently, we may cross the constraints and may find solutions outside the feasible region!

# Interior Point Methods

- To find the solution, we derive the Lagrangian according to variables to get stable points:

## Primal

$$L_p = \mathbf{c}\mathbf{x} + \mu \sum_{j=1}^n \log(x_j) - \pi(\mathbf{A}\mathbf{x} - \mathbf{b})$$

## Dual

$$L_D = \pi\mathbf{b} - \mu \sum_{j=1}^n \log(z_j) - (\pi\mathbf{A} - \mathbf{z} - \mathbf{c})\mathbf{x}$$

$$\frac{\partial L_p}{\partial \pi_i} = 0$$

$$\sum_{j=1}^n a_{ij}x_j = b_i, \quad i = 1, \dots, m$$

$$\frac{\partial L_D}{\partial \pi_i} = 0$$

$$\sum_{j=1}^n a_{ij}x_j = b_i, \quad i = 1, \dots, m$$

# Interior Point Methods

- To find the solution, we have to minimize the objective function according to variables  $x$

This is usually called the primal feasibility, once the primal constraint is

$$Ax = b$$

Primal

$$L_p = \mathbf{c}\mathbf{x}$$

$$(\mathbf{z} - \mathbf{c})\mathbf{x}$$

We still have one more derivative...

$$\frac{\partial L_p}{\partial \pi_i} = 0$$

$$\sum_{j=1}^n a_{ij}x_j = b_i, \quad i = 1, \dots, m$$

$$\frac{\partial L_D}{\partial \pi_i} = 0$$

$$\sum_{j=1}^n a_{ij}x_j = b_i, \quad i = 1, \dots, m$$

# Interior Point Methods

- To find the solution, we derive the Lagrangian according to variables to get stable points:

## Primal

$$L_p = \mathbf{c}\mathbf{x} + \mu \sum_{j=1}^n \log(x_j) - \pi(\mathbf{A}\mathbf{x} - \mathbf{b})$$

## Dual

$$L_D = \pi\mathbf{b} - \mu \sum_{j=1}^n \log(z_j) - (\pi\mathbf{A} - \mathbf{z} - \mathbf{c})\mathbf{x}$$

$$\frac{\partial L_D}{\partial x_j} = 0$$

$$\sum_{i=1}^m a_{ij}\pi_j - z_j = c_j, \quad j = 1, \dots, n$$

# Interior Point Methods

- To find the solution, we have to solve the system of linear equations for the variables  $x_1, x_2, \dots, x_n$ .

**Primal**

$$L_p = \mathbf{c}\mathbf{x} + \mathbf{b}$$

This one is called the dual feasibility because it provides the dual constraint!

$$- \mathbf{c})\mathbf{x}$$

$$\frac{\partial L_D}{\partial x_j} = 0$$

$$\sum_{i=1}^m a_{ij} \pi_i - z_j = c_j, \quad j = 1, \dots, n$$

**What can we conclude after Lagrange multipliers?**

- To solve this problem we simply have **to guarantee the solution for every equation we got after derivation**
- So we build a system of equations
  - To simplify we define two  $n \times n$  diagonal matrices containing the elements of  $\mathbf{x}$  and  $\mathbf{z}$  in form:
$$\mathbf{X} = \text{diag}\{x_1, x_2, \dots, x_n\}$$
$$\mathbf{Z} = \text{diag}\{z_1, z_2, \dots, z_n\}$$
  - And let  $\mathbf{e} = (1, 1, \dots, 1)^T$  be a column vector of length  $n$

# Interior Point Methods

- So we have the following system of equations to solve:

$$\begin{cases} \mathbf{Ax} - \mathbf{b} = 0 & \text{Primal feasibility} \\ \mathbf{A}^T \boldsymbol{\pi}^T - \mathbf{z} - \mathbf{c}^T = 0 & \text{Dual feasibility} \\ \mathbf{XZ}\mathbf{e} - \mu\mathbf{e} = 0 & \mu - \text{complementary slackness} \end{cases}$$

- So we have the following system of equations to solve:

$$\begin{cases} \mathbf{Ax} - \mathbf{b} = 0 & \text{Primal feasibility} \\ \mathbf{A}^T \boldsymbol{\pi}^T - \mathbf{z} - \mathbf{c}^T = 0 & \text{Dual feasibility} \\ \mathbf{XZ}\mathbf{e} - \boldsymbol{\mu}\mathbf{e} = 0 & \boldsymbol{\mu} - \text{complementary slackness} \end{cases}$$

So what happens if we solve this linear system?

# Interior Point Methods

- So we have the following system of equations to solve:

$$\begin{cases} \mathbf{Ax} - \mathbf{b} = 0 & \text{Primal feasibility} \\ \mathbf{A}^T \boldsymbol{\pi}^T - \mathbf{z} - \mathbf{c}^T = 0 & \text{Dual feasibility} \\ \mathbf{XZ}\mathbf{e} - \mu\mathbf{e} = 0 & \mu - \text{complementary slackness} \end{cases}$$

Remember Lagrange multipliers find solutions where the objective function and constraints have parallel gradients!

- So we have the following system of equations to solve:

$$\begin{cases} \mathbf{Ax} - \mathbf{b} = 0 & \text{Primal feasibility} \\ \mathbf{A}^T \boldsymbol{\pi}^T - \mathbf{z} - \mathbf{c}^T = 0 & \text{Dual feasibility} \\ \mathbf{XZ}\mathbf{e} - \mu\mathbf{e} = 0 & \mu - \text{complementary slackness} \end{cases}$$

So we would find some solution laying  
on the boundary?

Probably somewhere close, because  
we used barrier terms!

The closeness is defined by  $\mu$

- So we have the following system of equations to solve:

$$\begin{cases} \mathbf{Ax} - \mathbf{b} = 0 & \text{Primal feasibility} \\ \mathbf{A}^T \boldsymbol{\pi}^T - \mathbf{z} - \mathbf{c}^T = 0 & \text{Dual feasibility} \\ \mathbf{XZ}\mathbf{e} - \boldsymbol{\mu}\mathbf{e} = 0 & \boldsymbol{\mu} - \text{complementary slackness} \end{cases}$$

So the analytic solution provided by Lagrange multipliers is good enough?

Not necessarily, consider a convex function with a well defined minimum at the center of the feasible region.  
In this case we may find some point far from such minimum

- So we have the following system of equations to solve:

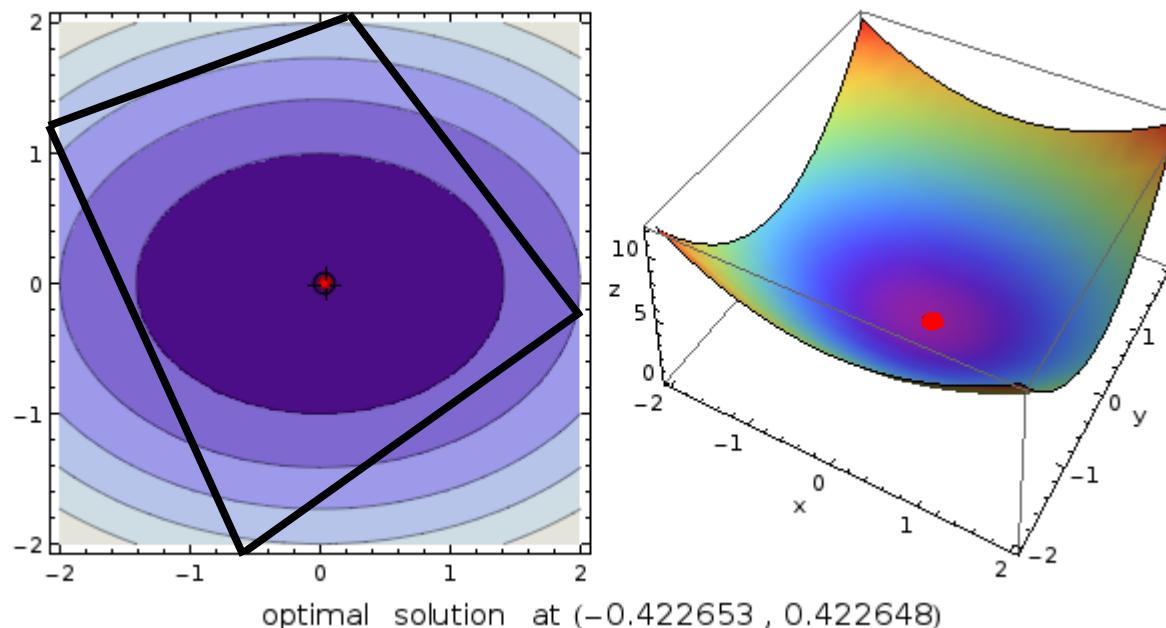
$$\begin{cases} \mathbf{Ax} - \mathbf{b} = 0 & \text{Primal feasibility} \\ \mathbf{A}^T \boldsymbol{\pi}^T - \mathbf{z} - \mathbf{c}^T = 0 & \text{Dual feasibility} \\ \mathbf{XZ}\mathbf{e} - \mu\mathbf{e} = 0 & \mu - \text{complementary slackness} \end{cases}$$

Thus, what can we do?

# Interior Point Methods

- This system summarizes the constraints and the objective function given the barrier terms:

$$\begin{cases} \mathbf{Ax} - \mathbf{b} = 0 & \text{Primal feasibility} \\ \mathbf{A}^T \boldsymbol{\pi}^T - \mathbf{z} - \mathbf{c}^T = 0 & \text{Dual feasibility} \\ \mathbf{XZ}\mathbf{e} - \mu\mathbf{e} = 0 & \mu - \text{complementary slackness} \end{cases}$$

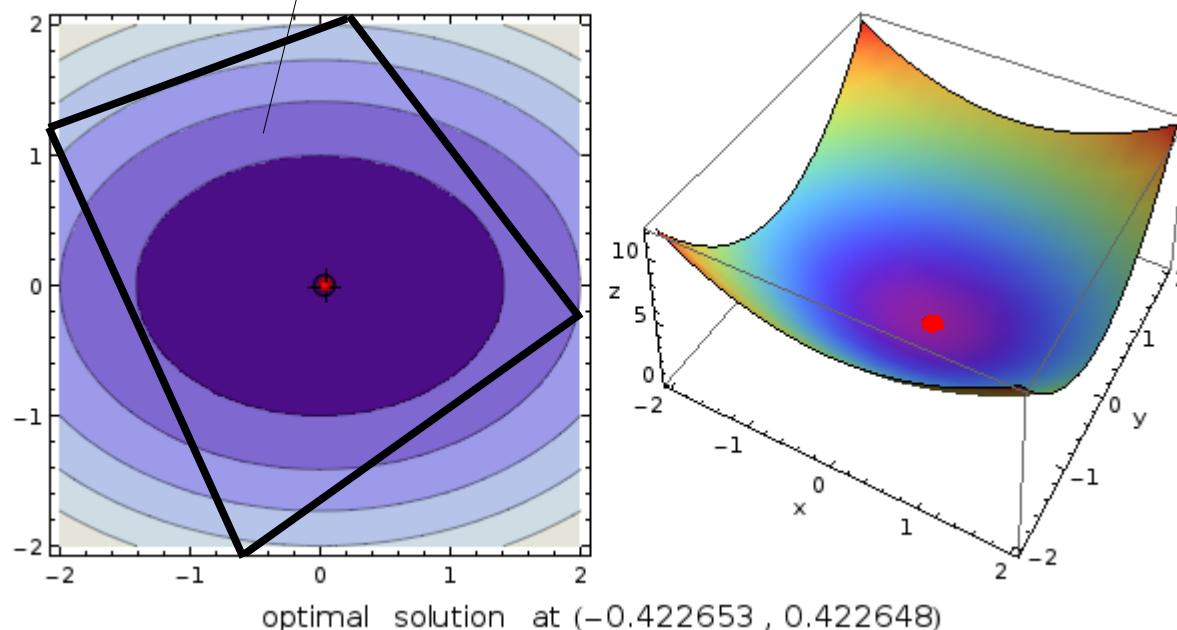


# Interior Point Methods

- This system summarizes the constraints and the objective function given the barrier terms:

$$\begin{cases} \mathbf{Ax} - \mathbf{b} = 0 \\ \mathbf{A}^T \boldsymbol{\pi}^T - \mathbf{z} - \mathbf{c}^T = 0 \\ \mathbf{XZ}\mathbf{e} - \mu\mathbf{e} = 0 \end{cases}$$

Consider this is the feasible region

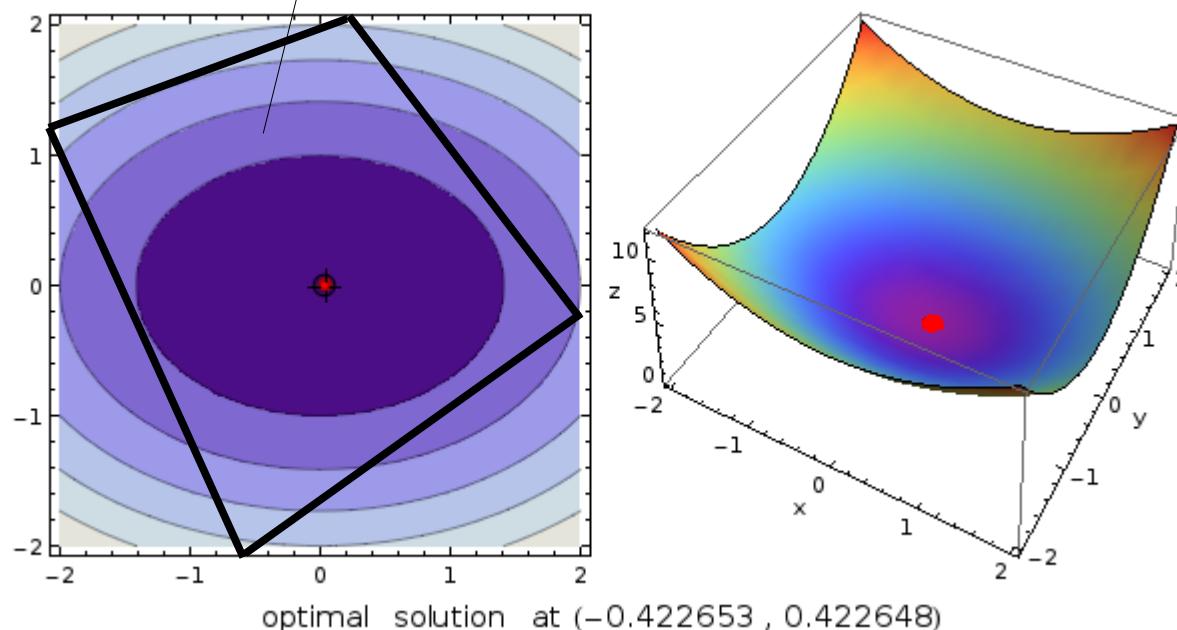


# Interior Point Methods

- This system summarizes the constraints and the objective function given the barrier terms:

$$\begin{cases} \mathbf{Ax} - \mathbf{b} = 0 \\ \mathbf{A}^T \boldsymbol{\pi}^T - \mathbf{z} - \mathbf{c}^T = 0 \\ \mathbf{XZ}\mathbf{e} - \mu\mathbf{e} = 0 \end{cases}$$

We will “walk” inside this region

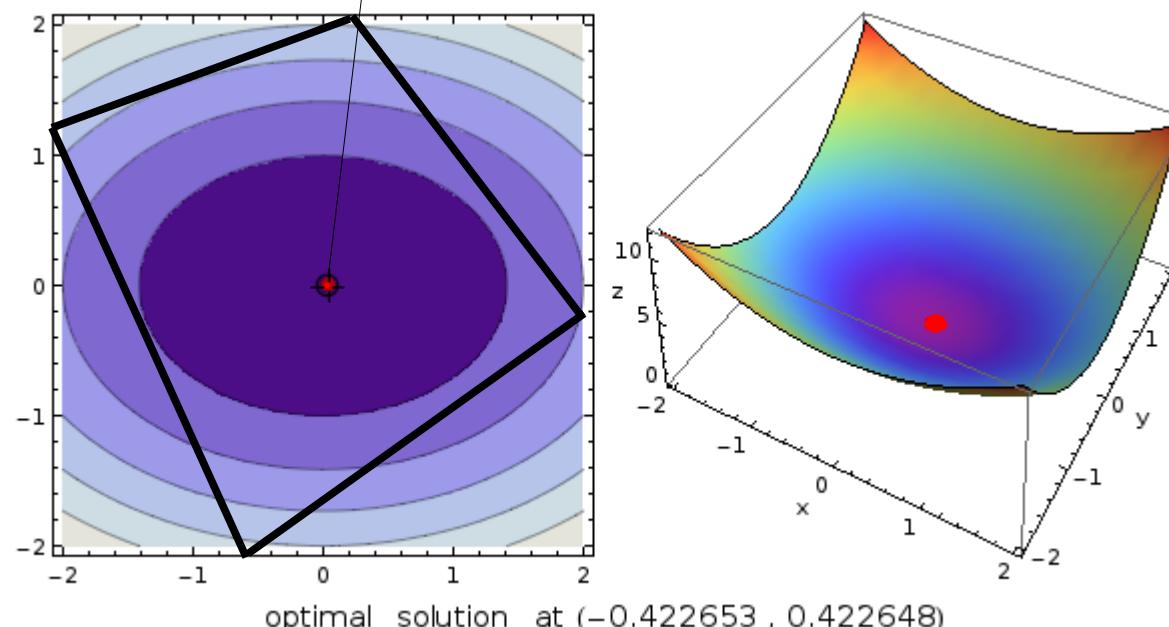


# Interior Point Methods

- This system summarizes the constraints and the objective function given the barrier terms:

$$\begin{cases} \mathbf{Ax} - \mathbf{b} = 0 \\ \mathbf{A}^T \boldsymbol{\pi}^T - \mathbf{z} - \mathbf{c}^T = 0 \\ \mathbf{XZ}\mathbf{e} - \boldsymbol{\mu}\mathbf{e} = 0 \end{cases}$$

In this case we intend to find this minimum for the primal problem



- This system summarizes the constraints and the objective function given the barrier terms:

$$\begin{cases} \mathbf{Ax} - \mathbf{b} = 0 & \text{Primal feasibility} \\ \mathbf{A}^T \boldsymbol{\pi}^T - \mathbf{z} - \mathbf{c}^T = 0 & \text{Dual feasibility} \\ \mathbf{XZ}\mathbf{e} - \boldsymbol{\mu}\mathbf{e} = 0 & \boldsymbol{\mu} - \text{complementary slackness} \end{cases}$$

As we solve this system of equations, we “walk” inside the feasible region and get closer to the solution

This system requires us to find where those three equations assume values equal to zero

Let's see how to find those zeros!

# **Using the Newton's method to find zeros**

# Newton's Method

- Remember Newton's Method (or Newton-Raphson Method) to find the zeros for a function:
  - It considers the following equation to approximate of the roots of a function  $f$  (or zeros)

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

- until a sufficiently accurate value is reached

# Newton's Method

- Newton's Method:

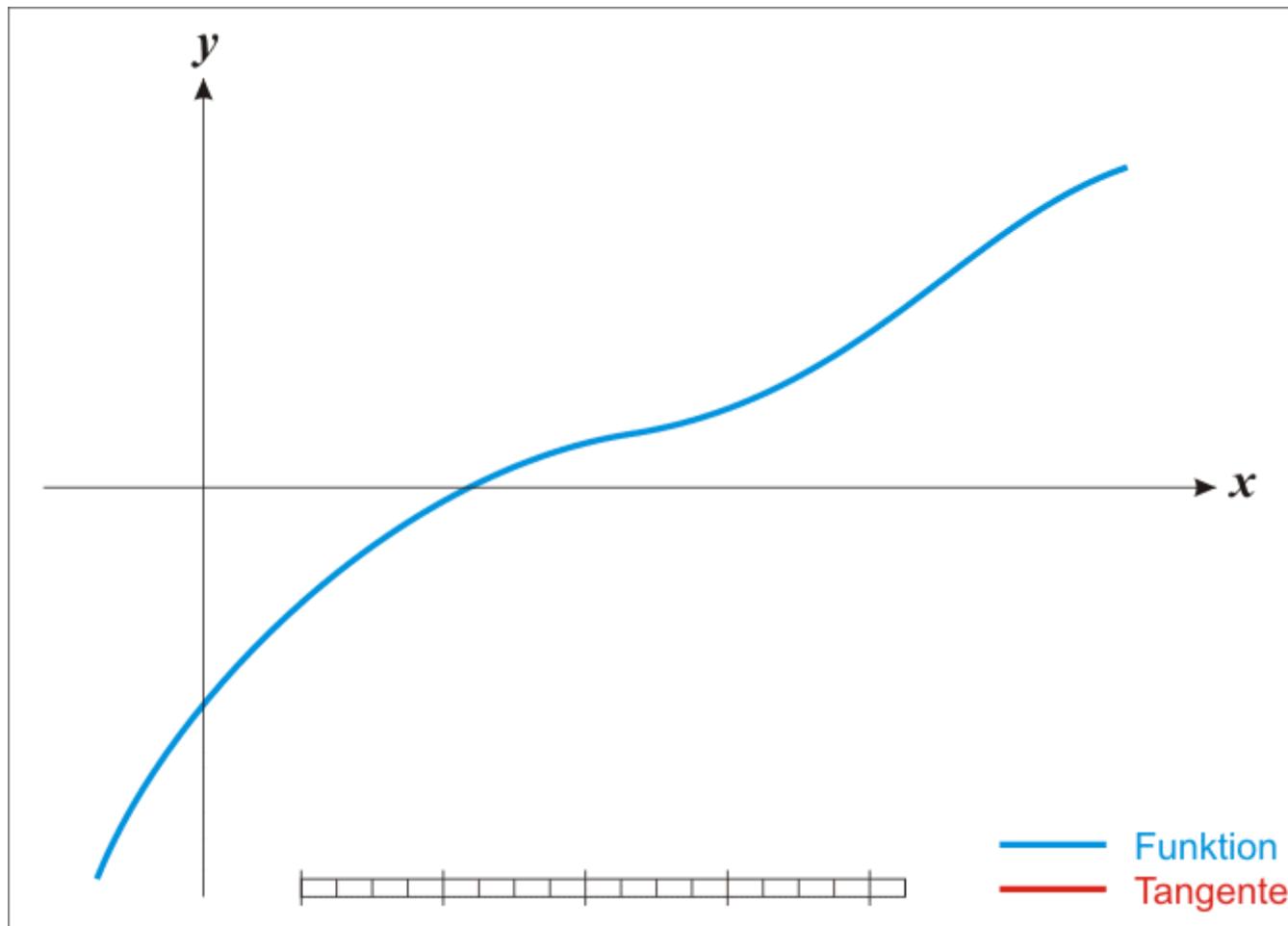


Figure available at: [http://en.wikipedia.org/wiki/Newton%27s\\_method](http://en.wikipedia.org/wiki/Newton%27s_method)

# Newton's Method

- From equation:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

- We have:

$$x_{n+1} - x_n = -\frac{f(x_n)}{f'(x_n)}$$

- Which we can write as follows:

$$\Delta_x = -\frac{f(x_n)}{f'(x_n)}$$

- From equation:

$$\Delta_x = -\frac{f(x_n)}{f'(x_n)}$$

- We finally obtain the form we will use later:

$$f'(x_n)\Delta_x = -f(x_n)$$

- From equation:

$$\Delta_x = -\frac{f(x_n)}{f'(x_n)}$$

- We finally obtain the form we will use later:

$$f'(x_n)\Delta_x = -f(x_n)$$

However this is only for one variable  $x$   
and we have a multivariable problem!  
So, how to approach it?

# Newton's Method

- To translate the single-variable Newton's Method:

$$f'(x_n)\Delta_x = -f(x_n)$$

- Into a multivariable approach we need the Jacobian matrix:

$$\mathbf{J}(\mathbf{x}^n) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_k} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_k} \\ \vdots & \vdots & & \vdots \\ \frac{\partial f_k}{\partial x_1} & \frac{\partial f_k}{\partial x_2} & \cdots & \frac{\partial f_k}{\partial x_k} \end{pmatrix}$$

- Considering k as the number of variables and n as the current iteration

# Newton's Method

- Thus we solve the multivariable Newton's Method as follows:

$$\mathbf{J}(\mathbf{x}^n)\Delta_{\mathbf{x}} = -\mathbf{f}(\mathbf{x}^n)$$

- This is what we will use to solve our system of equations!

**Using the Newton's Method to solve the system of equations**

- This is the system we need to solve:

$$\begin{cases} \mathbf{Ax} - \mathbf{b} = 0 & \text{Primal feasibility} \\ \mathbf{A}^T \boldsymbol{\pi}^T - \mathbf{z} - \mathbf{c}^T = 0 & \text{Dual feasibility} \\ \mathbf{XZ}\mathbf{e} - \boldsymbol{\mu}\mathbf{e} = 0 & \boldsymbol{\mu} - \text{complementary slackness} \end{cases}$$

- First we build the Jacobian matrix by deriving as follows:

$$\mathbf{J}(\mathbf{x}, \boldsymbol{\pi}, \mathbf{z}) = \begin{pmatrix} \frac{\partial f_1}{\partial \mathbf{x}} & \frac{\partial f_1}{\partial \boldsymbol{\pi}} & \frac{\partial f_1}{\partial \mathbf{z}} \\ \frac{\partial f_2}{\partial \mathbf{x}} & \frac{\partial f_2}{\partial \boldsymbol{\pi}} & \frac{\partial f_2}{\partial \mathbf{z}} \\ \frac{\partial f_3}{\partial \mathbf{x}} & \frac{\partial f_3}{\partial \boldsymbol{\pi}} & \frac{\partial f_3}{\partial \mathbf{z}} \end{pmatrix}$$

We differentiate in terms of the variables we can change value. This is very important!

In this manner, we can change slack variables in order to get closer to the solution

We can also change primal and dual variables to find the best solution considering both paths, i.e., the primal and the dual Problems

This modification in variables is what justifies an interactive approach instead of the analytical one.

need to solve:

Primal feasibility

Dual feasibility

$\mu$  – complementary slackness

in matrix by deriving as follows:

$$= \begin{pmatrix} \frac{\partial f_1}{\partial \mathbf{x}} & \frac{\partial f_1}{\partial \pi} & \frac{\partial f_1}{\partial \mathbf{z}} \\ \frac{\partial f_2}{\partial \mathbf{x}} & \frac{\partial f_2}{\partial \pi} & \frac{\partial f_2}{\partial \mathbf{z}} \\ \frac{\partial f_3}{\partial \mathbf{x}} & \frac{\partial f_3}{\partial \pi} & \frac{\partial f_3}{\partial \mathbf{z}} \end{pmatrix}$$

- This is the system we need to solve:

$$\begin{cases} \mathbf{Ax} - \mathbf{b} = 0 & \text{Primal feasibility} \\ \mathbf{A}^T \boldsymbol{\pi}^T - \mathbf{z} - \mathbf{c}^T = 0 & \text{Dual feasibility} \\ \mathbf{XZ}\mathbf{e} - \boldsymbol{\mu}\mathbf{e} = 0 & \boldsymbol{\mu} - \text{complementary slackness} \end{cases}$$

- First we build the Jacobian matrix by deriving as follows:

$$\mathbf{J}(\mathbf{x}, \boldsymbol{\pi}, \mathbf{z}) = \begin{pmatrix} \frac{\partial f_1}{\partial \mathbf{x}} & \frac{\partial f_1}{\partial \boldsymbol{\pi}} & \frac{\partial f_1}{\partial \mathbf{z}} \\ \frac{\partial f_2}{\partial \mathbf{x}} & \frac{\partial f_2}{\partial \boldsymbol{\pi}} & \frac{\partial f_2}{\partial \mathbf{z}} \\ \frac{\partial f_3}{\partial \mathbf{x}} & \frac{\partial f_3}{\partial \boldsymbol{\pi}} & \frac{\partial f_3}{\partial \mathbf{z}} \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}^T & -\mathbf{I} \\ \mathbf{Z} & \mathbf{0} & \mathbf{X} \end{pmatrix}$$

We do not need a solver in here,  
because we can solve it  
algebraically, right?

solve:

Primal feasibility

Dual feasibility

– complementary slackness

- First we build the Jacobian matrix by deriving as follows:

$$\mathbf{J}(\mathbf{x}, \boldsymbol{\pi}, \mathbf{z}) = \begin{pmatrix} \frac{\partial f_1}{\partial \mathbf{x}} & \frac{\partial f_1}{\partial \boldsymbol{\pi}} & \frac{\partial f_1}{\partial \mathbf{z}} \\ \frac{\partial f_2}{\partial \mathbf{x}} & \frac{\partial f_2}{\partial \boldsymbol{\pi}} & \frac{\partial f_2}{\partial \mathbf{z}} \\ \frac{\partial f_3}{\partial \mathbf{x}} & \frac{\partial f_3}{\partial \boldsymbol{\pi}} & \frac{\partial f_3}{\partial \mathbf{z}} \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}^T & -\mathbf{I} \\ \mathbf{Z} & \mathbf{0} & \mathbf{X} \end{pmatrix}$$

- This is the system we need to solve:

$$\begin{cases} \mathbf{Ax} - \mathbf{b} = 0 & \text{Primal feasibility} \\ \mathbf{A}^T \boldsymbol{\pi}^T - \mathbf{z} - \mathbf{c}^T = 0 & \text{Dual feasibility} \\ \mathbf{XZ}\mathbf{e} - \mu\mathbf{e} = 0 & \mu - \text{complementary slackness} \end{cases}$$

- Now suppose we have a starting point  $(\mathbf{x}^0, \boldsymbol{\pi}^0, \mathbf{z}^0)$ , thus we have the following as result for function  $\mathbf{f}$ :

$$\mathbf{f}(\mathbf{x}^0) = \mathbf{Ax}^0 - \mathbf{b}$$

$$\mathbf{f}(\boldsymbol{\pi}^0) = -\mathbf{c}^T + \mathbf{A}^T (\boldsymbol{\pi}^0)^T - \mathbf{z}^0$$

$$\mathbf{f}(\mathbf{z}^0) = \mathbf{X}^0 \mathbf{Z}^0 \mathbf{e} - \mu \mathbf{e}$$

- This is the system we need to solve:

$$\begin{cases} \mathbf{Ax} - \mathbf{b} = 0 & \text{Primal feasibility} \\ \mathbf{A}^T \boldsymbol{\pi}^T - \mathbf{z} - \mathbf{c}^T = 0 & \text{Dual feasibility} \\ \mathbf{XZ}\mathbf{e} - \boldsymbol{\mu}\mathbf{e} = 0 & \boldsymbol{\mu} - \text{complementary slackness} \end{cases}$$

- Now we build Newton's method:

$$\mathbf{J}(\mathbf{x}^n) \Delta_{\mathbf{x}} = -\mathbf{f}(\mathbf{x}^n)$$

- As follows:

$$\begin{pmatrix} \mathbf{A} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}^T & -\mathbf{I} \\ \mathbf{Z} & \mathbf{0} & \mathbf{X} \end{pmatrix} \begin{pmatrix} \mathbf{d}_{\mathbf{x}} \\ \mathbf{d}_{\boldsymbol{\pi}} \\ \mathbf{d}_{\mathbf{z}} \end{pmatrix} = - \begin{pmatrix} \mathbf{f}(\mathbf{x}^0) \\ \mathbf{f}(\boldsymbol{\pi}^0) \\ \mathbf{f}(\mathbf{z}^0) \end{pmatrix}$$

- Thus we will solve the Newton's method for:

$$\begin{pmatrix} A & 0 & 0 \\ 0 & A^T & -I \\ Z & 0 & X \end{pmatrix} \begin{pmatrix} d_x \\ d_\pi \\ d_z \end{pmatrix} = - \begin{pmatrix} f(x^0) \\ f(\pi^0) \\ f(z^0) \end{pmatrix}$$

- Or in a detailed form:

$$\begin{pmatrix} A & 0 & 0 \\ 0 & A^T & -I \\ Z & 0 & X \end{pmatrix} \begin{pmatrix} d_x \\ d_\pi \\ d_z \end{pmatrix} = - \begin{pmatrix} Ax - b \\ -c^T + A^T(\pi)^T - z \\ XZe - \mu e \end{pmatrix}$$

- After some algebraic work we go from:

$$\begin{pmatrix} \mathbf{A} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}^T & -\mathbf{I} \\ \mathbf{Z} & \mathbf{0} & \mathbf{X} \end{pmatrix} \begin{pmatrix} \mathbf{d}_x \\ \mathbf{d}_\pi \\ \mathbf{d}_z \end{pmatrix} = - \begin{pmatrix} \mathbf{Ax} - \mathbf{b} \\ -\mathbf{c}^T + \mathbf{A}^T(\boldsymbol{\pi})^T - \mathbf{z} \\ \mathbf{XZe} - \mu \mathbf{e} \end{pmatrix}$$

- To:

$$\mathbf{d}_\pi = (\mathbf{AZ}^{-1}\mathbf{XA}^T)^{-1}(-\mathbf{b} + \mu \mathbf{AZ}^{-1}\mathbf{e} + \mathbf{AZ}^{-1}\mathbf{Xf}(\boldsymbol{\pi}))$$

$$\mathbf{d}_z = -\mathbf{f}(\boldsymbol{\pi}) + \mathbf{A}^T \mathbf{d}_\pi$$

$$\mathbf{d}_x = \mathbf{Z}^{-1}(\mu \mathbf{e} - \mathbf{XZe} - \mathbf{Xd}_z)$$

- This is what we will use to implement the IPM!
  - In this case it is easy to solve analytically because this Jacobian matrix allows it!

**Implementing the Interior Point Method to solve our  
first optimization problem**

# Implementing the Interior Point Method

- This is the optimization problem we started solving:

## Primal

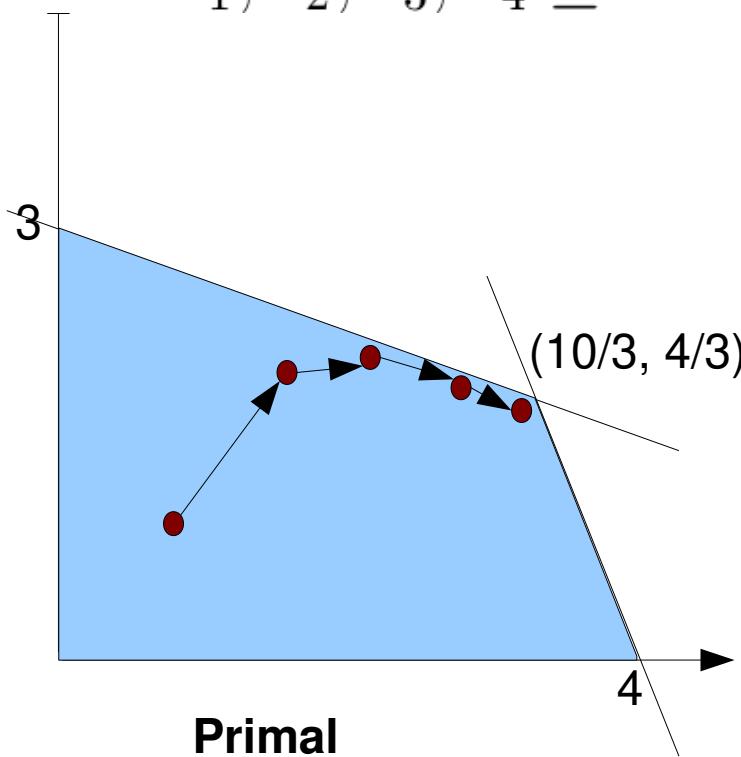
$$\text{Maximize } z_p = 2x_1 + 3x_2$$

Subject to:

$$2x_1 + x_2 + x_3 = 8$$

$$x_1 + 2x_2 + x_4 = 6$$

$$x_1, x_2, x_3, x_4 \geq 0$$



## Dual

$$\text{Minimize } z_d = 8\pi_1 + 6\pi_2$$

Subject to:

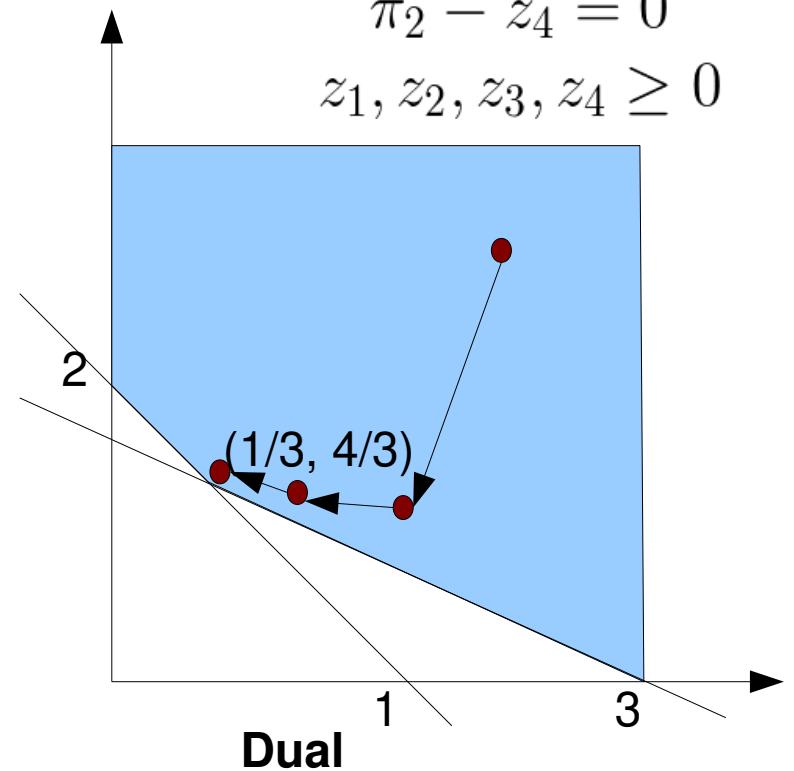
$$2\pi_1 + \pi_2 - z_1 = 2$$

$$\pi_1 + 2\pi_2 - z_2 = 3$$

$$\pi_1 - z_3 = 0$$

$$\pi_2 - z_4 = 0$$

$$z_1, z_2, z_3, z_4 \geq 0$$



# Implementing the Interior Point Method

- Algorithm:
  - Step 1
    - Start with some point inside the feasible region:  
 $\mathbf{x}^0 > \mathbf{0}, \mathbf{z}^0 > \mathbf{0}, \boldsymbol{\pi}^0$
    - Set the iteration counter  $k = 0$
  - Step 2
    - If  $\text{Gap} < \text{threshold}$  then stop, otherwise go to Step 3
  - Step 3
    - Compute Newton's method
  - Step 4
    - Update the current solution by adapting  $\mathbf{x}, \boldsymbol{\pi}$  and  $\mathbf{z}$
    - Increment  $k$  and go to Step 2

# Implementing the Interior Point Method

- Let's implement it! See source code ***02-ipm.r***

## Primal

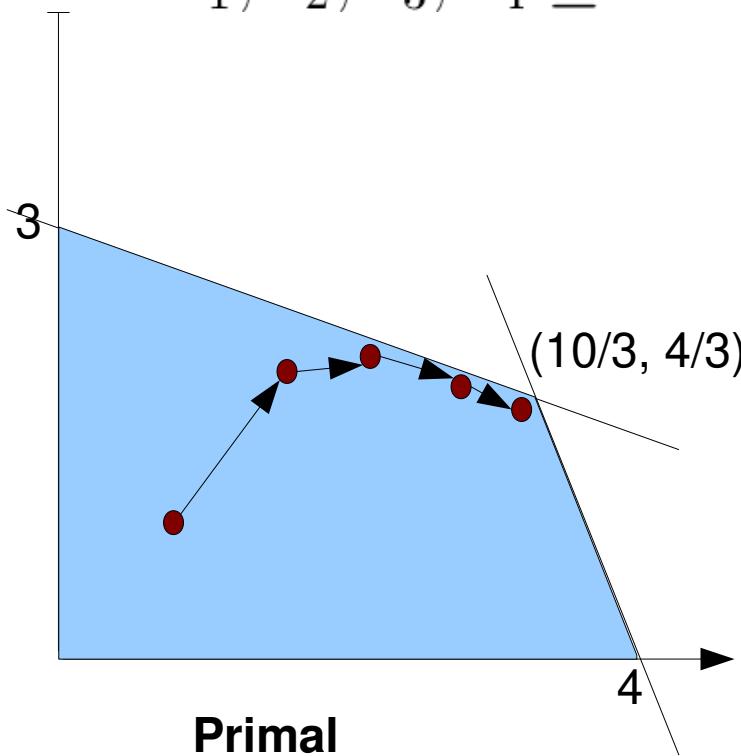
$$\text{Maximize } z_p = 2x_1 + 3x_2$$

Subject to:

$$2x_1 + x_2 + x_3 = 8$$

$$x_1 + 2x_2 + x_4 = 6$$

$$x_1, x_2, x_3, x_4 \geq 0$$



## Dual

$$\text{Minimize } z_d = 8\pi_1 + 6\pi_2$$

Subject to:

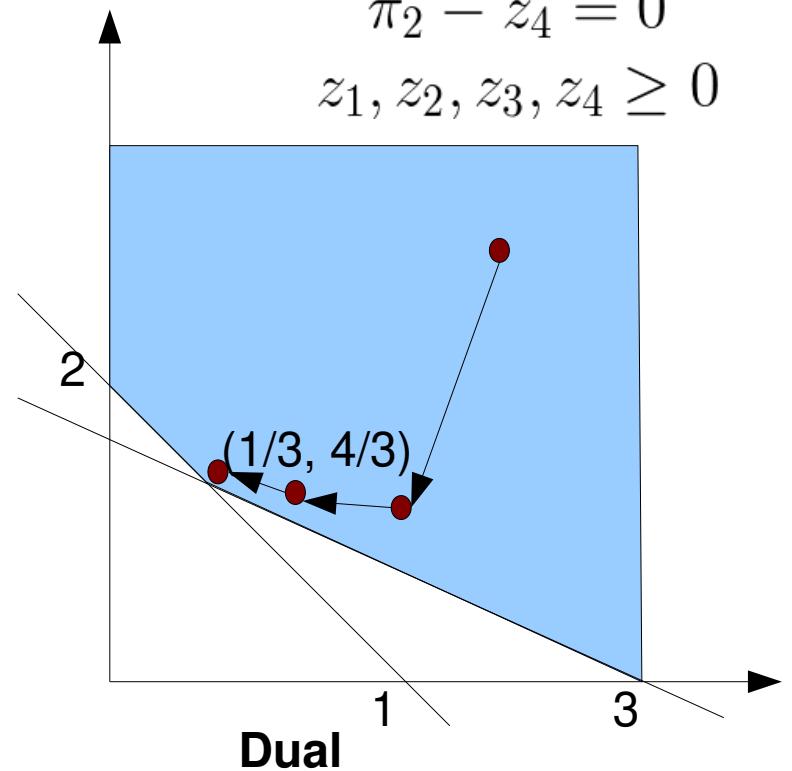
$$2\pi_1 + \pi_2 - z_1 = 2$$

$$\pi_1 + 2\pi_2 - z_2 = 3$$

$$\pi_1 - z_3 = 0$$

$$\pi_2 - z_4 = 0$$

$$z_1, z_2, z_3, z_4 \geq 0$$



# **Implementing the Interior Point Method to approach SVM**

Based on the paper:

Fine S.; Scheinberg, K. (2001) Efficient SVM Training Using Low-Rank Kernel Representations, Journal of Machine Learning Research, 2 (2001), 243-264

# IPM to approach SVM

- First of all we have the optimization problem in its primal and dual forms as follows:

## PRIMAL

$$\underset{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}, \xi \in \mathbb{R}^m}{\text{Minimize}} \quad \tau(\mathbf{w}, \xi) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \xi_i$$

$$\begin{aligned} & \text{subject to: } y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i \text{ for all } i = 1, \dots, m \\ & \quad \xi_i \geq 0 \text{ for all } i = 1, \dots, m \end{aligned}$$

## DUAL

$$\underset{\alpha \in \mathbb{R}^m}{\text{Maximize}} \quad W(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle$$

$$\begin{aligned} & \text{subject to } 0 \leq \alpha_i \leq C, \text{ for all } i = 1, \dots, m \text{ and} \\ & \quad \sum_{i=1}^m \alpha_i y_i = 0 \end{aligned}$$

# IPM to approach SVM

- First of all we have the optimization problem in its primal and dual forms as follows:

## PRIMAL

Minimize  
 $\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}$

subject to

As first step we will formulate both  
problem using the matrix form

Maximize  
 $\alpha \in \mathbb{R}^m$

subject to  $0 \leq \alpha_i \leq C, i = 1, 2, \dots, m$  and

$$\sum_{i=1}^m \alpha_i y_i = 0$$

# **Formulating the primal problem in matrix form**

# IPM to approach SVM

- Formulating it in a matrix-form problem

## PRIMAL

$$\underset{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}, \xi \in \mathbb{R}^m}{\text{Minimize}} \quad \tau(\mathbf{w}, \xi) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \xi_i$$

subject to:  $y_i(<\mathbf{w}, \mathbf{x}_i> + b) \geq 1 - \xi_i$  for all  $i = 1, \dots, m$

$\xi_i \geq 0$  for all  $i = 1, \dots, m$

- When we built the Lagrangian for the primal problem we found:

$$\frac{\partial \Lambda_{\text{Primal}}}{\partial \mathbf{w}} = \mathbf{w} - \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i = 0$$

$$\mathbf{w} = \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i$$

# IPM to approach SVM

- Formulating it in a matrix-form problem

**PRIMAL**

$$\begin{aligned} & \text{Minimize}_{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}, \xi \in \mathbb{R}^m} \quad \tau(\mathbf{w}, b) \\ & \text{subject to} \quad \mathbf{w}^\top \mathbf{x}_i + b - \xi_i \geq 1, \quad i = 1, 2, \dots, m \end{aligned}$$

- When we multiply the second equation by  $y_i$  and sum over all  $i$ , we get the bound:

So, we can proceed with some simplification and substitution

$$\mathbf{w} = \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i$$

# IPM to approach SVM

- So we can simplify the problem

## PRIMAL

$$\underset{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}, \xi \in \mathbb{R}^m}{\text{Minimize}} \quad \tau(\mathbf{w}, \xi) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \xi_i$$

subject to:  $y_i(\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1 - \xi_i$  for all  $i = 1, \dots, m$

$\xi_i \geq 0$  for all  $i = 1, \dots, m$

- As follows:

## PRIMAL

$$\underset{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}, \xi \in \mathbb{R}^m}{\text{Minimize}} \quad \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \xi_i$$

subject to  $y_i \langle \mathbf{w}, \mathbf{x}_i \rangle + y_i b + \xi_i \geq e$

$\xi_i \geq 0$  for all  $i = 1, \dots, m$

# IPM to approach SVM

- So we can rewrite the problem

## PRIMAL

$$\begin{aligned} & \underset{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}, \xi \in \mathbb{R}^m}{\text{Minimize}} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^m \xi_i \\ & \text{subject to } y_i \langle \mathbf{w}, \mathbf{x}_i \rangle + y_i b + \xi_i \geq e \\ & \quad \xi_i \geq 0 \text{ for all } i = 1, \dots, m \end{aligned}$$

- By substituting  $\mathbf{w}$  as follows:

## PRIMAL

$$\begin{aligned} & \underset{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}, \xi \in \mathbb{R}^m}{\text{Minimize}} \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + C \sum_{i=1}^m \xi_i \\ & \text{subject to } \sum_{j=1}^m \alpha_j y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle + y_i b + \xi_i \geq e \\ & \quad \xi_i \geq 0 \text{ for all } i = 1, \dots, m \end{aligned}$$

# IPM to approach SVM

- Now we make another substitution...

## PRIMAL

$$\underset{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}, \xi \in \mathbb{R}^m}{\text{Minimize}} \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j [y_i y_j < \mathbf{x}_i, \mathbf{x}_j >] + C \sum_{i=1}^m \xi_i$$

$$\text{subject to } \sum_{j=1}^m \alpha_j [y_i y_j < \mathbf{x}_i, \mathbf{x}_j >] + y_i b + \xi_i \geq e \\ \xi_i \geq 0 \text{ for all } i = 1, \dots, m$$

Call this term as Q (matrix Q)

# IPM to approach SVM

- Now we make another substitution:

## PRIMAL

$$\begin{aligned} & \text{Minimize}_{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}, \xi \in \mathbb{R}^m} \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j [y_i y_j < \mathbf{x}_i, \mathbf{x}_j >] + C \sum_{i=1}^m \xi_i \\ & \text{subject to } \sum_{j=1}^m \alpha_j [y_i y_j < \mathbf{x}_i, \mathbf{x}_j >] + y_i b + \xi_i \geq e \\ & \quad \xi_i \geq 0 \text{ for all } i = 1, \dots, m \end{aligned}$$

- To obtain:

## PRIMAL

$$\begin{aligned} & \text{Minimize}_{\mathbf{w} \in \mathbb{H}, b \in \mathbb{R}, \xi \in \mathbb{R}^m} \frac{1}{2} \alpha^T \mathbf{Q} \alpha + C \sum_{i=1}^m \xi_i \\ & \text{subject to } \mathbf{Q} \alpha + \mathbf{y} b + \xi \geq \mathbf{e} \\ & \quad \xi_i \geq 0 \text{ for all } i = 1, \dots, m \end{aligned}$$

# IPM to approach SVM

- Now we make another substitution:

**PRIMAL**

$$\begin{aligned} \text{Minimize}_{\mathbf{w} \in \mathcal{H}, b \in \mathbb{R}, \xi \in \mathbb{R}^m} \quad & \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \mathbf{w}_i \mathbf{w}_j \mathbf{x}_i^T \mathbf{x}_j + C \sum_{i=1}^m \xi_i \\ \text{subject to} \quad & \mathbf{w}^T \mathbf{x}_i + b + \xi_i \geq 1 \quad \forall i \end{aligned}$$

However, after such substitutions  
We do not have  $\mathbf{w}$  anymore to minimize!

- To obtain

$$\begin{aligned} \text{Minimize}_{\mathbf{w} \in \mathbb{H}, b \in \mathbb{R}, \xi \in \mathbb{R}^m} \quad & \frac{1}{2} \alpha^T \mathbf{Q} \alpha + C \sum_{i=1}^m \xi_i \\ \text{subject to} \quad & \mathbf{Q} \alpha + \mathbf{y} b + \xi \geq \mathbf{e} \\ & \xi_i \geq 0 \text{ for all } i = 1, \dots, m \end{aligned}$$

# IPM to approach SVM

- So, from:

## PRIMAL

$$\begin{aligned} & \text{Minimize}_{\mathbf{w} \in \mathbb{H}, b \in \mathbb{R}, \xi \in \mathbb{R}^m} \quad \frac{1}{2} \alpha^T \mathbf{Q} \alpha + C \sum_{i=1}^m \xi_i \\ & \text{subject to} \quad \mathbf{Q} \alpha + \mathbf{y} b + \xi \geq \mathbf{e} \\ & \quad \xi_i \geq 0 \text{ for all } i = 1, \dots, m \end{aligned}$$

- Fine and Scheinberg obtain: Adding slack variables to simplify our problem to equality constraints

## PRIMAL

$$\begin{aligned} & \text{Minimize}_{\mathbf{s} \in \mathbb{R}^m, b \in \mathbb{R}, \xi \in \mathbb{R}^m} \quad \frac{1}{2} \alpha^T \mathbf{Q} \alpha + C \sum_{i=1}^m \xi_i \\ & \text{subject to} \quad \mathbf{Q} \alpha + \mathbf{y} b + \xi - \boxed{\mathbf{s}} = \mathbf{e} \\ & \quad \xi \geq 0, \mathbf{s} \geq \mathbf{0} \quad \text{for all } i = 1, \dots, m \end{aligned}$$

# **Formulating the dual problem in matrix form**

# IPM to approach SVM

- Formulating it in a matrix-form problem:

## DUAL

$$\underset{\alpha \in \mathbb{R}^m}{\text{Maximize}} \quad W(\alpha) = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle$$

subject to  $0 \leq \alpha_i \leq C$ , for all  $i = 1, \dots, m$  and

$$\sum_{i=1}^m \alpha_i y_i = 0$$

- By applying matrix  $\mathbf{Q}$  we have:

## DUAL

$$\underset{\alpha \in \mathbb{R}^m}{\text{Maximize}} \quad \mathbf{e}^T \alpha - \frac{1}{2} \alpha^T \mathbf{Q} \alpha$$

subject to  $\mathbf{0} \leq \alpha \leq \mathbf{C}$

$$\mathbf{y}^T \alpha = 0$$

**Now we have both formulations in  
matrix form**

# IPM to approach SVM

- These matrix-form problems are easier to work with:

## PRIMAL

$$\begin{aligned} & \underset{\mathbf{s} \in \mathbb{R}^m, b \in \mathbb{R}, \xi \in \mathbb{R}^m}{\text{Minimize}} \quad \frac{1}{2} \alpha^T \mathbf{Q} \alpha + C \sum_{i=1}^m \xi_i \\ & \text{subject to} \quad \mathbf{Q} \alpha + \mathbf{y} b + \xi - \mathbf{s} = \mathbf{e} \\ & \quad \xi \geq 0, \mathbf{s} \geq \mathbf{0} \end{aligned}$$

## DUAL

$$\begin{aligned} & \underset{\alpha \in \mathbb{R}^m}{\text{Maximize}} \quad \mathbf{e}^T \alpha - \frac{1}{2} \alpha^T \mathbf{Q} \alpha \\ & \text{subject to} \quad \mathbf{0} \leq \alpha \leq \mathbf{C} \\ & \quad \mathbf{y}^T \alpha = 0 \end{aligned}$$

# IPM to approach SVM

- These matrix-form problems are easier to work with:

## PRIMAL

Now, to implement the primal-dual path following interior point method, we first need to define the system of equations to be solved.

That is obtained using the Lagrangian

# IPM to approach SVM

- The Lagrangian for the primal form is built as follows:

**PRIMAL**

$$\begin{aligned} & \text{Minimize}_{\mathbf{s} \in \mathbb{R}^m, b \in \mathbb{R}, \xi \in \mathbb{R}^m} \quad \frac{1}{2} \alpha^T \mathbf{Q} \alpha + C \sum_{i=1}^m \xi_i \\ & \text{subject to} \quad \mathbf{Q} \alpha + \mathbf{y} b + \xi - \mathbf{s} = \mathbf{e} \\ & \quad \quad \quad \boxed{\xi \geq 0, \mathbf{s} \geq \mathbf{0}} \end{aligned}$$

$$\Lambda_{\text{Primal}}(\mathbf{s}, b, \xi, \lambda) = \frac{1}{2} \alpha^T \mathbf{Q} \alpha + C \sum_{i=1}^m \xi_i - \mu \left[ \sum_{i=1}^m \log \xi_i \right] - \mu \left[ \sum_{i=1}^m \log s_i \right] - \lambda (\mathbf{Q} \alpha + \mathbf{y} b + \xi - \mathbf{s} - \mathbf{e})$$

**Two barrier terms. We can have the same mu or different ones. In order to simplify formulation we will use the same**

# IPM to approach SVM

- Now we derive the Lagrangian to obtain the system of equations to be solved using Newton's Method:

$$\begin{aligned}\Lambda_{\text{Primal}}(\mathbf{s}, b, \xi, \lambda) = & \frac{1}{2} \boldsymbol{\alpha}^T \mathbf{Q} \boldsymbol{\alpha} + C \sum_{i=1}^m \xi_i - \mu \sum_{i=1}^m \log \xi_i - \mu \sum_{i=1}^m \log s_i \\ & - \lambda (\mathbf{Q} \boldsymbol{\alpha} + \mathbf{y}b + \boldsymbol{\xi} - \mathbf{s} - \mathbf{e})\end{aligned}$$

- The stationary point is found at:

$$\begin{aligned}\frac{\partial \Lambda}{\partial \mathbf{S}} &= -\mu \mathbf{S}^{-1} + \lambda = 0 \Rightarrow \mu = \mathbf{S} \lambda \\ \frac{\partial \Lambda}{\partial b} &= -\mathbf{y}^T \lambda = 0 \\ \frac{\partial \Lambda}{\partial \xi} &= C - \mu \xi^{-1} - \lambda = 0 \\ \frac{\partial \Lambda}{\partial \lambda} &= \mathbf{Q} \boldsymbol{\alpha} + \mathbf{y}b + \boldsymbol{\xi} - \mathbf{s} - \mathbf{e} = 0\end{aligned}$$

# IPM to approach SVM

- The Lagrangian for the dual form is built as follows:

**DUAL**

$$\begin{aligned} & \underset{\alpha \in \mathbb{R}^m}{\text{Maximize}} \quad \mathbf{e}^T \alpha - \frac{1}{2} \alpha^T \mathbf{Q} \alpha \\ & \text{subject to} \quad \boxed{\mathbf{0} \leq \alpha \leq \mathbf{C}} \\ & \quad \quad \quad \mathbf{y}^T \alpha = 0 \end{aligned}$$

$$\Lambda_{\text{Dual}}(\alpha, \beta) = \mathbf{e}^T \alpha - \frac{1}{2} \alpha^T \mathbf{Q} \alpha + \mu \left[ \sum_{i=1}^m \log \alpha_i \right] + \mu \left[ \sum_{i=1}^m \log(-\alpha_i + C) \right] - \beta (\mathbf{y}^T \alpha)$$

**Observe: We will have two barrier terms!**

# IPM to approach SVM

- Now we derive the Lagrangian:

$$\Lambda_{\text{Dual}}(\alpha, \beta) = \mathbf{e}^T \alpha - \frac{1}{2} \alpha^T \mathbf{Q} \alpha + \mu \sum_{i=1}^m \log \alpha_i + \mu \sum_{i=1}^m \log(-\alpha_i + C) - \beta (\mathbf{y}^T \alpha)$$

- Then, the stationary point is found at:

$$\frac{\partial \Lambda_{\text{Dual}}}{\partial \alpha} = \mathbf{e} - \mathbf{Q} \alpha + \mu \alpha^{-1} - \mu(C - \alpha)^{-1} - \beta \mathbf{y} = 0$$

$$\frac{\partial \Lambda_{\text{Dual}}}{\partial \beta} = -\mathbf{y}^T \alpha = 0$$

**We will now observe the stationary point found using  
the primal and the dual forms**

# IPM to approach SVM

- The stationary point using the primal form comes from the system:

$$\frac{\partial \Lambda}{\partial \mathbf{S}} = -\mu \mathbf{S}^{-1} + \lambda = 0 \Rightarrow \mu = \mathbf{S}\lambda$$

$$\frac{\partial \Lambda}{\partial b} = -\mathbf{y}^T \lambda = 0$$

$$\frac{\partial \Lambda}{\partial \xi} = C - \mu \xi^{-1} - \lambda = 0$$

$$\frac{\partial \Lambda}{\partial \lambda} = \mathbf{Q}\alpha + \mathbf{y}b + \xi - \mathbf{s} - \mathbf{e} = 0$$

- The stationary point using the Dual form comes from the system:

$$\frac{\Lambda_{\text{Dual}}}{\partial \alpha} = \mathbf{e} - \mathbf{Q}\alpha + \mu\alpha^{-1} - \mu(C - \alpha)^{-1} - \beta\mathbf{y} = 0$$

$$\frac{\Lambda_{\text{Dual}}}{\partial \beta} = -\mathbf{y}^T \alpha = 0$$

# IPM to approach SVM

- The stationary point using the primal form comes from the system:

$$\frac{\partial \Lambda}{\partial \mathbf{S}} = -\mu \mathbf{S}^{-1} + \lambda = 0 \Rightarrow \mu = \mathbf{S}\lambda$$

The idea behind the primal-dual path following approach is to consider both systems

- The stationary point using the dual form comes from the system:

$$\begin{aligned}\frac{\partial \Lambda_{\text{Dual}}}{\partial \alpha} &= \mathbf{e} - \mathbf{Q}\alpha + \mu\alpha - \beta\mathbf{y} = 0 \\ \frac{\partial \Lambda_{\text{Dual}}}{\partial \beta} &= -\mathbf{y}^T\alpha = 0\end{aligned}$$

# IPM to approach SVM

- The stationary point using the primal form comes from the system:

$$\frac{\partial \Lambda}{\partial \mathbf{S}} = -\mu \mathbf{S}^{-1} + \lambda = 0 \Rightarrow \mu = \mathbf{S}\lambda$$

- The sta

To avoid redundancy, we select the minimal set of equations to compose our final system

$$\begin{aligned}\frac{\partial \Lambda_{\text{Dual}}}{\partial \alpha} &= \mathbf{y}^T \alpha - b = 0 \\ \frac{\partial \Lambda_{\text{Dual}}}{\partial \beta} &= -\mathbf{y}^T \alpha = 0\end{aligned}$$

# IPM to approach SVM

- Stationary point found using the primal form:

$$\begin{aligned}\frac{\partial \Lambda}{\partial \mathbf{S}} &= -\mu \mathbf{S}^{-1} + \lambda = 0 \Rightarrow \mu = \boxed{\mathbf{S}\lambda} \\ \frac{\partial \Lambda}{\partial b} &= -\mathbf{y}^T \boxed{\lambda} = 0 \\ \frac{\partial \Lambda}{\partial \xi} &= C - \mu \xi^{-1} - \lambda = 0 \\ \frac{\partial \Lambda}{\partial \lambda} &= \mathbf{Q}\alpha + \mathbf{y}b + \xi - \mathbf{s} - \mathbf{e} = 0\end{aligned}$$

- Stationary point found using the Dual form:

$$\begin{aligned}\frac{\Lambda_{\text{Dual}}}{\partial \alpha} &= \mathbf{e} - \mathbf{Q}\alpha + \mu \alpha^{-1} - \mu(C - \alpha)^{-1} - \beta \mathbf{y} = 0 \\ \frac{\Lambda_{\text{Dual}}}{\partial \beta} &= -\mathbf{y}^T \boxed{\alpha} = 0\end{aligned}$$

Observe  $\lambda$  must be equal to  $\alpha$ , so we satisfy S and this feasibility constraint

# IPM to approach SVM

- The stationary point using the primal form comes from the system:

$$\frac{\partial \Lambda}{\partial \mathbf{S}} = -\mu \mathbf{S}^{-1} + \lambda = 0 \Rightarrow \mu = \mathbf{S}\lambda$$

$$\frac{\partial \Lambda}{\partial b} = -\mathbf{y}^T \lambda = 0$$

$$\frac{\partial \Lambda}{\partial \xi} = C - \mu \xi^{-1} - \lambda = 0$$

$$\frac{\partial \Lambda}{\partial \lambda} = \mathbf{Q}\alpha + \mathbf{y}b + \xi - \boxed{\mathbf{s}} - \mathbf{e} = 0$$

- The stationary point using the Dual form comes from the system:

Thus  $\mathbf{S}$  must be equal to

$$\frac{\Lambda_{\text{Dual}}}{\partial \alpha} = \mathbf{e} - \mathbf{Q}\alpha + \boxed{\mu \alpha^{-1}} - \mu(C - \alpha)^{-1} - \beta \mathbf{y} = 0$$

$$\frac{\Lambda_{\text{Dual}}}{\partial \beta} = -\mathbf{y}^T \alpha = 0$$

# IPM to approach SVM

- Stationary point found using the primal form:

$$\frac{\partial \Lambda}{\partial \mathbf{S}} = -\mu \mathbf{S}^{-1} + \lambda = 0 \Rightarrow \mu = \mathbf{S}\lambda$$

$$\frac{\partial \Lambda}{\partial b} = -\mathbf{y}^T \lambda = 0$$

$$\frac{\partial \Lambda}{\partial \xi} = C - \mu \xi^{-1} - \lambda = 0$$

$$\frac{\partial \Lambda}{\partial \lambda} = \mathbf{Q}\alpha + \mathbf{y}b + \boxed{\xi} - \mathbf{s} - \mathbf{e} = 0$$

- Stationary point found using the Dual form:

Observe  $\xi$  must be equal to

$$\frac{\Lambda_{\text{Dual}}}{\partial \alpha} = \mathbf{e} - \mathbf{Q}\alpha + \mu\alpha^{-1} - \boxed{\mu(C - \alpha)^{-1}} - \beta\mathbf{y} = 0$$

$$\frac{\Lambda_{\text{Dual}}}{\partial \beta} = -\mathbf{y}^T \alpha = 0$$

- So, the primal-dual path following IPM is based on solving the following system of equations:

Having  $\lambda = \alpha$  we have:

$$\begin{cases} \mathbf{Q}\alpha + \mathbf{y}^T b + \xi - \mathbf{s} - \mathbf{e} = 0 \\ -\mathbf{y}^T \alpha = 0 \\ \mathbf{S}\alpha - \mu = 0 \\ (C - \alpha)\xi - \mu = 0 \end{cases}$$

**How can we solve this system?**

# IPM to approach SVM

- We will apply the Newton's Method to solve the system:

Having  $\lambda = \alpha$  we have:

$$\begin{cases} \mathbf{Q}\alpha + \mathbf{y}b + \xi - \mathbf{s} - \mathbf{e} = 0 \\ -\mathbf{y}^T\alpha = 0 \\ \mathbf{S}\alpha - \mu = 0 \\ (C - \alpha)\xi - \mu = 0 \end{cases}$$

- So we will build the Jacobian matrix as follows:

$$\begin{pmatrix} \alpha & b & \mathbf{s} & \xi \\ \mathbf{Q} & \mathbf{y} & -\mathbf{I} & \mathbf{I} \\ -\mathbf{y}^T & 0 & 0 & 0 \\ \mathbf{S} & 0 & \alpha & 0 \\ -\xi & 0 & 0 & (C - \alpha) \end{pmatrix}$$

# IPM to approach SVM

- Thus we apply the Newton's Method

$$\mathbf{J}(\mathbf{x}^n) \Delta_{\mathbf{x}} = -\mathbf{f}(\mathbf{x}^n)$$

- To solve:

$$\begin{pmatrix} \mathbf{Q} & \mathbf{y} & -\mathbf{I} & \mathbf{I} \\ -\mathbf{y}^T & 0 & 0 & 0 \\ \mathbf{S} & 0 & \alpha & 0 \\ -\xi & 0 & 0 & (C - \alpha) \end{pmatrix} \begin{pmatrix} d_{\alpha} \\ d_b \\ d_s \\ d_{\xi} \end{pmatrix} = - \begin{pmatrix} \mathbf{Q}\alpha + \mathbf{y}b + \xi - \mathbf{s} - \mathbf{e} \\ -\mathbf{y}^T\alpha \\ \mathbf{S}\alpha - \mu \\ (C - \alpha)\xi - \mu \end{pmatrix}$$

# IPM to approach SVM

- Now we will implement this primal-dual path following interior point method to solve SVM
  - Instead of solving as we did with the previous IPM problem, we will use a **solver**
    - Because this problem is more complex
    - The Jacobian matrix of the other problem is almost in the **row canonical form**

# IPM to approach SVM

- As first step we need to figure out how the Jacobian matrix will be written to apply a solver

$$\begin{pmatrix} \mathbf{Q} & \mathbf{y} & -\mathbf{I} & \mathbf{I} \\ -\mathbf{y}^T & 0 & 0 & 0 \\ \mathbf{S} & 0 & \alpha & 0 \\ -\xi & 0 & 0 & (C - \alpha) \end{pmatrix} \begin{pmatrix} d_\alpha \\ d_b \\ d_s \\ d_\xi \end{pmatrix} = - \begin{pmatrix} \mathbf{Q}\alpha + \mathbf{y}b + \xi - \mathbf{s} - \mathbf{e} \\ -\mathbf{y}^T\alpha \\ \mathbf{S}\alpha - \mu \\ (C - \alpha)\xi - \mu \end{pmatrix}$$

- We will write it as follows:
  - For example, consider a training dataset with 200 observations
  - And let  $\mathbf{Q} = (\langle \mathbf{y}, \mathbf{y}^T \rangle)(\langle \mathbf{x}, \mathbf{x}^T \rangle)$ , as we defined before
  - Consider  $\mathbf{S}$ ,  $\boldsymbol{\alpha}$  and  $\xi$  as diagonal matrices

# IPM to approach SVM

- As first step we need to figure out how the Jacobian matrix will be written to apply a solver

$$\begin{pmatrix} \mathbf{Q} & \mathbf{y} & -\mathbf{I} & \mathbf{I} \\ -\mathbf{y}^T & 0 & 0 & 0 \\ \mathbf{S} & 0 & \alpha & 0 \\ -\xi & 0 & 0 & (C - \alpha) \end{pmatrix} \begin{pmatrix} d_\alpha \\ d_b \\ d_s \\ d_\xi \end{pmatrix} = - \begin{pmatrix} \mathbf{Q}\alpha + \mathbf{y}b + \xi - \mathbf{s} - \mathbf{e} \\ -\mathbf{y}^T\alpha \\ \mathbf{S}\alpha - \mu \\ (C - \alpha)\xi - \mu \end{pmatrix}$$

- So we will have a Jacobian such as:

$$\mathbf{J}(\mathbf{x}^n) = \begin{matrix} \begin{array}{c|c|c|c} \mathbf{Q} & \mathbf{y} & -\mathbf{I} & \mathbf{I} \\ 200 \times 200 & & 200 \times 200 & 200 \times 200 \end{array} \\ \hline \begin{array}{c|c} -\mathbf{y}^T & \boldsymbol{\alpha} \\ 1 \times 200 & 200 \times 200 \end{array} \\ \hline \begin{array}{c|c} \mathbf{S} & (\mathbf{C} - \boldsymbol{\alpha}) \\ 200 \times 200 & 200 \times 200 \end{array} \\ \hline \begin{array}{c|c} -\xi & \\ 200 \times 200 & \end{array} \end{matrix}$$

Observe this is a  $3n+1 \times 3n+1$  matrix, considering  $n$  as the number of training examples

It is a square matrix

Other cells assume zero

# IPM to approach SVM

- As first step we need to figure out how the Jacobian matrix will be written to apply a solver

$$\begin{pmatrix} \mathbf{Q} & \mathbf{y} & -\mathbf{I} & \mathbf{I} \\ -\mathbf{y}^T & 0 & 0 & 0 \\ \mathbf{S} & 0 & \alpha & 0 \\ -\xi & 0 & 0 & (C - \alpha) \end{pmatrix} \begin{pmatrix} d_\alpha \\ d_b \\ d_s \\ d_\xi \end{pmatrix} = - \begin{pmatrix} \mathbf{Q}\alpha + \mathbf{y}b + \xi - \mathbf{s} - \mathbf{e} \\ -\mathbf{y}^T\alpha \\ \mathbf{S}\alpha - \mu \\ (C - \alpha)\xi - \mu \end{pmatrix}$$

- Thus we will build the Jacobian matrix and the right-hand term of the equation below as an one-column matrix

$$\mathbf{J}(\mathbf{x}^n) \Delta_{\mathbf{x}} = -\mathbf{f}(\mathbf{x}^n)$$

- Then we will apply a solver

# IPM to approach SVM

- As first step we need to figure out how the Jacobian matrix will be written to apply a solver

$$\begin{pmatrix} \mathbf{Q} & \mathbf{y} & -\mathbf{I} & \mathbf{I} \\ -\mathbf{y}^T & 0 & 0 & 0 \\ \mathbf{S} & 0 & \alpha & 0 \\ -\xi & 0 & 0 & (C - \alpha) \end{pmatrix} \begin{pmatrix} d_\alpha \\ d_b \\ d_s \\ d_\xi \end{pmatrix} = - \begin{pmatrix} \mathbf{Q}\alpha + \mathbf{y}b + \xi - \mathbf{s} - \mathbf{e} \\ -\mathbf{y}^T\alpha \\ \mathbf{S}\alpha - \mu \\ (C - \alpha)\xi - \mu \end{pmatrix}$$

- The right-hand term or the answer will be in form:

$$-\mathbf{f}(\mathbf{x}^n) = \begin{matrix} 200 \times 1 \\ \hline 1 \times 1 \\ \hline 200 \times 1 \\ \hline 200 \times 1 \end{matrix}$$

# IPM to approach SVM

- As first step we need to figure out how the Jacobian matrix will be written to apply a solver

$$\begin{pmatrix} \mathbf{Q} & \mathbf{y} & -\mathbf{I} & \mathbf{I} \\ -\mathbf{y}^T & 0 & 0 & 0 \\ \mathbf{S} & 0 & \alpha & 0 \\ -\xi & 0 & 0 & (C - \alpha) \end{pmatrix} \begin{pmatrix} d_\alpha \\ d_b \\ d_s \\ d_\xi \end{pmatrix} = - \begin{pmatrix} \mathbf{Q}\alpha + \mathbf{y}b + \xi - \mathbf{s} - \mathbf{e} \\ -\mathbf{y}^T\alpha \\ \mathbf{S}\alpha - \mu \\ (C - \alpha)\xi - \mu \end{pmatrix}$$

- Besides that, we will also need:
  - An initial solution  $\boldsymbol{\alpha}$  with feasible values, i.e.,  $\boldsymbol{\alpha} \geq 0$
  - Define  $\xi$
  - Define an initial value for  $b$
  - Define  $\mathbf{S}$  in terms of  $\boldsymbol{\alpha}$ ,  $\xi$  and  $b$

# IPM to approach SVM

- We will consider  $\mu$  as a positive value but very close to zero, i.e., providing less relevance to the barrier term
- See source code ***03-svm-ipm.r***
  - Observe we sum a small number in the diagonal to guarantee solution for  $Q$ , otherwise it may be a singular matrix

# IPM to approach SVM

- Study the paper below to understand this method in details
  - Fine S., Scheinberg, K. (2001) Efficient SVM Training Using Low-Rank Kernel Representations, Journal of Machine Learning Research, 2 (2001), 243-264
- This paper proposes this method we saw as the basis, which is then modified to provide better results
  - Part of the improvements this paper considers are:
    - How to compute the step in the direction of the solution
    - It considers a predictor-corrector step to improve results
    - It considers the lowest rank of matrix Q, etc.

**Now we will see the package LowRankQP**

- This is a package for the R statistical software
- It implements the paper:
  - Fine S., Scheinberg, K. (2001) Efficient SVM Training Using Low-Rank Kernel Representations, Journal of Machine Learning Research, 2 (2001), 243-264

- Now we will simply use it...
  - See source ***04-polynomial-svm.r***
    - Test this package using a simple linear separable dataset
    - Test this package using more complex datasets
    - Try other kernels!

- These slides introduced optimization problems:
  - From linear problems
  - To convex problems
- Then it supported the formulation and implementation of SVM as a convex problem
  - Using the primal-dual path following interior point method

- **On Primal and Dual forms:**
  - <https://www.youtube.com/watch?v=KMmgF3ZaBRE>
- **On the Simplex Method:**
  - <https://www.youtube.com/watch?v=gRgsT9BB5-8>
  - <https://www.youtube.com/watch?v=yL7JByLfrew>
  - <https://www.youtube.com/watch?v=vVzjXpwW2xI>
  - <https://www.youtube.com/watch?v=lPm46c1pfvQ>
  - <https://www.youtube.com/watch?v=WeK4JjNLSgw>

# References

- **Newton's Method:**
  - Kendall E. Atkinson, An Introduction to Numerical Analysis, (1989) John Wiley & Sons, Inc, ISBN 0-471-62489-6
  - Tjalling J. Ypma, Historical development of the Newton-Raphson method, SIAM Review 37 (4), 531–551, 1995. doi:10.1137/1037125.
- **About Interior Point Methods:**
  - Gondzio, J. (2009) Using interior point methods for optimization in training very large scale Support Vector Machines, School of Mathematics, Montreal (Joint work with Kristian Woodsend)
    - Available at: [http://numml.kyb.tuebingen.mpg.de/numl09/talk\\_gondzio.pdf](http://numml.kyb.tuebingen.mpg.de/numl09/talk_gondzio.pdf)
  - Ormerod, J. T.; Wand, M.P.; Koch, I. (2007) Penalised Spline Support Vector Classifiers: Computational Issues, School of Mathematics and Statistics, University of New South Wales, Australia
  - Fine S.; Scheinberg, K. (2001) Efficient SVM Training Using Low-Rank Kernel Representations, Journal of Machine Learning Research, 2 (2001), 243-264 <http://www.jmlr.org/papers/volume2/fine01a/fine01a.pdf>

- **About Interior Point Methods:**

- Ormerod, J. T.; Wand, M. P. (2009) Package LowRankQP.
  - Available at:  
<http://cran.r-project.org/web/packages/LowRankQP/LowRankQP.pdf>
- Jensen, P. A.; Bard, J. F. (2002) Operations Research Models and Methods. Chapter: LP Methods.S4 - Interior Point Methods. Wiley. ISBN: 978-0-471-38004-7
- Andersen, E. D.; Gondzio, J.; Mészáros; Xu, X. (1993) Implementation of Interior Point Methods for Large Scale Linear Programming. Technical Report 1996.3. Logilab, HEC Geneva, Section of Management Studies, University of Geneva.
- Other source code using the Package LowRankQP.
  - Available at: <https://github.com/hkreeves>

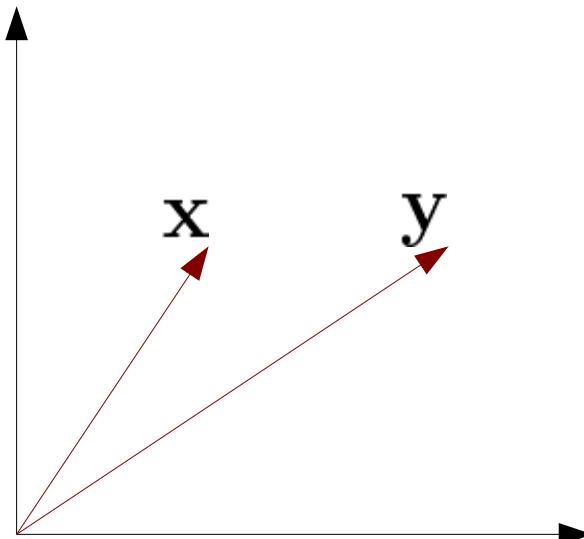
# **An Introduction on Kernels**

- Practical approach:
  - How kernels provide different feature spaces?
- Theoretical approach:
  - Some Linear Algebra foundation to support the understanding of kernels
    - Linear Transformation, Basis, Change in Basis, Orthonormal Basis, Eigenvalues and Eigenvectors, and Principal Component Analysis
  - More about Spaces to understand where the Hilbert space fits in
  - Back to the SVM optimization
    - Why to use Gram matrices?
    - How to assess whether a kernel is useful to tackle a problem?
    - What is the kernel trick?
    - Which theorem guarantees the kernel trick?

# Optimizer based on Package LowRankQP

- Now we have a reliable optimizer which uses the Package LowRankQP
  - Then we could try the most common kernels:
    - Linear
    - Polynomial
      - Homogeneous
      - Inhomogeneous
    - Sigmoidal
    - Radial

- Kernel:
  - Linear
- Equation:
$$\langle \mathbf{x}, \mathbf{y} \rangle$$
- Operation: Simply the **dot product**
- Space:



# Polynomial Kernel

- Kernel:
  - Polynomial (homogeneous and inhomogeneous)
- Equations:
  - Homogeneous:  $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle^d$
  - Inhomogeneous:  $k(\mathbf{x}_i, \mathbf{x}_j) = (\langle \mathbf{x}_i, \mathbf{x}_j \rangle + 1)^d$
- Operation (Homogeneous as example):

Consider  $\mathbb{R}^2$  and  $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ ,  $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$  then we have:

$$k(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x}, \mathbf{y} \rangle^2 = (x_1 y_1 + x_2 y_2)^2 = x_1^2 y_1^2 + 2x_1 y_1 x_2 y_2 + x_2^2 y_2^2$$

But how to represent that as a dot product in another space?

# Polynomial Kernel

- Kernel:
  - Polynomial (homogeneous and inhomogeneous)
- Operation (Homogeneous as example):

Observe the result

$$k(\mathbf{x}, \mathbf{y}) = x_1^2 y_1^2 + 2x_1 y_1 x_2 y_2 + x_2^2 y_2^2$$

can be written as the dot product

$$\begin{bmatrix} x_1^2 \\ x_1 x_2 \\ x_1 x_2 \\ x_2^2 \end{bmatrix} \cdot \begin{bmatrix} y_1^2 \\ y_1 y_2 \\ y_1 y_2 \\ y_2^2 \end{bmatrix}$$

so, how is the space where the dot product happens?

# Polynomial Kernel

- Kernel:
  - Polynomial (homogeneous and inhomogeneous)
- Space:
  - The space has **four** dimensions
    - So we leave a 2-dimensional space and go to a 4-dimensional space
  - Every input vector in the **data space** is transformed to another vector in the **feature space**
    - Observe two axes are the same in the feature space, what could we do to simplify it?

# Polynomial Kernel

- Kernel:
  - Polynomial (homogeneous and inhomogeneous)
- Space:
  - Simplifying the feature space we will have three dimensions:

Observe the result

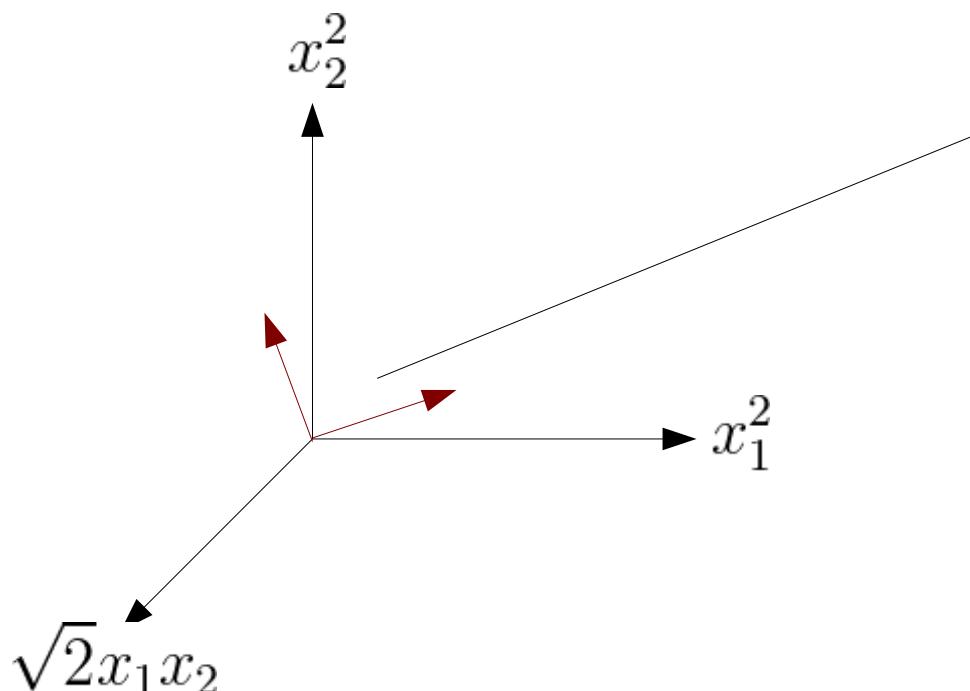
$$k(\mathbf{x}, \mathbf{y}) = x_1^2 y_1^2 + 2x_1 y_1 x_2 y_2 + x_2^2 y_2^2$$

we can also write as the dot product

$$\begin{bmatrix} x_1^2 \\ \sqrt{2}x_1 x_2 \\ x_2^2 \end{bmatrix} \cdot \begin{bmatrix} y_1^2 \\ \sqrt{2}y_1 y_2 \\ y_2^2 \end{bmatrix}$$

# Polynomial Kernel

- Kernel:
  - Polynomial (homogeneous and inhomogeneous)
- Space:
  - For example, this 3-dimensional space is something like (for every vector  $\mathbf{x}$ ):



So the dot product now happens between these two other transformed vectors

# Gaussian (Radial Basis) Kernel

- Kernel:
  - Gaussian
- Equation:  $k(\mathbf{x}, \mathbf{y}) = \exp(-\gamma \|\mathbf{x} - \mathbf{y}\|^2)$ , for  $\gamma > 0$ .  
Sometimes parametrized using  $\gamma = 1/2\sigma^2$
- Operation:

Consider  $\mathbb{R}^2$  and  $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ ,  $\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$  then we have:

$$\begin{aligned} k(\mathbf{x}, \mathbf{y}) &= \exp(-\gamma \|\mathbf{x} - \mathbf{y}\|^2) = \exp(-\gamma \left\| \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} - \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \right\|^2) = \\ &= \exp(-\gamma \left\| \begin{bmatrix} x_1 - y_1 \\ x_2 - y_2 \end{bmatrix} \right\|^2) \end{aligned}$$

But, how will be this space?

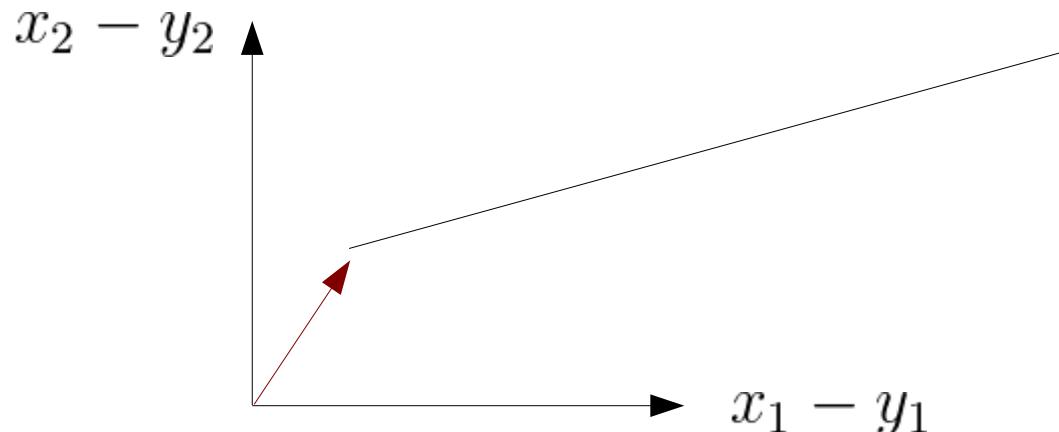
# Gaussian (Radial Basis) Kernel

- Kernel:
  - Gaussian
- Equation:  $k(\mathbf{x}, \mathbf{y}) = \exp(-\gamma \|\mathbf{x} - \mathbf{y}\|^2)$ , for  $\gamma > 0$ .  
Sometimes parametrized using  $\gamma = 1/2\sigma^2$
- Space:
  - Every possible pair of vectors in **data space** will result in another difference vector in **feature space**
    - Observe we basically compute the norm of the difference vector
$$k(\mathbf{x}, \mathbf{y}) = \exp(-\gamma \left\| \begin{bmatrix} x_1 - y_1 \\ x_2 - y_2 \end{bmatrix} \right\|^2)$$
    - After having the norm we apply the Gaussian activation function

# Gaussian (Radial Basis) Kernel

- Kernel:
  - Gaussian
- Equation:
$$k(\mathbf{x}, \mathbf{y}) = \exp(-\gamma \|\mathbf{x} - \mathbf{y}\|^2), \text{ for } \gamma > 0.$$

Sometimes parametrized using  $\gamma = 1/2\sigma^2$
- Space:
  - So if the data space has 2 dimensions, the feature space will also have 2 dimensions



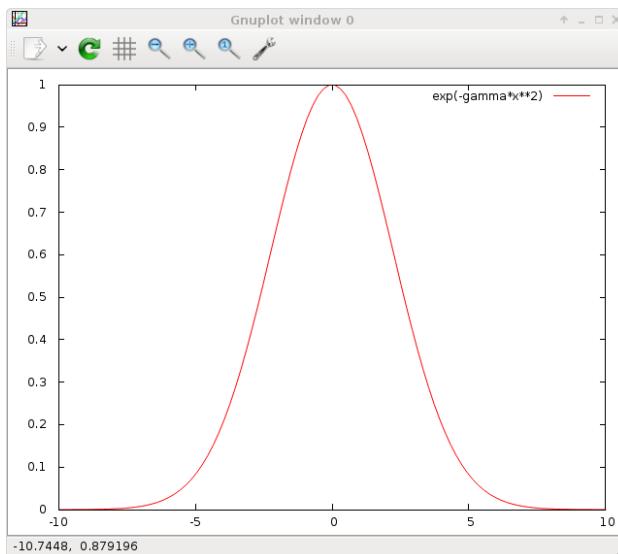
This is the difference vector.  
The only thing that matters  
to the Gaussian kernel is the  
length of this vector!

As shorter it is, the more  
similar vectors  $\mathbf{x}$  and  $\mathbf{y}$  are

# Gaussian (Radial Basis) Kernel

- Kernel:
  - Gaussian
- Equation:
$$k(\mathbf{x}, \mathbf{y}) = \exp(-\gamma \|\mathbf{x} - \mathbf{y}\|^2), \text{ for } \gamma > 0.$$

Sometimes parametrized using  $\gamma = 1/2\sigma^2$
- Space:
  - After having the length of the difference vector, we simply apply the Gaussian activation function as follows:



**Gamma modifies how the Gaussian is spread around the center**

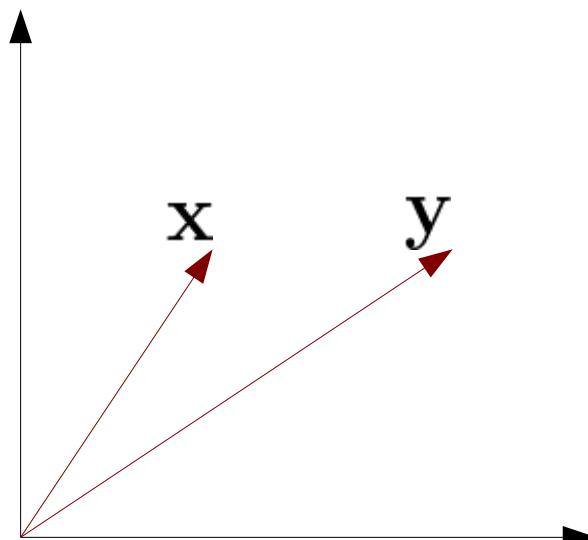
# Sigmoidal Kernel

- Kernel:
  - Sigmoidal

- Equation:

$$k(\mathbf{x}, \mathbf{y}) = \tanh(\kappa \mathbf{x} \cdot \mathbf{y} + c), \text{ for some (not every) } \kappa > 0 \text{ and } c < 0$$

- Operation: It proceeds with the **dot product**
- Space:



The space is the same as the Linear Kernel, however kappa is used to modify the magnitude of the dot product

Besides c is summed to the result to shift the tanh curve

And mostly important the tanh is used to provide a continuous output in range [-1, 1]

We will now see a practical example using two different kernels: linear and 2<sup>nd</sup>-order polynomial

# Practical example of kernels

- Now we will see the results of those kernels for the dataset produced with the R commands:

```
require(tseriesChaos)  
data_space = cbind(embedd(sin(seq(0,9,len=1000)), m=2, d=175), rep(0, 825))  
data_space = rbind(data_space, cbind(rnorm(mean=0, sd=0.1, n=825),  
rnorm(mean=0, sd=0.1, n=825), rep(1, 825)))  
plot(data_space[,1:2], col=data_space[,3]+1)
```

- What about?
  - We apply the linear kernel and see results?
  - We apply the polynomial kernel with order equals to 2 and check out results?

# Practical example of kernels

- What can we say about the spaces provided by these two kernels?
  - The linear kernel uses the original data space to proceed with the classification
  - What about the polynomial kernel?

# Practical example of kernels

- What can we say about the spaces provided by these two kernels?
  - The linear kernel uses the original data space to proceed with the classification
  - What about the polynomial kernel?
    - It builds the **feature space** with vectors in form:

```
require(rgl)
feature_space =
  cbind(data_space[,1]^2, sqrt(2)*data_space[,1]*data_space[,2],
        data_space[,2]^2)
plot3d(feature_space, col=data_space[,3]+1)
```
    - Are classes linearly separable in this feature space?
      - This is simply what a kernel does!

# Practical example of kernels

- What conclusions can we draw?
  - What is the best feature space for some data?
    - It is the one which provides linear separation

# Practical example of kernels

- What conclusions can we draw?
  - Can we say SVM works just building linear hyperplanes?
    - Yes, we can. But SVM builds those linear hyperplanes in features space

# Practical example of kernels

- What conclusions can we draw?
  - Is there any classification algorithm that could provide better results than SVM?
    - Well, if you find the best kernel to transform the data space to the features space, NO!
      - The maximal margin theorem guarantees SVM provides the best linear separation between classes

# Practical example of kernels

- What conclusions can we draw?
  - Is there any classification algorithm that could provide better results than SVM?
    - Well, if you find the best kernel to transform the data space to the features space, NO!
      - The maximal margin theorem guarantees SVM provides the best linear separation between classes
      - So if you have the best (not necessarily the best, but a fair enough) features space, how could another classification algorithm outperform SVM?

# Practical example of kernels

- What conclusions can we draw?
  - Is there any classification algorithm that could provide better results than SVM?
    - Well, if you find the best kernel to transform the data space to the features space, NO!
      - The maximal margin theorem guarantees SVM provides the best linear separation between classes
      - So if you have the best (not necessarily the best, but a fair enough) features space, how could another classification algorithm outperform SVM?
      - So, does it make sense to design new classification algorithms?

# Practical example of kernels

- What conclusions can we draw?
  - So what does make sense?
    - To study your data space and find a kernel that provides a fair enough features space!

# Practical example of kernels

- What conclusions can we draw?
  - So what does make sense?
    - To study your data space and find a kernel that provides a fair enough features space!
    - So, what does it mean when I compare SVM to another technique such as MLP and this second provides greater accuracy?

# Practical example of kernels

- What conclusions can we draw?
  - So what does make sense?
    - To study your data space and find a kernel that provides a fair enough features space!
    - So, what does it mean when I compare SVM to another technique such as MLP and this second provides greater accuracy?
      - Either you consider more hyperplanes for MLP, thus the space has to be “cut” or “shattered” more times! What may be necessary

# Practical example of kernels

- What conclusions can we draw?
  - So what does make sense?
    - To study your data space and find a kernel that provides a fair enough features space!
    - So, what does it mean when I compare SVM to another technique such as MLP and this second provides greater accuracy?
      - Either you consider more hyperplanes for MLP, thus the space has to be “cut” or “shattered” more times! What may be necessary
      - Or MLP is overfitting

# Practical example of kernels

- What conclusions can we draw?
  - So what does make sense?
    - To study your data space and find a kernel that provides a fair enough features space!
    - So, what does it mean when I compare SVM to another technique such as MLP and this second provides greater accuracy?
      - Either you consider more hyperplanes for MLP, thus the space has to be “cut” or “shattered” more times! What may be necessary
      - Or MLP is overfitting
      - How could I approach this with SVM?
        - Build a good kernel to get a linearly separable features space

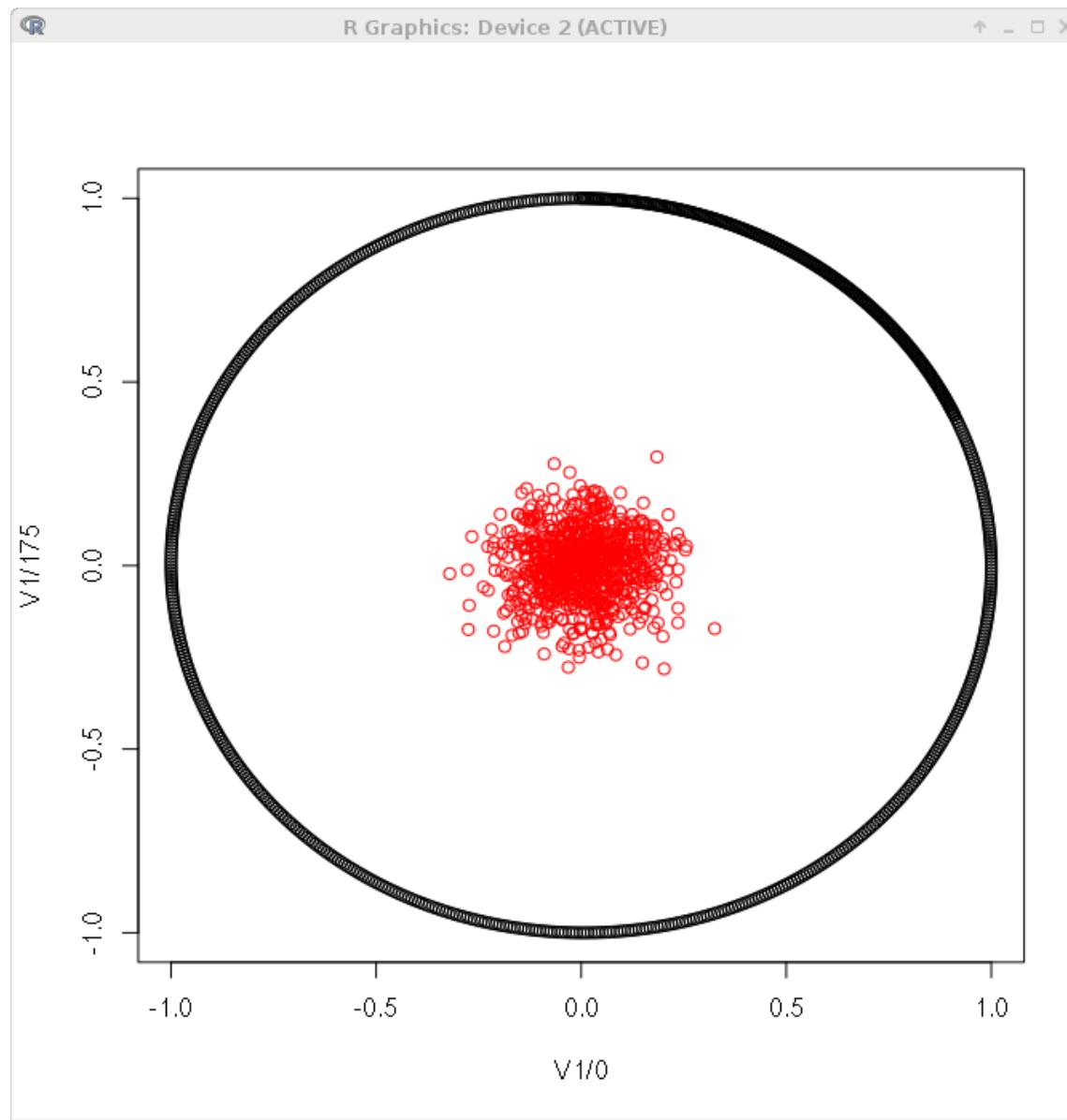
# Practical example of kernels

- Now, we will see an example using:
  - MLP
  - SVM
- Let the data space be defined by:

```
require(tseriesChaos)
data_space = cbind(embedd(sin(seq(0,9,len=1000)), m=2, d=175), rep(0, 825))
data_space = rbind(data_space, cbind(rnorm(mean=0, sd=0.1, n=825),
rnorm(mean=0, sd=0.1, n=825), rep(1, 825)))
plot(data_space[,1:2], col=data_space[,3]+1)
```

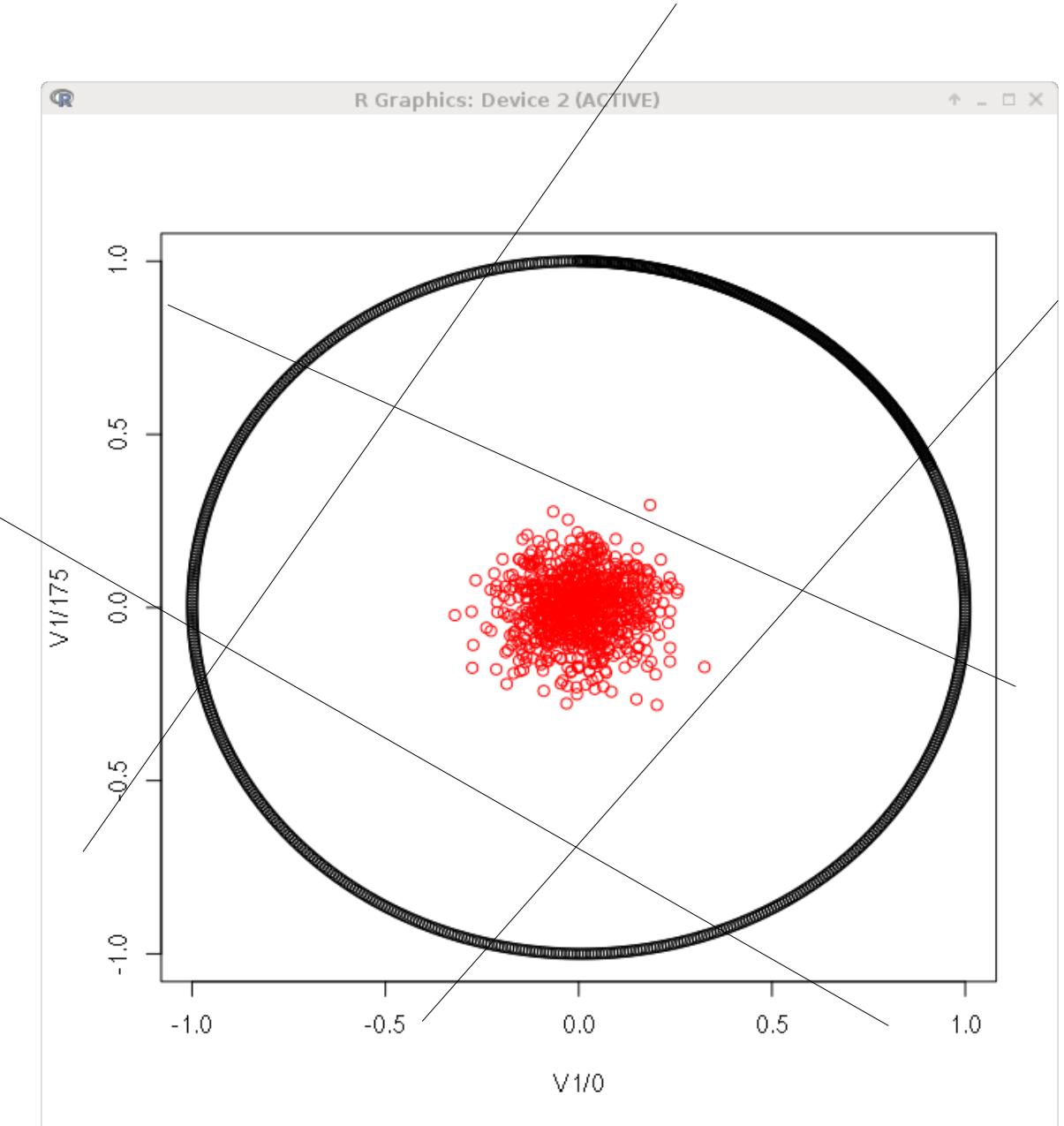
# Practical example of kernels

- Data space:



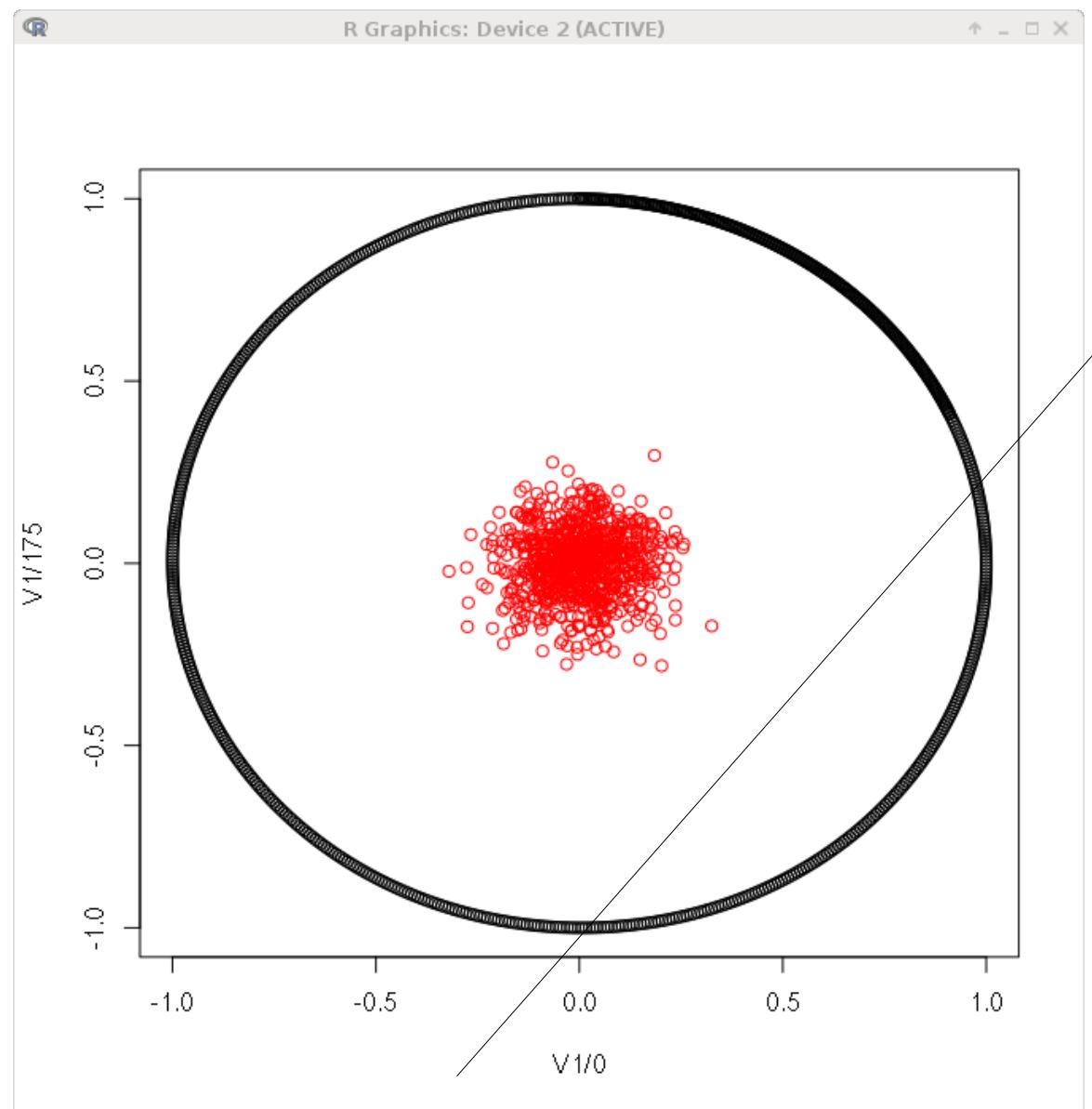
# Practical example of kernels

- MLP on Data space
  - With four hyperplanes MLP can separate classes
  - Observe hyperplanes do not guarantee maximal margin
  - How will be the VC dimension? And consequently the generalization guarantees as we train with more examples?



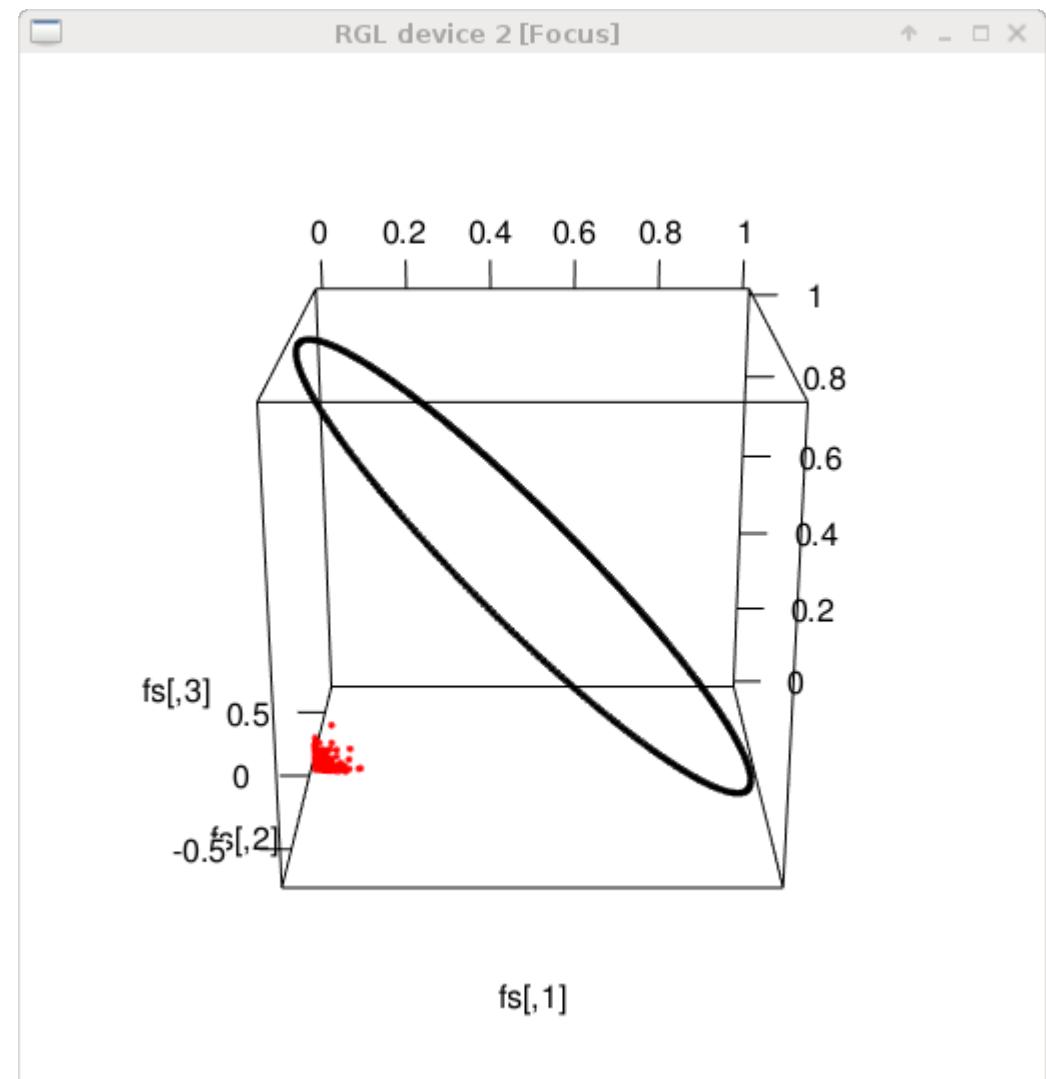
# Practical example of kernels

- SVM on Data space
  - As we have two classes, SVM will optimize the maximal margin for a single hyperplane
  - Results will be good? No, because data space is not ideal
  - What happens if we use a second-order polynomial kernel?



# Practical example of kernels

- SVM on the features space provided by the polynomial kernel of order 2
  - Now we can find a single linear hyperplane to separate both classes!
  - What happens with the VC dimension? And consequently the generalization guarantees as we train with more examples?



# Practical example of kernels

- Homework:
  - As application examples, perform experiments using UCI datasets such as (select only two classes to simplify):
    - Iris
    - Wine
    - Abalone
    - Breast Cancer Wisconsin
    - Ecoli
    - Teaching Assistant Evaluation
    - Yeast
    - KDD Cup 1999
    - Perfume
  - Look for SVM implementations using more than two classes
    - For example: <https://www.sec.in.tum.de/assets/lehre/ws0910/ml/slideslecture9.pdf>

Now we will remember some Linear Algebra to carry on  
analyzing the SVM optimization and Kernels

# Linear Transformation

- What is a linear transformation?

Consider  $T : \mathbb{R}^n \rightarrow \mathbb{R}^n$  as a transformation.

It is called linear transformation iff:

1.  $T(\mathbf{a} + \mathbf{b}) = T(\mathbf{a}) + T(\mathbf{b})$
2.  $T(c\mathbf{a}) = cT(\mathbf{a})$

- As homework, prove the following **is a linear transformation**:

$$T(x_1, x_2) = (x_1 + x_2, 3x_1) \text{ for any vector } \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in \mathbb{R}^2$$

- To remember:

[https://pt.khanacademy.org/math/linear-algebra/matrix\\_transformations/linear\\_transformations/v/linear-transformations](https://pt.khanacademy.org/math/linear-algebra/matrix_transformations/linear_transformations/v/linear-transformations)

# Linear Transformation

- As homework, prove the following **is not a linear transformation**:

$$T(x_1, x_2) = (x_1^2, \sqrt{2}x_1x_2, x_2^2) \text{ for any vector } \mathbf{x} \in \mathbb{R}^2$$

# Linear Transformation

- Can we represent a linear transformation using matrices?
  - Yes
- Let's consider the previous example:

$$T(x_1, x_2) = (x_1 + x_2, 3x_1) \text{ for any vector } \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in \mathbb{R}^2$$

- How would be such a matrix?

# Linear Transformation

- Can we represent a linear transformation using matrices?
  - Yes
- Let's consider the previous example:

$$T(x_1, x_2) = (x_1 + x_2, 3x_1) \text{ for any vector } \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in \mathbb{R}^2$$

- How would be such a matrix?

$$A = \begin{bmatrix} 1 & 1 \\ 3 & 0 \end{bmatrix}$$

# Linear Transformation

- Can we represent a linear transformation using matrices?
  - Yes
- Let's consider the previous example:

$$T(x_1, x_2) = (x_1 + x_2, 3x_1) \text{ for any vector } \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in \mathbb{R}^2$$

- How would be such a matrix?

$$A = \begin{bmatrix} 1 & 1 \\ 3 & 0 \end{bmatrix}$$

- How to apply such a linear transformation?

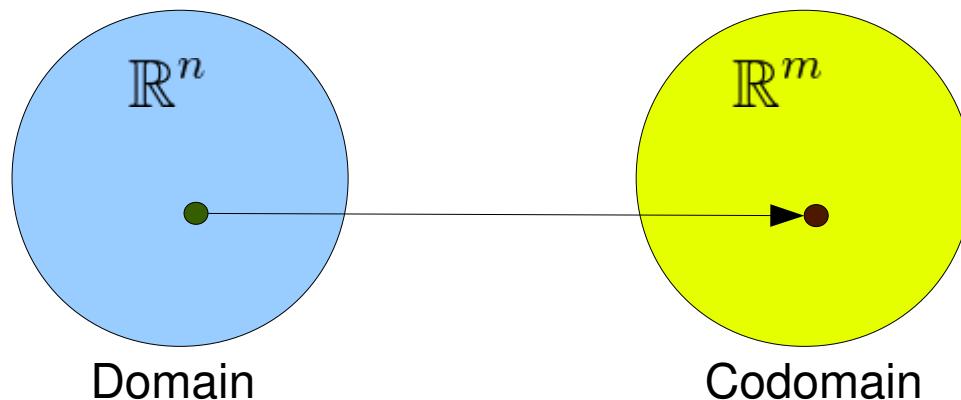
$$T(\mathbf{x}) = A\mathbf{x} = \begin{bmatrix} 1 & 1 \\ 3 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_1 + x_2 \\ 3x_1 \end{bmatrix}$$

# Linear Transformation

- We can see this linear transformation as follows:

$$T : \mathbb{R}^n \rightarrow \mathbb{R}^m$$

- Which maps the elements (vectors in this case) of the first set to the second set

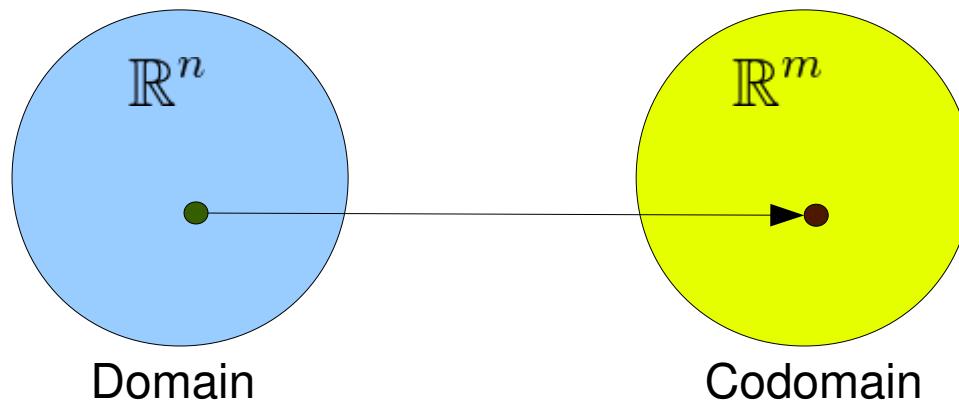


# Linear Transformation

- We can see this linear transformation as follows:

$$T : \mathbb{R}^n \rightarrow \mathbb{R}^m$$

- Which maps the elements (vectors in this case) of the first set to the second set



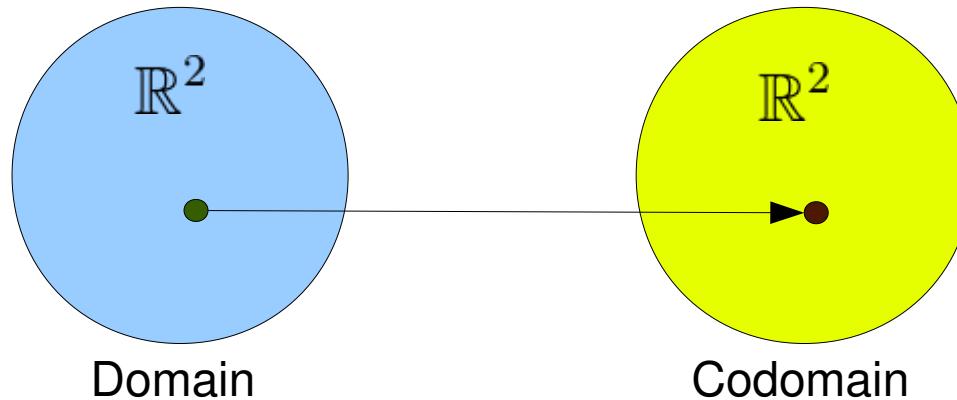
- What are we doing here? Can we use this to find a corresponding element in another space? **Yes**

# Linear Transformation

- In the example:

$$T(\mathbf{x}) = A\mathbf{x} = \begin{bmatrix} 1 & 1 \\ 3 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_1 + x_2 \\ 3x_1 \end{bmatrix}$$

- Our linear transformation works on sets:



- See more at:

[https://pt.khanacademy.org/math/linear-algebra/matrix\\_transformations/linear\\_transformations/v/vector-transformations](https://pt.khanacademy.org/math/linear-algebra/matrix_transformations/linear_transformations/v/vector-transformations)

# Linear Transformation

- Can we check it out what is the effect of the Linear Transformation:

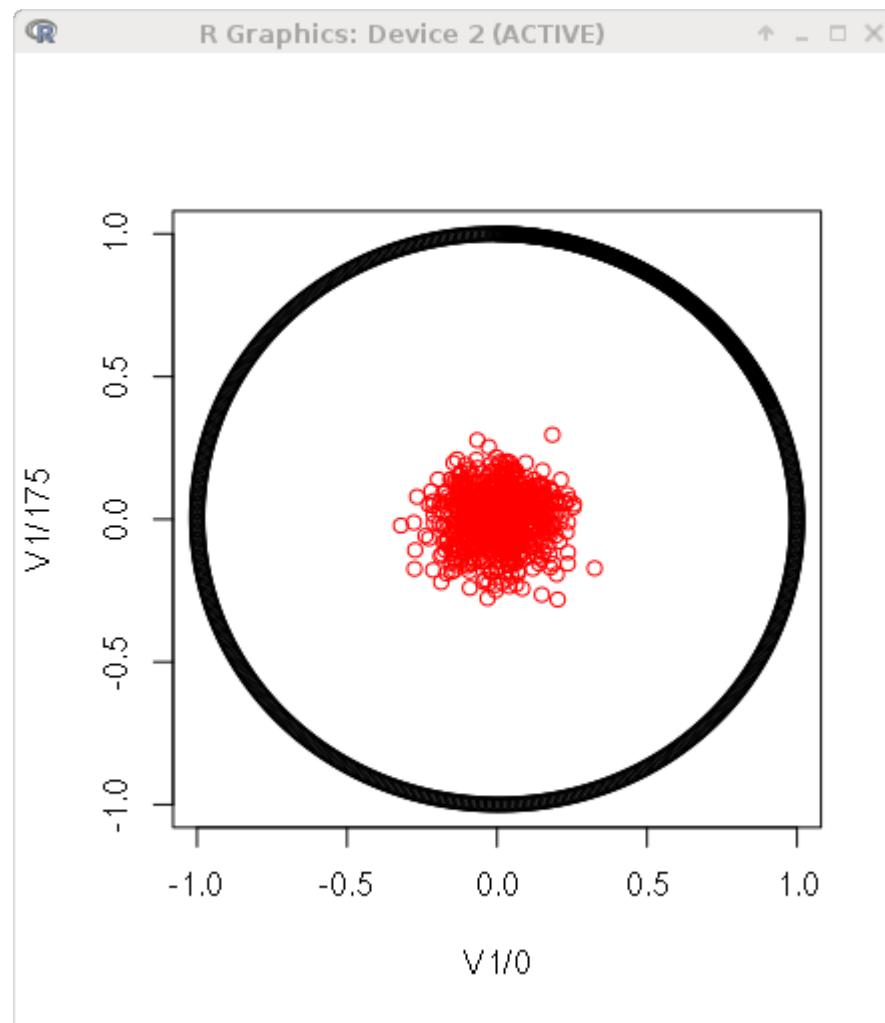
$$T(\mathbf{x}) = A\mathbf{x} = \begin{bmatrix} 1 & 1 \\ 3 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_1 + x_2 \\ 3x_1 \end{bmatrix}$$

- on the last dataset we created using?

```
require(tseriesChaos)
data_space = cbind(embedd(sin(seq(0,9,len=1000)), m=2, d=175), rep(0, 825))
data_space = rbind(data_space, cbind(rnorm(mean=0, sd=0.1, n=825),
                                     rnorm(mean=0, sd=0.1, n=825), rep(1, 825)))
plot(data_space[,1:2], col=data_space[,3]+1)
```

# Linear Transformation

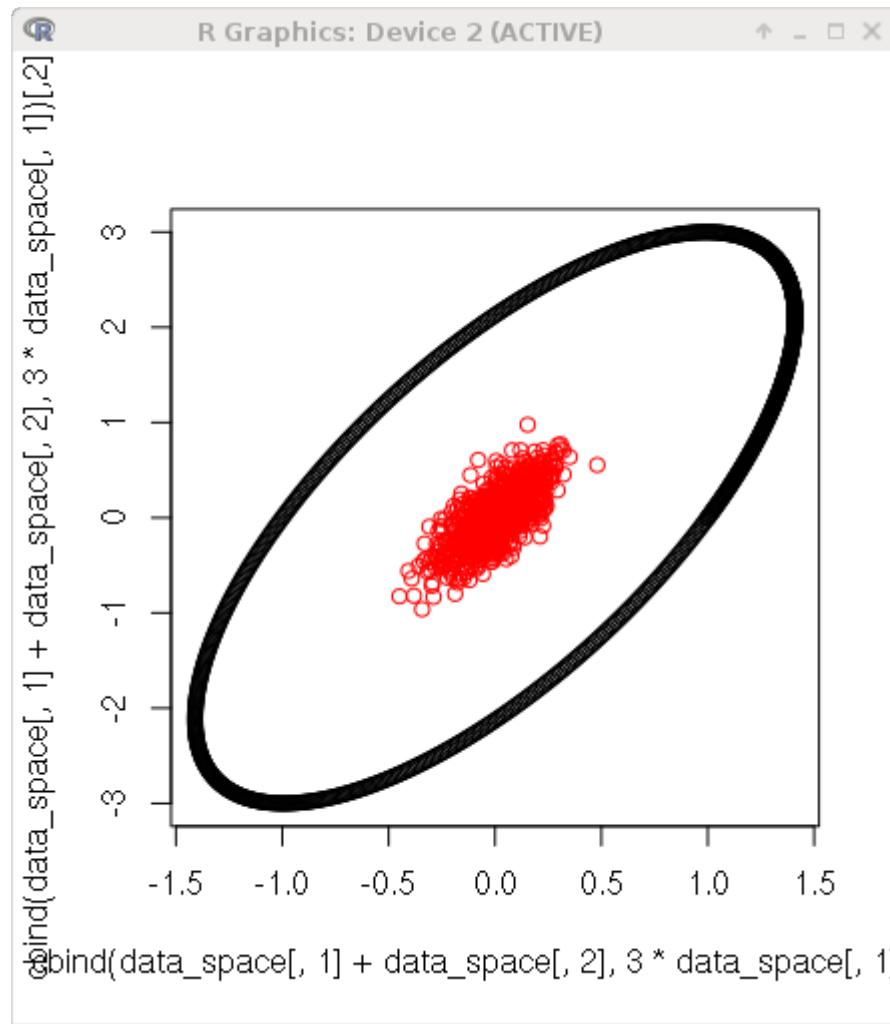
- Data space:



# Linear Transformation

- Features space using:

$$T(\mathbf{x}) = A\mathbf{x} = \begin{bmatrix} 1 & 1 \\ 3 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_1 + x_2 \\ 3x_1 \end{bmatrix}$$

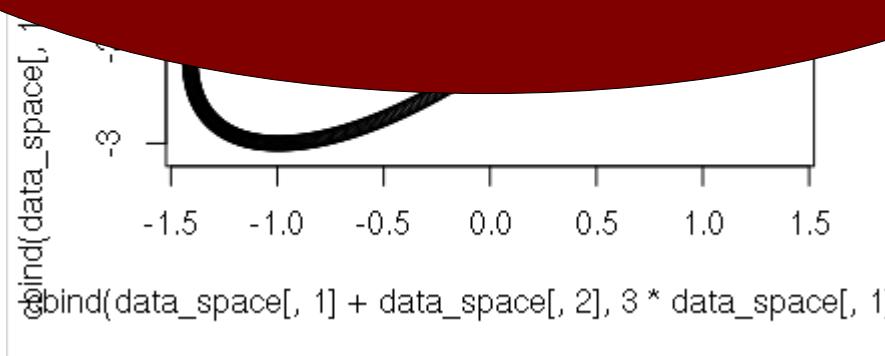


# Linear Transformation

- Features space using:

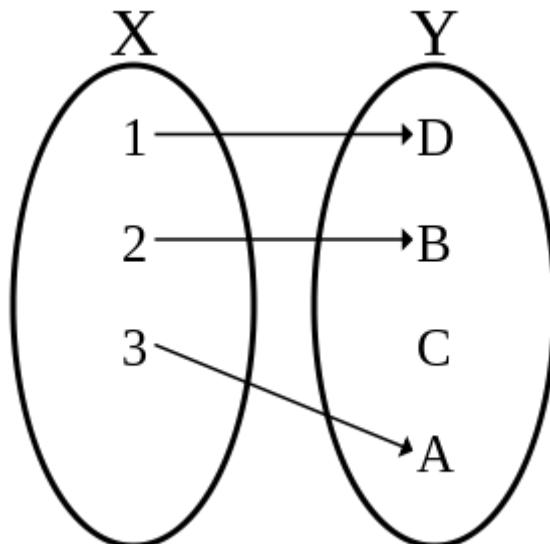
$$T(\mathbf{x}) = A\mathbf{x} = \begin{bmatrix} 1 & 1 \\ 3 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} x_1 + x_2 \\ 3x_1 \end{bmatrix}$$

Besides the poor features space we obtained, we already see some dependency between Linear Algebra and the study of Kernels!

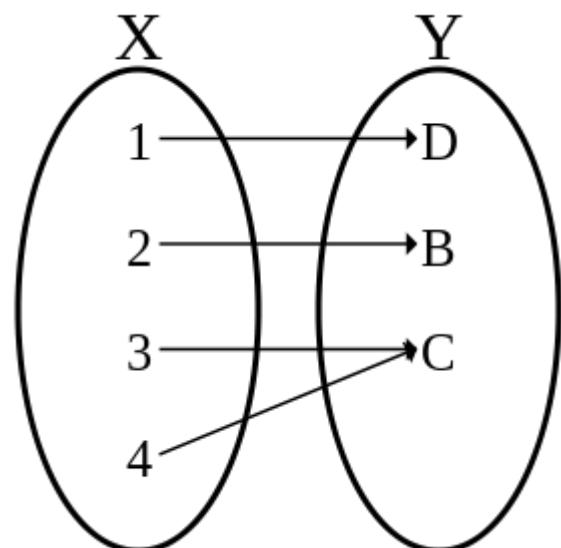


# Inverses of Linear Transformations

- Some important concepts are formalized in Linear Algebra using Linear Transformations:
  - Injective (one-to-one) and surjective (onto) functions



**Injective:** Never maps distinct elements of its domain to the same element of its codomain

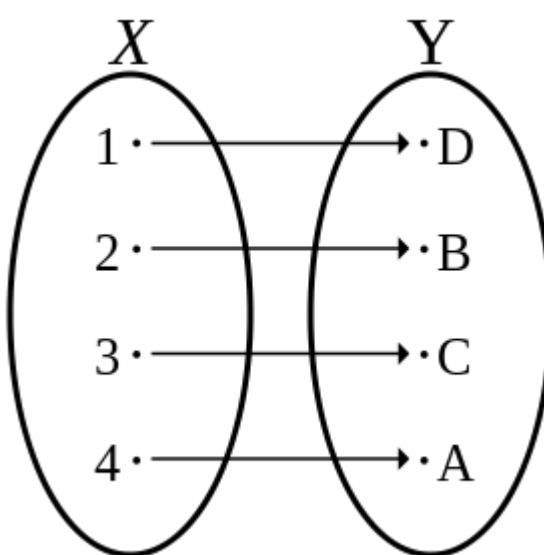


**Surjective:** Every element in Y has a corresponding element  $x$  in X such that  $f(x) = y$

- See more at:  
[https://pt.khanacademy.org/math/linear-algebra/matrix\\_transformations/inverse\\_transformations/v/linear-algebra-introduction-to-the-inverse-of-a-function](https://pt.khanacademy.org/math/linear-algebra/matrix_transformations/inverse_transformations/v/linear-algebra-introduction-to-the-inverse-of-a-function)

# Inverses of Linear Transformations

- Some important concepts are formalized in Linear Algebra using Linear Transformations:
  - Bijective function



**Bijective:** It is injective and surjective

**Importance:** Those functions have inverse!

So can we go to another space and get back? Yes!

- See more at:  
[https://pt.khanacademy.org/math/linear-algebra/matrix\\_transformations/inverse\\_transformations/v/linear-algebra-introduction-to-the-inverse-of-a-function](https://pt.khanacademy.org/math/linear-algebra/matrix_transformations/inverse_transformations/v/linear-algebra-introduction-to-the-inverse-of-a-function)

# Inverses of Linear Transformations

- **How relevant is to go to another space** (not necessarily using a linear transformation)?
  - It is common that a transformation is simpler in another space
    - Let's say we will reflect vectors with respect to one of the dimensions
    - Let's say we will transform vectors in another sense
    - Let's say we need another space to build a simpler classifier, i.e., to find the best linear separation hyperplane
- **How relevant is to go another space and get back?**
  - For example: We can conduct some operation in another space, which is easier to work on, and get back simply because our application may need the original space
    - Fourier Transform is an example

# Inverses of Linear Transformations

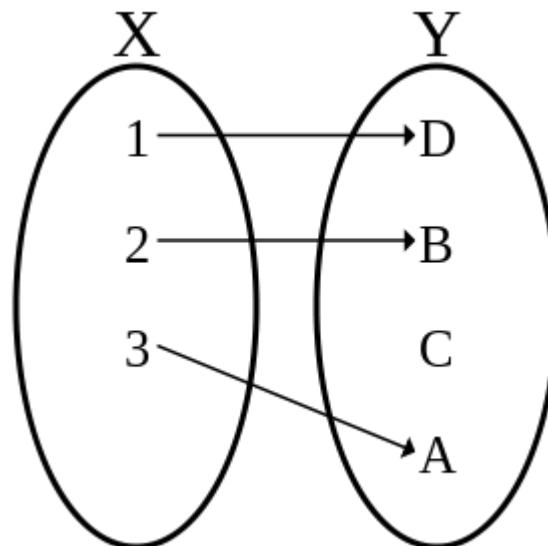
- Consider a linear transformation in form:

$$T : \mathbb{R}^n \rightarrow \mathbb{R}^m$$

- Observe:
  - We can have an inverse of T only if T is injective and surjective (also called bijective when both conditions are satisfied)

# Inverses of Linear Transformations

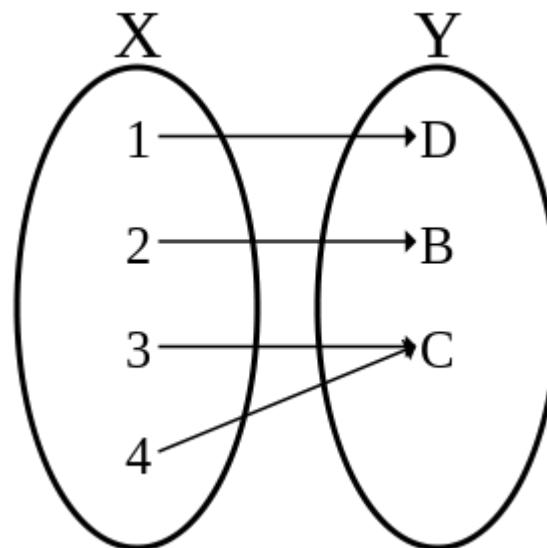
- So, could we have an inverse in this situation (which is only injective)?



- No, because there is one element in the codomain which does not have any corresponding element in the domain

# Inverses of Linear Transformations

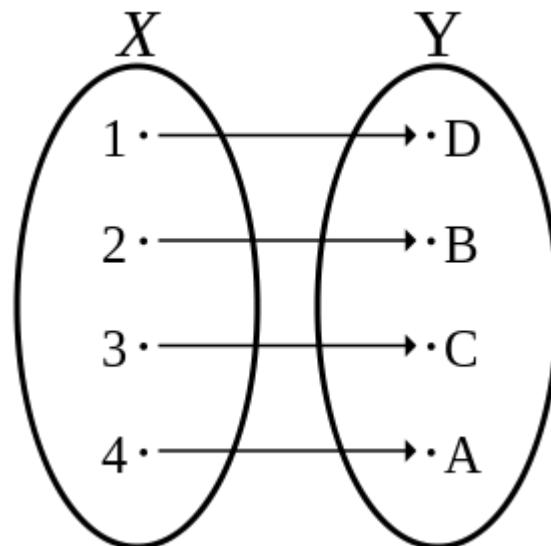
- So, could we have an inverse in this situation (which is only surjective)?



- No, because how could we map the element C in the codomain to two different elements in the domain?

# Inverses of Linear Transformations

- So, we only have inverses when the linear transformation is bijective



# Inverses of Linear Transformations

- Now, we will check the linear transformation:

$$T : \mathbb{R}^n \rightarrow \mathbb{R}^m \text{ for } m > n$$

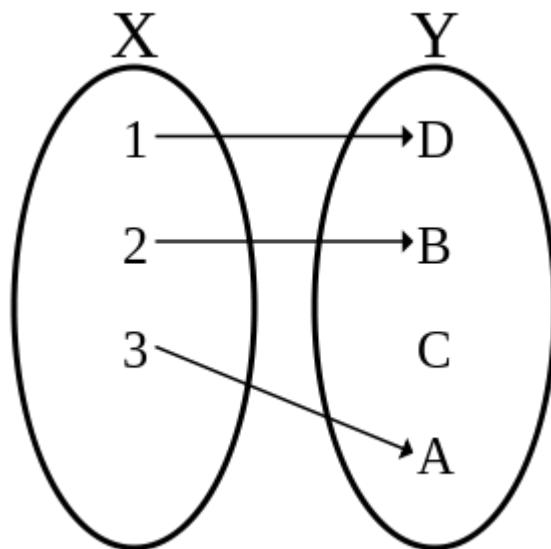
- Can we have an inverse for T?

# Inverses of Linear Transformations

- No way (ok, there are pseudoinverses), because the number of elements in the codomain is greater than the domain

$$T : \mathbb{R}^n \rightarrow \mathbb{R}^m \text{ for } m > n$$

- So, we will have something like:



# Inverses of Linear Transformations

- So we only have an inverse for linear transformations in form:

$$T : \mathbb{R}^n \rightarrow \mathbb{R}^n$$

- What means the domain and codomain have the same cardinality
- Thus, we will only have **inverses for square matrices**
  - The ones that transform from the domain space to the same space

# Inverses of Linear Transformations

- Are there other conditions to have an inverse for?

$$T : \mathbb{R}^n \rightarrow \mathbb{R}^n$$

- Yes:
  - The transformation matrix in the **reduced row echelon form** has to be equal to the identity matrix
    - What means every column vector is **linearly independent** of each other
    - What means the **rank** for the transformation matrix is maximal
    - What means we have a basis for  $\mathbb{R}^n$

# Inverses of Linear Transformations

- For example, consider the linear transformation:

$$T : \mathbb{R}^3 \rightarrow \mathbb{R}^3$$

- Given by:

$$A = \begin{bmatrix} 1 & 2 & 5 \\ 2 & 5 & 7 \\ 3 & 9 & 8 \end{bmatrix}$$

- R commands:

```
require(pracma)
```

```
A = matrix(data=c(1, 2, 3, 2, 5, 9, 5, 7, 8, 0, 0, 0), nrow=3, ncol=4, byrow=F)
```

```
rref(A)
```

# Inverses of Linear Transformations

- For example, consider the linear transformation:

$$T : \mathbb{R}^3 \rightarrow \mathbb{R}^3$$

- Finding the inverse using R commands:

```
require(MASS)
```

```
A = matrix(data=c(1, 2, 3, 2, 5, 9, 5, 7, 8), nrow=3, ncol=3, byrow=F)
```

```
Ai = ginv(A)
```

# Inverses of Linear Transformations

- For example, consider the linear transformation:

$$T : \mathbb{R}^3 \rightarrow \mathbb{R}^3$$

- Now consider an example:

```
require(tseriesChaos)
require(rgl)
data = embedd(lorenz.ts, m=3, d=5)
plot3d(data, t="l")
```

# Inverses of Linear Transformations

- For example, consider the linear transformation:

$$T : \mathbb{R}^3 \rightarrow \mathbb{R}^3$$

- Now we will apply the linear transformation given by matrix A:

```
require(tseriesChaos)
require(rgl)
data = embedd(lorenz.ts, m=3, d=5)
result = NULL
for (i in 1:nrow(data)) {  # or A %*% t(data)
  T_of_A = A %*% data[i,]
  result = rbind(result, t(T_of_A))
}
plot3d(result, t="l")
```

# Inverses of Linear Transformations

- For example, consider the linear transformation:

$$T : \mathbb{R}^3 \rightarrow \mathbb{R}^3$$

- Now we will apply the inverse of this linear transformation:

```
inv = NULL  
for (i in 1:nrow(result)) {  # or Ai %*% t(result)  
    T_inv = Ai %*% result[i,]  
    inv = rbind(inv, t(T_inv))  
}  
plot3d(inv, t="l")
```

# Inverses of Linear Transformations

- For example, consider the linear transformation:

$$T : \mathbb{R}^3 \rightarrow \mathbb{R}^3$$

- What is the difference among the original data and after applying the inverse?

`sum((inv - data)^2)`

- It is just the computer precision error

# Inverses of Linear Transformations

- Are we necessarily worried about having inverses for SVM optimization?
  - No, not necessarily
    - We are much more worried about taking data points to another space where we can proceed with the linear separation of classes
- But there are kernels which have inverses:
  - Example: the PCA kernel

# Inverses of Linear Transformations

- To remember:
  - There is also **another way** to confirm if a squared matrix A is **invertible**:
    - If  $\det(A) \neq 0$ , A is invertible
    - If  $\det(A) = 0$ , the matrix is called **singular** or **degenerate**
    - The set of inverses is dense
      - We will now test with a random square matrix
        - N-by-N random matrices have inverses with probability equals to 1
- See more about inverses of Linear Transformations at:
  - [https://pt.khanacademy.org/math/linear-algebra/matrix\\_transformations/inverse\\_transformations/v/relating-invertibility-to-being-onto-and-one-to-one](https://pt.khanacademy.org/math/linear-algebra/matrix_transformations/inverse_transformations/v/relating-invertibility-to-being-onto-and-one-to-one)
  - [https://pt.khanacademy.org/math/linear-algebra/matrix\\_transformations/inverse\\_transformations/v/linear-algebra-simplifying-conditions-for-invertibility](https://pt.khanacademy.org/math/linear-algebra/matrix_transformations/inverse_transformations/v/linear-algebra-simplifying-conditions-for-invertibility)
  - [https://pt.khanacademy.org/math/linear-algebra/matrix\\_transformations/inverse\\_of\\_matrices/v/linear-algebra-derived-a-method-for-determining-inverses](https://pt.khanacademy.org/math/linear-algebra/matrix_transformations/inverse_of_matrices/v/linear-algebra-derived-a-method-for-determining-inverses)

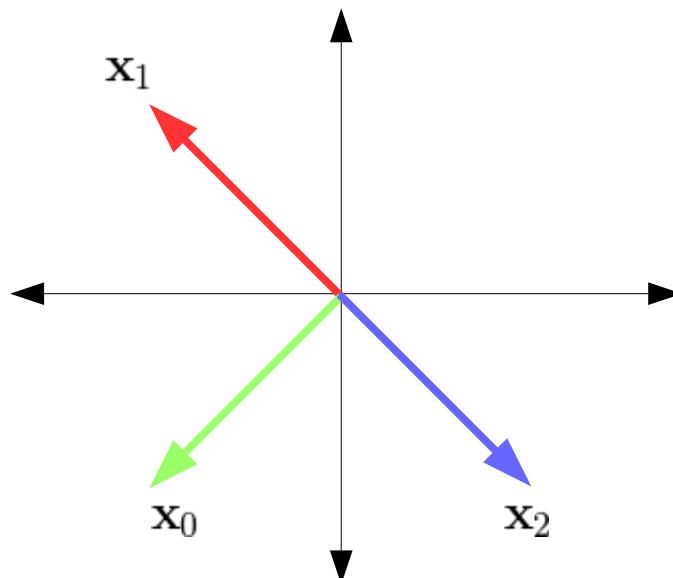
Linear Algebra: Applying Linear Transformations to help understanding kernels

# 1. Image of a Subset under a Linear Transformation

# Applying Linear Transformations

- Consider three vectors:

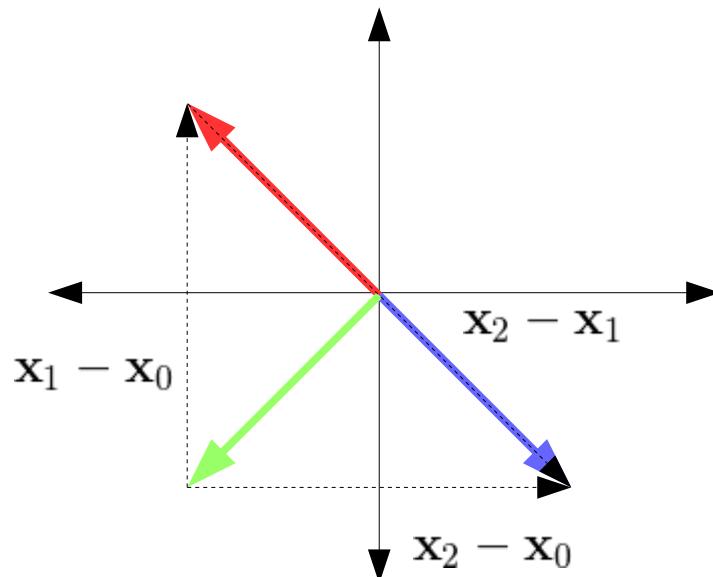
$$\mathbf{x}_0 = \begin{bmatrix} -2 \\ -2 \end{bmatrix}; \mathbf{x}_1 = \begin{bmatrix} -2 \\ 2 \end{bmatrix}; \mathbf{x}_2 = \begin{bmatrix} 2 \\ -2 \end{bmatrix}$$



# Applying Linear Transformations

- Now consider we connect the line segments to form a triangle:

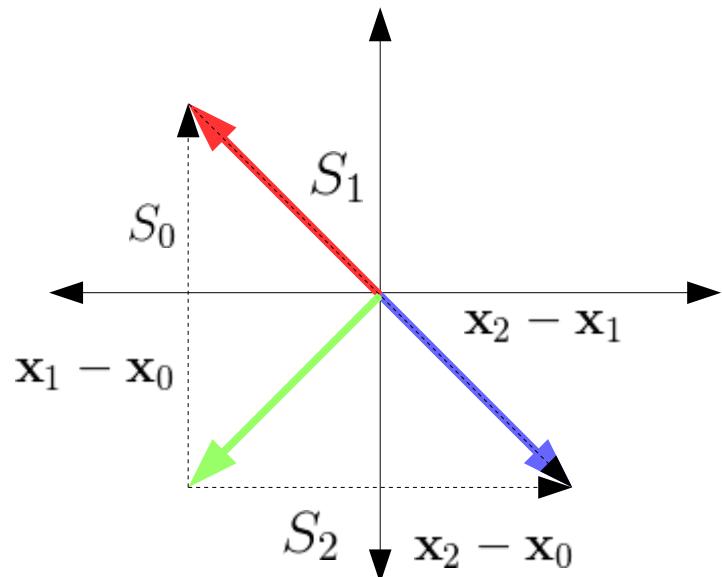
$$\mathbf{x}_0 = \begin{bmatrix} -2 \\ -2 \end{bmatrix}; \mathbf{x}_1 = \begin{bmatrix} -2 \\ 2 \end{bmatrix}; \mathbf{x}_2 = \begin{bmatrix} 2 \\ -2 \end{bmatrix}$$



# Applying Linear Transformations

- Describing the line segments:

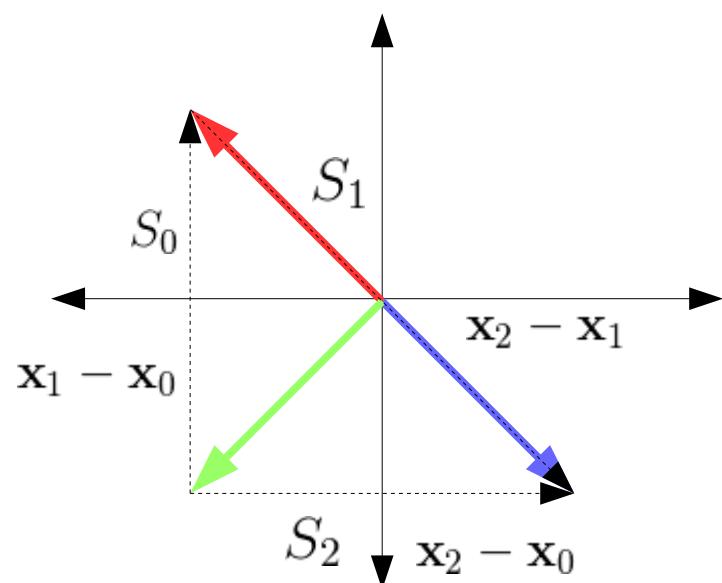
$$\mathbf{x}_0 = \begin{bmatrix} -2 \\ -2 \end{bmatrix}; \mathbf{x}_1 = \begin{bmatrix} -2 \\ 2 \end{bmatrix}; \mathbf{x}_2 = \begin{bmatrix} 2 \\ -2 \end{bmatrix}$$



$$S_0 = \left\{ \mathbf{x}_0 + t(\mathbf{x}_1 - \mathbf{x}_0) \mid 0 \leq t \leq 1 \right\}$$
$$S_1 = \left\{ \mathbf{x}_1 + t(\mathbf{x}_2 - \mathbf{x}_1) \mid 0 \leq t \leq 1 \right\}$$
$$S_2 = \left\{ \mathbf{x}_0 + t(\mathbf{x}_2 - \mathbf{x}_0) \mid 0 \leq t \leq 1 \right\}$$

# Applying Linear Transformations

- We now define the set called Shape which forms a triangle by connecting those position vectors

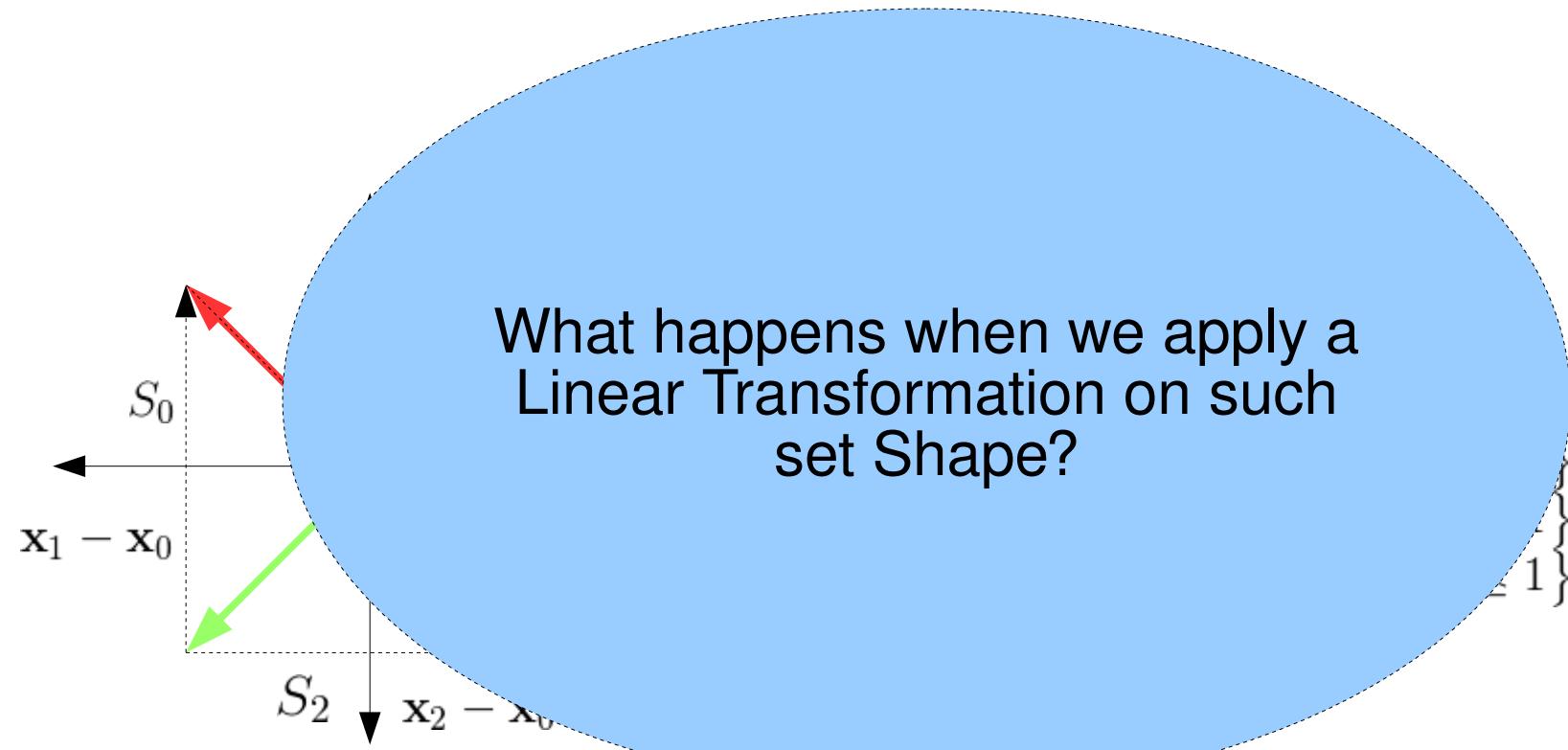


$$\text{Shape} = \{S_0, S_1, S_2\}$$

$$S_0 = \{\mathbf{x}_0 + t(\mathbf{x}_1 - \mathbf{x}_0) \mid 0 \leq t \leq 1\}$$
$$S_1 = \{\mathbf{x}_1 + t(\mathbf{x}_2 - \mathbf{x}_1) \mid 0 \leq t \leq 1\}$$
$$S_2 = \{\mathbf{x}_0 + t(\mathbf{x}_2 - \mathbf{x}_0) \mid 0 \leq t \leq 1\}$$

# Applying Linear Transformations

- We now define the set called Shape which forms a triangle by connecting those position vectors



# Applying Linear Transformations

- So, consider the linear transformation given by matrix:

$$A = \begin{bmatrix} 1 & -1 \\ 2 & 0 \end{bmatrix}$$

- In form:

$$T(\mathbf{x}) = A\mathbf{x} = \begin{bmatrix} 1 & -1 \\ 2 & 0 \end{bmatrix} \mathbf{x}$$

- How can we apply it on Shape?

# Applying Linear Transformations

- We start with the transformation on  $S_0$ :

$$T(S_0) = \{ T(\mathbf{x}_0 + t(\mathbf{x}_1 - \mathbf{x}_0)) \mid 0 \leq t \leq 1 \}$$

# Applying Linear Transformations

- We start with the transformation on  $S_0$ :

$$\begin{aligned}T(S_0) &= \left\{ T(\mathbf{x}_0 + t(\mathbf{x}_1 - \mathbf{x}_0)) \mid 0 \leq t \leq 1 \right\} \\&= \left\{ T(\mathbf{x}_0) + T(t(\mathbf{x}_1 - \mathbf{x}_0)) \mid 0 \leq t \leq 1 \right\} \\&= \left\{ T(\mathbf{x}_0) + t \cdot T(\mathbf{x}_1 - \mathbf{x}_0) \mid 0 \leq t \leq 1 \right\} \\&= \left\{ T(\mathbf{x}_0) + t \cdot [T(\mathbf{x}_1) - T(\mathbf{x}_0)] \mid 0 \leq t \leq 1 \right\}\end{aligned}$$

- So, the transformation on  $S_0$  depends on the transformations of  $\mathbf{x}_0$ ,  $\mathbf{x}_1$  and  $\mathbf{x}_2$

# Applying Linear Transformations

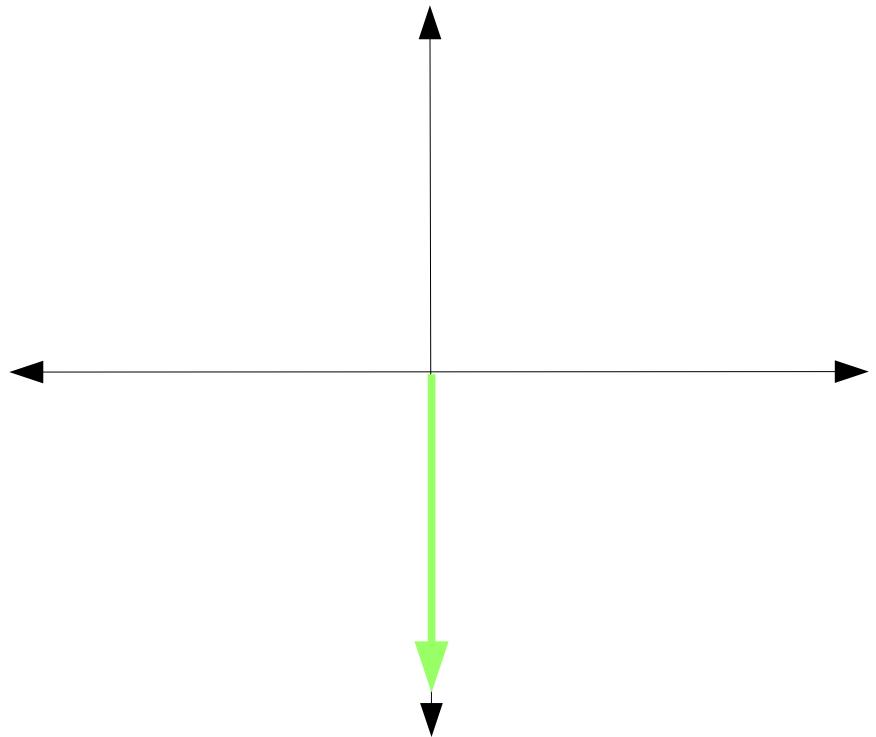
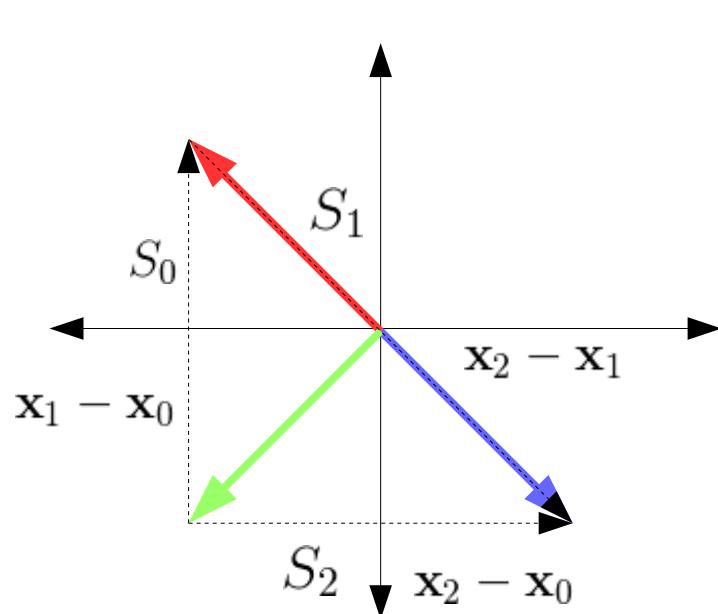
- We now compute the transformation on  $\mathbf{x}_0$ :

$$T(\mathbf{x}_0) = \begin{bmatrix} 1 & -1 \\ 2 & 0 \end{bmatrix} \begin{bmatrix} -2 \\ -2 \end{bmatrix} = \begin{bmatrix} 0 \\ -4 \end{bmatrix}$$

# Applying Linear Transformations

- We now compute the transformation on  $\mathbf{x}_0$ :

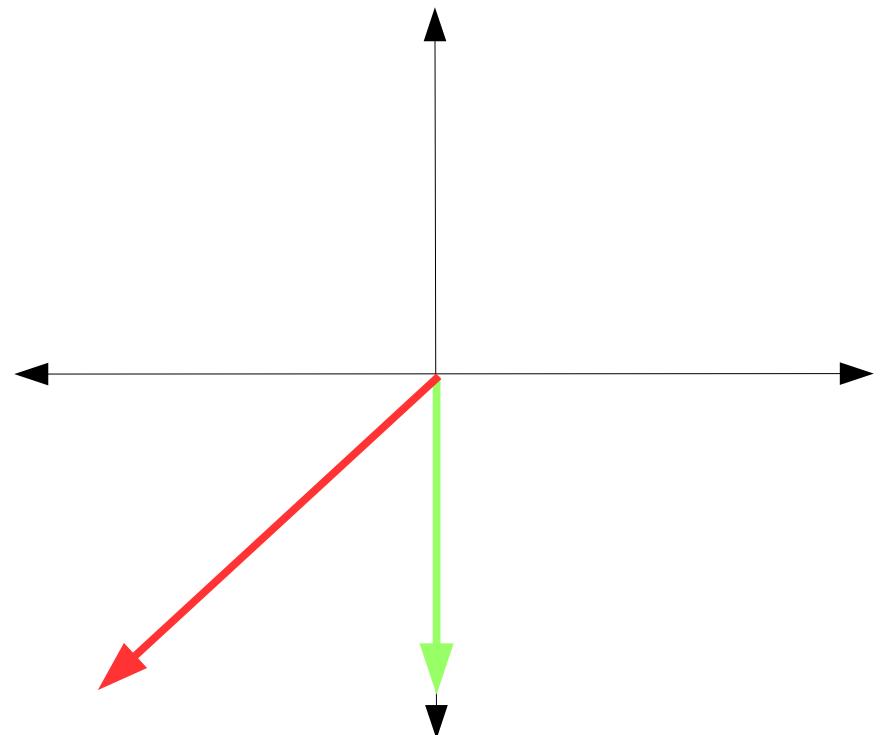
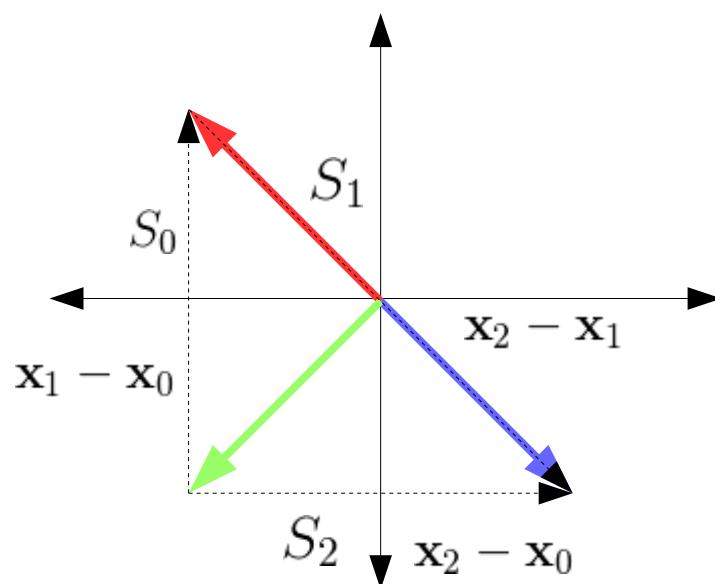
$$T(\mathbf{x}_0) = \begin{bmatrix} 1 & -1 \\ 2 & 0 \end{bmatrix} \begin{bmatrix} -2 \\ -2 \end{bmatrix} = \begin{bmatrix} 0 \\ -4 \end{bmatrix}$$



# Applying Linear Transformations

- We now compute the transformation on  $\mathbf{x}_1$ :

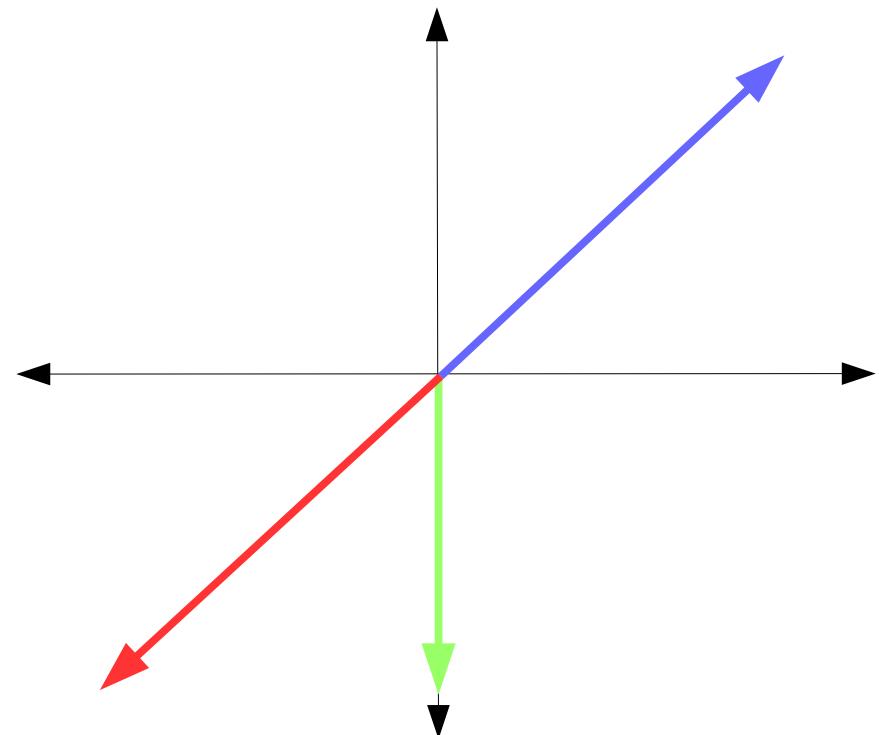
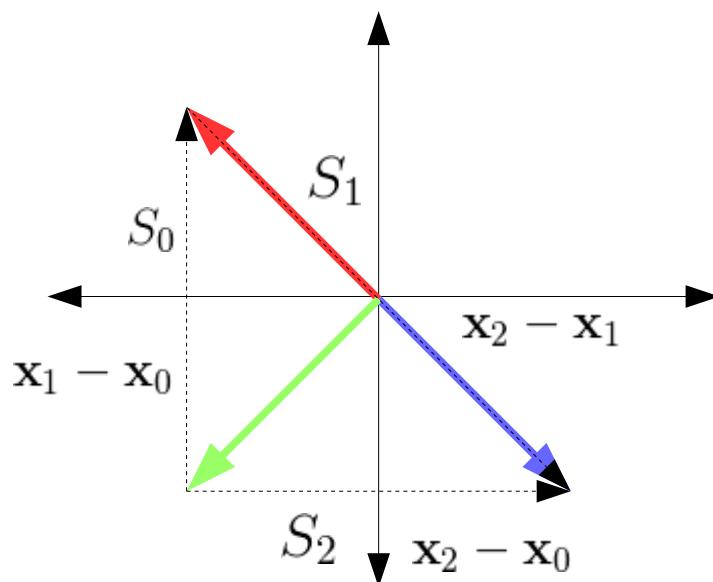
$$T(\mathbf{x}_1) = \begin{bmatrix} 1 & -1 \\ 2 & 0 \end{bmatrix} \begin{bmatrix} -2 \\ 2 \end{bmatrix} = \begin{bmatrix} -4 \\ -4 \end{bmatrix}$$



# Applying Linear Transformations

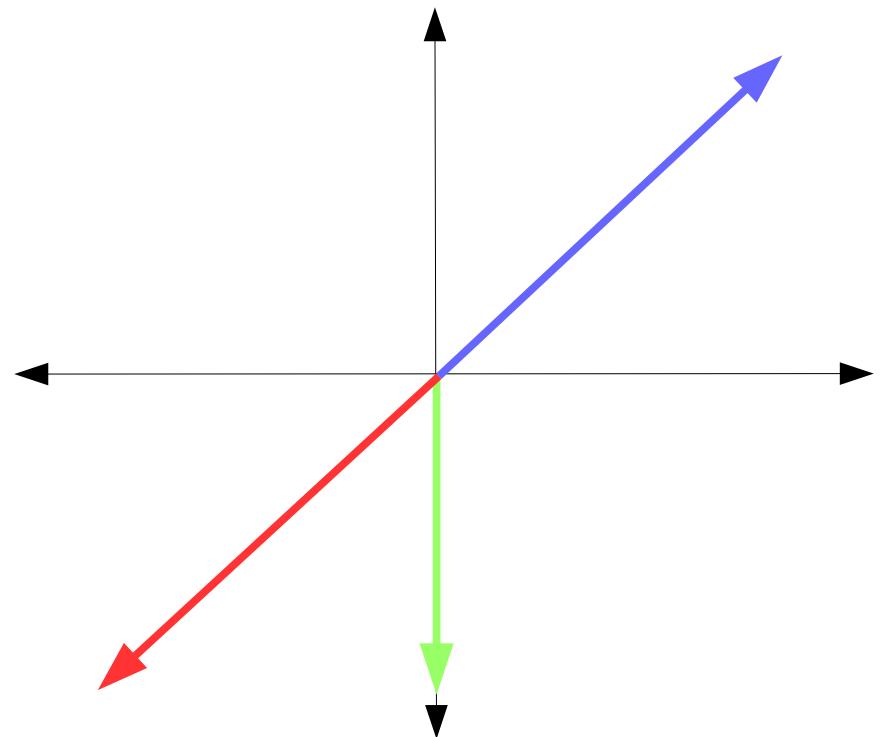
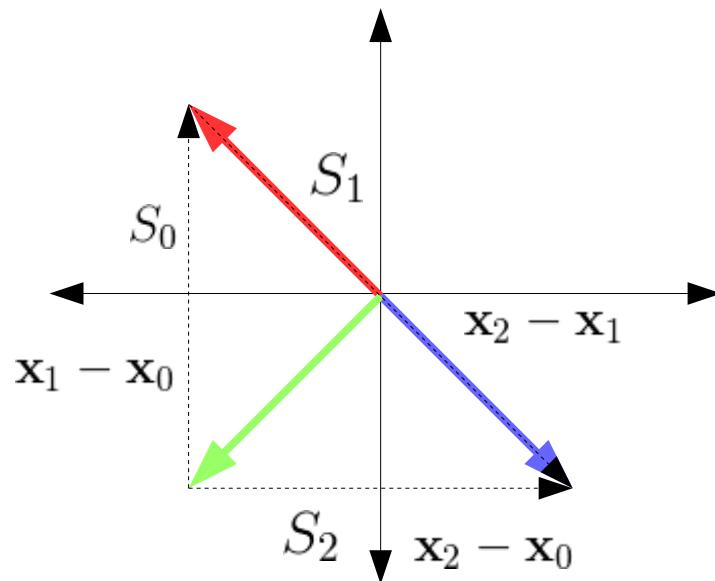
- We now compute the transformation on  $\mathbf{x}_2$ :

$$T(\mathbf{x}_2) = \begin{bmatrix} 1 & -1 \\ 2 & 0 \end{bmatrix} \begin{bmatrix} 2 \\ -2 \end{bmatrix} = \begin{bmatrix} 4 \\ 4 \end{bmatrix}$$



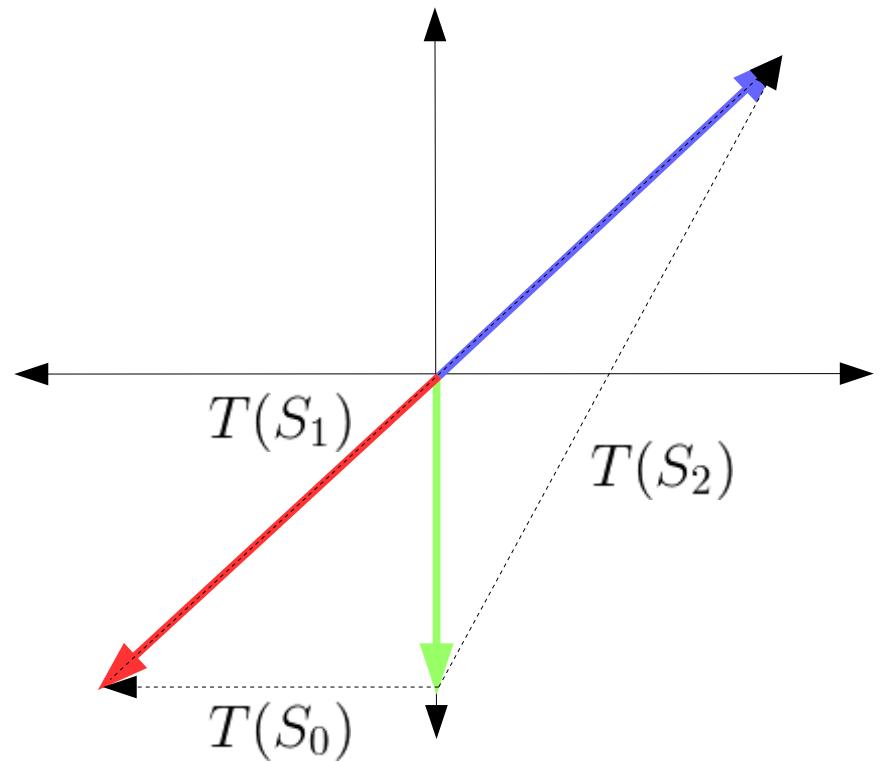
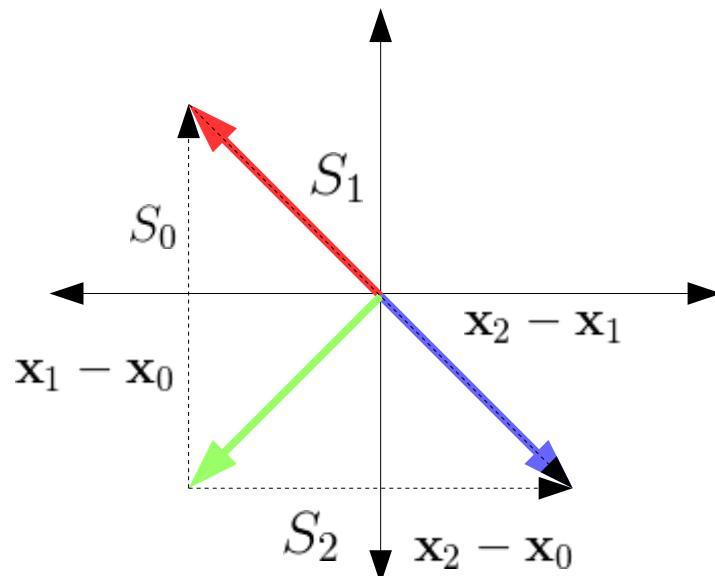
# Applying Linear Transformations

- The other line segments also depend on vectors  $\mathbf{x}_0$ ,  $\mathbf{x}_1$  and  $\mathbf{x}_2$ 
  - What happened to the set Shape?



# Applying Linear Transformations

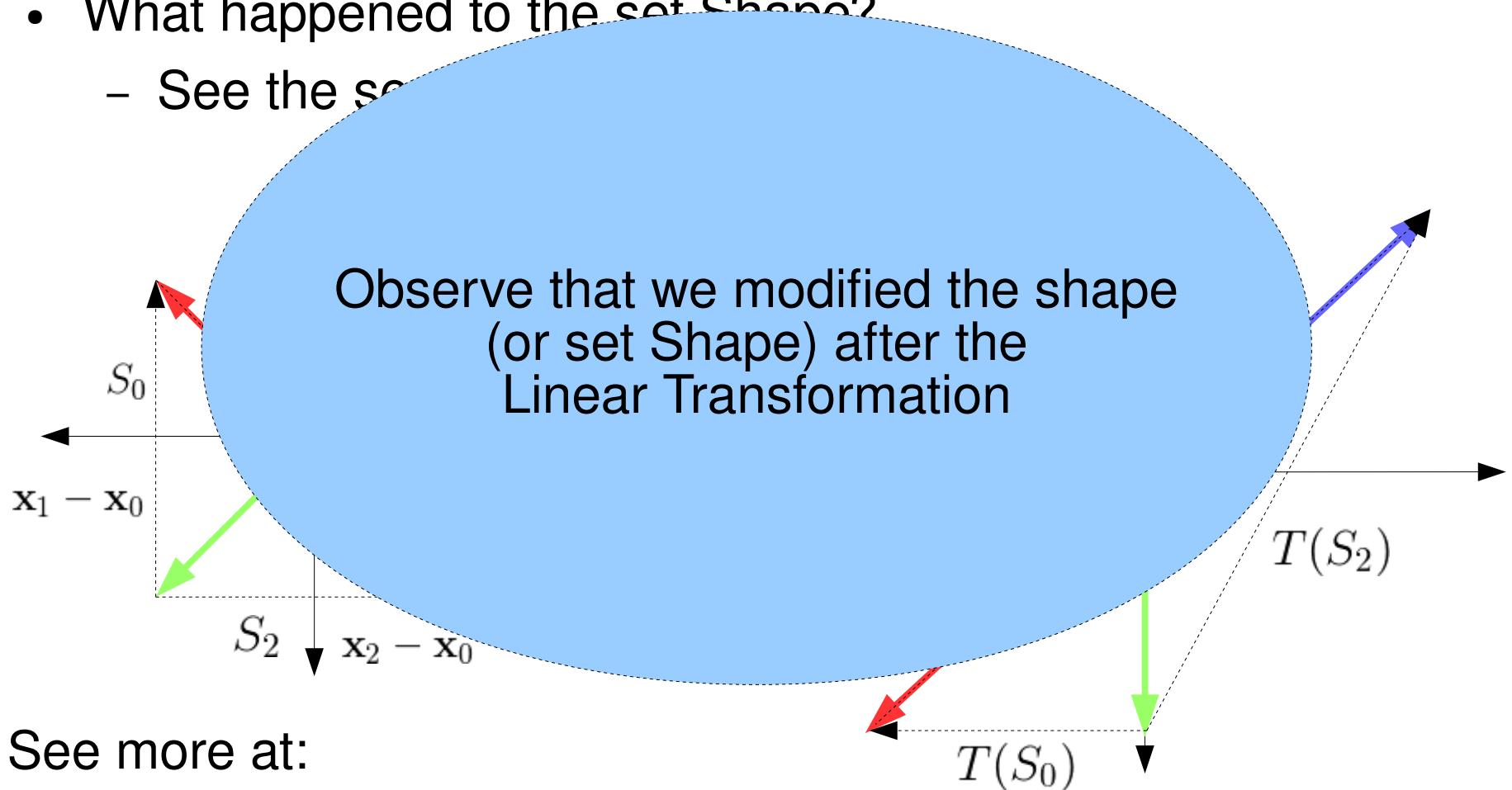
- The other line segments also depend on vectors  $\mathbf{x}_0$ ,  $\mathbf{x}_1$  and  $\mathbf{x}_2$ 
  - What happened to the set Shape?
    - See the segments after transformation



- See more at:
  - [https://pt.khanacademy.org/math/linear-algebra/matrix\\_transformations/linear\\_transformations/v/image-of-a-subset-under-a-transformation](https://pt.khanacademy.org/math/linear-algebra/matrix_transformations/linear_transformations/v/image-of-a-subset-under-a-transformation)

# Applying Linear Transformations

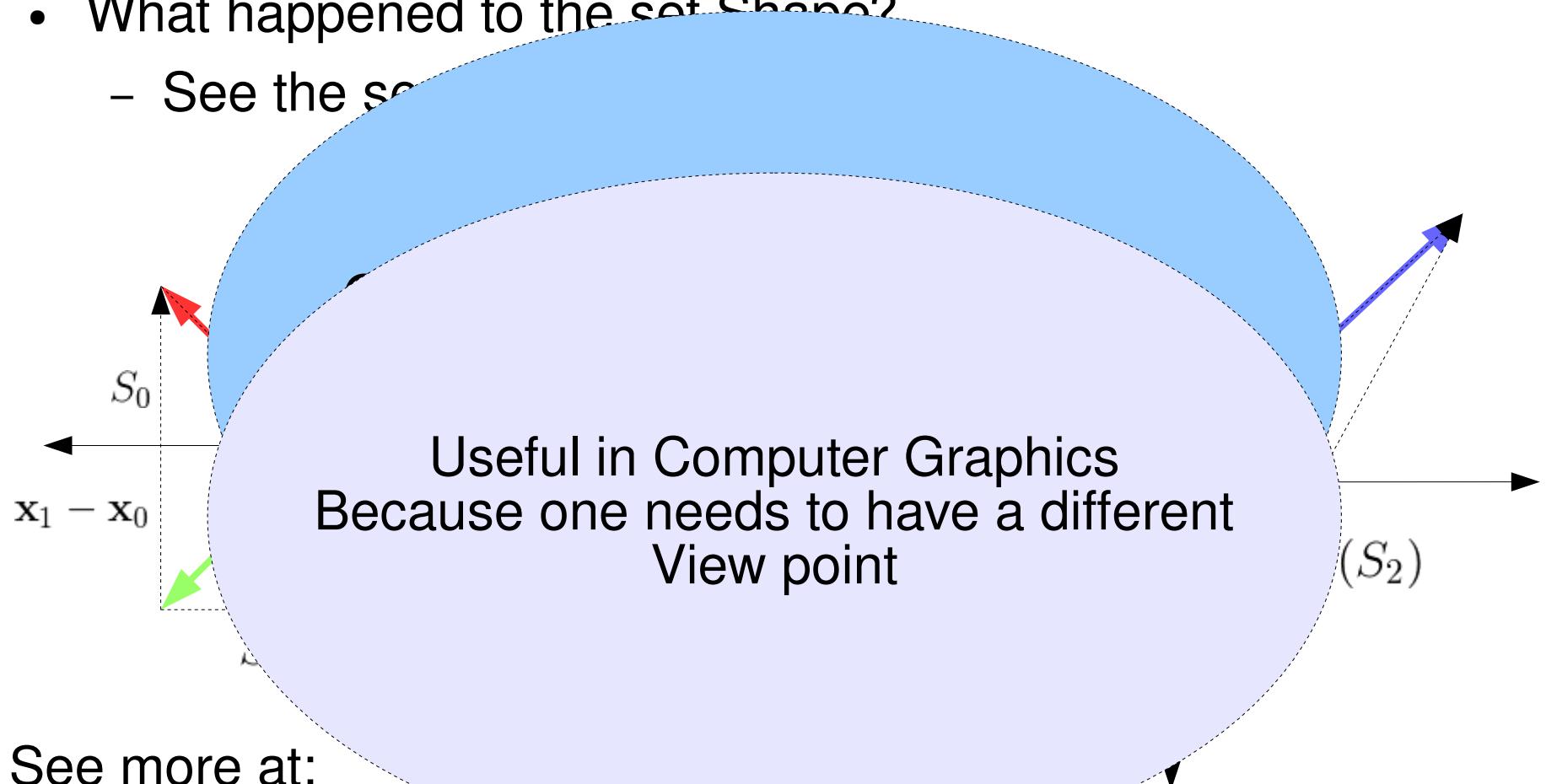
- The other line segments also depend on vectors  $\mathbf{x}_0$ ,  $\mathbf{x}_1$  and  $\mathbf{x}_2$ 
  - What happened to the set Shape?
    - See the set  $S_2$



- See more at:
  - [https://pt.khanacademy.org/math/linear-algebra/matrix\\_transformations/linear\\_tra](https://pt.khanacademy.org/math/linear-algebra/matrix-transformations/linear-transformations/v/image-of-a-subset-under-a-transformation)  
nsformations/v/image-of-a-subset-under-a-transformation

# Applying Linear Transformations

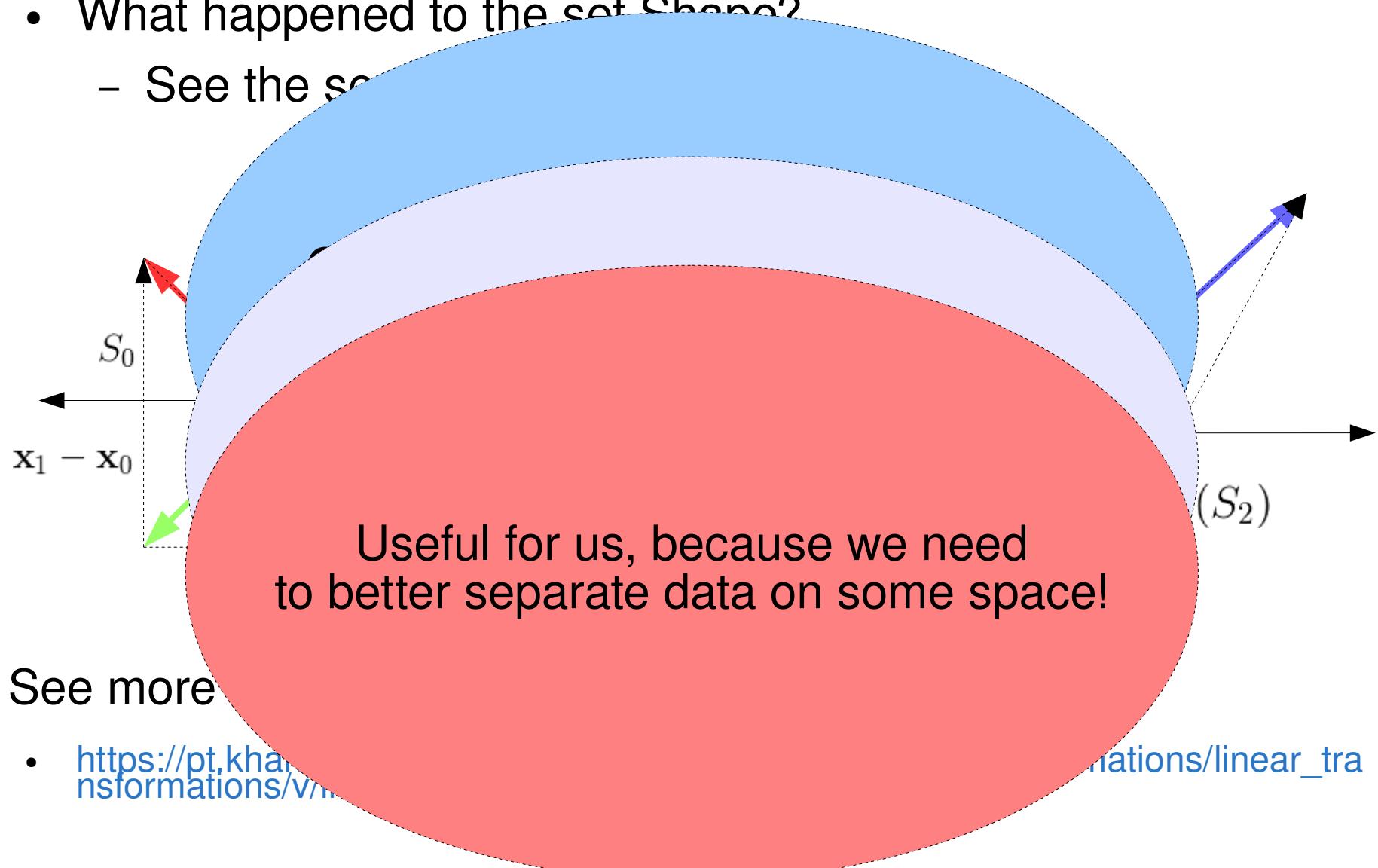
- The other line segments also depend on vectors  $\mathbf{x}_0$ ,  $\mathbf{x}_1$  and  $\mathbf{x}_2$ 
  - What happened to the set  $S_0$ ?
    - See the set  $(S_2)$



- See more at:
  - <https://pt.khanacademy.org/math/matrix-transformations/linear-transformations/v/image-of-a-subset-under-a-transformation>

# Applying Linear Transformations

- The other line segments also depend on vectors  $\mathbf{x}_0$ ,  $\mathbf{x}_1$  and  $\mathbf{x}_2$ 
  - What happened to the set  $S_0$ ?
    - See the set  $(S_2)$

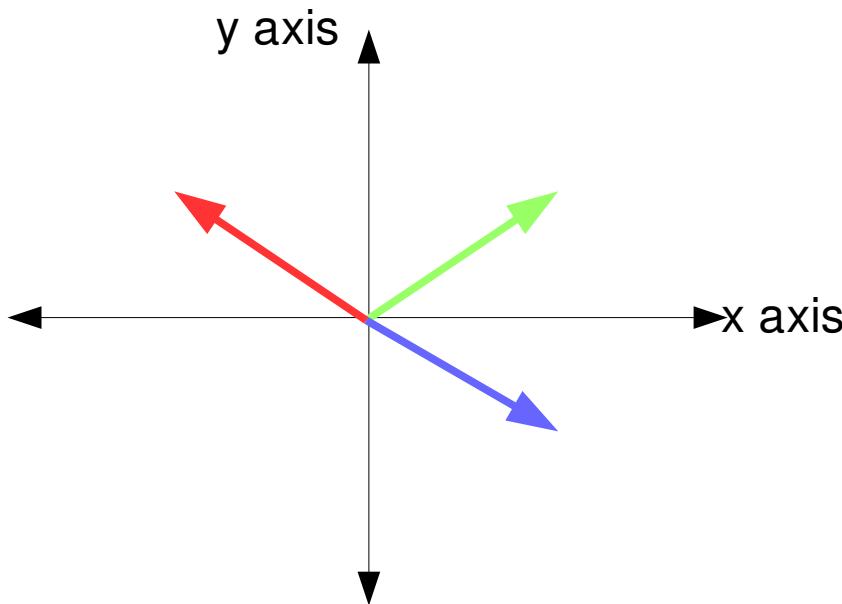


## 2. Scaling and Reflections using Linear Transformations

# Applying Linear Transformations

- Consider we have three vectors:

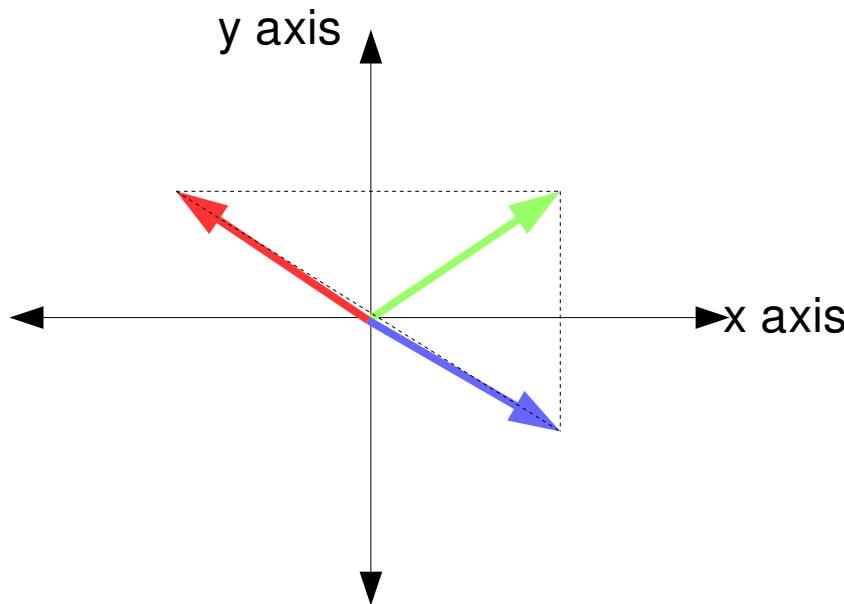
$$\mathbf{x}_0 = \begin{bmatrix} 3 \\ 2 \end{bmatrix}; \quad \mathbf{x}_1 = \begin{bmatrix} -3 \\ 2 \end{bmatrix}; \quad \mathbf{x}_2 = \begin{bmatrix} 3 \\ -2 \end{bmatrix}$$



# Applying Linear Transformations

- Now consider we have line segments forming a triangular shape as follows:

$$\mathbf{x}_0 = \begin{bmatrix} 3 \\ 2 \end{bmatrix}; \mathbf{x}_1 = \begin{bmatrix} -3 \\ 2 \end{bmatrix}; \mathbf{x}_2 = \begin{bmatrix} 3 \\ -2 \end{bmatrix}$$



# Applying Linear Transformations

- Now consider we have line segments forming a triangular shape as follows:

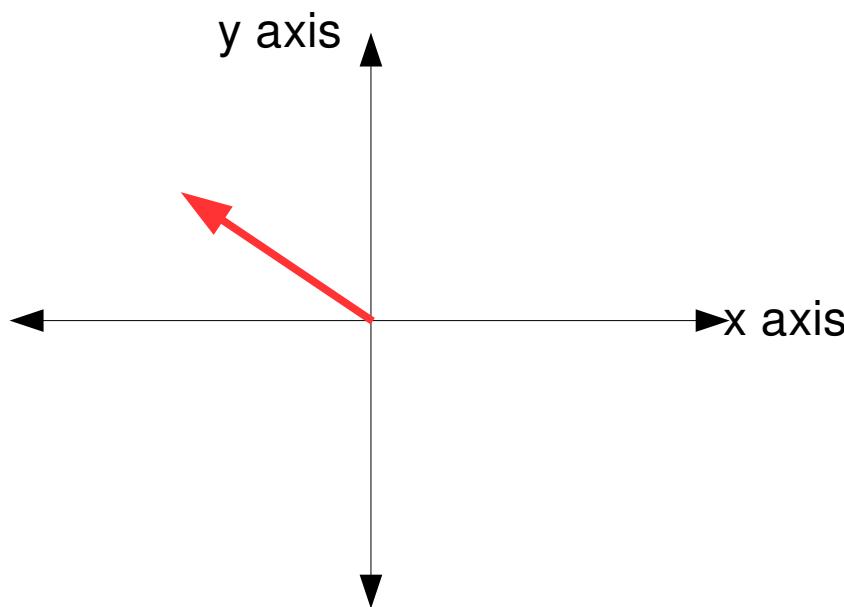
$$\mathbf{x}_0 = \begin{bmatrix} 3 \\ 2 \end{bmatrix}; \quad \mathbf{x}_1 = \begin{bmatrix} -3 \\ 0 \end{bmatrix}; \quad \mathbf{x}_2 = \begin{bmatrix} 3 \\ -2 \end{bmatrix}$$

Suppose we want to:

- 1) Reflect this shape in relation to y
- 2) “Stretch” twice in the y direction

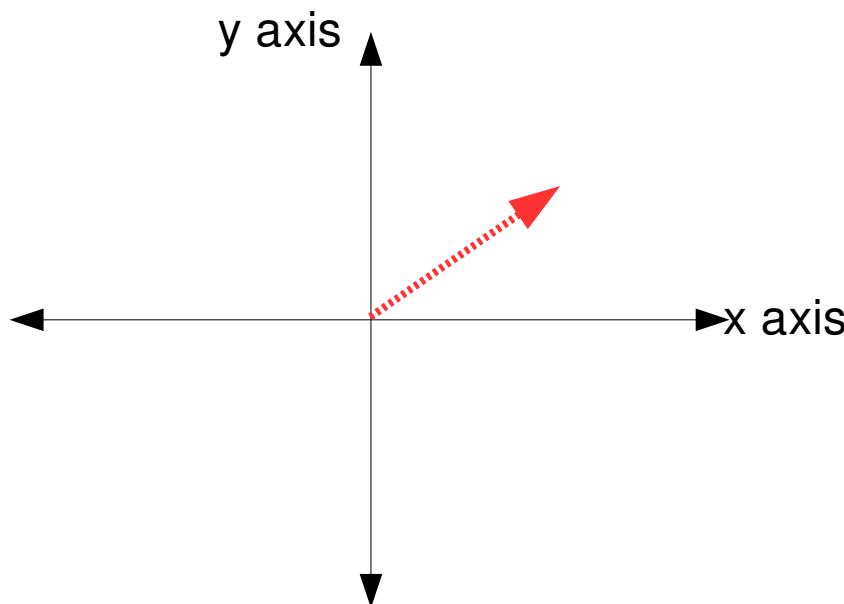
# Applying Linear Transformations

- Firstly, what happens to vector  $x_1$ , when we reflect it in relation to y axis?



# Applying Linear Transformations

- Firstly, what happens to vector  $\mathbf{x}_1$ , when we reflect it in relation to y axis?
  - It would go to the other side of the y axis
  - Observe the y value does not change, only the x value
    - We change the sign of x



# Applying Linear Transformations

- Firstly, what happens to vector  $x_1$ , when we reflect it in relation to y axis?
  - It would go to the other side of the y axis
  - Observe the vector  $x_1$ .
    - We change only the x value

The same happens to the other vectors

Only the sign of x changes



# Applying Linear Transformations

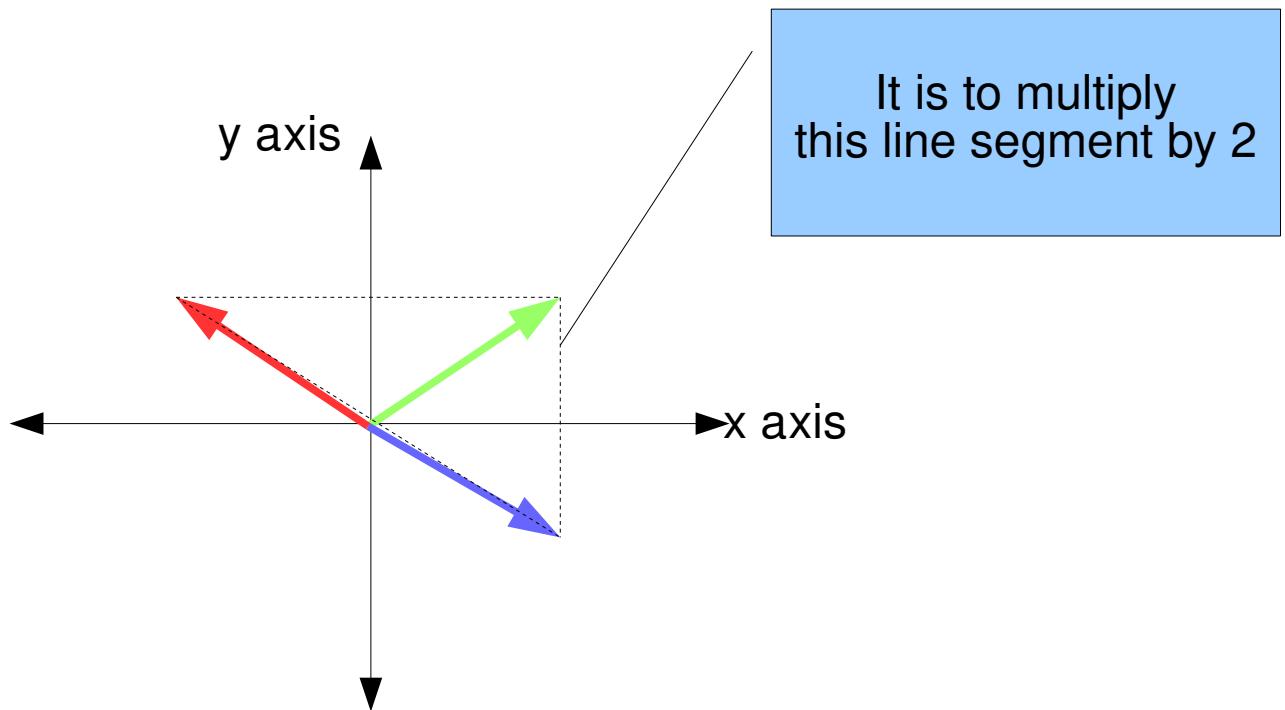
- Firstly, what happens to vector  $x_1$ , when we reflect it in relation to y axis?
  - It would go to the other side of the y axis
  - Observe the x value
    - We change the sign of the x value only

The same happens to all other vectors

So, we multiply the x value by -1  
to reflect it in terms of y

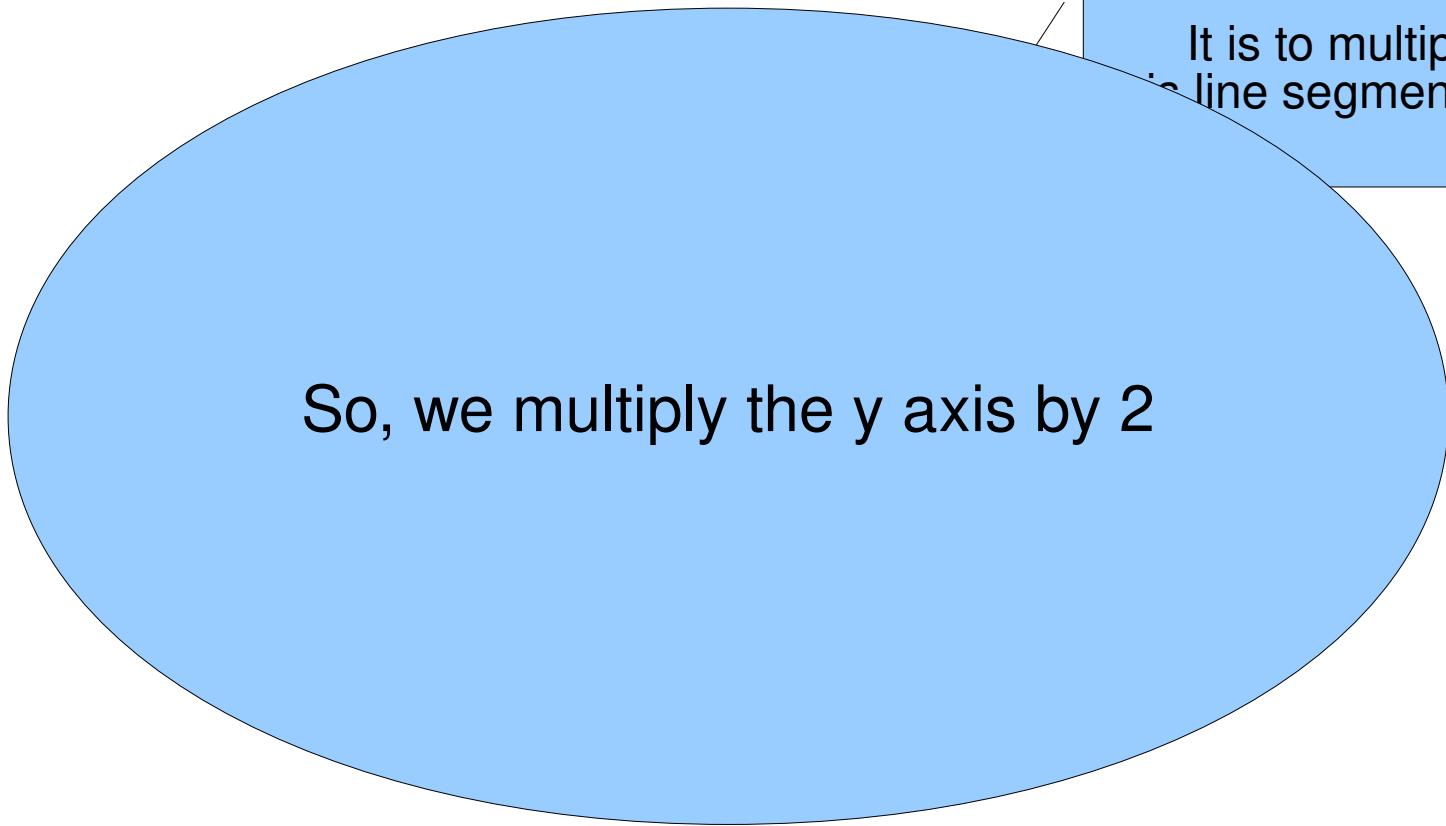
# Applying Linear Transformations

- Now, what is to “stretch” twice in terms of the y axis?



# Applying Linear Transformations

- Now, what is to “stretch” twice in terms of the y axis?



It is to multiply  
its line segment by 2

So, we multiply the y axis by 2

# Applying Linear Transformations

- How can we write our transformation?

$$T \begin{pmatrix} [x] \\ [y] \end{pmatrix} = \begin{bmatrix} -x \\ 2y \end{bmatrix}$$

- But how to write this transformation in terms of a matrix A?

# Applying Linear Transformations

- How can we write our transformation?

$$T \begin{pmatrix} [x] \\ [y] \end{pmatrix} = \begin{bmatrix} -x \\ 2y \end{bmatrix}$$

- But how to write this transformation in terms of a matrix A?
  - We do it by using the identity matrix

# Applying Linear Transformations

- Consider matrix A as the identity matrix:

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

- We can apply the transformation on every column of A to obtain the matrix we desire:

$$T(A) = [T(e_1) \ T(e_2)] = \begin{bmatrix} -1 \cdot 1 & -1 \cdot 0 \\ 2 \cdot 0 & 2 \cdot 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 2 \end{bmatrix}$$

# Applying Linear Transformations

- So, to transform any vector  $\mathbf{x}$  we can apply:

$$\mathbf{x}_0 = \begin{bmatrix} 3 \\ 2 \end{bmatrix}; \quad \mathbf{x}_1 = \begin{bmatrix} -3 \\ 2 \end{bmatrix}; \quad \mathbf{x}_2 = \begin{bmatrix} 3 \\ -2 \end{bmatrix}$$

$$T\left(\begin{bmatrix} 3 \\ 2 \end{bmatrix}\right) = A \begin{bmatrix} 3 \\ 2 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 3 \\ 2 \end{bmatrix} = \begin{bmatrix} -3 \\ 4 \end{bmatrix}$$

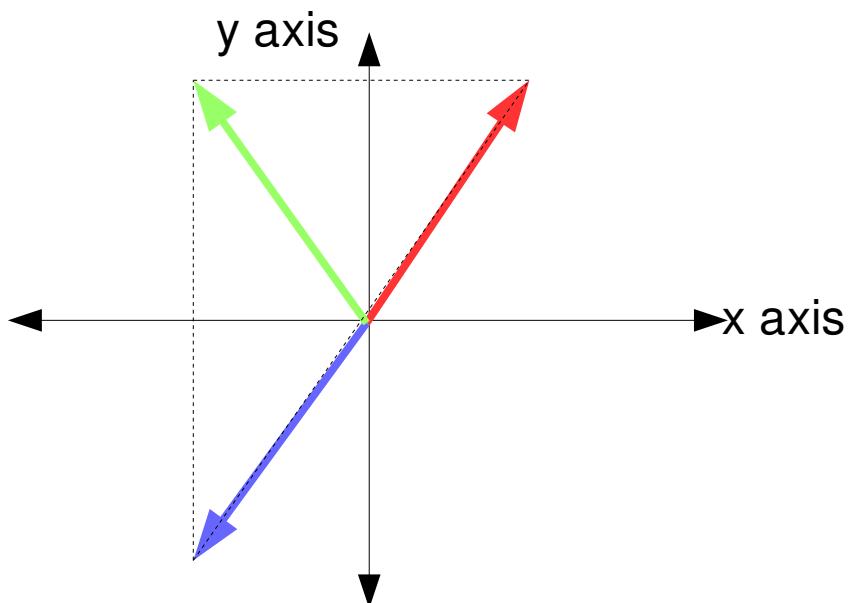
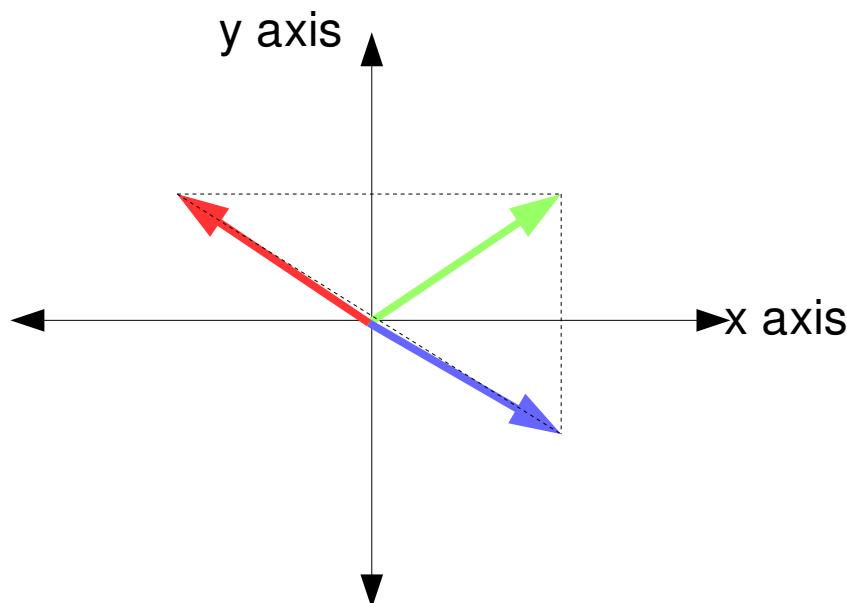
$$T\left(\begin{bmatrix} -3 \\ 2 \end{bmatrix}\right) = A \begin{bmatrix} -3 \\ 2 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} -3 \\ 2 \end{bmatrix} = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$$

$$T\left(\begin{bmatrix} 3 \\ -2 \end{bmatrix}\right) = A \begin{bmatrix} 3 \\ -2 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 2 \end{bmatrix} \begin{bmatrix} 3 \\ -2 \end{bmatrix} = \begin{bmatrix} -3 \\ -4 \end{bmatrix}$$

# Applying Linear Transformations

- So, we will have:

$$\begin{bmatrix} -3 \\ 4 \end{bmatrix}; \begin{bmatrix} 3 \\ 4 \end{bmatrix}; \begin{bmatrix} -3 \\ -4 \end{bmatrix}$$



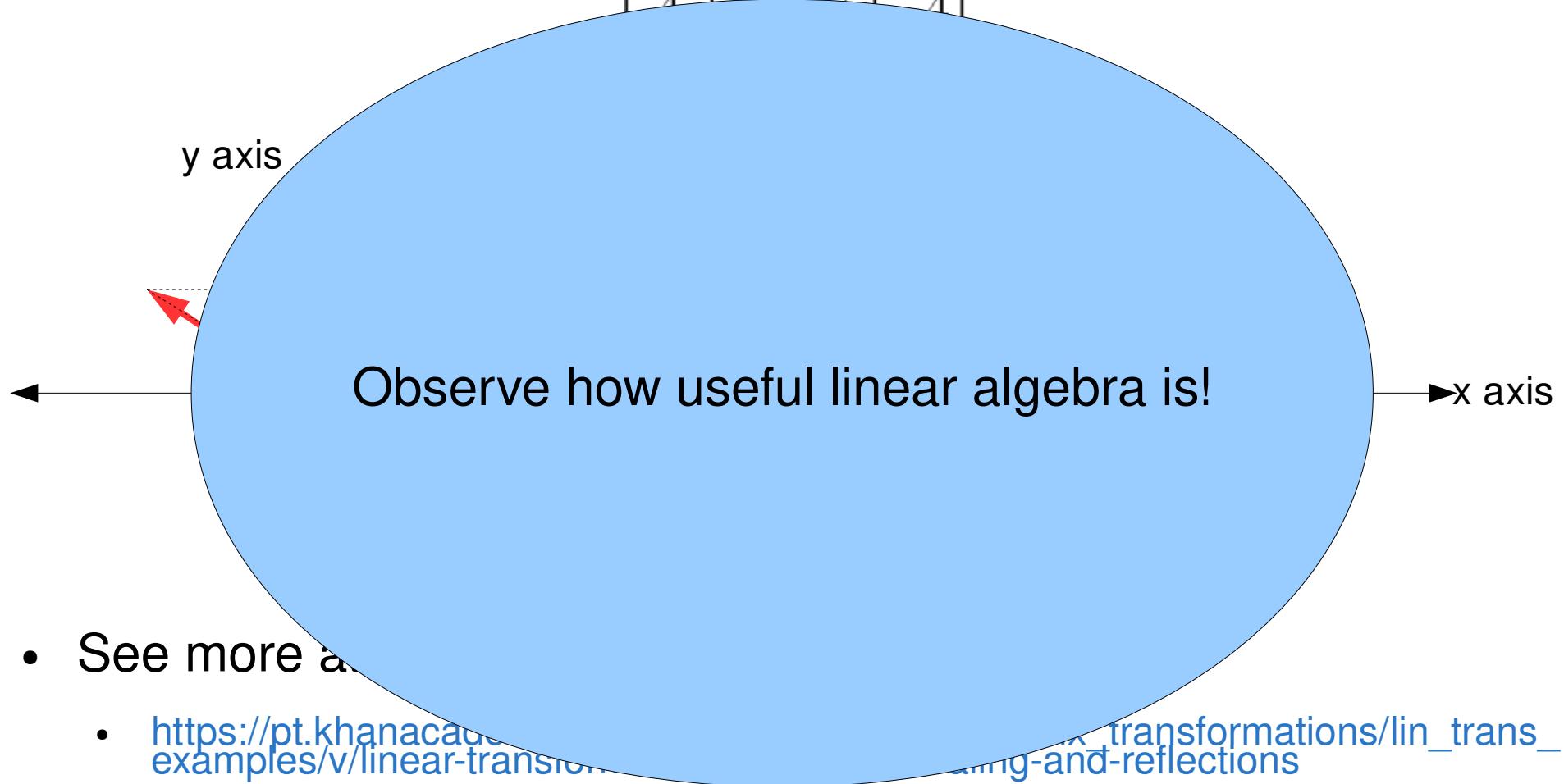
- See more at:

- [https://pt.khanacademy.org/math/linear-algebra/matrix\\_transformations/lin\\_trans\\_examples/v/linear-transformation-examples-scaling-and-reflections](https://pt.khanacademy.org/math/linear-algebra/matrix_transformations/lin_trans_examples/v/linear-transformation-examples-scaling-and-reflections)

# Applying Linear Transformations

- So, we will have:

$$\begin{bmatrix} -3 \\ 4 \end{bmatrix}; \begin{bmatrix} 3 \\ 4 \end{bmatrix}; \begin{bmatrix} -3 \\ -4 \end{bmatrix}$$



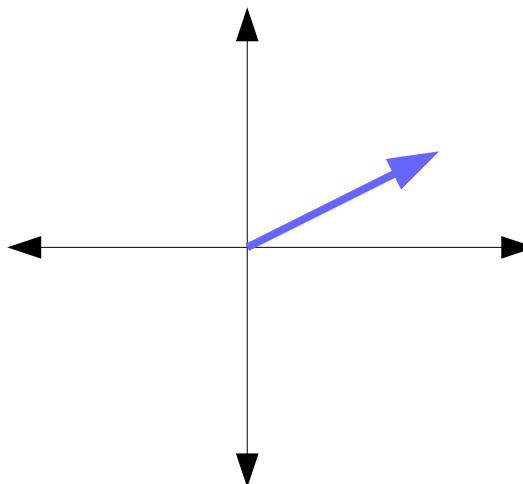
### 3. Rotations in $\mathbb{R}^2$

# Applying Linear Transformations

- Consider we have a vector and want to rotate it some degrees

$$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$$

- For example:

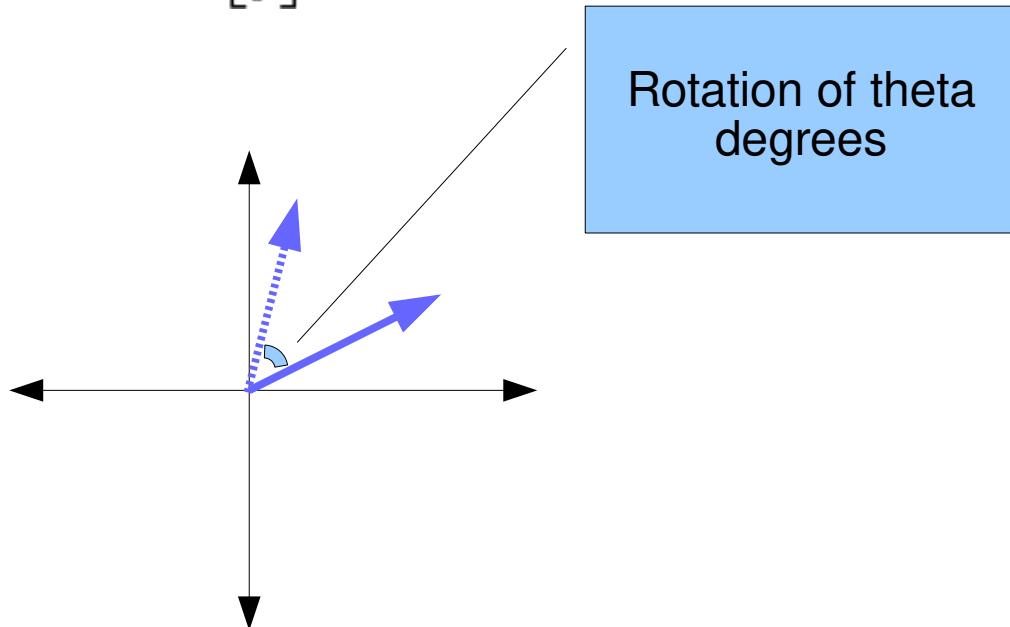


# Applying Linear Transformations

- Consider we have a vector and want to rotate it some degrees

$$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$$

- For example:



# Applying Linear Transformations

- Consider we have a vector and want to rotate it some degrees

$$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix}$$

- So we will build a matrix  $A$  to apply this linear transformation on such vector:

$$\text{Rot}_\theta(\mathbf{x}) : \mathbb{R}^2 \rightarrow \mathbb{R}^2$$

$$\text{Rot}_\theta(\mathbf{x}) = A\mathbf{x}$$

# Applying Linear Transformations

- We start building A using the identity matrix:

$$\begin{aligned}\text{Rot}_\theta(\mathbf{x}) &: \mathbb{R}^2 \rightarrow \mathbb{R}^2 \\ \text{Rot}_\theta(\mathbf{x}) &= A\mathbf{x}\end{aligned}$$

- Then we apply the rotation on every column vector to obtain the transformation matrix:

$$\begin{aligned}A &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \\ \text{Rot}_\theta(A) &= \left[ \text{Rot}_\theta \left( \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right) \quad \text{Rot}_\theta \left( \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right) \right]\end{aligned}$$

# Applying Linear Transformations

- We start building A using the identity matrix:

$$\text{Rot}_\theta(\mathbf{x}) : \mathbb{R}^2 \rightarrow \mathbb{R}^2$$

$$\text{Rot}_{-\theta}(\mathbf{x})$$

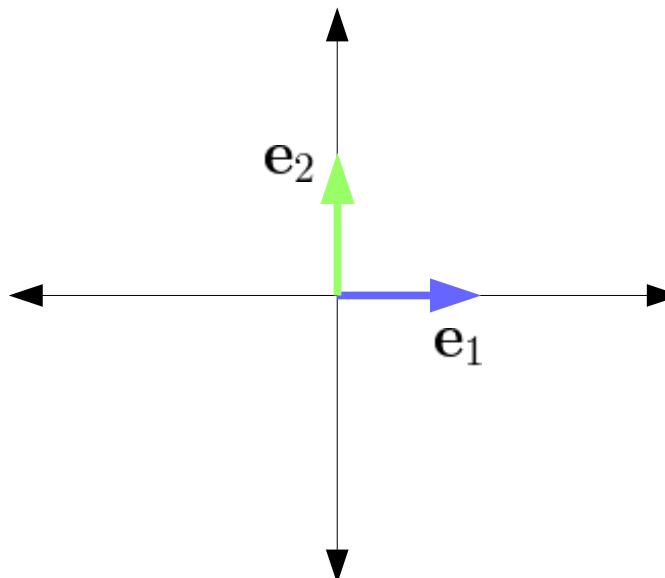
- Then we apply the transformation  $\text{Rot}_{-\theta}$  to obtain the final transformation.

But how to build such rotation?

# Applying Linear Transformations

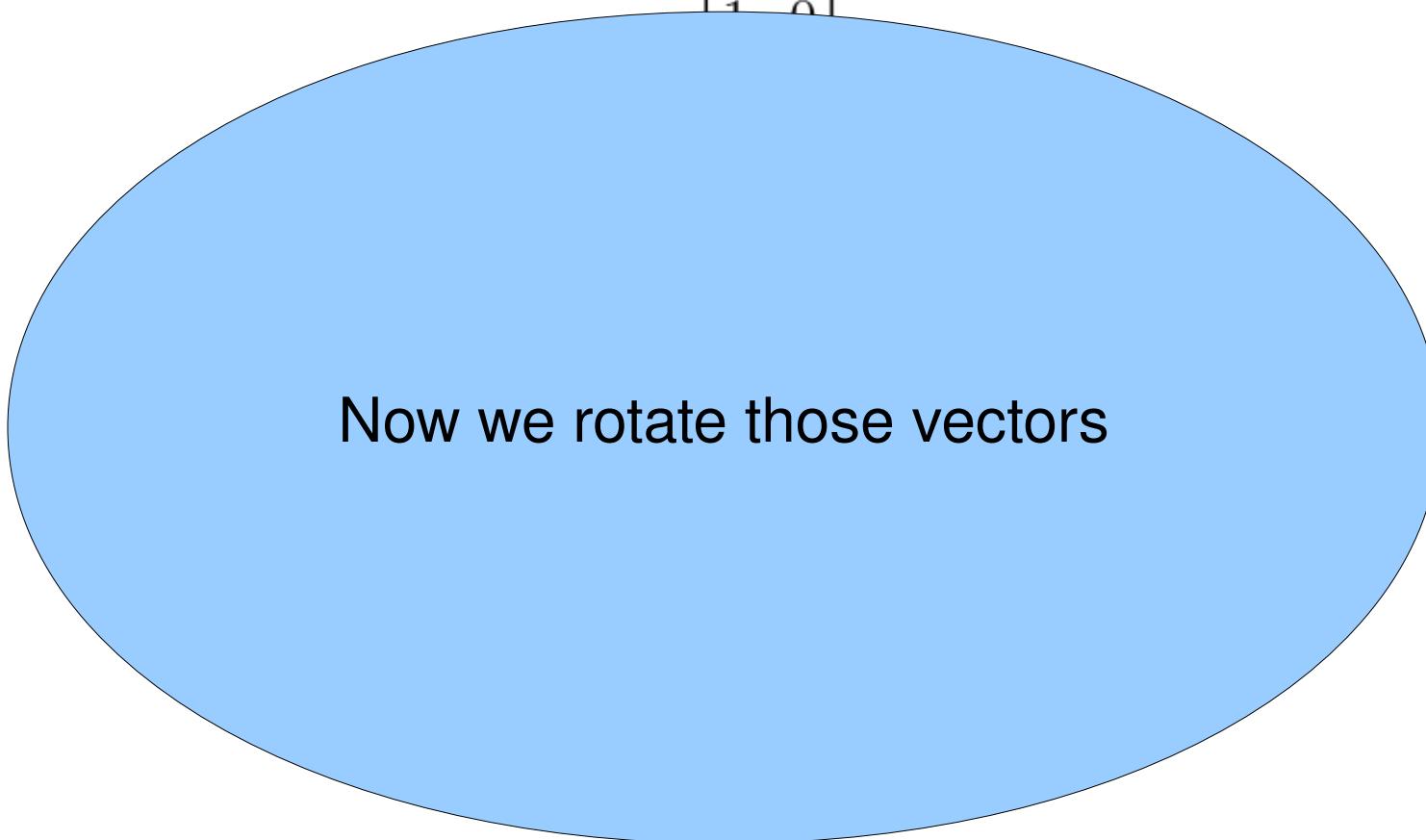
- We start representing both column vectors ( $e_1$  and  $e_2$ ) of the identity matrix:

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$



# Applying Linear Transformations

- We start representing both column vectors ( $e_1$  and  $e_2$ ) of the identity matrix:

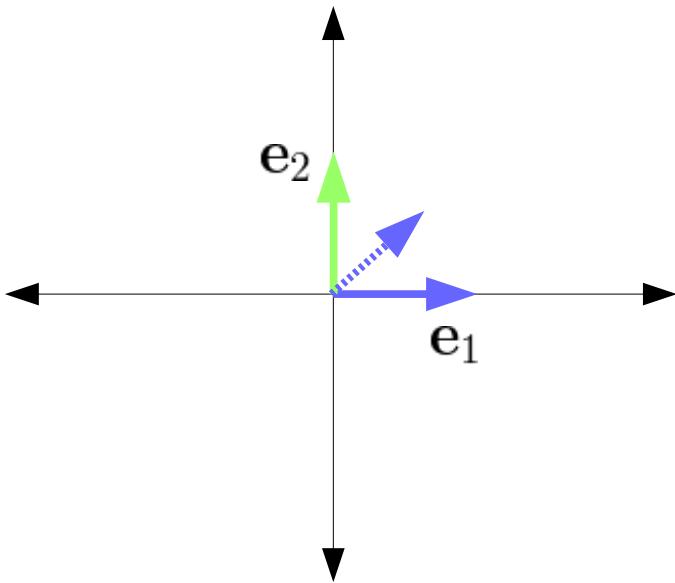


Now we rotate those vectors

$$[e_1 \ e_2]$$

# Applying Linear Transformations

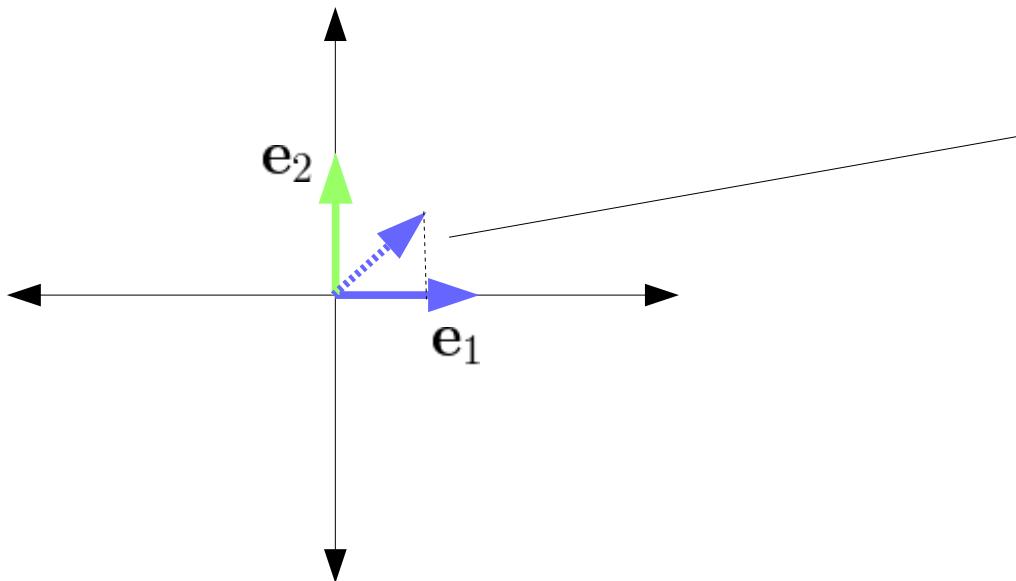
- Let's try a rotation over  $e_1$ :



We now use some trigonometry  
Observe the triangle formed

# Applying Linear Transformations

- Let's try a rotation over  $e_1$ :

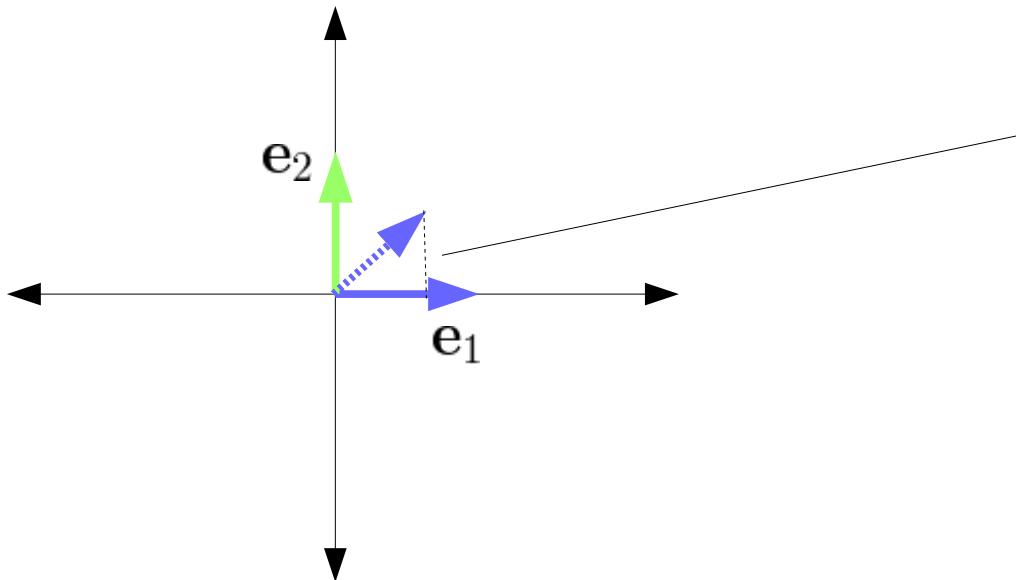


This is the triangle!

Now remember the vector norm  
is the same because we just  
rotate it

# Applying Linear Transformations

- Let's try a rotation over  $e_1$ :

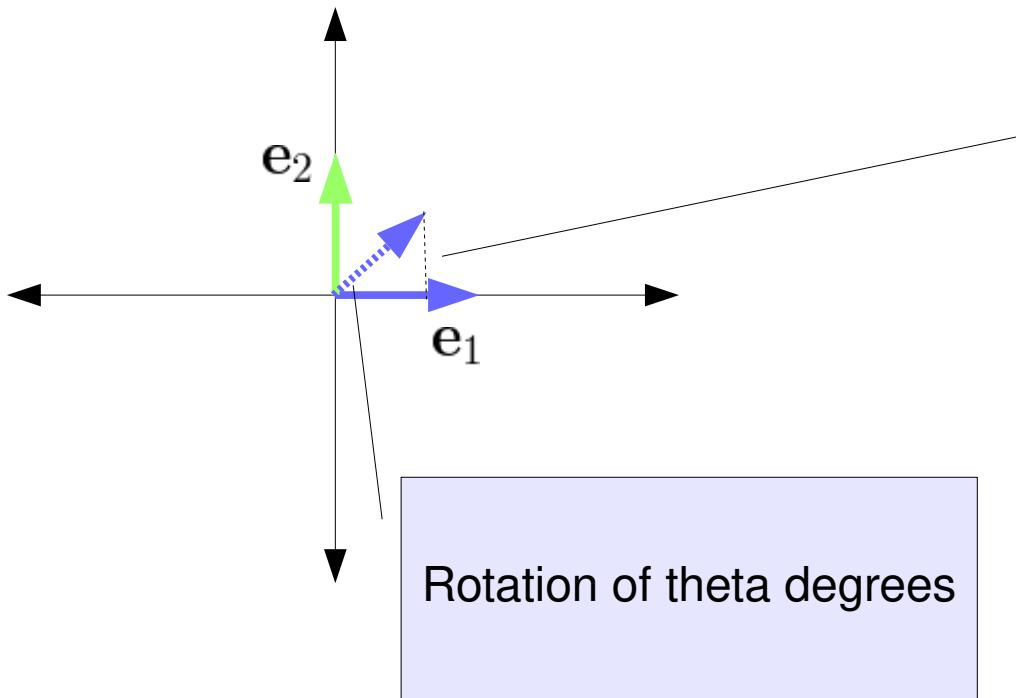


Now observe the values in x and y axis have changed!

What is the value for the x-axis?

# Applying Linear Transformations

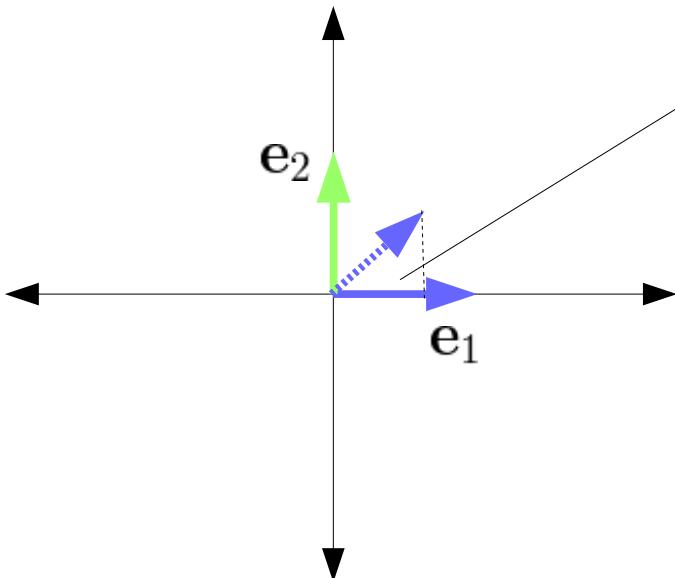
- Let's try a rotation over  $e_1$ :



What is the value for the x-axis?  
We need to find the adjacent side  
to the angle of this rotation!

# Applying Linear Transformations

- Let's try a rotation over  $e_1$ :



What is the value for the x-axis?

We need to find the adjacent side to the angle of this rotation!

Recall the adjacent side can be found as follows:

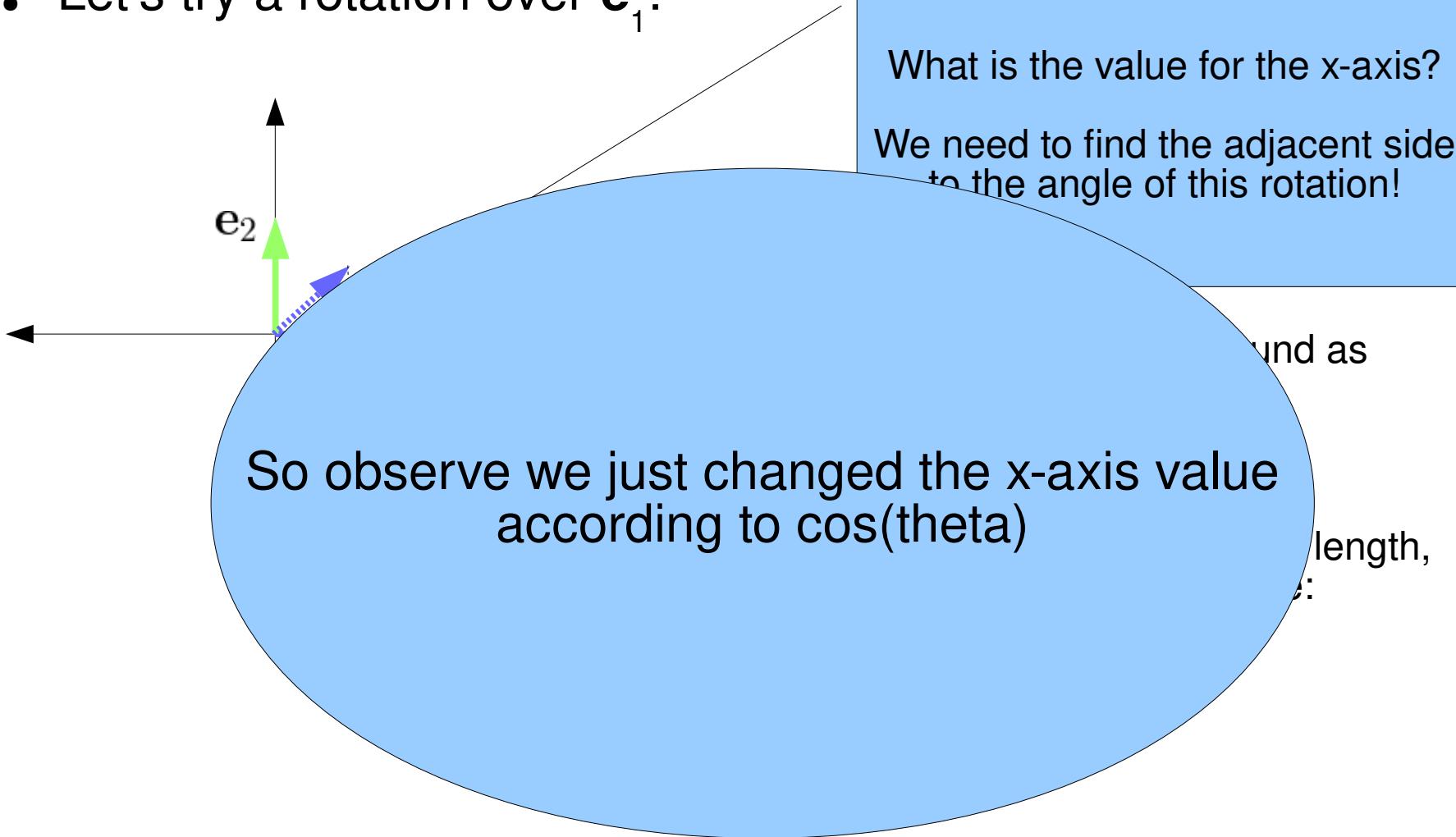
$$\cos(\theta) = \frac{\text{adj}}{\text{hyp}}$$

As the hypotenuse did not change its length, so it is still equals to one, so we have:

$$\cos(\theta) = \frac{\text{adj}}{1}$$

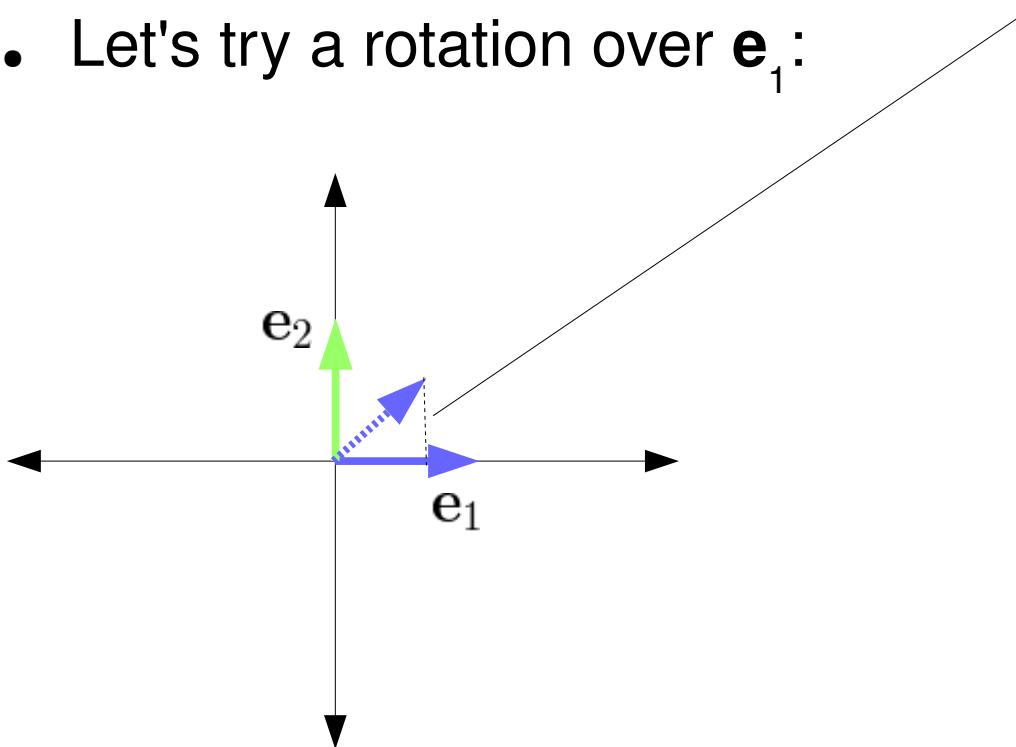
# Applying Linear Transformations

- Let's try a rotation over  $e_1$ :



# Applying Linear Transformations

- Let's try a rotation over  $e_1$ :



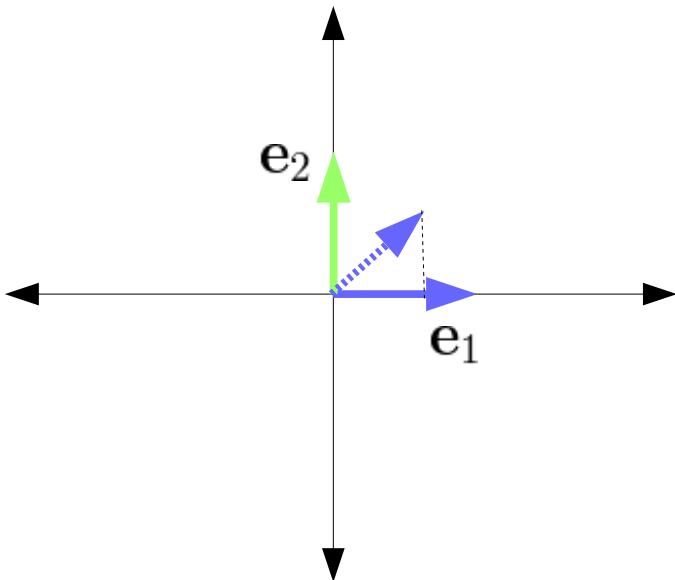
What about y-axis?

Observe the length of this  
dashed-line segment

It corresponds to the  $\sin(\theta)$

# Applying Linear Transformations

- Let's try a rotation over  $\mathbf{e}_1$ :

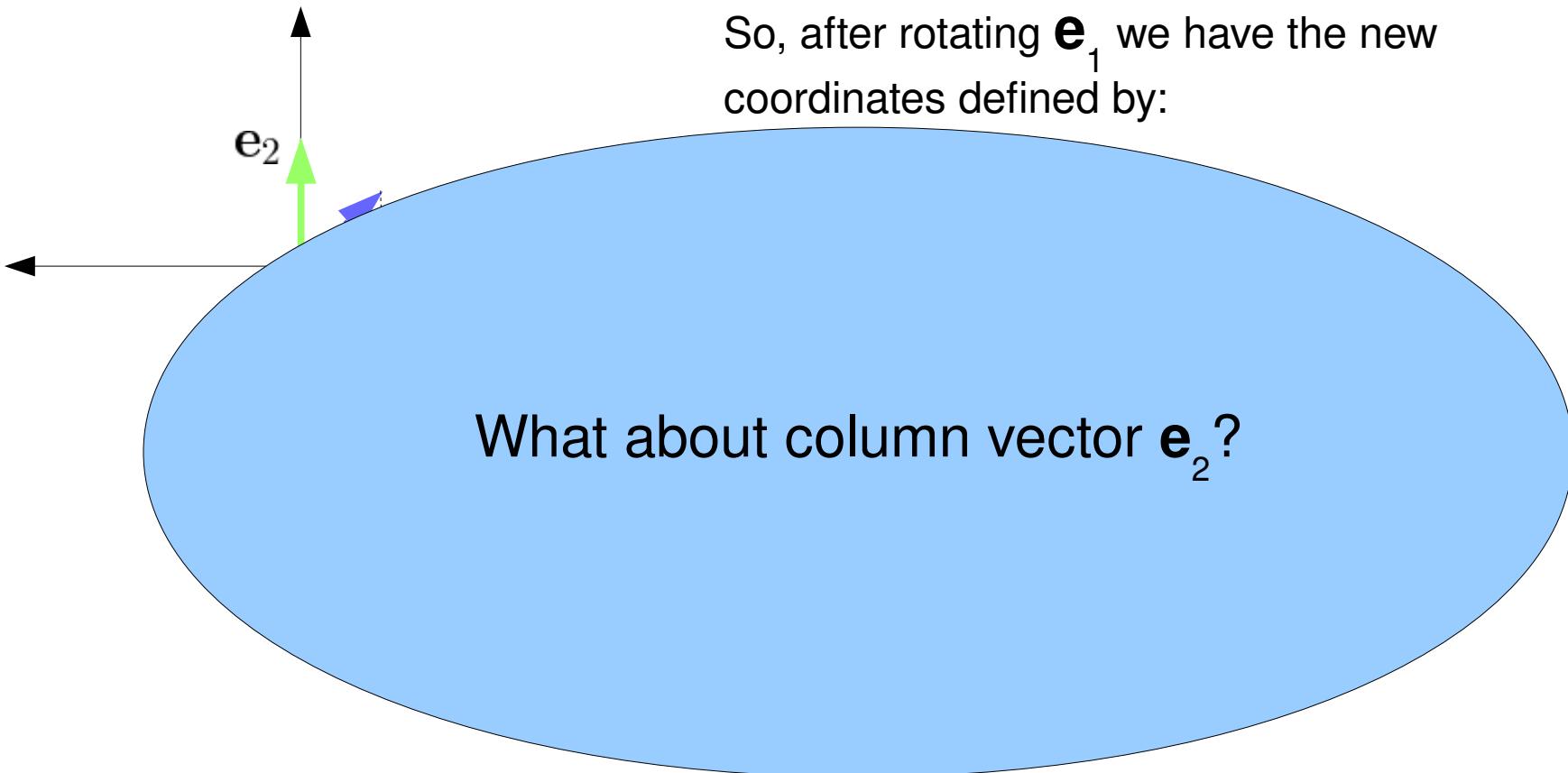


So, after rotating  $\mathbf{e}_1$  we have the new coordinates defined by:

$$\begin{bmatrix} \cos(\theta) \\ \sin(\theta) \end{bmatrix}$$

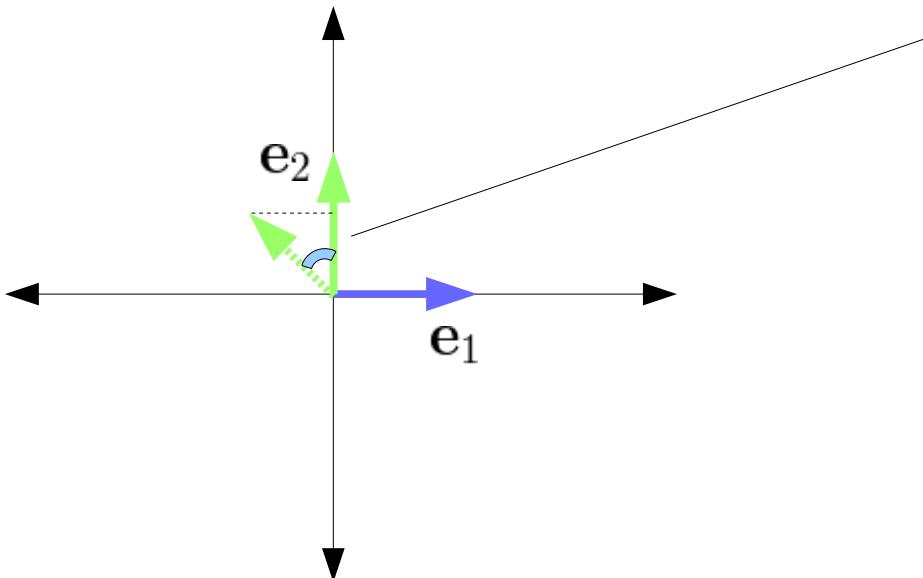
# Applying Linear Transformations

- Let's try a rotation over  $\mathbf{e}_1$ :



# Applying Linear Transformations

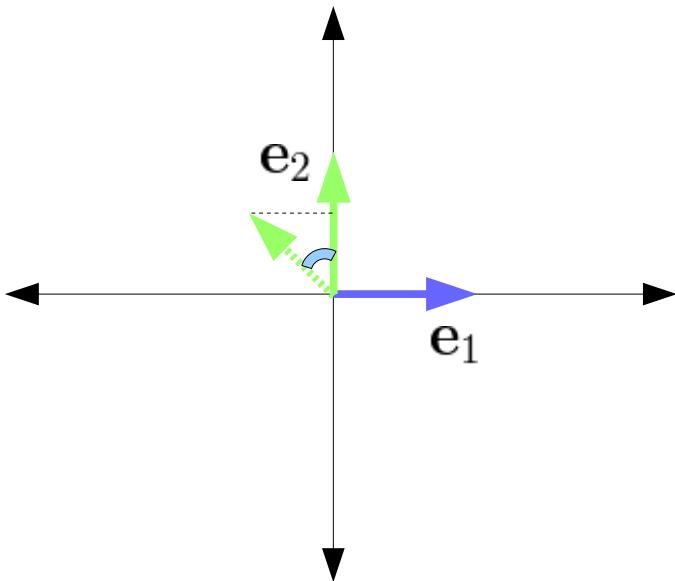
- Let's try a rotation over  $e_2$ :



Rotating theta degrees  
Observe the new values for both axes!  
See that the x-axis value depends on  
the  $\sin(\theta)$  → opposite to the angle  
But the x-axis is negative!  
While the y-axis depends on  
the  $\cos(\theta)$  → adjacent to the angle

# Applying Linear Transformations

- Let's try a rotation over  $\mathbf{e}_2$ :



So,  $\mathbf{e}_2$  will have the following values after such rotation:

$$\begin{bmatrix} -\sin(\theta) \\ \cos(\theta) \end{bmatrix}$$

# Applying Linear Transformations

- Now, getting back to our transformation matrix, we will have:

$$A = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

- What is the importance of this result?
  - Now we can represent the rotation using algebra
  - So the rotation is given by:

$$\text{Rot}_\theta(\mathbf{x}) : \mathbb{R}^2 \rightarrow \mathbb{R}^2$$

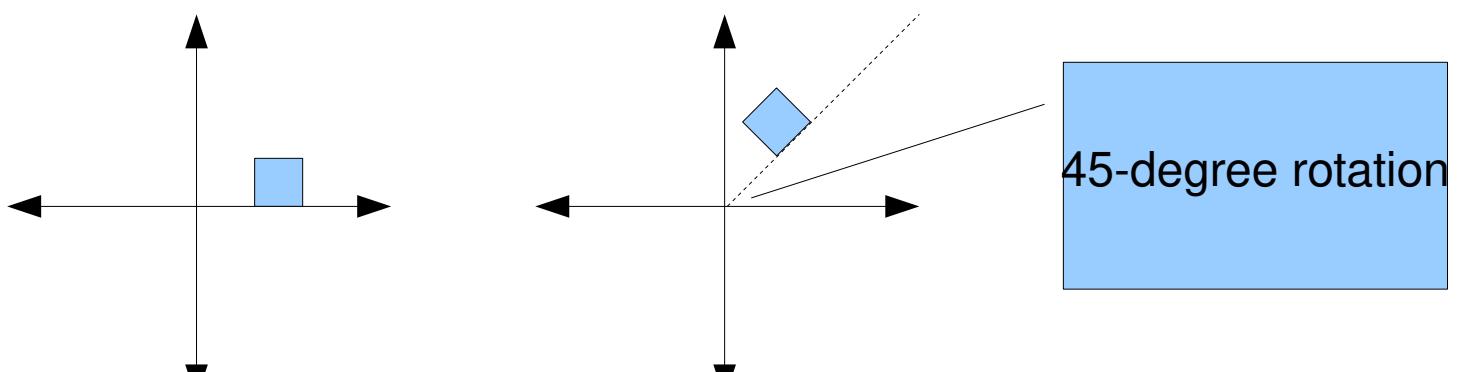
$$\text{Rot}_\theta(\mathbf{x}) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

# Applying Linear Transformations

- As an example, suppose we want to rotate a vector in 45 degrees:

$$\text{Rot}_{45^\circ}(\mathbf{x}) = \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- To better understand, suppose you have a shape formed using vectors
  - By applying a 45-degree rotation, the shape will have such rotation too



- See more at:
  - [https://pt.khanacademy.org/math/linear-algebra/matrix\\_transformations/lin\\_trans\\_examples/v/linear-transformation-examples-rotations-in-r2](https://pt.khanacademy.org/math/linear-algebra/matrix_transformations/lin_trans_examples/v/linear-transformation-examples-rotations-in-r2)

# Applying Linear Transformations

- As an example, suppose we want to rotate a vector in 45 degrees:

$$\text{Rot}_{45^\circ}(\mathbf{x}) = \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- To better understand, suppose you have a shape formed using vectors

- By applying linear transformations like rotation, scaling, etc., we can transform it.

This is very useful in Computer Graphics, Games, etc.

- See more examples

- [https://ptpmathbits.com/examples/video-tutorials/linear-transformations/rotations/lin\\_trans\\_rotation.html](https://ptpmathbits.com/examples/video-tutorials/linear-transformations/rotations/lin_trans_rotation.html)

## 4. Linear Algebra: Basis

- Consider two vectors that form a basis as follows:

$$\mathbf{v}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}; \quad \mathbf{v}_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

$$B = \{\mathbf{v}_1, \mathbf{v}_2\} \rightarrow \text{Basis for } \mathbb{R}^2$$

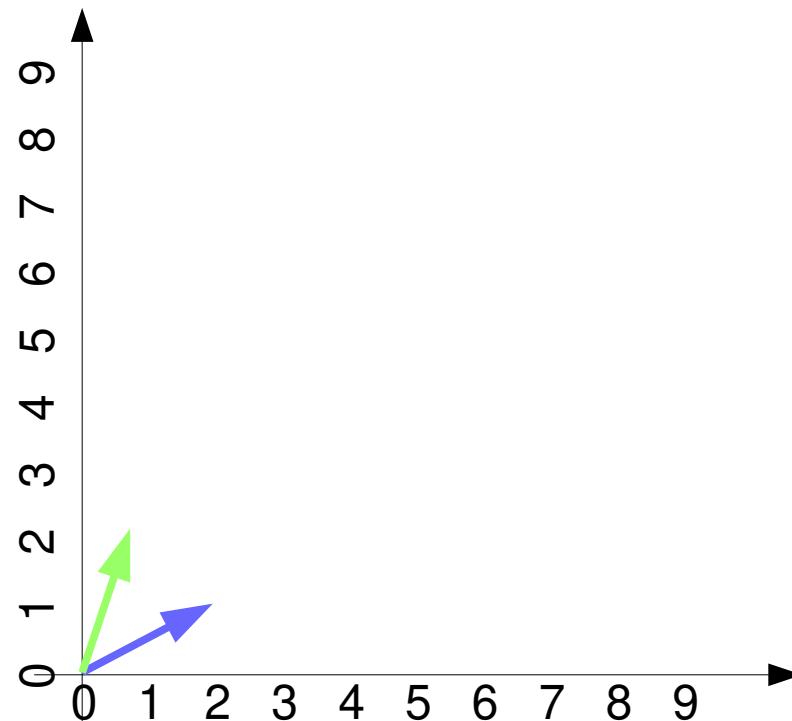
- Now suppose we want to obtain the vector:

$$\mathbf{a} = 3\mathbf{v}_1 + 2\mathbf{v}_2 = \begin{bmatrix} 8 \\ 7 \end{bmatrix}$$

- Consider two vectors that form a basis as follows:

$$\mathbf{v}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}; \quad \mathbf{v}_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$
$$B = \{\mathbf{v}_1, \mathbf{v}_2\} \rightarrow \text{Basis for } \mathbb{R}^2$$

$$\mathbf{a} = 3\mathbf{v}_1 + 2\mathbf{v}_2 = \begin{bmatrix} 8 \\ 7 \end{bmatrix}$$

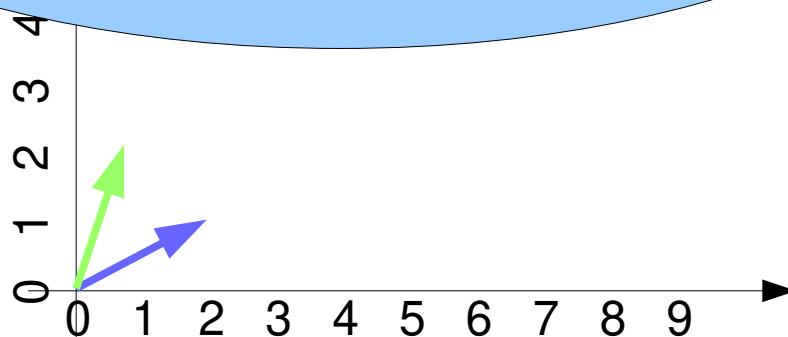


- Consider two vectors that form a basis as follows:

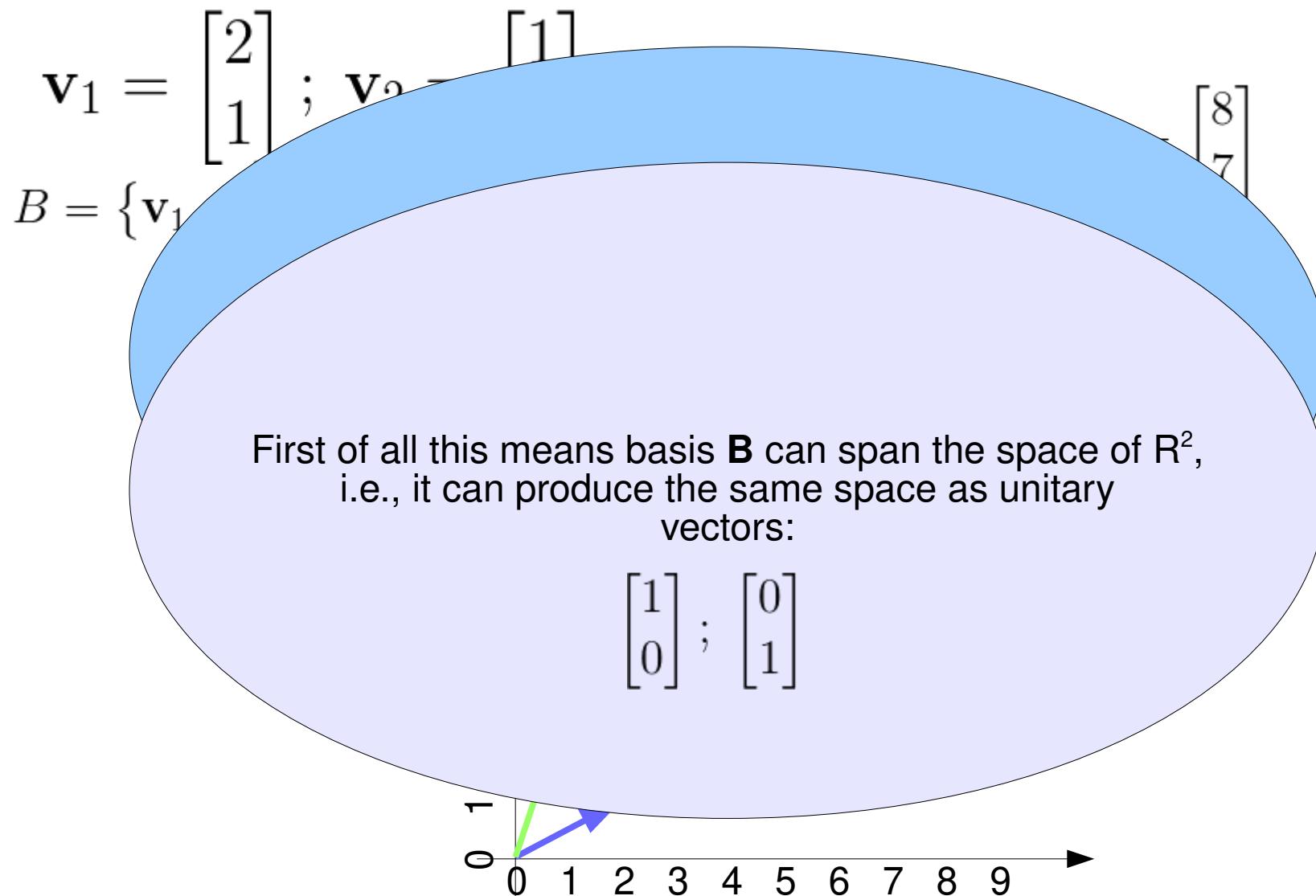
$$\mathbf{v}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}; \quad \mathbf{v}_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$
$$B = \{\mathbf{v}_1, \mathbf{v}_2\}$$

As vector  $\mathbf{a}$  is given by a linear combination of basis  $B$ , we can write it in terms of this basis as follows:

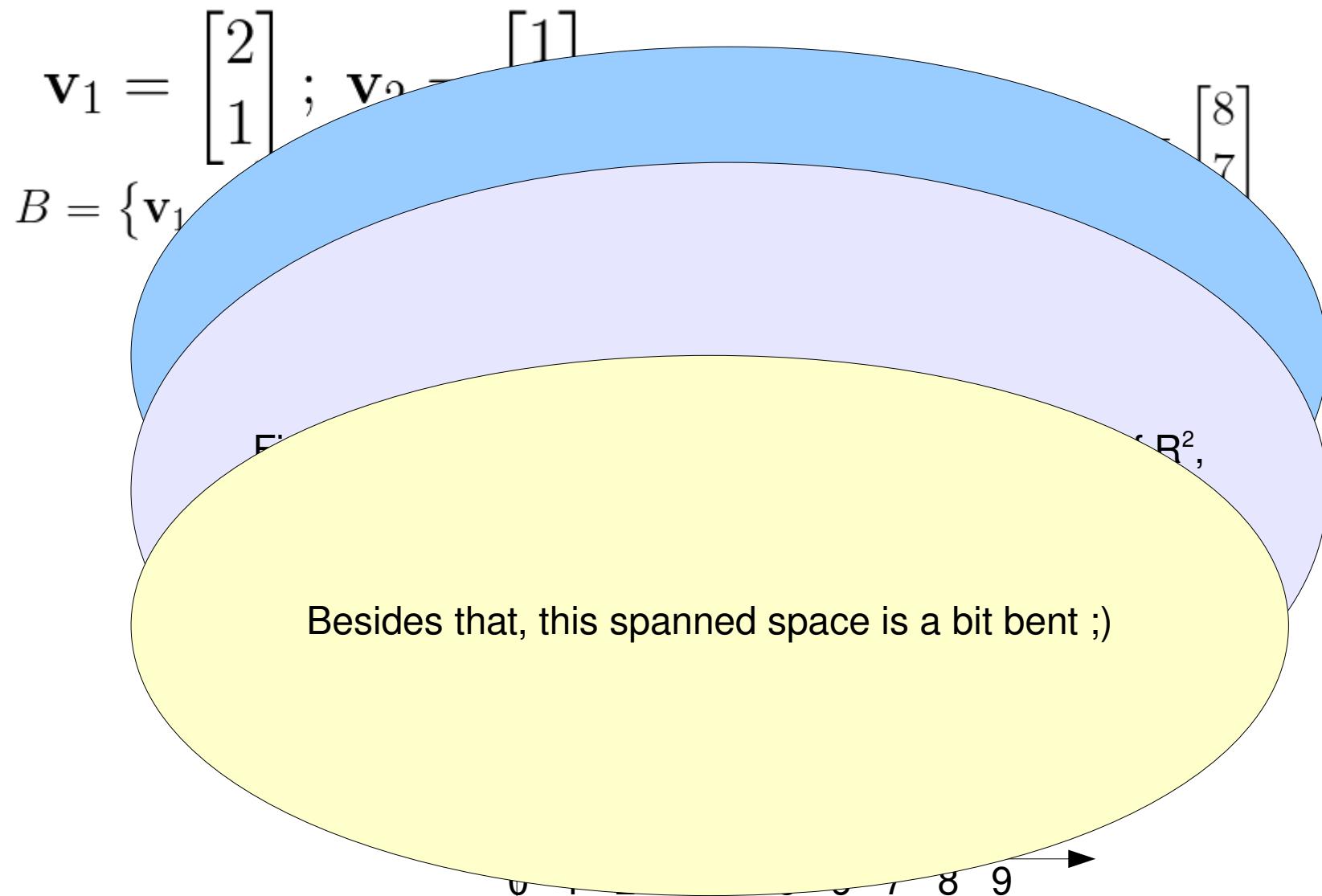
$$[\mathbf{a}]_B = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$



- Consider two vectors that form a basis as follows:



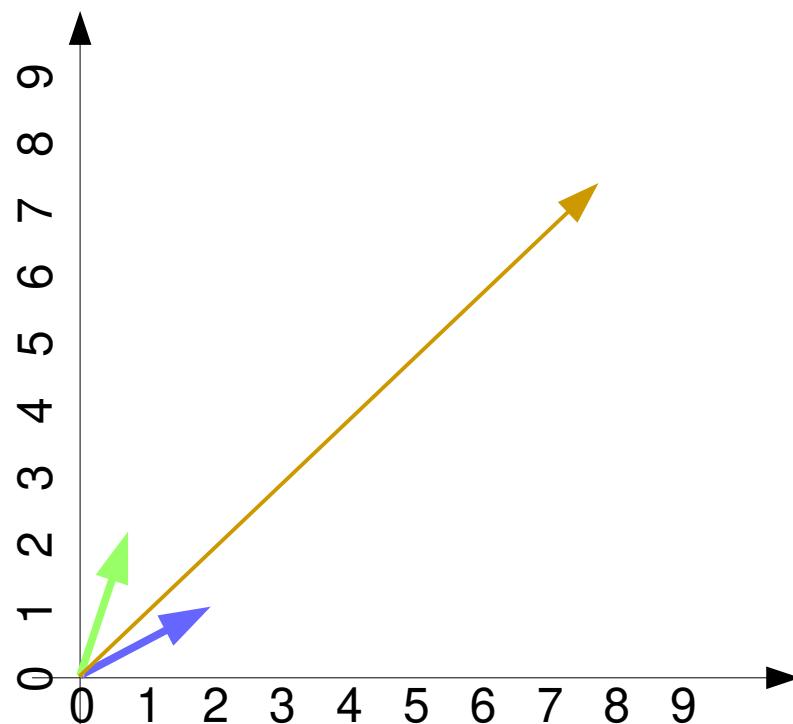
- Consider two vectors that form a basis as follows:



- We now draw the position vector  $\mathbf{a}$ :

$$\mathbf{v}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}; \mathbf{v}_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$
$$B = \{\mathbf{v}_1, \mathbf{v}_2\} \rightarrow \text{Basis for } \mathbb{R}^2$$

$$\mathbf{a} = 3\mathbf{v}_1 + 2\mathbf{v}_2 = \begin{bmatrix} 8 \\ 7 \end{bmatrix}$$



Let's understand what means to write vector  $\mathbf{a}$  in terms of basis B

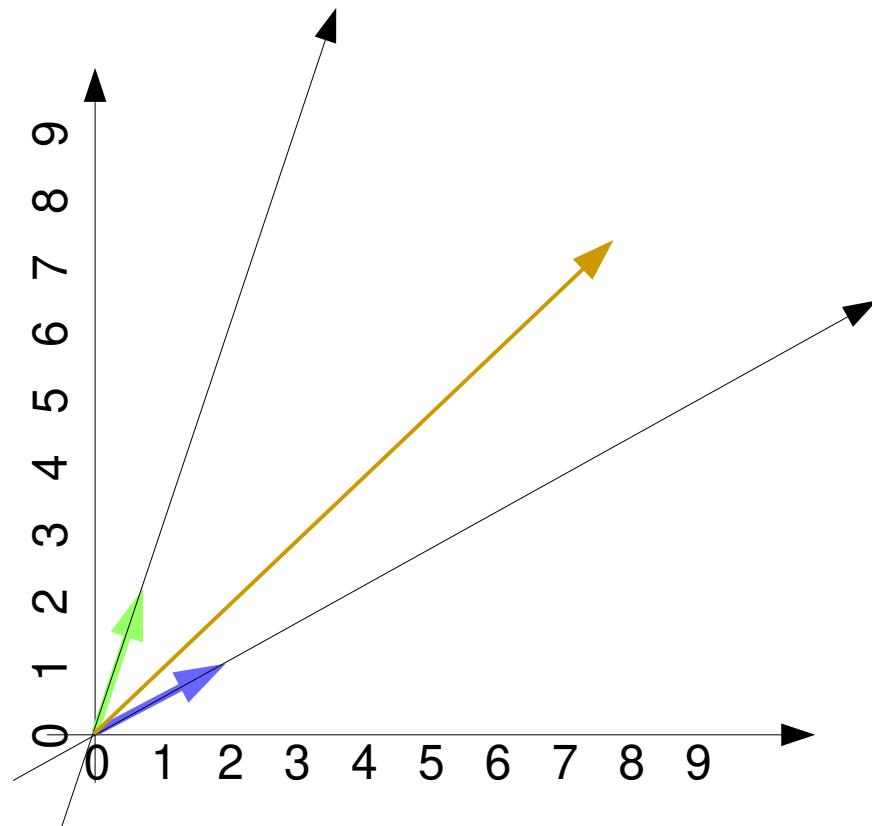
$$[\mathbf{a}]_B = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

- We now draw the position vector  $\mathbf{a}$ :

$$\mathbf{v}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}; \mathbf{v}_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

$B = \{\mathbf{v}_1, \mathbf{v}_2\} \rightarrow \text{Basis for } \mathbb{R}^2$

$$\mathbf{a} = 3\mathbf{v}_1 + 2\mathbf{v}_2 = \begin{bmatrix} 8 \\ 7 \end{bmatrix}$$



Let's understand what means to write vector  $\mathbf{a}$  in terms of basis  $B$

$$[\mathbf{a}]_B = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

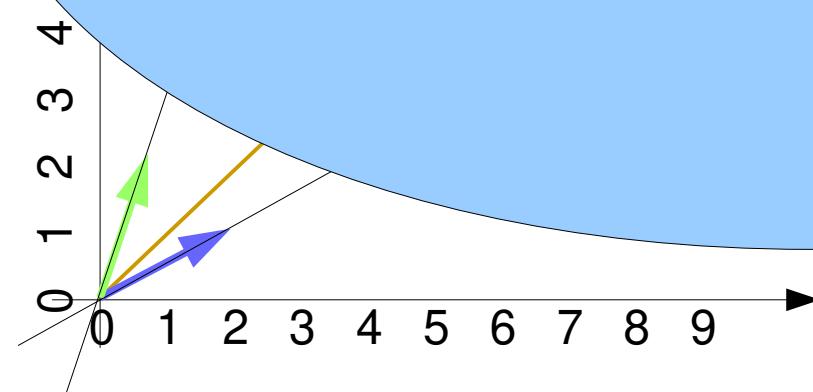
This is the same as considering axes formed by those two vectors  $\mathbf{v}_1$  and  $\mathbf{v}_2$

- We now draw the position vector **a**:

$$\mathbf{v}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}; \quad \mathbf{v}_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$
$$B = \{\mathbf{v}_1, \mathbf{v}_2\}$$

$$2\mathbf{v}_2 = \begin{bmatrix} 8 \\ 7 \end{bmatrix}$$

To use this new space representation,  
we draw parallels to axes as we usually do  
in the traditional cartesian plane

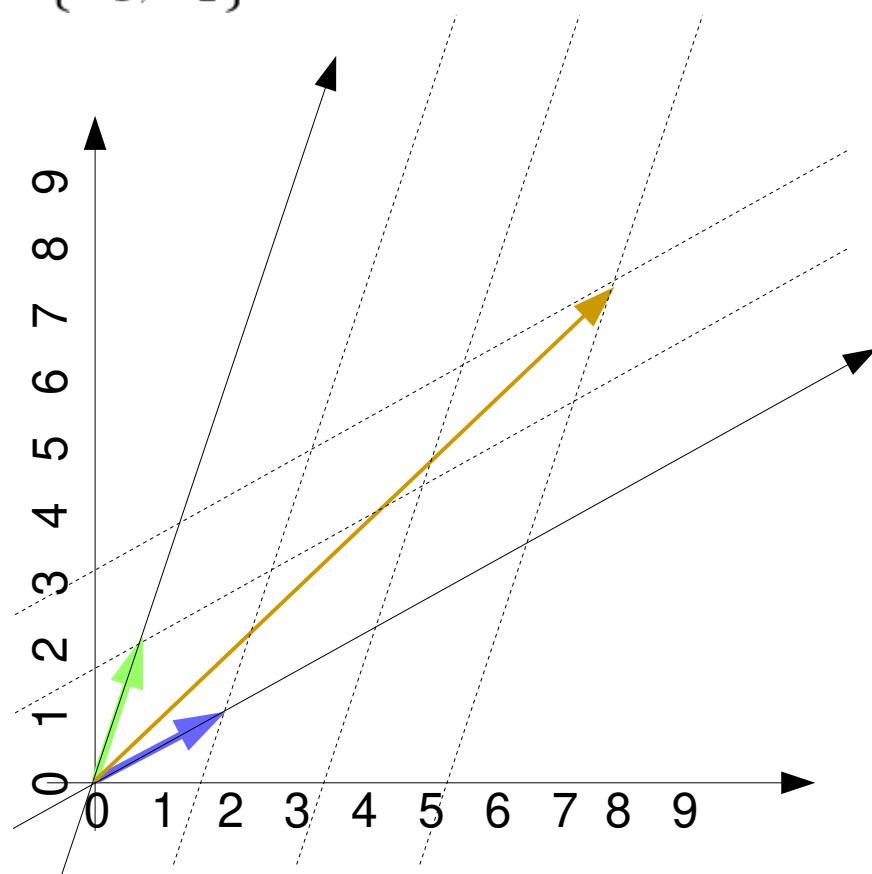


- We now draw the position vector  $\mathbf{a}$ :

$$\mathbf{v}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}; \mathbf{v}_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

$B = \{\mathbf{v}_1, \mathbf{v}_2\} \rightarrow \text{Basis for } \mathbb{R}^2$

$$\mathbf{a} = 3\mathbf{v}_1 + 2\mathbf{v}_2 = \begin{bmatrix} 8 \\ 7 \end{bmatrix}$$



Let's understand what means to write vector  $\mathbf{a}$  in terms of basis  $B$

$$[\mathbf{a}]_B = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

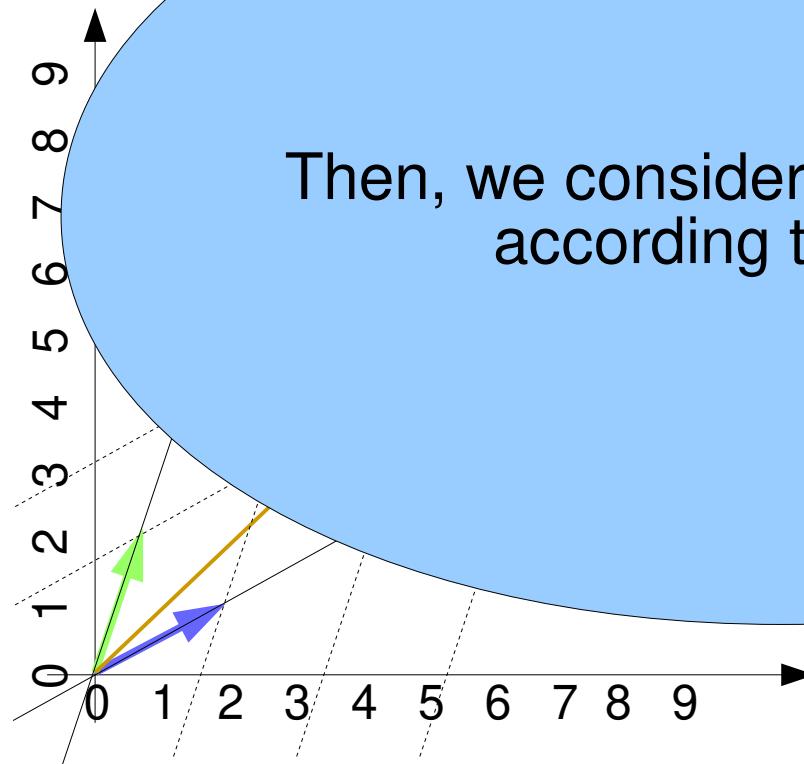
This is the same as considering axes formed by those two vectors  $\mathbf{v}_1$  and  $\mathbf{v}_2$

- We now draw the position vector  $\mathbf{a}$ :

$$\mathbf{v}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}; \mathbf{v}_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

$$B = \{\mathbf{v}_1, \mathbf{v}_2\} \rightarrow \text{Basis } B$$

$$\mathbf{a} = 3\mathbf{v}_1 + 2\mathbf{v}_2 = \begin{bmatrix} 8 \\ 7 \end{bmatrix}$$



Then, we consider the coordinates  
according to basis B

means to  
of basis B

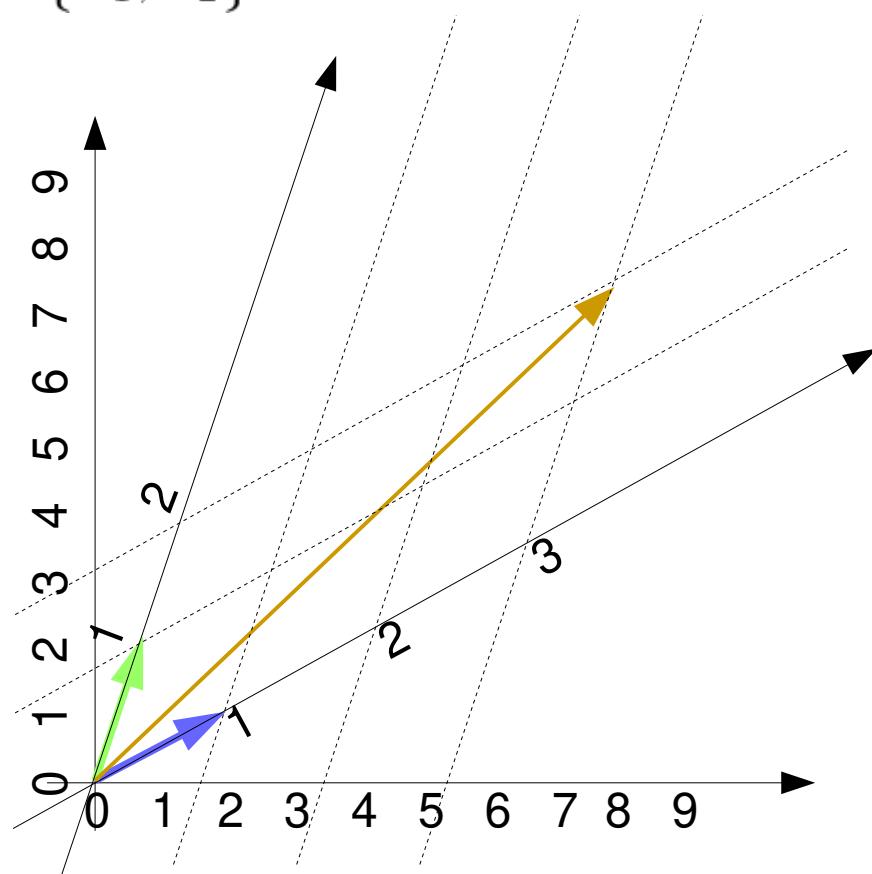
considering  
two vectors  $\mathbf{v}_1$

- We now draw the position vector  $\mathbf{a}$ :

$$\mathbf{v}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}; \mathbf{v}_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

$B = \{\mathbf{v}_1, \mathbf{v}_2\} \rightarrow \text{Basis for } \mathbb{R}^2$

$$\mathbf{a} = 3\mathbf{v}_1 + 2\mathbf{v}_2 = \begin{bmatrix} 8 \\ 7 \end{bmatrix}$$



Let's understand what means to write vector  $\mathbf{a}$  in terms of basis  $B$

$$[\mathbf{a}]_B = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

This is the same as considering axes formed by those two vectors  $\mathbf{v}_1$  and  $\mathbf{v}_2$

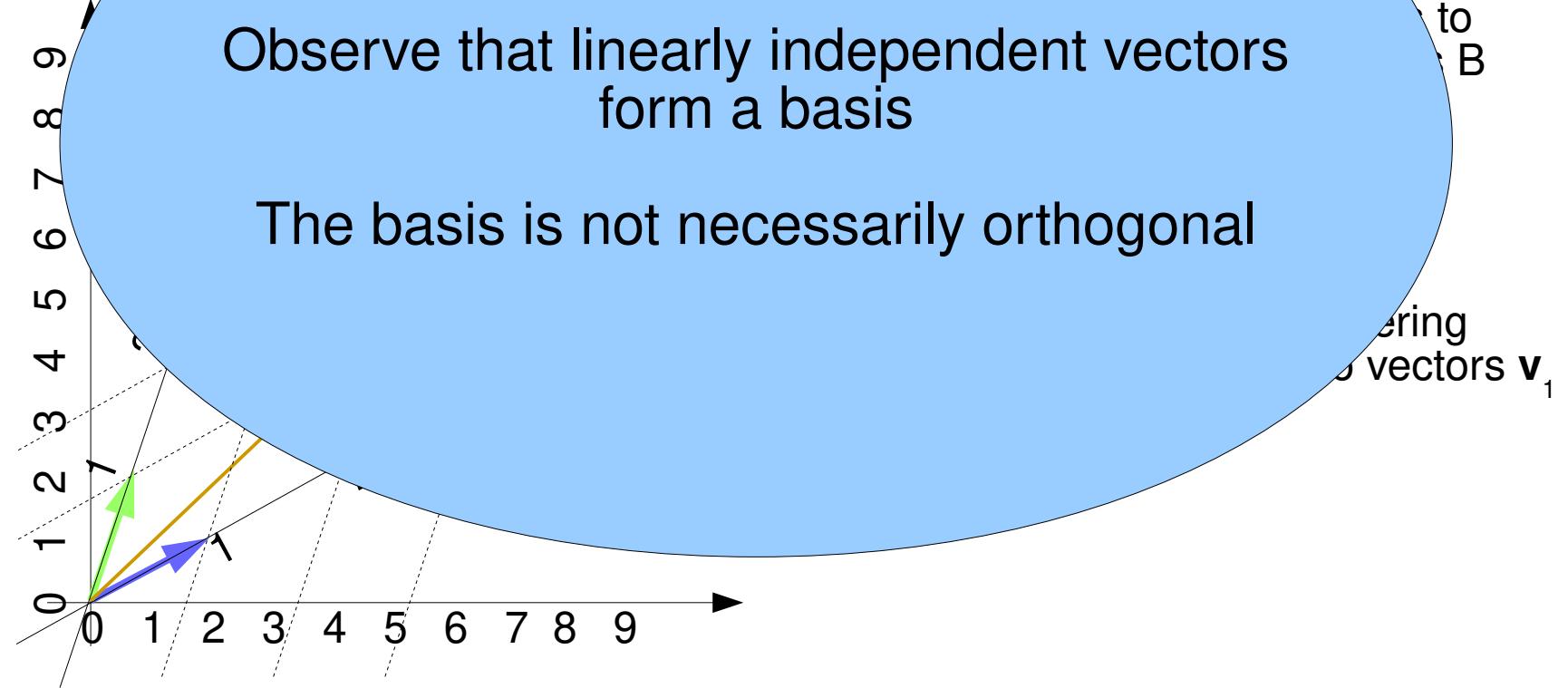
- We now draw the position vector  $\mathbf{a}$ :

$$\mathbf{v}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}; \quad \mathbf{v}_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$
$$B = \{\mathbf{v}_1, \mathbf{v}_2\}$$

$$2\mathbf{v}_2 = \begin{bmatrix} 8 \\ 7 \end{bmatrix}$$

Observe that linearly independent vectors form a basis

The basis is not necessarily orthogonal



- See more at:
  - [https://pt.khanacademy.org/math/linear-algebra/alternate\\_bases/change\\_of\\_bases/v/linear-algebra-coordinates-with-respect-to-a-basis](https://pt.khanacademy.org/math/linear-algebra/alternate_bases/change_of_bases/v/linear-algebra-coordinates-with-respect-to-a-basis)

## 5. Linear Algebra: Changing Basis

# Changing Basis

- Consider two vectors that form a basis as follows:

$$\mathbf{v}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}; \quad \mathbf{v}_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

$$B = \{\mathbf{v}_1, \mathbf{v}_2\} \rightarrow \text{Basis for } \mathbb{R}^2$$

- Now consider we have a vector **a** with respect to basis B

$$[\mathbf{a}]_B = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

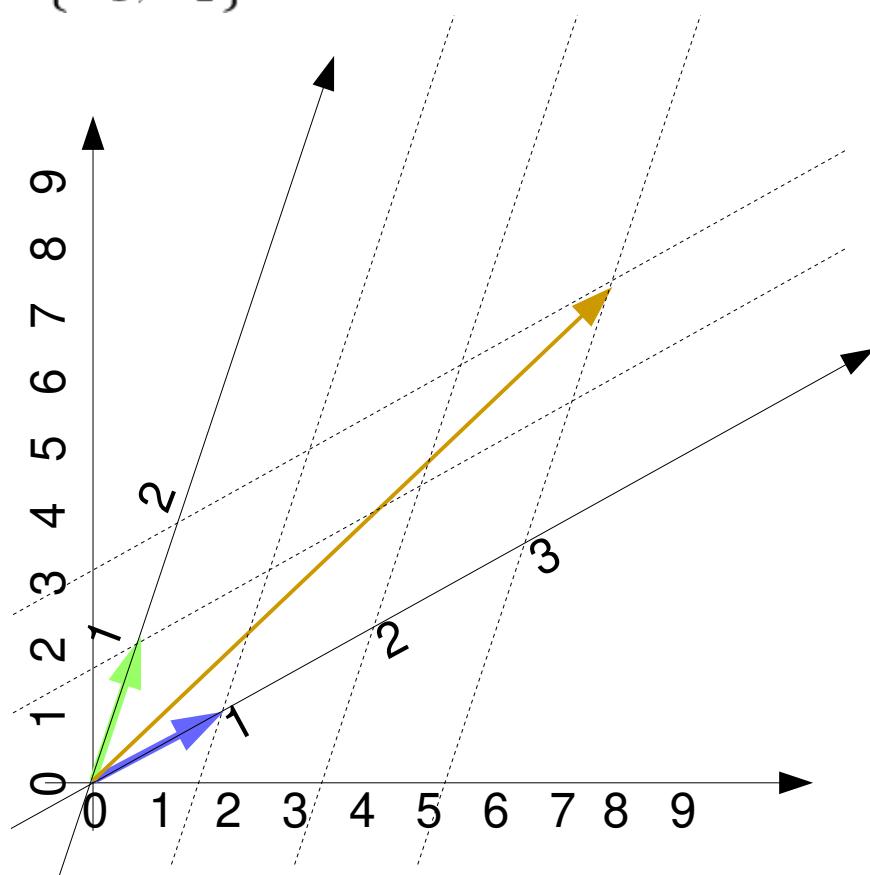
- What does that mean?

# Changing Basis

- It means the following:

$$\mathbf{v}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}; \mathbf{v}_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

$$B = \{\mathbf{v}_1, \mathbf{v}_2\} \rightarrow \text{Basis for } \mathbb{R}^2$$



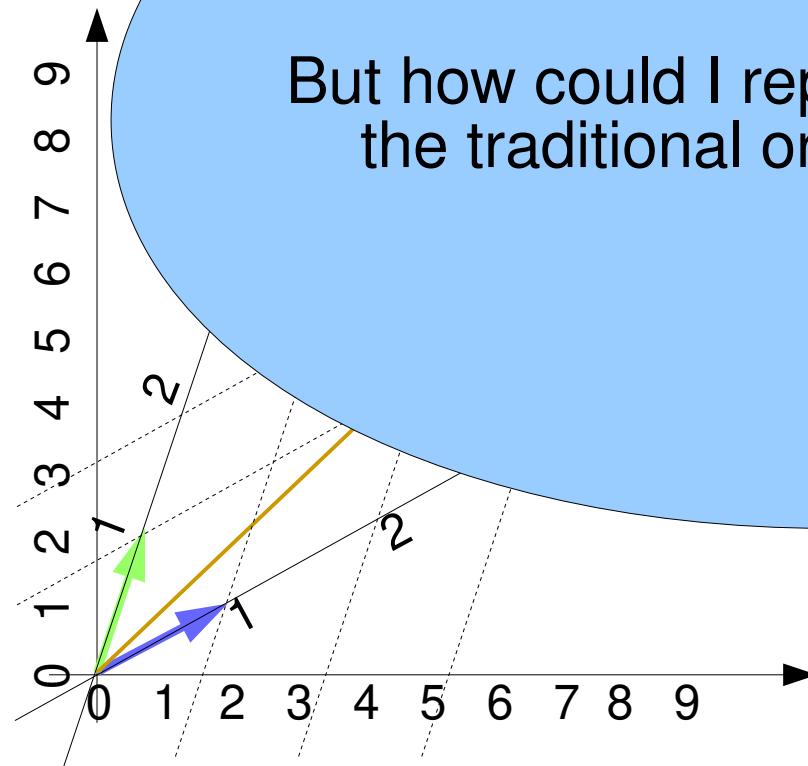
Here vector  $\mathbf{a}$  is represented according to basis  $B$ , i.e.,

$$[\mathbf{a}]_B = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

# Changing Basis

- It means the following:

$$\mathbf{v}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}; \quad \mathbf{v}_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$
$$B = \{\mathbf{v}_1, \mathbf{v}_2\}$$



But how could I represent it in terms of  
the traditional orthonormal basis?

# Changing Basis

- We simply apply a linear transformation on  $[\mathbf{a}]_B$  as follows:

$$C [\mathbf{a}]_B = \mathbf{a}$$

- In which C is a transformation matrix formed with the column vectors of basis B as follows:

$$C = \begin{bmatrix} \vdots & & \vdots \\ \mathbf{v}_1 & \dots & \mathbf{v}_k \\ \vdots & & \vdots \end{bmatrix}$$

- Which is here referred to as **changing basis matrix**

# Changing Basis

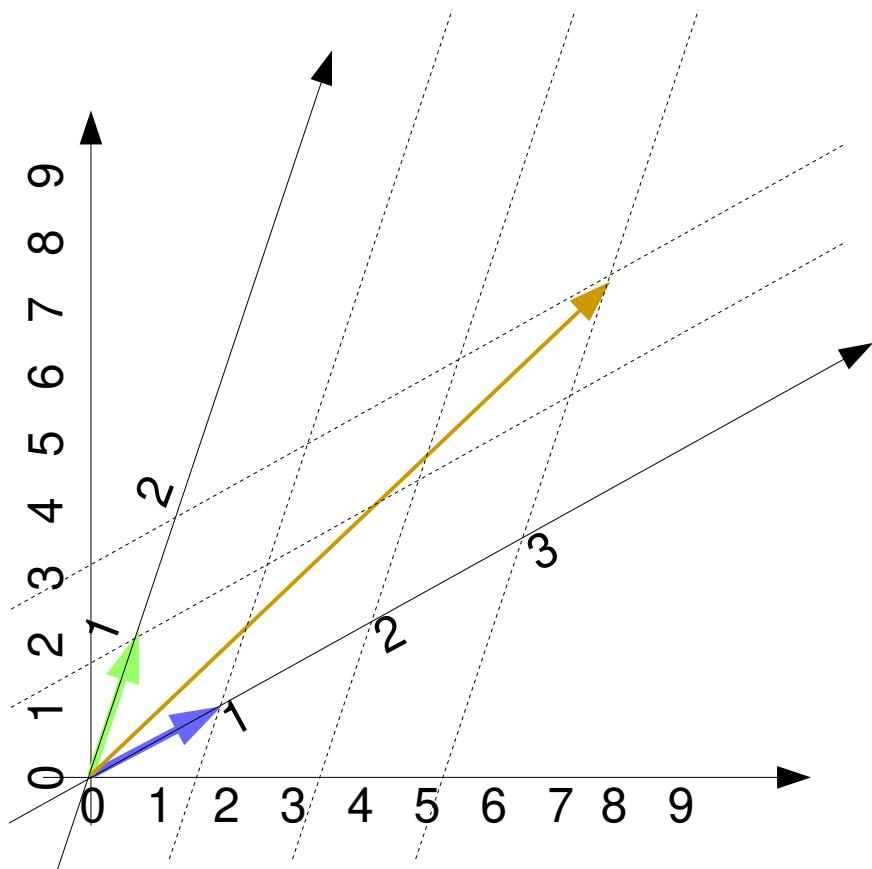
- So, in the standard basis we will have:

$$C [\mathbf{a}]_B = \mathbf{a}$$

$$\mathbf{a} = C [\mathbf{a}]_B = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 8 \\ 7 \end{bmatrix}$$

# Changing Basis

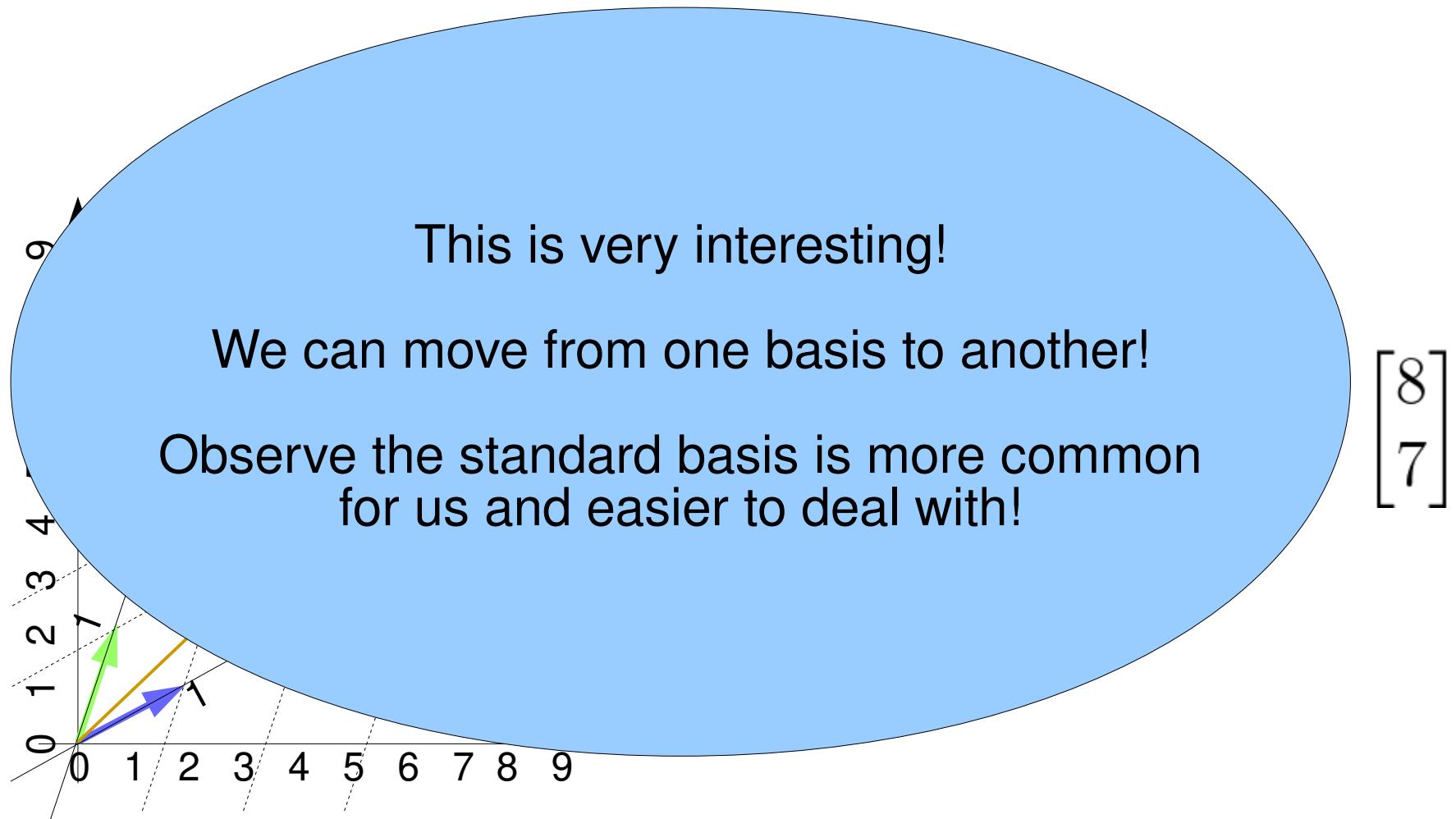
- This is exactly the result we expected!



$$[\mathbf{a}]_B = \begin{bmatrix} 3 \\ 2 \end{bmatrix}; \mathbf{a} = \begin{bmatrix} 8 \\ 7 \end{bmatrix}$$

# Changing Basis

- This is exactly the result we expected!



# Changing Basis

- We simply define matrix C using the column vectors used to form basis B

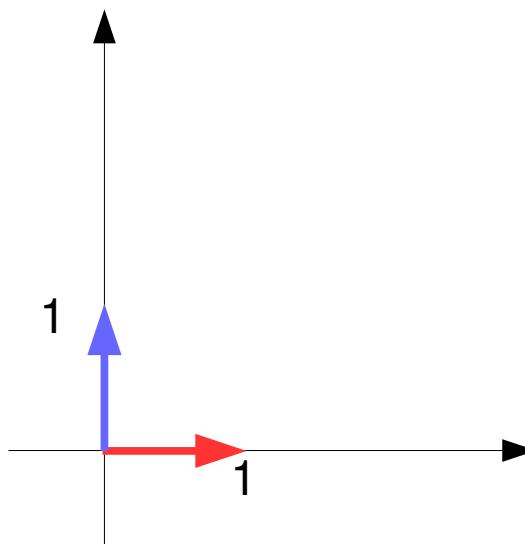
$$\mathbf{v}_1 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}; \mathbf{v}_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$$

$$B = \{\mathbf{v}_1, \mathbf{v}_2\} \rightarrow \text{Basis for } \mathbb{R}^2$$

- So now, what means the inverse of matrix C ??
- See more at:
  - [https://pt.khanacademy.org/math/linear-algebra/alternate\\_bases/change\\_of\\_basis/v/linear-algebra-change-of-basis-matrix](https://pt.khanacademy.org/math/linear-algebra/alternate_bases/change_of_basis/v/linear-algebra-change-of-basis-matrix)

## 6. Linear Algebra: Orthonormal basis

- Orthonormal basis
  - Every column vector has length equal to 1
    - Normalized
  - The dot product of every possible pair of different column vectors is equal to 0
    - Orthogonal, therefore linearly independent



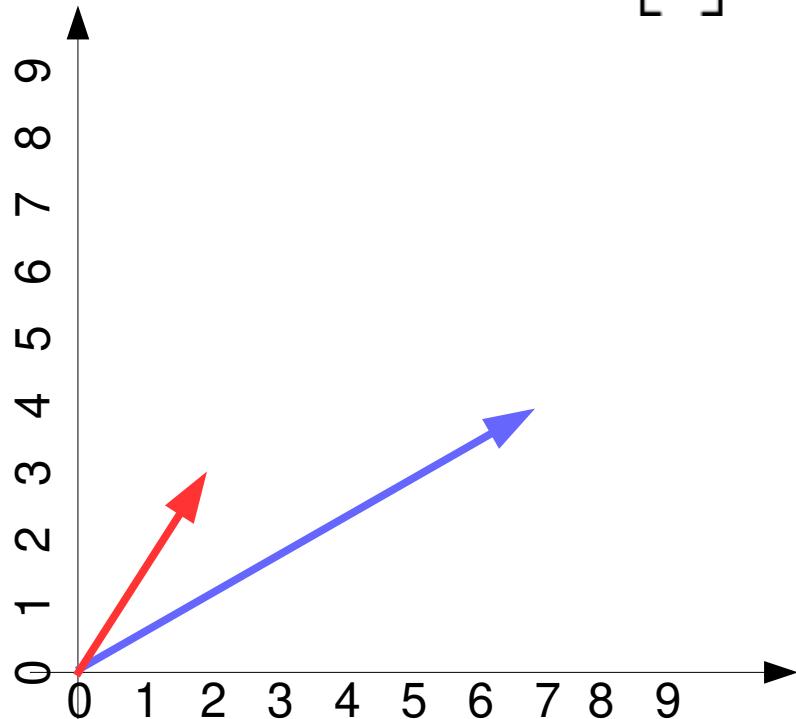
- Why Orthonormal bases?
  - Because they form **optimal coordinate systems**
    - Simpler to work with: every vector in space is formed by multiplying each column vector by a scalar and summing up all results
  - Example:

Standard basis for  $\mathbb{R}^n$  : 
$$\left\{ \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix}, \dots, \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix} \right\}$$

# Orthonormal Basis

For example, consider the two vectors which are capable of spanning  $\mathbb{R}^2$

$$\mathbf{v}_1 = \begin{bmatrix} 2 \\ 3 \end{bmatrix}; \quad \mathbf{v}_2 = \begin{bmatrix} 7 \\ 4 \end{bmatrix}$$

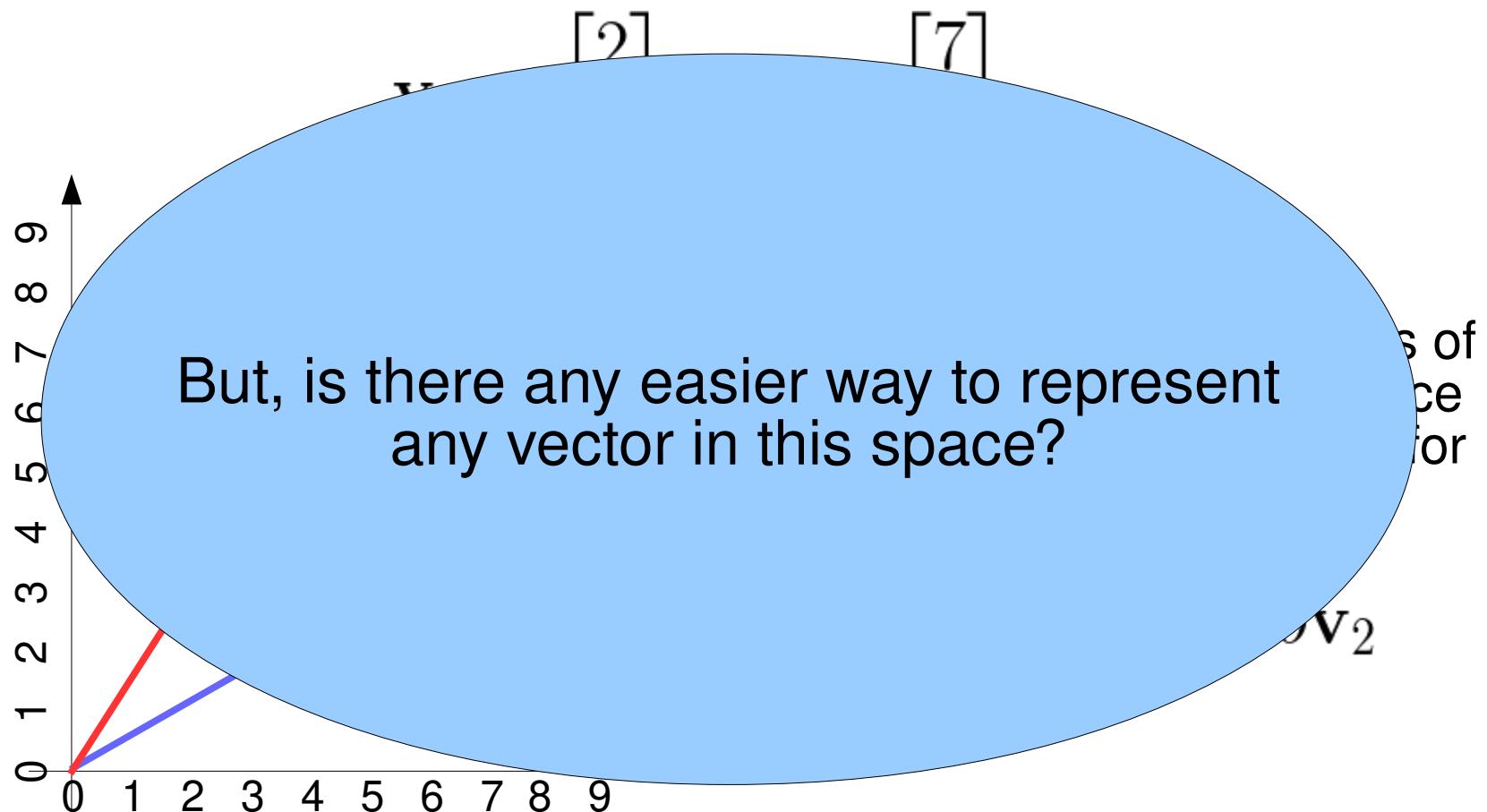


So, linear combinations of both vectors can produce any other vector in  $\mathbb{R}^2$ , for example:

$$\mathbf{x} = a\mathbf{v}_1 + b\mathbf{v}_2$$

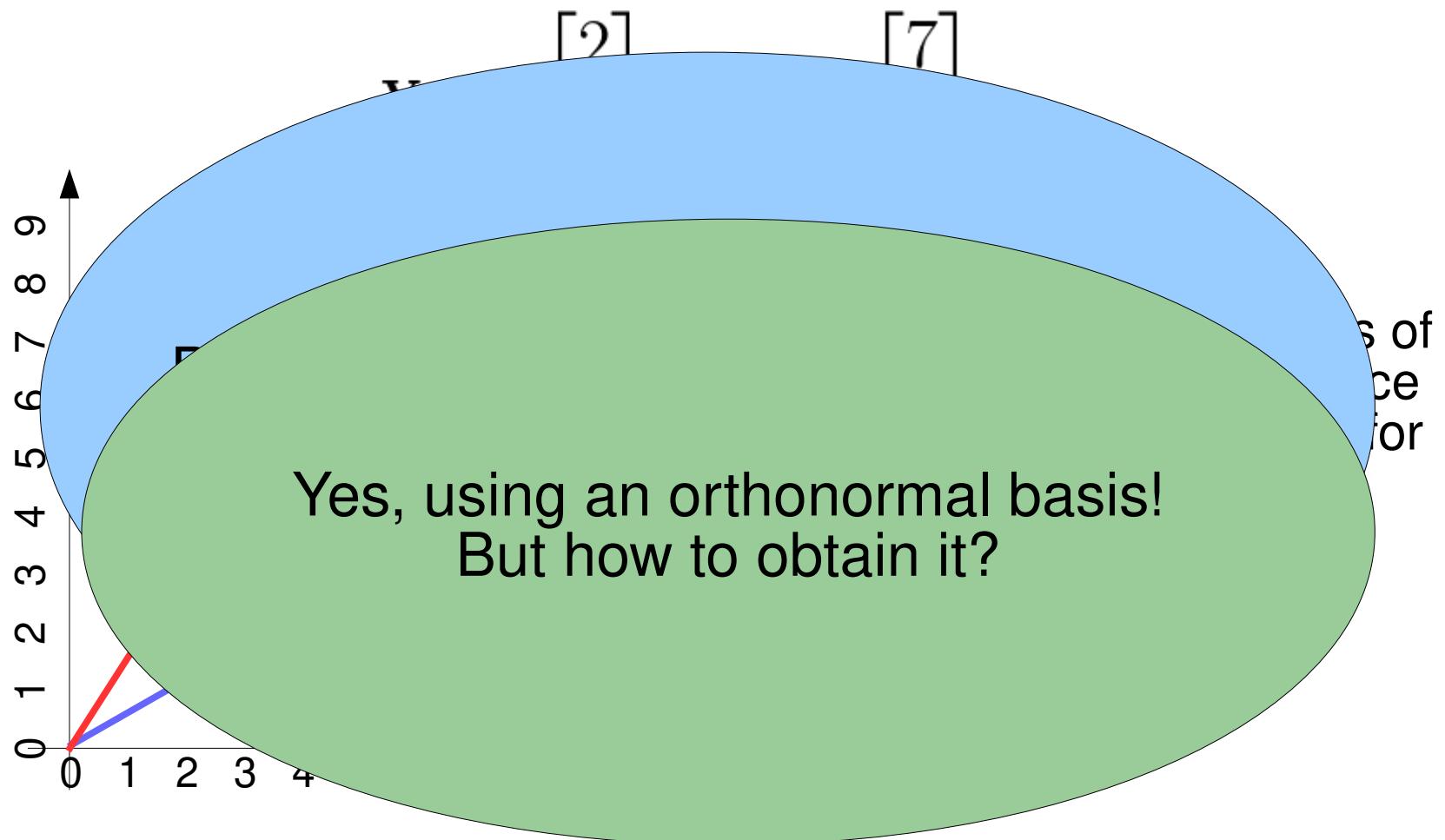
# Orthonormal Basis

For example, consider the two vectors which are capable of spanning  $\mathbb{R}^2$



# Orthonormal Basis

For example, consider the two vectors which are capable of spanning  $\mathbb{R}^2$



We will use the Gram-Schmidt process:

- First of all consider we have a basis for the space, in this case:

$$B = \{\mathbf{v}_1, \mathbf{v}_2\}$$

- Given by:

$$\mathbf{v}_1 = \begin{bmatrix} 2 \\ 3 \end{bmatrix}; \quad \mathbf{v}_2 = \begin{bmatrix} 7 \\ 4 \end{bmatrix}$$

Gram-Schmidt process:

- We now proceed to produce an orthonormal vector for the first column vector, without considering any other

$$\mathbf{v}_1 = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

Gram-Schmidt process:

- We now proceed to produce an orthonormal vector for the first column vector, without considering any other

$$\mathbf{v}_1 = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

- As no other vector is considered, then it is already orthogonal, thus we only need to normalize it as follows:

$$\mathbf{u}_1 = \frac{\mathbf{v}_1}{\|\mathbf{v}_1\|} = \frac{1}{\sqrt{2^2 + 3^2}} \begin{bmatrix} 2 \\ 3 \end{bmatrix} = \frac{1}{\sqrt{13}} \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

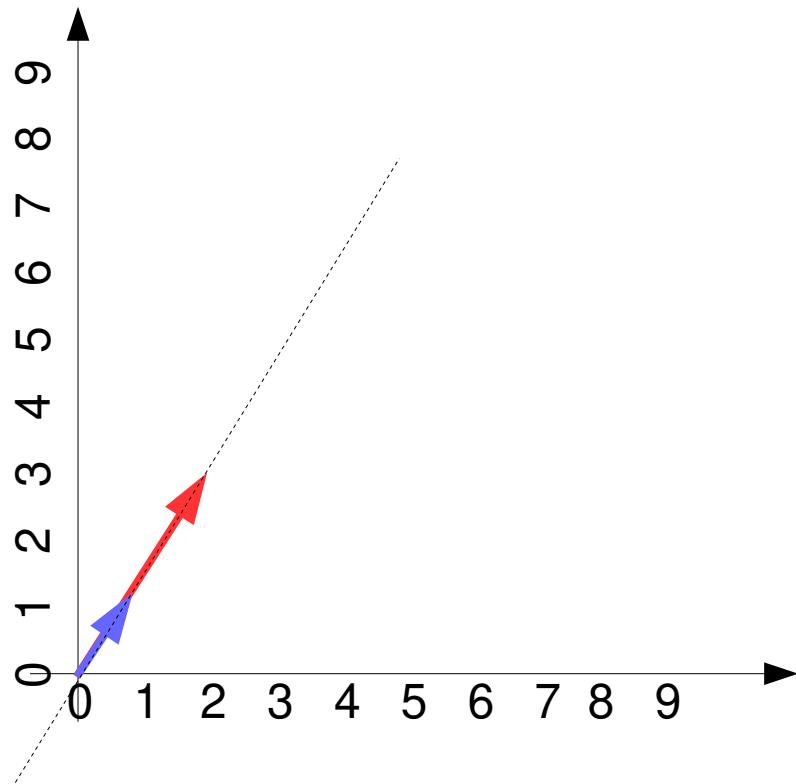
Gram-Schmidt process:

- We now proceed to produce an orthonormal vector for the first column vector, without considering any other
- This vector  $\mathbf{u}_1$  has length equals to 1 and alone spans  $\mathbb{R}^1$

Gram-Schmidt process:

- See the span for:

$$S_1 = \text{span}(\mathbf{u}_1)$$



See  $\mathbf{v}_1$  in red and its normalized version  $\mathbf{u}_1$  in blue

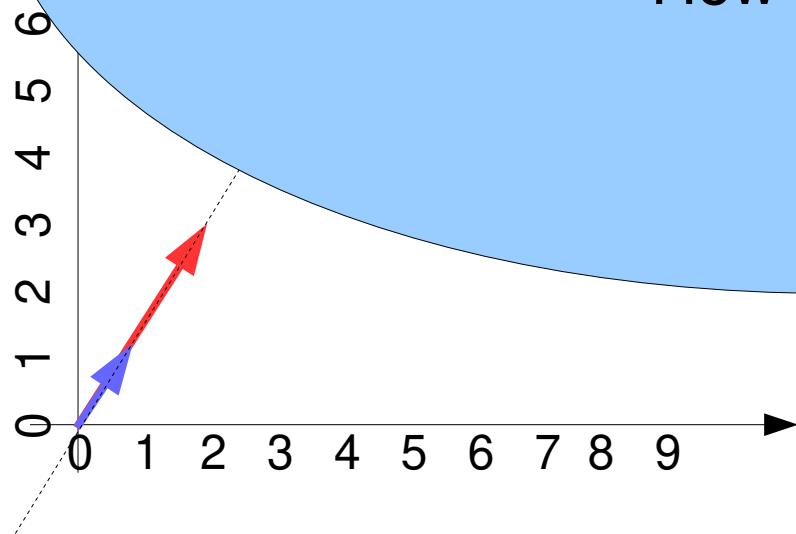
The space  $S_1$  is produced when the basis vector  $\mathbf{u}_1$  is multiplied by any real scalar (see dashed line, which goes to infinity to both sides)

Gram-Schmidt process:

- See the span for:

Yes, now we have at least one basis vector as orthonormal, but we still need to transform the other!

How to do it?



Gram-Schmidt process:

- Now we want to transform the basis:

$$B_1 = \{\mathbf{u}_1, \mathbf{v}_2\}$$

- In an orthonormal basis by changing  $\mathbf{v}_2$

Gram-Schmidt process:

- Now we want to transform the basis:

$$B_1 = \{\mathbf{u}_1, \mathbf{v}_2\}$$

- In an orthonormal basis by changing  $\mathbf{v}_2$
- To obtain an orthonormal basis we need to project  $\mathbf{v}_2$  into the space produced by  $\mathbf{u}_1$ , which is  $S_1$

$$\text{Proj}_{S_1}(\mathbf{v}_2) = \frac{\mathbf{v}_2 \cdot \mathbf{u}_1}{\|\mathbf{u}_1\|} \mathbf{u}_1$$

Gram-Schmidt process:

- Now we want to transform the basis:

- In an n-dimensional space
- To an orthonormal basis

What is the interpretation?

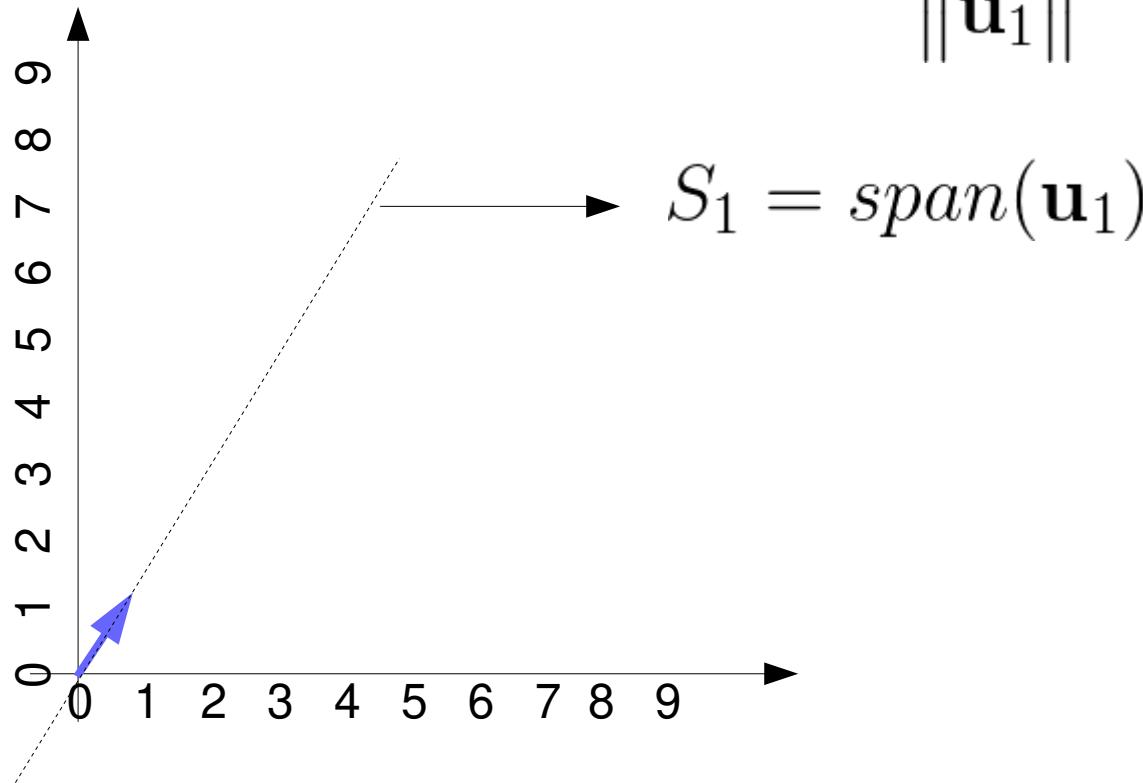
into the

$$\text{Proj}_{S_1}(\mathbf{v}_2) = \frac{\mathbf{v}_2 - \mathbf{u}_1}{\|\mathbf{u}_1\|} \mathbf{u}_1$$

Gram-Schmidt process:

- To obtain an orthonormal basis we need to project  $\mathbf{v}_2$  into the space produced by  $\mathbf{u}_1$ , which is  $S_1$

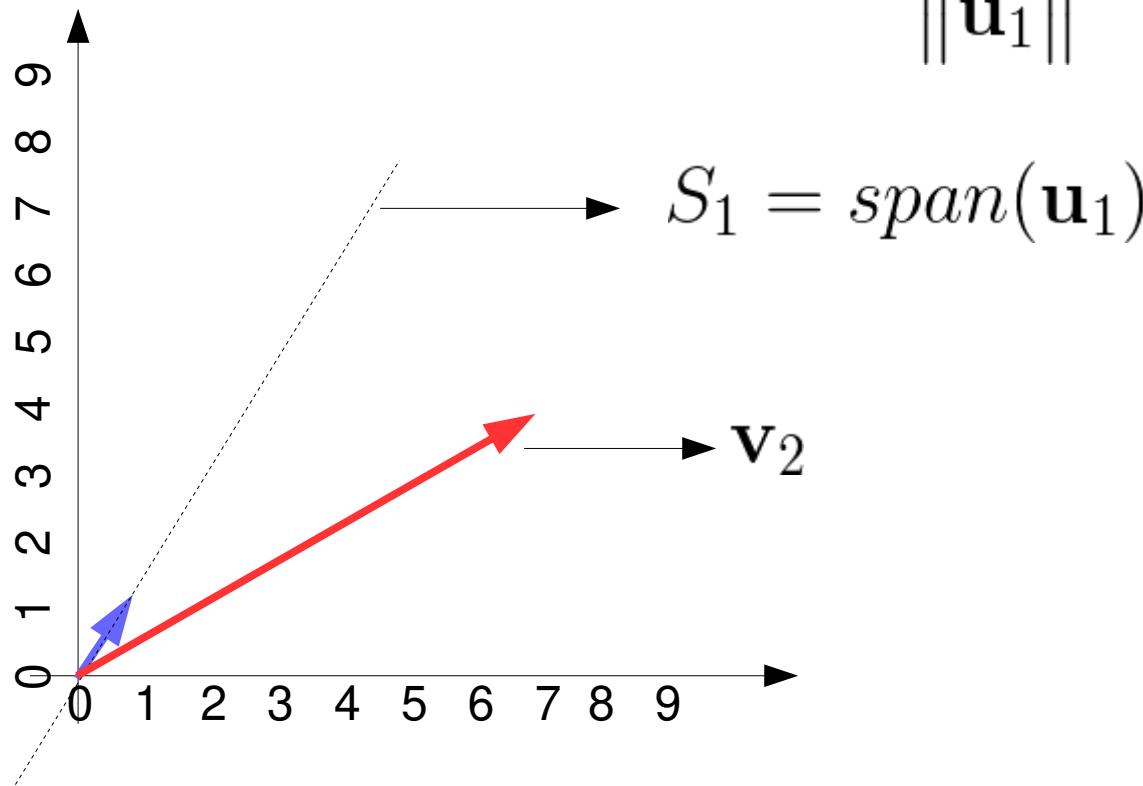
$$\text{Proj}_{S_1}(\mathbf{v}_2) = \frac{\mathbf{v}_2 \cdot \mathbf{u}_1}{\|\mathbf{u}_1\|} \mathbf{u}_1$$



Gram-Schmidt process:

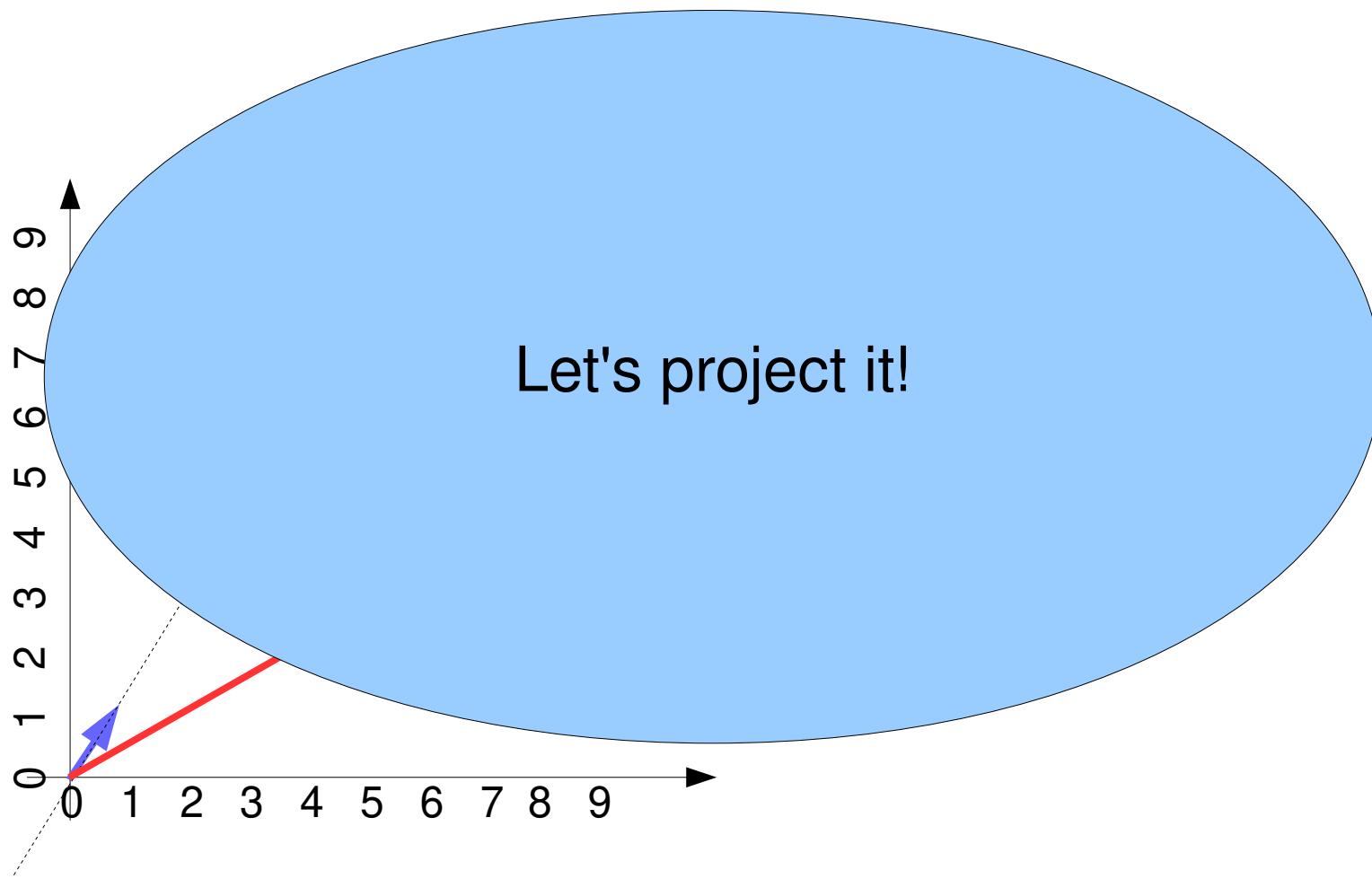
- To obtain an orthonormal basis we need to project  $\mathbf{v}_2$  into the space produced by  $\mathbf{u}_1$ , which is  $S_1$

$$\text{Proj}_{S_1}(\mathbf{v}_2) = \frac{\mathbf{v}_2 \cdot \mathbf{u}_1}{\|\mathbf{u}_1\|} \mathbf{u}_1$$



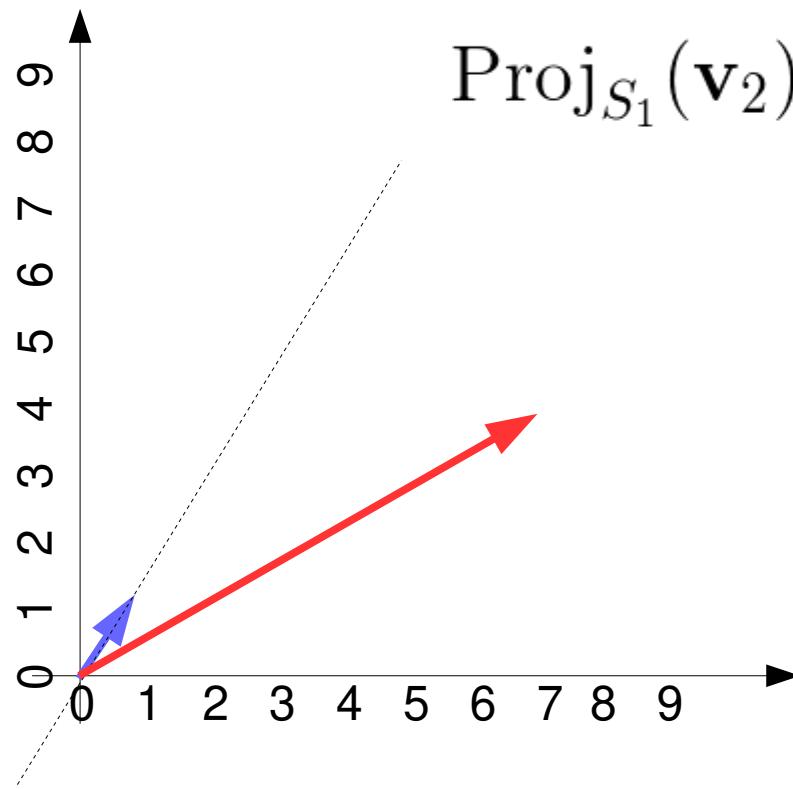
Gram-Schmidt process:

- To obtain an orthonormal basis we need to project  $\mathbf{v}_2$  into the space produced by  $\mathbf{u}_1$ , which is  $S_1$



Gram-Schmidt process:

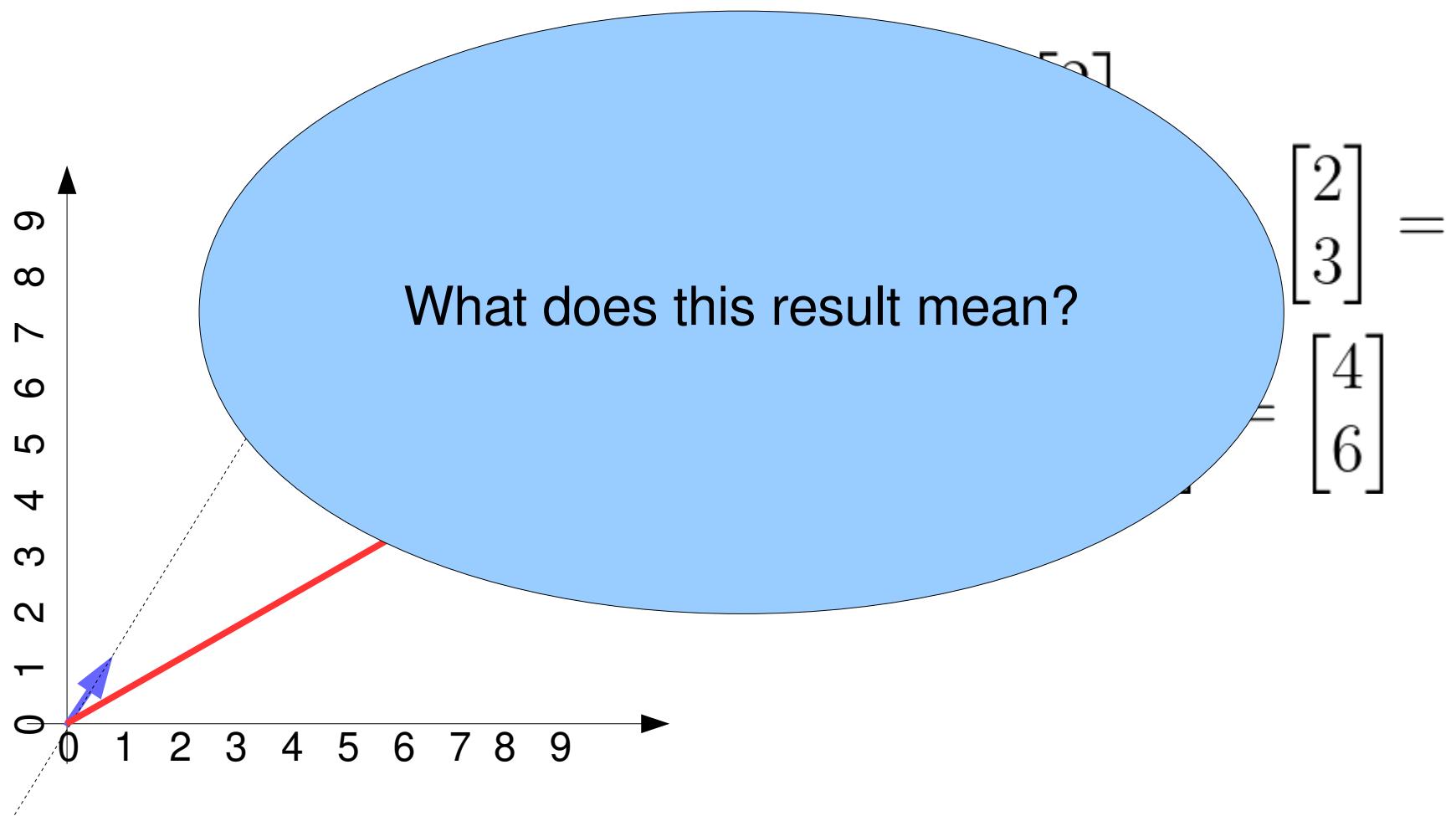
- To obtain an orthonormal basis we need to project  $\mathbf{v}_2$  into the space produced by  $\mathbf{u}_1$ , which is  $S_1$



$$\text{Proj}_{S_1}(\mathbf{v}_2) = \frac{\begin{bmatrix} 7 \\ 4 \end{bmatrix} \cdot \frac{1}{\sqrt{13}} \begin{bmatrix} 2 \\ 3 \end{bmatrix}}{1} \frac{1}{\sqrt{13}} \begin{bmatrix} 2 \\ 3 \end{bmatrix} = \frac{14 + 12}{\sqrt{13}} \frac{1}{\sqrt{13}} \begin{bmatrix} 2 \\ 3 \end{bmatrix} = \begin{bmatrix} 4 \\ 6 \end{bmatrix}$$

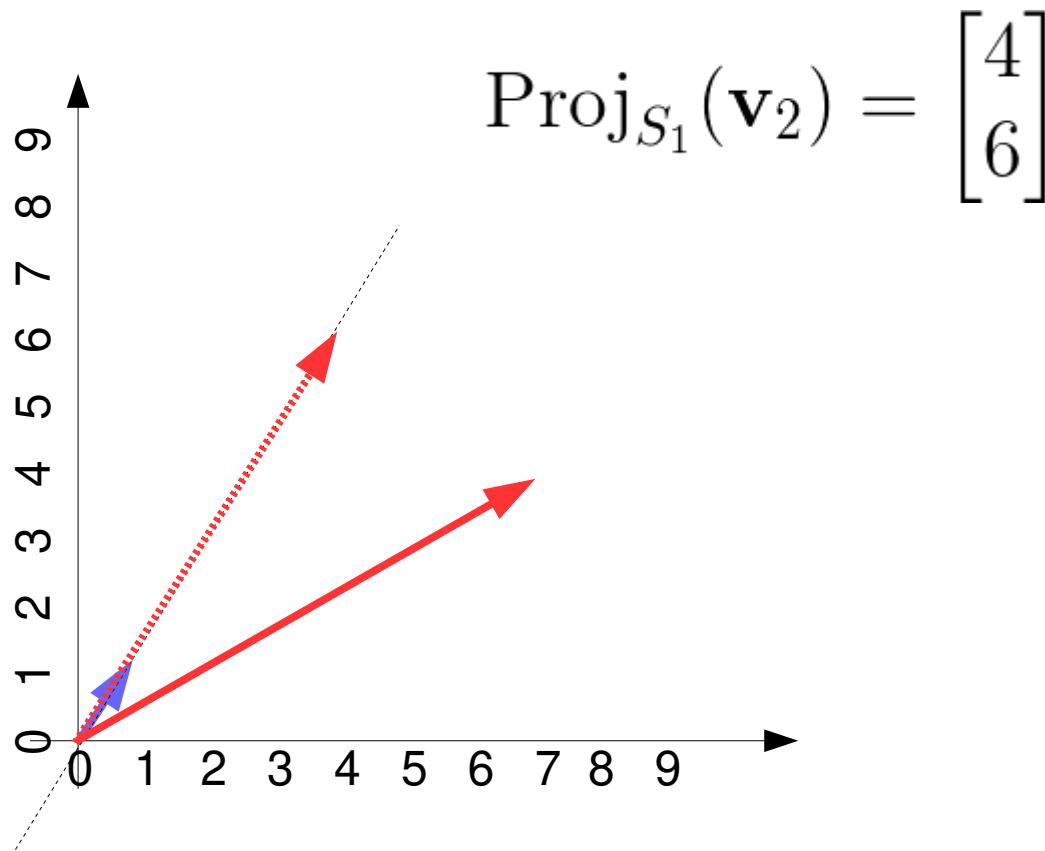
Gram-Schmidt process:

- To obtain an orthonormal basis we need to project  $\mathbf{v}_2$  into the space produced by  $\mathbf{u}_1$ , which is  $S_1$



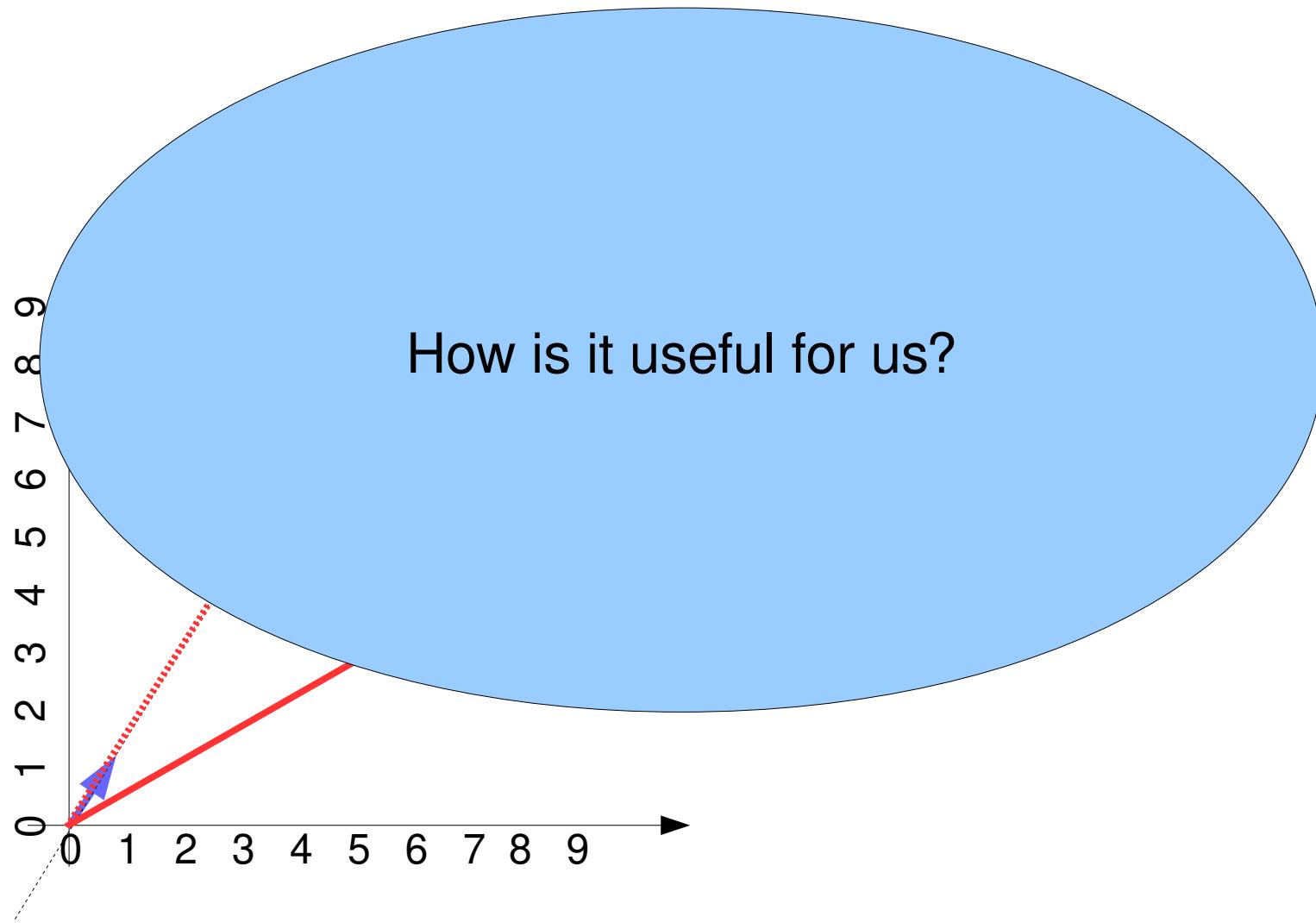
Gram-Schmidt process:

- It means the projection below:



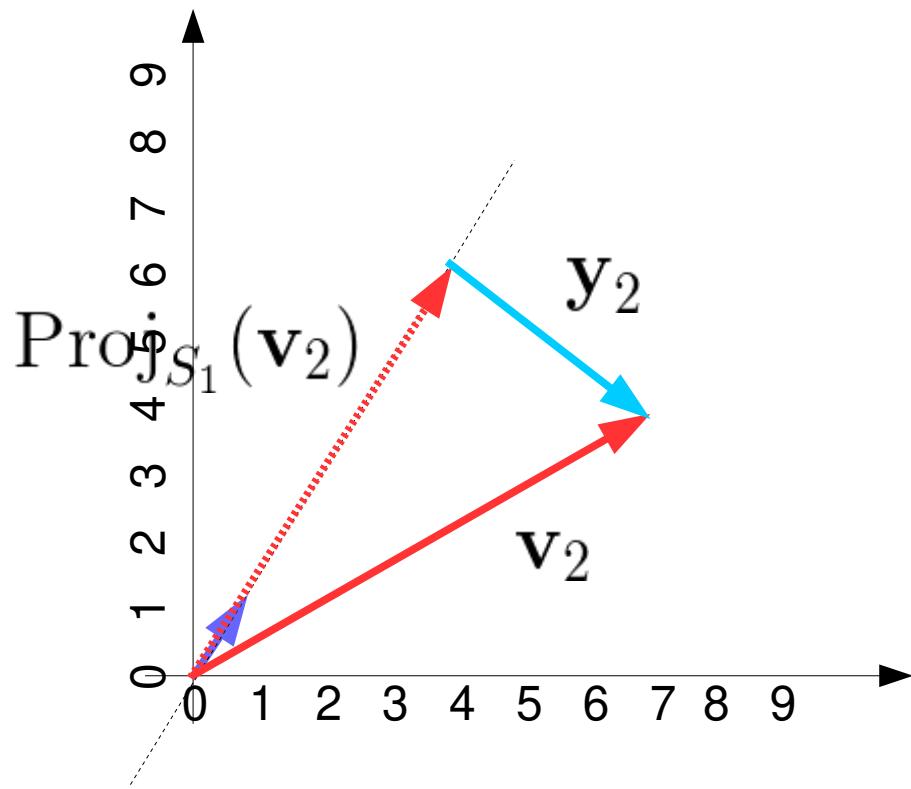
Gram-Schmidt process:

- It means the projection below:



Gram-Schmidt process:

- Observe we can sum the projection to some vector  $\mathbf{y}_2$  to obtain  $\mathbf{v}_2$
- Observe  $\mathbf{y}_2$  is orthogonal to  $\mathbf{u}_1$  as we desire!



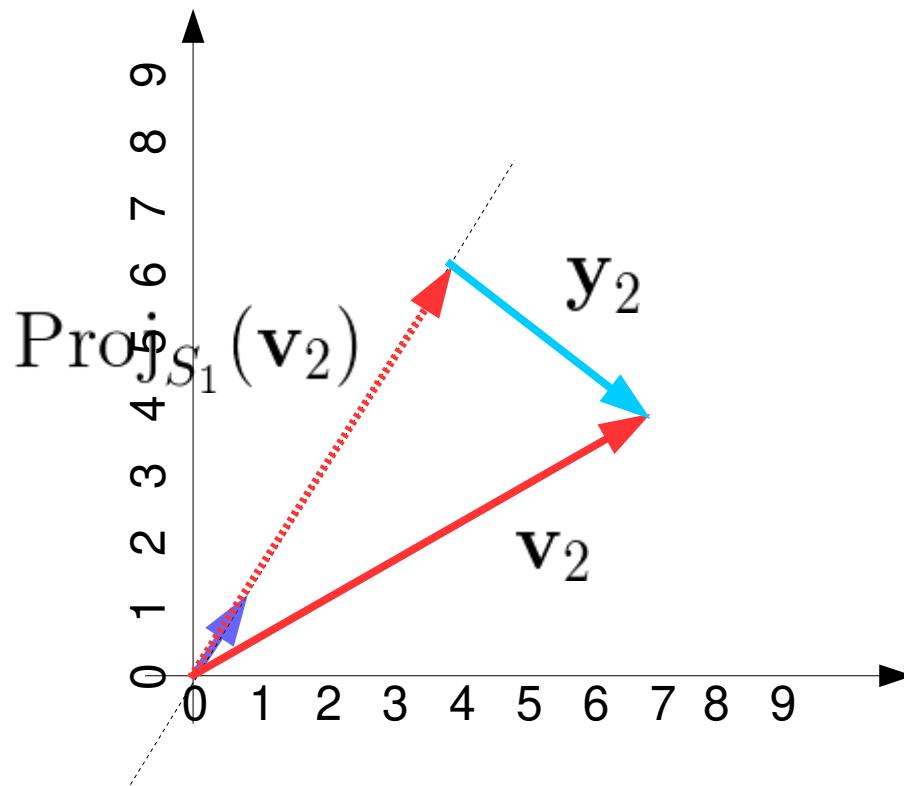
$$\mathbf{v}_2 = \mathbf{y}_2 + \text{Proj}_{S_1}(\mathbf{v}_2)$$

$$\mathbf{y}_2 = \mathbf{v}_2 - \text{Proj}_{S_1}(\mathbf{v}_2)$$

Gram-Schmidt process:

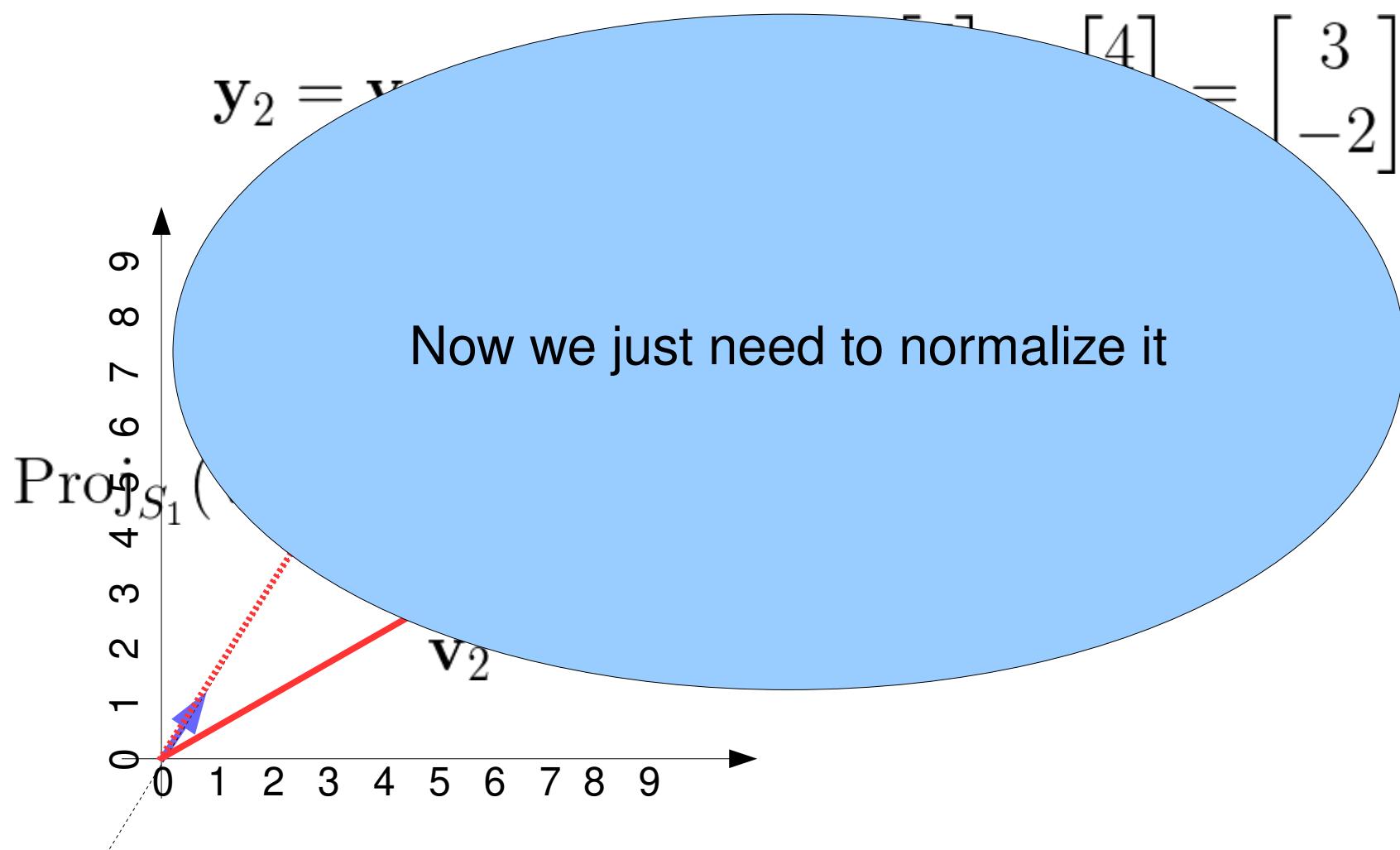
- Then we found it as follows:

$$\mathbf{y}_2 = \mathbf{v}_2 - \text{Proj}_{S_1}(\mathbf{v}_2) = \begin{bmatrix} 7 \\ 4 \end{bmatrix} - \begin{bmatrix} 4 \\ 6 \end{bmatrix} = \begin{bmatrix} 3 \\ -2 \end{bmatrix}$$



Gram-Schmidt process:

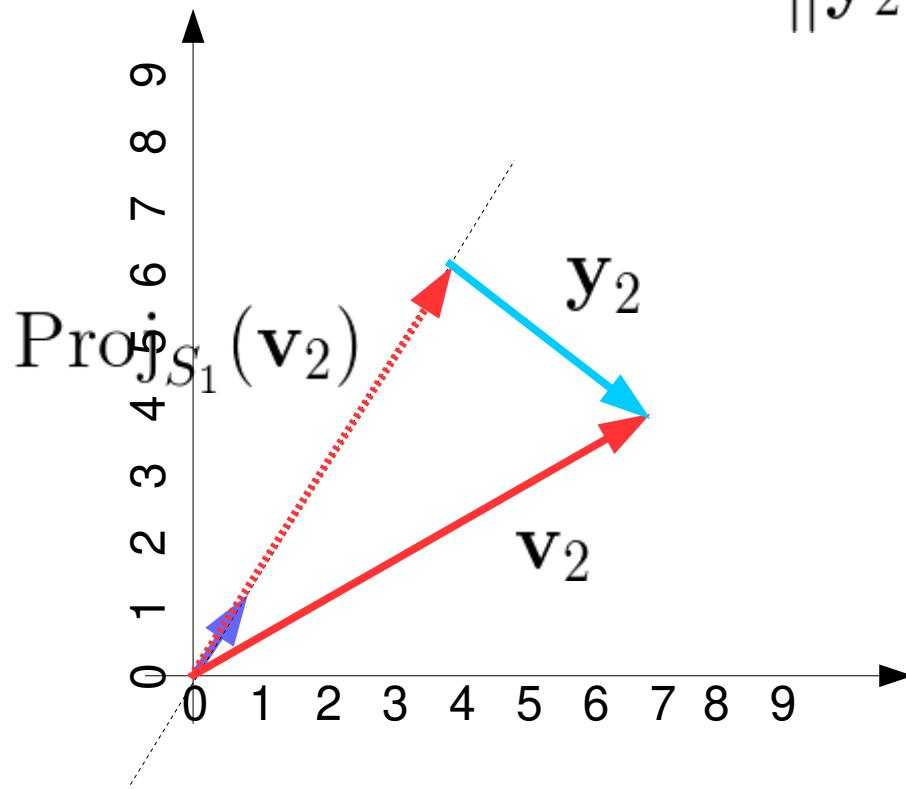
- Then we found it as follows:



Gram-Schmidt process:

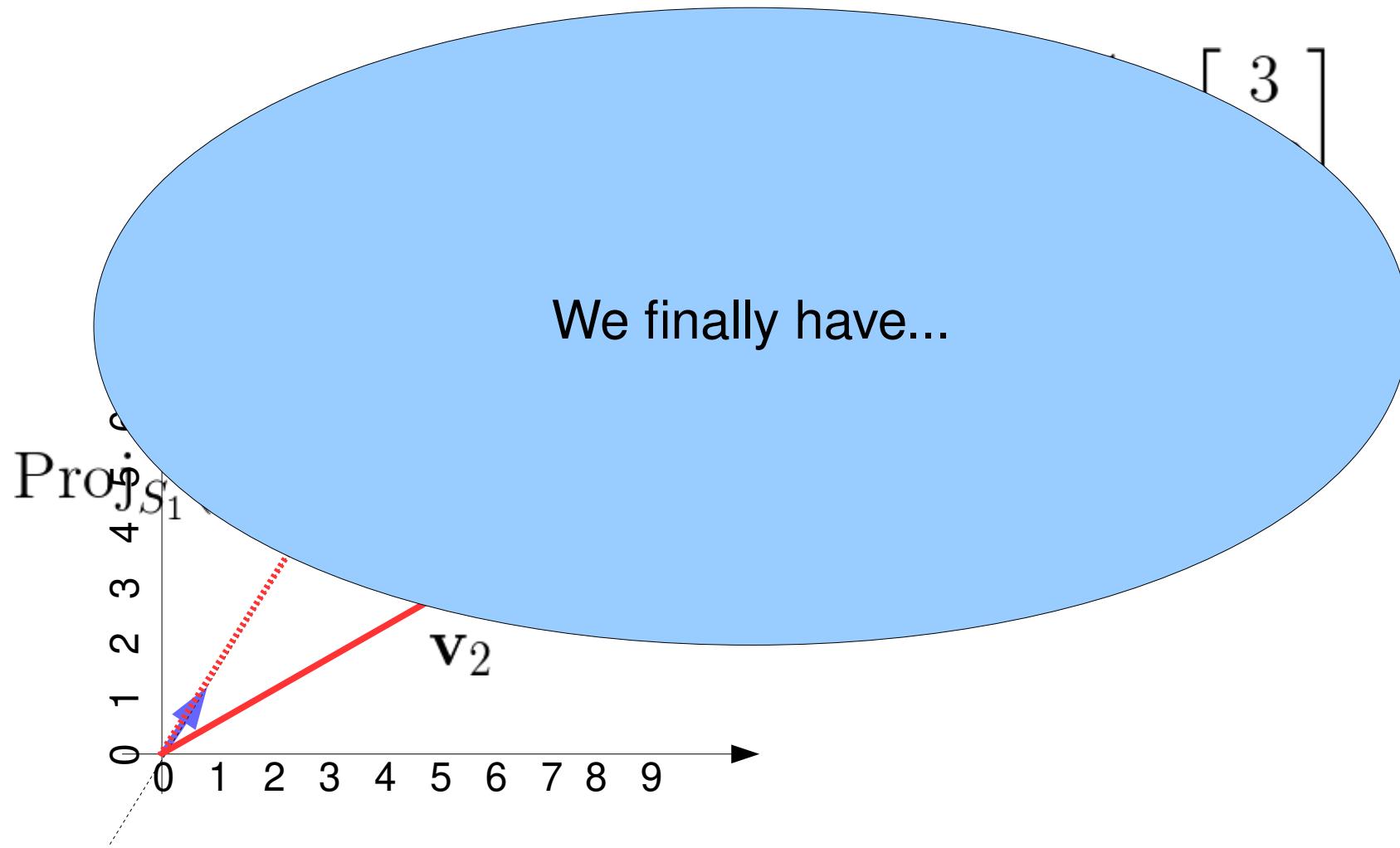
- Normalizing it:

$$\mathbf{u}_2 = \frac{1}{\|\mathbf{y}_2\|} \begin{bmatrix} 3 \\ -2 \end{bmatrix} = \frac{1}{\sqrt{13}} \begin{bmatrix} 3 \\ -2 \end{bmatrix}$$



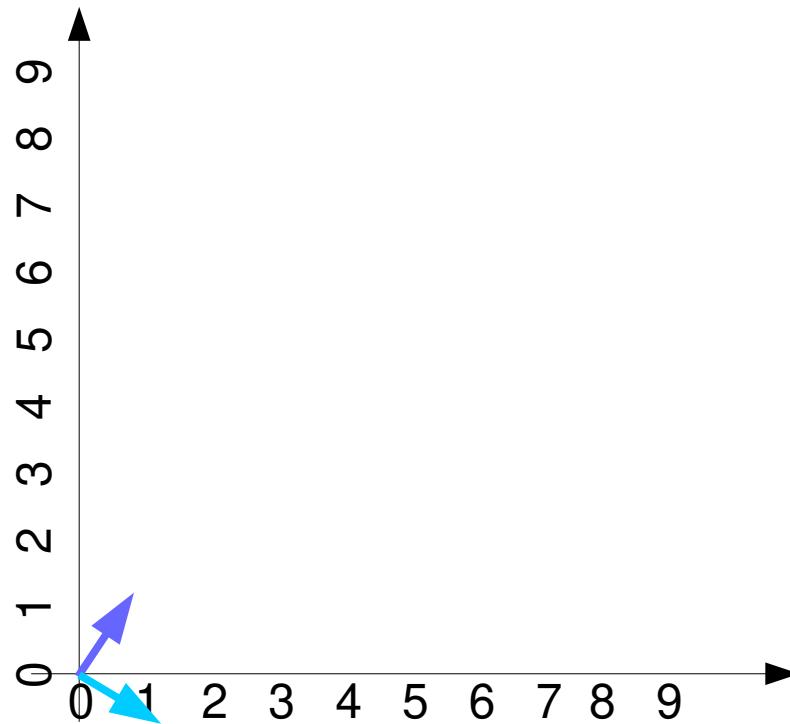
Gram-Schmidt process:

- Normalizing it:



Gram-Schmidt process:

- The orthonormal basis



Gram-Schmidt process:

- The orthonormal basis:

$$B_2 = \{\mathbf{u}_1, \mathbf{u}_2\}$$

- And we can say:

$$\text{span}(\mathbf{v}_1, \mathbf{v}_2) = \text{span}(\mathbf{u}_1, \mathbf{u}_2)$$

- However, now, the basis is simpler to work with

# Orthonormal Basis

- This is the Gram-Schmidt process to find orthonormal basis for a space
  - See more at:
    - [https://pt.khanacademy.org/math/linear-algebra/alternate\\_bases/orthonormal\\_basis/v/linear-algebra-the-gram-schmidt-process](https://pt.khanacademy.org/math/linear-algebra/alternate_bases/orthonormal_basis/v/linear-algebra-the-gram-schmidt-process)
    - [https://pt.khanacademy.org/math/linear-algebra/alternate\\_bases/orthonormal\\_basis/v/linear-algebra-gram-schmidt-process-example](https://pt.khanacademy.org/math/linear-algebra/alternate_bases/orthonormal_basis/v/linear-algebra-gram-schmidt-process-example)

# Orthonormal Basis

- See more at:

- [https://pt.khanacademy.org/math/linear-algebra/alternate\\_bases/orthonormal\\_bases/v/linear-algebra-introduction-to-orthonormal-bases](https://pt.khanacademy.org/math/linear-algebra/alternate_bases/orthonormal_bases/v/linear-algebra-introduction-to-orthonormal-bases)
- [https://pt.khanacademy.org/math/linear-algebra/alternate\\_bases/orthonormal\\_bases/v/linear-algebra-coordinates-with-respect-to-orthonormal-bases](https://pt.khanacademy.org/math/linear-algebra/alternate_bases/orthonormal_bases/v/linear-algebra-coordinates-with-respect-to-orthonormal-bases)
- [https://pt.khanacademy.org/math/linear-algebra/alternate\\_bases/orthonormal\\_bases/v/lin-alg-projections-onto-subspaces-with-orthonormal-bases](https://pt.khanacademy.org/math/linear-algebra/alternate_bases/orthonormal_bases/v/lin-alg-projections-onto-subspaces-with-orthonormal-bases)
- [https://pt.khanacademy.org/math/linear-algebra/alternate\\_bases/orthonormal\\_bases/v/lin-alg-finding-projection-onto-subspace-with-orthonormal-basis-example](https://pt.khanacademy.org/math/linear-algebra/alternate_bases/orthonormal_bases/v/lin-alg-finding-projection-onto-subspace-with-orthonormal-basis-example)
- [https://pt.khanacademy.org/math/linear-algebra/alternate\\_bases/orthonormal\\_bases/v/lin-alg-example-using-orthogonal-change-of-basis-matrix-to-find-transformation-matrix](https://pt.khanacademy.org/math/linear-algebra/alternate_bases/orthonormal_bases/v/lin-alg-example-using-orthogonal-change-of-basis-matrix-to-find-transformation-matrix)
- [https://pt.khanacademy.org/math/linear-algebra/alternate\\_bases/orthonormal\\_bases/v/lin-alg-orthogonal-matrices-preserve-angles-and-lengths](https://pt.khanacademy.org/math/linear-algebra/alternate_bases/orthonormal_bases/v/lin-alg-orthogonal-matrices-preserve-angles-and-lengths)
- [https://pt.khanacademy.org/math/linear-algebra/alternate\\_bases/orthonormal\\_bases/v/linear-algebra-the-gram-schmidt-process](https://pt.khanacademy.org/math/linear-algebra/alternate_bases/orthonormal_bases/v/linear-algebra-the-gram-schmidt-process)
- [https://pt.khanacademy.org/math/linear-algebra/alternate\\_bases/orthonormal\\_bases/v/linear-algebra-gram-schmidt-process-example](https://pt.khanacademy.org/math/linear-algebra/alternate_bases/orthonormal_bases/v/linear-algebra-gram-schmidt-process-example)
- Orthogonal transformation preserves vector length and angles – [http://en.wikipedia.org/wiki/Orthogonal\\_transformation](http://en.wikipedia.org/wiki/Orthogonal_transformation)

## 7. Linear Algebra: Eigenvalues and Eigenvectors

# Eigenvalues and Eigenvectors

- Consider the linear transformation given by matrix A:

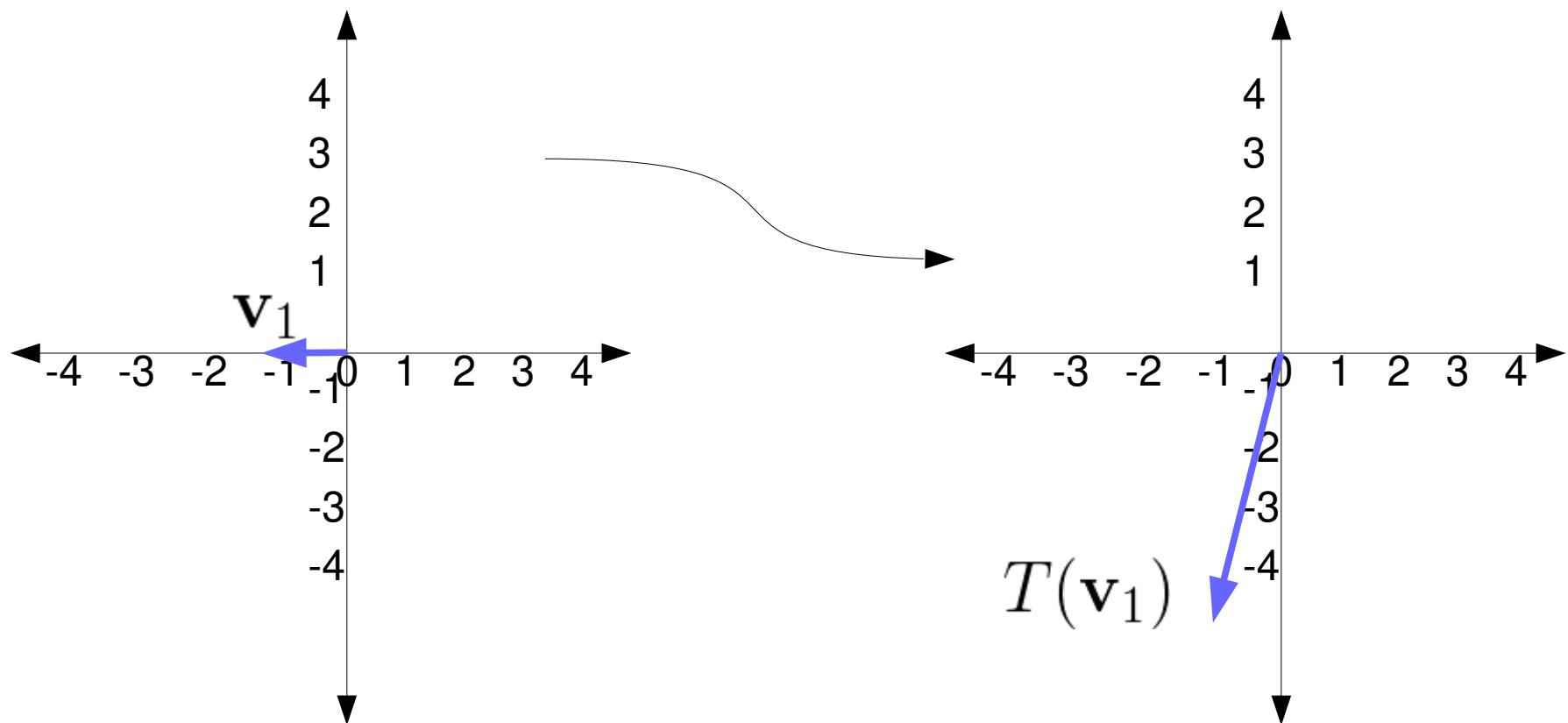
$$T(\mathbf{x}) = A\mathbf{x}$$
$$A = \begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix}$$

- What is the result for any vector in  $\mathbb{R}^2$  when we apply such transformation?

# Eigenvalues and Eigenvectors

- As example, consider some vectors:

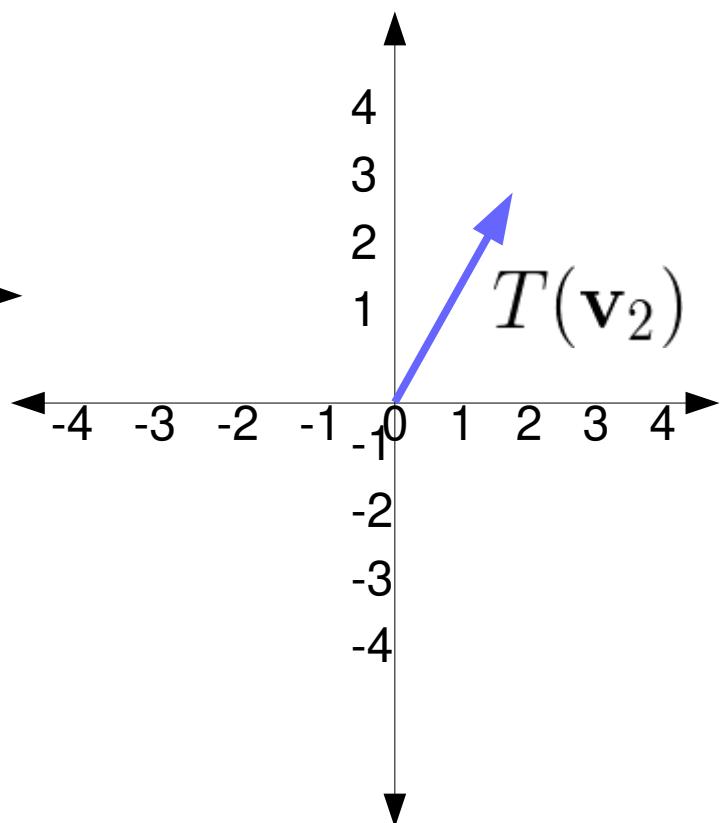
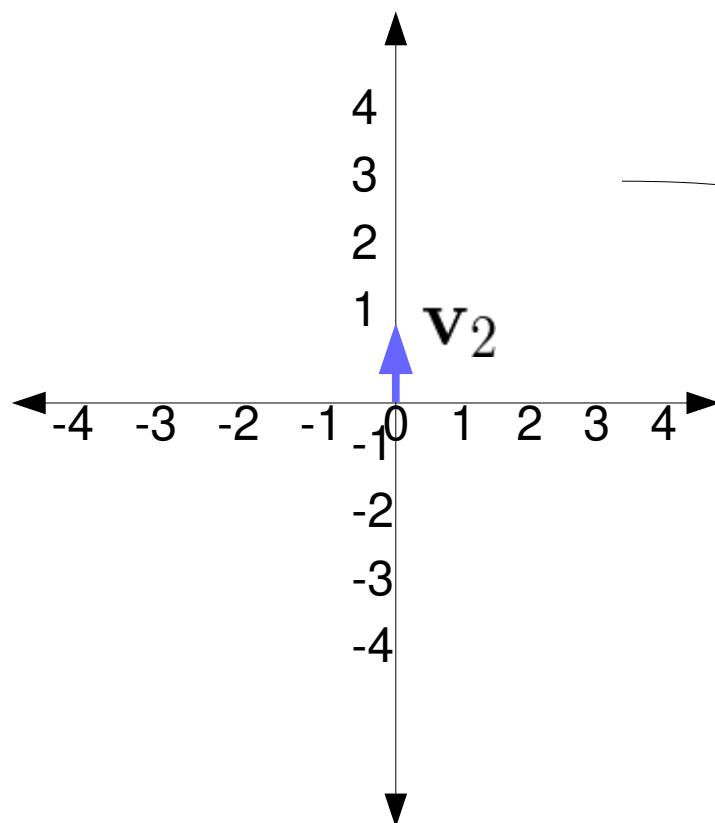
$$\mathbf{v}_1 = \begin{bmatrix} -1 \\ 0 \end{bmatrix}; \quad \mathbf{v}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}; \quad \mathbf{v}_3 = \begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}; \quad \mathbf{v}_4 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$



# Eigenvalues and Eigenvectors

- As example, consider some vectors:

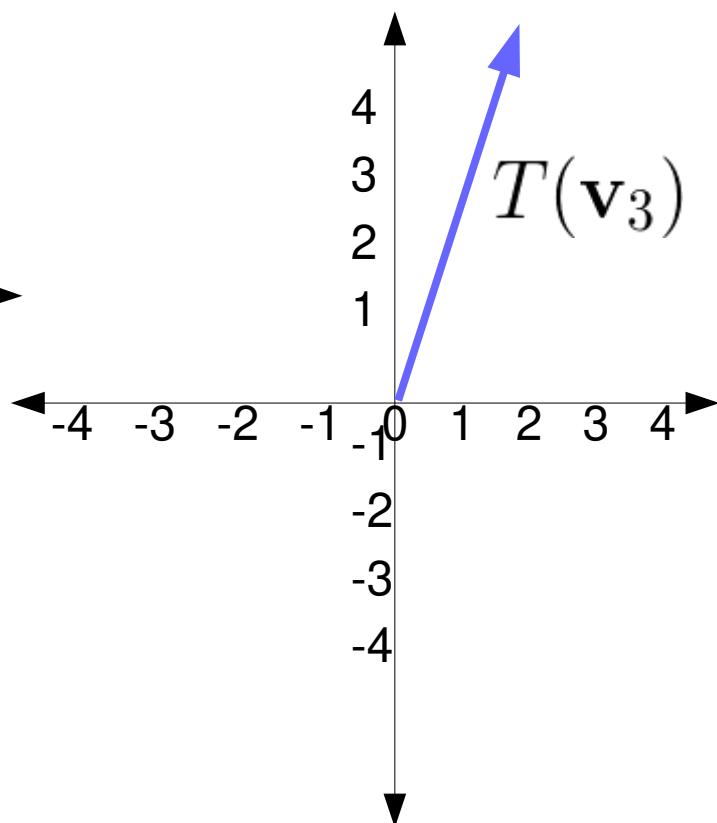
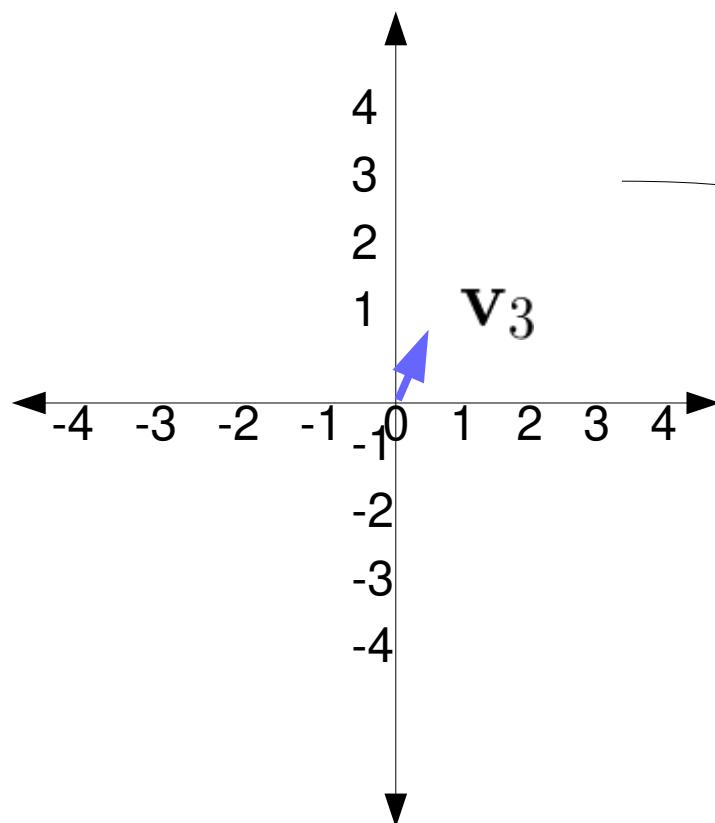
$$\mathbf{v}_1 = \begin{bmatrix} -1 \\ 0 \end{bmatrix}; \quad \mathbf{v}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}; \quad \mathbf{v}_3 = \begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}; \quad \mathbf{v}_4 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$



# Eigenvalues and Eigenvectors

- As example, consider some vectors:

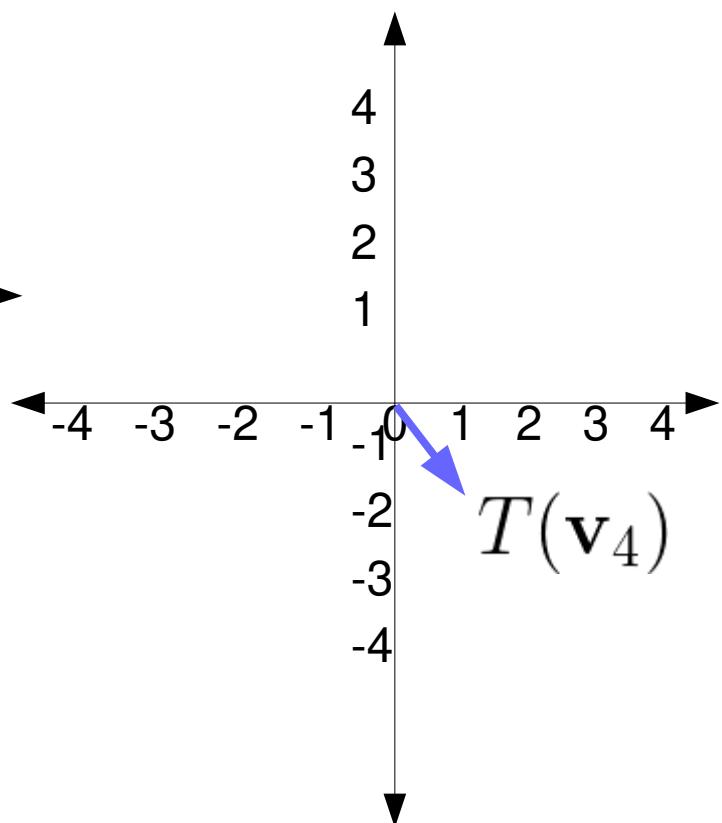
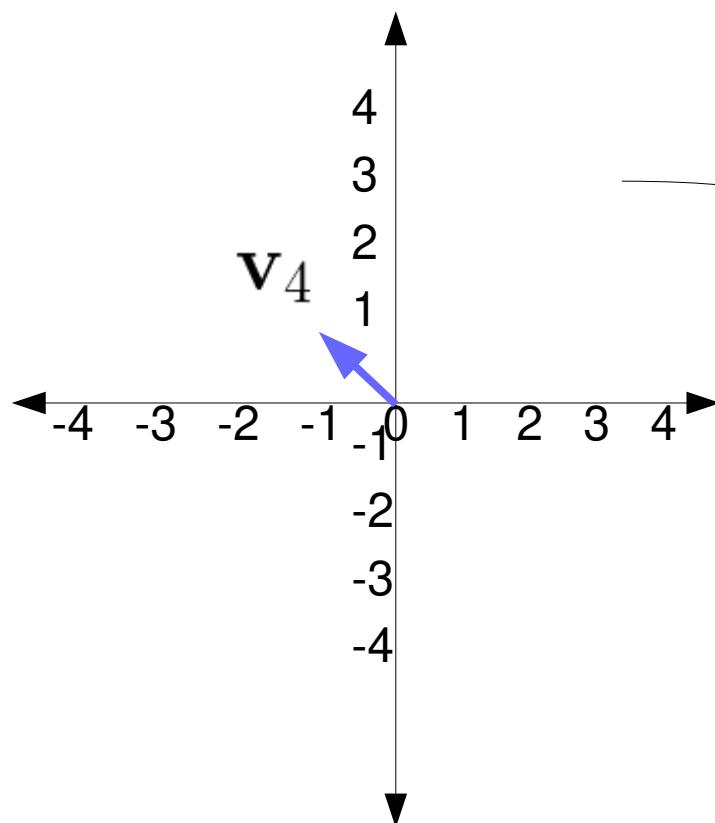
$$\mathbf{v}_1 = \begin{bmatrix} -1 \\ 0 \end{bmatrix}; \quad \mathbf{v}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}; \quad \mathbf{v}_3 = \begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}; \quad \mathbf{v}_4 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$



# Eigenvalues and Eigenvectors

- As example, consider some vectors:

$$\mathbf{v}_1 = \begin{bmatrix} -1 \\ 0 \end{bmatrix}; \quad \mathbf{v}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}; \quad \mathbf{v}_3 = \begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}; \quad \mathbf{v}_4 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

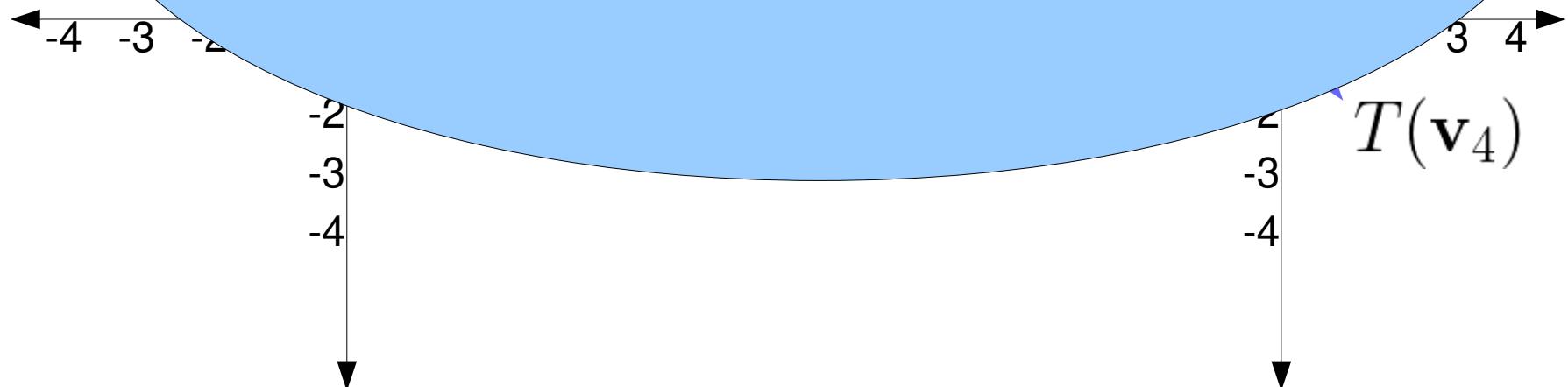


# Eigenvalues and Eigenvectors

- As example, consider some vectors:

$$\mathbf{v}_1 = \begin{bmatrix} -1 \\ 0 \end{bmatrix}; \mathbf{v}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}; \mathbf{v}_3 = \begin{bmatrix} \frac{1}{2} \\ 0 \end{bmatrix}; \mathbf{v}_4 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

Observe vectors change after the transformation

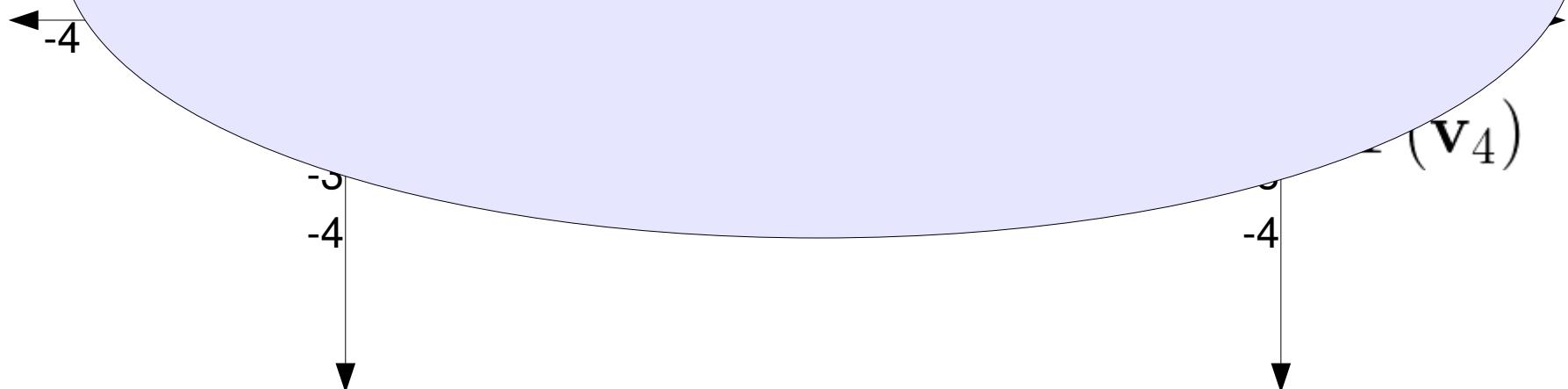


# Eigenvalues and Eigenvectors

- As example, consider some vectors:

$$\mathbf{v}_1 = \begin{bmatrix} -1 \\ 0 \end{bmatrix}; \mathbf{v}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}; \mathbf{v}_3 = \begin{bmatrix} \frac{1}{2} \\ 0 \end{bmatrix}; \mathbf{v}_4 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

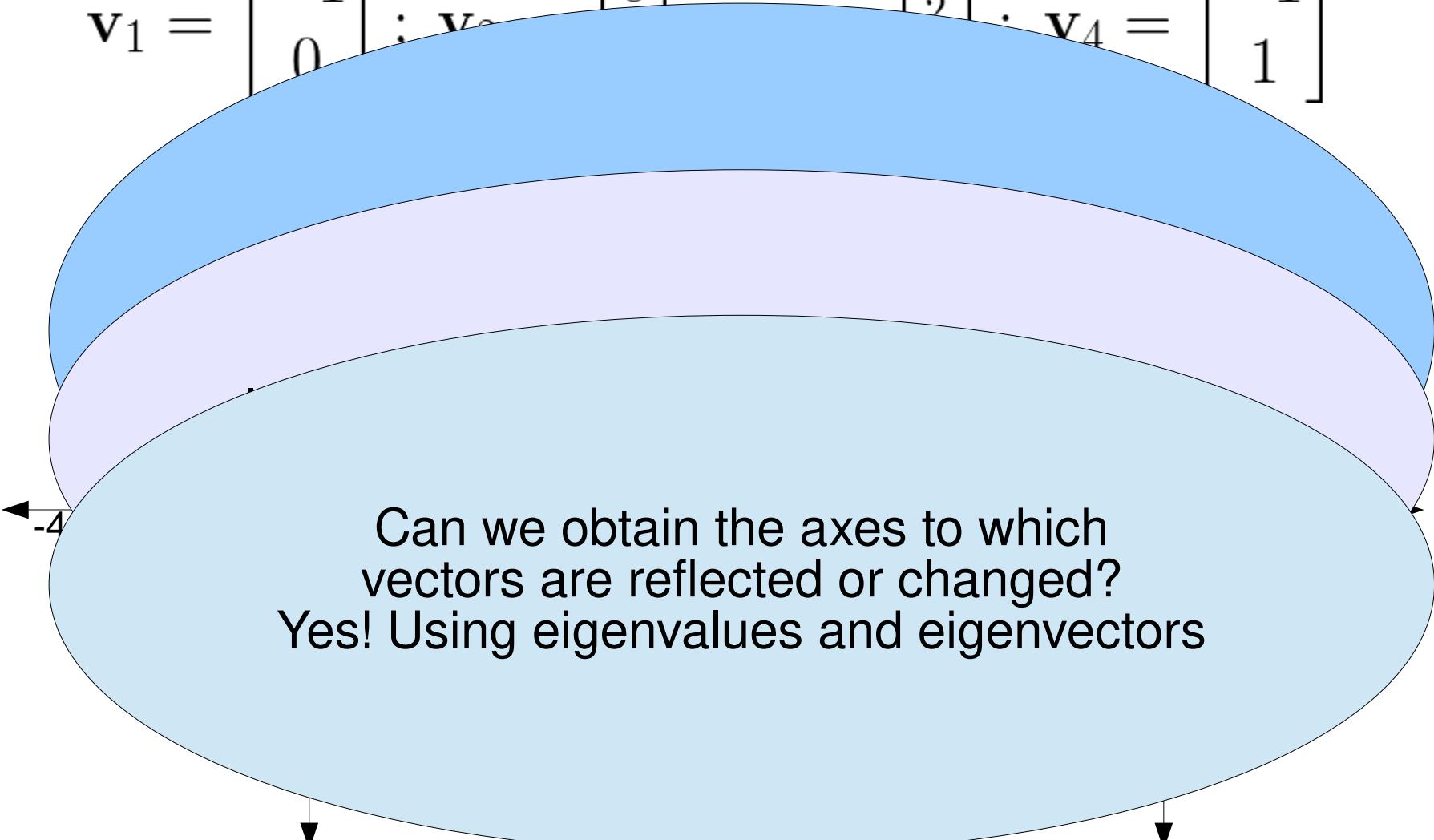
In one of the cases we could see something like a reflection of the vector



# Eigenvalues and Eigenvectors

- As example, consider some vectors:

$$\mathbf{v}_1 = \begin{bmatrix} -1 \\ 0 \end{bmatrix}; \mathbf{v}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}; \mathbf{v}_3 = \begin{bmatrix} \frac{1}{2} \\ 0 \end{bmatrix}; \mathbf{v}_4 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

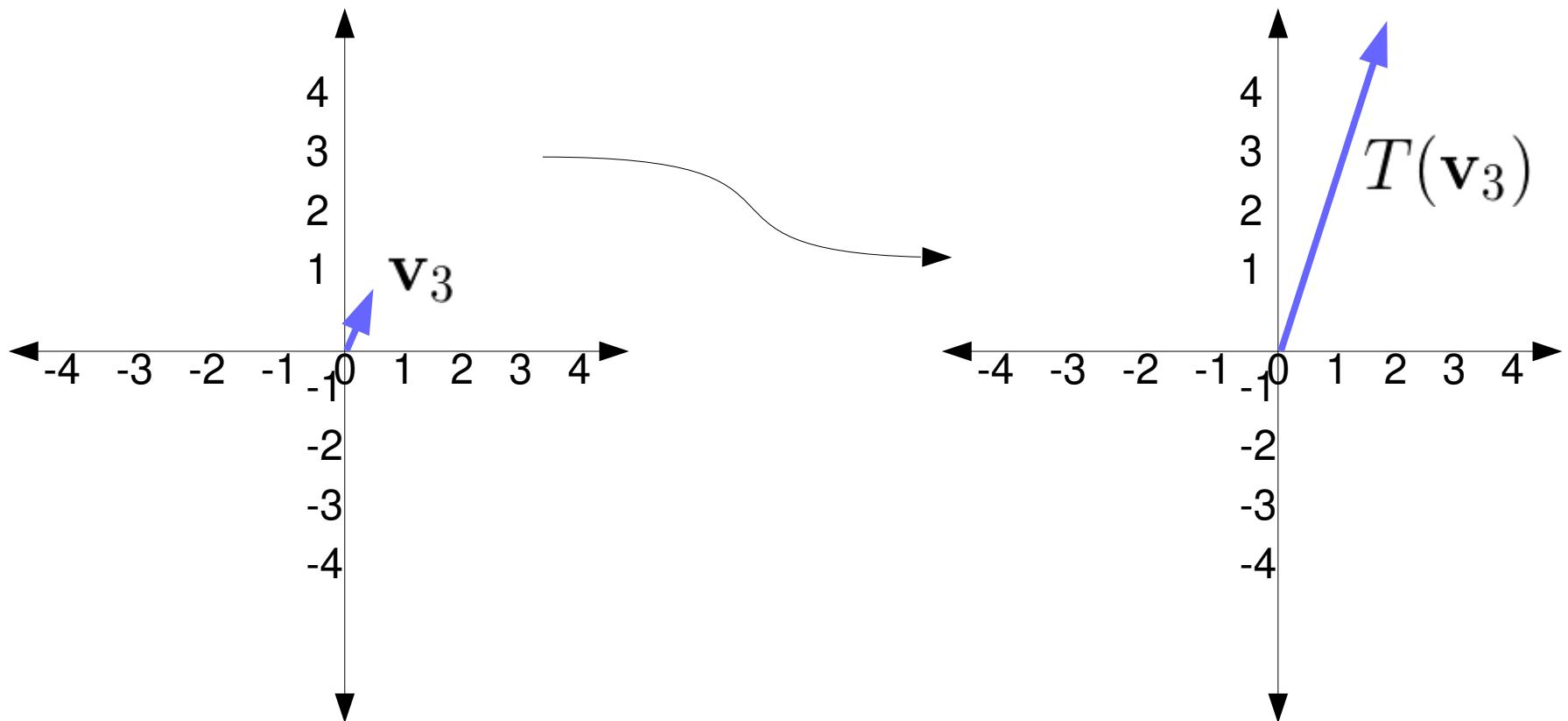


Can we obtain the axes to which  
vectors are reflected or changed?  
Yes! Using eigenvalues and eigenvectors

# Eigenvalues and Eigenvectors

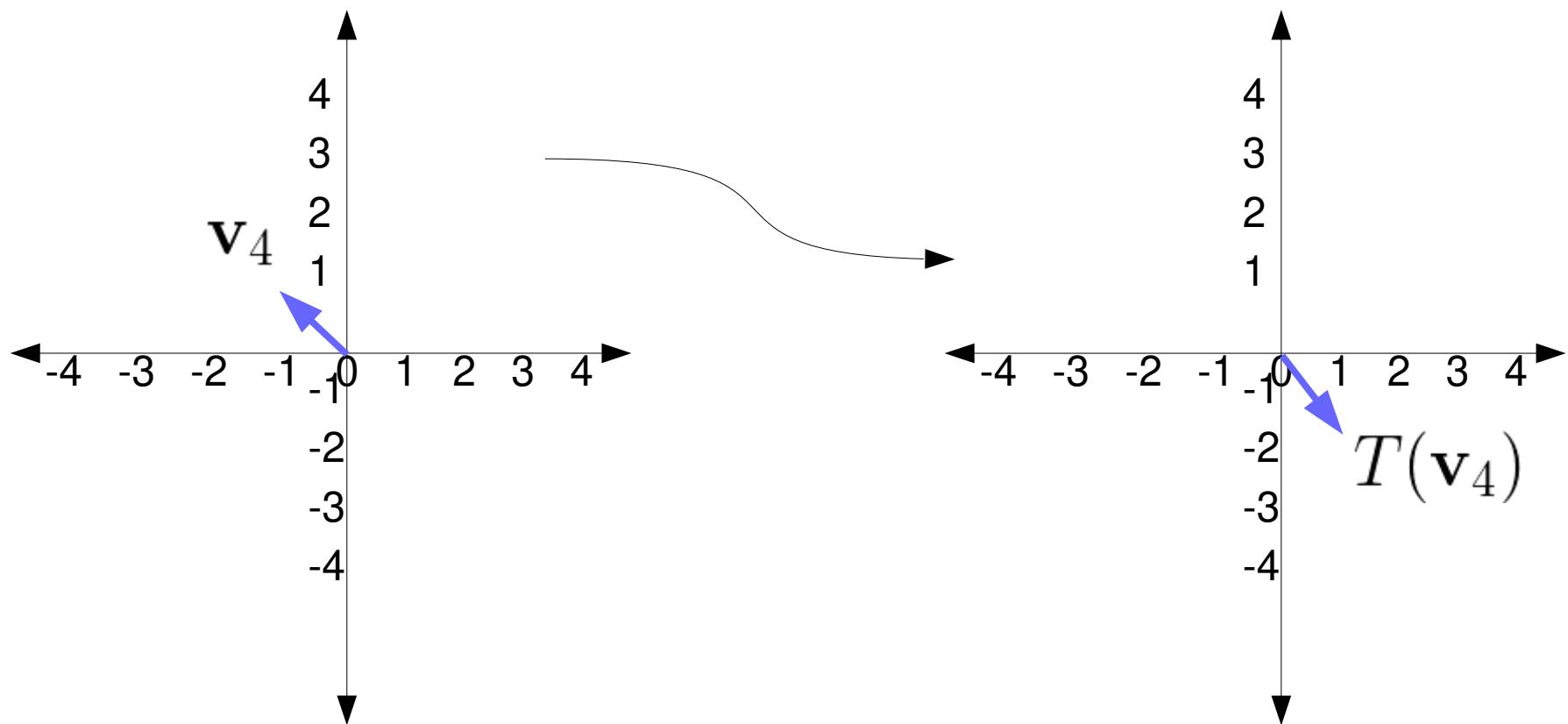
- **What is an eigenvector?**

- It is the vector to which the linear transformation may change the vector direction and magnitude only!
- Example:



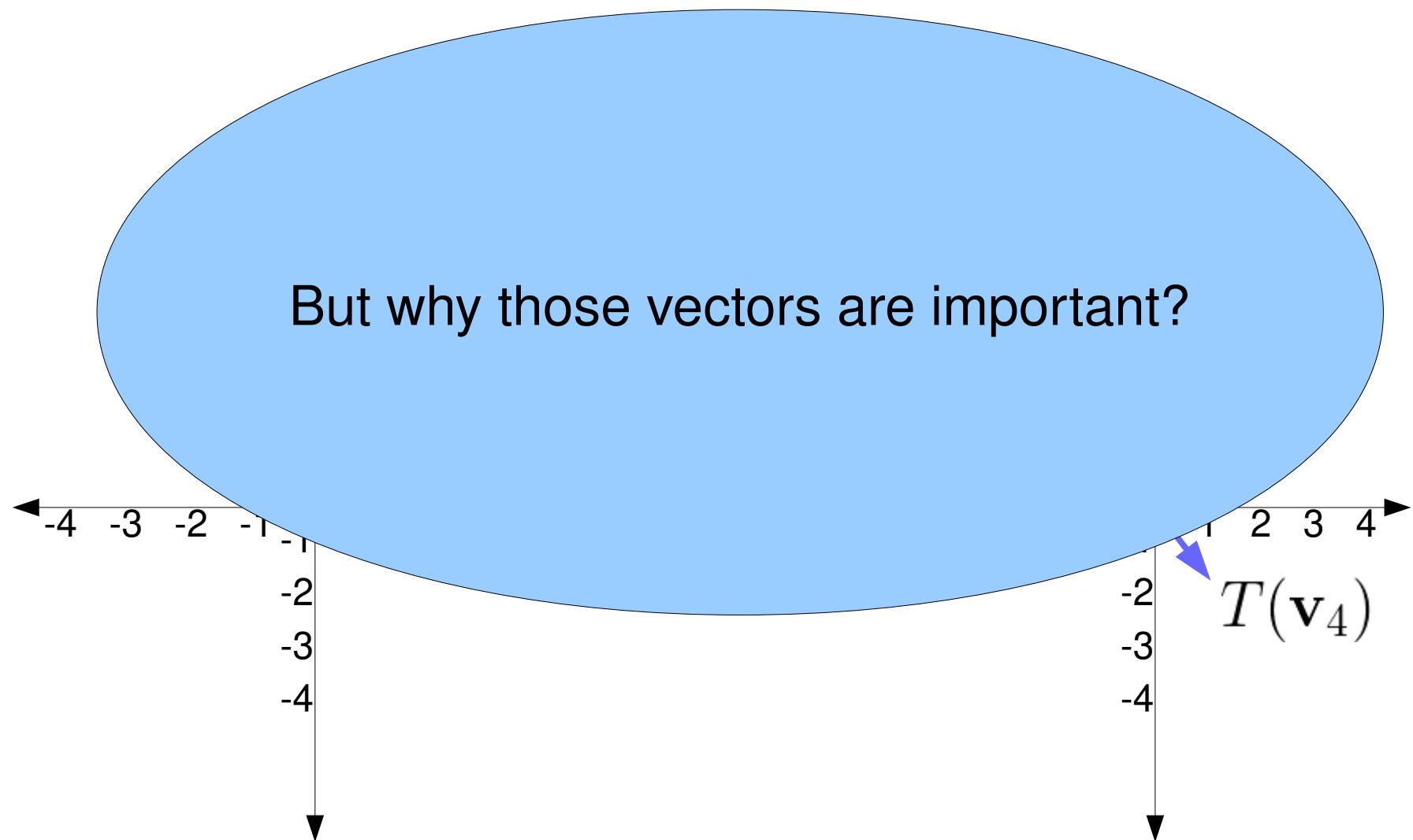
# Eigenvalues and Eigenvectors

- Another example:
  - Changing direction



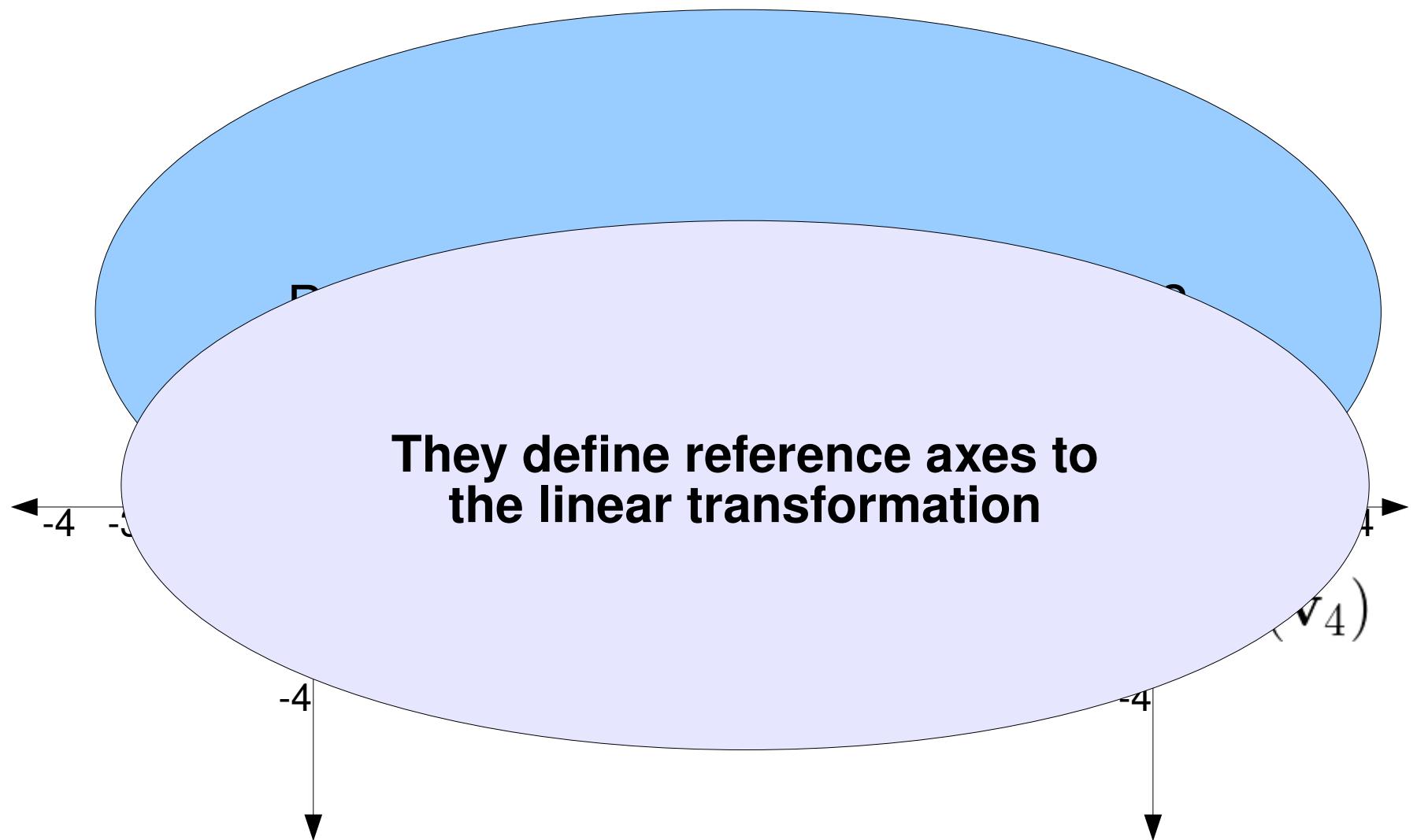
# Eigenvalues and Eigenvectors

- Another example:
  - Changing direction



# Eigenvalues and Eigenvectors

- Another example:
  - Changing direction



# Eigenvalues and Eigenvectors

- In this case we have two eigenvectors:

$$\mathbf{v}_3 = \begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}; \quad \mathbf{v}_4 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

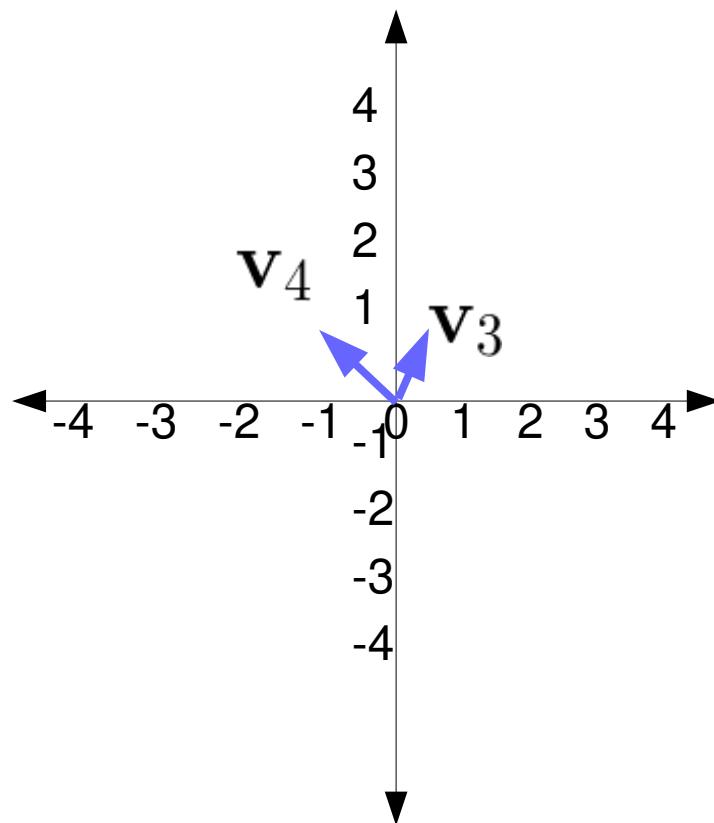
- Let's plot them...

# Eigenvalues and Eigenvectors

- In this case we have two eigenvectors:

$$\mathbf{v}_3 = \begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}; \quad \mathbf{v}_4 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

- Let's plot them...



# Eigenvalues and Eigenvectors

- In this case we have two eigenvectors:

$$\mathbf{v}_3 = \begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}; \quad \mathbf{v}_4 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

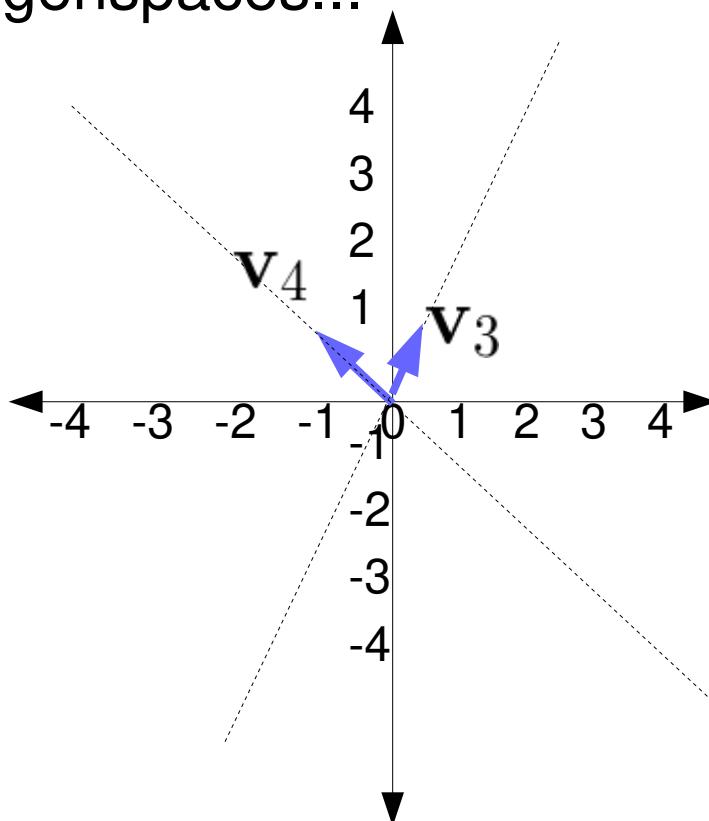
- Let's plot their eigenspaces, i.e., the spaces produced by each one of them

# Eigenvalues and Eigenvectors

- In this case we have two eigenvectors:

$$\mathbf{v}_3 = \begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}; \quad \mathbf{v}_4 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

- Let's plot their eigenspaces...



# Eigenvalues and Eigenvectors

- In this case we have two eigenvectors:

$$\mathbf{v}_3 = \begin{bmatrix} \frac{1}{2} \\ 2 \end{bmatrix}, \quad \mathbf{v}_4 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

- Let's now

So the linear transformation produce changes in vectors using these axes as references?

Yes!

-4

# Eigenvalues and Eigenvectors

- In this case we have two eigenvectors:

- Let's

$\begin{bmatrix} 1 \\ 1 \end{bmatrix}$

$\begin{bmatrix} -1 \\ 1 \end{bmatrix}$

But what is the change produced?

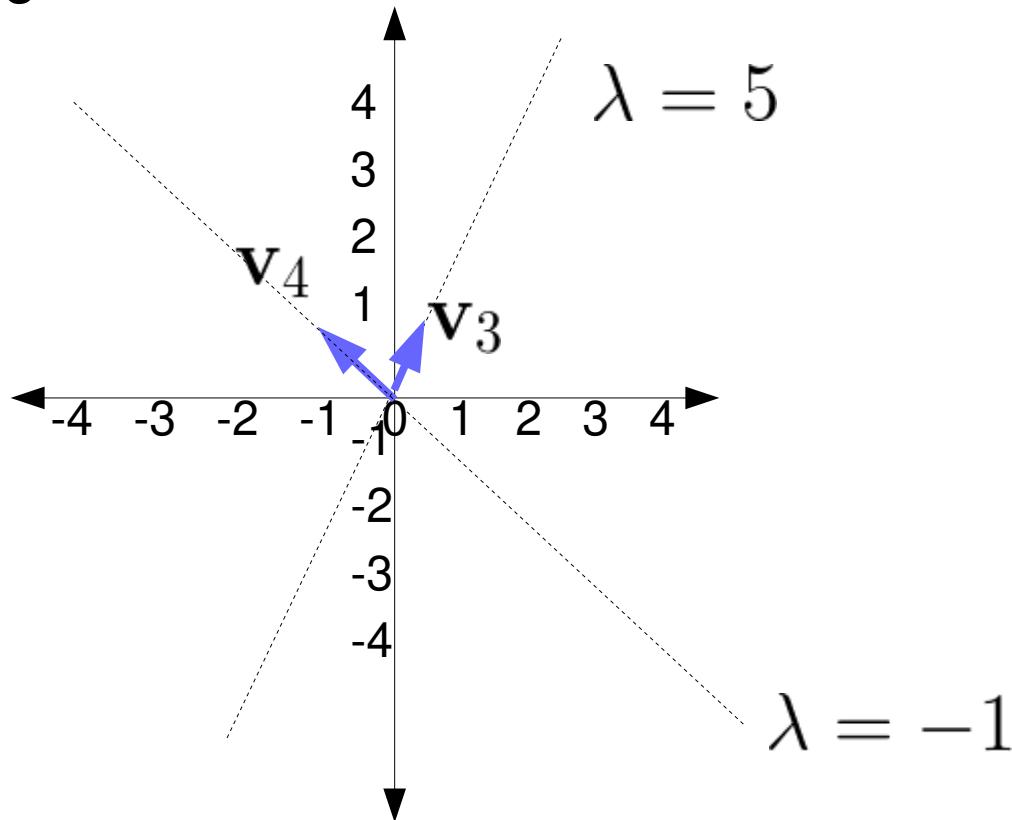
**That is defined by the eigenvalues!**

# Eigenvalues and Eigenvectors

- In this case we have two eigenvectors:

$$\mathbf{v}_3 = \begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}; \quad \mathbf{v}_4 = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

- These are the eigenvalues associated to each one of the eigenvectors...



# Eigenvalues and Eigenvectors

- In this case we have two eigenvectors:

$$\mathbf{v}_3 = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}; \mathbf{v}_4 = \begin{bmatrix} -1 \\ 1 \\ 1 \end{bmatrix}$$

- These are the eigenvectors belonging to the eigenvalue  $\lambda = -1$ .

What does that mean?

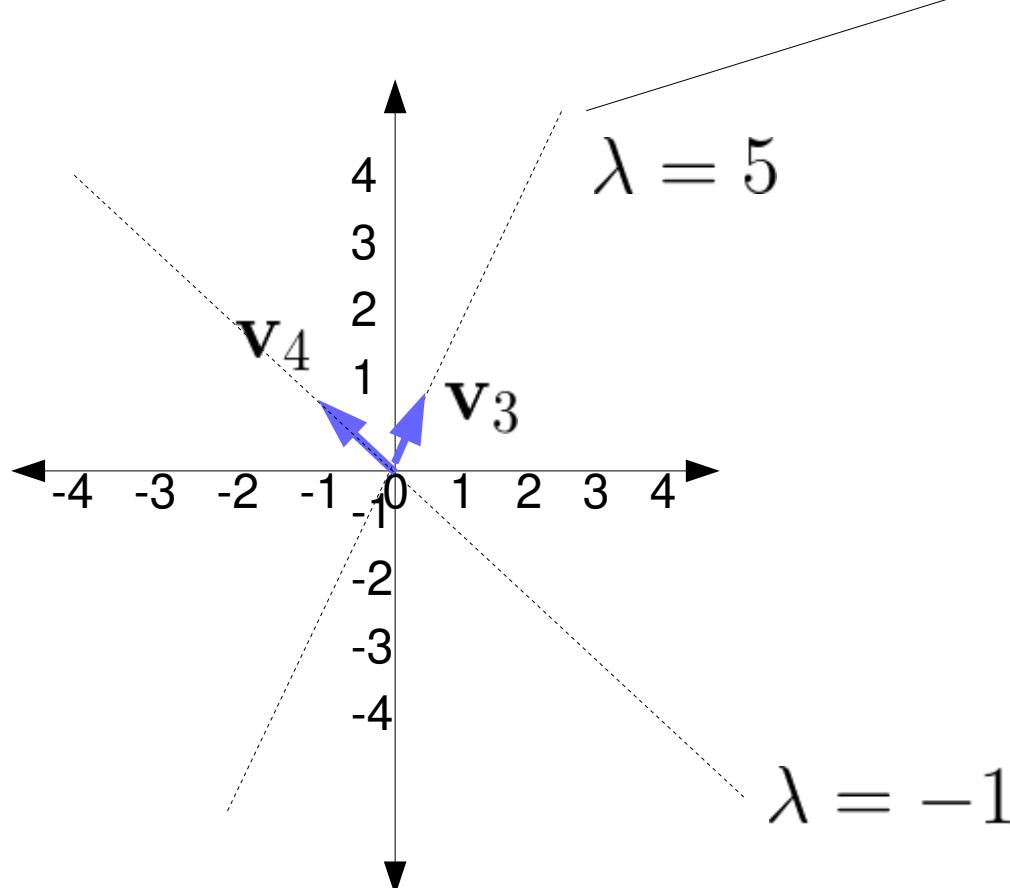
-3

-4

$$\lambda = -1$$

# Eigenvalues and Eigenvectors

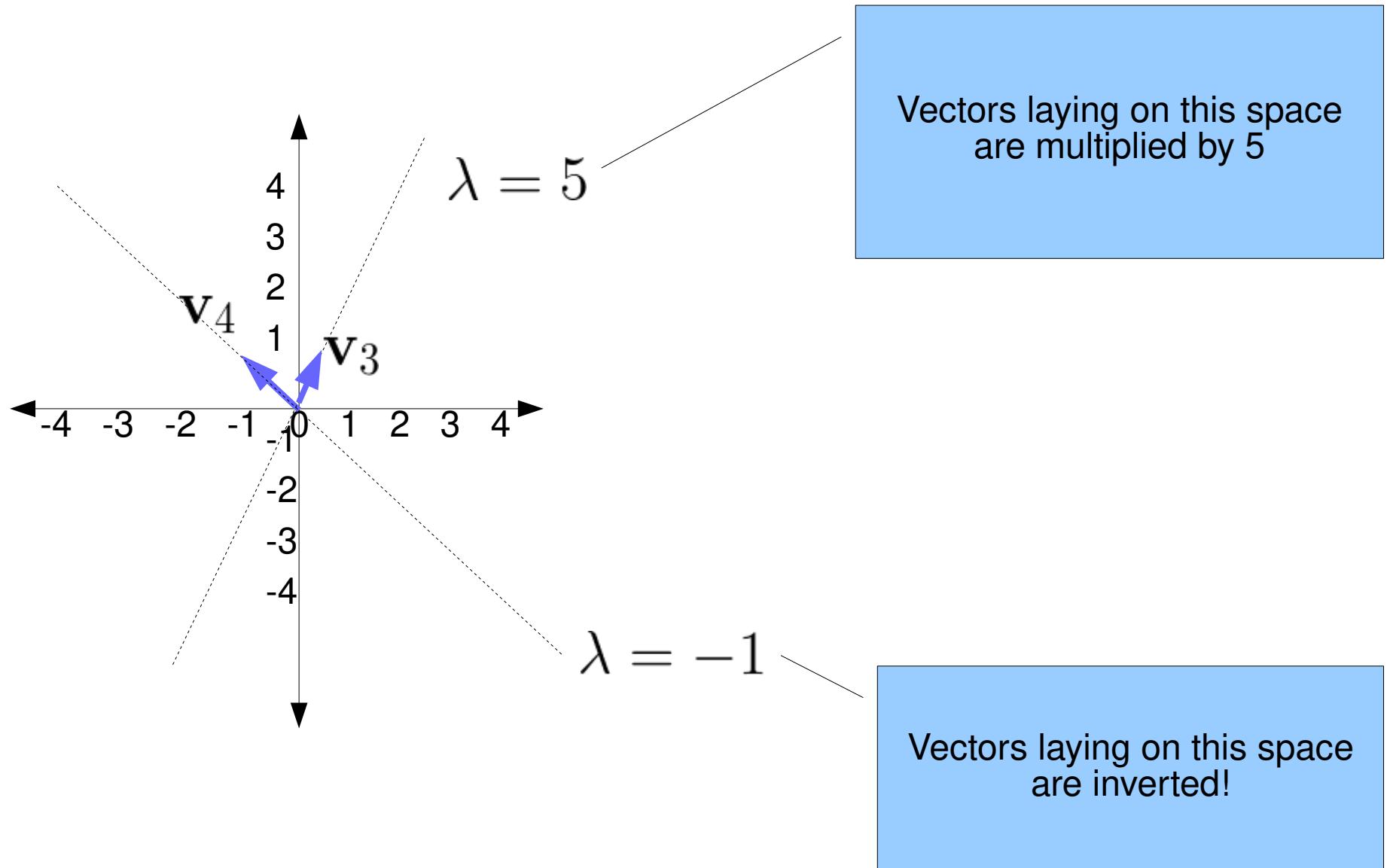
- Meaning:



Vectors laying on this space  
are multiplied by 5

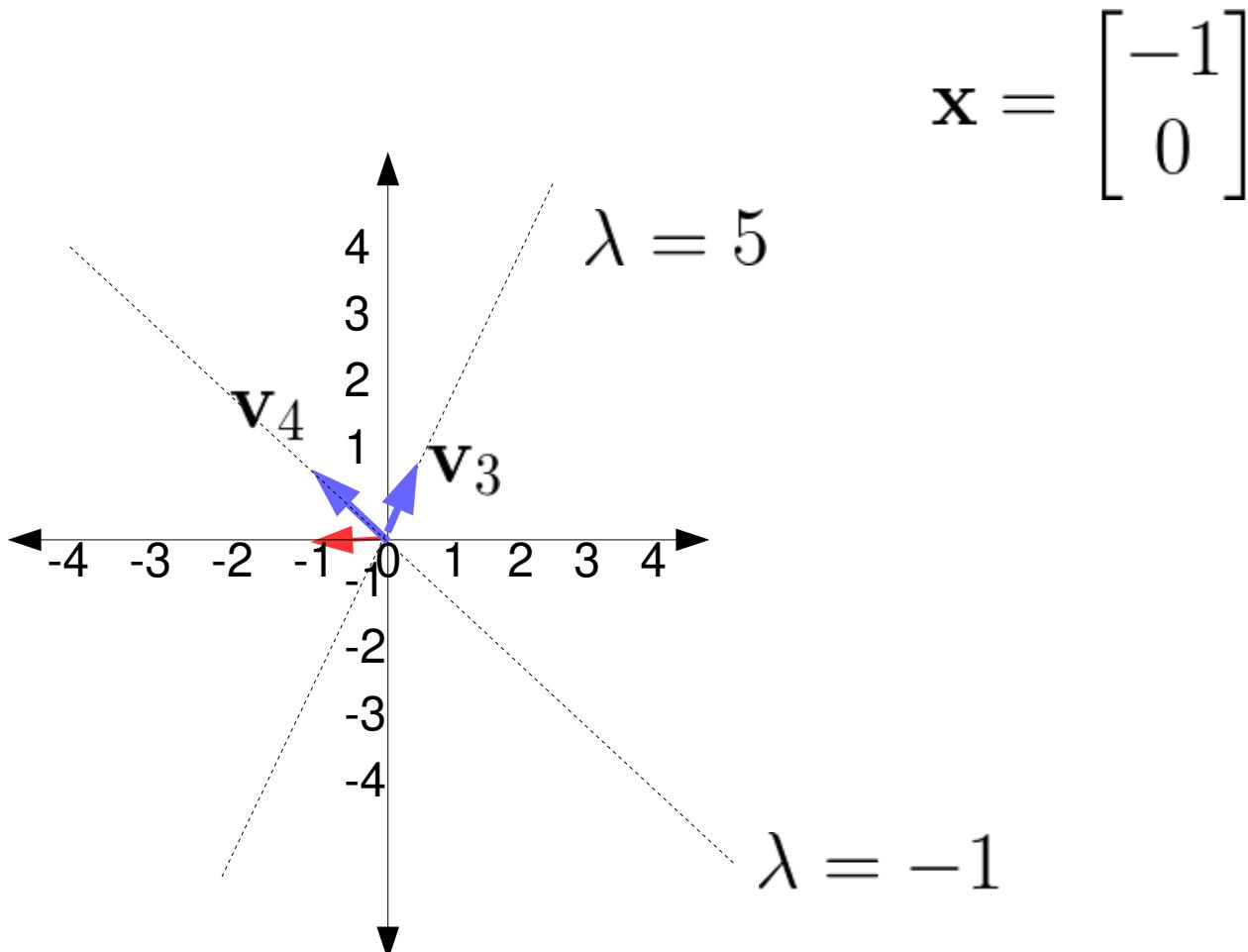
# Eigenvalues and Eigenvectors

- Meaning:



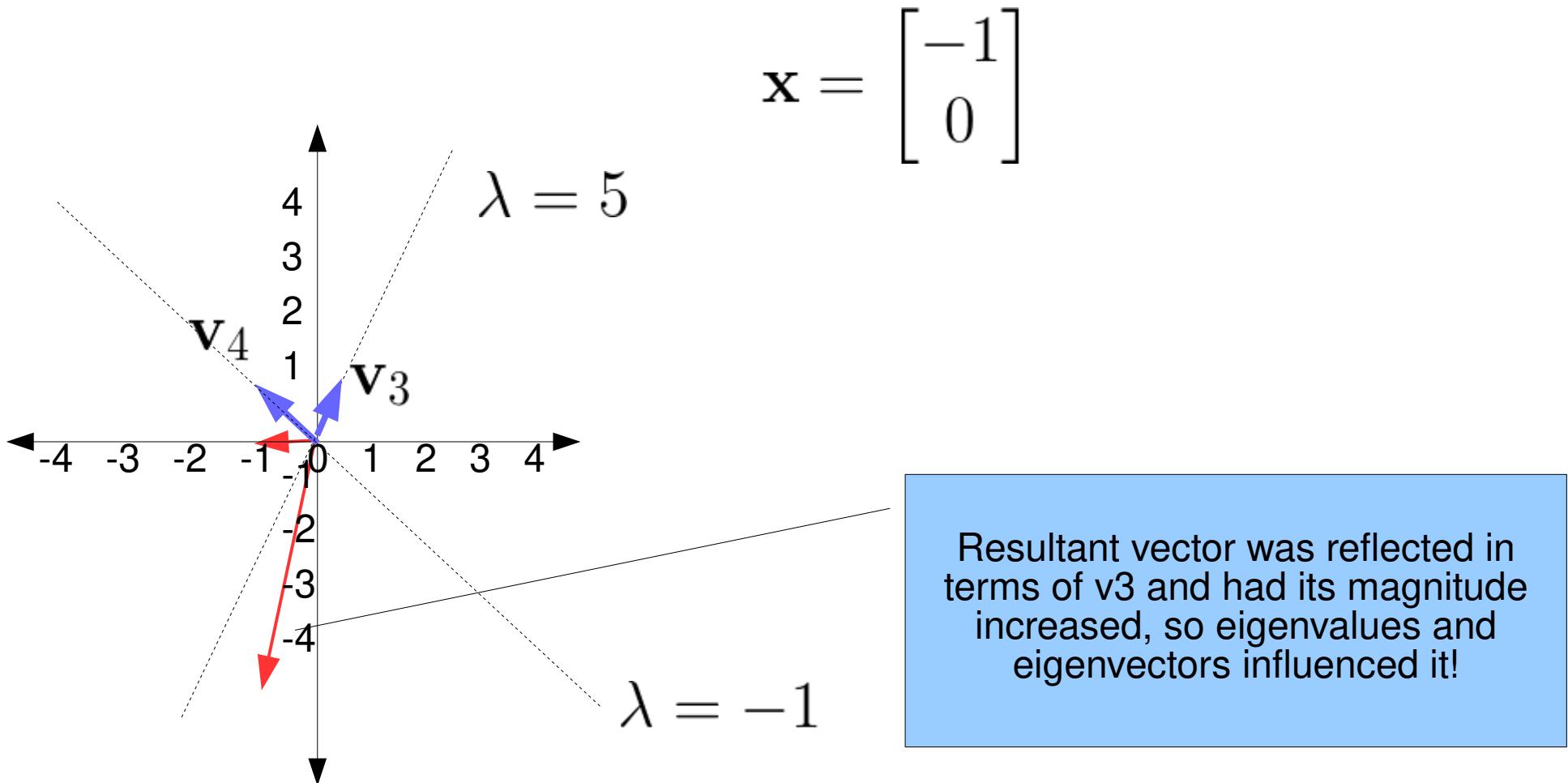
# Eigenvalues and Eigenvectors

- What about this other vector?



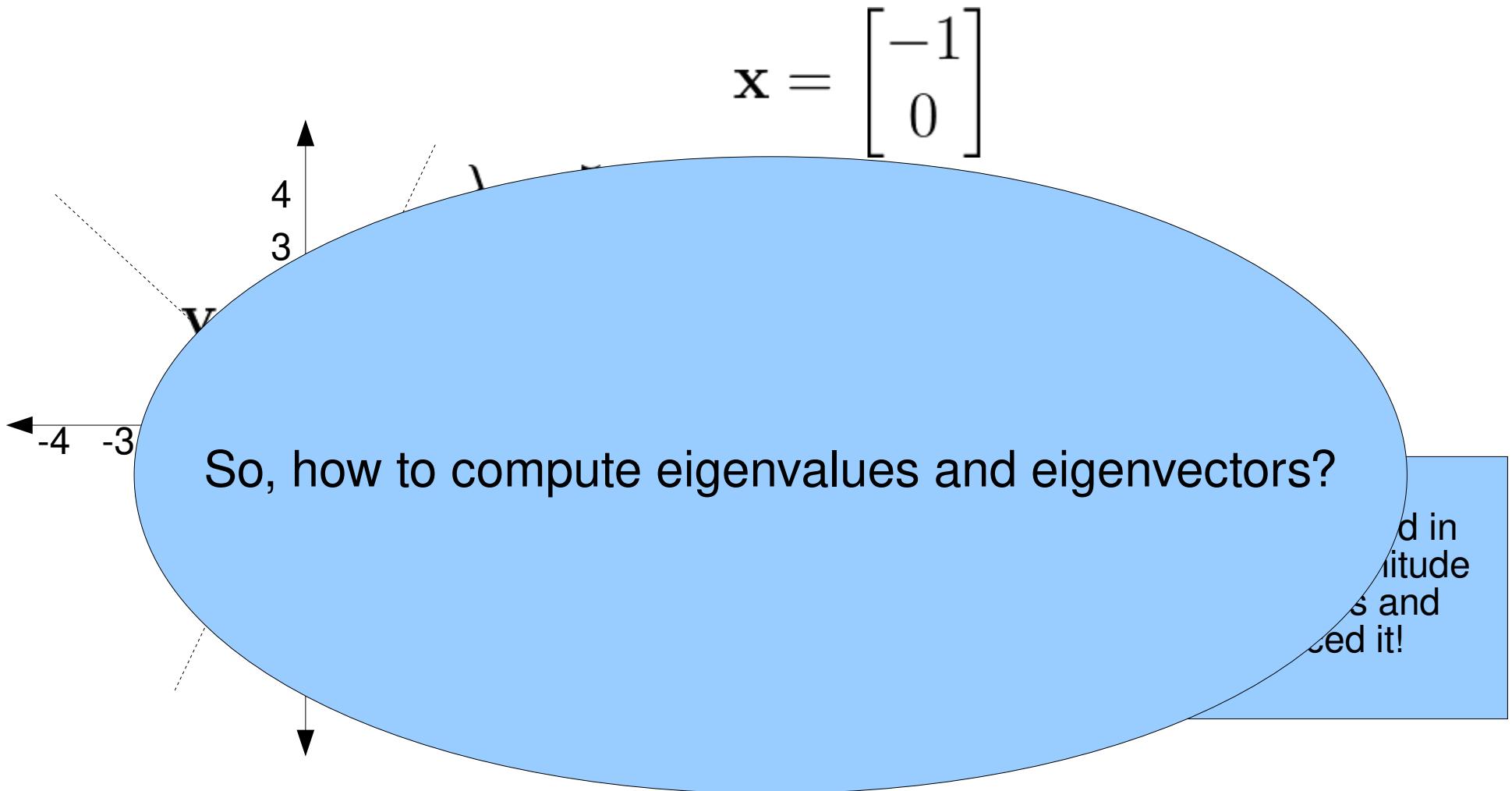
# Eigenvalues and Eigenvectors

- It will be reflected and have its magnitude changed!



# Eigenvalues and Eigenvectors

- It will be reflected and have its magnitude changed!



# Eigenvalues and Eigenvectors

- First of all, as we have seen, eigenvectors are special cases in which the linear transformation may change their directions and magnitudes only, so we can formulate:

$$T(\mathbf{v}) : \mathbb{R}^n \rightarrow \mathbb{R}^n$$

$$T(\mathbf{v}) = A\mathbf{v} = \lambda\mathbf{v}$$

- In which:
  - $\mathbf{v}$  is an eigenvector and
  - Lambda is the eigenvalue associated to it.

# Eigenvalues and Eigenvectors

- From that we have:

$$A\mathbf{v} - \lambda\mathbf{v} = \mathbf{0}$$

- Now remember:

$$\mathbf{v} = I_n \mathbf{v}$$

- Because the identity matrix is the neutral element in matrix multiplication

# Eigenvalues and Eigenvectors

- Thus we can write:

$$A\mathbf{v} - \lambda\mathbf{v} = \mathbf{0}$$

- as:

$$A\mathbf{v} - \lambda I_n \mathbf{v} = \mathbf{0}$$

- What allows us to use the distributive property for matrix multiplication to obtain:

$$(A - \lambda I_n) \mathbf{v} = \mathbf{0}$$

# Eigenvalues and Eigenvectors

- We can use the following equality:

$$B = A - \lambda I_n$$

- To obtain:

$$B\mathbf{v} = \mathbf{0}$$

- Observe the trivial solution is when  $\mathbf{v}$  is the null vector, then it will always produce the zero vector
  - Thus  $\mathbf{v} = \mathbf{0}$  is always in  $N(B)$
  - But we do not want this trivial solution, so we want to find other vectors  $\mathbf{v}$  which are different from the zero vector  $\mathbf{0}$

Observe vector  $\mathbf{v}$  is in the null space of  $B$ , i.e.,  $N(B)$

# Eigenvalues and Eigenvectors

- To remember:
  - Column vectors of  $B$  are linearly independent if and only if

$$N(B) = \{\mathbf{0}\}$$

- i.e., the null space of  $B$  only contains the trivial solution
- So, in our case, column vectors of  $B$  must be linearly dependent!

# Eigenvalues and Eigenvectors

- To remember:
  - If column vectors of  $B$  are linearly dependent, then there is no inverse for  $B$  (remember  $B$  is from  $\mathbb{R}^n$  to  $\mathbb{R}^n$ )
    - As another property, when there is no inverse for  $B$ , its determinant must be equal to zero!

$$\det(B) = 0$$

# Eigenvalues and Eigenvectors

- Finally we have the formulation to find the eigenvalues and eigenvectors:

$$\det(A - \lambda I_n) = 0$$

- Thus  $A\mathbf{v} = \lambda\mathbf{v}$  for other than the zero vector, if and only if:

$$\det(A - \lambda I_n) = 0$$

# Eigenvalues and Eigenvectors

- Finally we have the formulation to find the eigenvalues and eigenvectors:

- Thus

Now we can solve an example...

# Eigenvalues and Eigenvectors

- Suppose the transformation matrix:

$$A = \begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix}$$

- To compute the eigenvectors and eigenvalues we will use:

$$\det(A - \lambda I_n) = 0$$

# Eigenvalues and Eigenvectors

- So we compute:

$$A = \begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix}$$

$$\det(A - \lambda I_n) = 0$$

$$\det \left( \begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right) = 0$$

$$\det \left( \begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix} \right) = \det \left( \begin{bmatrix} 1 - \lambda & 2 \\ 4 & 3 - \lambda \end{bmatrix} \right) = 0$$

# Eigenvalues and Eigenvectors

- Then we have:

$$\det \begin{pmatrix} 1 - \lambda & 2 \\ 4 & 3 - \lambda \end{pmatrix} = 0$$
$$(1 - \lambda)(3 - \lambda) - (4 \cdot 2) = 0$$

$\lambda^2 - 4\lambda - 5 = 0$

This is called the Characteristic polynomial

$\lambda = -1 \text{ or } \lambda = 5$

These are the eigenvalues for matrix A!

# Eigenvalues and Eigenvectors

- Then we have:

$$\det \begin{pmatrix} [1 - \lambda & 2] \\ [4 & 3 - \lambda] \end{pmatrix} = 0$$

Now we go for the eigenvectors!

This is done.

These are the eigenvalues for matrix A!

# Eigenvalues and Eigenvectors

- Having the eigenvalues:

$$\lambda = -1 \text{ or } \lambda = 5$$

- We now go back to the following equation to find the eigenvectors:

$$(A - \lambda I_n)\mathbf{v} = \mathbf{0}$$

# Eigenvalues and Eigenvectors

- Applying the first lambda:

$$\lambda = -1$$

$$(A - \lambda I_n) \mathbf{v} = \mathbf{0}$$

$$\left( \begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix} - (-1) \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right) \mathbf{v} = \mathbf{0}$$

$$\left( \begin{bmatrix} 2 & 2 \\ 4 & 4 \end{bmatrix} \right) \mathbf{v} = \mathbf{0}$$

- We now apply the **reduced row echelon form** to find the null space for this problem:

$$\begin{bmatrix} 2 & 2 \\ 4 & 4 \end{bmatrix} \text{ Dividing first row by 2} \rightarrow \begin{bmatrix} 1 & 1 \\ 4 & 4 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 \\ 4 & 4 \end{bmatrix} \text{ 2.o row} = -4 \times 1.o \text{ row} + 2.o \text{ row} \rightarrow \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}$$

# Eigenvalues and Eigenvectors

- For the first lambda we have:

$$\lambda = -1$$

$$\begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$v_1 + v_2 = 0$$

$$v_1 = -v_2$$

# Eigenvalues and Eigenvectors

- Suppose we want to formalize the eigen space for this first lambda, then we will have:

$$v_1 = -v_2$$

Assuming  $v_2 = t$ ,  $t \in \mathbb{R}$ , we have:

$$E_{\lambda=-1} = N(A - \lambda I_n) = \left\{ \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = t \begin{bmatrix} -1 \\ 1 \end{bmatrix}, t \in \mathbb{R} \right\}$$

# Eigenvalues and Eigenvectors

- Suppose we want to formalize the eigen space for this first lambda, then we will have:

$$v_1 = -v_2$$

Assuming  $v_2 = t$ ,  $t \in \mathbb{R}$ , we have:

$$E_{\lambda=-1} = N(A - \lambda I_n) = \left\{ \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = t \begin{bmatrix} -1 \\ 1 \end{bmatrix}, t \in \mathbb{R} \right\}$$

The eigen space is given by the null space of  $(A - \lambda I)$

This is the eigenvector

# Eigenvalues and Eigenvectors

- For the second lambda we have:

$$\lambda = 5$$

$$(A - \lambda I_n) \mathbf{v} = \mathbf{0}$$

$$\left( \begin{bmatrix} 1 & 2 \\ 4 & 3 \end{bmatrix} - (5) \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \right) \mathbf{v} = \mathbf{0}$$

$$\begin{pmatrix} -4 & 2 \\ 4 & -2 \end{pmatrix} \mathbf{v} = \mathbf{0}$$

- Then we apply the **rref** to obtain:

$$\begin{bmatrix} -4 & 2 \\ 4 & -2 \end{bmatrix} \quad 2.\text{o row} = 2.\text{o row} + 1.\text{o row} \rightarrow \begin{bmatrix} -4 & 2 \\ 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} -4 & 2 \\ 0 & 0 \end{bmatrix} \quad 1.\text{o row} = 1.\text{o row} / -4 \rightarrow \begin{bmatrix} 1 & -\frac{1}{2} \\ 0 & 0 \end{bmatrix}$$

# Eigenvalues and Eigenvectors

- For the second lambda we have:

$$\begin{bmatrix} 1 & -\frac{1}{2} \\ 0 & 0 \end{bmatrix} \mathbf{v} = \mathbf{0}$$

$$\begin{bmatrix} 1 & -\frac{1}{2} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$v_1 - \frac{1}{2}v_2 = 0$$

$$v_1 = \frac{1}{2}v_2$$

# Eigenvalues and Eigenvectors

- And the eigenspace for the second lambda is:

$$v_1 = \frac{1}{2}v_2$$

Assuming  $v_2 = t$ ,  $t \in \mathbb{R}$ , we have:

$$E_{\lambda=5} = N(A - \lambda I_n) = \left\{ \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = t \begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}, t \in \mathbb{R} \right\}$$

# Eigenvalues and Eigenvectors

- And the eigenspace for the second lambda is:

$$v_1 = \frac{1}{2}v_2$$

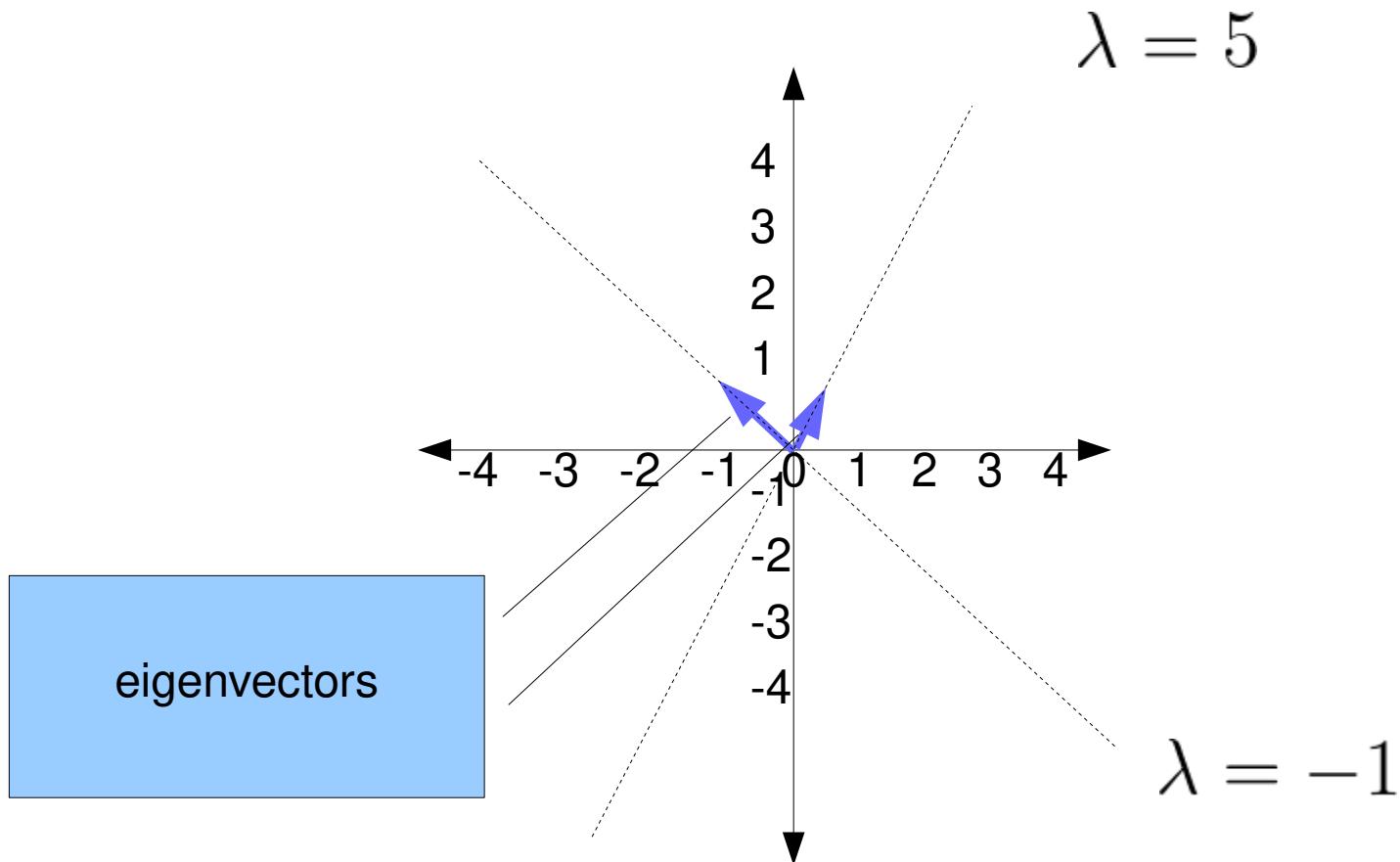
Assuming  $v_2 = t$ ,  $t \in \mathbb{R}$ , we have:

$$E_{\lambda=5} = N(A - \lambda I_n) = \left\{ \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = t \begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}, t \in \mathbb{R} \right\}$$

This is the eigenvector

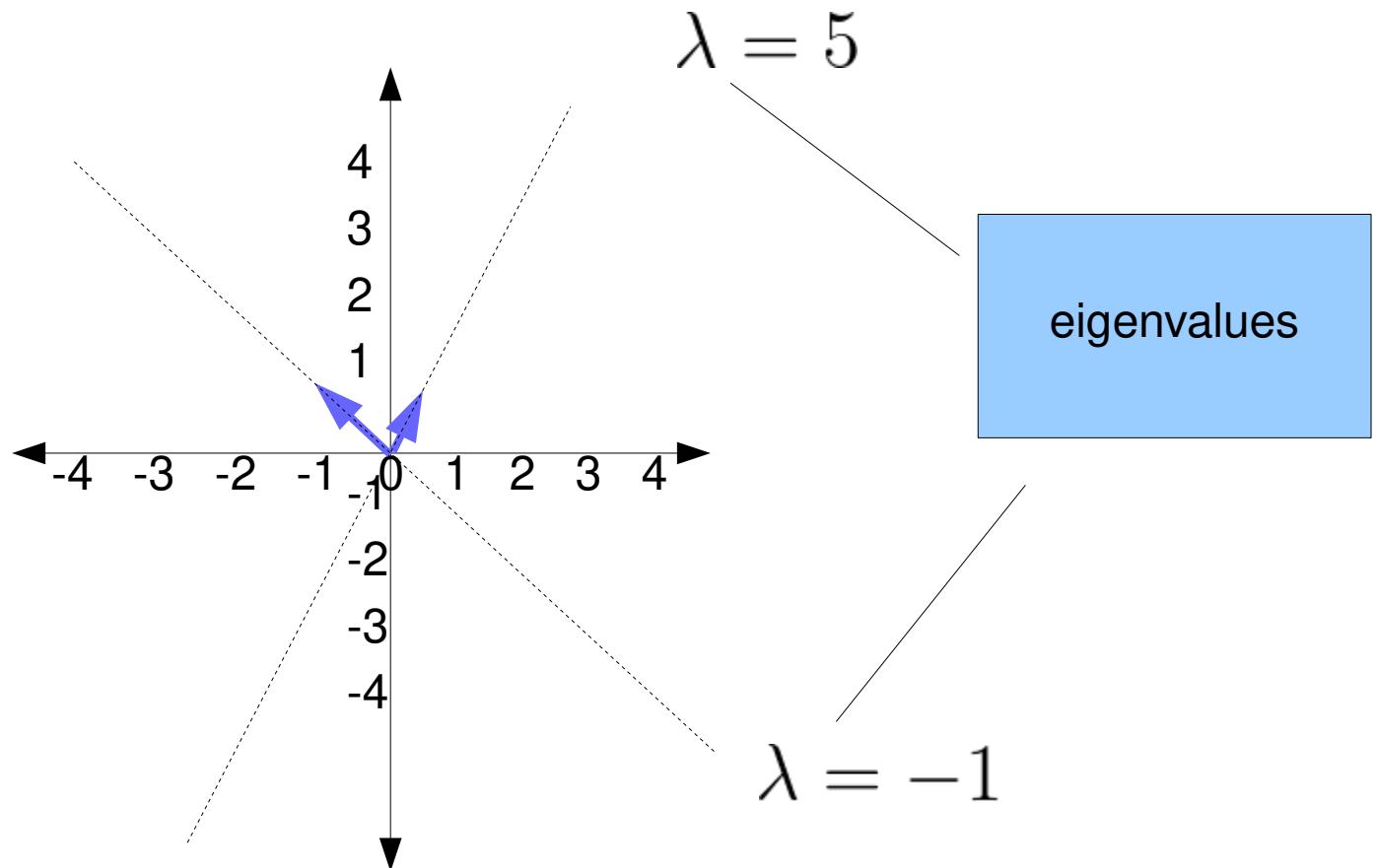
# Eigenvalues and Eigenvectors

- What did we find out?



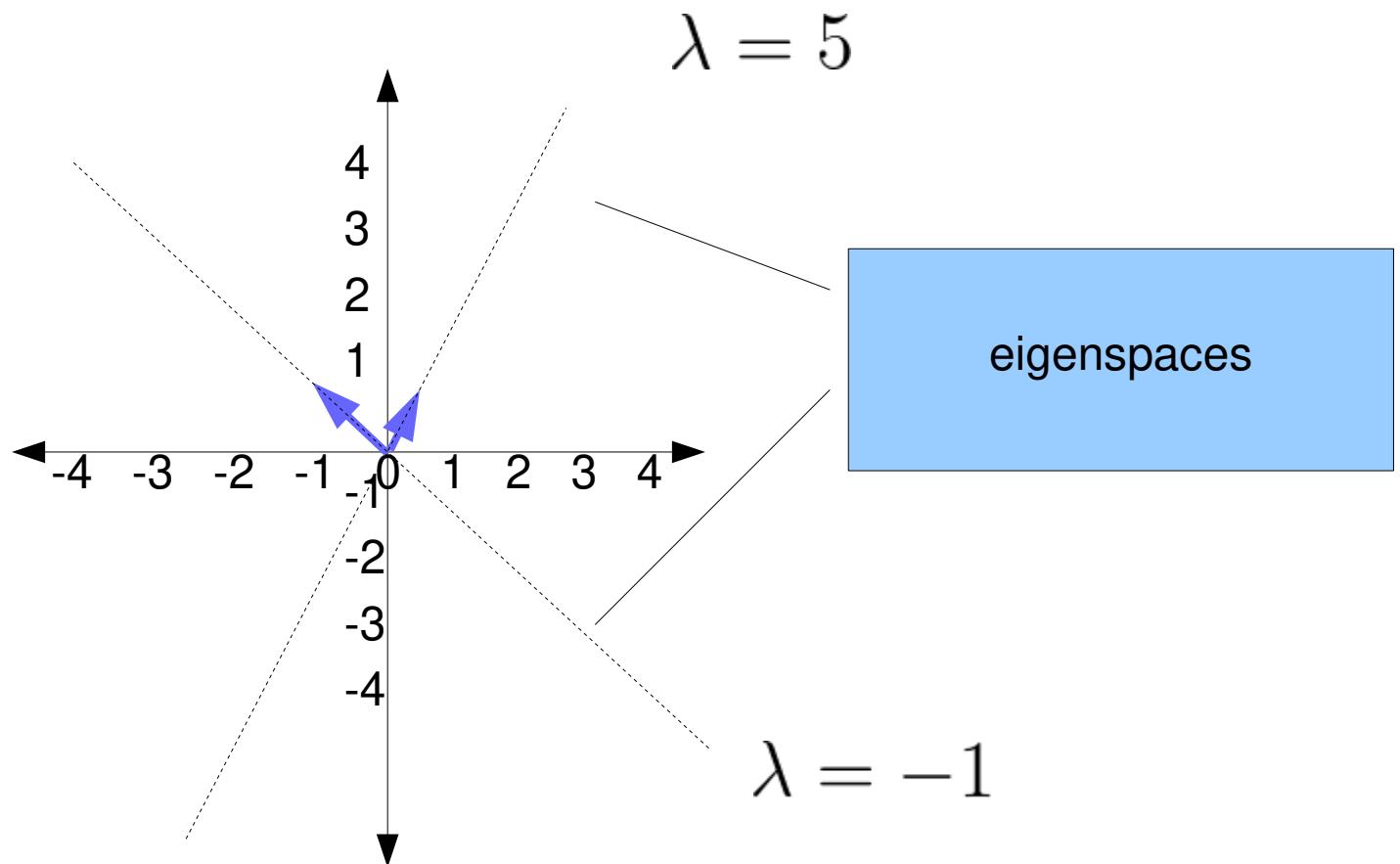
# Eigenvalues and Eigenvectors

- What did we find out?



# Eigenvalues and Eigenvectors

- What did we find out?



# Eigenvalues and Eigenvectors

- See more at:

- [https://pt.khanacademy.org/math/linear-algebra/alternate\\_bases/eigen\\_everything/v/linear-algebra-introduction-to-eigenvalues-and-eigenvectors](https://pt.khanacademy.org/math/linear-algebra/alternate_bases/eigen_everything/v/linear-algebra-introduction-to-eigenvalues-and-eigenvectors)
- [https://pt.khanacademy.org/math/linear-algebra/alternate\\_bases/eigen\\_everything/v/linear-algebra-proof-of-formula-for-determining-eigenvalues](https://pt.khanacademy.org/math/linear-algebra/alternate_bases/eigen_everything/v/linear-algebra-proof-of-formula-for-determining-eigenvalues)
- [https://pt.khanacademy.org/math/linear-algebra/alternate\\_bases/eigen\\_everything/v/linear-algebra-example-solving-for-the-eigenvalues-of-a-2x2-matrix](https://pt.khanacademy.org/math/linear-algebra/alternate_bases/eigen_everything/v/linear-algebra-example-solving-for-the-eigenvalues-of-a-2x2-matrix)
- [https://pt.khanacademy.org/math/linear-algebra/alternate\\_bases/eigen\\_everything/v/linear-algebra-finding-eigenvectors-and-eigenspaces-example](https://pt.khanacademy.org/math/linear-algebra/alternate_bases/eigen_everything/v/linear-algebra-finding-eigenvectors-and-eigenspaces-example)

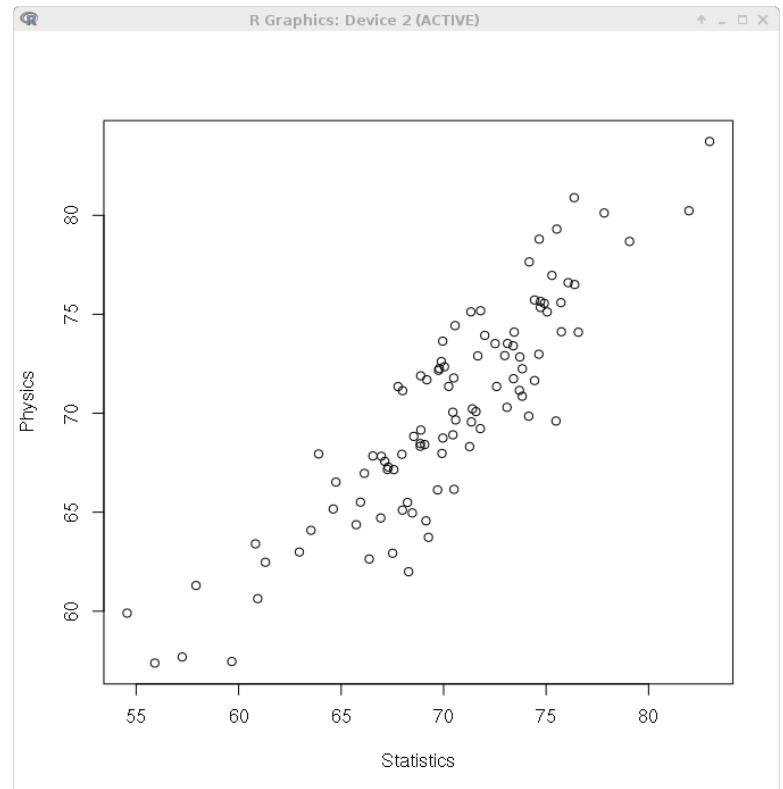
## 8. Principal Component Analysis

- Principal Component Analysis
  - Commonly considered for **feature selection**
  - It is used to compute the relevance of dimensions or data variables
  - Statistical procedure to transform **linearly** correlated variables into uncorrelated variables
    - The uncorrelated variables are called **Principal Components**
  - PCA allows data transformation to another space which is usually called **feature space**

- As an example consider a data set produced as follows:

```
require(splus2R)
data = as.data.frame( rmvnorm( 100, rho=0.9 ) * 5+70)
colnames(data) = c("Statistics", "Physics")
plot(data)
```

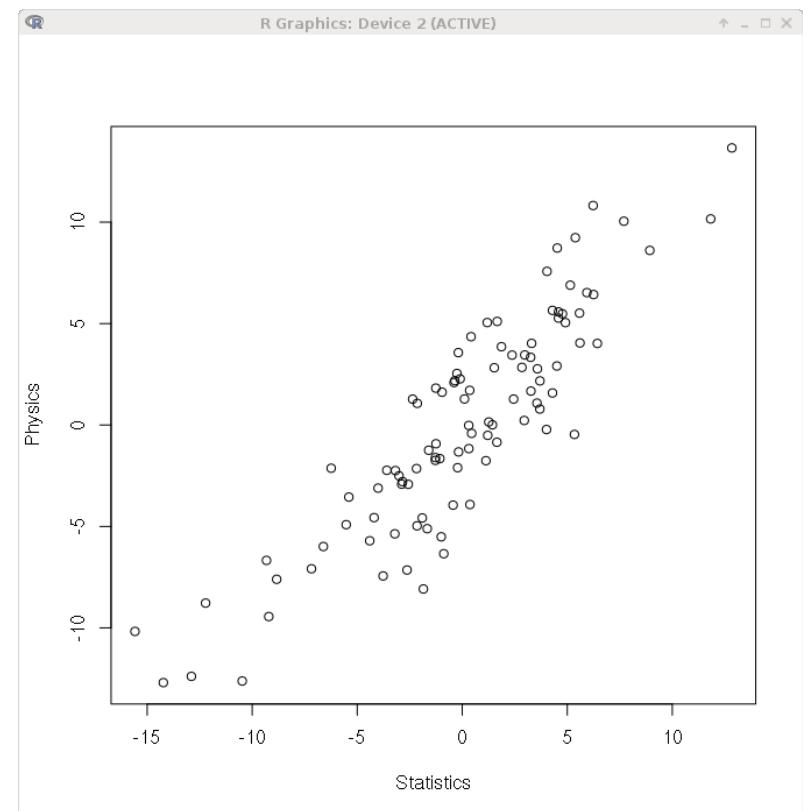
- We can notice some linear correlation between variables



- As first step we reescale data:

```
scaled = apply(data, 2, function(x) { x - mean(x)} )  
plot(scaled)
```

- This step is used to centralize data around point (0,0)



- As next step:
  - **If variables have the same units of measurement**
    - We only consider mean centering " $x - \text{mean}(x)$ " as before
    - Compute the covariance matrix
  - **If variables have different units**
    - We consider mean centering " $x - \text{mean}(x)$ " as before
    - But also divide the result of each dimension by the standard deviation as follows " $(x - \text{mean}(x))/ \text{sd}(x)$ "
    - Compute the correlation matrix

- Now we will apply this step:
  - In our case units of measurement are the same for grades in Statistics and Physics , so we compute the covariance matrix:

**scaled.cov = cov(scaled)**

	Statistics	Physics
Statistics	25.52658	23.5505
Physics	23.55050	27.4904

- As third step we compute the eigenvalues and eigenvectors for the correlation matrix:

```
eigens = eigen(scaled.cov)
```

```
rownames(eigens$vectors) = c("Statistics", "Physics")
```

```
colnames(eigens$vectors) = c("PC1", "PC2")
```

```
eigens
```

```
$values
```

```
[1] 50.079447 2.937531
```

```
$vectors
```

	PC1	PC2
--	-----	-----

Statistics	0.6922220	-0.7216847
------------	-----------	------------

Physics	0.7216847	0.6922220
---------	-----------	-----------

Eigenvalues

Eigenvectors

- Some interesting observations:
  - The sum of eigenvalues results in the sum of variances for both variables in the scaled data:

```
eigens$values
```

```
[1] 50.079447 2.937531
```

- Thus:

```
sum(eigens$values)
```

```
[1] 53.01698
```

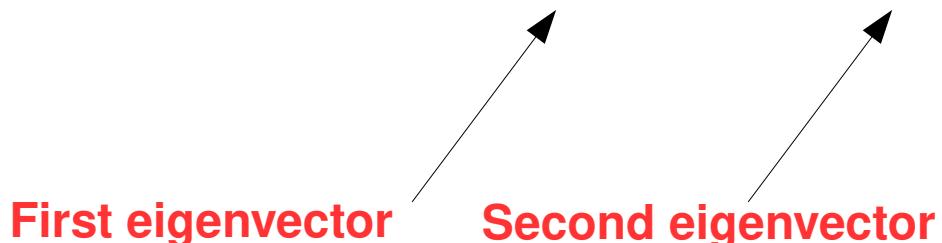
```
var(scaled[,1])+var(scaled[,2])
```

```
[1] 53.01698
```

- More observations:
  - Eigenvectors are known as the Principal Components, that is why the name PCA:

eigens\$vectors

	PC1	PC2
Statistics	0.6922220	-0.7216847
Physics	0.7216847	0.6922220



Each individual value of an eigenvector indicates the association strength of each variable with the corresponding component

For example, Statistics has an association of 0.69 with PC1 and Physics a strength of 0.72 with the same component.

Observe that for PC1, the variables Statistics and Physics have positive associations, i.e., to the **same direction!**

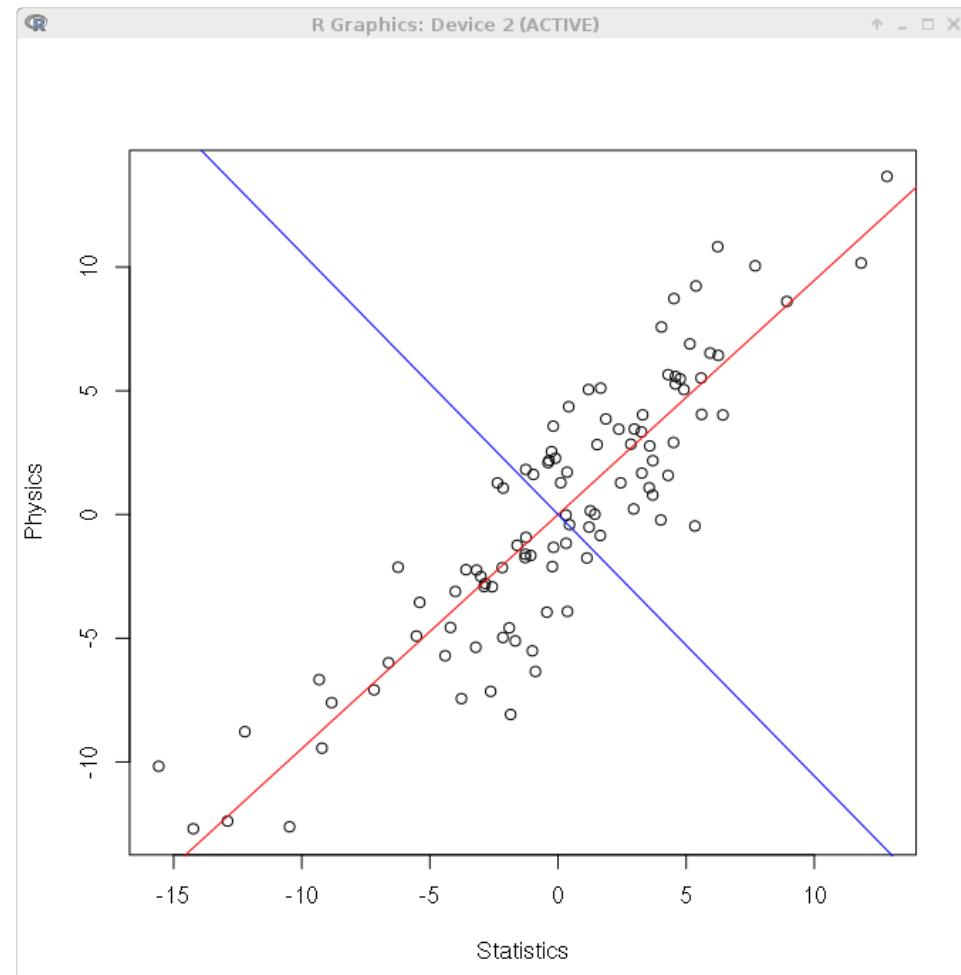
For PC2, it occurs the opposite, associations are inverse to each other

**Eigenvectors** are also called **loadings** when using PCA

- Now we can plot the eigenvectors to observe their behaviors:

```
pc1.slope =  
  eigens$vectors[1,1]/eigens$vectors[2,1]  
pc2.slope =  
  eigens$vectors[1,2]/eigens$vectors[2,2]  
abline(0, pc1.slope, col="red")  
abline(0, pc2.slope, col="blue")
```

- So we observe the relationship among data and eigenvectors



- The eigenvectors are orthogonal as expected
  - PC1, in red, corresponds to the direction with greater data variation
  - Greatest variation components are the most relevant ones
- The percentage of variation in each direction is given by eigenvalues, here we have:

**eigens\$values**

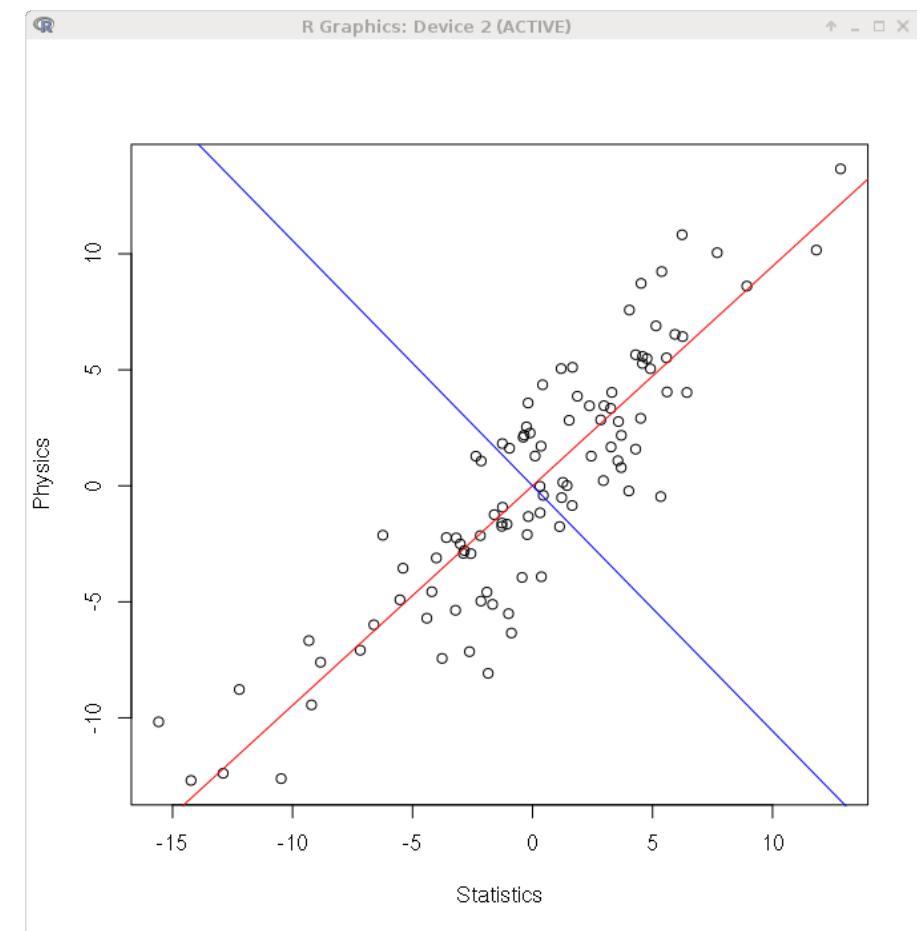
**[1] 50.079447 2.937531**

- Thus:

**eigens\$values/sum(eigens\$values)\*100**

**[1] 94.459263 5.540737**

- Consequently the first component, i.e., PC1, is the most relevant!

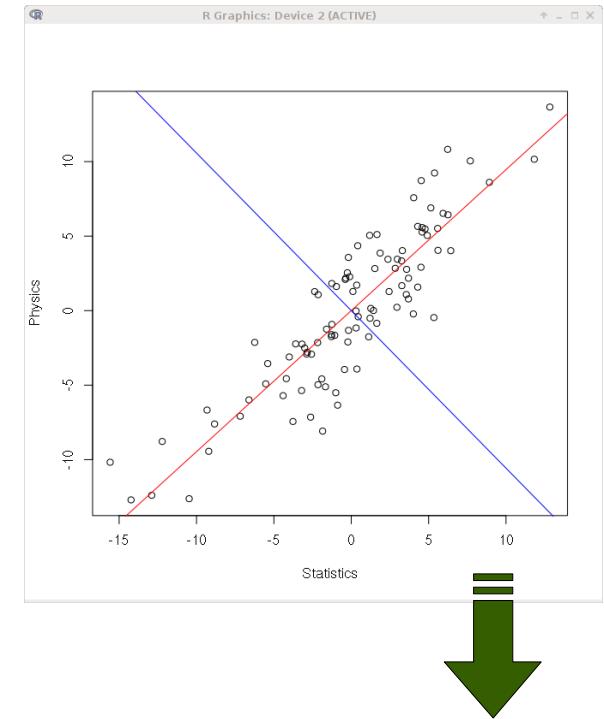


- The next step transforms data according to the principal components
  - This is called the score plot

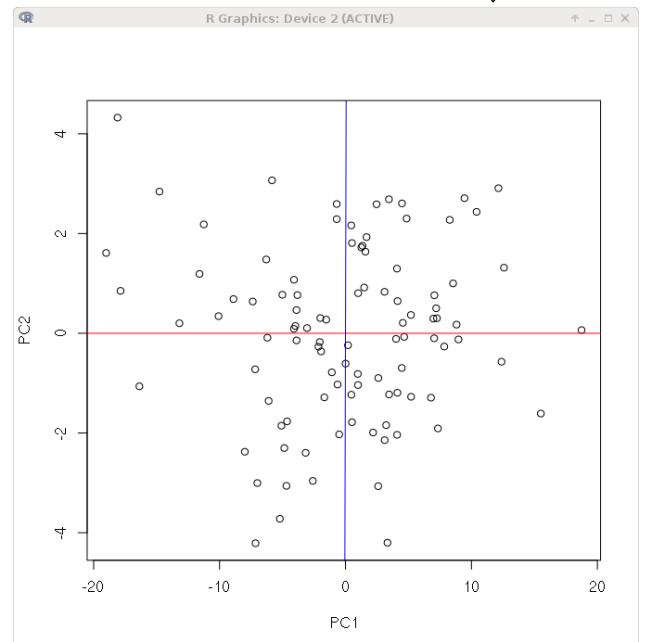
```

scores = scaled %*% eigens$vectors
plot(scores)
abline(0,0,col="red")
abline(0,90,col="blue")
    
```

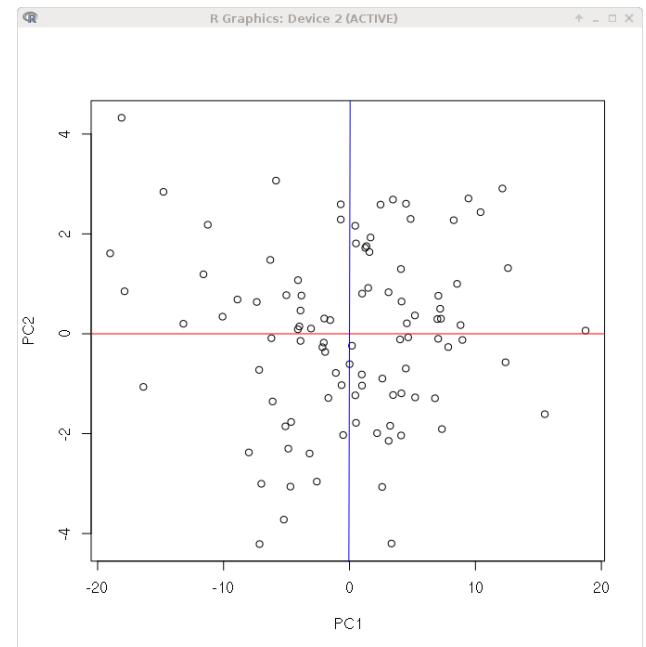
**This works as a linear transformation!**



- Then data is rotated to remove linear correlation:
  - x-axis represents PC1
  - y-axis represents PC2

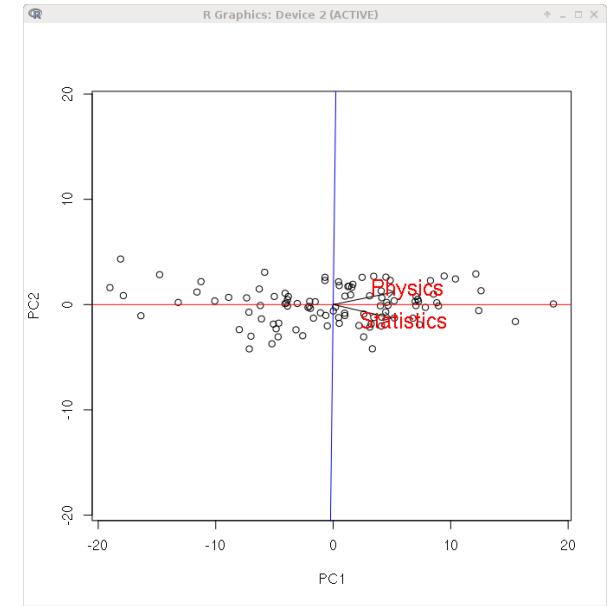


- After obtaining the score plot, we can graph the Biplot
  - This is a very typical chart for PCA
  - It is called Biplot because we graph scores and variables (Statistics and Physics) together
    - Scores are plotted as points
    - Variables are plotted as vectors
  - This plot allows us to observe:
    - The relationship among scores and variables
    - The relationship among variables



- Biplot is produced as follows:

```
plot(scores, xlim=range(scores), ylim=range(scores))
abline(0,0,col="red"); abline(0,90,col="blue")
sd = sqrt(eigens$values); factor=1
arrows(0,0,eigens$vectors[,1]*sd[1]/factor,
       eigens$vectors[,2]*sd[2]/factor, length=0.1, col=1)
text(eigens$vectors[,1]*sd[1]/factor*1.2,
     eigens$vectors[,2]*sd[2]/factor*1.2,
     c("Statistics", "Physics"), cex=1.6, col=2)
```



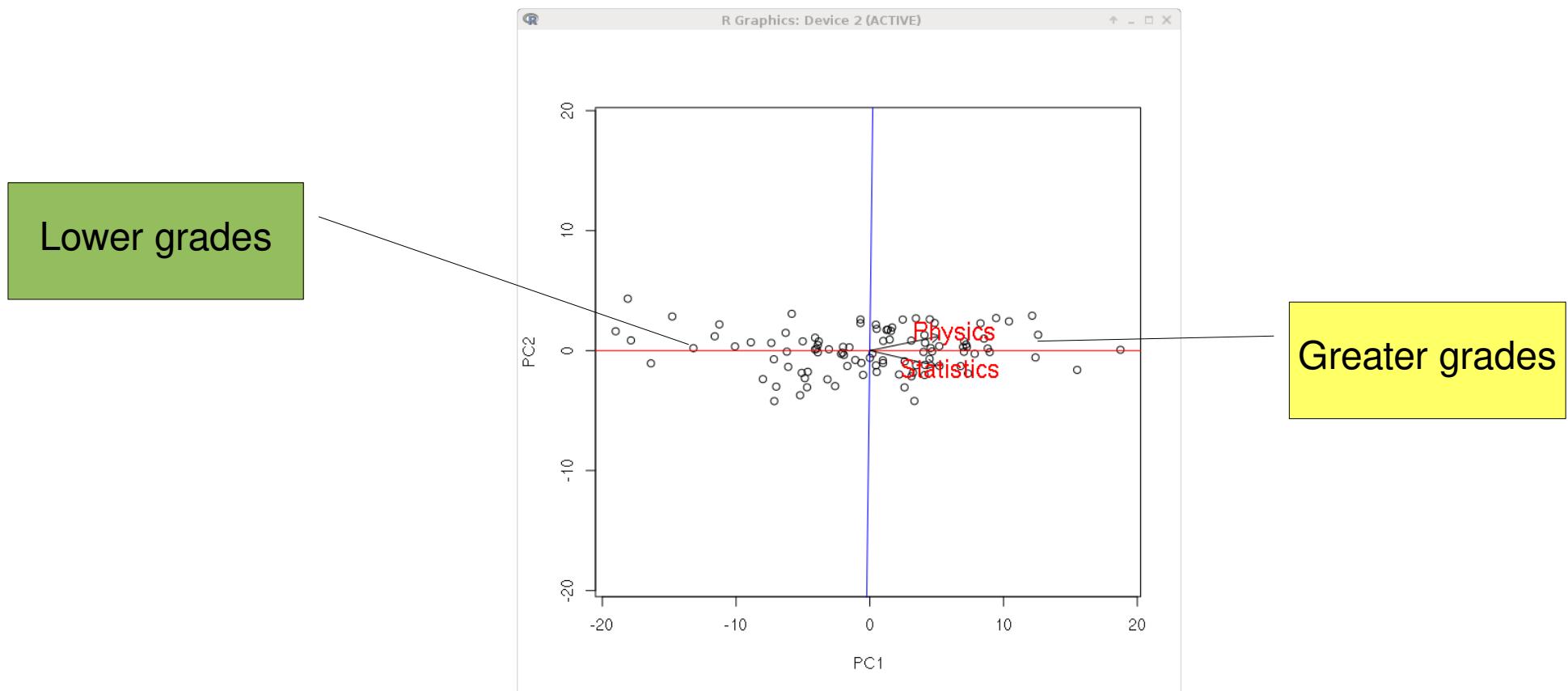
- The cosine of the angle between vectors corresponds to the correlation between variables (Statistics and Physics)

- In this case vectors with a small angle in between, present a greater correlation (0.888):

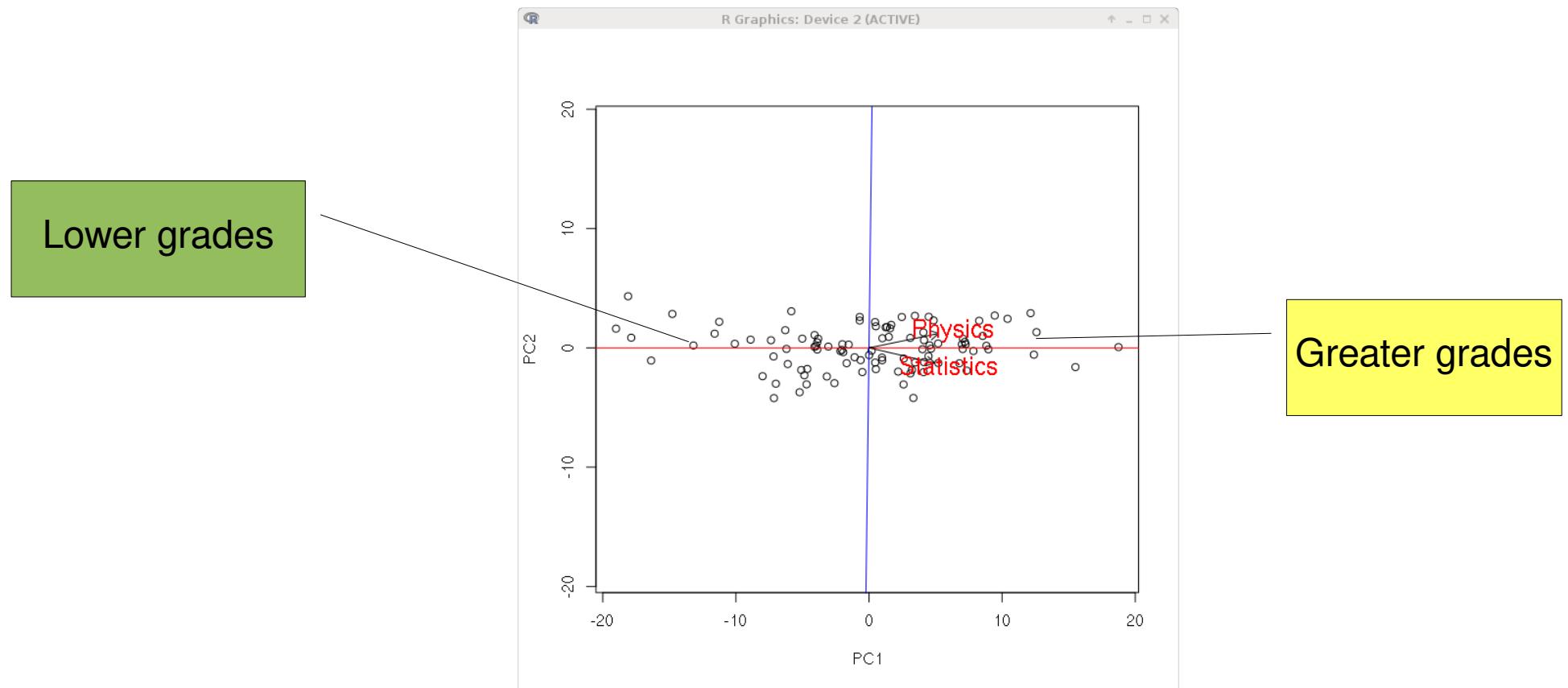
```
cor(data)
```

	Statistics	Physics
Statistics	1.0000000	0.8883099
Physics	0.8883099	1.0000000

- By analyzing the Biplot:
  - Points close to vector Statistics correspond to a greater grade in that course, this also happens for points close to Physics
  - Points in the direction of vectors correspond to greater grades



- By analyzing the Biplot:
  - As there is correlation, variable vectors pull to the nearby directions (small angle between vectors)
  - If there were anticorrelation, vectors would indicate opposite directions



- What happens if the covariance matrix is computed for data rows instead of columns?
  - Look for Low-Rank Representation (LRR)

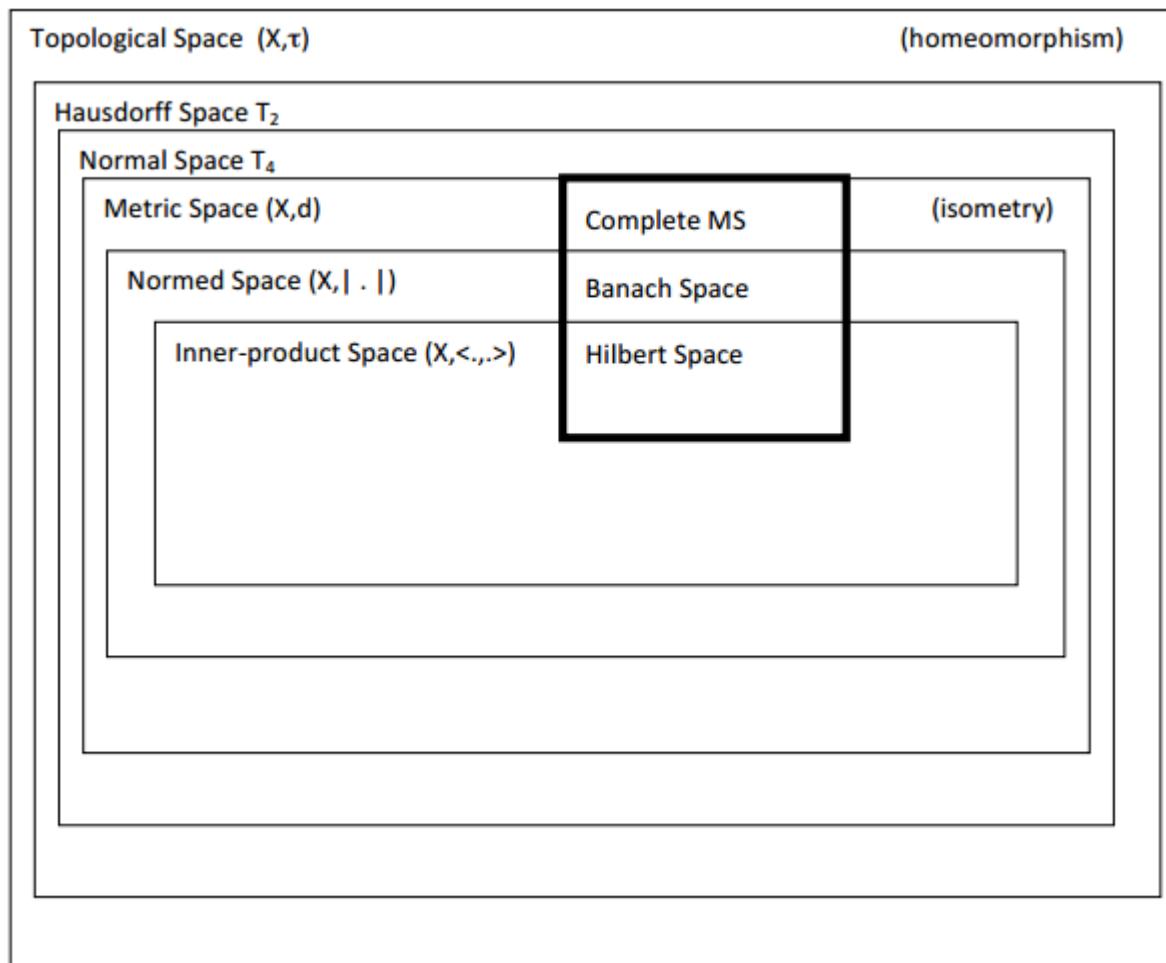
- As homework:
  - Compute PCA for some UCI Datasets
    - <http://archive.ics.uci.edu/ml/>
  - Suggestions:
    - Apply PCA on the Iris dataset and measure the percentage of variation for eigenvalues
      - This allows you to select most important features
      - Then classify this dataset using any technique but considering only the two most relevant PCs and compare the results against the classification over the original dataset (4 attributes)

- As homework:
  - Compute PCA for some UCI Datasets
    - <http://archive.ics.uci.edu/ml/>
  - Suggestions:
    - Do the same for other datasets such as:
      - Wine
      - Etc.

## 9. Understanding the type of space that SVM kernels work on

# More about Spaces

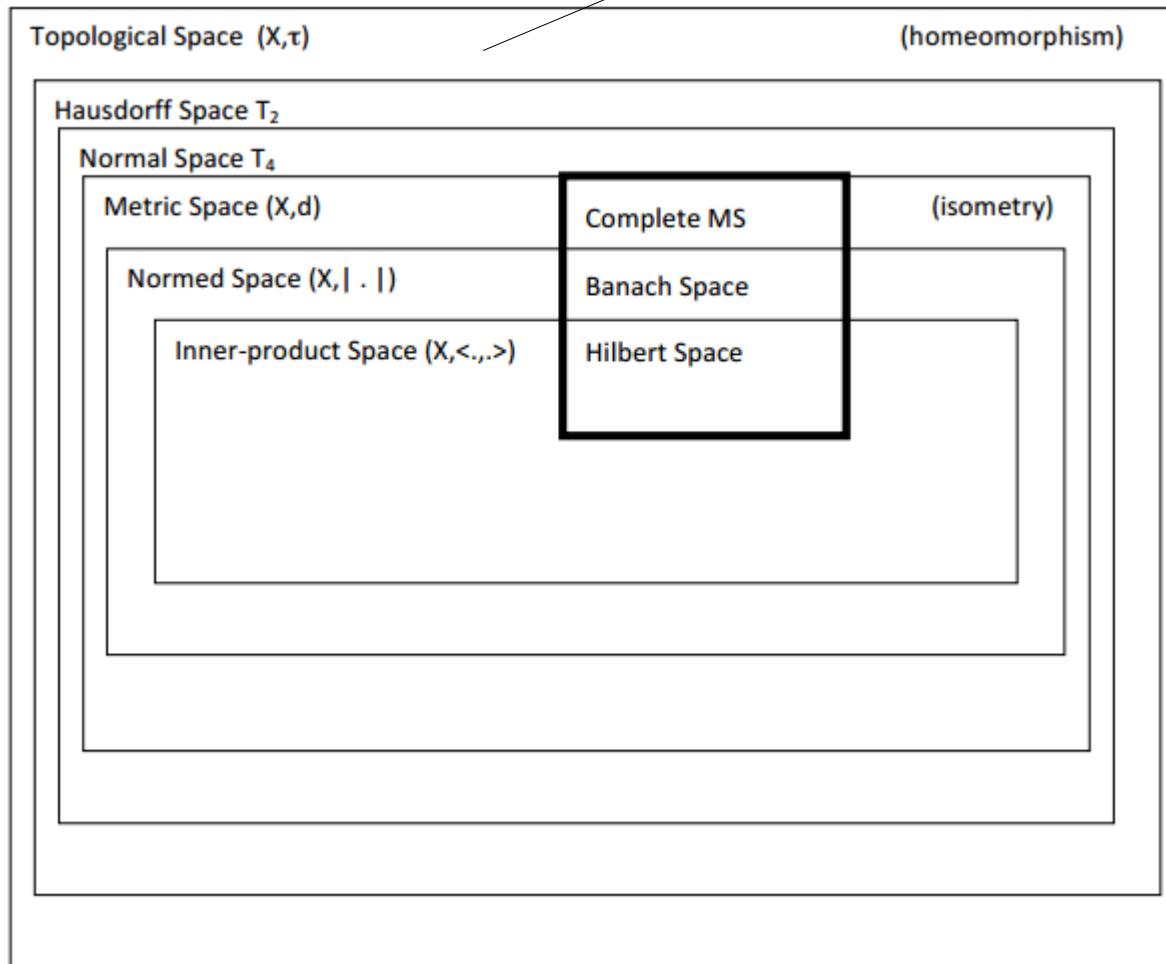
- Spaces:



# More about Spaces

- Spaces:

It requires a set of points  $X$  and their neighborhoods



# More about Spaces

- Spaces:

This is also called separated space or T2 space, in which points in X have disjoint neighborhoods

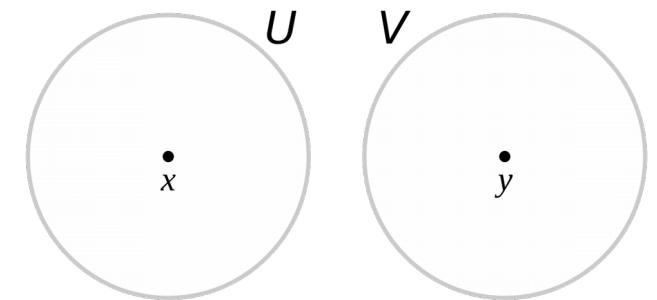
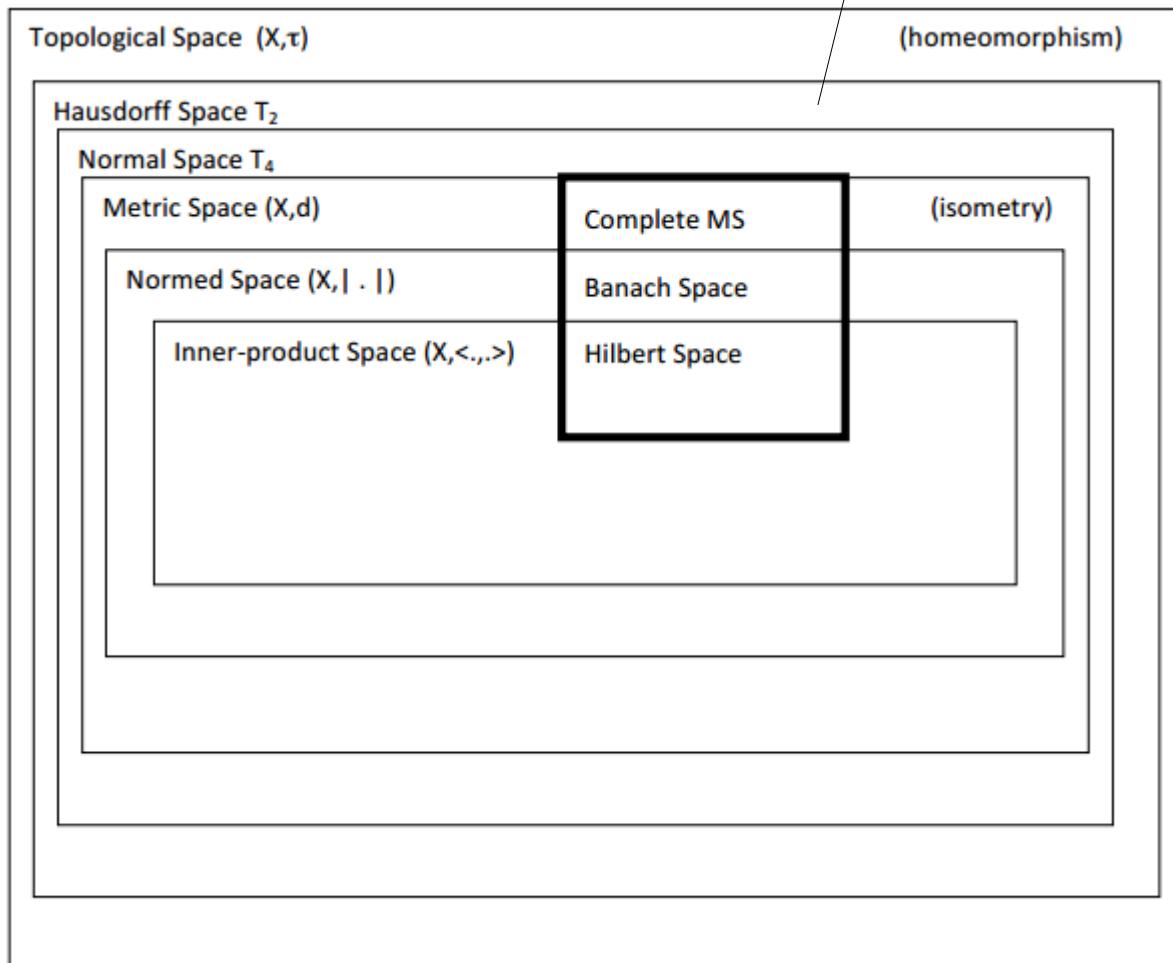


Figure obtained at [http://en.wikipedia.org/wiki/Hausdorff\\_space](http://en.wikipedia.org/wiki/Hausdorff_space)

# More about Spaces

- Spaces:

Every two disjoint closed sets of  $X$  have disjoint open neighborhoods

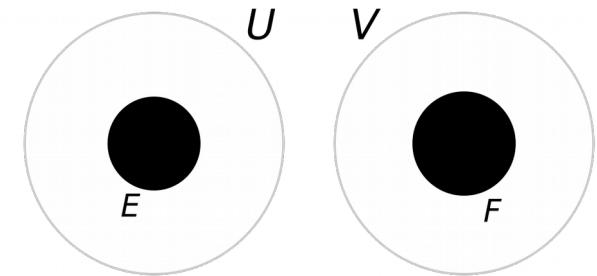
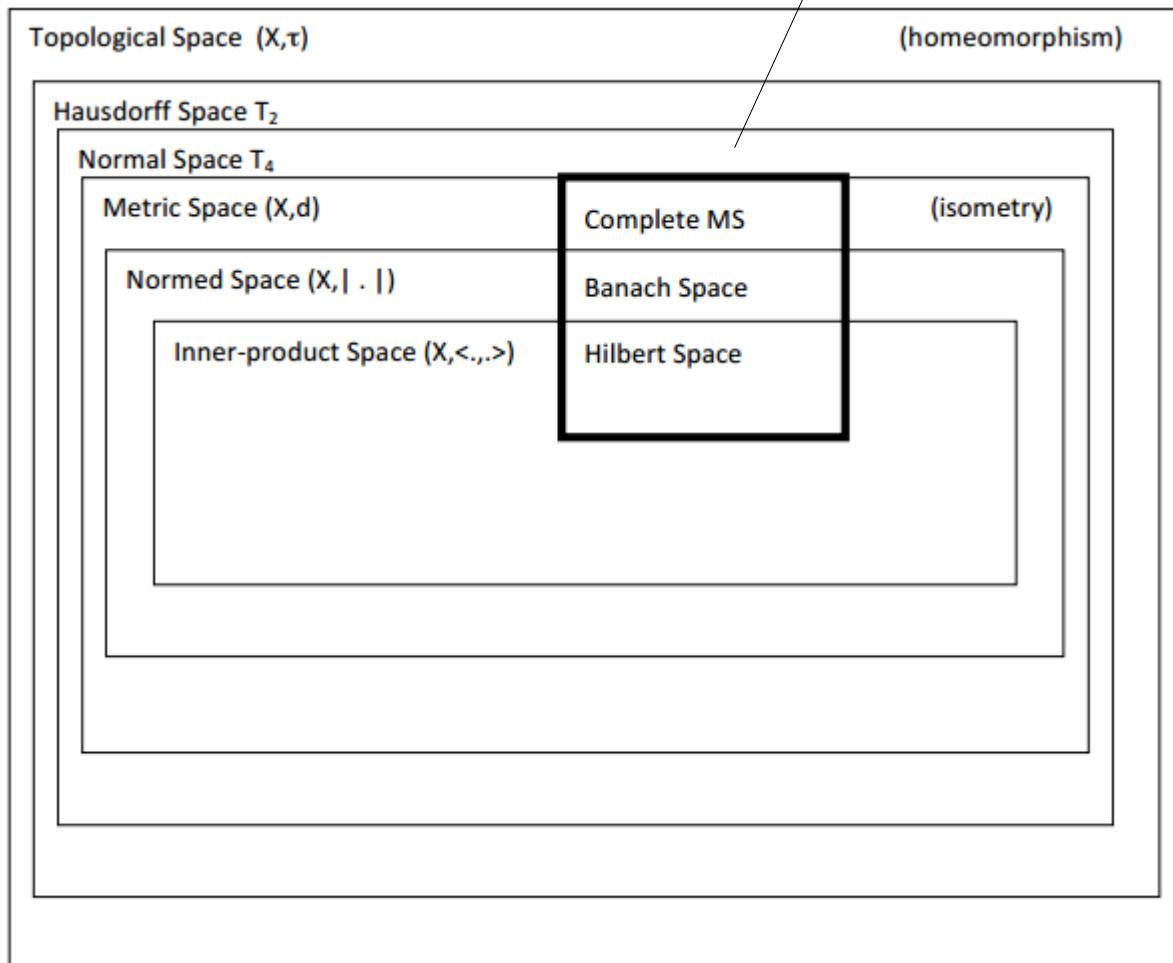
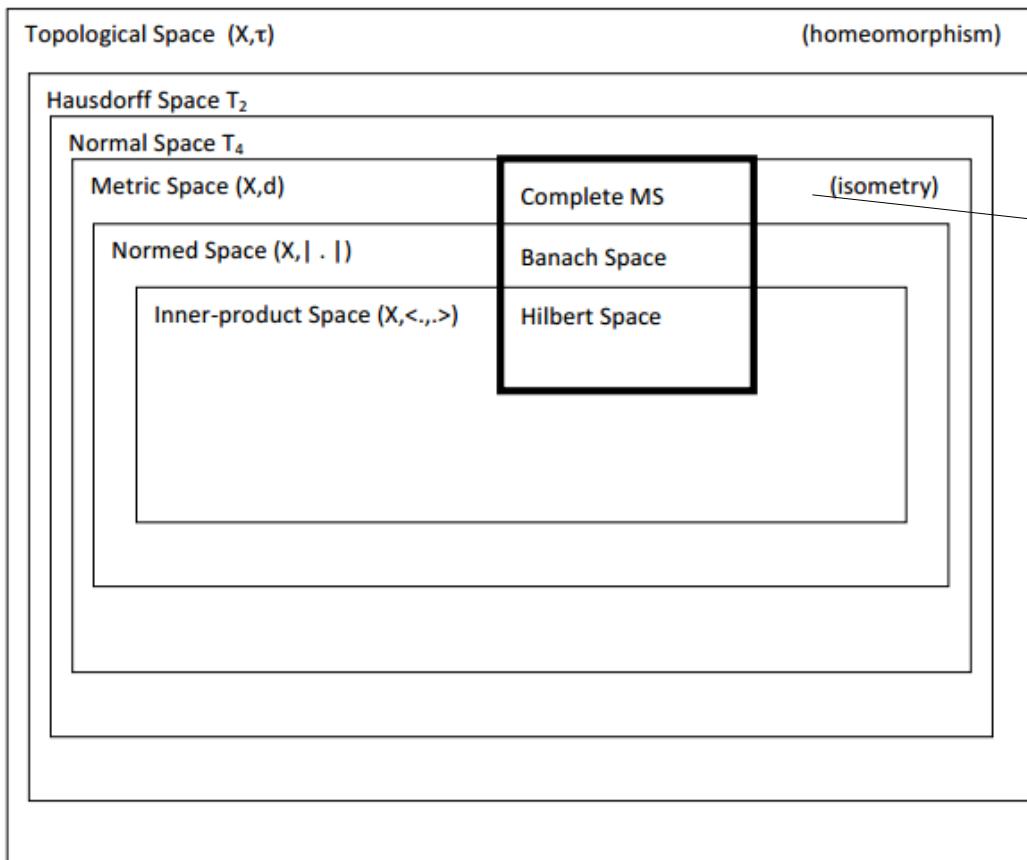


Figure obtained at [http://en.wikipedia.org/wiki/Normal\\_space](http://en.wikipedia.org/wiki/Normal_space)

# More about Spaces



Recall a metric must satisfy:

$$d: X \times X \rightarrow \mathbb{R}$$

such that for any  $x, y, z \in X$ , the following holds:

$$d(x, y) \geq 0 \text{ (non-negative),}$$

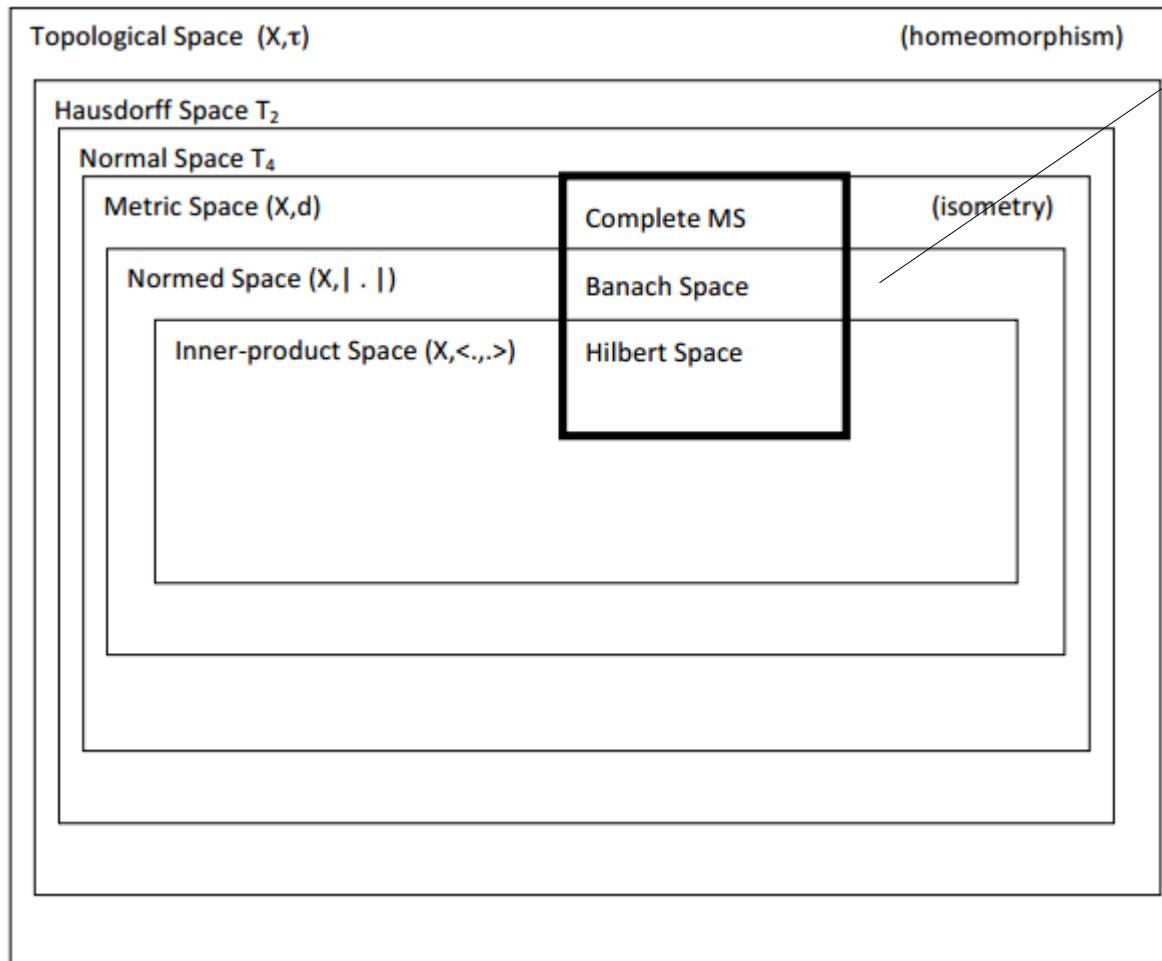
$$d(x, y) = 0 \iff x = y \text{ (identity of indiscernibles)}$$

$$d(x, y) = d(y, x) \text{ (symmetry) and}$$

$$d(x, z) \leq d(x, y) + d(y, z) \text{ (triangle inequality)}$$

# More about Spaces

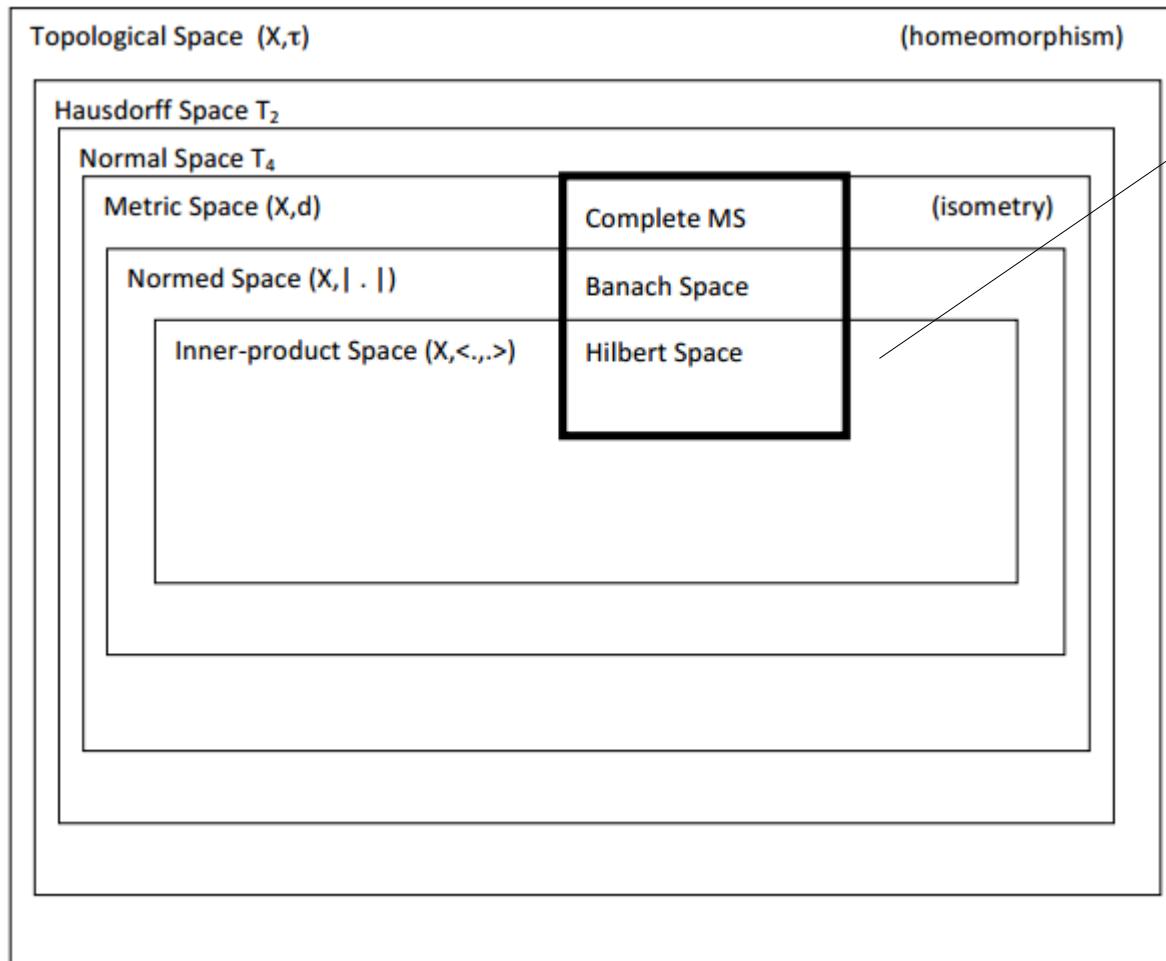
- Spaces:



It has a set of points  $X$  and some form of computing the vector length (norm)

# More about Spaces

- Spaces:



- In the Hilbert space:
  - A distance metric is defined
  - Vector norm is also defined
  - The inner product is also defined:
    - It supports projection of vectors into spaces
      - Therefore it supports geometry
- By supporting the inner product, we can proceed with projections as we did with our first classification algorithm based on geometrical features
  - By using the inner product we can:
    - verify how similar a vector is to each other
    - make projections to build separating hyperplanes

Back to the SVM optimization

- Back to the SVM problem in matrix form:

## PRIMAL

$$\begin{aligned} & \underset{\mathbf{s} \in \mathbb{R}^m, b \in \mathbb{R}, \xi \in \mathbb{R}^m}{\text{Minimize}} \quad \frac{1}{2} \alpha^T \mathbf{Q} \alpha + C \sum_{i=1}^m \xi_i \\ & \text{subject to} \quad \mathbf{Q} \alpha + \mathbf{y} b + \xi - \mathbf{s} = \mathbf{e} \\ & \quad \xi \geq 0, \mathbf{s} \geq \mathbf{0} \end{aligned}$$

## DUAL

$$\begin{aligned} & \underset{\alpha \in \mathbb{R}^m}{\text{Maximize}} \quad \mathbf{e}^T \alpha - \frac{1}{2} \alpha^T \mathbf{Q} \alpha \\ & \text{subject to} \quad \mathbf{0} \leq \alpha \leq \mathbf{C} \\ & \quad \mathbf{y}^T \alpha = 0 \end{aligned}$$

- Back to the SVM problem in matrix form:

## PRIMAL

$$\begin{aligned} & \text{Minimize}_{\mathbf{s} \in \mathbb{R}^m, b \in \mathbb{R}, \xi \in \mathbb{R}^m} \quad \frac{1}{2} \alpha^T \mathbf{Q} \alpha + C \sum_{i=1}^m \xi_i \\ & \text{subject to} \quad \mathbf{Q} \alpha + \mathbf{y} b + \xi - \mathbf{s} = \mathbf{e} \\ & \quad \xi \geq 0, \mathbf{s} \geq \mathbf{0} \end{aligned}$$

## DUAL

$$\begin{aligned} & \text{Maximize}_{\alpha \in \mathbb{R}^m} \quad \mathbf{e}^T \alpha - \frac{1}{2} \alpha^T \mathbf{Q} \alpha \\ & \text{subject to} \quad \mathbf{0} \leq \alpha \leq \mathbf{C} \\ & \quad \mathbf{y}^T \alpha = 0 \end{aligned}$$

This is the main term  
to study! Why?

- Observe the problem in its primal and dual form has one of the following representations:

$$\text{Minimize } \alpha^T \mathbf{Q} \alpha$$

- Or:

$$\text{Maximize } -\alpha^T \mathbf{Q} \alpha$$

- But why?

- Observe the problem in its primal and dual form has one of the following representations:

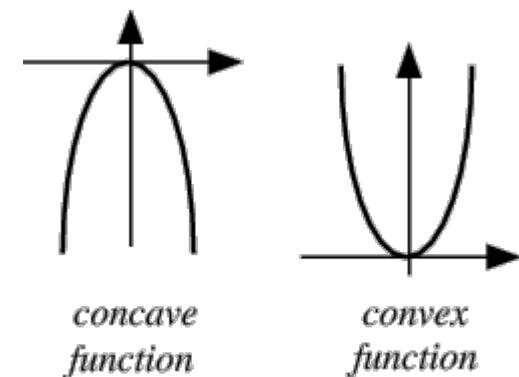
$$\text{Minimize } \alpha^T \mathbf{Q} \alpha$$

- Or:

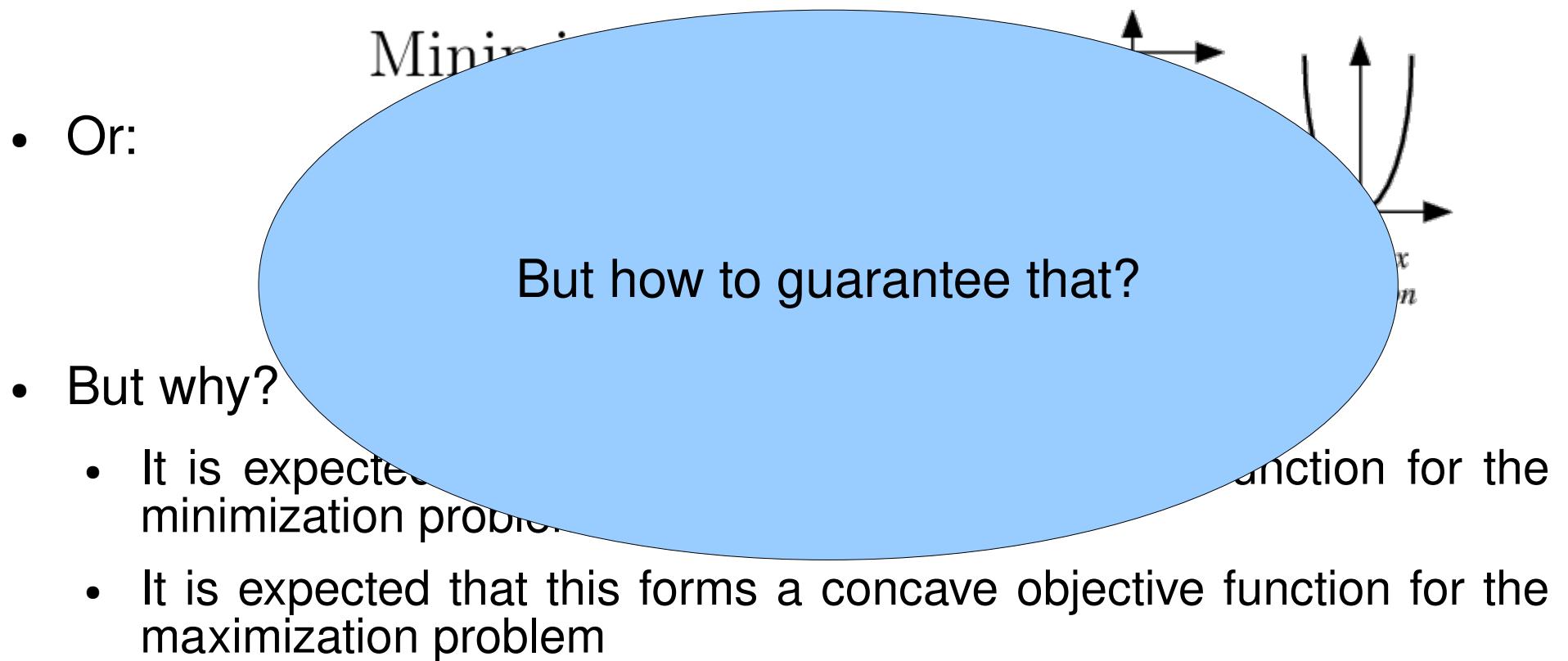
$$\text{Maximize } -\alpha^T \mathbf{Q} \alpha$$

- But why?

- It is expected that this forms a convex objective function for the minimization problem
- It is expected that this forms a concave objective function for the maximization problem



- Observe the problem in its primal and dual form has one of the following representations:



- First of all, remember what is  $\mathbf{Q}$ 
  - For example, for the polynomial kernel it is:

$$\mathbf{Q} = (\mathbf{y}\mathbf{y}^T)(c + \mathbf{x}\mathbf{x}^T)^{\text{order}}$$

- In which:
  - $\mathbf{y}$  represents the vector of classes
  - $\mathbf{x}$  corresponds to the data matrix with training examples
  - $c$  is a coefficient to build the homogeneous (equal to zero) or the inhomogeneous (typically equal to one) kernel
  - order is the polynomial order

- First of all, remember what is  $\mathbf{Q}$ 
  - For example, for the polynomial kernel it is:

$$\mathbf{Q} = (\mathbf{y}\mathbf{y}^T)(c + \mathbf{x}\mathbf{x}^T)^{\text{order}}$$

- In which:

- $\mathbf{y}$  represents the class labels
- $\mathbf{x}$  represents the feature vectors
- $c$  is a constant
- $\mathbf{Q}$  is a matrix of size  $n \times n$  ( $n$  is the number of examples)
- the diagonal elements of  $\mathbf{Q}$  are equal to one (the bias term)
- the off-diagonal elements of  $\mathbf{Q}$  are zero (the kernel trick)

In fact we expect  $\mathbf{Q}$  to form a positive semi-definite matrix!

- What means  $\mathbf{Q}$  to be a **positive semi-definite matrix**?

$$\alpha^T \mathbf{Q} \alpha \geq 0$$

- For example, let's try some matrix  $\mathbf{Q}$ :

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

- What means  $\mathbf{Q}$  to be a **positive semi-definite matrix**?

$$\alpha^T \mathbf{Q} \alpha \geq 0$$

- For example, let's try some matrix  $\mathbf{Q}$ :

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

- We now verify it for all possible real values for:

$$\alpha \in \mathbb{R}^2$$

- What means  $\mathbf{Q}$  to be a **positive semi-definite matrix**?

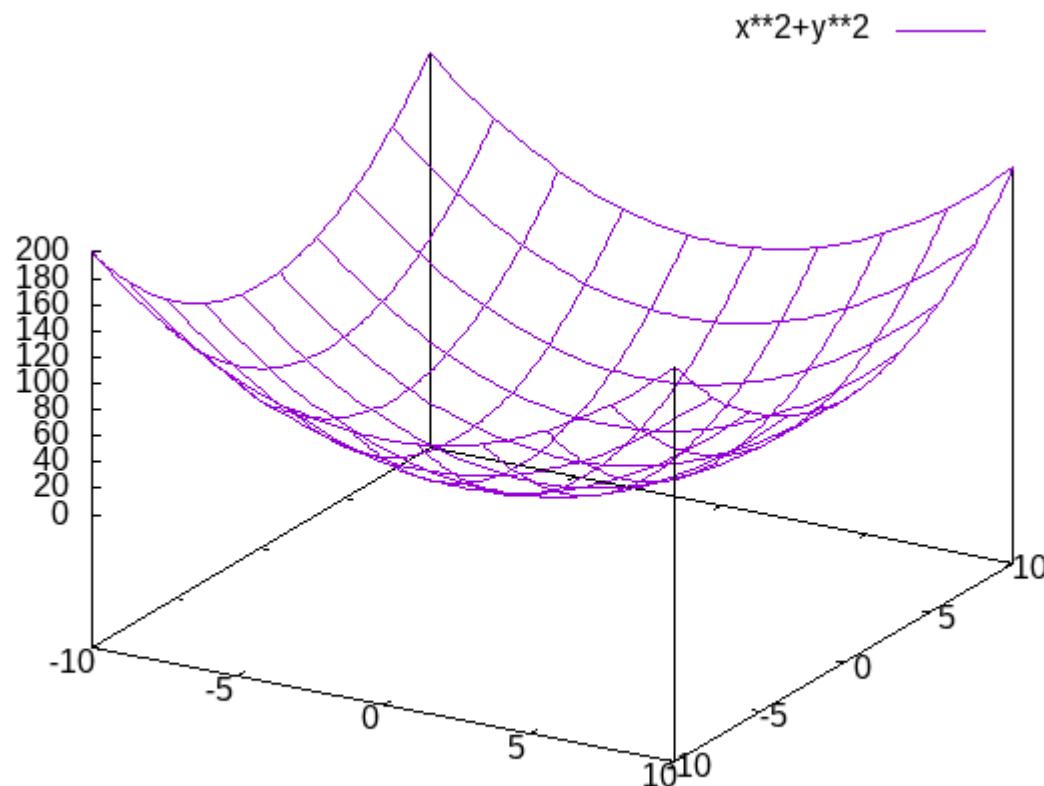
$$\alpha^T \mathbf{Q} \alpha \geq 0$$

- We will have:

$$\begin{aligned} [\alpha_1 \ \alpha_2] \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} &= [1\alpha_1 + 0\alpha_2 \ 0\alpha_1 + 1\alpha_2] \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} \\ &= [\alpha_1 \ \alpha_2] \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = \alpha_1^2 + \alpha_2^2 \end{aligned}$$

# SVM Problem

- What take us to:  
$$\begin{bmatrix} \alpha_1 & \alpha_2 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = \begin{bmatrix} 1\alpha_1 + 0\alpha_2 & 0\alpha_1 + 1\alpha_2 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix}$$
$$= \begin{bmatrix} \alpha_1 & \alpha_2 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = \alpha_1^2 + \alpha_2^2$$



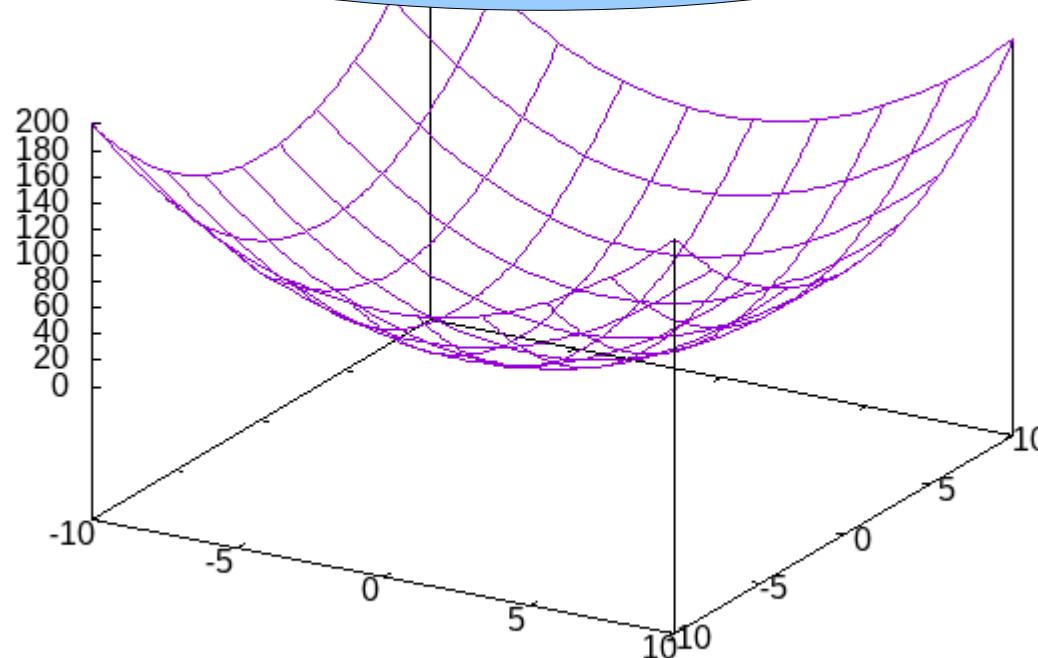
# SVM Problem

- What take us to?

Observe this matrix  $\mathbf{Q}$  takes us to  
a convex surface!

Can we find its minimum? Yes!  
There is only one!

$$0\alpha_1 + 1\alpha_2] \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} + \alpha_2^2$$



- What happens in this case?

$$\mathbf{Q} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

- What happens in this case?

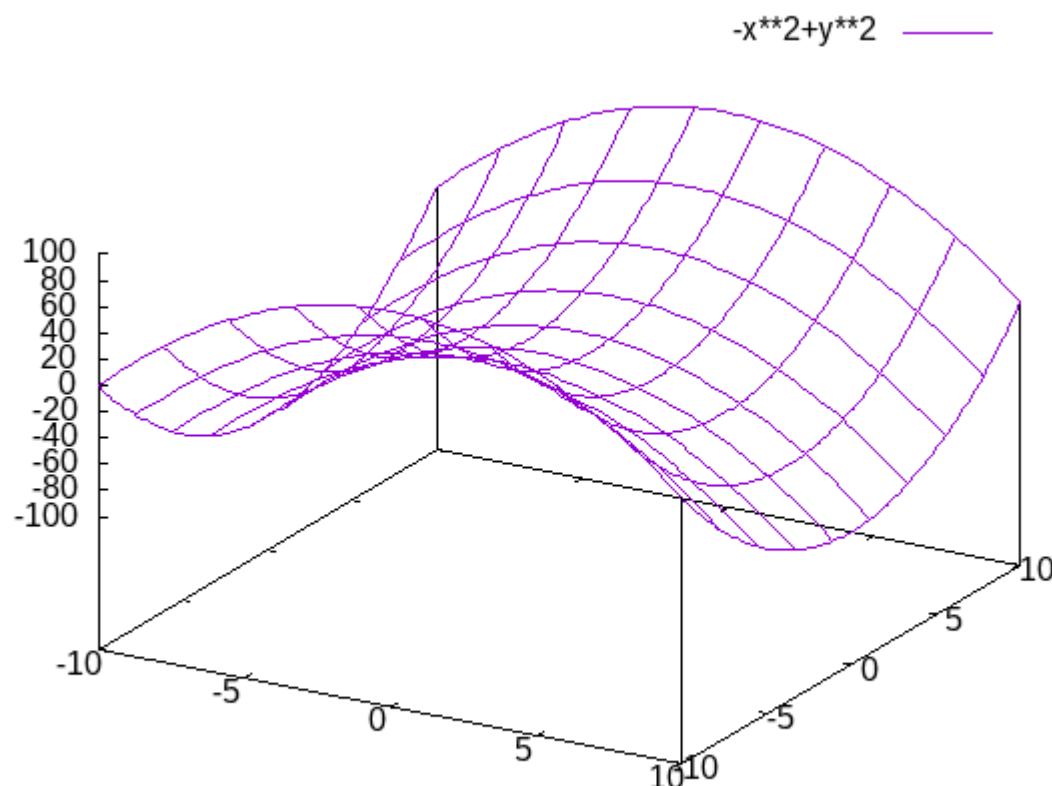
$$\mathbf{Q} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

- Solving for  $\mathbf{Q}$  we have:

$$[\alpha_1 \ \alpha_2] \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = [-\alpha_1 \ \alpha_2] \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = -\alpha_1^2 + \alpha_2^2$$

- Observe the surface we obtained in this case:

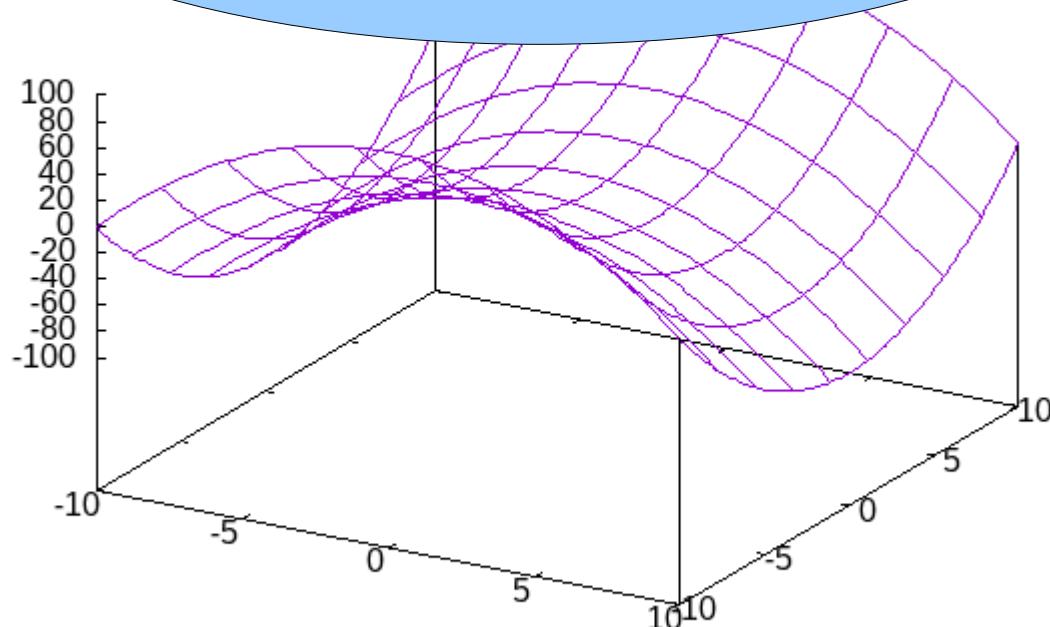
$$[\alpha_1 \ \alpha_2] \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = [-\alpha_1 \ \alpha_2] \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = -\alpha_1^2 + \alpha_2^2$$



- Observe the surface we obtained in this case:

$$[\alpha_1 \quad \alpha_2]^T \Gamma = -\alpha_1^2 - \alpha_2^2$$

It has become a saddle!  
Can we find its minimum for maximum?

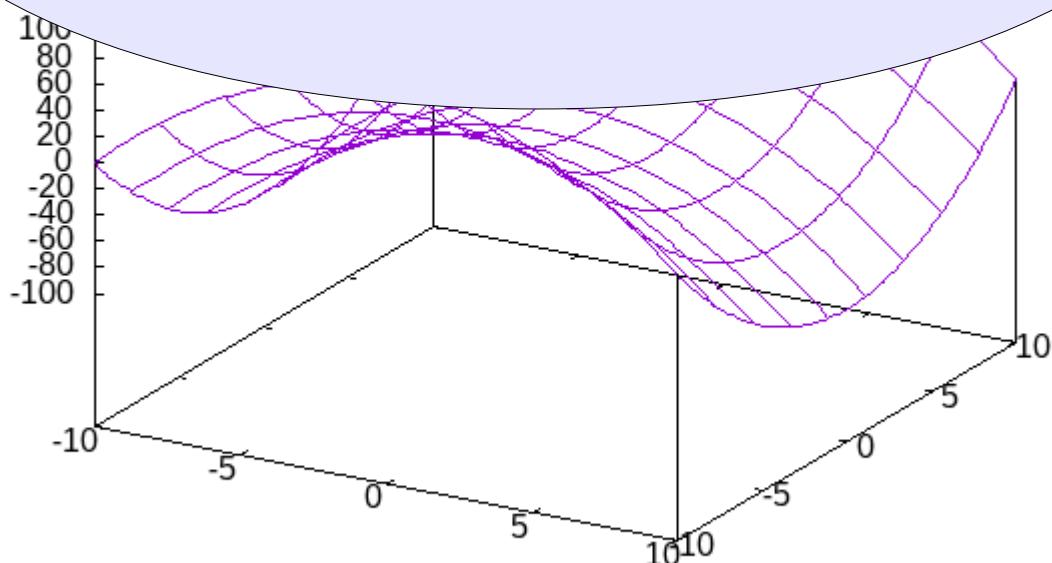


# SVM Problem

- Observe the surface we obtained in this case:

$$[\alpha_1 \quad \alpha_2]^\top \Gamma - \alpha_1^2 + \alpha_2^2$$

No, we cannot!  
So, is it applicable to the SVM  
optimization problem? NO WAY!



- So, what is the main difference between those two forms of  $\mathbf{Q}$ ?

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\mathbf{Q} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

- The first is **positive semi-definite** and the second is not!

- So, what is the main difference between those two forms of  $\mathbf{Q}$ ?

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\mathbf{Q} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

- The first is **positive semi-definite** and the second is not!
  - So, every SVM kernel must guarantee positive semi-definiteness?
    - Yes!

- So, what is the main difference between those two forms of  $\mathbf{Q}$ ?

$$\mathbf{Q} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\mathbf{Q} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$

- The first is **positive semi-definite** and the second is not!
  - So, every SVM kernel must guarantee positive semi-definiteness?
    - Yes!
  - What happens when a kernel does not guarantee that?
    - We are not sure about the result for the minimization and the maximization problems

- But what happens when we have a more complex matrix  $\mathbf{Q}$ ?

- But what happens when we have a more complex matrix  $\mathbf{Q}$ ?
  - Now consider:

$$\mathbf{Q} = \begin{bmatrix} 2.3898474 & 2.832079 & 2.650333 \\ 0.9126086 & 3.420085 & 2.254460 \\ 3.1522972 & 3.880369 & 1.866679 \end{bmatrix}$$

- How can we confirm if this is a positive semi-definite matrix?

- But what happens when we have a more complex matrix  $\mathbf{Q}$ ?
  - Now consider:

$$\mathbf{Q} = \begin{bmatrix} 2.3898474 & 2.832079 & 2.650333 \\ 0.9126086 & 3.420085 & 2.254460 \\ 3.1522972 & 3.880369 & 1.866679 \end{bmatrix}$$

- How can we confirm if this is a positive semi-definite matrix?
  - By computing its eigenvalues

- But what happens when we have a more complex matrix  $\mathbf{Q}$ ?
  - Now consider:

$$\mathbf{Q} = \begin{bmatrix} 2.3898474 & 2.832079 & 2.650333 \\ 0.9126086 & 3.420085 & 2.254460 \\ 3.1522972 & 3.880369 & 1.866679 \end{bmatrix}$$

- How can we confirm if this is a positive semi-definite matrix?
  - By computing its eigenvalues
    - Remember eigenvalues represent the direction and magnitude of eigenvectors!

- But what happens when we have a more complex matrix  $\mathbf{Q}$ ?
  - Now consider:

$$\mathbf{Q} = \begin{bmatrix} 2.3898474 & 2.832079 & 2.650333 \\ 0.9126086 & 3.420085 & 2.254460 \\ 3.1522972 & 3.880369 & 1.866679 \end{bmatrix}$$

- How can we confirm if this is a positive semi-definite matrix?
  - By computing its eigenvalues
    - Remember eigenvalues represent the direction and magnitude of eigenvectors!
    - If no eigenvalue is negative, there is no reflection in the linear transformation

- But what happens when we have a more complex matrix  $\mathbf{Q}$ ?
  - Now consider:

$$\mathbf{Q} = \begin{bmatrix} 2.3898474 & 2.832079 & 2.650333 \\ 0.9126086 & 3.420085 & 2.254460 \\ 3.1522972 & 3.880369 & 1.866679 \end{bmatrix}$$

- How can we confirm if this is a positive semi-definite matrix?
  - By computing its eigenvalues
    - Remember eigenvalues represent the direction and magnitude of eigenvectors!
    - If no eigenvalue is negative, there is no reflection in the linear transformation
    - Linear transformations without reflection correspond to positive semi-definite matrices! Great!

- For this case we have:

$$\mathbf{Q} = \begin{bmatrix} 2.3898474 & 2.832079 & 2.650333 \\ 0.9126086 & 3.420085 & 2.254460 \\ 3.1522972 & 3.880369 & 1.866679 \end{bmatrix}$$

- with the following eigenvalues:

$$\text{eigenvalues} = \{7.599484, 1.159373, -1.082246\}$$

- For this case we have:

$$\mathbf{Q} = \begin{bmatrix} 2.3898474 & 2.832079 & 2.650333 \\ 0.9126086 & 3.420085 & 2.254460 \\ 3.1522972 & 3.880369 & 1.866679 \end{bmatrix}$$

- with the following eigenvalues:

$$\text{eigenvalues} = \{7.599484, 1.159373, -1.082246\}$$

- Is it positive semi-definite? No, because one of the eigenvalues is negative!

- Now try another example:

```
data = cbind(rnorm(mean=0, sd=1, n=500), rnorm(mean=0, sd=1, n=500))
data = rbind(data, cbind(rnorm(mean=10, sd=1, n=500), rnorm(mean=10, sd=1, n=500)))
y = c(rep(0,500), rep(1,500))
Q = (y%*%t(y))*(data%*%t(data))
```

- Compute the eigenvalues for **Q**
- Observe values...

- Now try another example:

```
data = cbind(rnorm(mean=0, sd=1, n=500), rnorm(mean=0, sd=1, n=500))
data = rbind(data, cbind(rnorm(mean=10, sd=1, n=500), rnorm(mean=10, sd=1, n=500)))
y = c(rep(0,500), rep(1,500))
Q = (y%*%t(y))*(data%*%t(data))
```

- Compute the eigenvalues for **Q**
- Observe values...
  - Some of them may be slightly negative due to precision error
  - Can we confirm if this a positive semi-definite matrix using another method?

- Now try another example:

```
data = cbind(rnorm(mean=0, sd=1, n=500), rnorm(mean=0, sd=1, n=500))
data = rbind(data, cbind(rnorm(mean=10, sd=1, n=500), rnorm(mean=10, sd=1, n=500)))
y = c(rep(0,500), rep(1,500))
Q = (y%*%t(y))*(data%*%t(data))
```

- Compute the eigenvalues for **Q**
- Observe values...
  - Some of them may be slightly negative due to precision error
  - Can we confirm if this a positive semi-definite matrix using another method?
    - Yes, using the Cholesky decomposition

- The Cholesky decomposition produces:

$$\mathbf{A} = \mathbf{L}\mathbf{L}^T$$

- for matrix  $\mathbf{A}$  containing real values
- What will we obtain for the code below?

```
Q = (y%*%t(y))*(data%*%t(data))
chol(Q)
```

- The Cholesky decomposition produces:

$$\mathbf{A} = \mathbf{L}\mathbf{L}^T$$

- for matrix  $\mathbf{A}$  containing real values
- What will we obtain for the code below?

```
Q = (y%*%t(y))*(data%*%t(data))
chol(Q)
```

- Observe the decomposition does not work properly!
  - Why? There are some small perturbations in data which produce some very small but negative eigenvalues

- The Cholesky decomposition produces:

$$\mathbf{A} = \mathbf{L}\mathbf{L}^T$$

- for matrix  $\mathbf{A}$  to be positive definite
- What if  $\mathbf{A}$  is not positive definite?  
How can we work around it?
- Observe that Cholesky decomposition fails to work properly!
  - Why? There are some small perturbations in data which produce some very small but negative eigenvalues

- What is necessary for matrix  $\mathbf{A}$  to be positive definite?
  - In fact Cholesky is used for positive definite matrices and not positive semi-definite matrices...

- What is necessary for matrix  $\mathbf{A}$  to be positive definite?
  - In fact Cholesky is used for positive definite matrices and not positive semi-definite matrices...
  - Great hint:
    - Every positive definite matrix is diagonally dominant
    - So what about if we sum a very small value to every term in diagonal?
      - Gershgorin Circle Theorem

- Now we try to sum a small value and compute the total divergence between the original matrix  $\mathbf{Q}$  and the one obtained after the Cholesky decomposition:

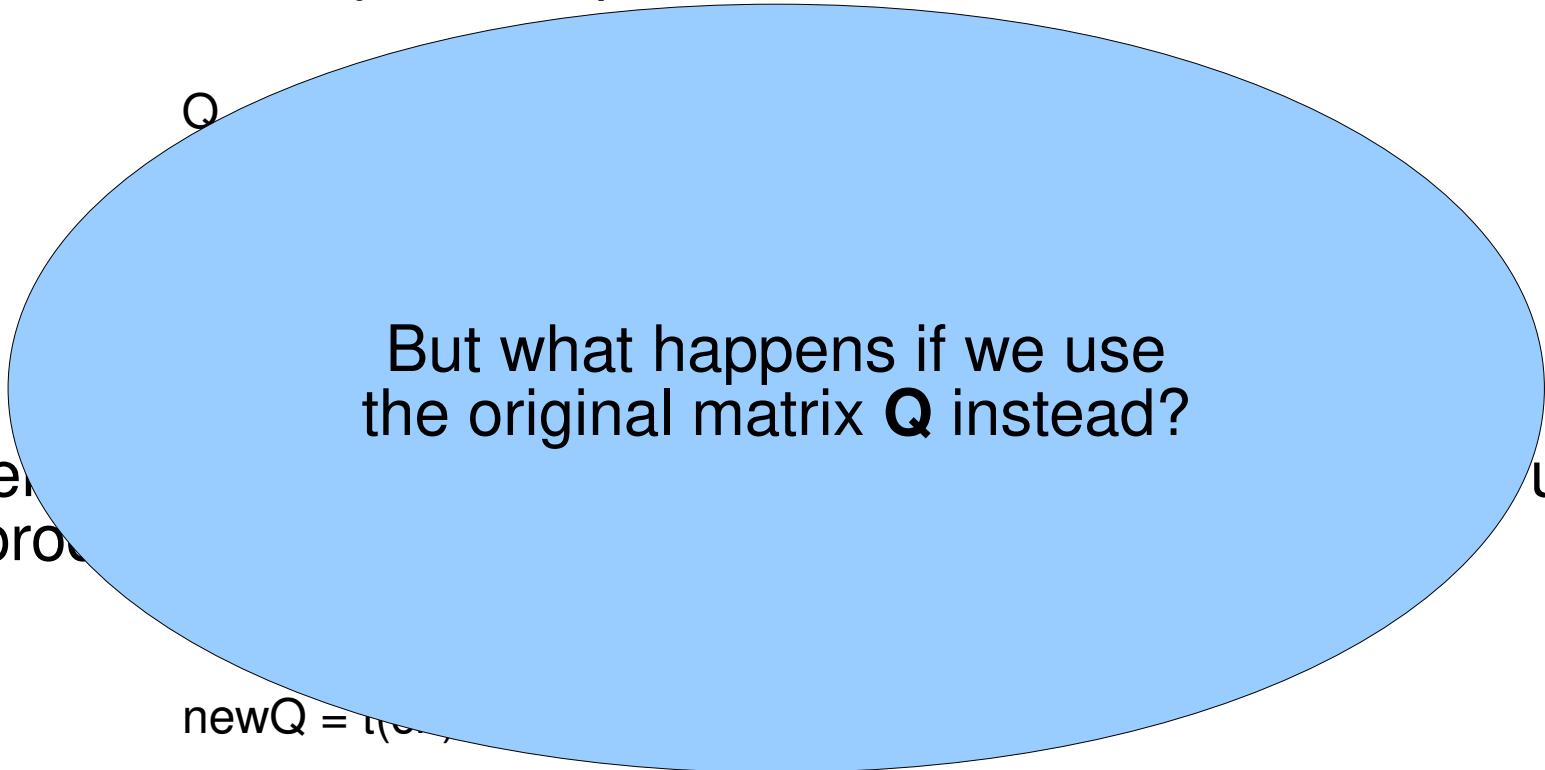
```
Q = (y%*%t(y))*(data%*%t(data))
newQ = Q
for (i in 1:1000) { newQ[i,i] = newQ[i,i] + 1e-10; }
ch = chol(newQ)
total_divergence = sum((Q - t(ch)%*%ch)^2)
```

- Observe now we have indeed a positive definite matrix using the product:

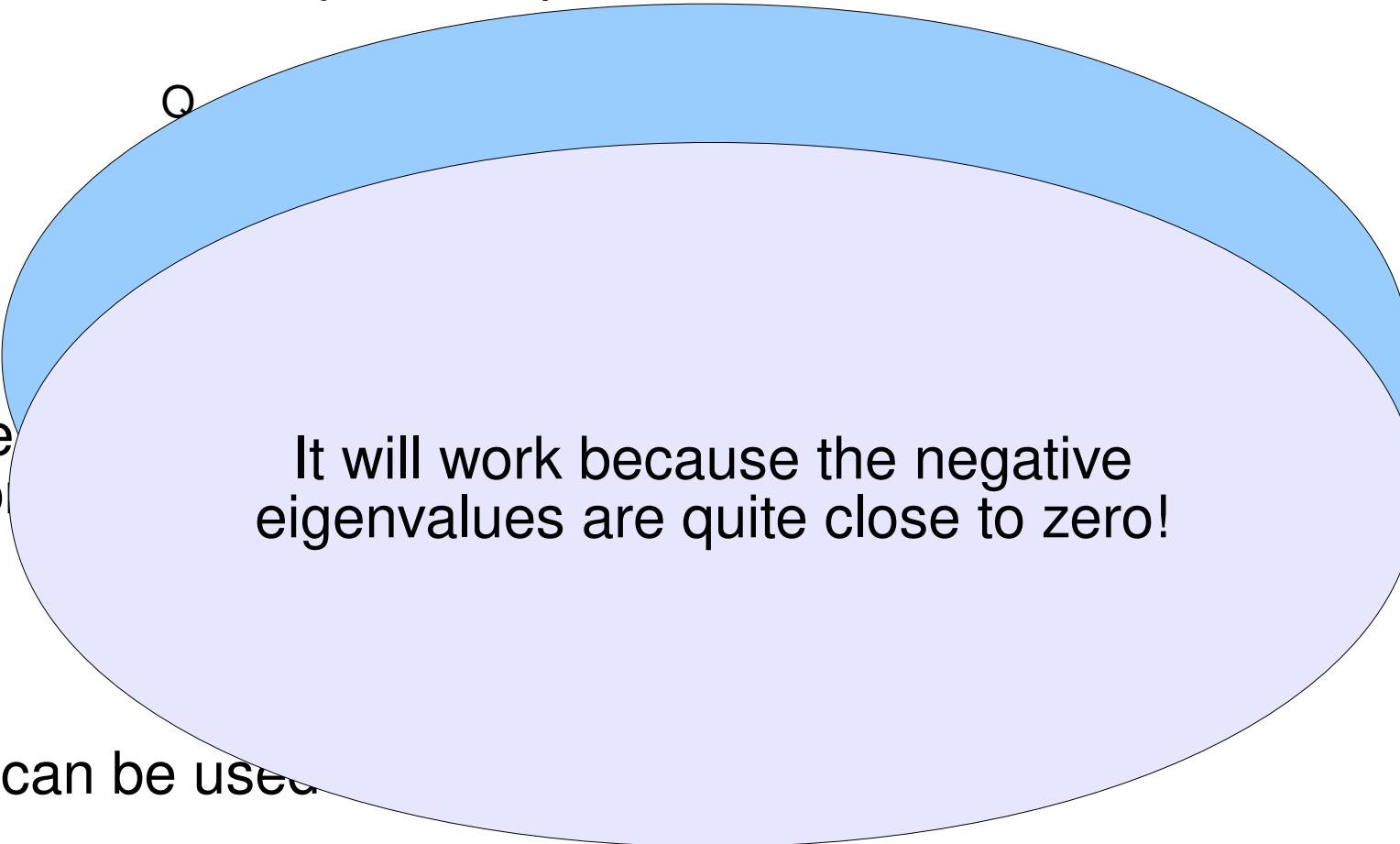
```
newQ = t(ch)%*%ch
```

- This can be used to optimize the SVM problem!

- Now we try to sum a small value and compute the total divergence between the original matrix  $\mathbf{Q}$  and the one obtained after the Cholesky decomposition:

- 
- But what happens if we use  
the original matrix  $\mathbf{Q}$  instead?
- Observing the process using
  - $\text{newQ} = \text{I}(\epsilon)$
  - This can be used to optimize the SVM problem!

- Now we try to sum a small value and compute the total divergence between the original matrix  $\mathbf{Q}$  and the one obtained after the Cholesky decomposition:

- 
- Observe the problem using
  - This can be used
- It will work because the negative eigenvalues are quite close to zero!

- Now we try to sum a small value and compute the total divergence between the original matrix  $\mathbf{Q}$  and the one obtained after the Cholesky decomposition:

- Observe the pattern using
- Just as observation: Notice  $\mathbf{Q}$  is symmetrical
- This is good, because there are guarantees that eigenvalues will be real, otherwise they may be complex
- This can be done using

- But what happens when the total divergence after the Cholesky decomposition gets greater?
  - In fact there are situations we have to sum a greater value in diagonal of  $\mathbf{A}$  to make it positive definite and, therefore, decompose it using Cholesky

- But what happens when the total divergence after the Cholesky decomposition gets greater?
  - In fact there are situations we have to sum a greater value in diagonal of  $\mathbf{A}$  to make it positive definite and, therefore, decompose it using Cholesky
    - This will cause a greater total divergence
    - That may modify a lot the original problem and then we do not have it anymore
      - What we will have is an approximation to our problem
        - Is such approximation enough?
          - Probably not!

- There are adaptations in the Cholesky decomposition to sum the least possible value in diagonal cells
  - It does not add such value in every diagonal cell, but only when necessary
- **Another observation:** the Cholesky decomposition works only for symmetric matrices, what is indeed our case!

- This type of matrix is also called a Gram matrix (or Gramian matrix)
  - The Gramian matrix is positive semi-definite
  - It is symmetric

- This type of matrix is also called a Gram matrix (or Gramian matrix)
  - The Gramian matrix is positive semi-definite
  - It is symmetric
- Therefore, as long as a SVM kernel produces a Gram matrix we can solve the SVM optimization problem
  - **Observe something very important:** in fact we do not need to know anything about the kernel itself, once we indeed obtain a Gram matrix after applying it

- Therefore, as long as a SVM kernel produces a Gram matrix:
  - We have guarantees we will find the solution for the SVM optimization problem
- The so called **kernel trick** is in fact:
  - The SVM optimization problem is still valid since the kernel produces a Gram matrix

- So how to verify if any kernel has produced a Gram matrix?
  - Check out if such matrix is:
    - Symmetrical
    - Positive semi-definite
      - Eigenvalues will help a lot!
      - Cholesky decomposition will also help in that situation

- There are still some interesting stuff:
  - Covariance matrices are Gram matrices
    - Nice!

- There are still some interesting stuff:
  - Covariance matrices are Gram matrices
    - Nice!
  - Let's create any covariance matrix:
    - Verify if it is symmetrical
    - Verify if its eigenvalues are positive

- There is still one subject:
  - There is a theorem called **Mercer's theorem**
    - It is analogue to the study of the eigenvalues to ensure positive definiteness
      - i.e., if eigenvalues are positive
    - The only difference is that this theorem is capable of evaluating kernels as an object in an infinite-dimensional space
- See a simple introduction to Mercer's theorem at:
  - <http://www.quora.com/What-is-an-intuitive-explanation-of-Mercers-Theorem>

## References

- Schölkopf, B., Smola, A. J., Learning With Kernels: Support Vector Machines, Regularization, Optimization, and Beyond, MIT, 2002
- Press, William H.; Saul A. Teukolsky; William T. Vetterling; Brian P. Flannery (1992). Numerical Recipes in C: The Art of Scientific Computing (second edition). Cambridge University England EPress. p. 994. ISBN 0-521-43108-5.
- Fang, Haw-ren; O'Leary, Dianne P. (8 August 2006). "Modified Cholesky Algorithms: A Catalog with New Approaches"