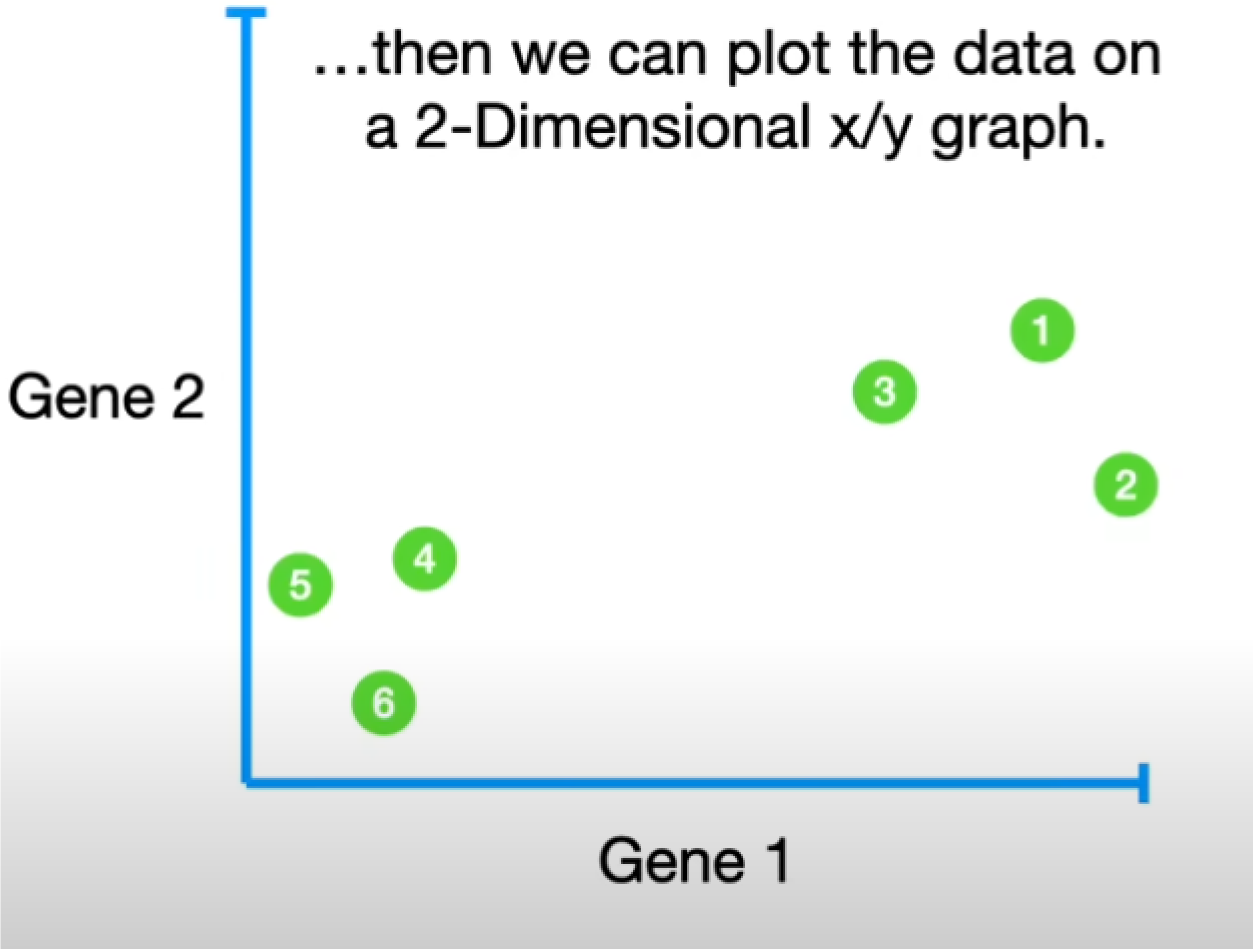
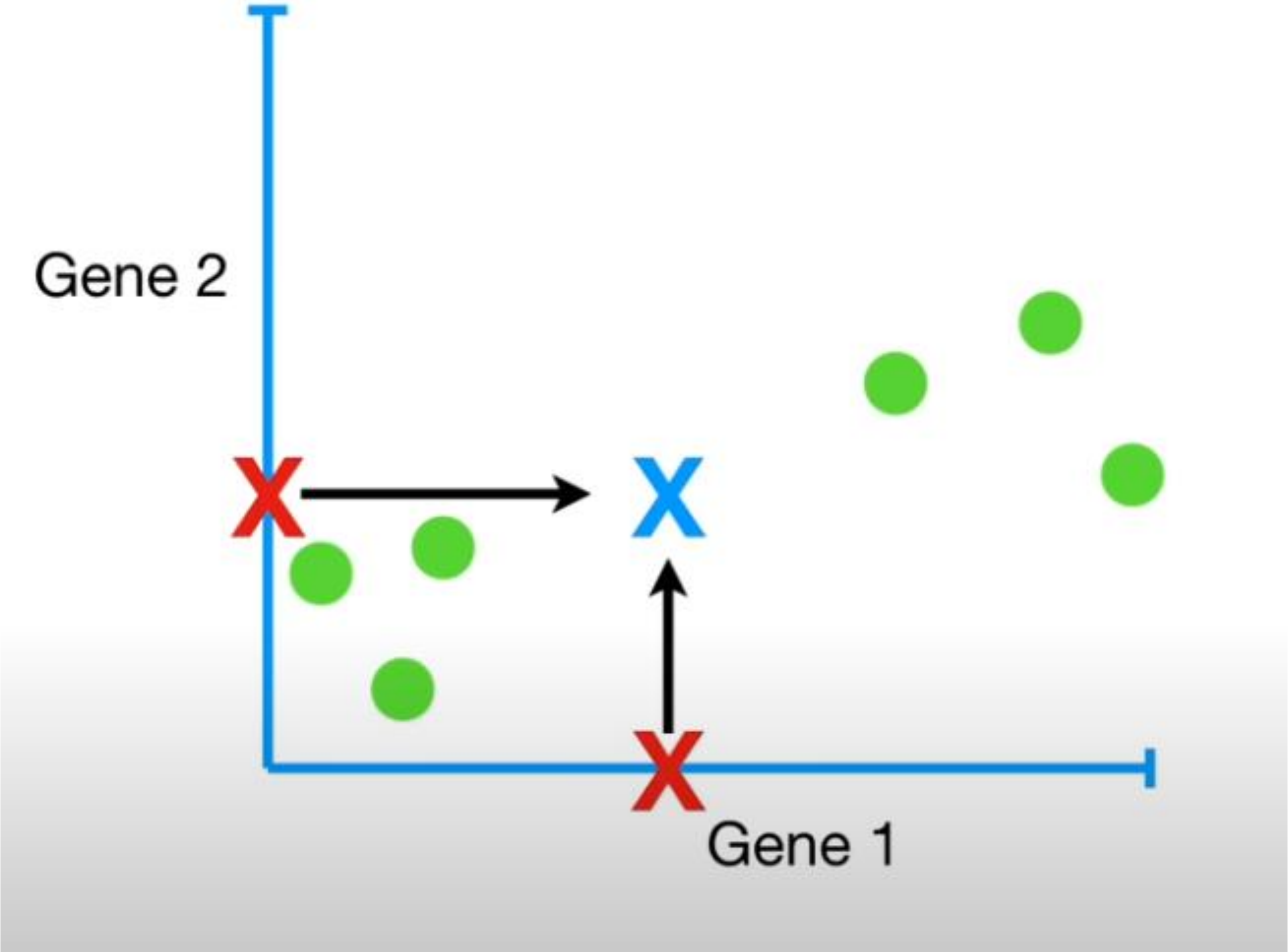
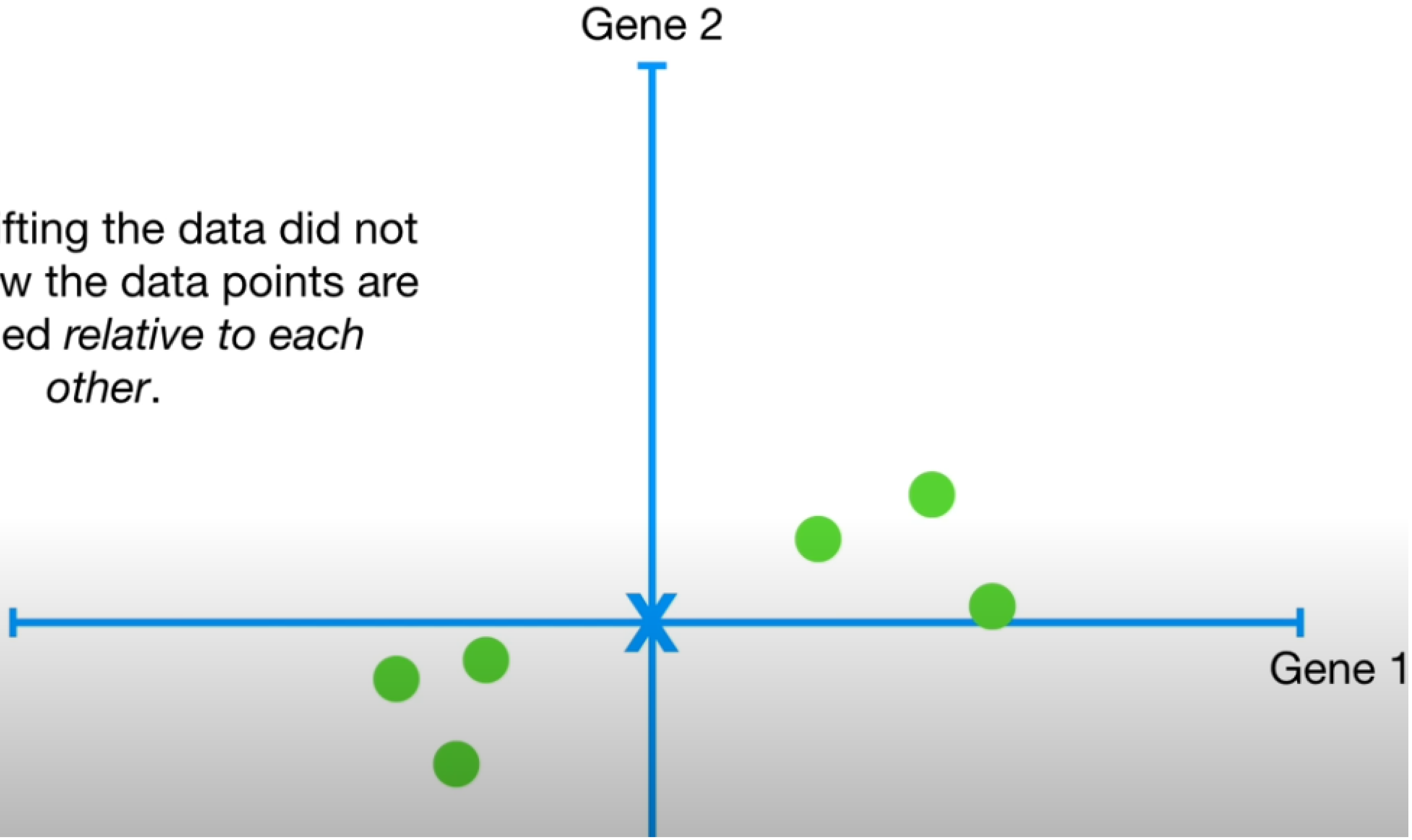
PCA

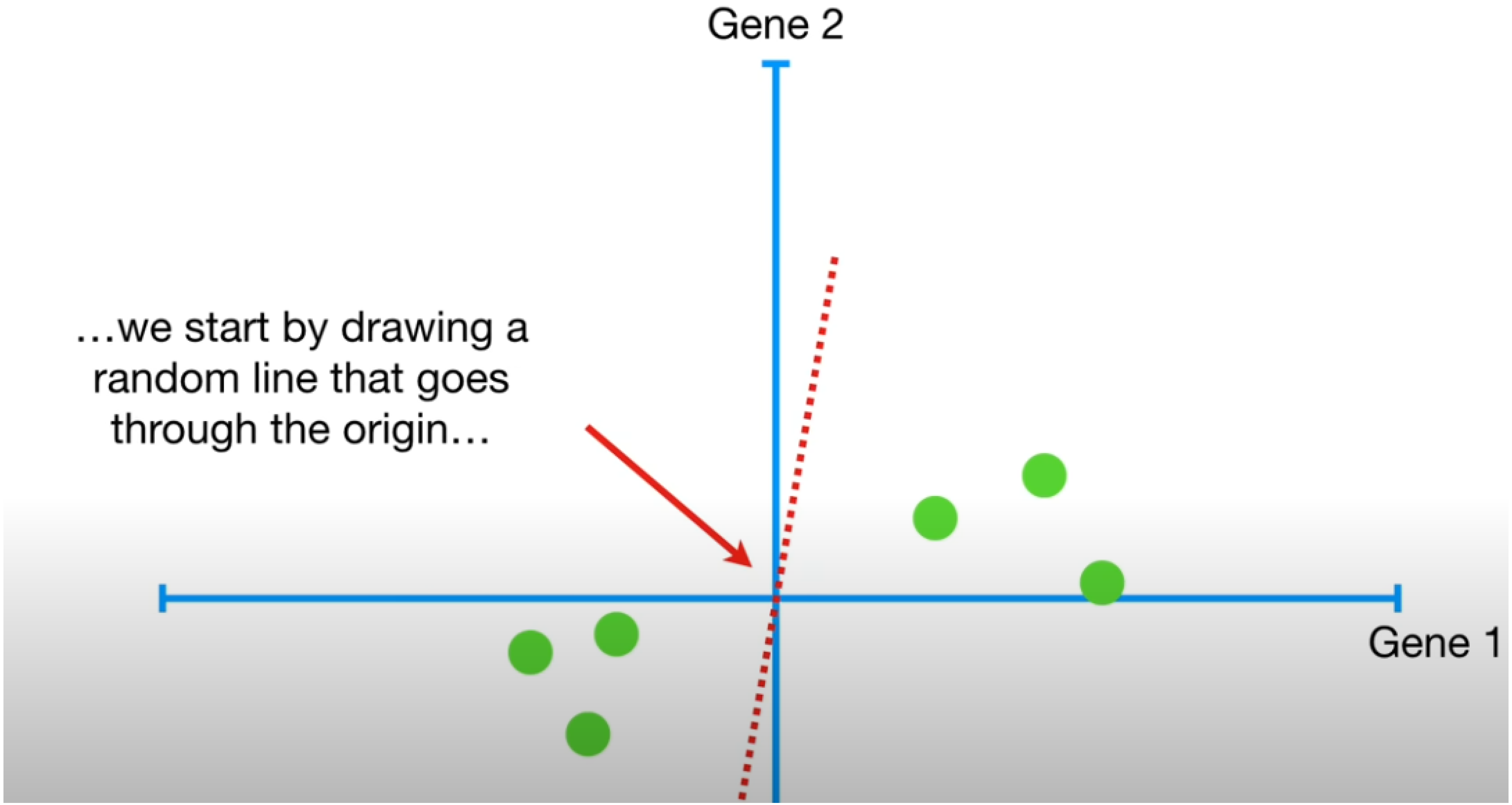


…then we can plot the data on a 2-Dimensional x/y graph.

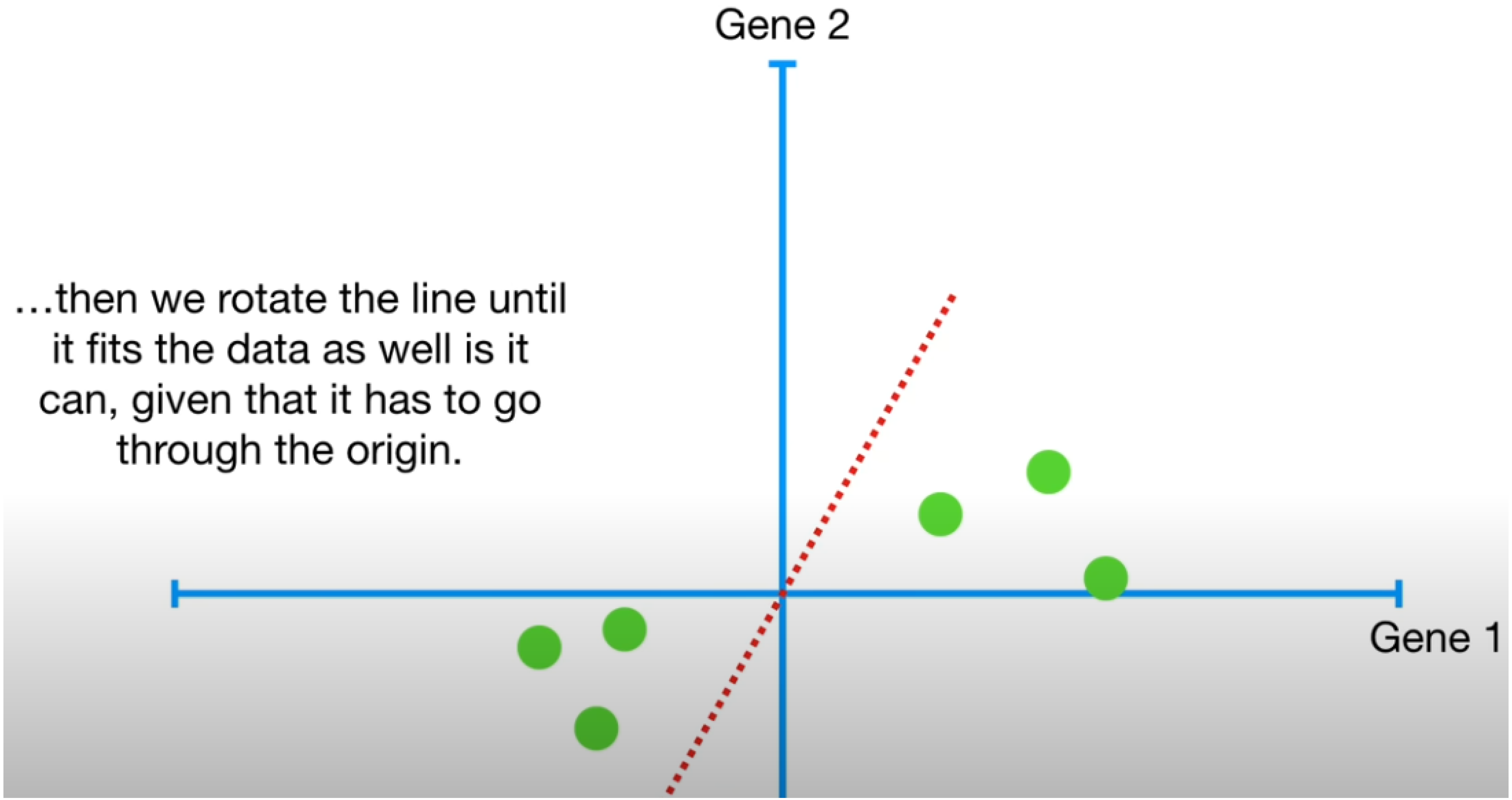

Gene 2
X
X
X
Gene 1

NOTE: Shifting the data did not change how the data points are positioned *relative to each other*.

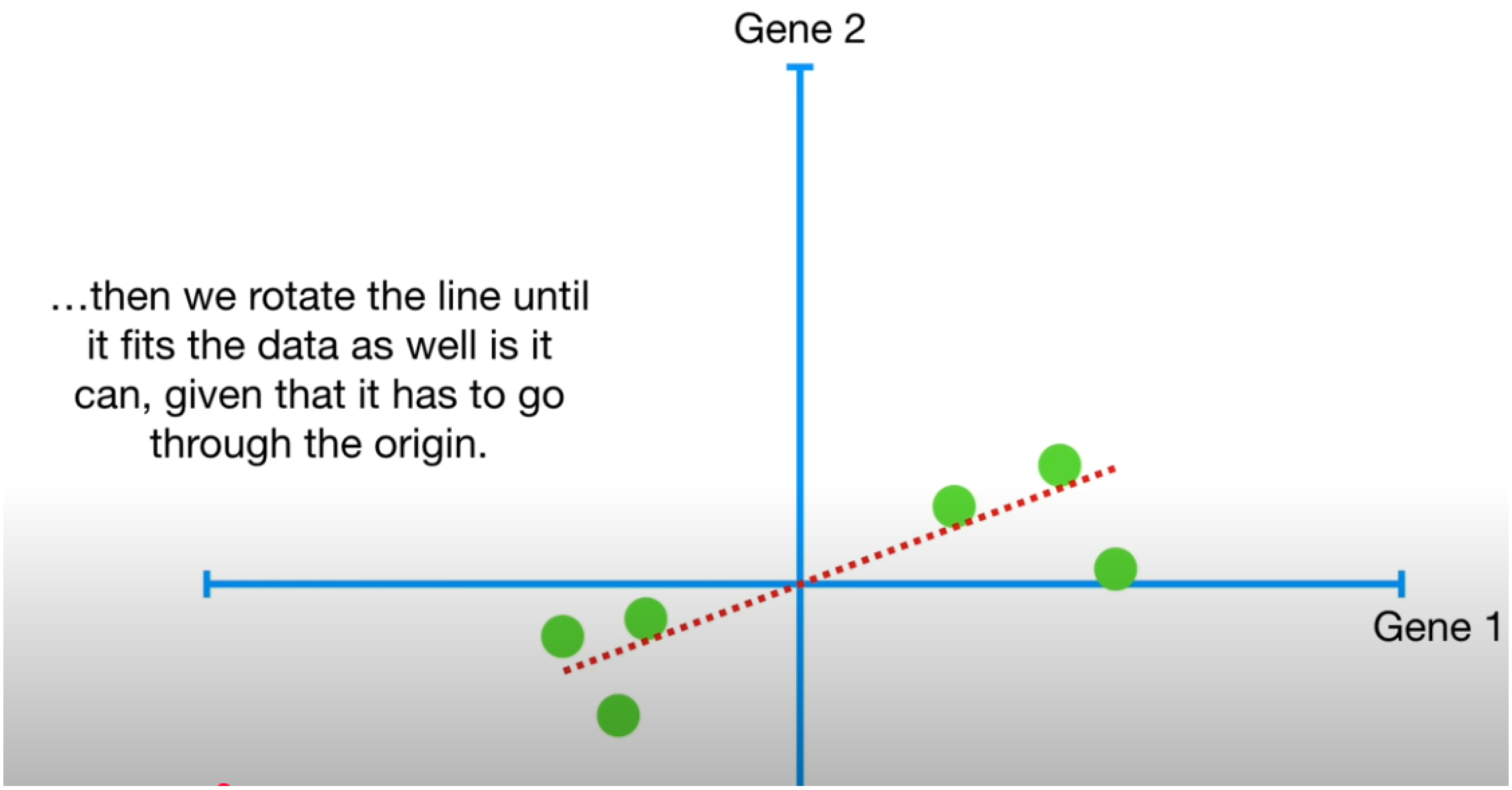




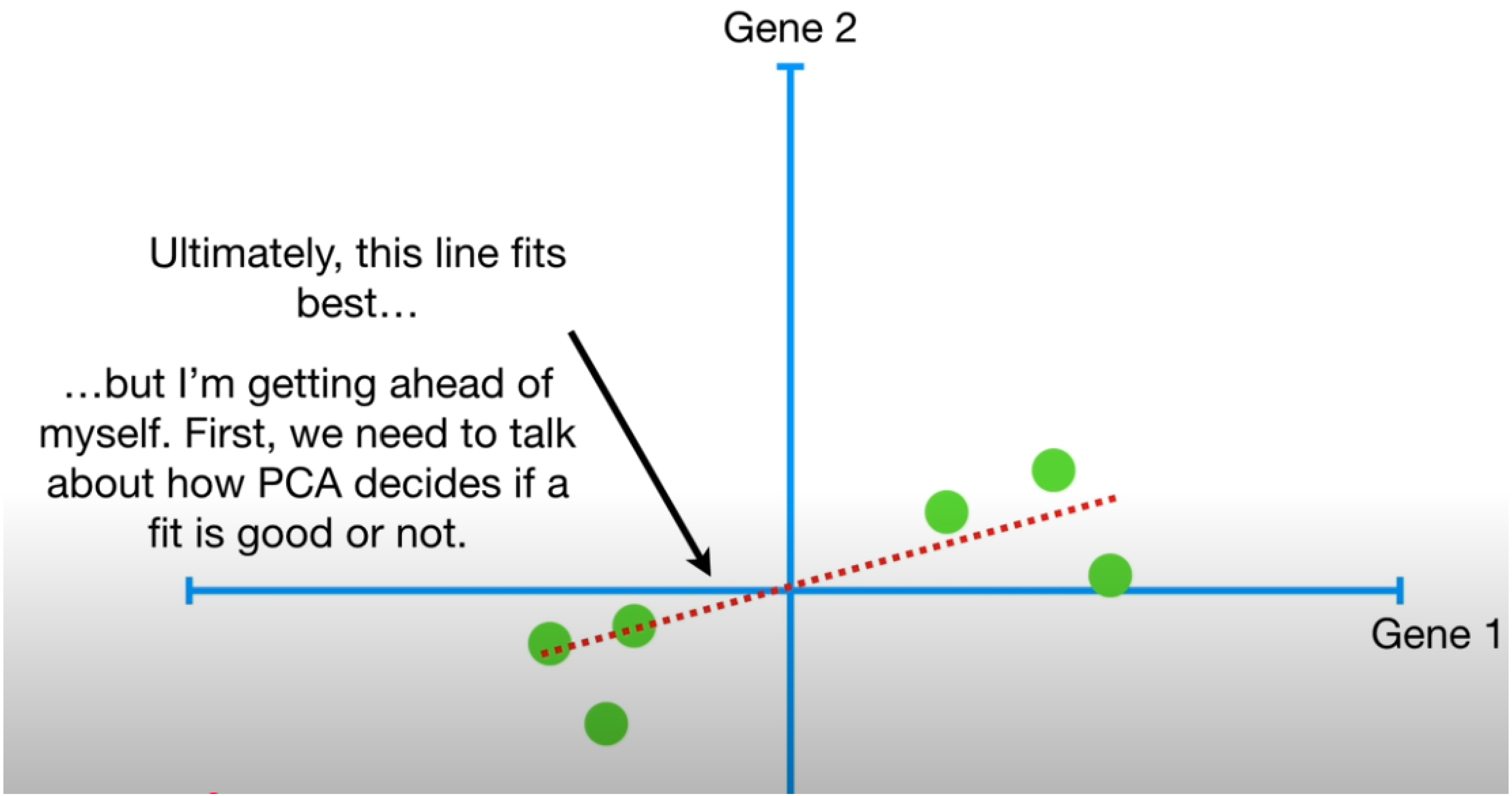
…we start by drawing a random line that goes through the origin…



…then we rotate the line until it fits the data as well is it can, given that it has to go through the origin.



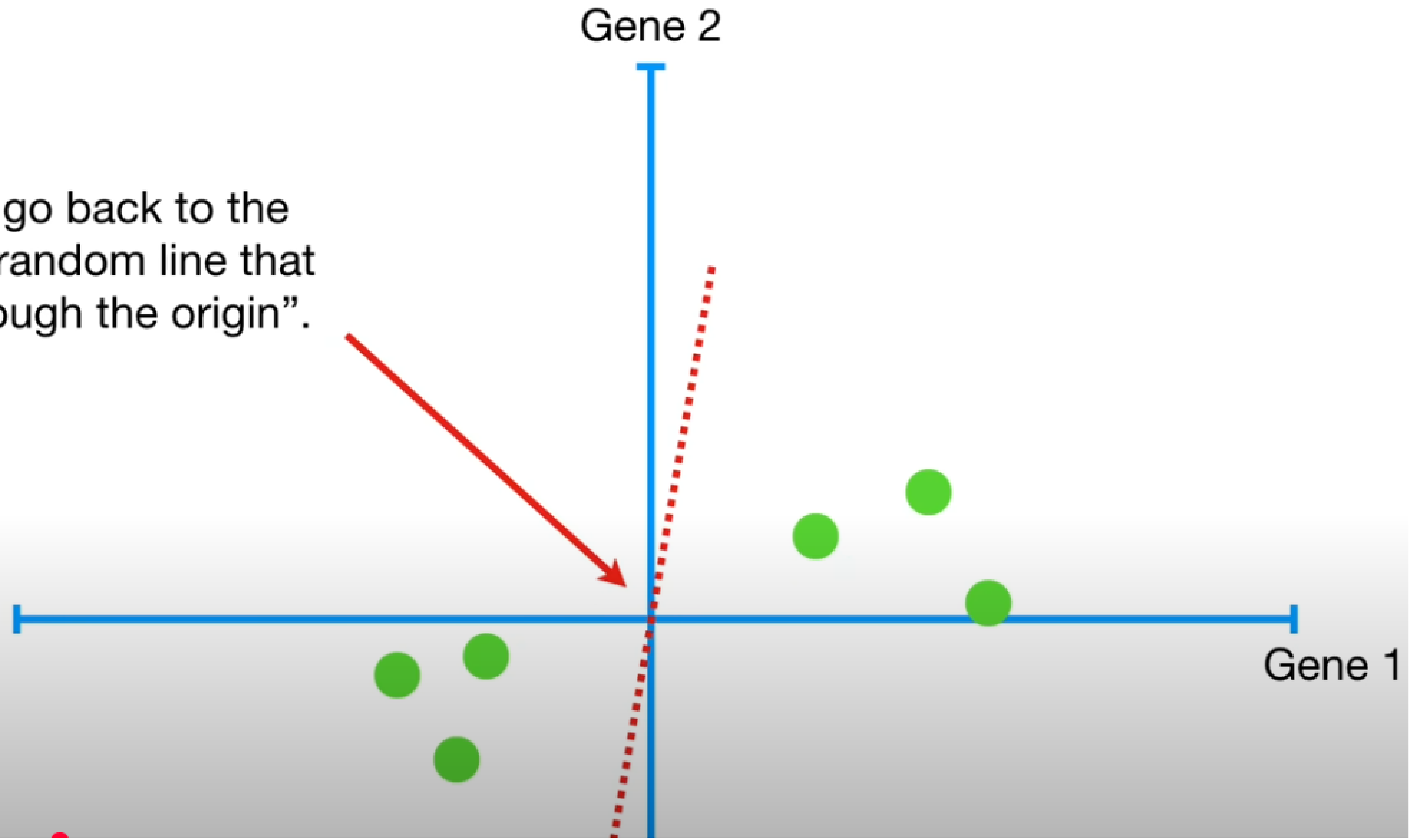
…then we rotate the line until it fits the data as well is it can, given that it has to go through the origin.

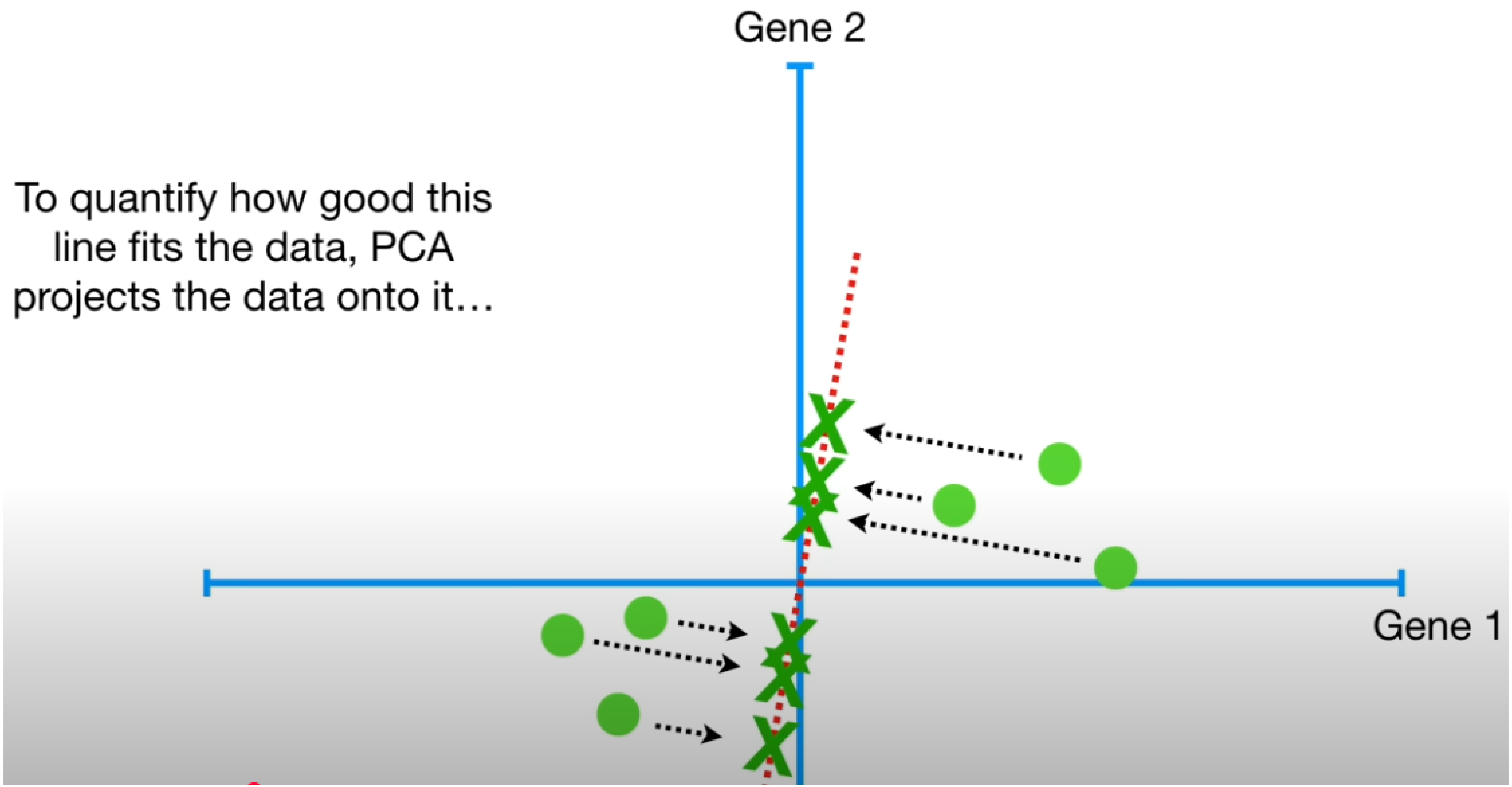


Ultimately, this line fits best…

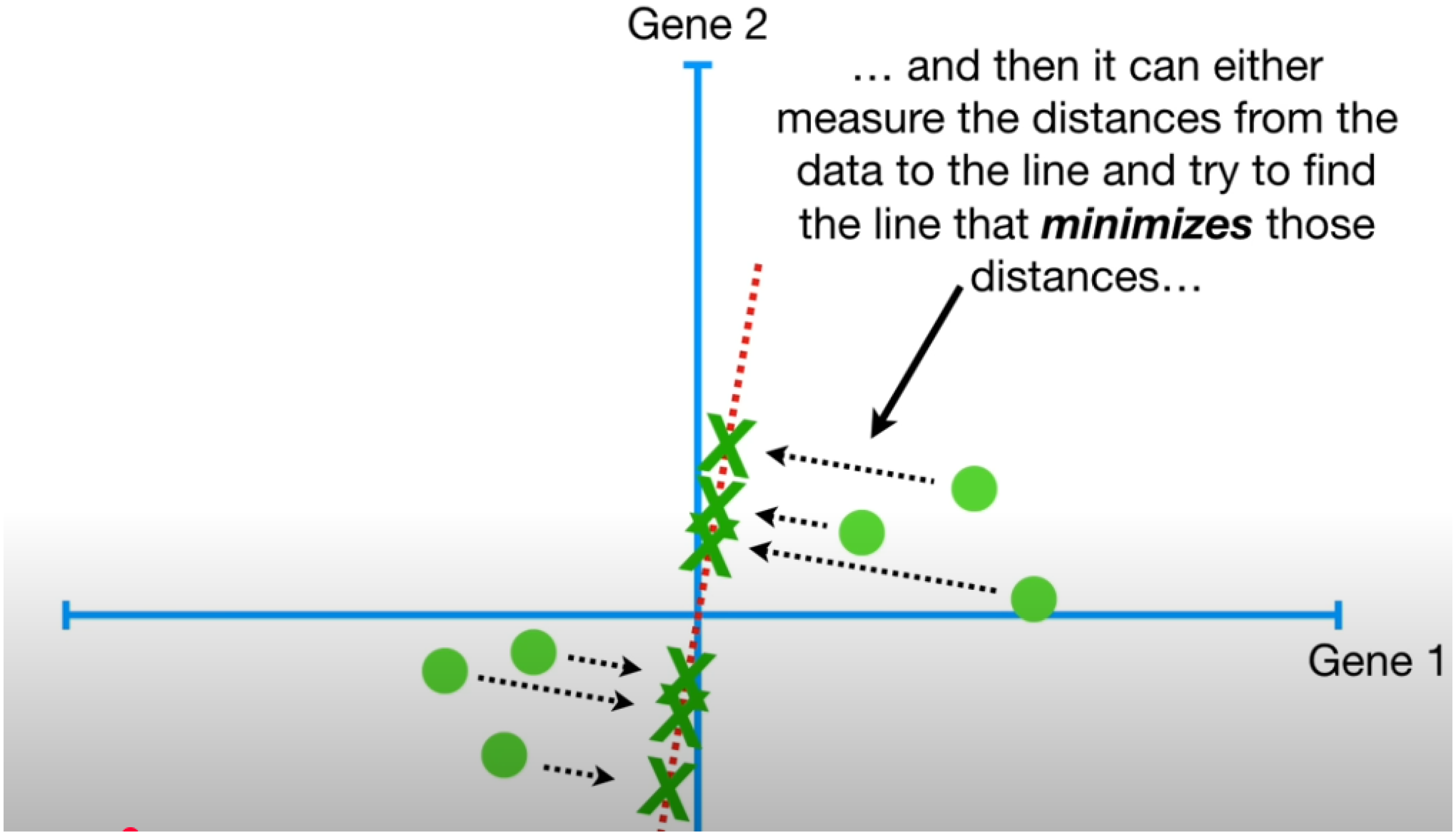
…but I'm getting ahead of myself. First, we need to talk about how PCA decides if a fit is good or not.

So let’s go back to the original “random line that goes through the origin”.

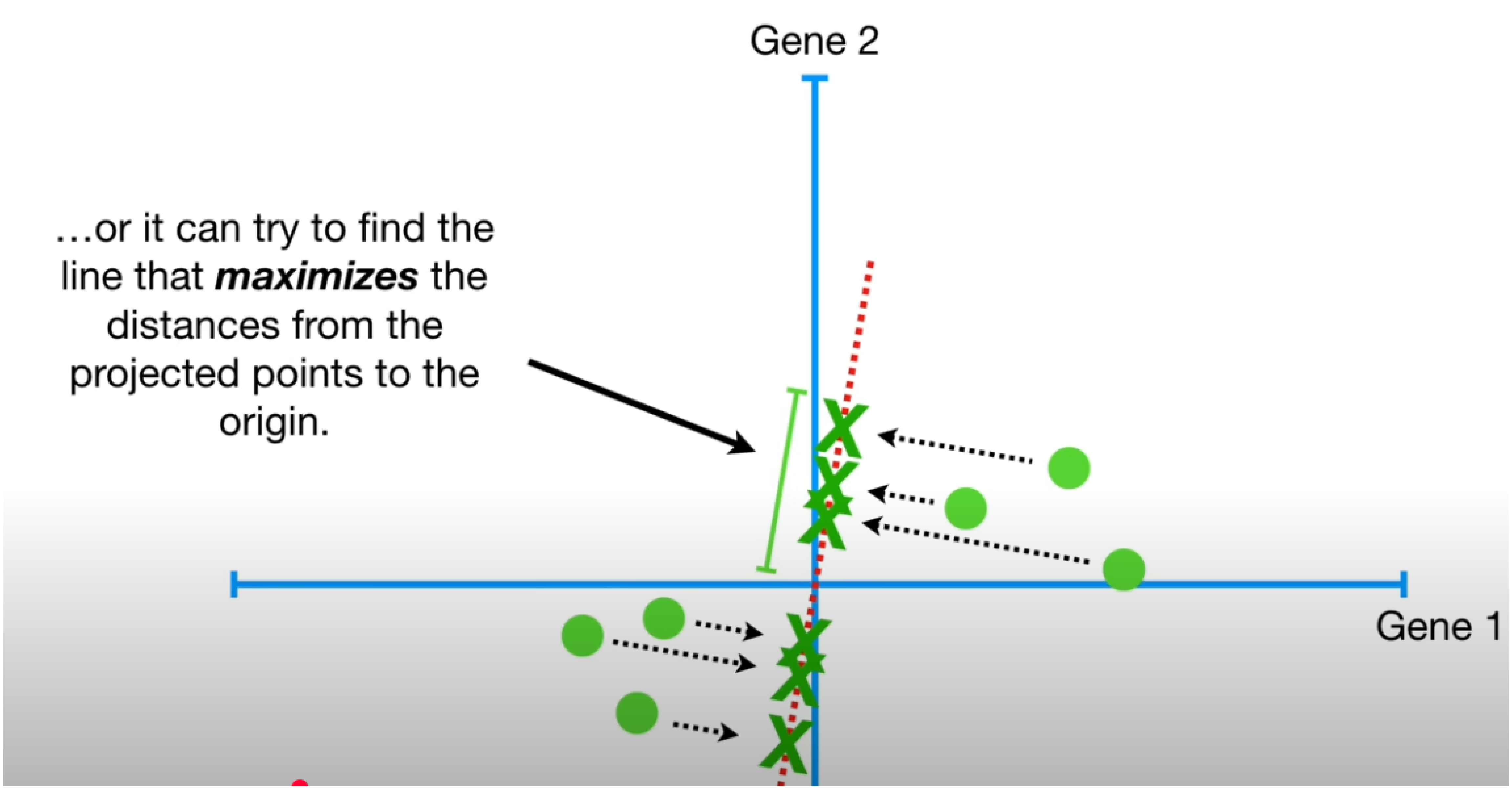




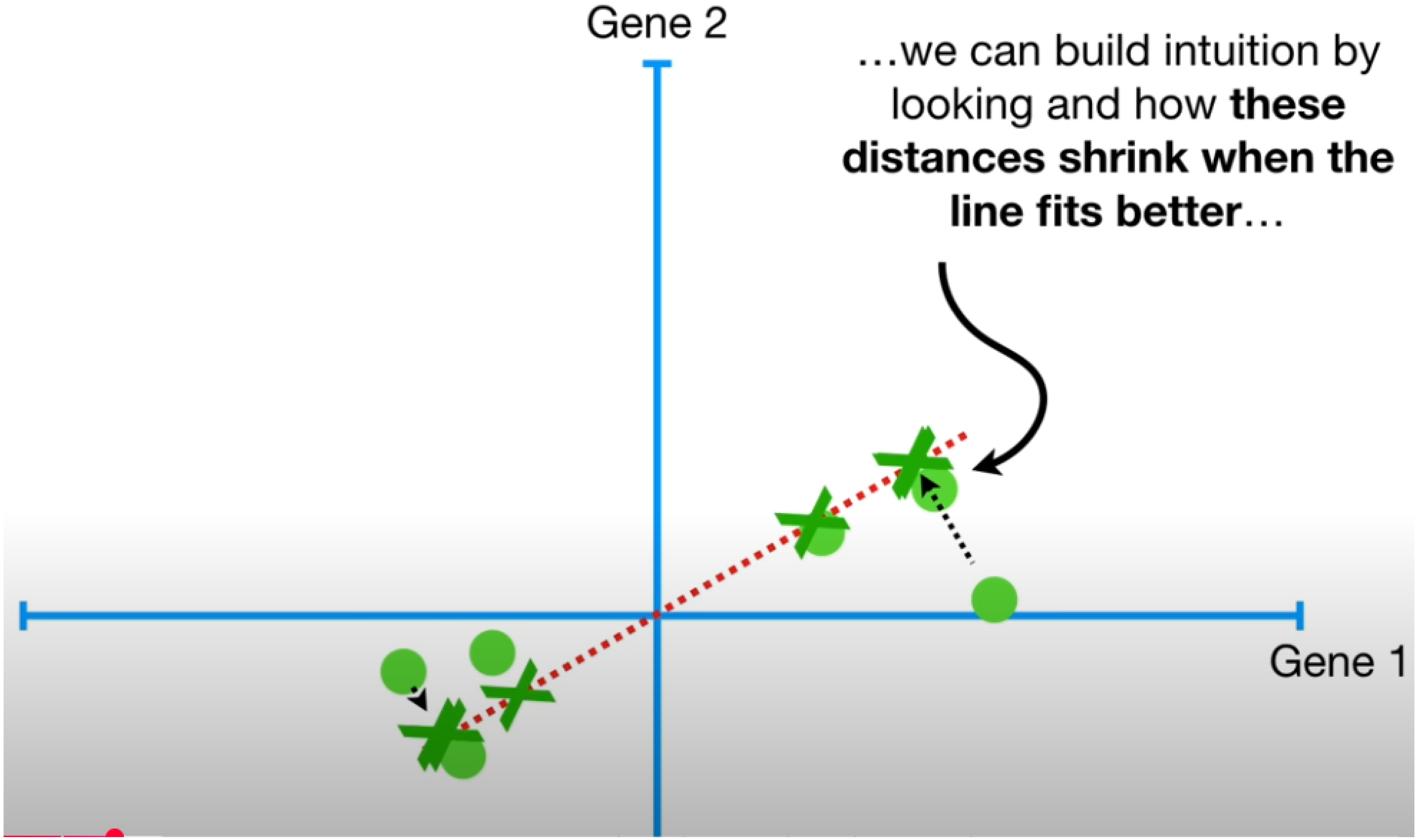
To quantify how good this line fits the data, PCA projects the data onto it…



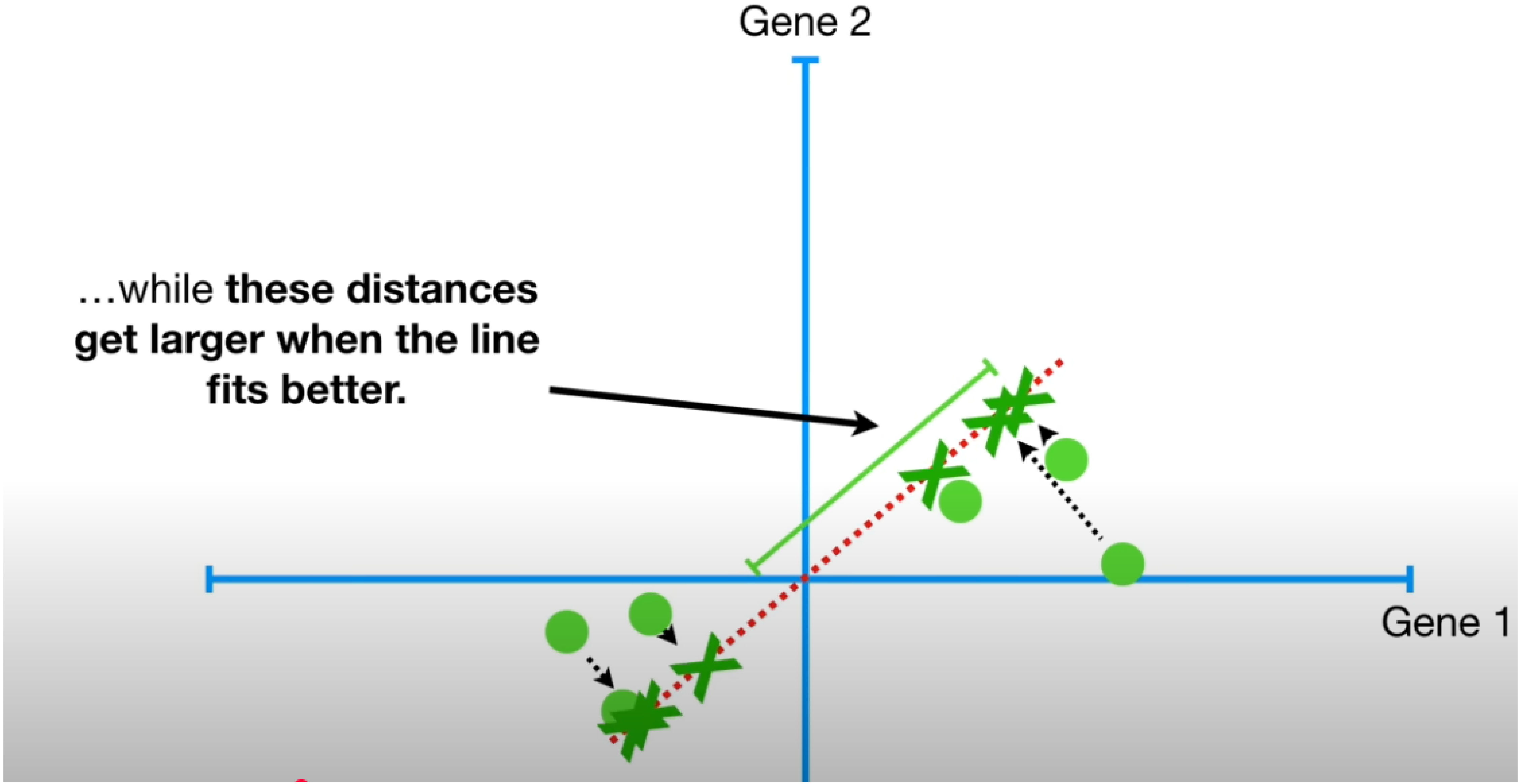
… and then it can either measure the distances from the data to the line and try to find the line that ***minimizes*** those distances…



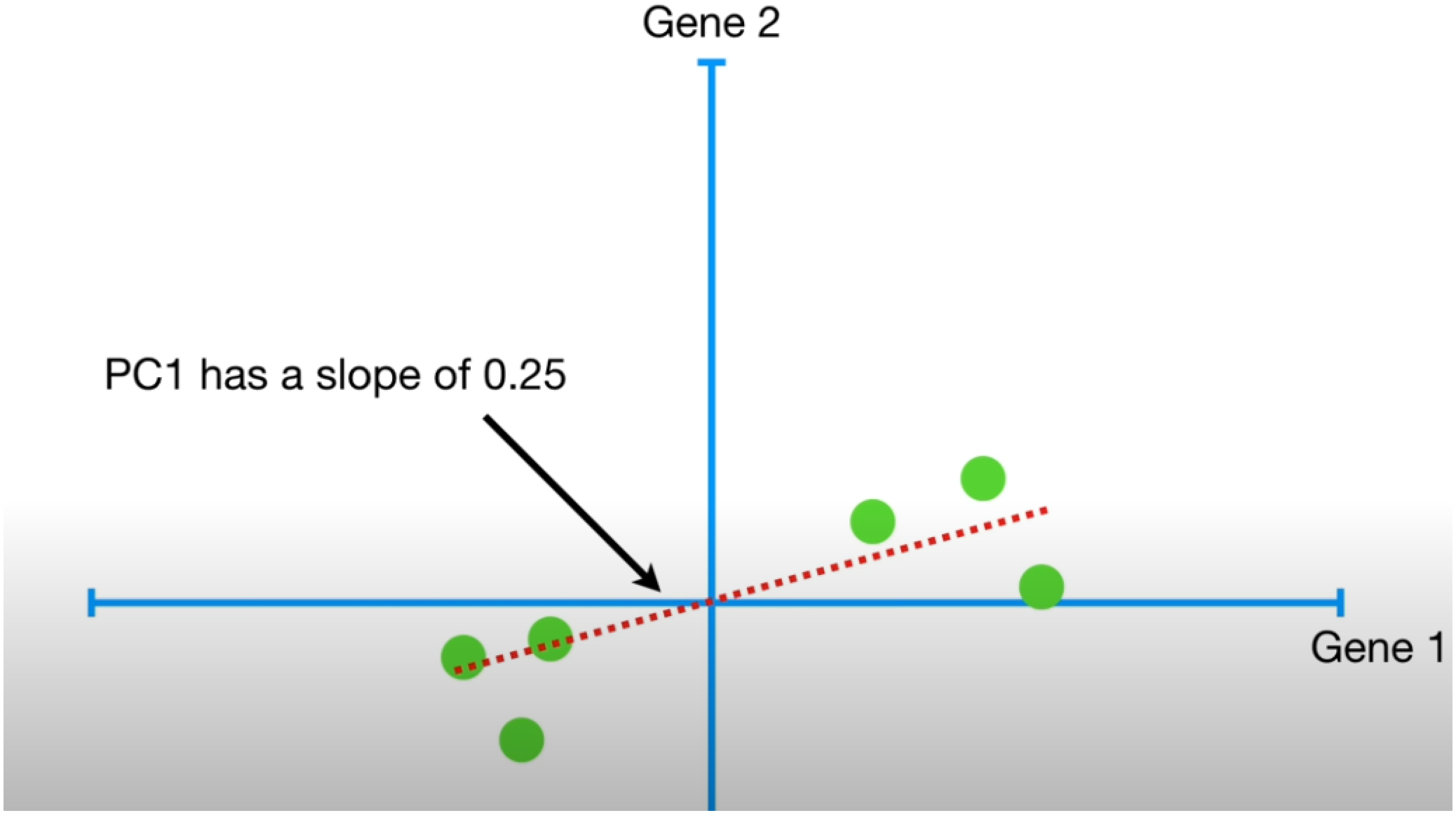
…or it can try to find the line that ***maximizes*** the distances from the projected points to the origin.



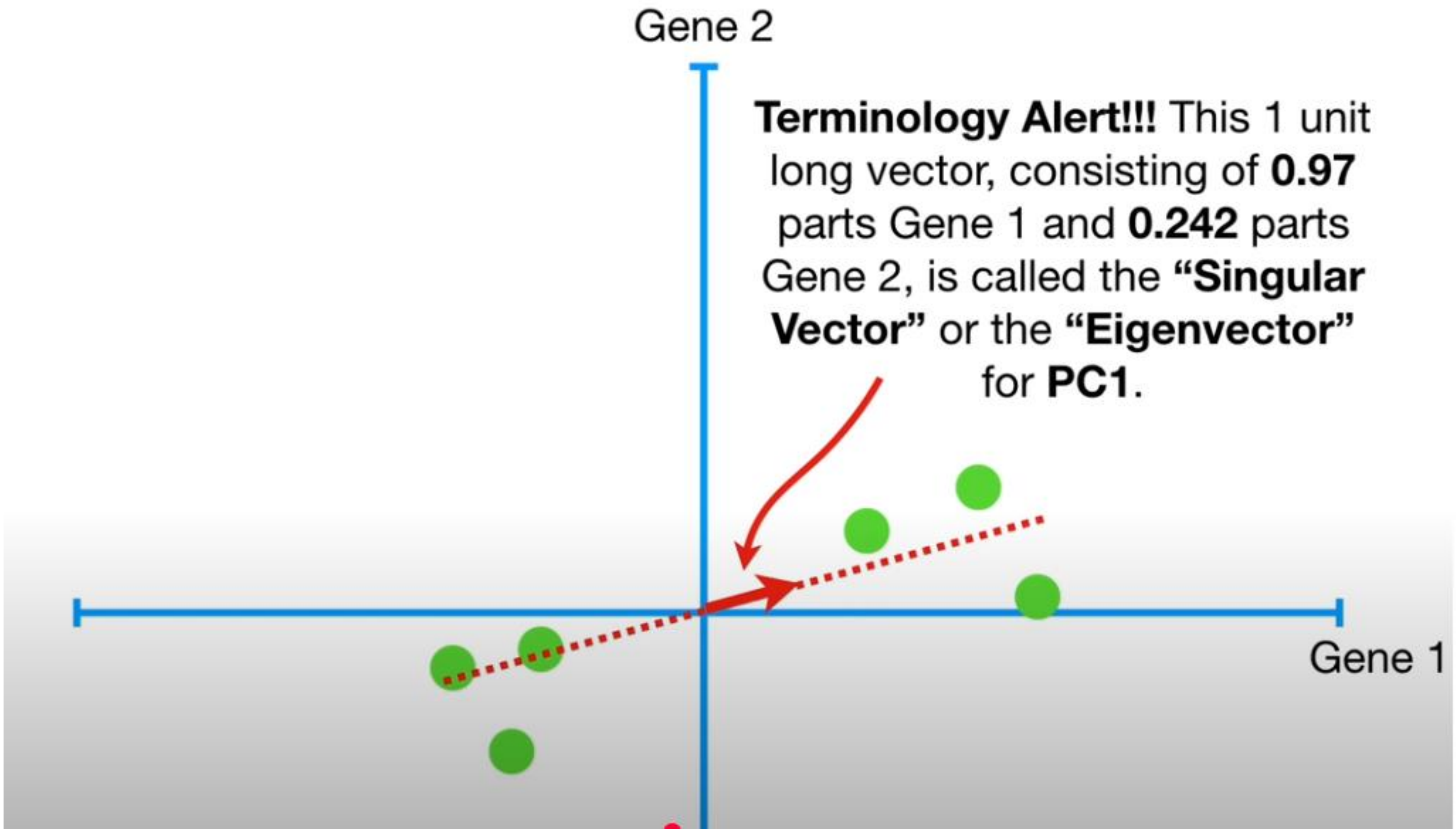
…we can build intuition by looking and how **these distances shrink when the line fits better**…



…while **these distances get larger when the line fits better.**



PC1 has a slope of 0.25



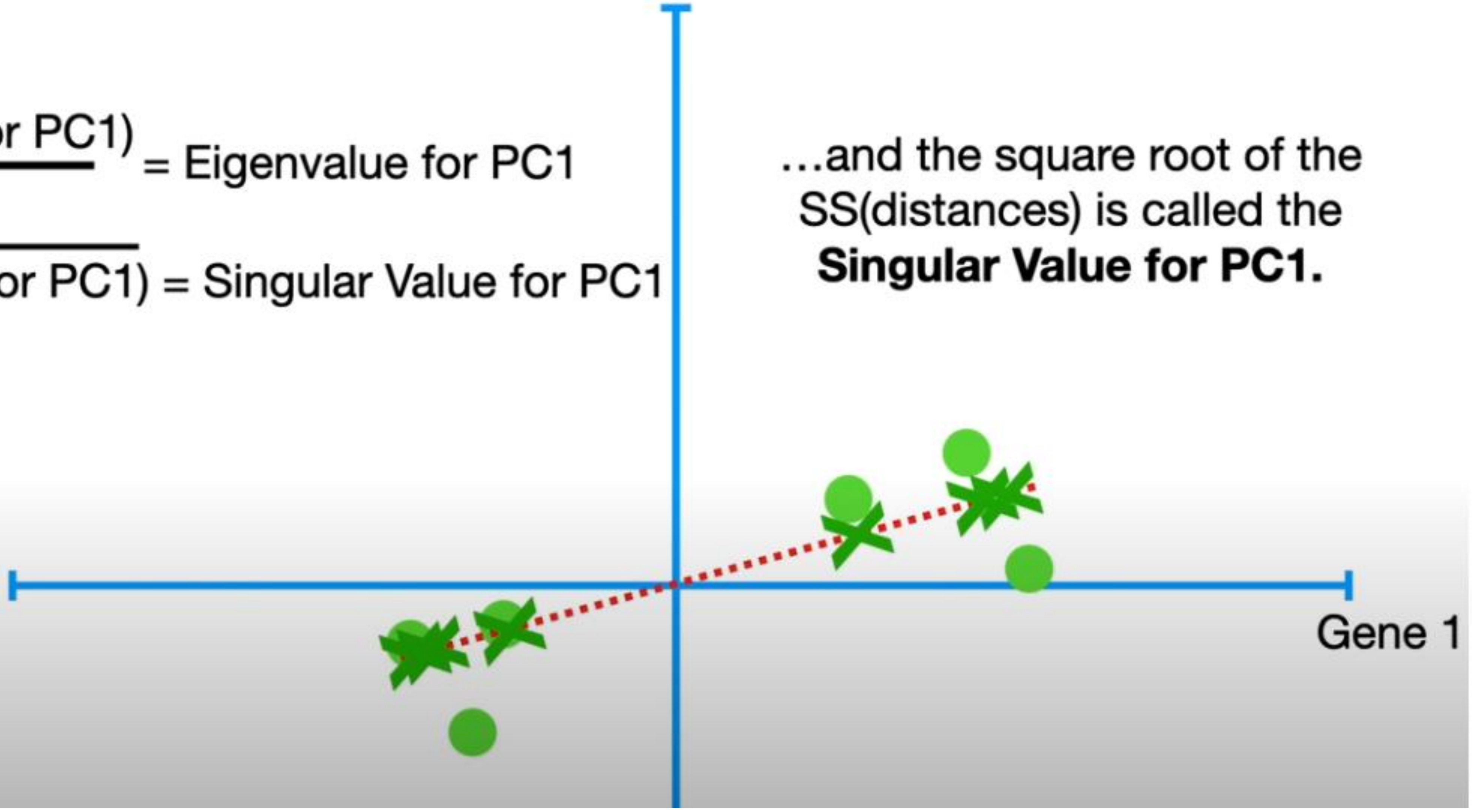
Terminology Alert!!! This 1 unit long vector, consisting of **0.97** parts Gene 1 and **0.242** parts Gene 2, is called the **“Singular Vector”** or the **“Eigenvector”** for **PC1**.

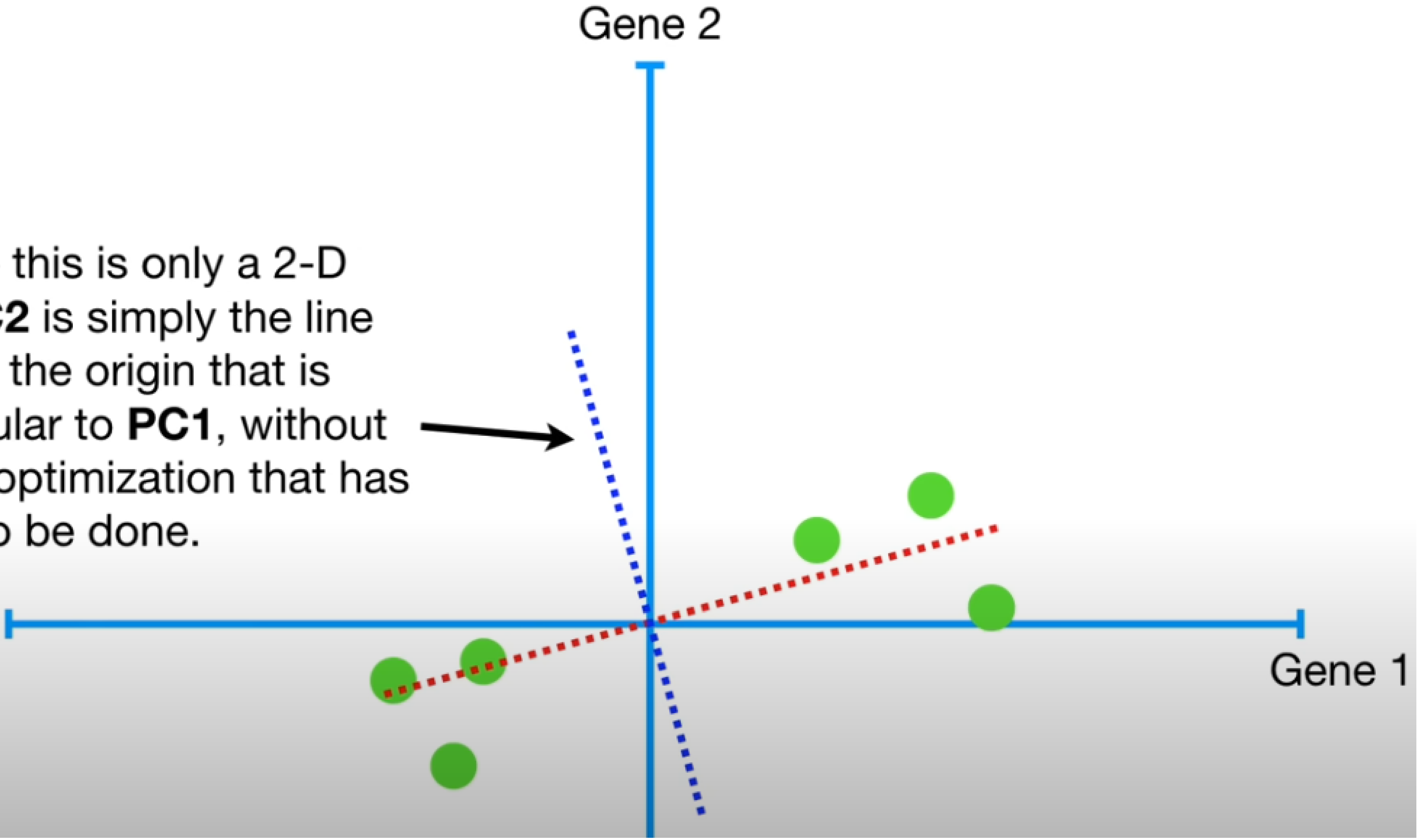
$$\boldsymbol{d_1}^2 + \boldsymbol{d_2}^2 + \boldsymbol{d_3}^2 + \boldsymbol{d_4}^2 + \boldsymbol{d_5}^2 + \boldsymbol{d_6}^2 = \text{sum of squared distances} = \text{SS(distances)}$$

$$\frac{\text{SS(distances for PC1)}}{n - 1} = \text{Eigenvalue for PC1}$$

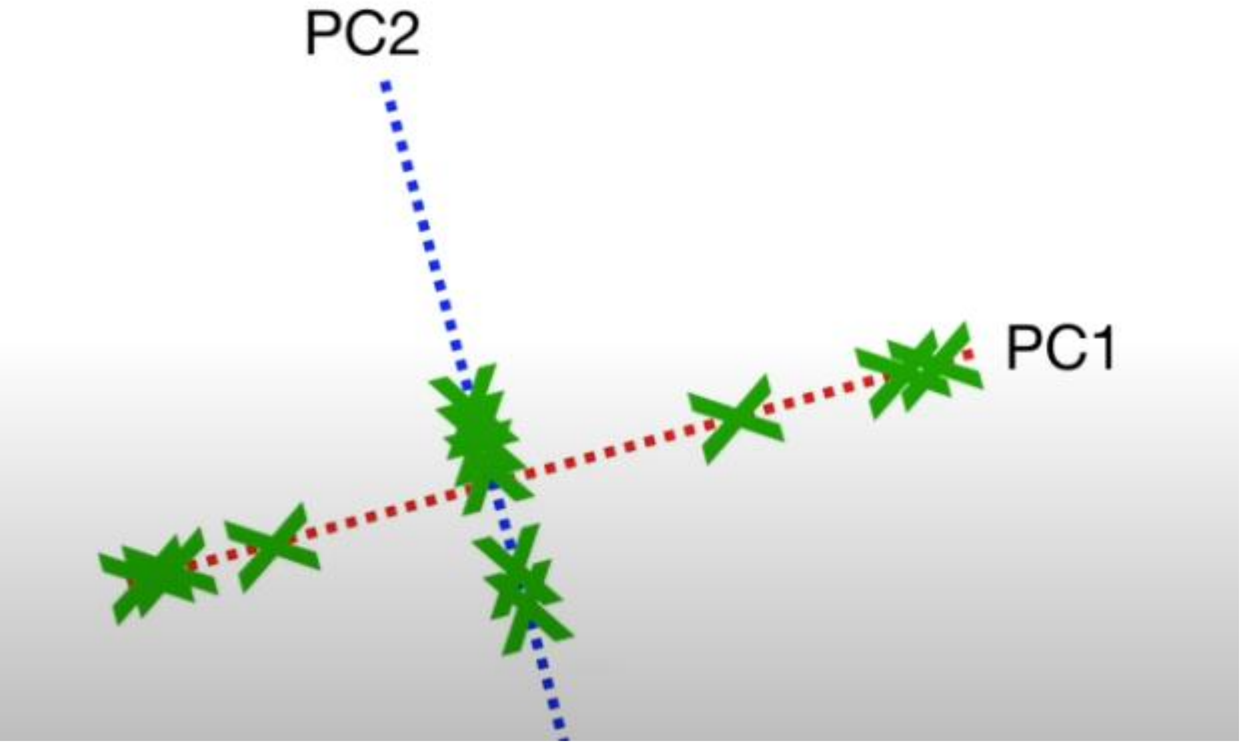
$$\sqrt{\text{SS(distances for PC1)}} = \text{Singular Value for PC1}$$

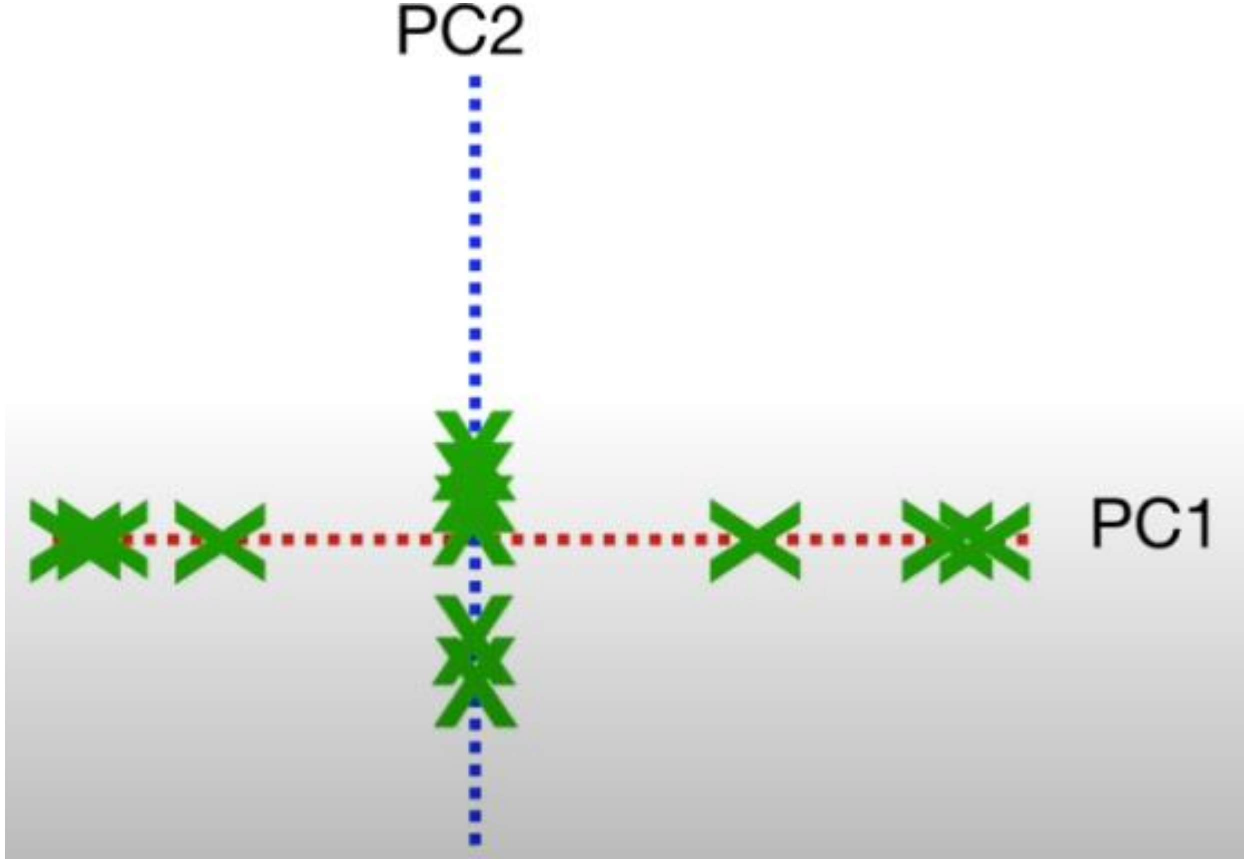
...and the square root of the SS(distances) is called the **Singular Value for PC1.**



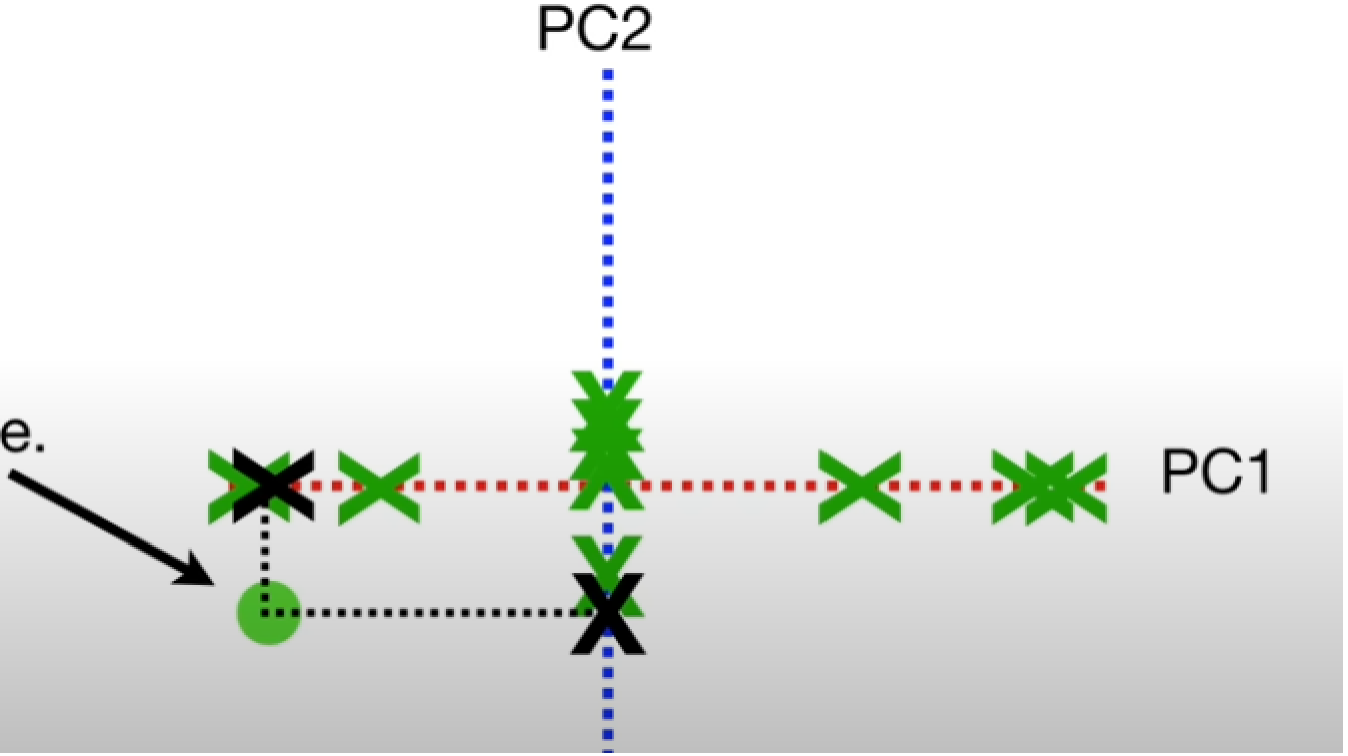


Because this is only a 2-D graph, **PC2** is simply the line through the origin that is perpendicular to **PC1**, without any further optimization that has to be done.


PC2
PC1

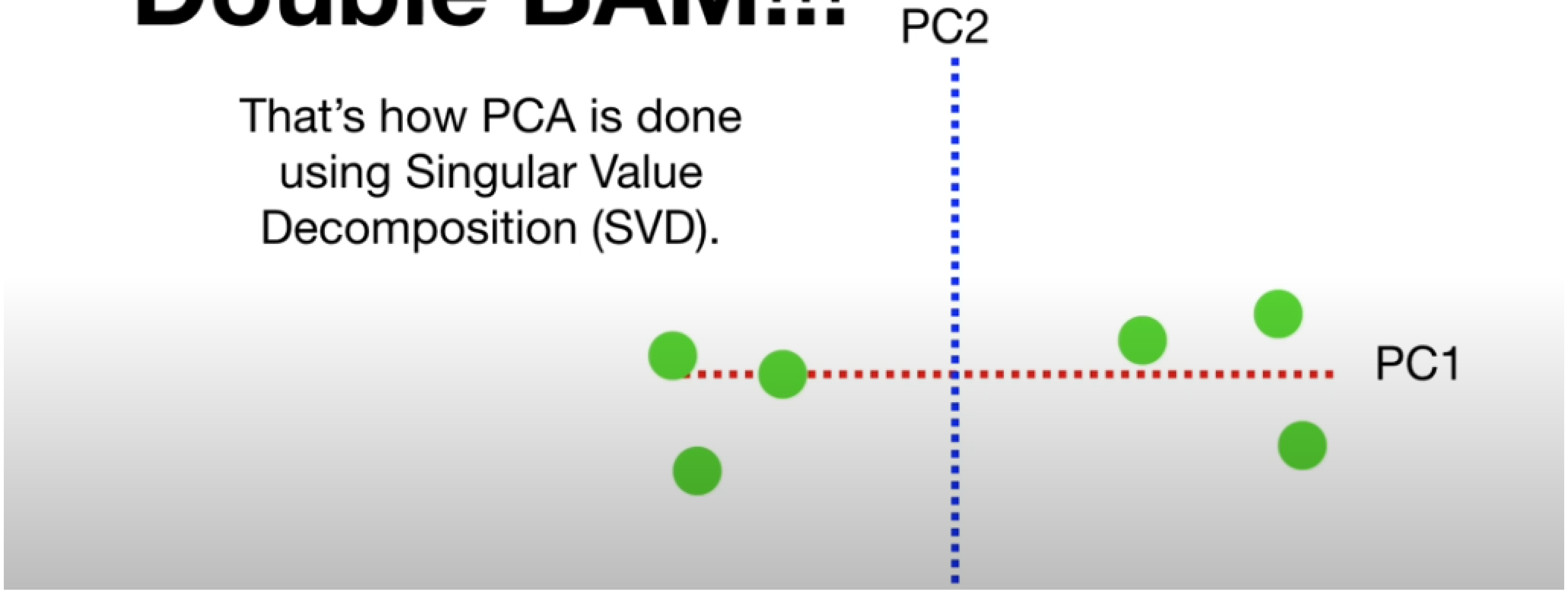

PC2
PC1

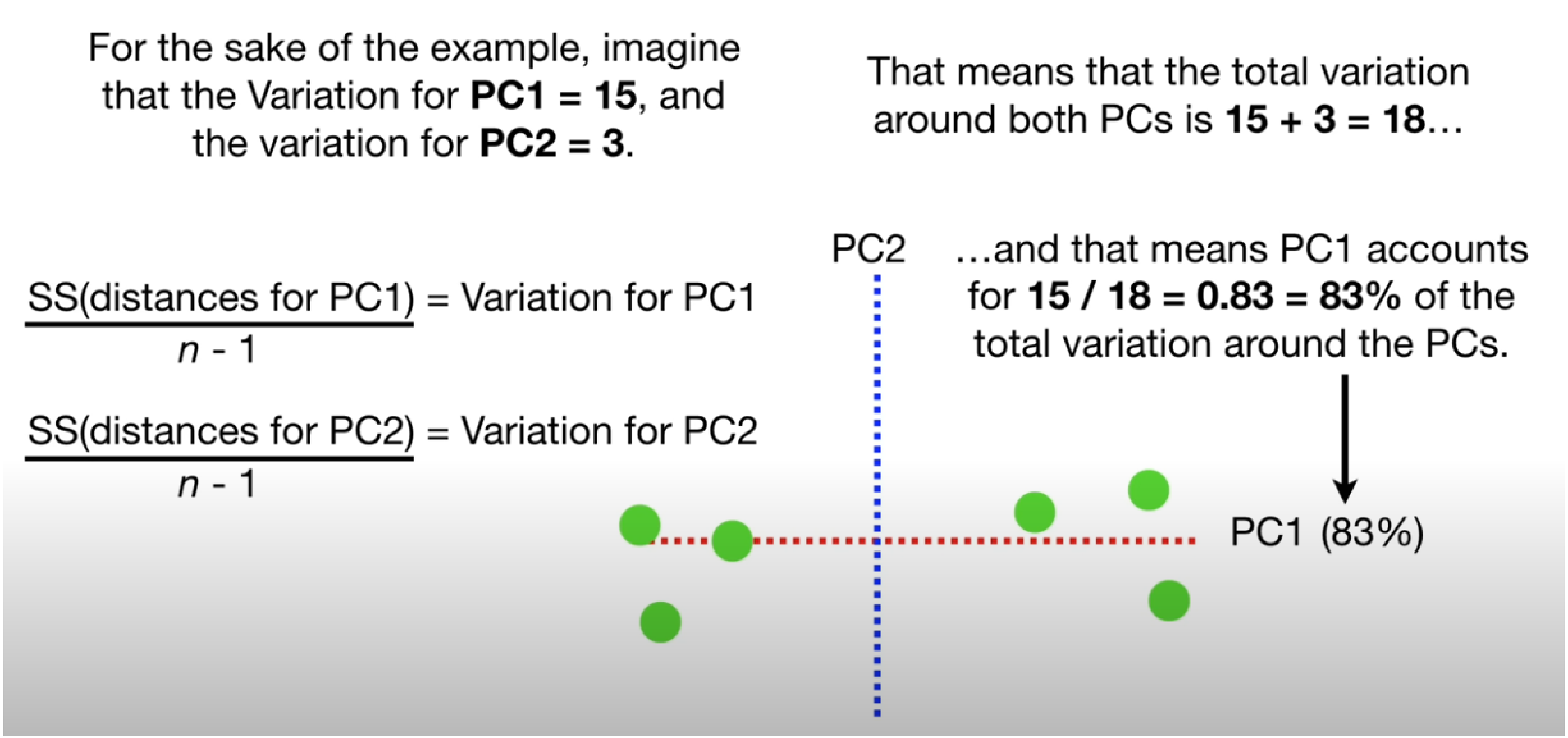
…so Sample 6 goes here.



Double BAM!!!

That's how PCA is done using Singular Value Decomposition (SVD).





For the sake of the example, imagine that the Variation for **PC1 = 15**, and the variation for **PC2 = 3**.

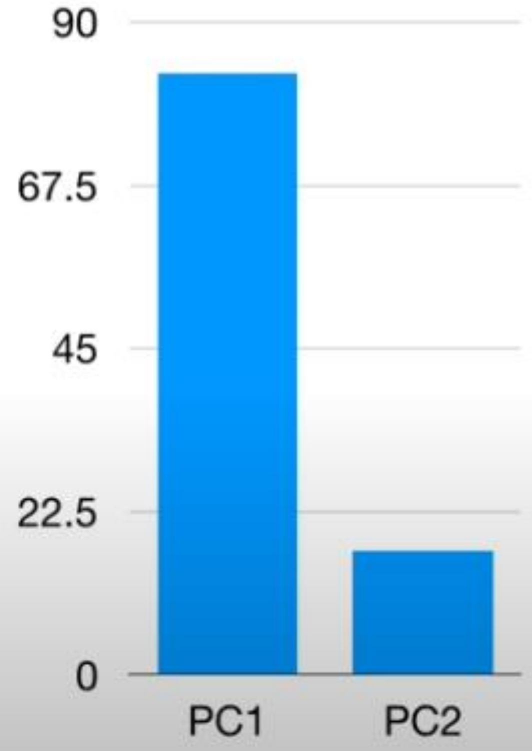
That means that the total variation around both PCs is **15 + 3 = 18**…

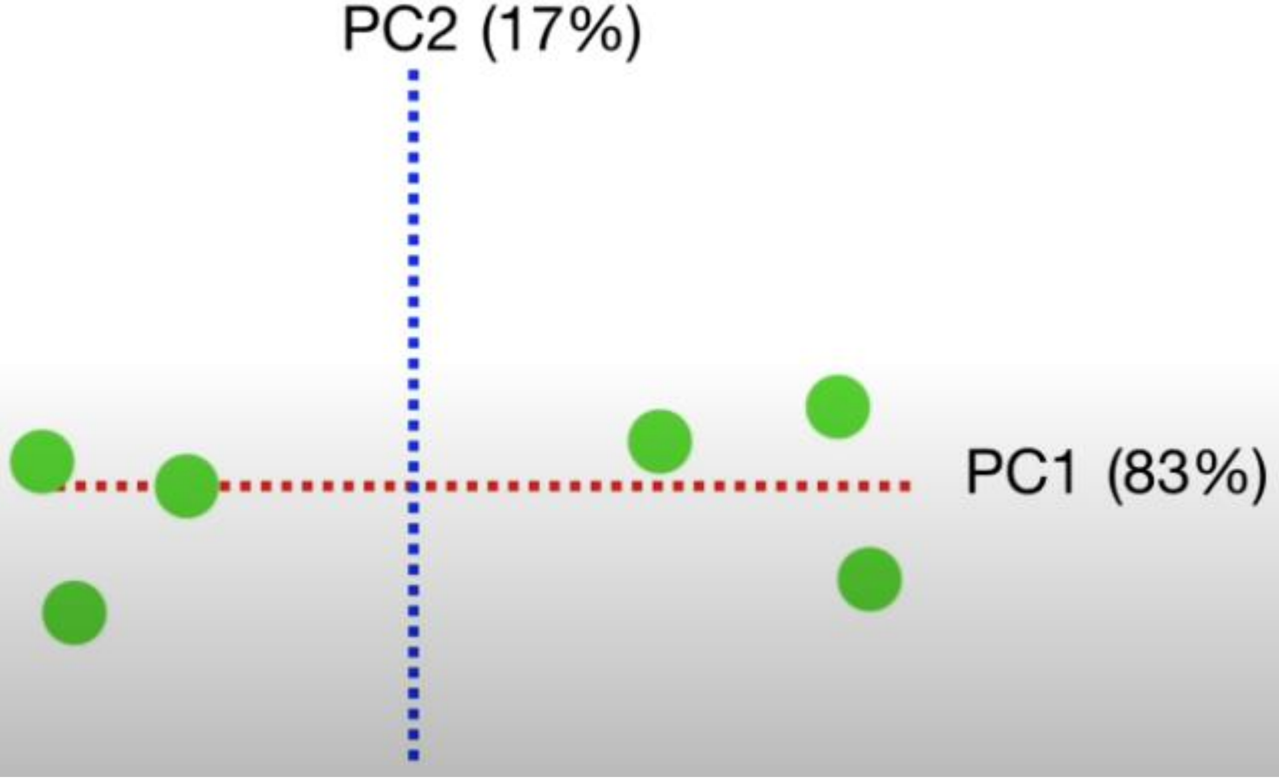
$$\frac{\text{SS(distances for PC1)}}{n - 1} = \text{Variation for PC1}$$

$$\frac{\text{SS(distances for PC2)}}{n - 1} = \text{Variation for PC2}$$

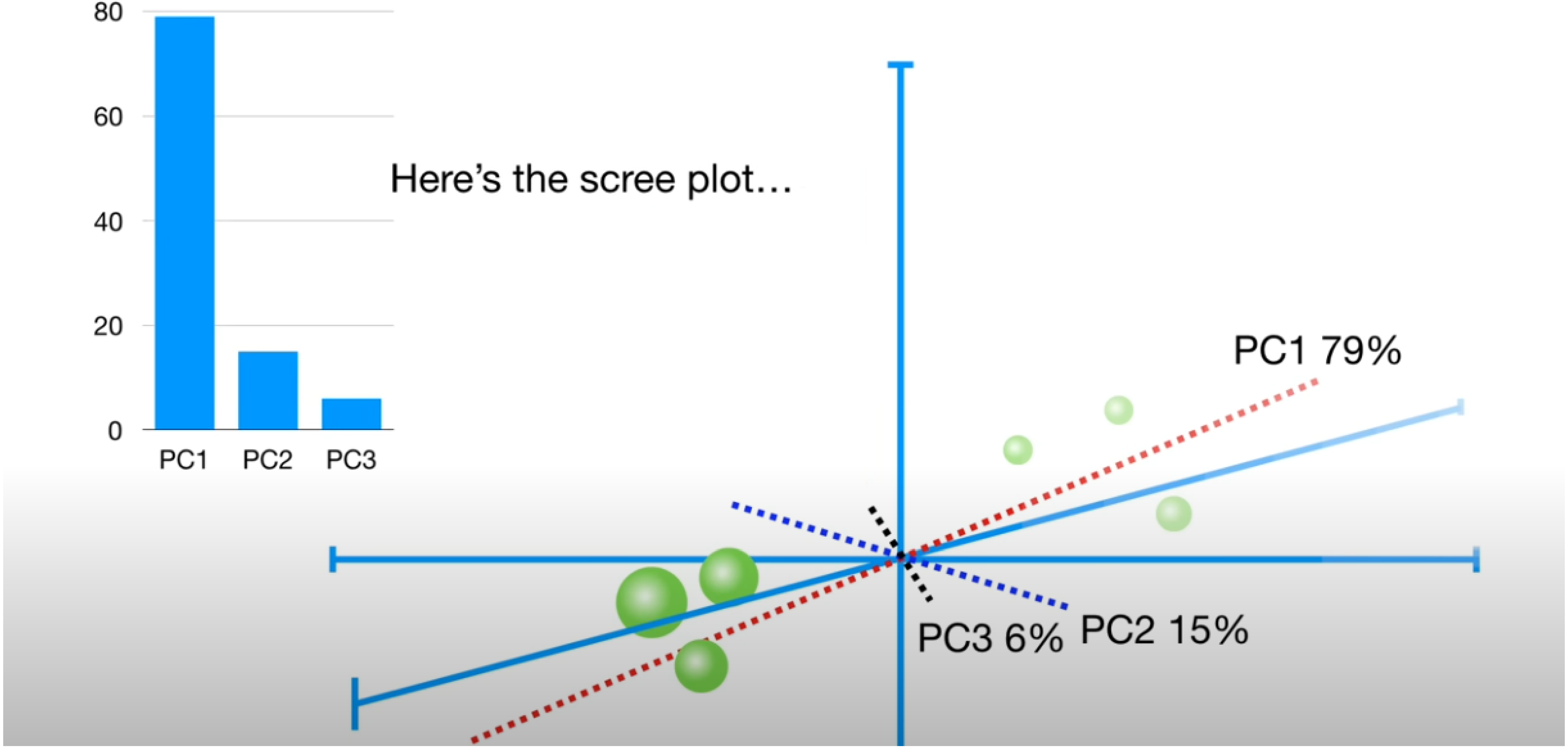
…and that means PC1 accounts for **15 / 18 = 0.83 = 83%** of the total variation around the PCs.

TERMINOLOGY ALERT!!!! A **Scree Plot** is a graphical representation of the percentages of variation that each PC accounts for.





3D



Here’s the scree plot…