PCA

chapter 12 of bishop book.

Given an input of D dimensions, our objective is to find a subspace that maximizes the variance of the data, as we need compress the data but as make it as lossless as we can.

We delete the directions/dimensions with less variance, meaning that we lose the dimensions that carry less information.

Motivation

- Compress the dimensions, but keep most of information.
- Preserve tha variance of the data as much as possible.

PCA example

Recall that with PCA, we can kill the dimensions that we don't really need to explain the data well enough.

Now, imagine a TV.

A tv basically retains all the information of the 2 dimensions, but doesn't have the third one(depth).

And in fact, we don't really need the third dimension to understand what we are looking at, 2D is just enough!

Output of PCA

The PCA outputs the transformation matrix that we need to apply to the original data matrix to kill the dimensions.

That is often called the U matrix, because it is the U that results from SVD or Spectral decomposition.

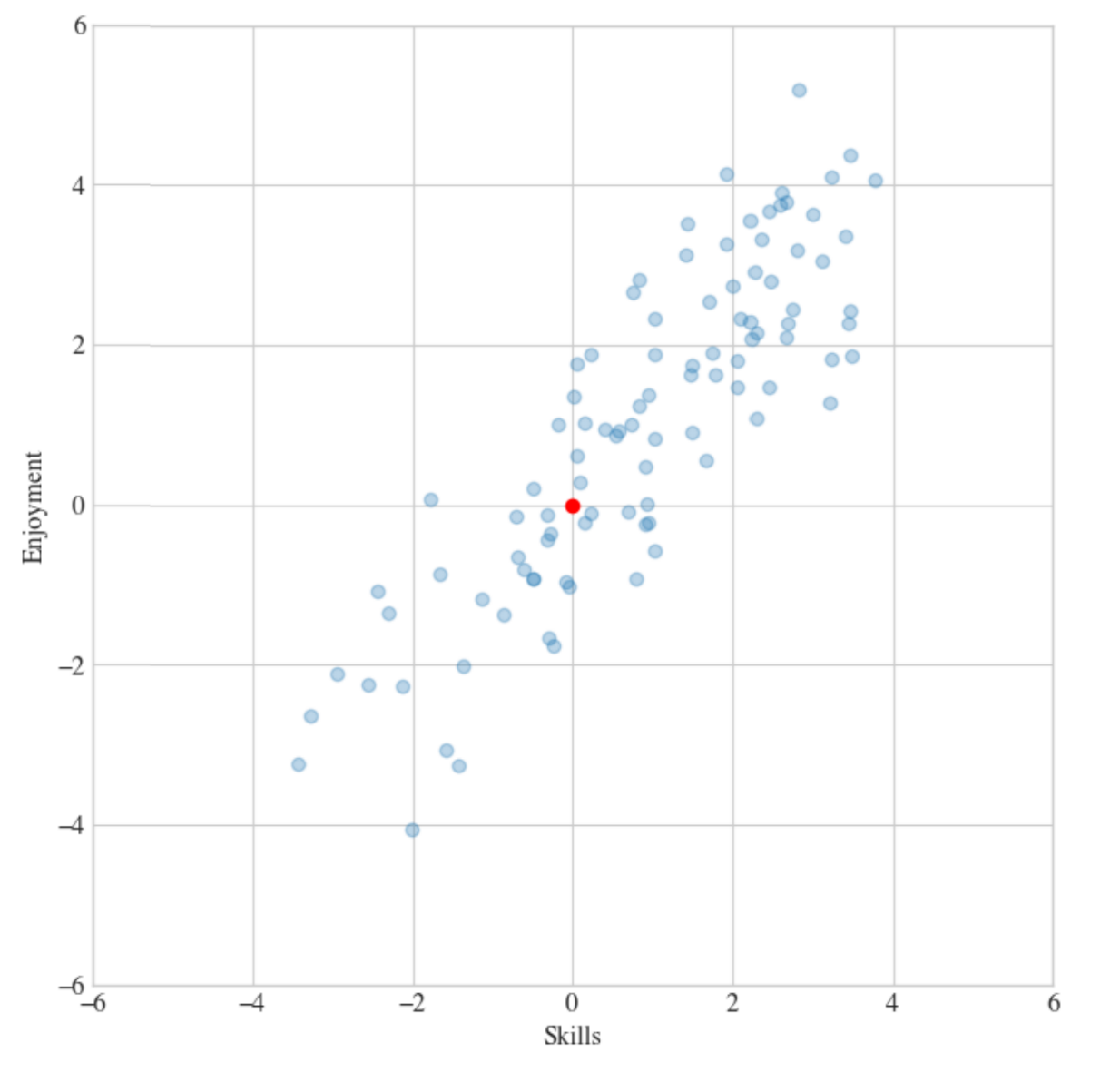
If we project the data using this matrix, the space will actually shrink and lose variance, while still retaining most of the information.

Example

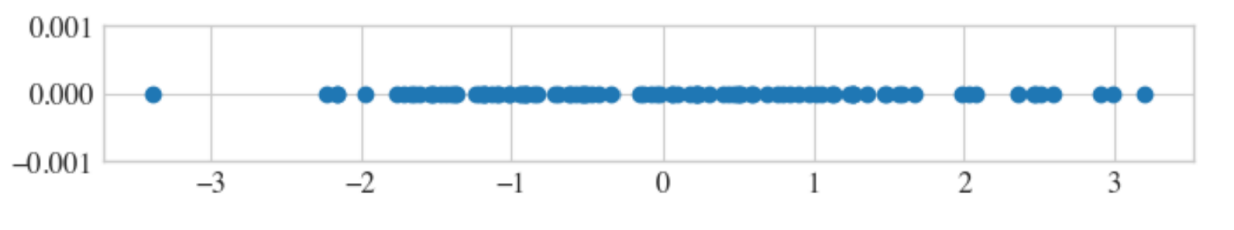
Starting from a 2D point cloud, we can reduce the dimensions by 1 if the energy remains > 95%.

In this specific case, we are killing a single dimension.

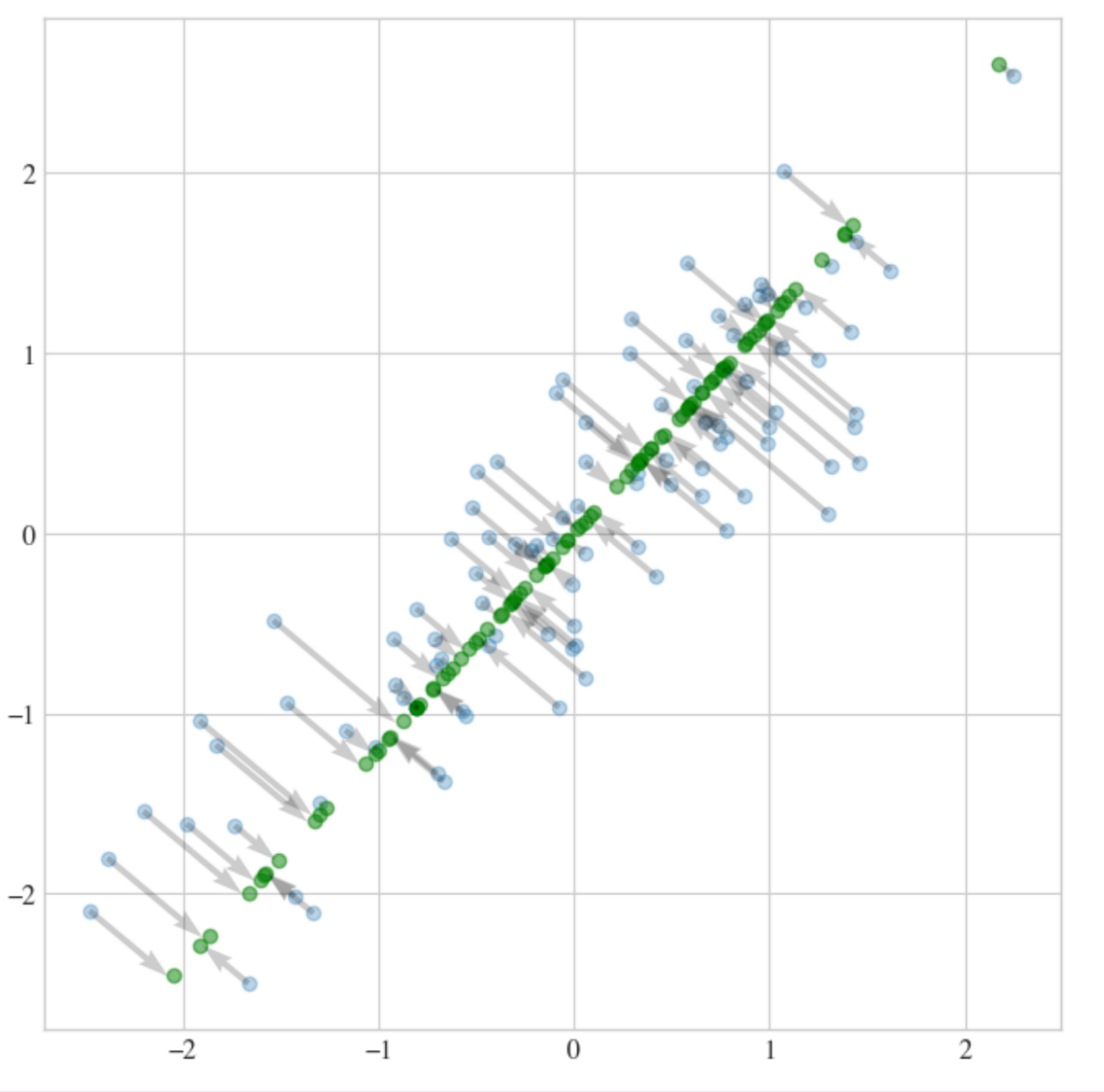
Before applying U:



After applying U:



After reconstruction(this is not part of the process, it's done by projecting with U^T):



All that variance has been lost.

How to choose dimensionality of subspace?

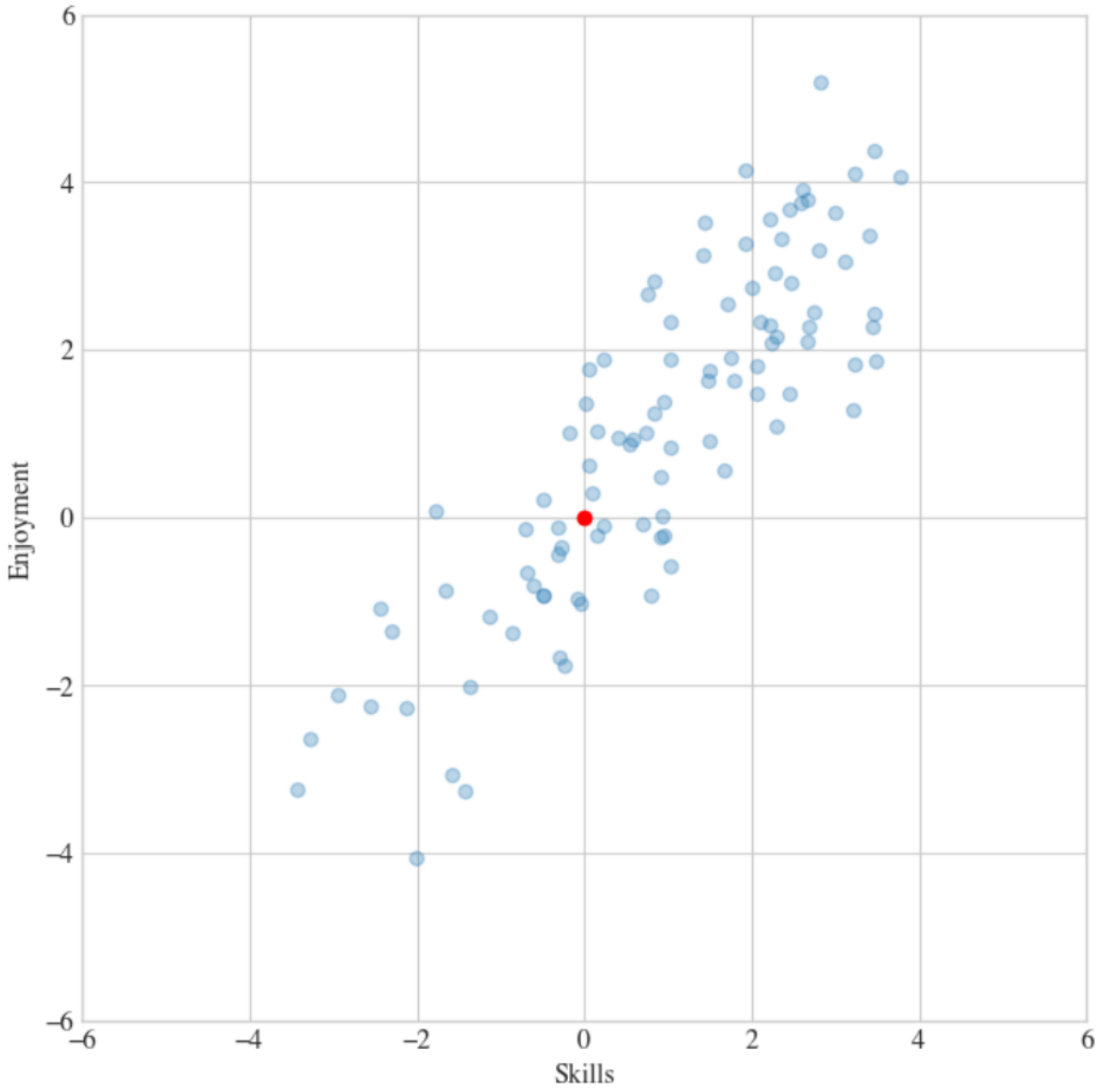
We can actually measure how much relative variance the dimensions have compared to the total variance.

- We either sort the dimensions and keep only a certain number of them, a piacere.
- Or we sort the dimensions and kill only the ones that can be killable without too much loss.

In the end, we are just given all the dimensions and how much variance they carry, we choose how to act.

How to:

Suppose we have a point cloud X:



1. Standardize the data

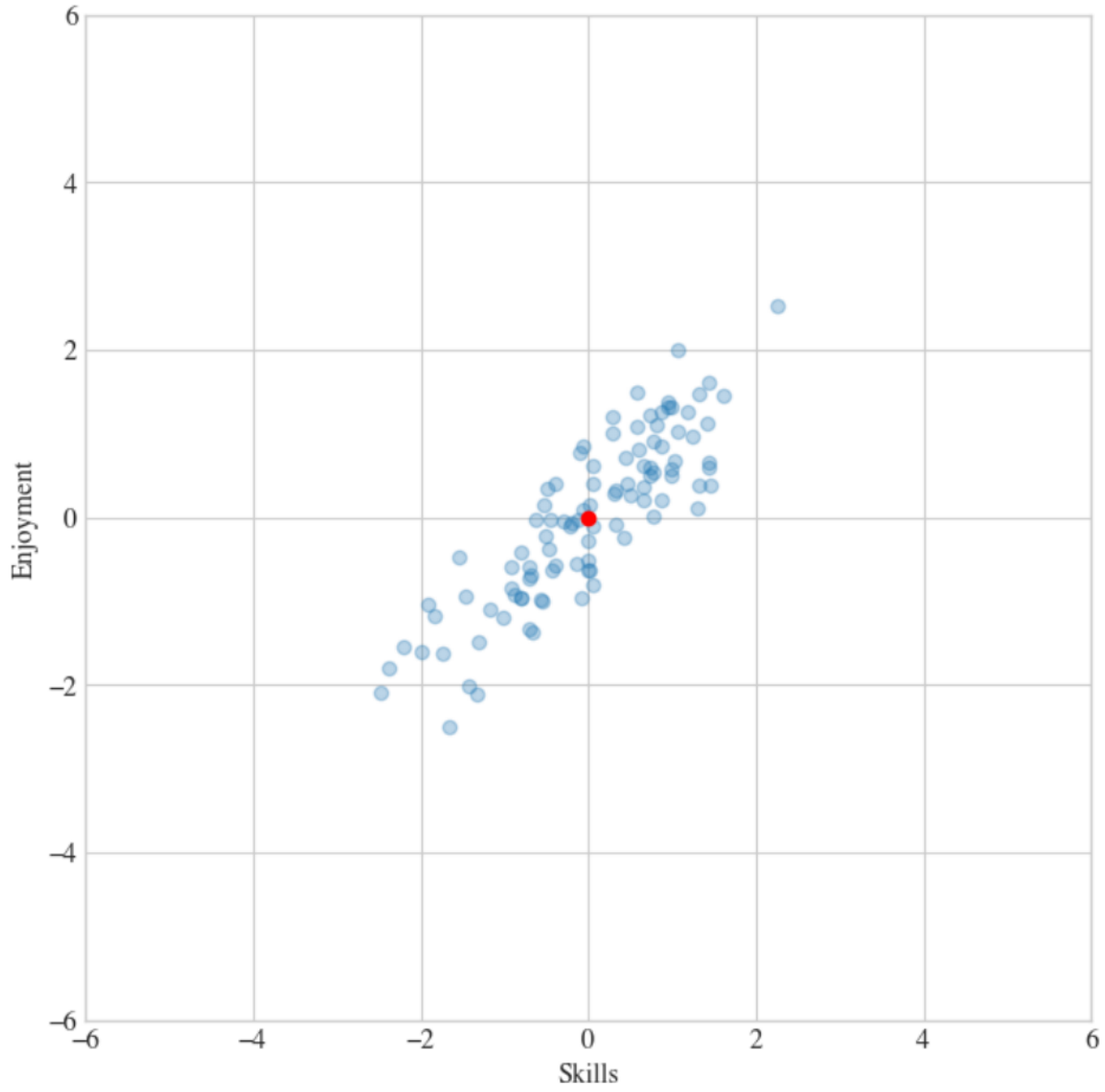
First step, we standardize the data :

$$X' = \frac{X - \mu}{\sigma}$$

We just shifted the mean to 0 and rescaled all axis, so that the standard deviation is 1.

```
center = X.mean(axis=0) #X shape is 100x2
std = X.std(axis=0)
Xp = (X-center)/std
```

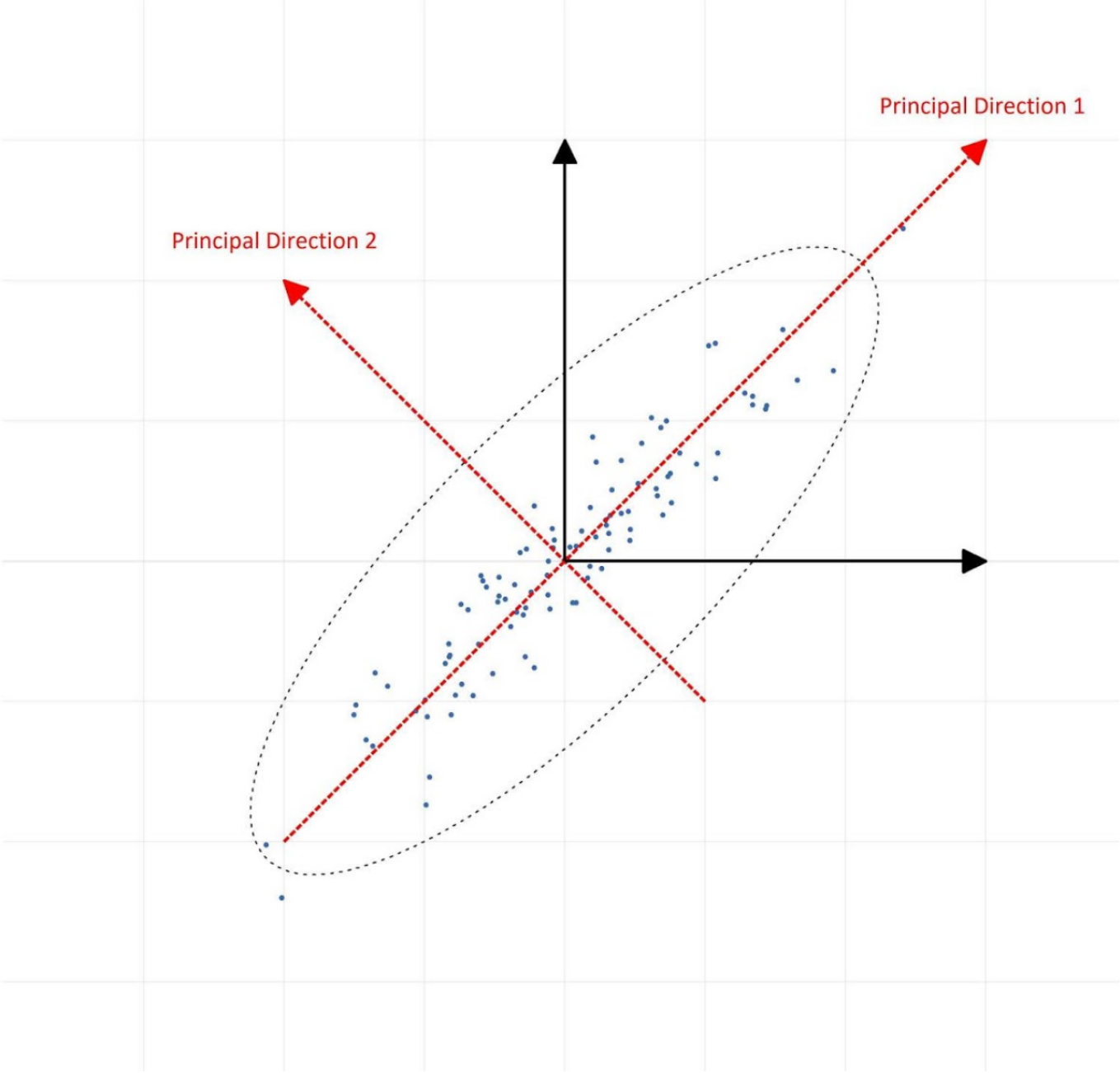
Output:



2. Use SVD or Spectral decomposition to find U

U is basically the matrix containing the eigenvectors of the data matrix. Meaning that it contains the directions of the data.

Those eigenvectors are called the principal components of the data.
The first eigenvectors is the vector that fits best the point cloud, the rest are vectors orthogonal to the first.
Think of it as we were fitting an ellipse through the point cloud.



Mathematically, U's columns are the eigenvectors of the covariance matrix.

Remember, we are working in the standardized version of the data X', so in this case the covariance matrix would be $\frac{1}{n}X'X'^T$.

In conclusion, to get the principal components of the point cloud, we need to get the eigenvectors of the covariance matrix $\frac{1}{n}X'X'^T$.

Spectral decomposition vs SVD

We have a problem here, it's called the large dimensionality curse.

Basically if the dimensionality of the data is too damn big, than the covariance matrix would be very expensive to compute. Thus we need to find a way to avoid this.

Spectral decomposition:

If we use the spectral decomposition to compute the PCA, we would be forced to compute the covariance matrix.

SVD:

Recall that in SVD there are two matrices, U and V.

If X is NxD, then U will be DxD and V will be NxN.

The thing is, if we transpose X, we can still understand the data, so if DxD is bigger than NxN, we can literally transpose X and get the smallest covariance matrix through SVD.

Another way to look at it, is that you are just computing V instead of U, if U is larger than V.

Professors take on this:

$$X = USV^T$$

$\frac{1}{N}X'X'^T = \frac{1}{N}(USV^T)(USV^T)^T$

$\frac{1}{N}X'X'^T = \frac{1}{N}^T USV^T VS^T U^T = \frac{1}{N}US^2U^T$

Here he eliminated one of the two matrices from the formula. We just eliminate the bigger one.

Let's say we wanna go through with the Spectral decomposition.
We compute the spectral decomposition of the covariance matrix:

$$\frac{1}{N}X^TX = U\Sigma U^T$$

Where:

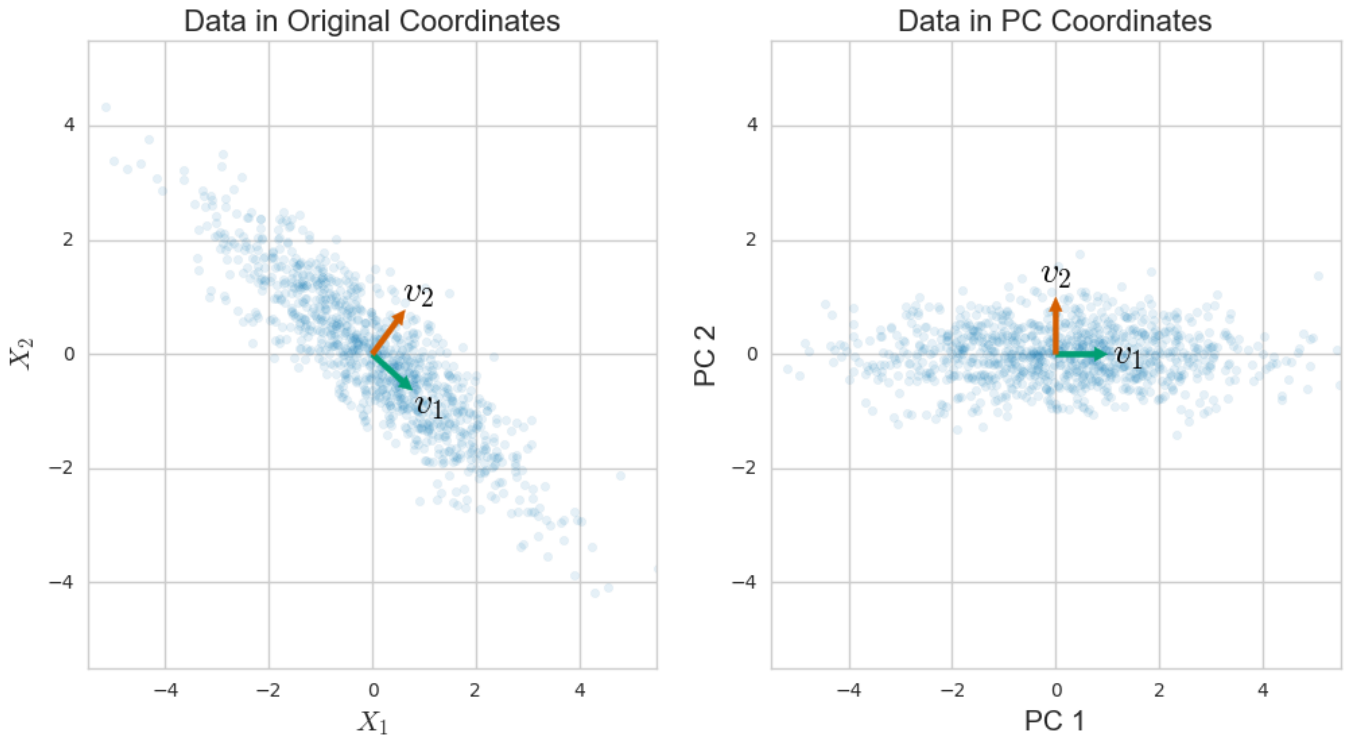
- X is the original data
- U is the matrix that contains the eigenvectors of the matrix we are decomposing, in this case $\frac{1}{N}XX^T$.
- Σ is the matrix containing the eigenvalues of the eigenvectors. Basically they tell how extended each of the principal components are. It should look like this:

$$\Sigma = Q\begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{bmatrix} Q^{-1},$$

Data decorrelation

By decorrelating the data we mean that we are putting the covariances to zero across all the axes.

We can do this by projecting X with U^T, which is basically just rotating the principal components to align with the basis vectors.



As you can see, there is no correlation anymore between the axes. No direction of growth, no nothing.

Seealso

http://ufldl.stanford.edu/tutorial/unsupervised/PCAWhitening/
https://stats.stackexchange.com/questions/194278/meaning-of-reconstruction-error-in-pca-and-lda