Scientific Computing

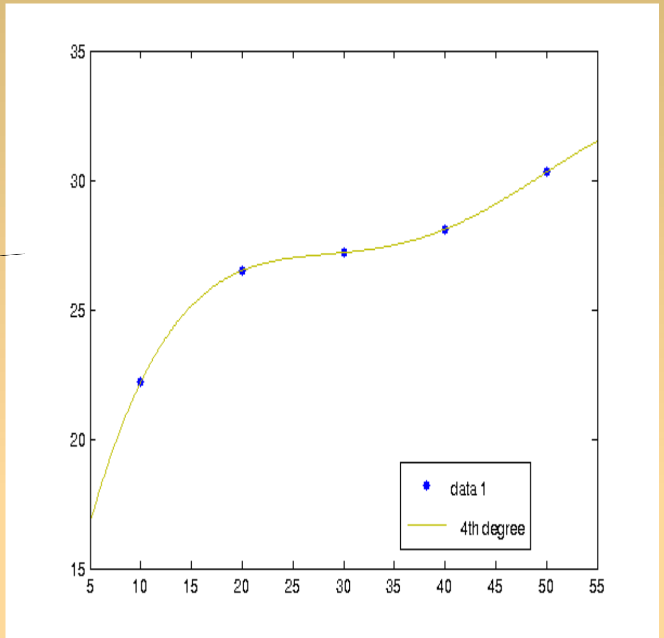
Maastricht Science Program

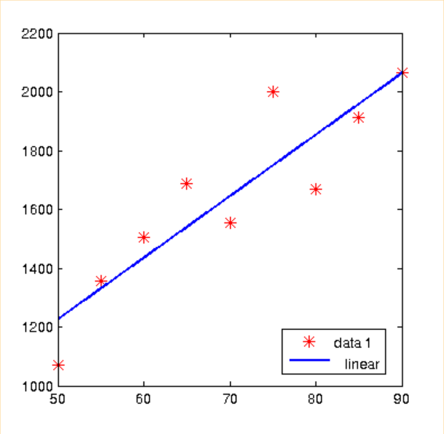
Week 4

Frans Oliehoek
<frans.oliehoek@maastrichtuniversity.nl>

Recap Last Week

- Approximation of Data and Functions
  - find a function *f* mapping x → y
  - Interpolation
    - *f* goes through the data points
    - piecewise or not
  - linear regression
    - lossy fit
    - minimizes SSE
- Linear Algebra
  - Solving systems of linear equations
    - GEM, LU factorization





Recap Least-Squares Method

number of data points:

$$N = n + 1$$

- 'the function unknown'
  - it is only known at certain points $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)$
  - want to predict *y* given *x*

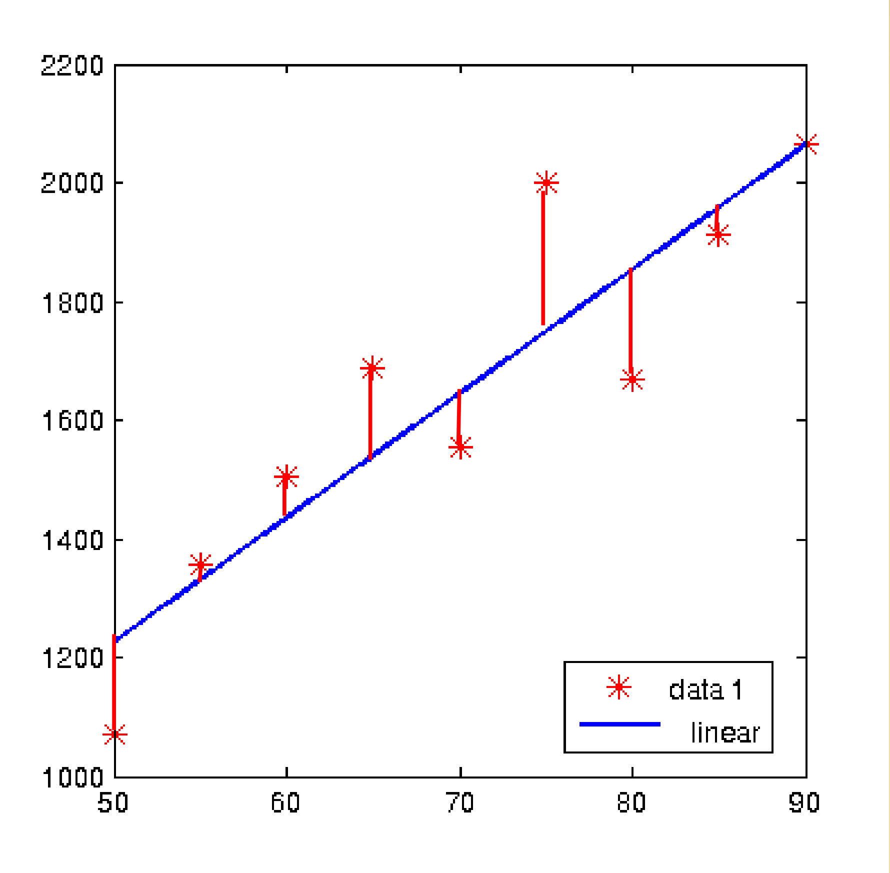
- Least Squares Regression:
  - find a function that minimizes the prediction error
  - better for noisy data.

Recap Least-Squares Method

- Minimize sum of the squares of the errors

$$\tilde{y} = \tilde{f}(x) = a_0 + a_1 x$$

$$SSE(\tilde{f}) = \sum_{i=0}^{n} [\tilde{f}(x_i) - y_i]^2$$



- pick the $\tilde{f}$ with min. SSE
  (that means: pick $a_{0,} a_1$)

This Lecture

- Last week: *labeled* data (also 'supervised learning')
  - data: (x,y)-pairs
- This week: *unlabeled* data (also 'unsupervised learning')
  - data: just x

- Finding structure in data
- 2 Main methods:
  - Clustering
  - Principle Components analysis (PCA)

Part 1: Clustering

Clustering

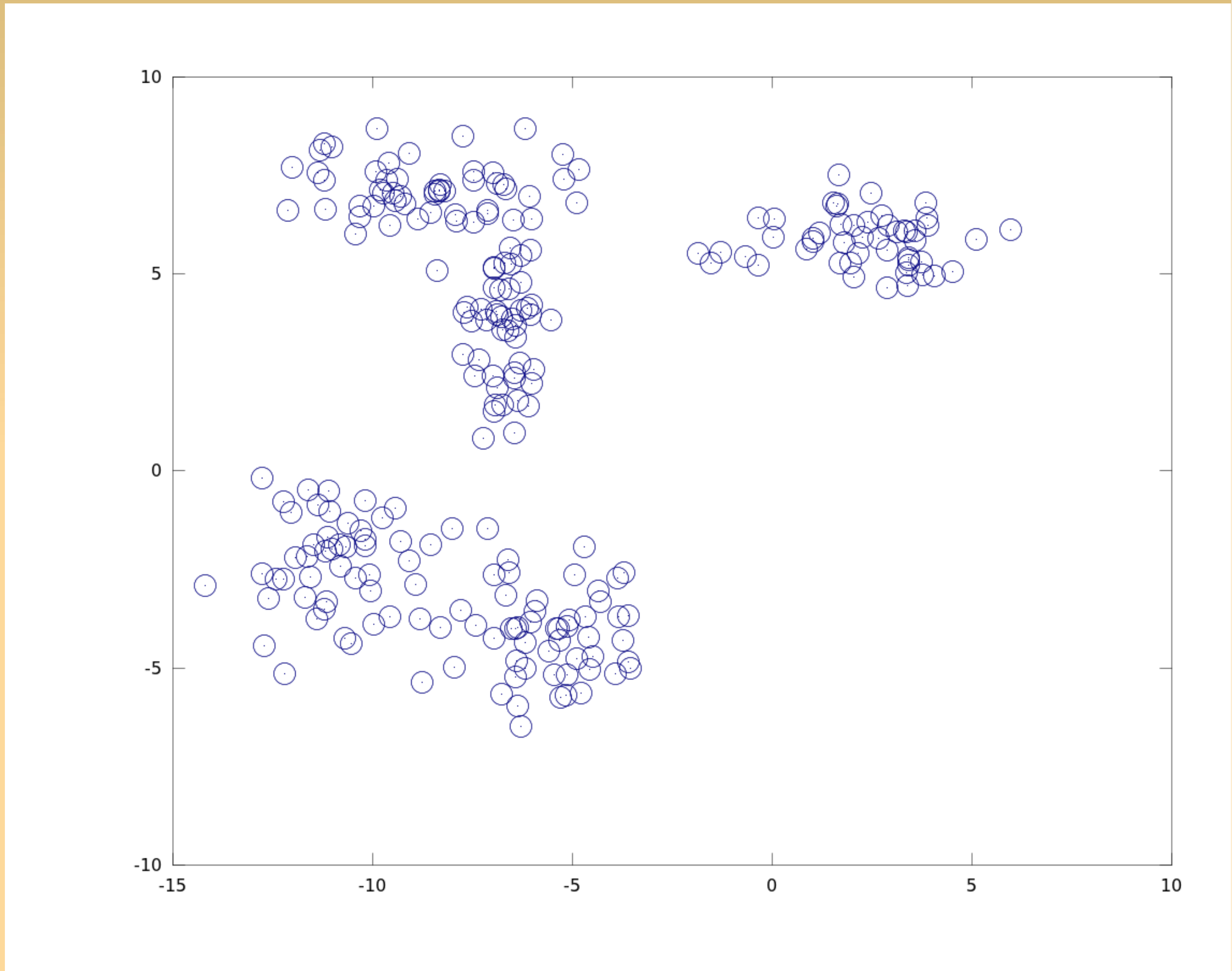
- data set

$$\{(x^{(0)}, y^{(0)}), \dots, (x^{(n)}, y^{(n)})\}$$

- but now: unlabeled

$$\{(x_1^{(0)}, x_2^{(0)}), \dots, (x_1^{(n)}, x_2^{(n)})\}$$

- now what?
  - structure?
  - summarize this data?



Clustering

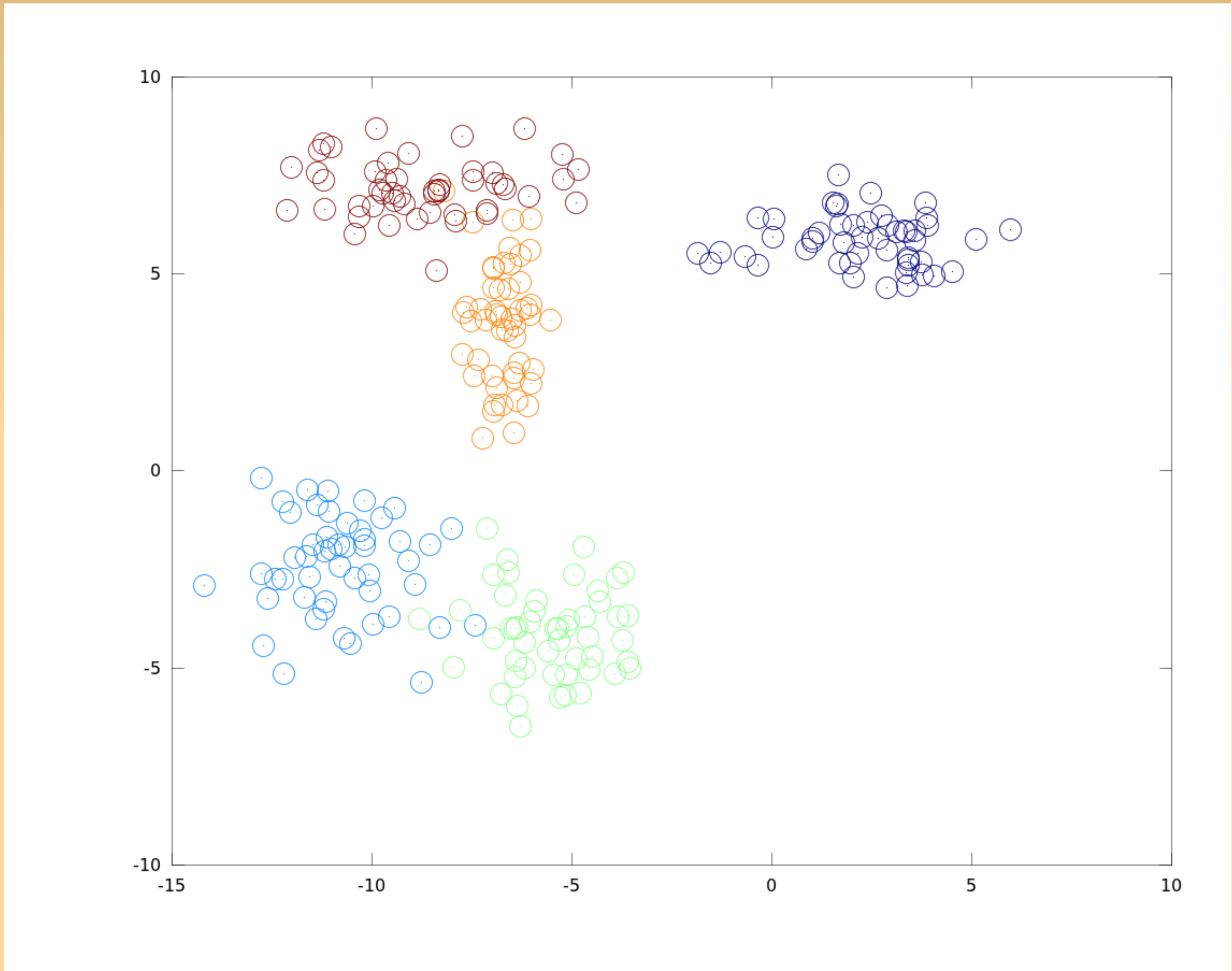
- data set

$$\{(x^{(0)}, y^{(0)}), \ldots, (x^{(n)}, y^{(n)})\}$$

- but now: unlabeled

$$\{(x_1^{(0)}, x_2^{(0)}), \ldots, (x_1^{(n)}, x_2^{(n)})\}$$

- now what?
  - structure?
  - summarize this data?

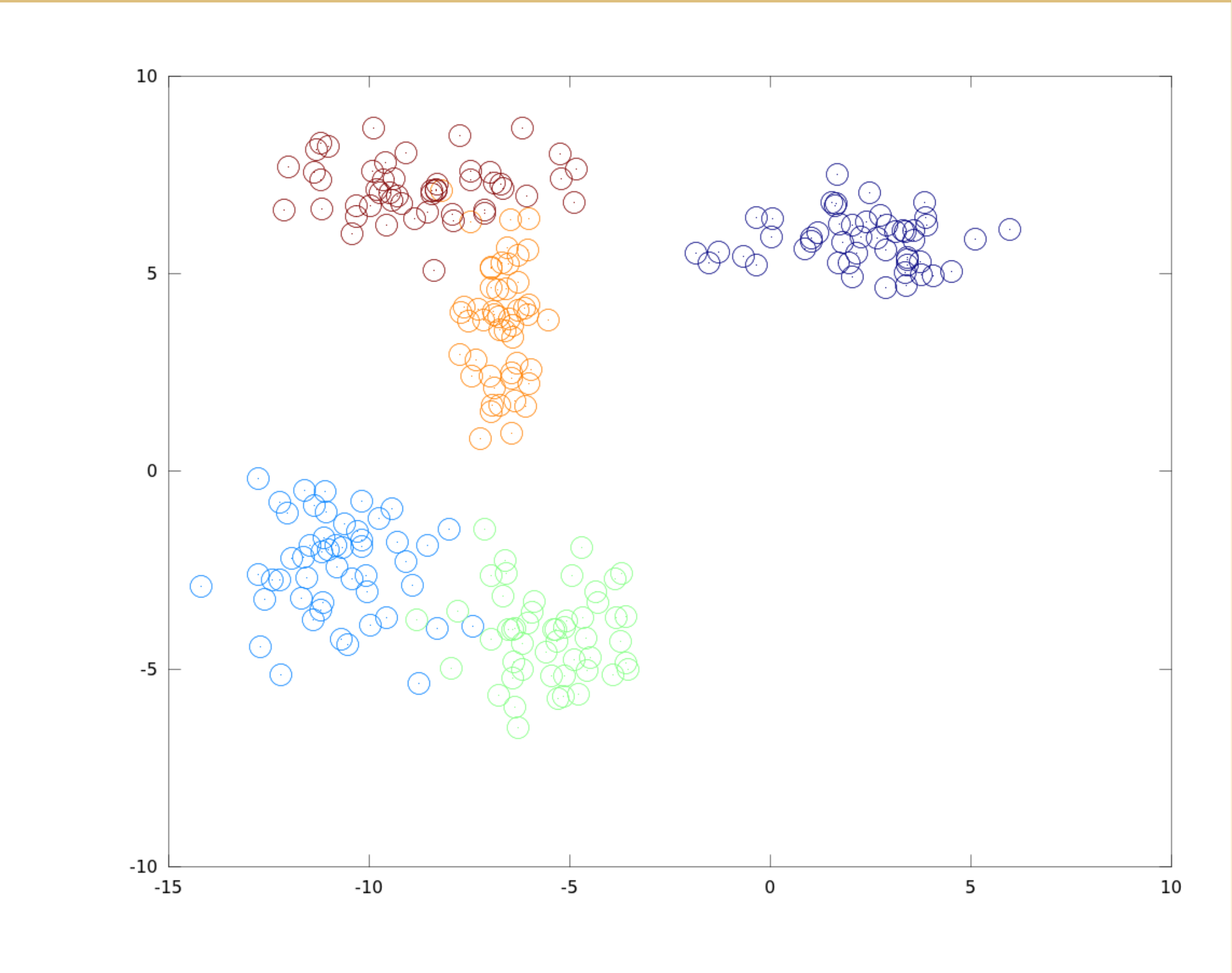


Clustering

- data set

$$\{(x_1^{(0)}, x_2^{(0)}), \ldots, (x_1^{(n)}, x_2^{(n)})\}$$

- try to find the different clusters!
- How?

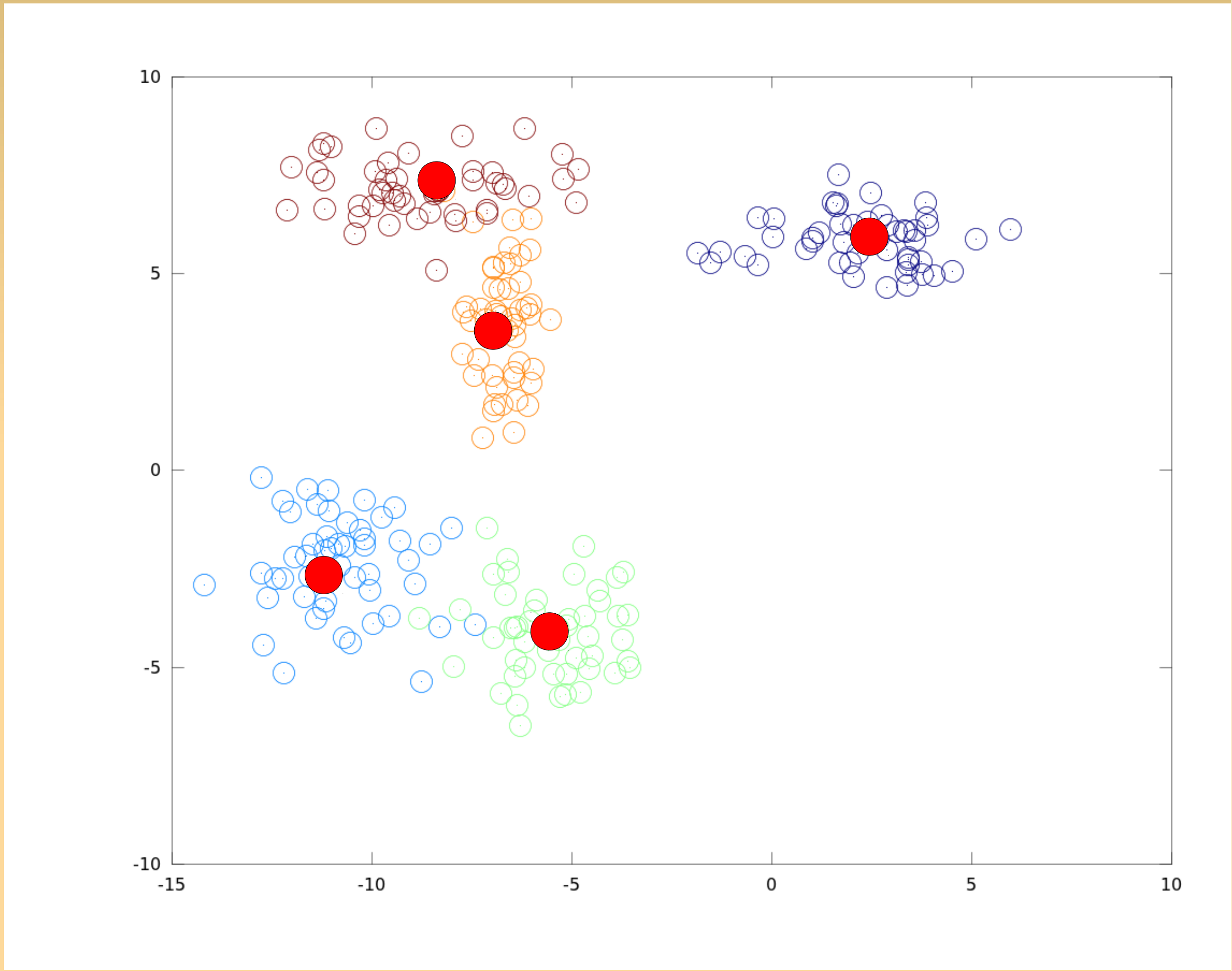


Clustering

- data set

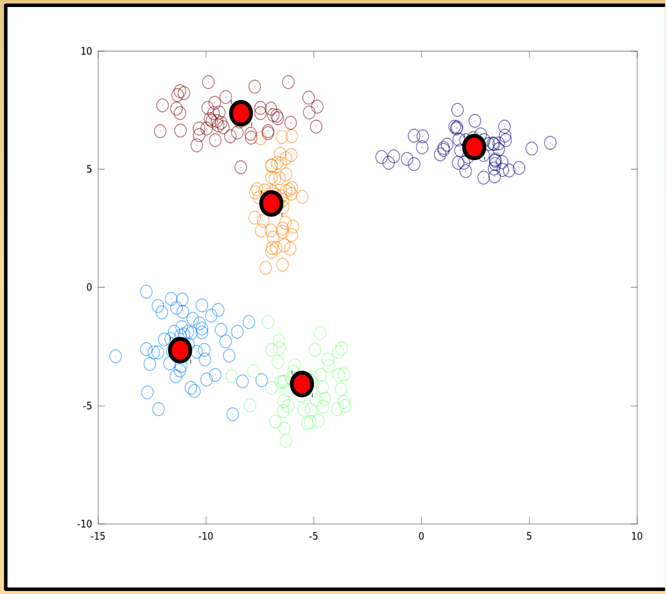
$$\{(x_1^{(0)}, x_2^{(0)}), \dots, (x_1^{(n)}, x_2^{(n)})\}$$

- try to find the different clusters!
- One way:
  - find centroids



Clustering – Applications

- *Clustering* or *Cluster Analysis* has many applications
- Understanding
  - Astronomy: new types of stars
  - Biology:
    - create taxonomies of living things
    - clustering based on genetic information
  - Climate: find patterns in the atmospheric pressure
  - etc.
- Data (pre)processing
  - summarization of data set
  - compression



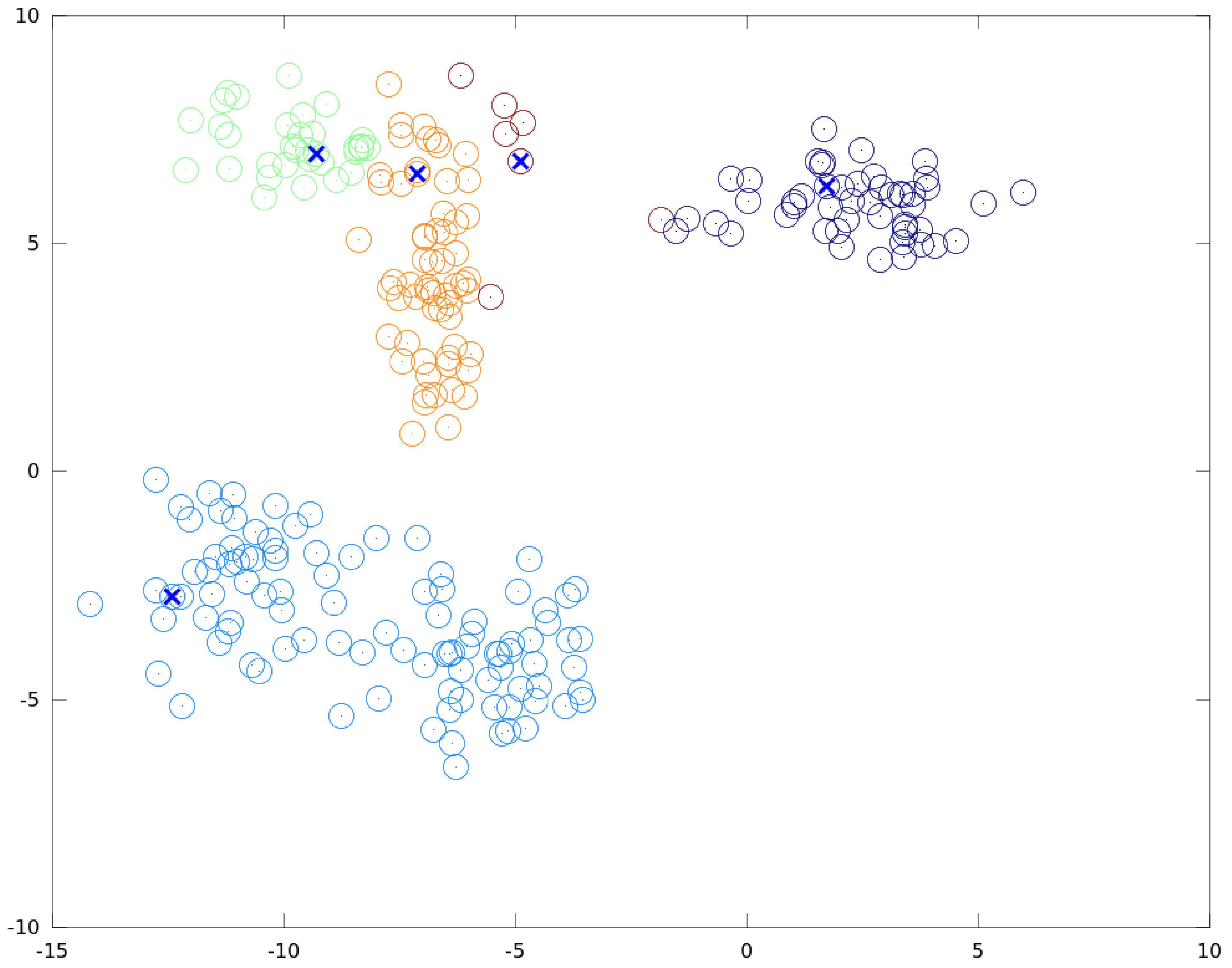
Cluster Methods

- Many types of clustering!
- We will treat one method: k-Means clustering
  - the standard text-book method
  - not necessarily the best
  - but the simplest
- You will implement k-Means
  - Use it to compress an image

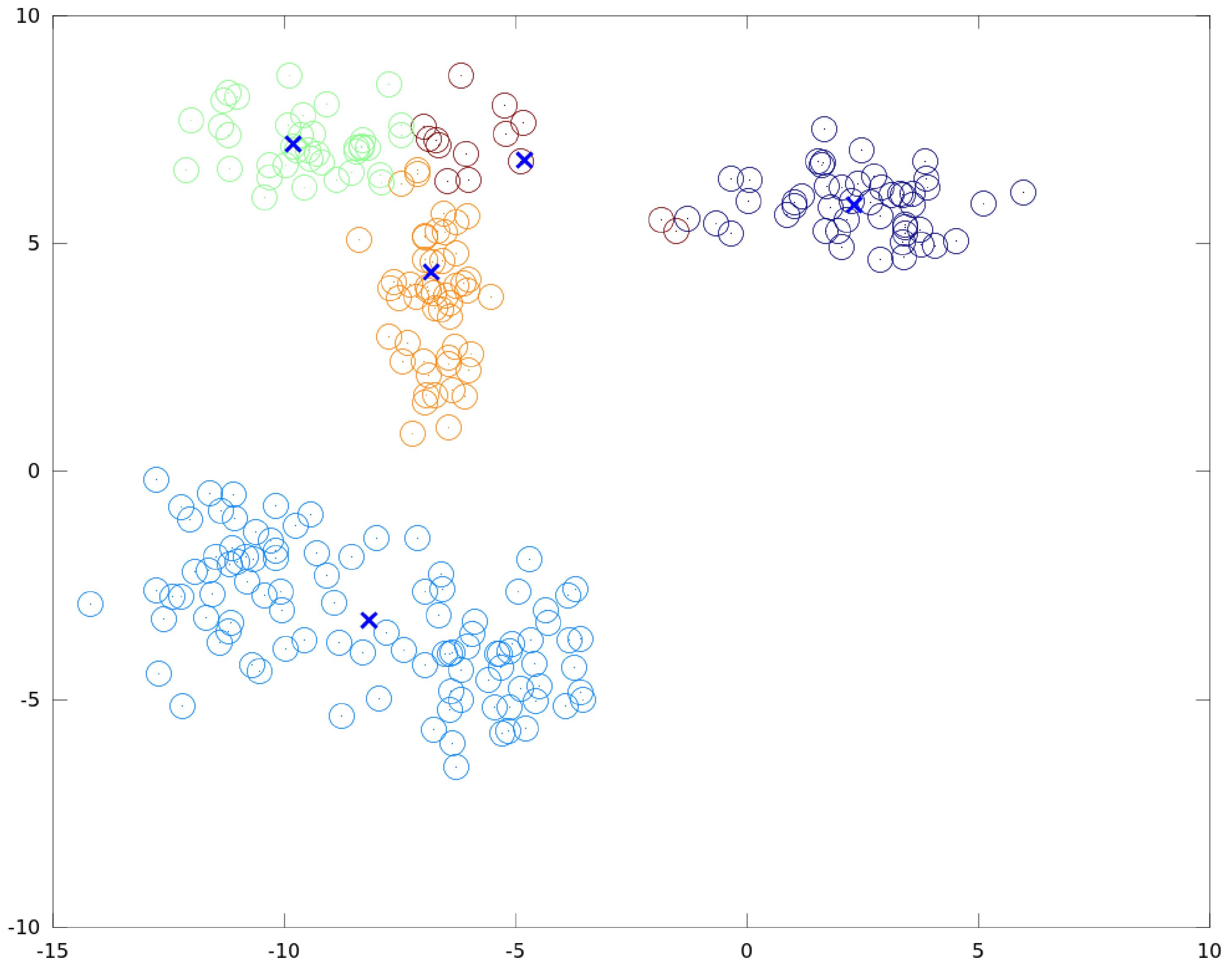
k-Means Clustering

- The main idea
  - clusters are represented by 'centroids'
  - start with random centroids
  - then repeatedly
    - find all data points that are nearest to a centroid
    - update each centroid based on its data points

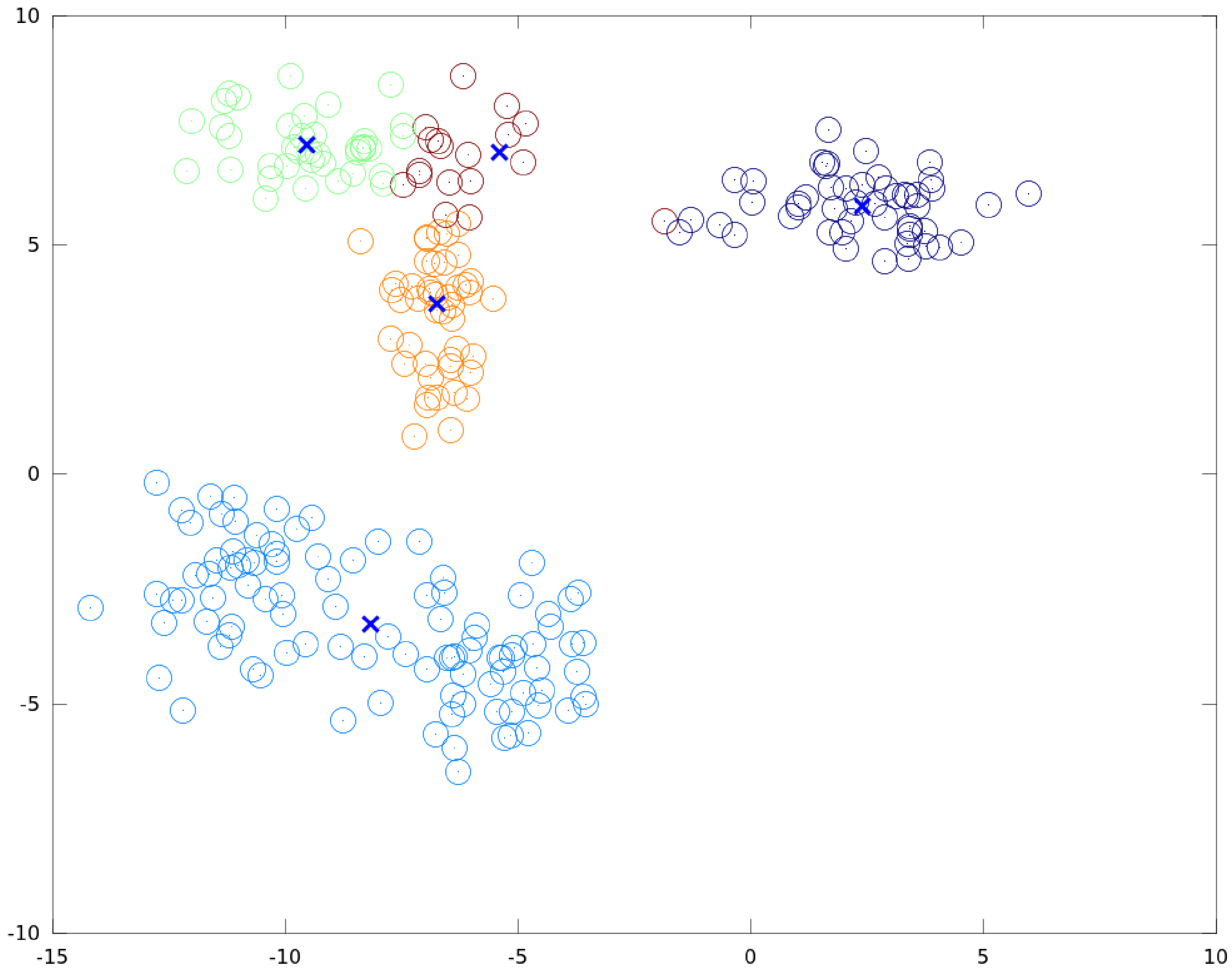
k-Means Clustering: Example



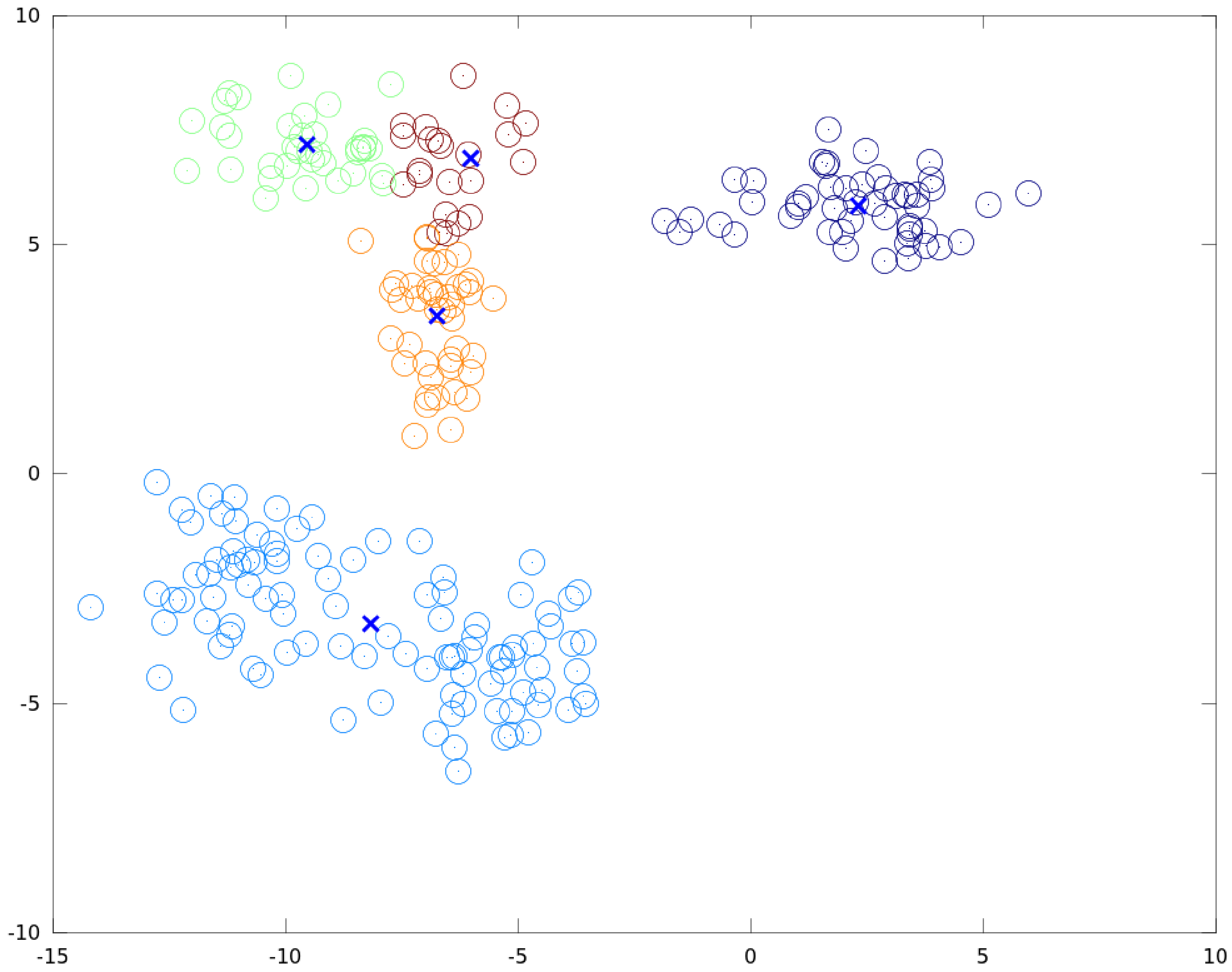
k-Means Clustering: Example



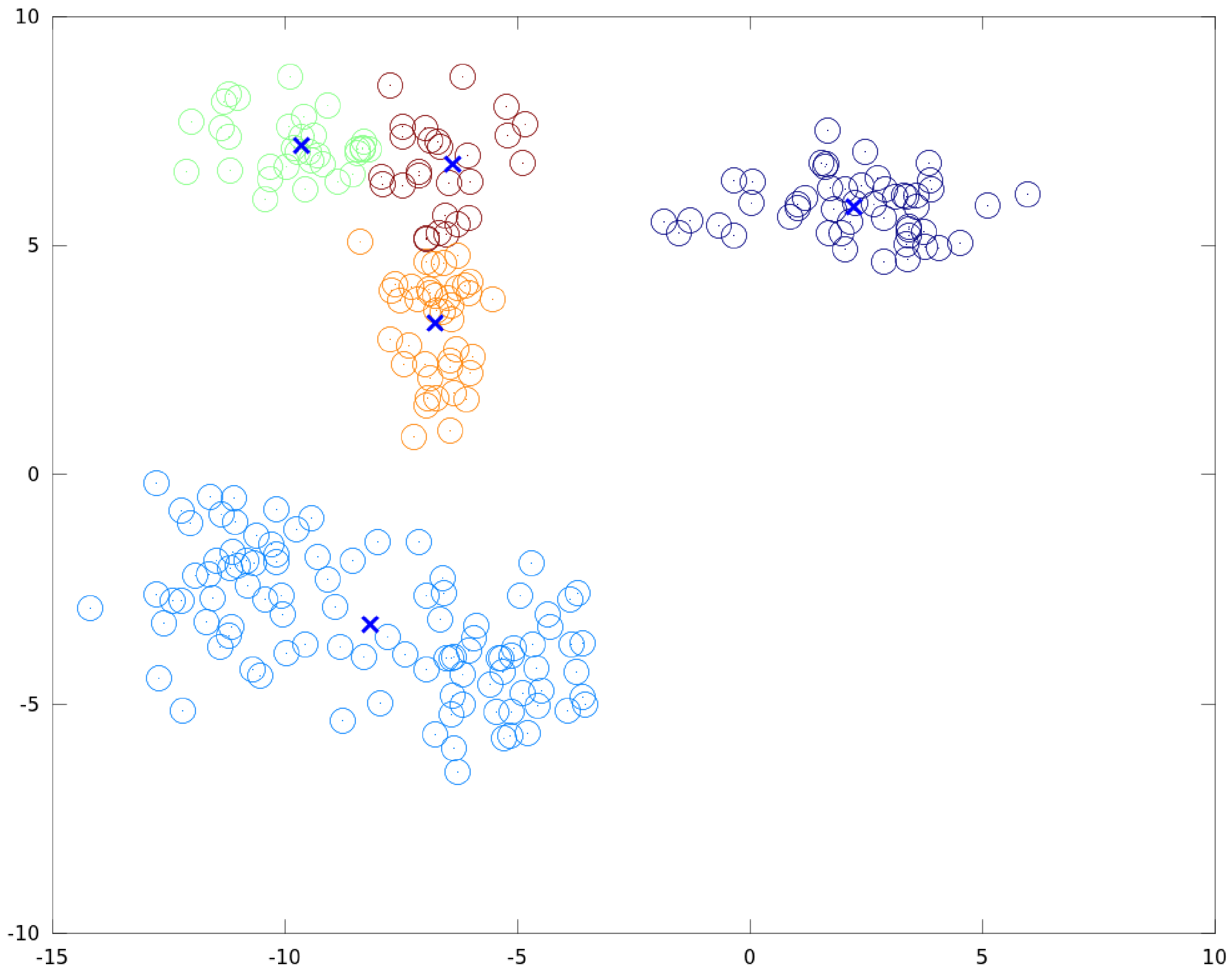
k-Means Clustering: Example



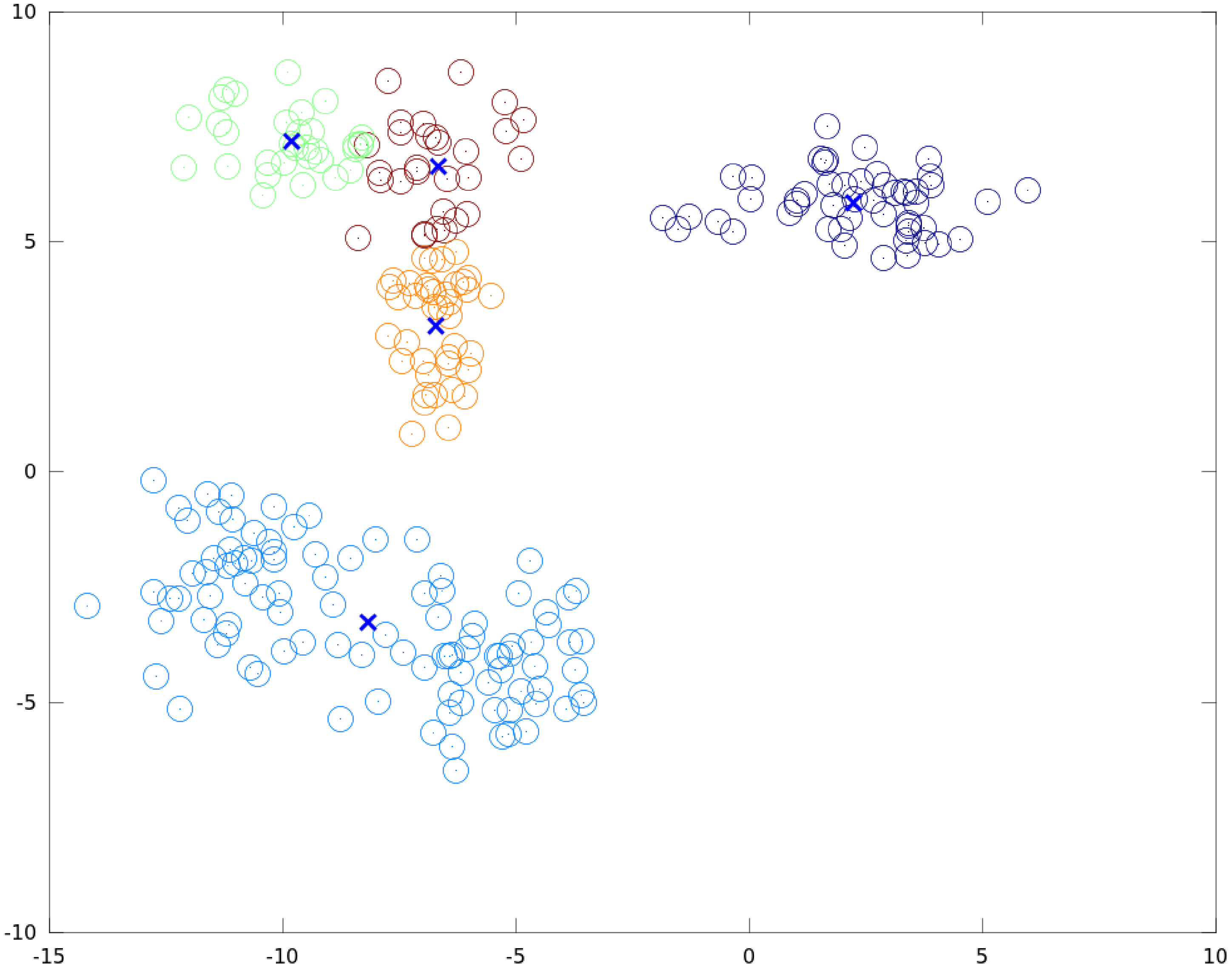
k-Means Clustering: Example



k-Means Clustering: Example



k-Means Clustering: Example



k-Means Algorithm

```
%% k-means PSEUDO CODE
%
% X              - the data
% centroids      - initial centroids
%                  (given by random initialization on data points)

iterations = 1
done = 0
while (~done && iterations < max_iters)

    labels = NearestCentroids(X, centroids);
    centroids = UpdateCentroids(X, labels);

    iterations = iterations + 1;
    if centroids did not change
        done = 1
    end
end
```

Part 2: Principal Component Analysis

Dimension Reduction

- Clustering allows us to summarize data using centroids
  - summary of a point: what cluster is belongs to.

- Different idea: $(x_1, x_2, \ldots, x_D) \rightarrow (z_1, z_2, \ldots, z_d)$
  - reduce the number of variables
  - i.e., reduce the number of dimensions from D to d
    $d < D$

Dimension Reduction

- Clustering allows us to summarize data using centroids
  - summary of a point: what cluster is belongs to.

- Different idea: $(x_{1,} x_{2,} \dots, x_D) \rightarrow (z_{1,} z_{2,} \dots, z_d)$
  - reduce the number of variables
  - i.e., reduce the number of dimensions from D to d
    $d < D$

This is what **Principal Component Analysis (PCA)** does.

PCA – Goals

$N = n + 1$

- Given a data set X of N data point of D variables
  → convert to data set Z of N data points of d variables

$$\left(x_1^{(0)}, x_2^{(0)}, \ldots, x_D^{(0)}\right) \rightarrow \left(z_1^{(0)}, z_2^{(0)}, \ldots, z_d^{(0)}\right)$$
$$\left(x_1^{(1)}, x_2^{(1)}, \ldots, x_D^{(1)}\right) \rightarrow \left(z_1^{(1)}, z_2^{(1)}, \ldots, z_d^{(1)}\right)$$
$$\ldots$$
$$\left(x_1^{(n)}, x_2^{(n)}, \ldots, x_D^{(n)}\right) \rightarrow \left(z_1^{(n)}, z_2^{(n)}, \ldots, z_d^{(n)}\right)$$

PCA – Goals

- Given a data set X of N data point of D variables
  → convert to data set Z of N data points of d variables

$$\left(x_1^{(0)}, x_2^{(0)}, \ldots, x_D^{(0)}\right) \rightarrow \left(z_1^{(0)}, z_2^{(0)}, \ldots, z_d^{(0)}\right)$$
$$\left(x_1^{(1)}, x_2^{(1)}, \ldots, x_D^{(1)}\right) \rightarrow \left(z_1^{(1)}, z_2^{(1)}, \ldots, z_d^{(1)}\right)$$
$$\ldots$$
$$\left(x_1^{(n)}, x_2^{(n)}, \ldots, x_D^{(n)}\right) \rightarrow \left(z_1^{(n)}, z_2^{(n)}, \ldots, z_d^{(n)}\right)$$

The vector $\left(z_i^{(0)}, z_i^{(1)}, \ldots, z_i^{(n)}\right)$

is called the *i*-th **principal component** (of the data set)

PCA – Goals

- Given a data set X of N data point of D variables
  → convert to data set Z of N data points of d variables

$$\left(x_1^{(0)}, x_2^{(0)}, \ldots, x_D^{(0)}\right) \rightarrow \left(z_1^{(0)}, z_2^{(0)}, \ldots, z_d^{(0)}\right)$$
$$\left(x_1^{(1)}, x_2^{(1)}, \ldots, x_D^{(1)}\right) \rightarrow \left(z_1^{(1)}, z_2^{(1)}, \ldots, z_d^{(1)}\right)$$
$$\ldots$$
$$\left(x_1^{(n)}, x_2^{(n)}, \ldots, x_D^{(n)}\right) \rightarrow \left(z_1^{(n)}, z_2^{(n)}, \ldots, z_d^{(n)}\right)$$

The vector $\left(z_i^{(0)}, z_i^{(1)}, \ldots, z_i^{(n)}\right)$

is called the *i*-th **principal component** (of the data set)

- PCA performs a **linear** transformation:
  → variables z_i are linear combinations of $x_1, \ldots, x_D$

PCA Goals – 2

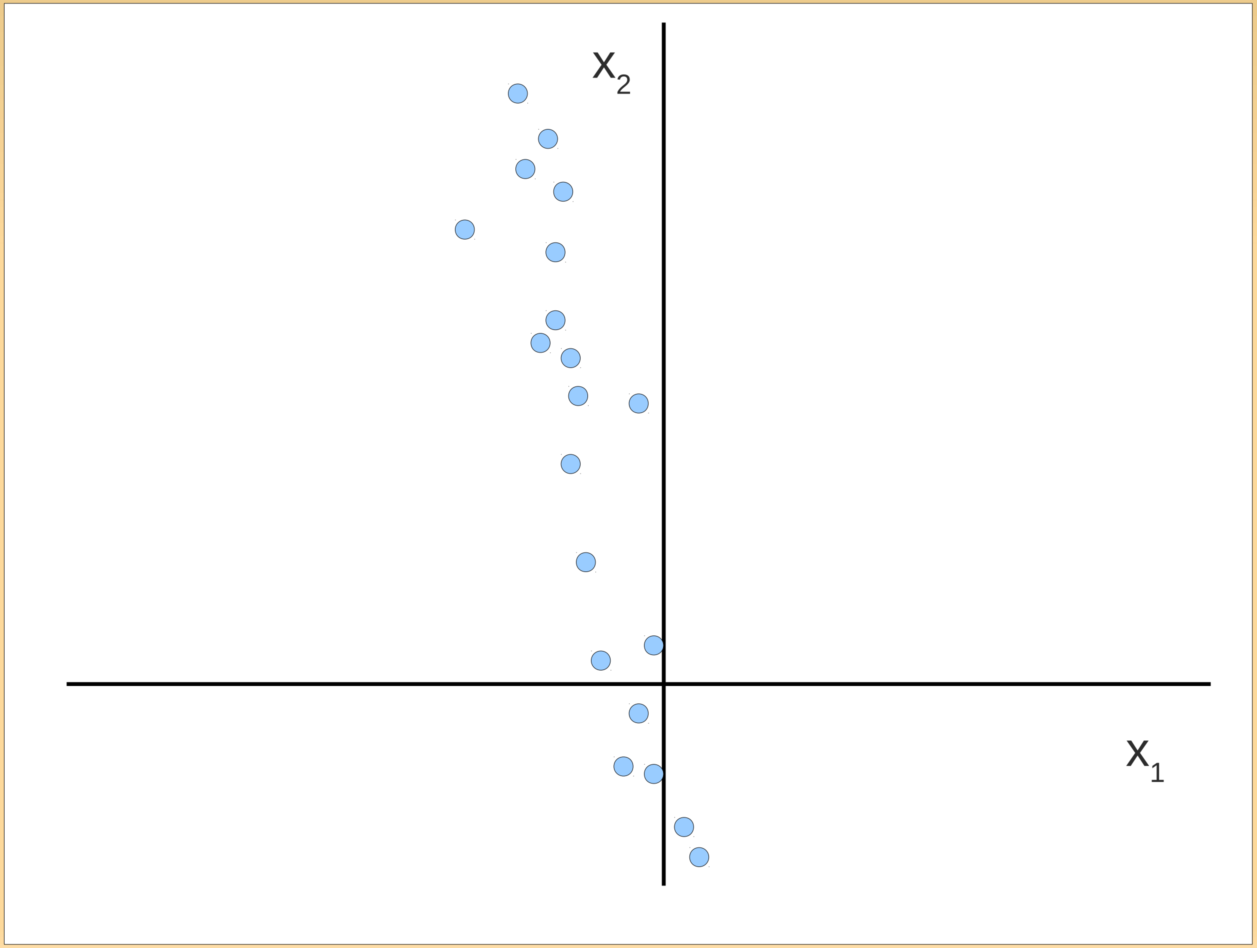
- Of course many possible transformations possible...
  - Reducing the number of variables: loss of information
  - PCA makes this loss minimal

- PCA is very useful
  - Exploratory analysis of the data
  - Visualization of high-D data
  - Data preprocessing
  - Data compression

PCA – Intuition

- How would you summarize this data using 1 dimension?

    (what variable contains the most information?)



PCA – Intuition

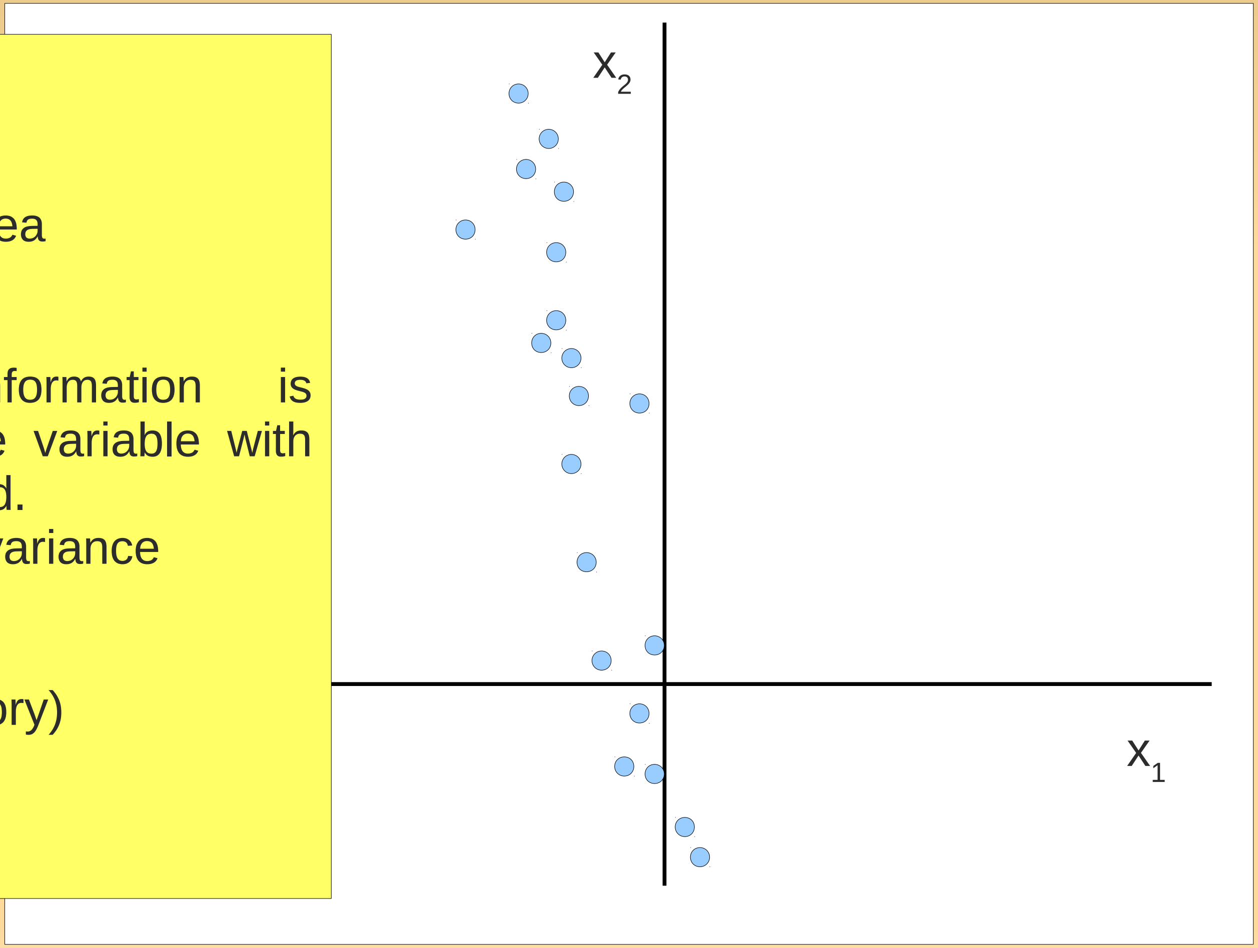
- How would you summarize this data using 1 dimension?

  (what variable contains the most information?)

Very important idea

The most information is contained by the variable with the largest spread.
- i.e., highest variance

(Information Theory)



PCA – Intuition

- How would you summarize this data using 1 dimension?

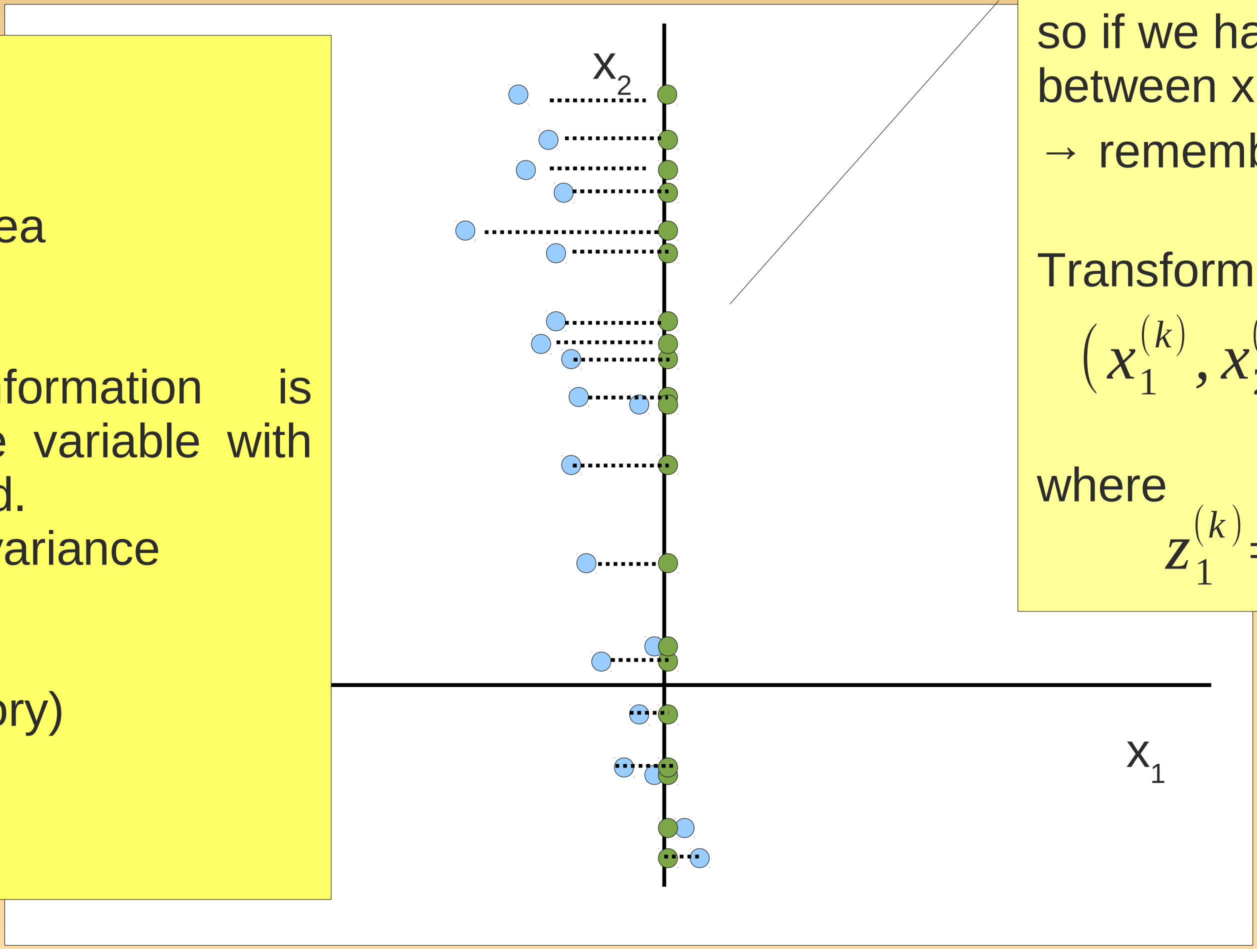
  (what variable contains the most information?)

Very important idea

The most information is contained by the variable with the largest spread.

- i.e., highest variance

(Information Theory)



so if we have to chose between x_1 and x_2
→ remember x_2

Transform of *k*-th point:

$$\left(x_1^{(k)}, x_2^{(k)}\right) \rightarrow \left(z_1^{(k)}\right)$$

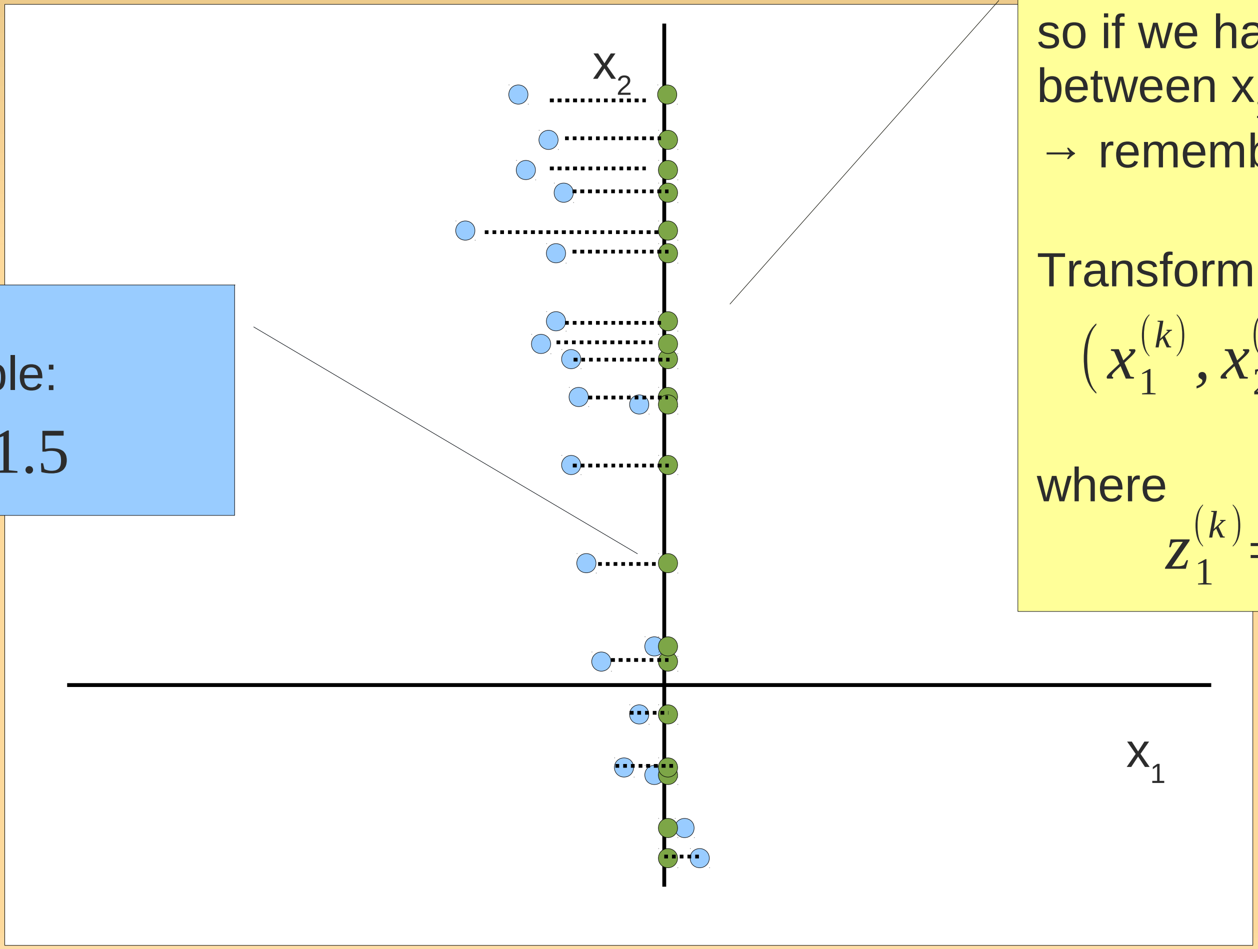
where

$$z_1^{(k)} = x_2^{(k)}$$

PCA – Intuition

- How would you summarize this data using 1 dimension?

(what variable contains the most information?)



Example:

$z_1^{(k)} = 1.5$

so if we have to chose between x_1 and x_2
→ remember x_2

Transform of *k*-th point:

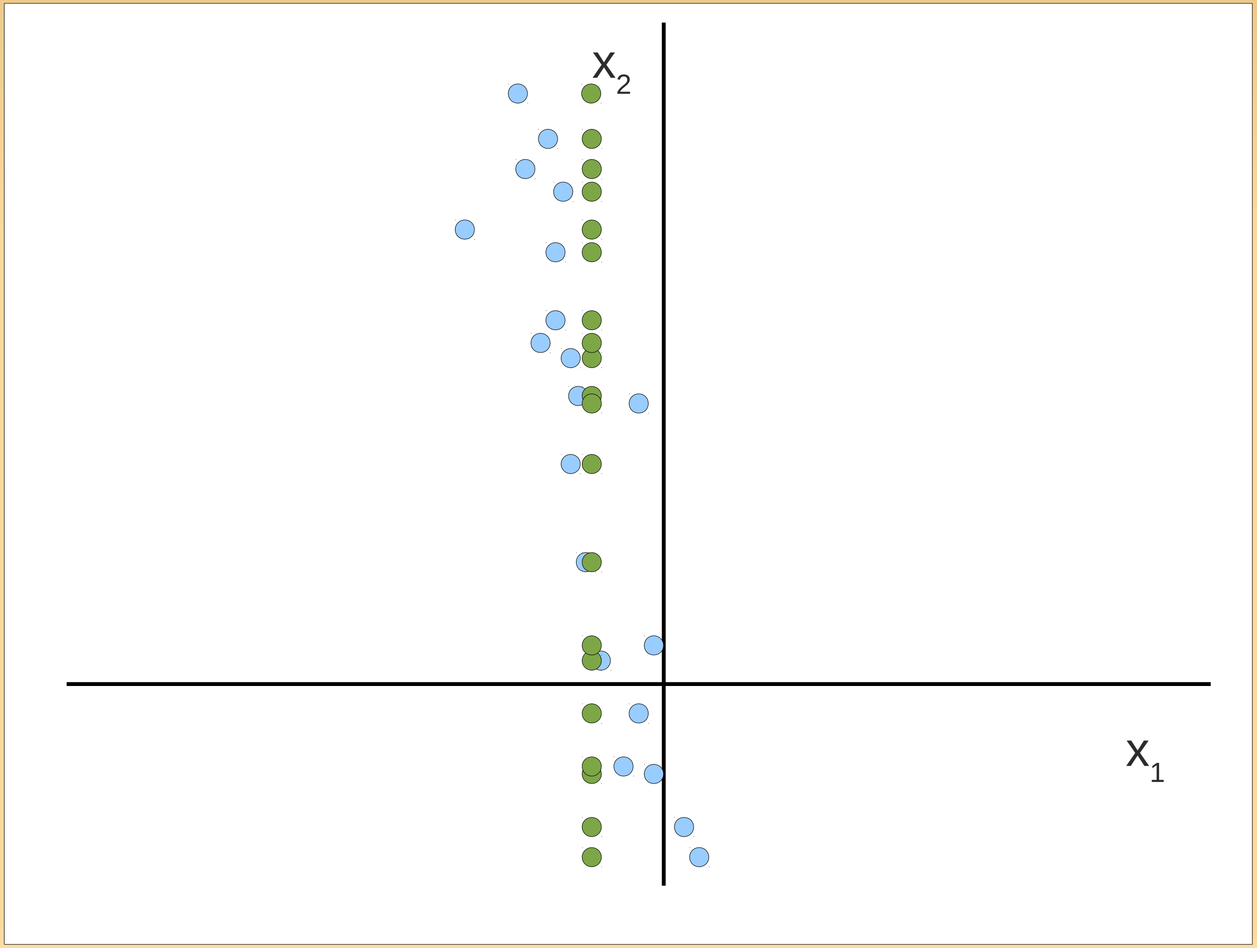
$$\left(x_1^{(k)}, x_2^{(k)}\right) \rightarrow \left(z_1^{(k)}\right)$$

where

$$z_1^{(k)} = x_2^{(k)}$$

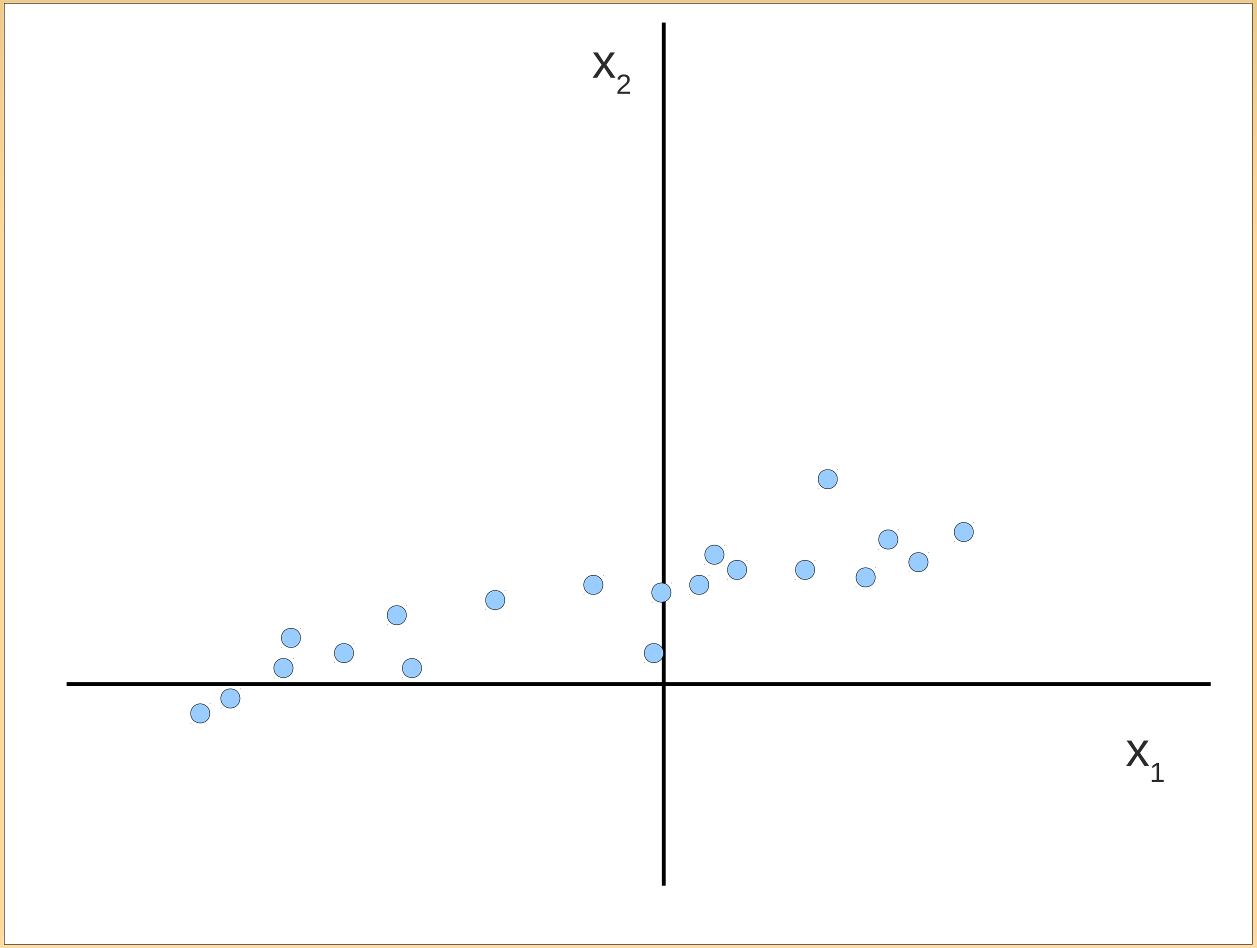
PCA – Intuition

- Reconstruction based on x_2
  → only need to remember mean of x_1



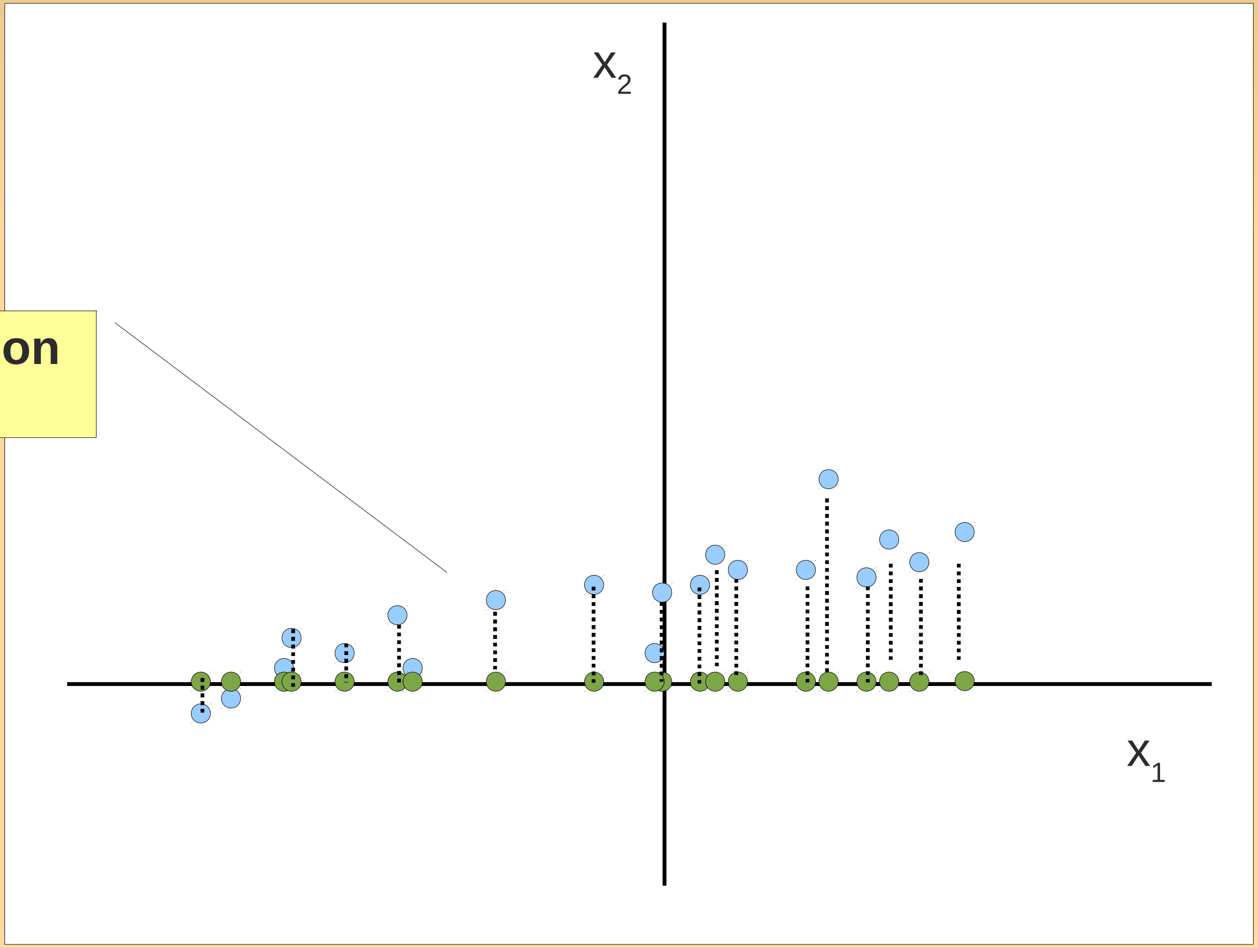
PCA – Intuition

- How would you summarize this data using 1 dimension?



PCA – Intuition

- How would you summarize this data using 1 dimension?



This is a **projection** on the x1 axis.

Question

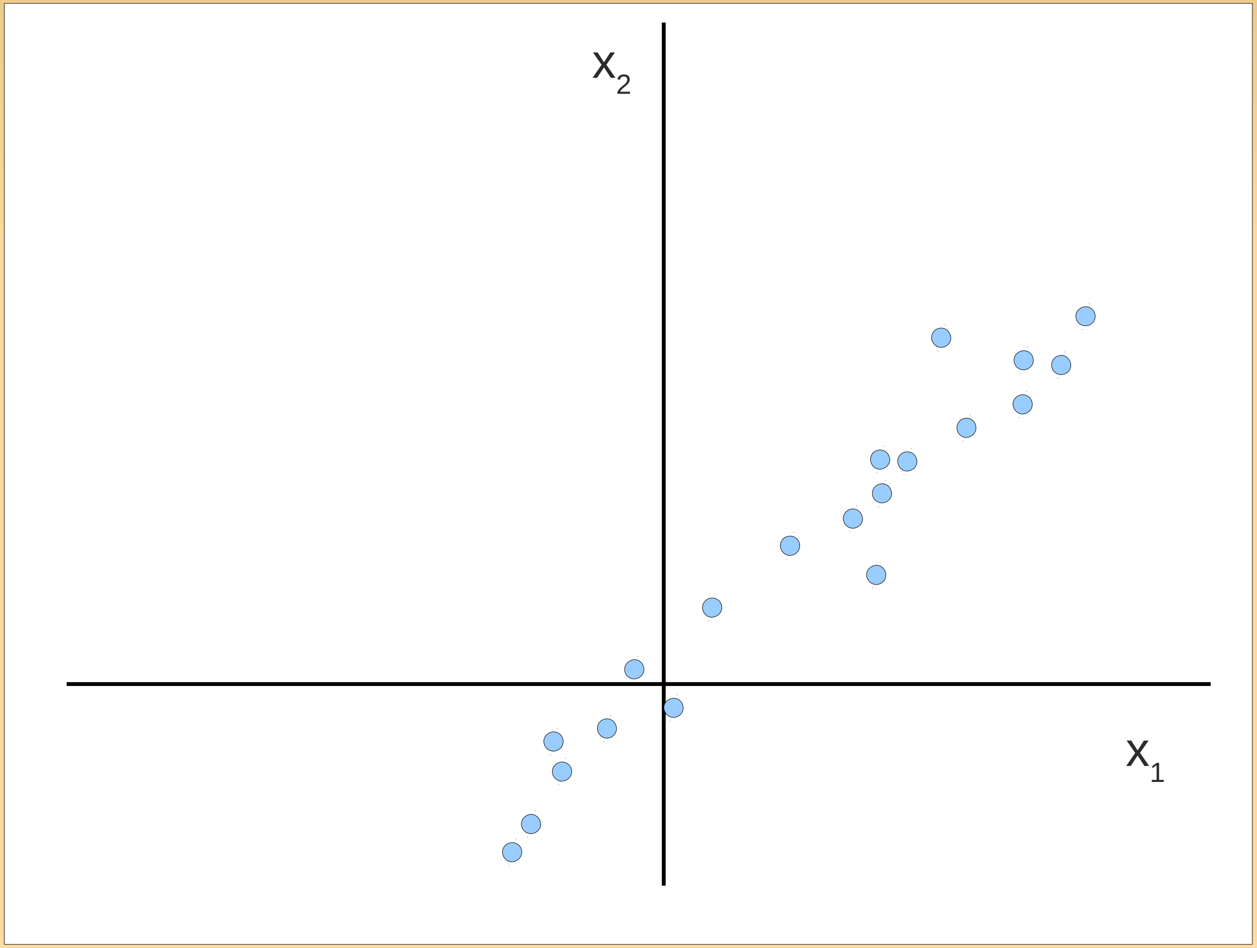
- Suppose the data is now 3-dimensional
  - $x = (x_{1,} x_{2,} x_3)$
- Can you think of an example where we could project it to 2 dimensions:

$$(x_{1,} x_{2,} x_3) \rightarrow (z_{1,} z_2)$$

?

PCA – Intuition

- How would you summarize this data using 1 dimension?

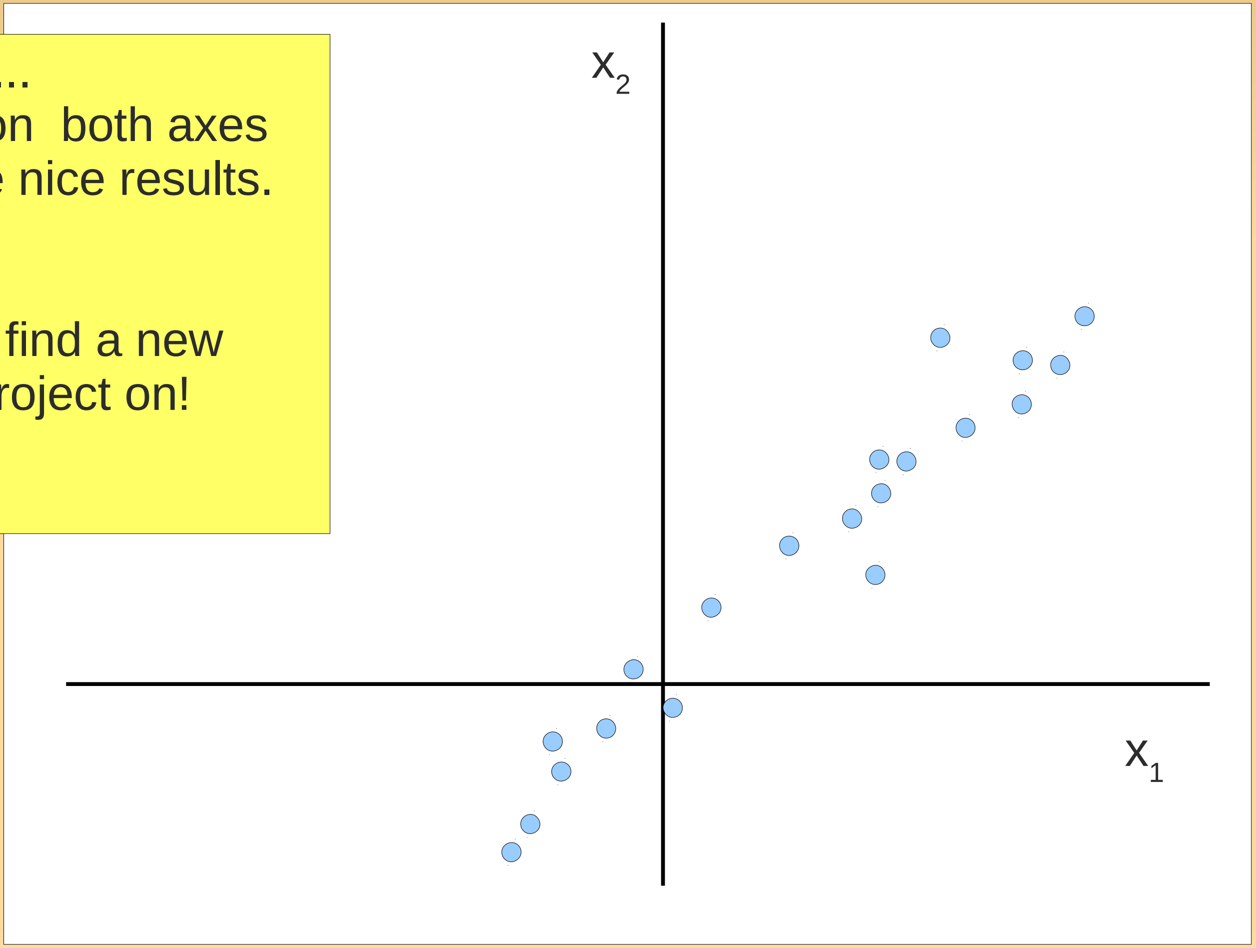


PCA – Intuition

- How would you summarize this data using 1 dimension?

- More difficult...
  ...projection on both axes does not give nice results.

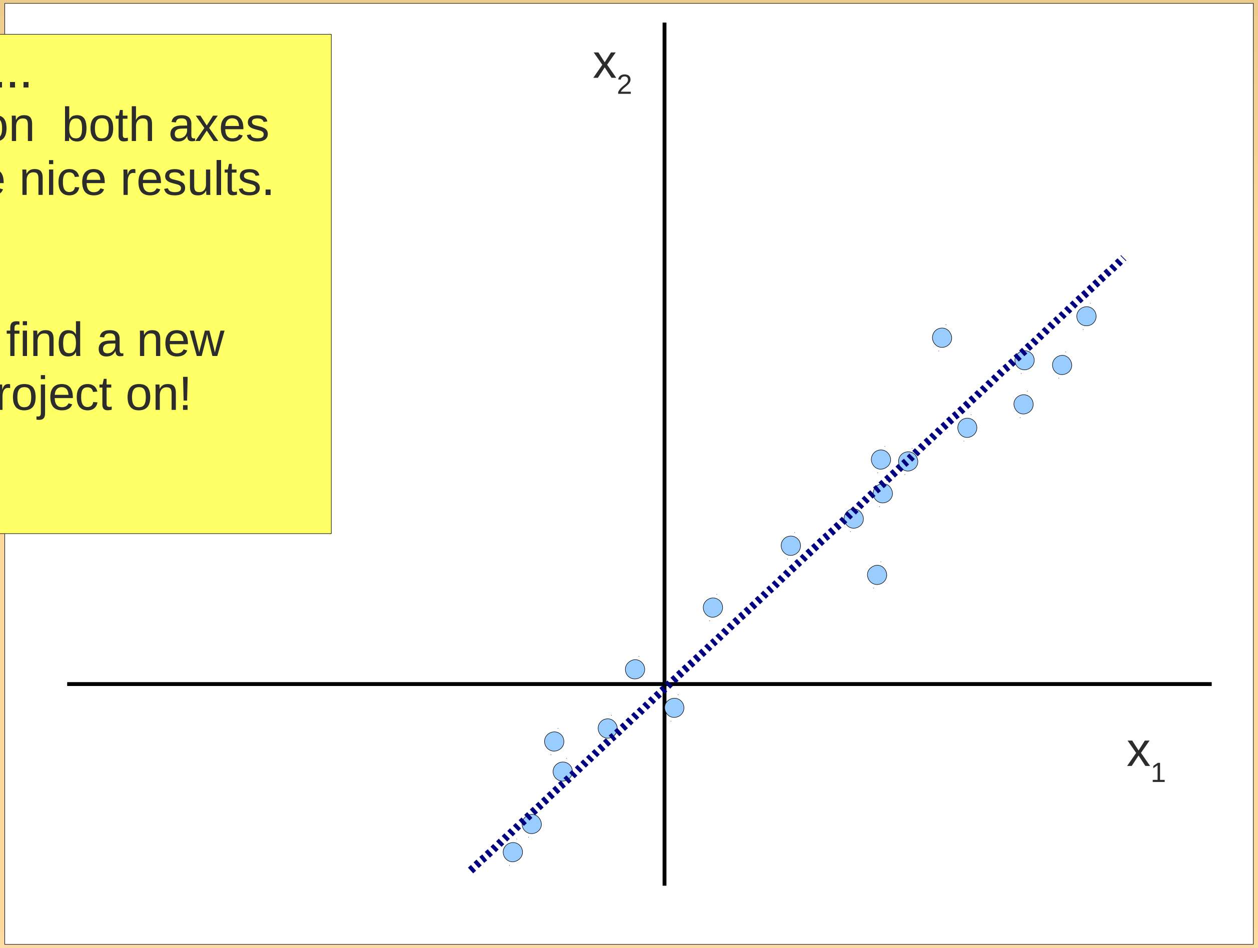
- Idea of PCA: find a new direction to project on!



PCA – Intuition

- How would you summarize this data using 1 dimension?

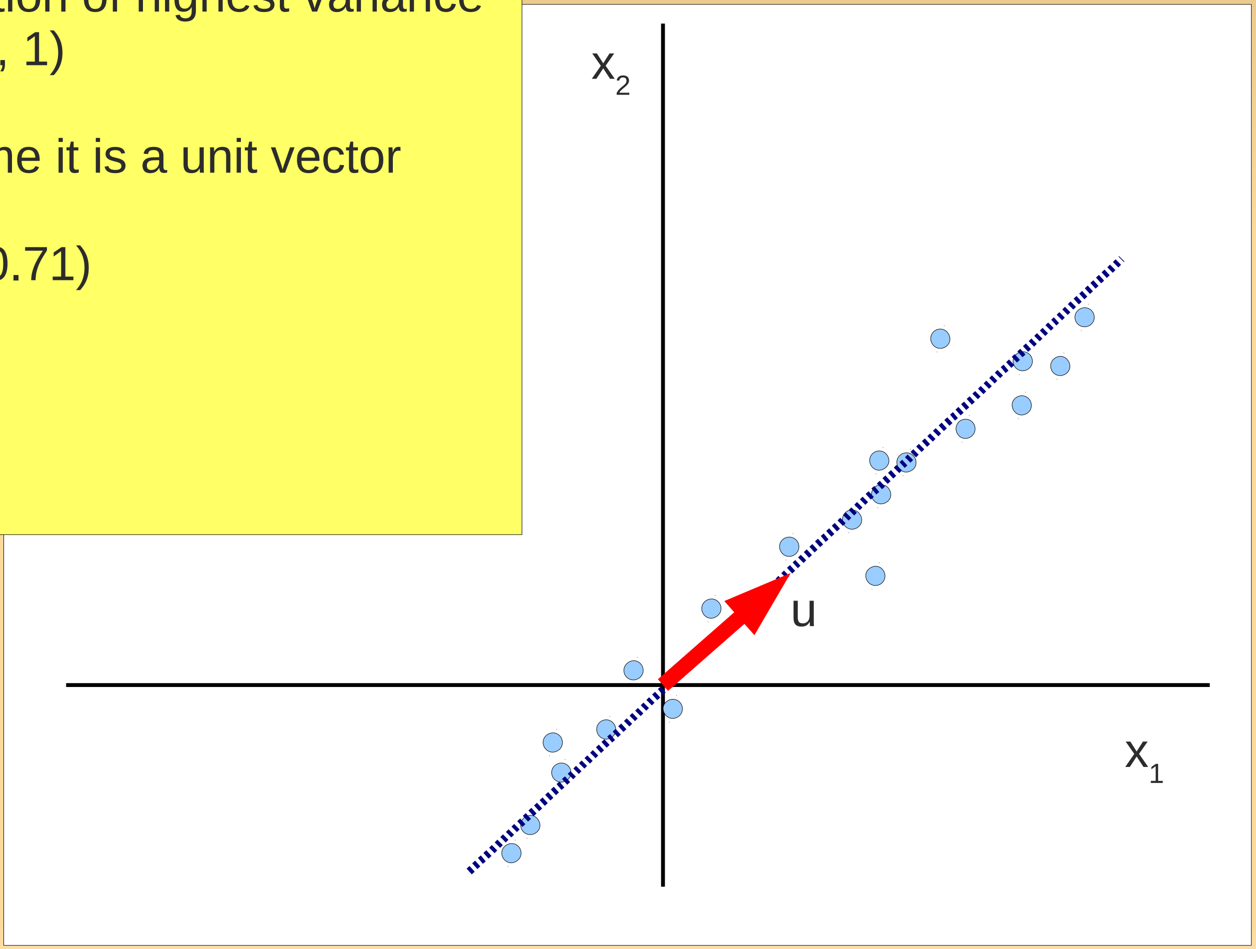
- More difficult...
  ...projection on both axes does not give nice results.
- Idea of PCA: find a new direction to project on!



PCA – Intuition

- How would you summarize this data using 1 dimension?

- u is the direction of highest variance
  - e.g., u = (1, 1)
- we will assume it is a unit vector
  - length = 1
  - u = (0.71, 0.71)



PCA – Intuition

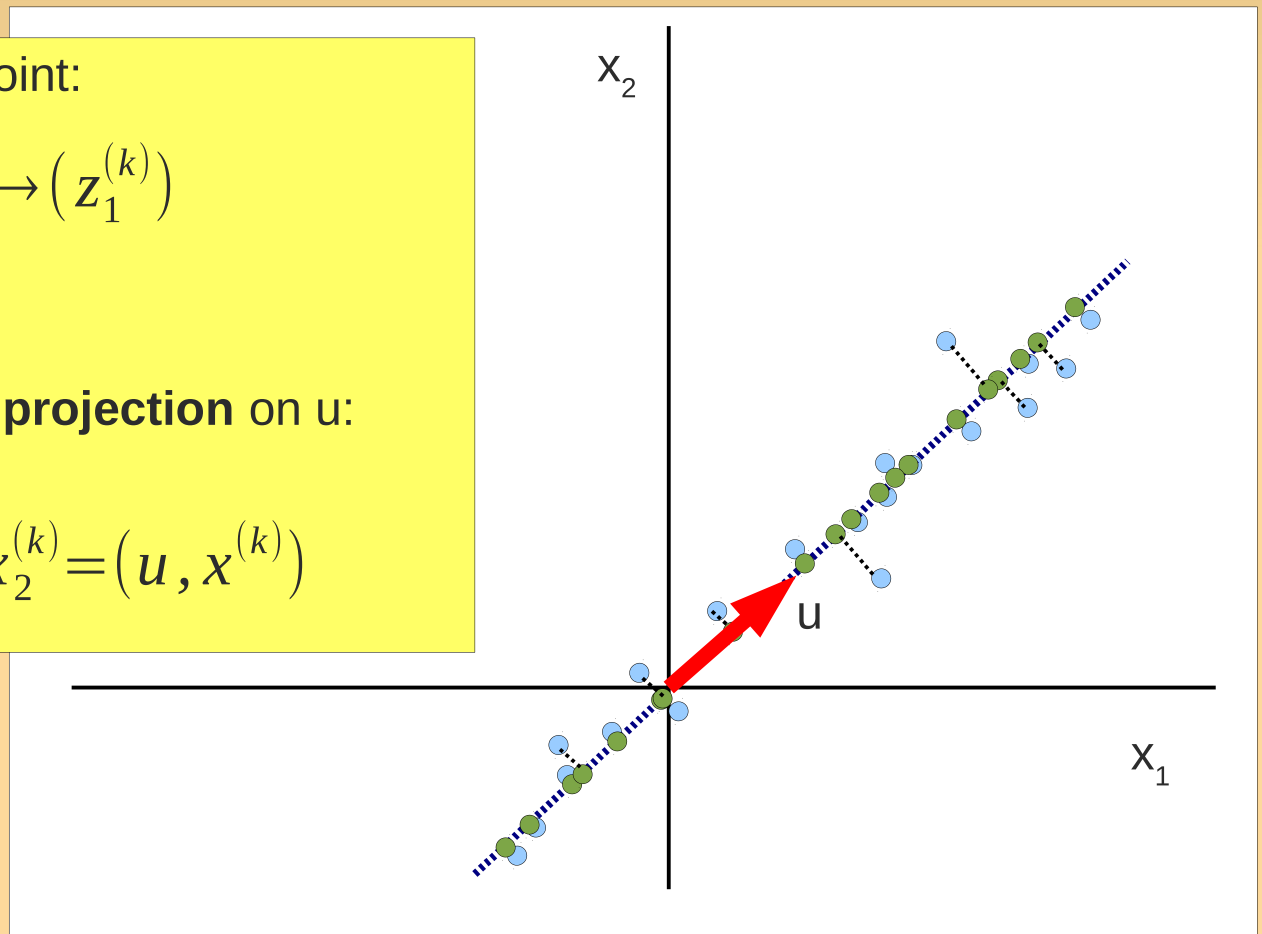
- How would you summarize this data using 1 dimension?

Transform of *k*-th point:

$$\left(x_1^{(k)}, x_2^{(k)}\right) \rightarrow \left(z_1^{(k)}\right)$$

where z_1 is the
orthogonal scalar projection on u:

$$z_1^{(k)} = u_1 x_1^{(k)} + u_2 x_2^{(k)} = \left(u, x^{(k)}\right)$$



PCA – Intuition

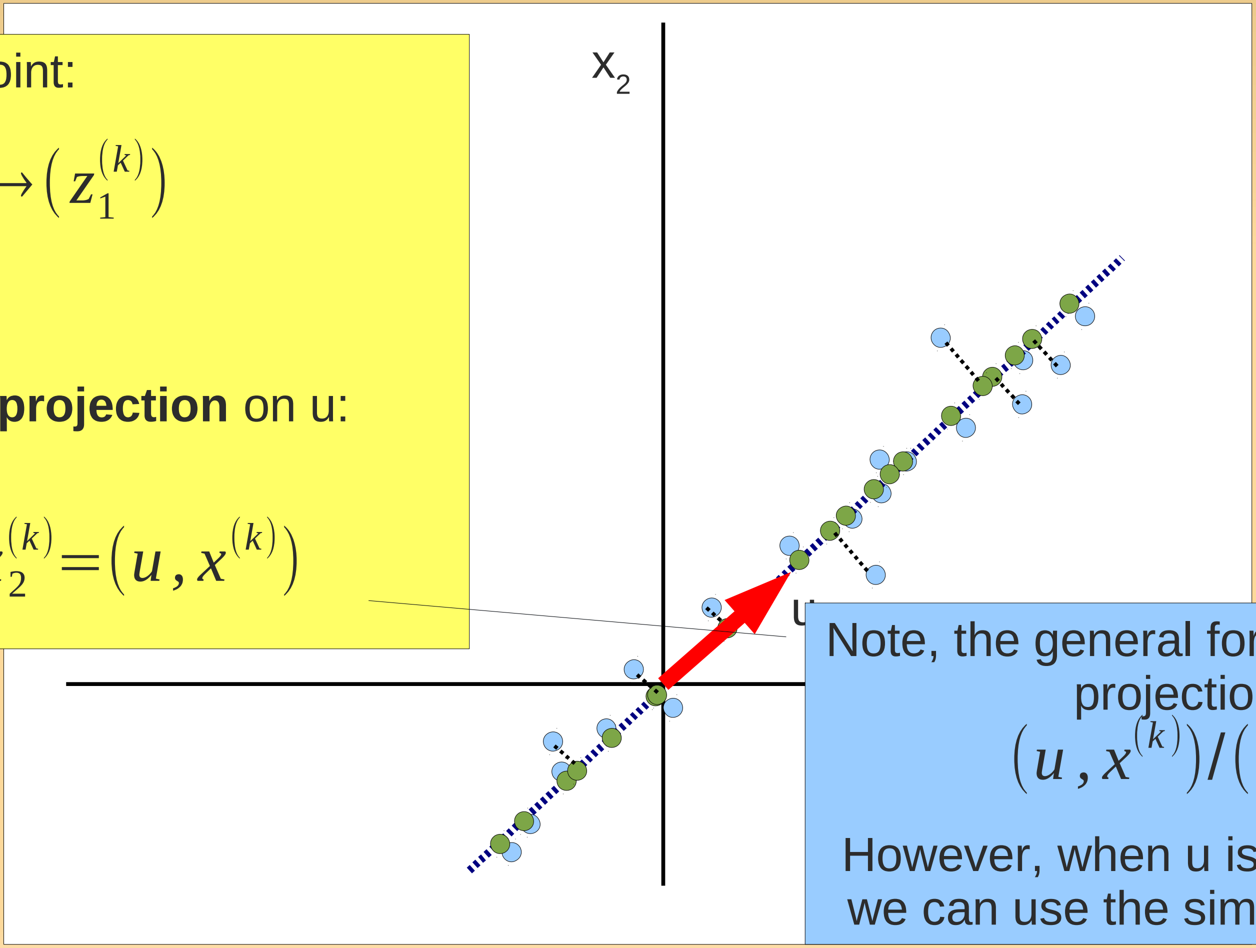
- How would you summarize this data using 1 dimension?

Transform of *k*-th point:

$$\left(x_1^{(k)}, x_2^{(k)}\right) \rightarrow \left(z_1^{(k)}\right)$$

where z_1 is the
orthogonal scalar projection on u:

$$z_1^{(k)} = u_1 x_1^{(k)} + u_2 x_2^{(k)} = \left(u, x^{(k)}\right)$$



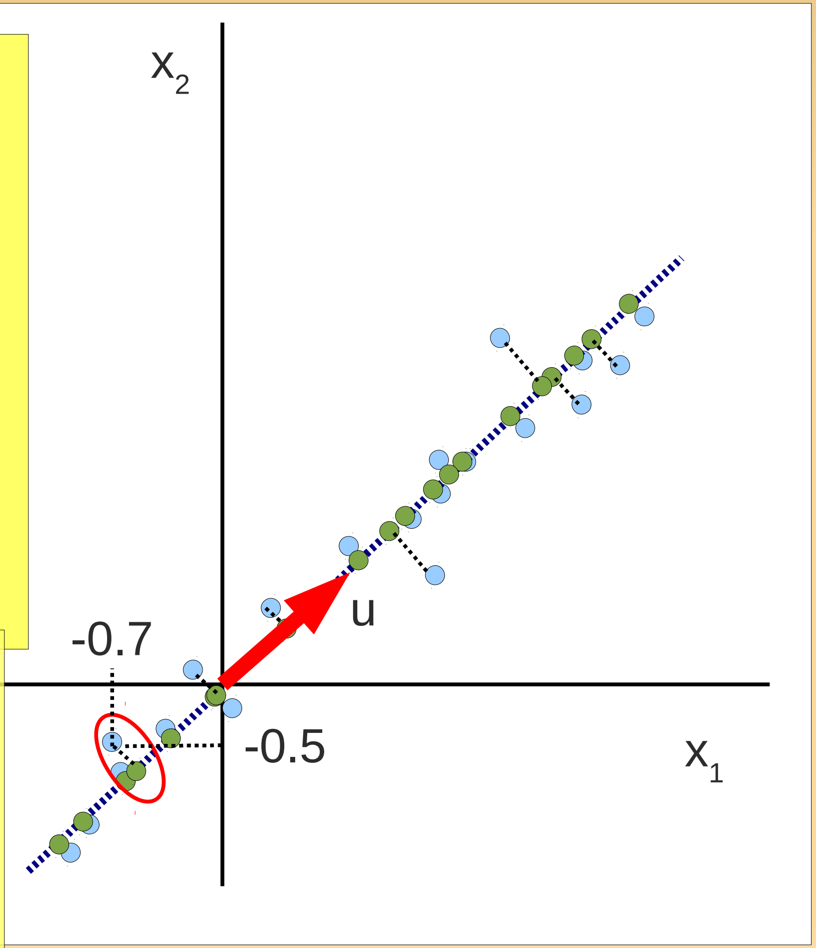
Note, the general formula for scalar projection is

$$\left(u, x^{(k)}\right) / \left(u, u\right)$$

However, when u is a unit vector, we can use the simplified formula

PCA – Intuition

- How would you summarize this data using 1 dimension?



Transform of *k*-th point:

$$\left(x_1^{(k)}, x_2^{(k)}\right) \rightarrow \left(z_1^{(k)}\right)$$

where z_1 is the **orthogonal scalar projection** on u:

$$z_1^{(k)} = u_1 x_1^{(k)} + u_2 x_2^{(k)} = \left(u, x^{(k)}\right)$$

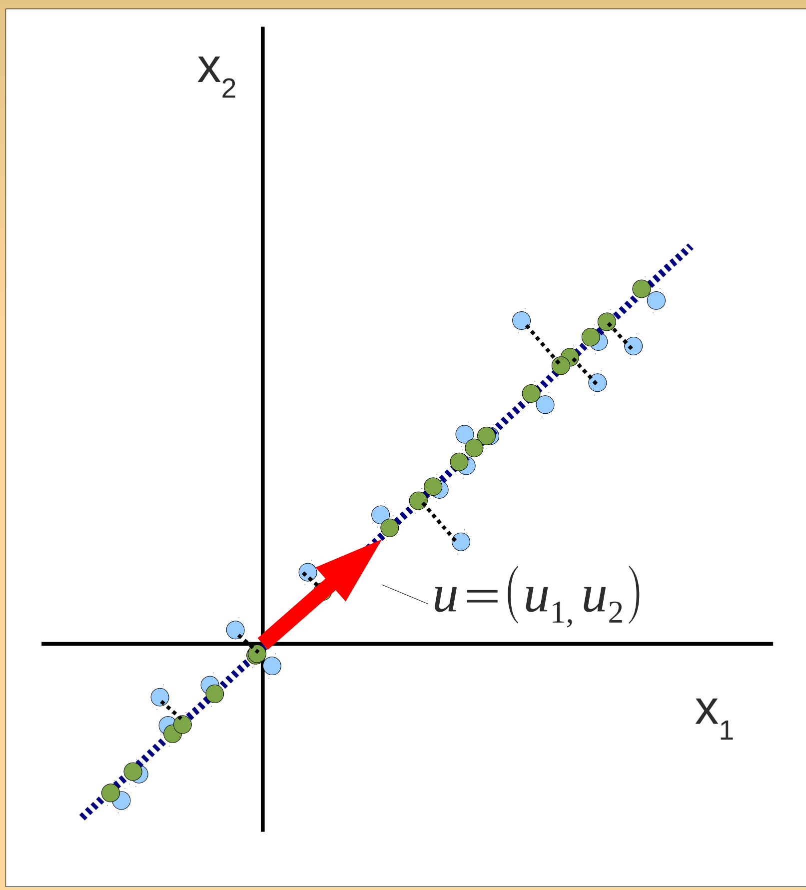
E.g.:

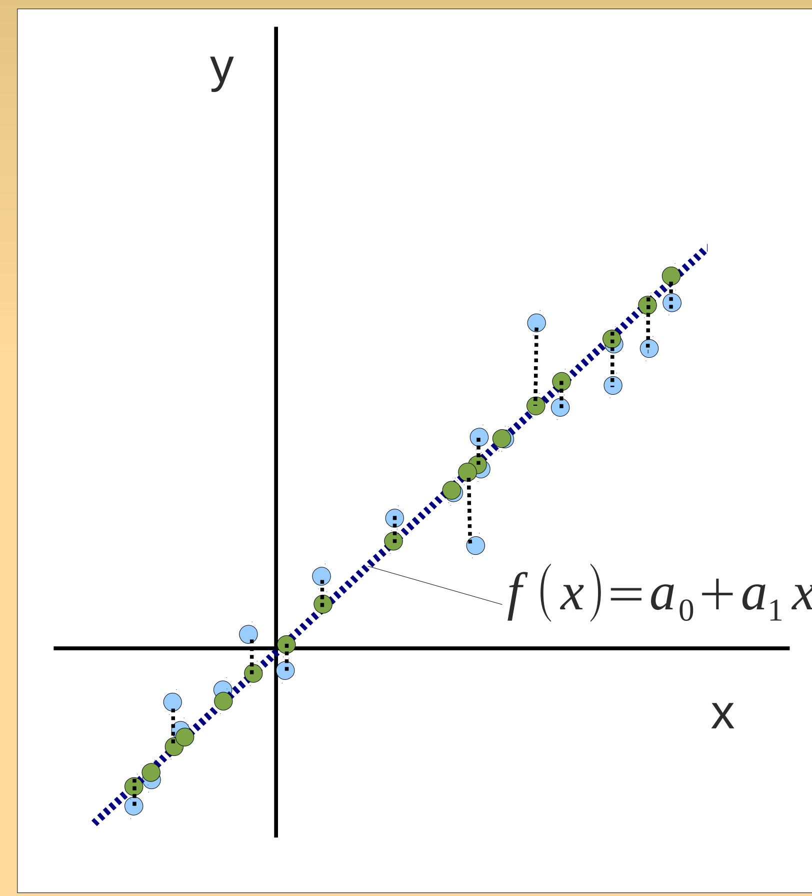
$$z_1 = 0.7(-0.7) + 0.7(-.5) = -0.84$$

is the first principal component of this data point

PCA vs. Least Squares

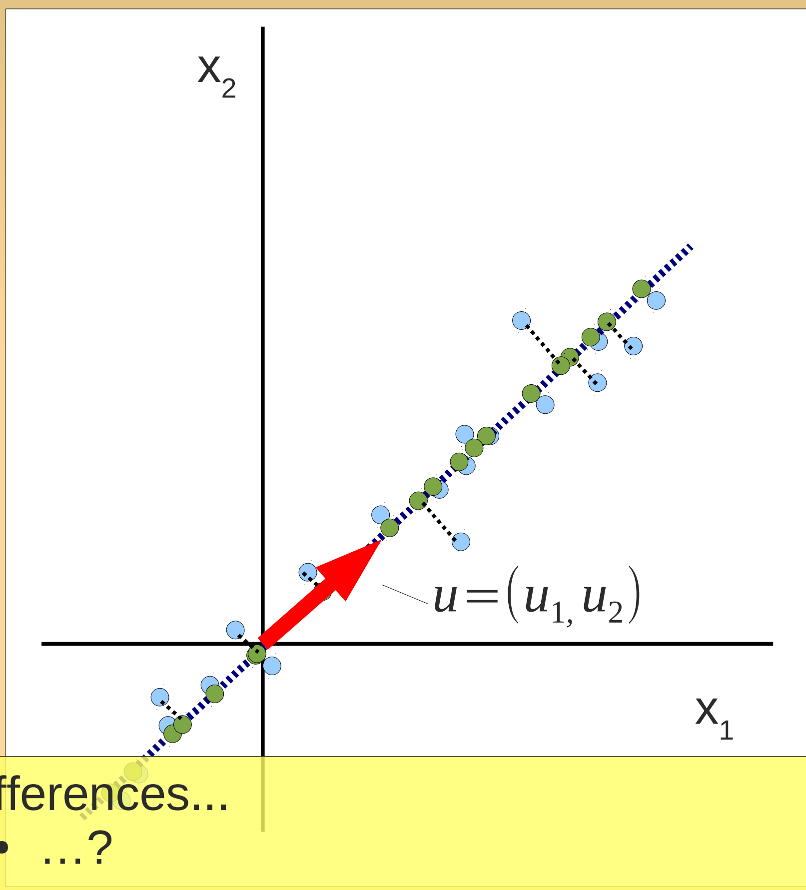
- PCA and Least Squares Regression appear similar...

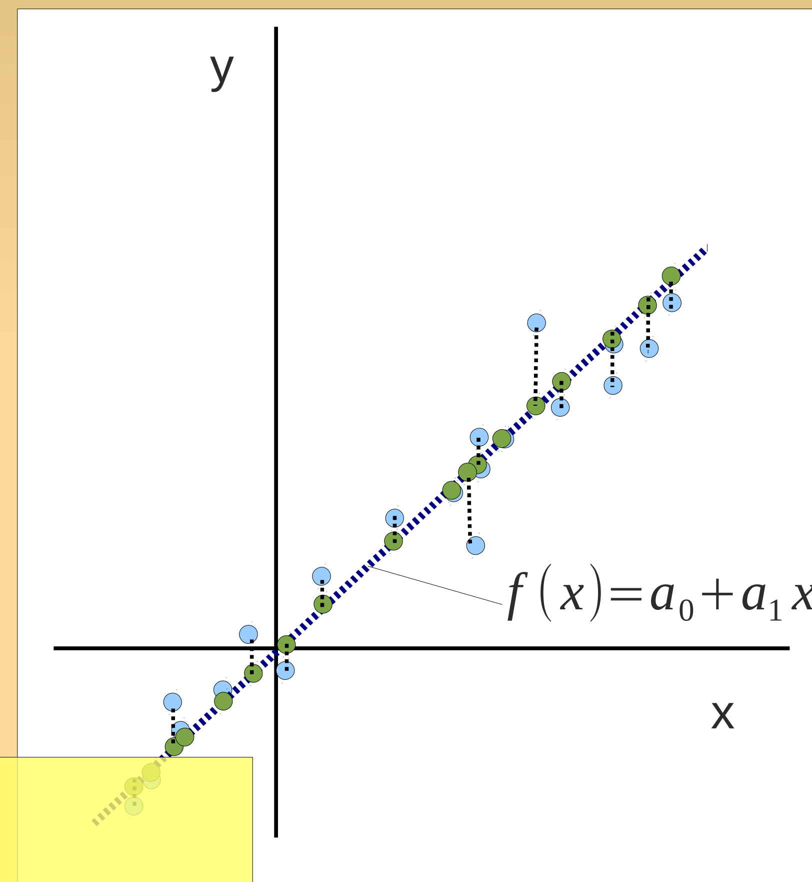




PCA vs. Least Squares

- PCA and Least Squares Regression appear similar...



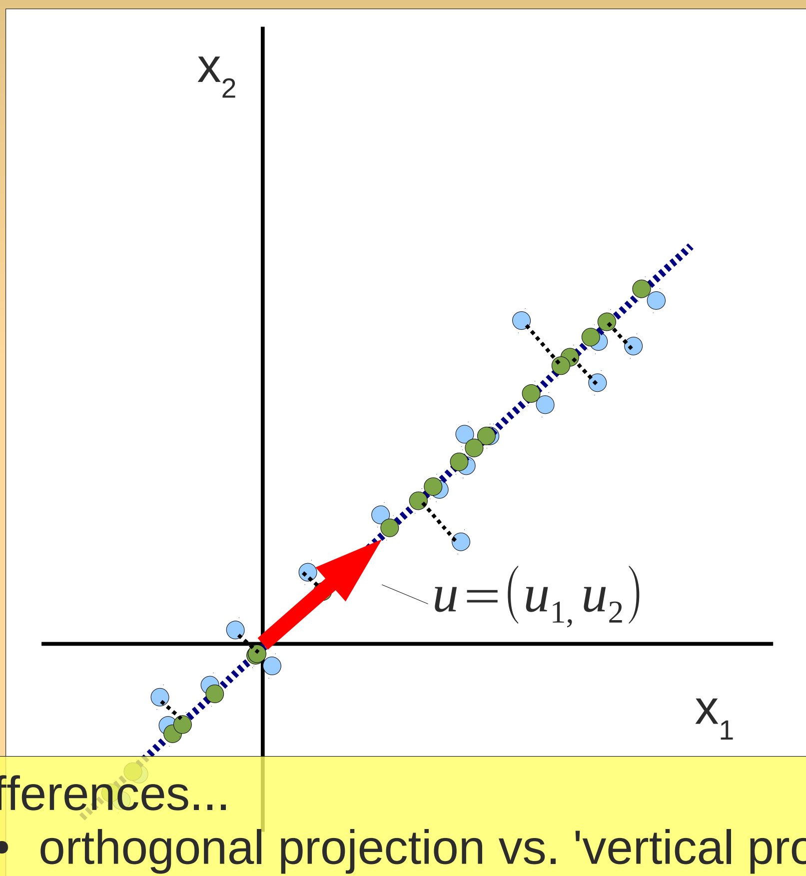


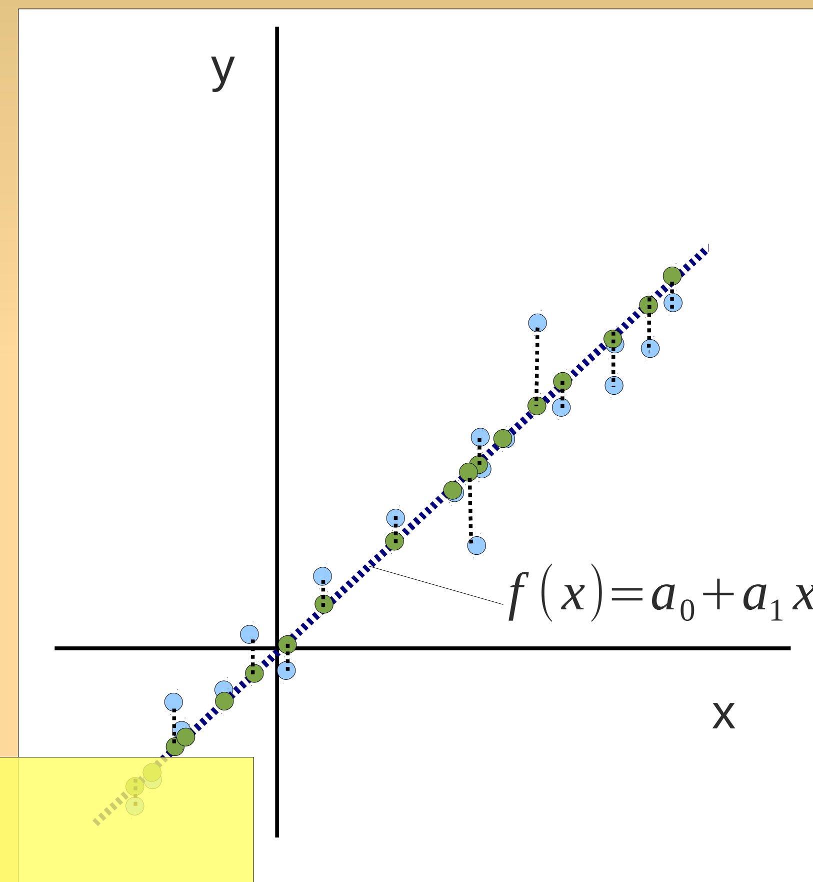
Differences...

- ...?

PCA vs. Least Squares

- PCA and Least Squares Regression appear similar...



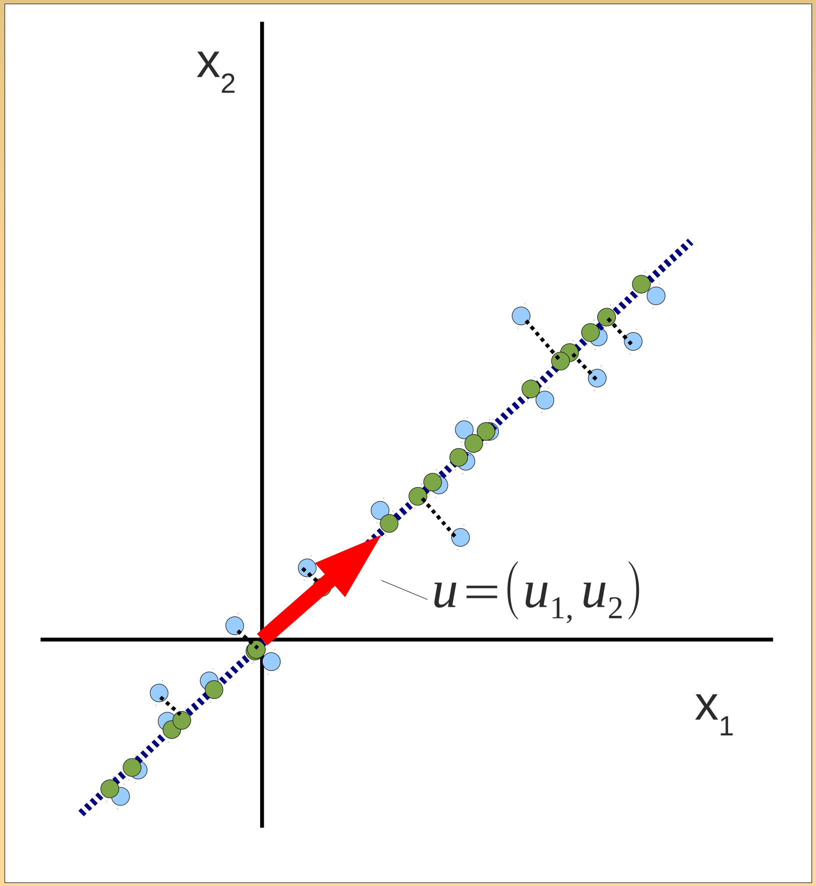


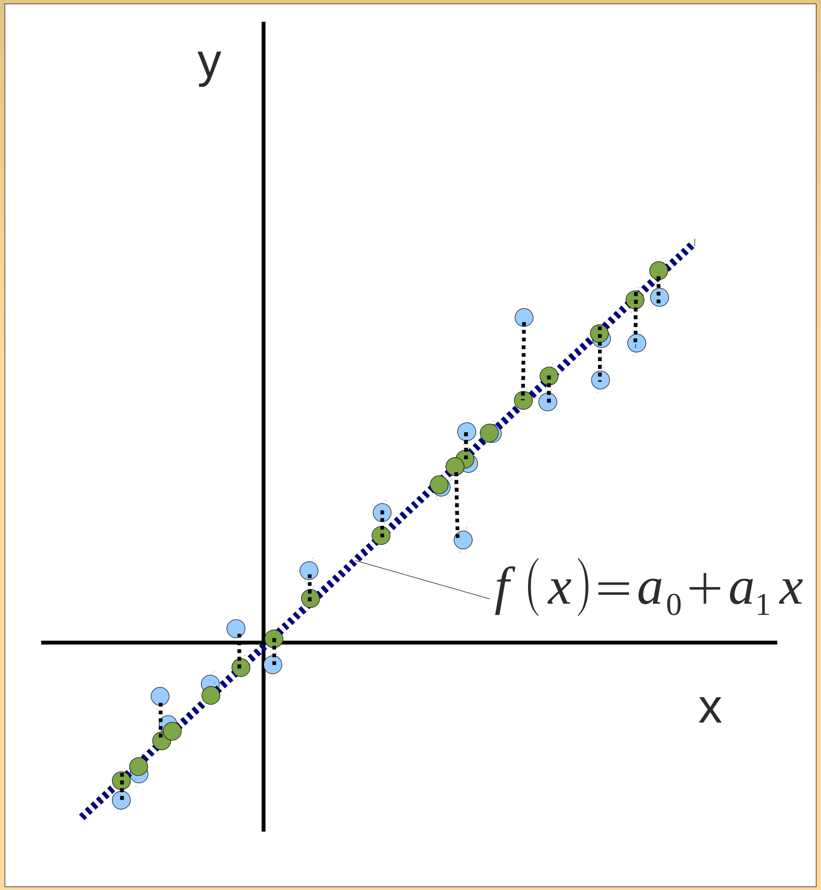
Differences...

- orthogonal projection vs. 'vertical projection'
- special status of *y* variable
- *u* is a direction, while *f* is a function
- computation is completely different

PCA vs. Least Squares

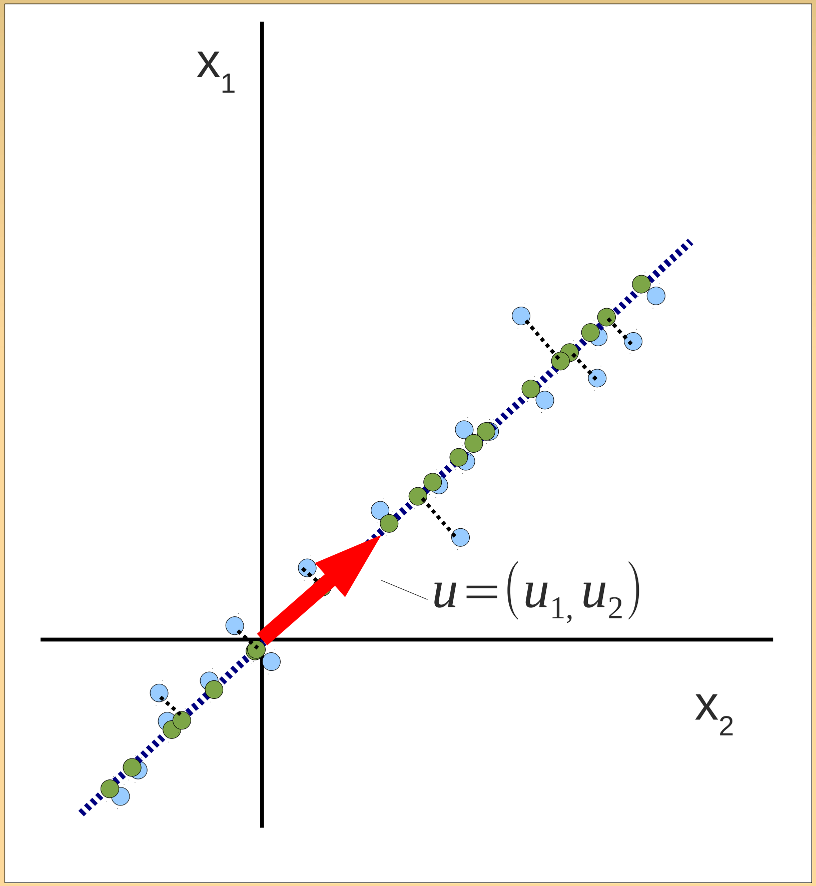
- What would happen when switching the axes...?

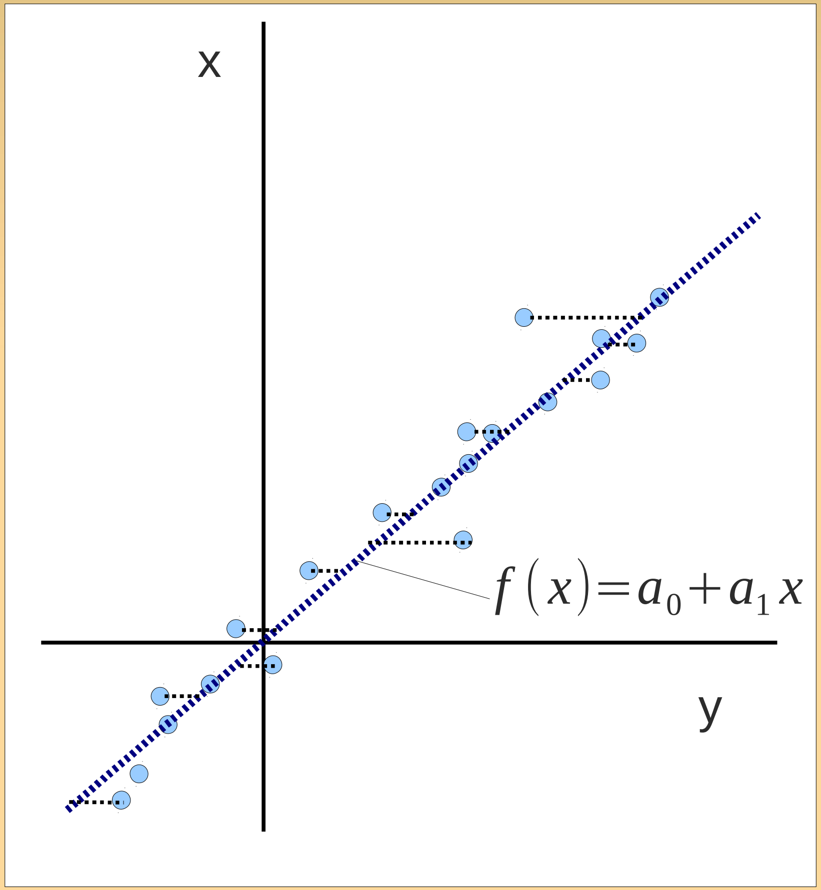




PCA vs. Least Squares

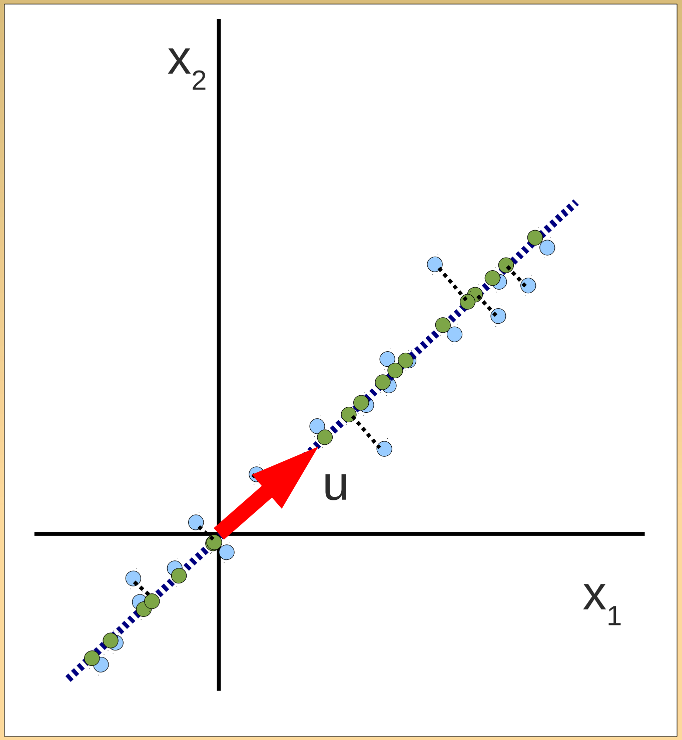
- What would happen when switching the axes...?





PCA – Intuition

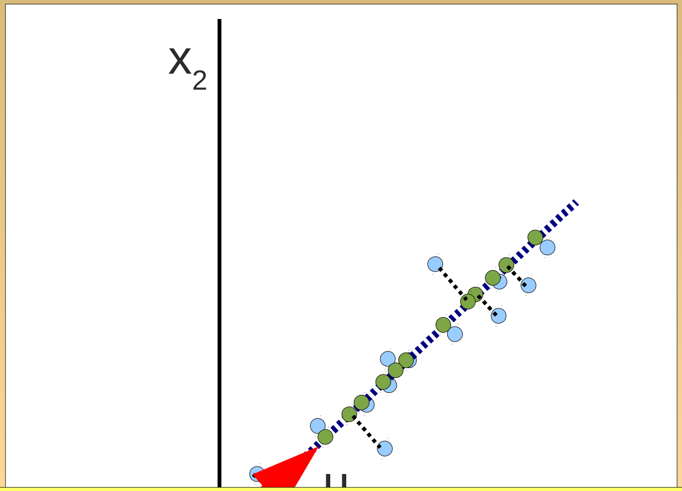
- PCA so far...
  - find the direction *u* of highest variance
  - project data on *u* → z_1 the **first** principle component (PC)



- Next...
  - find **more directions** of high variance
    → *u* is $u^{(1)}$, the direction of the first PC
    → find $u^{(2)}$, $u^{(3)}$,..., $u^{(D)}$
    (the directions of the other PCs)
  - **How** to find these directions!

PCA – Intuition

- PCA so far...
  - find the direction *u* of highest variance
  - project data on *u* → z_1 the **first** principle component (PC)



- Next...
  - find **more directions** of high varia
    → *u* is $u^{(1)}$, the direction of the first
    → find $u^{(2)}$, $u^{(3)}$,..., $u^{(D)}$
    (the directions of the other PCs
  - **How** to find these directions!

The name **Principle Components**

- variables z_i are linear combinations of data $x_1,\ldots,x_D$

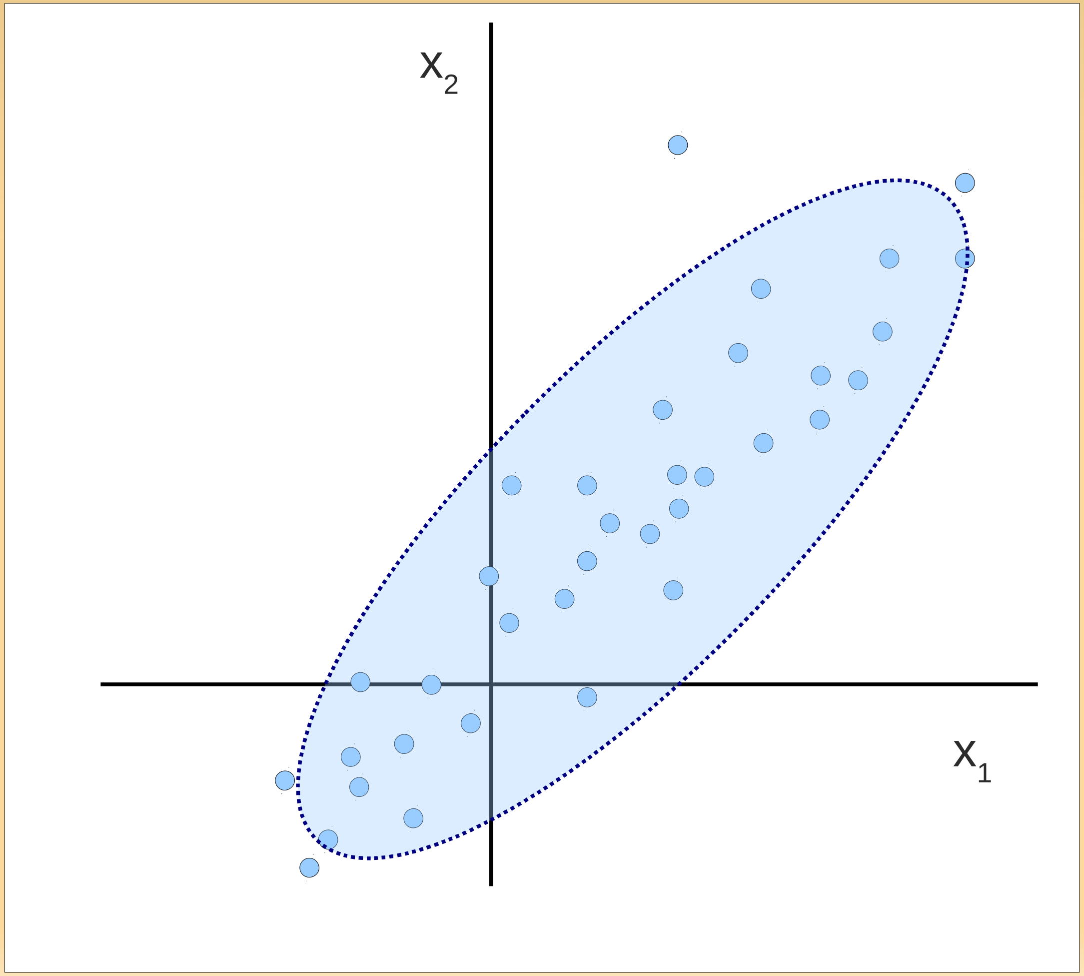
$$z_i^{(k)} = u_1^{(i)} x_1^{(k)} + \ldots + u_D^{(i)} x_D^{(k)}$$

- But (later): x_i are linear also combinations of PCs $z_1,\ldots,z_D$!

$$x_i^{(k)} = u_i^{(1)} z_1^{(k)} + \ldots + u_i^{(D)} z_D^{(k)}$$

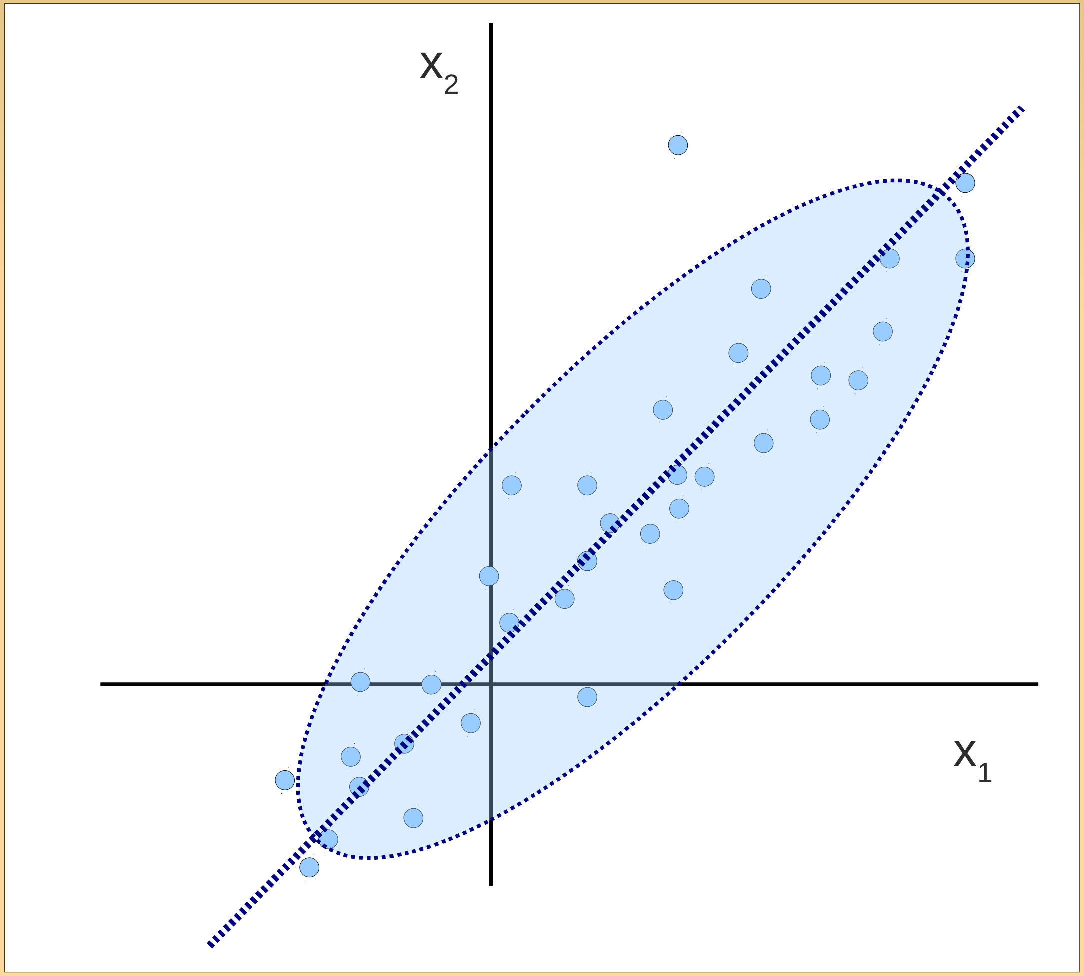
More Principle Components

- Given this data, what is $u^{(1)}$?
  (i.e., the direction of the first PC)



More Principle Components

- $u^{(1)}$ explains the most variance
- What is $u^{(2)}$? (the direction of the 2nd PC) ?

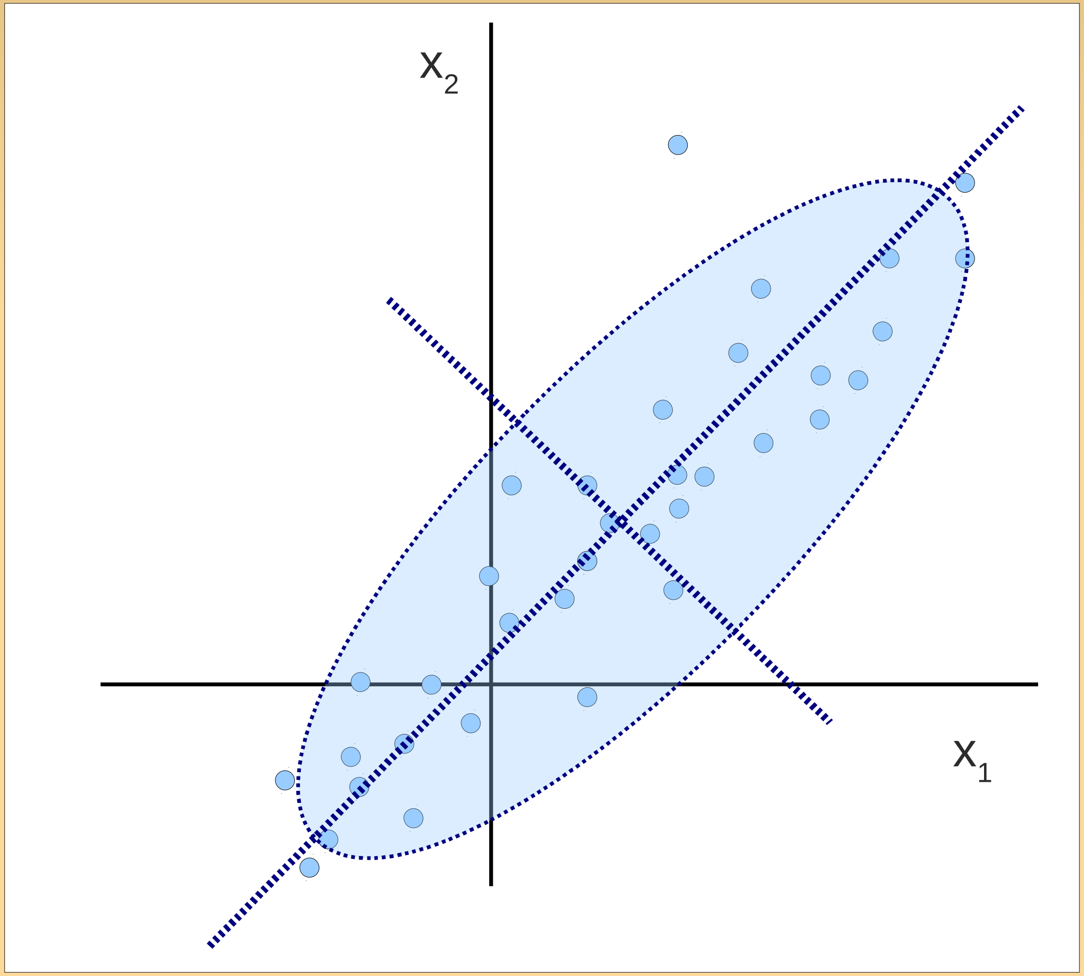


More Principle Components

- $u^{(2)}$ is the direction with most 'remaining' variance
  - orthogonal to $u^{(1)}$!
- Data is 2D, so can find only two directions
- Each point $x^{(k)}$ can be converted to $z^{(k)}$

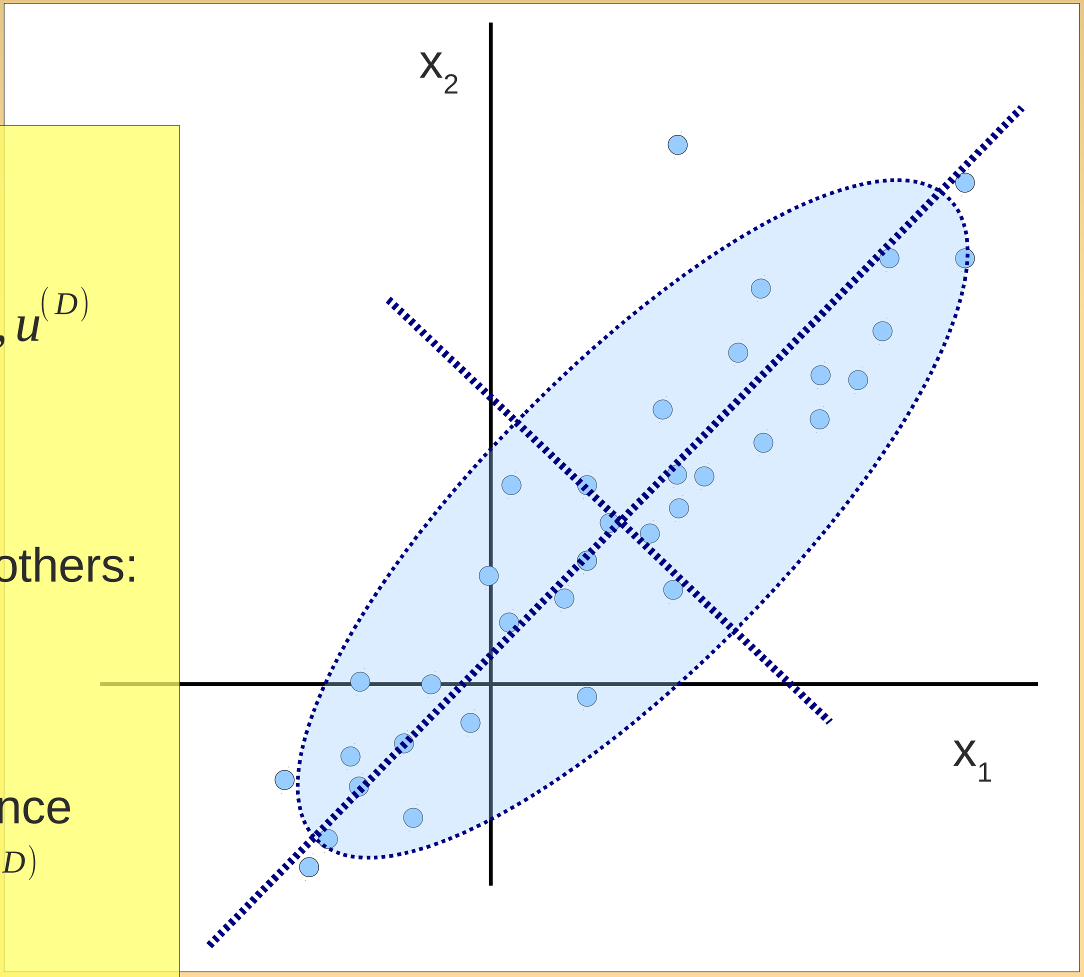
$$\left(x_1^{(k)}, x_2^{(k)}\right) \Leftrightarrow \left(z_1^{(k)}, z_2^{(k)}\right)$$

$$z_i^{(k)} = \left(u^{(i)}, x^{(k)}\right)$$



More Principle Components

- $u^{(2)}$ is the direction with most 'remaining' variance
  - orthogonal to $u^{(1)}$!



In general

- If the data is D-dimensional
- We can find D directions $u^{(1)},\ldots,u^{(D)}$
- Each direction itself is a D-vector:

$$u^{(i)}=(u_{1,}^{(i)}\ldots,u_{D}^{(i)})$$

- Each direction is orthogonal to the others:

$$(u^{(i)},u^{(j)})=0$$

- The first direction is has most variance
- The least variance is in direction $u^{(D)}$