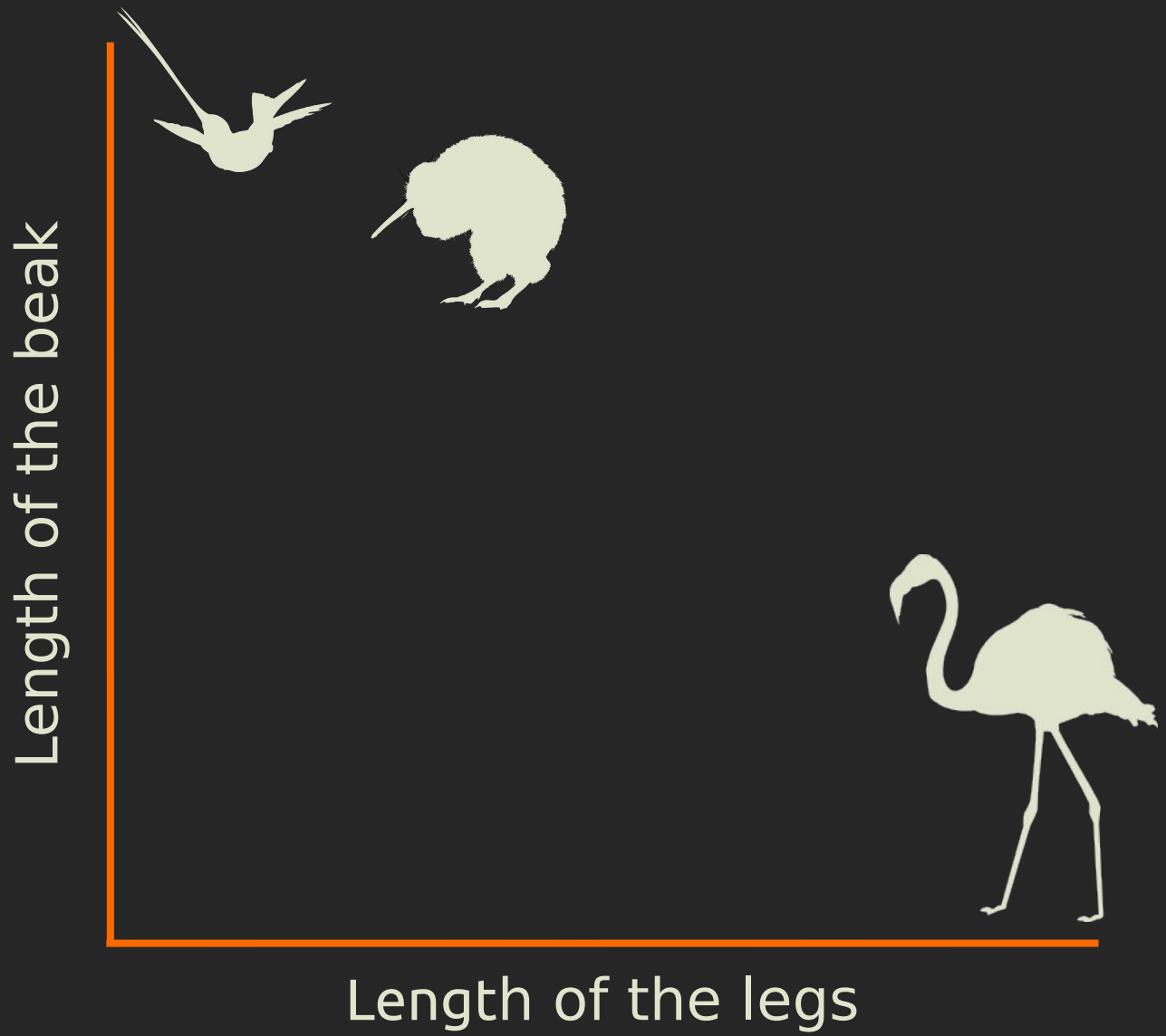
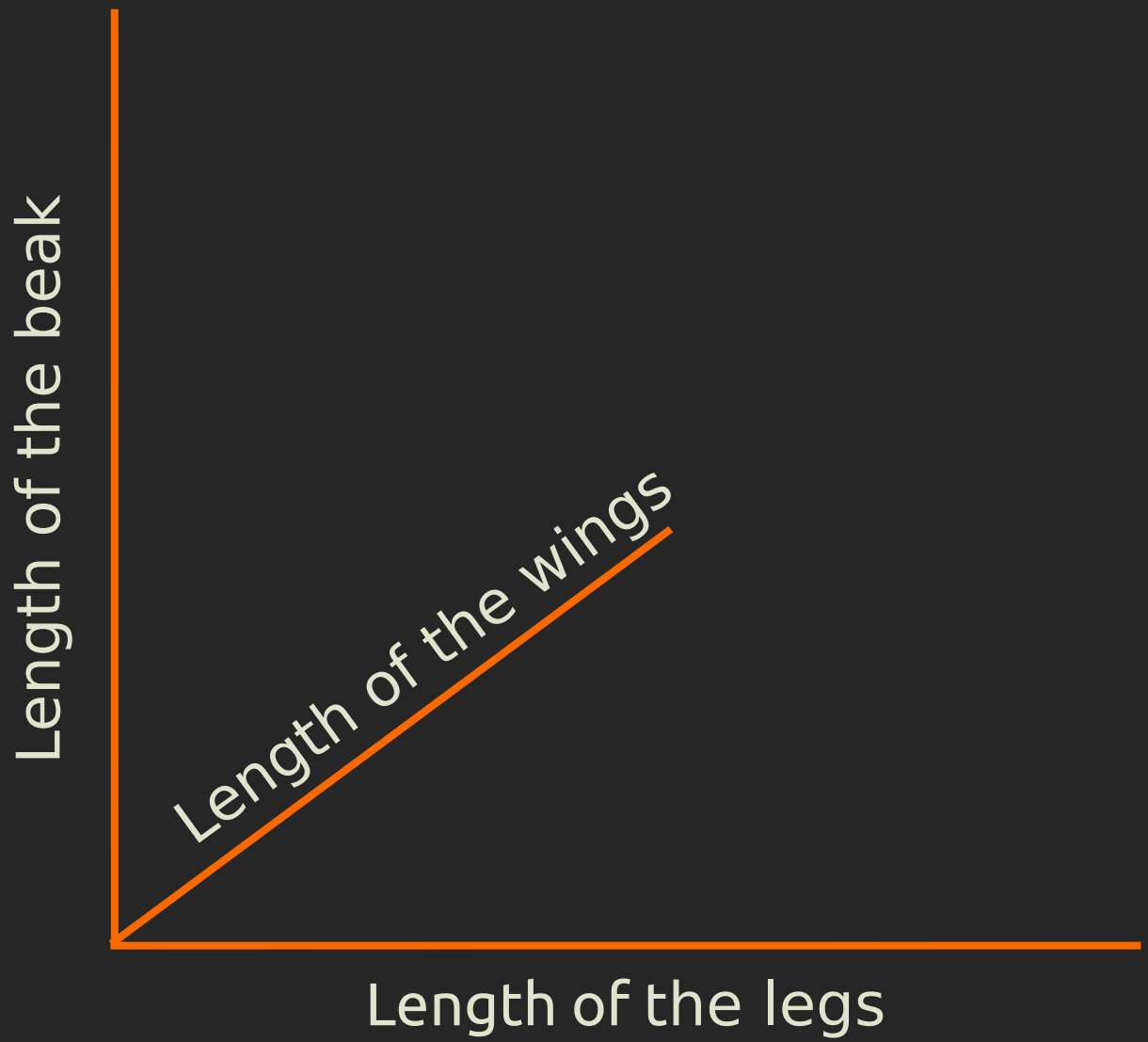
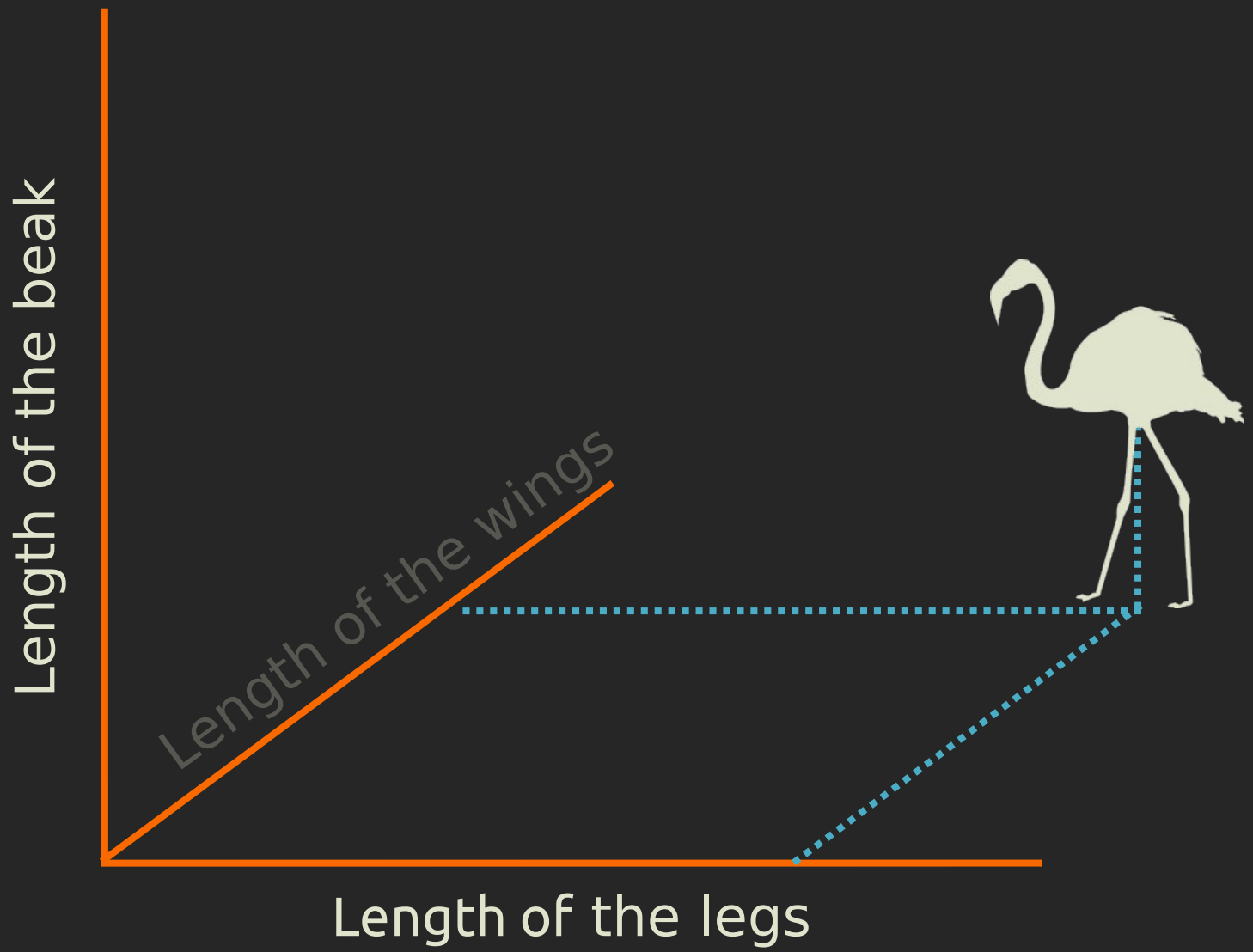
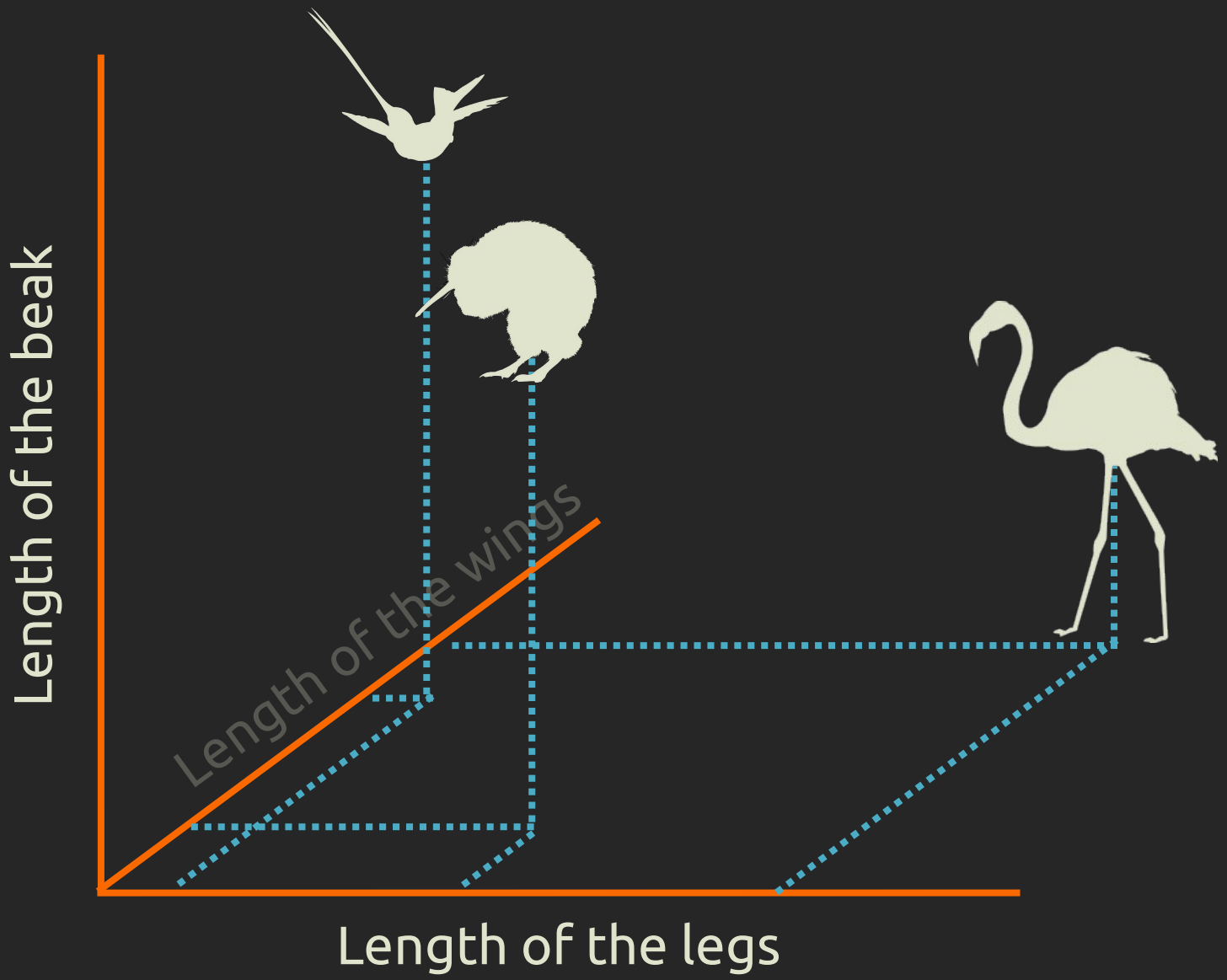
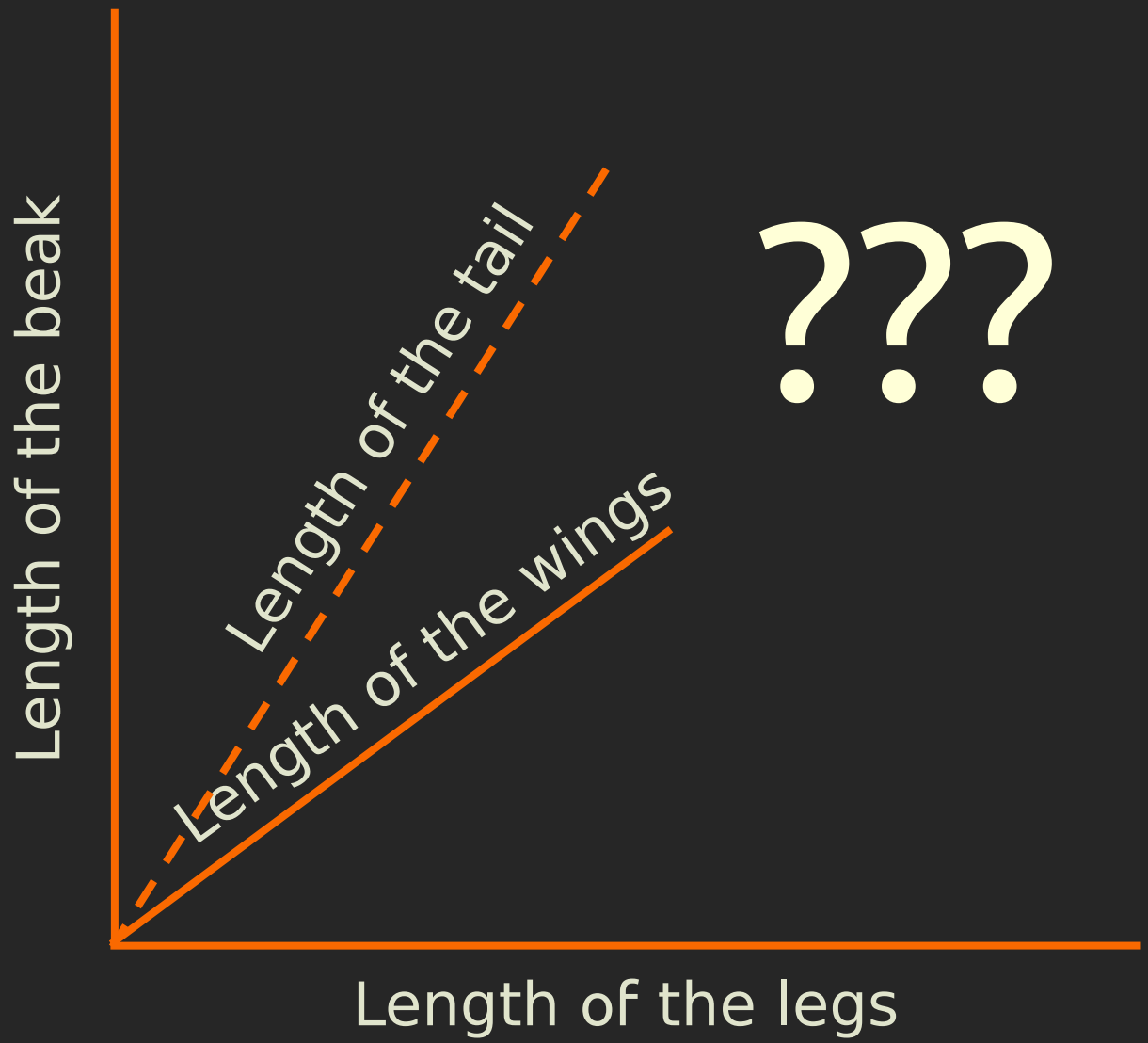
Ordinations


Length of the beak
Length of the legs


Length of the beak
Length of the wings
Length of the legs


Length of the beak
Length of the wings
Length of the legs

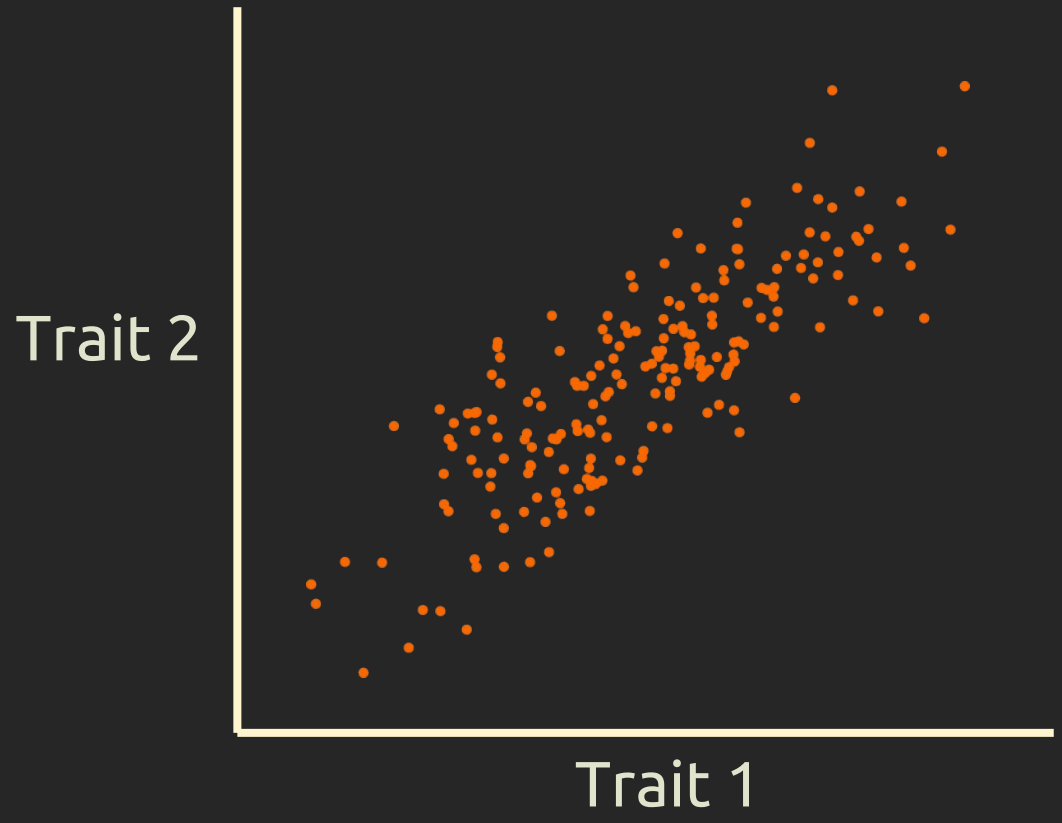

Length of the beak
Length of the wings
Length of the legs


Length of the beak
Length of the tail
Length of the wings
???
Length of the legs

PCA

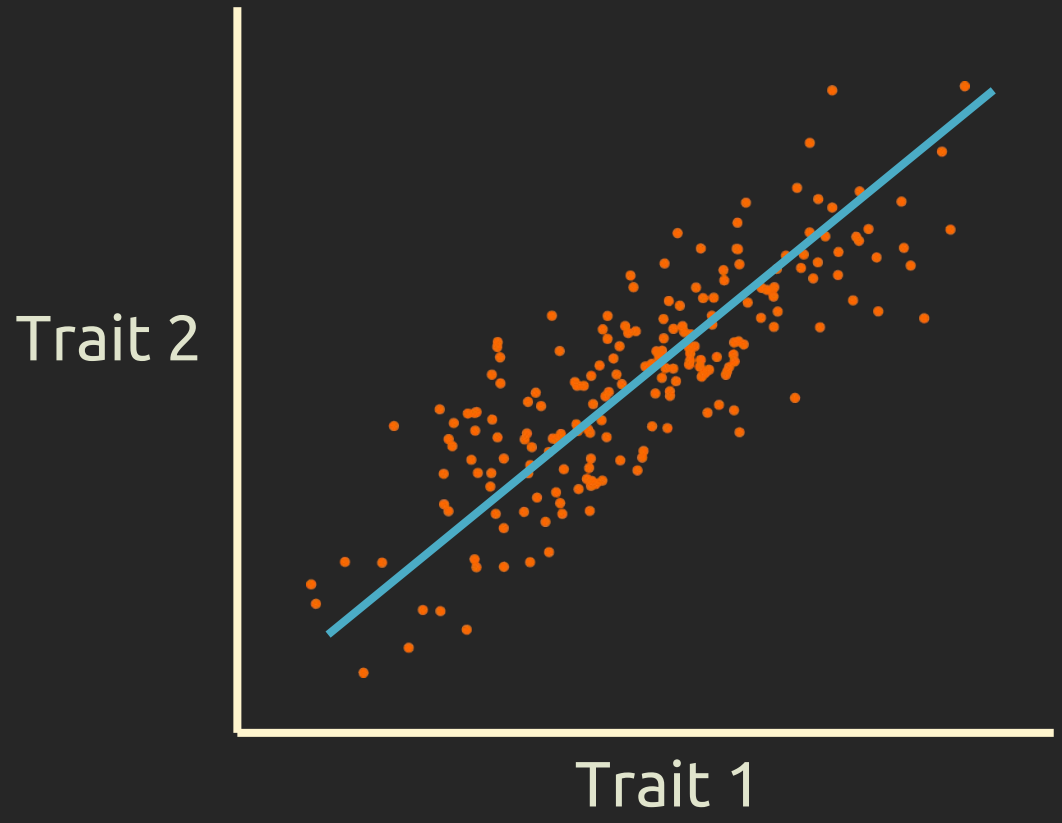
TL;DR:

PCA = rotating and stretching your data to maximise variance covariance



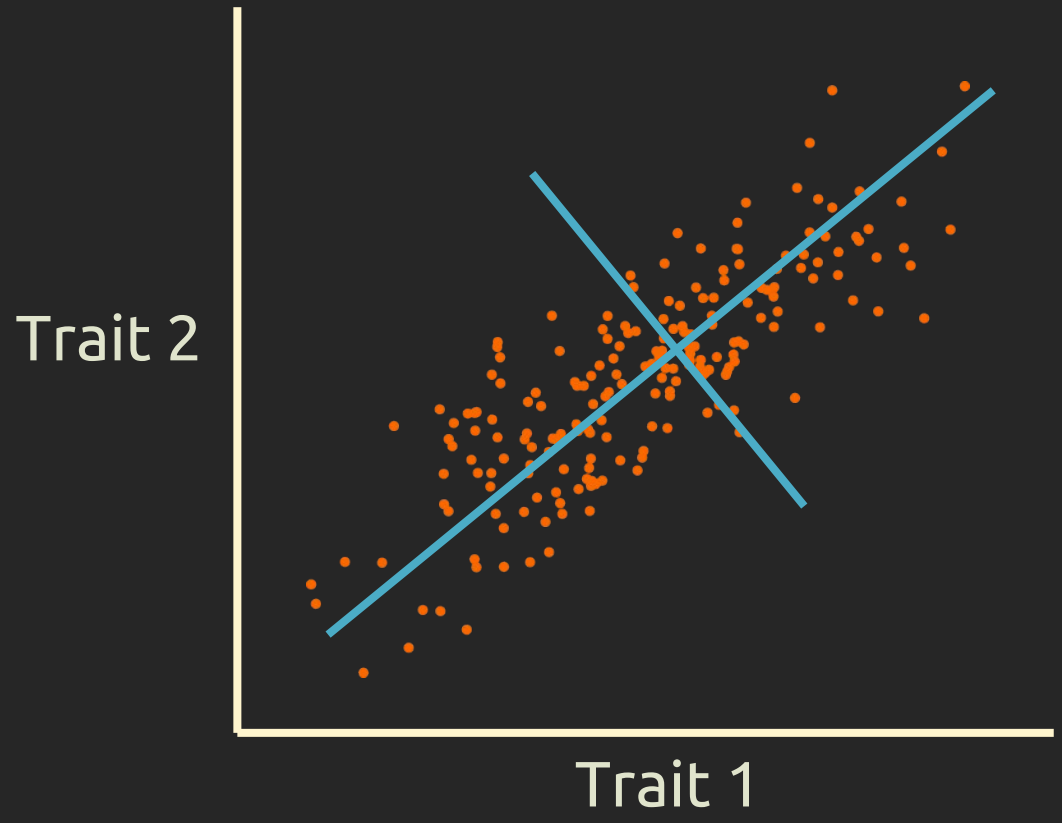
TL;DR:

PCA = rotating and stretching your data to maximise variance covariance



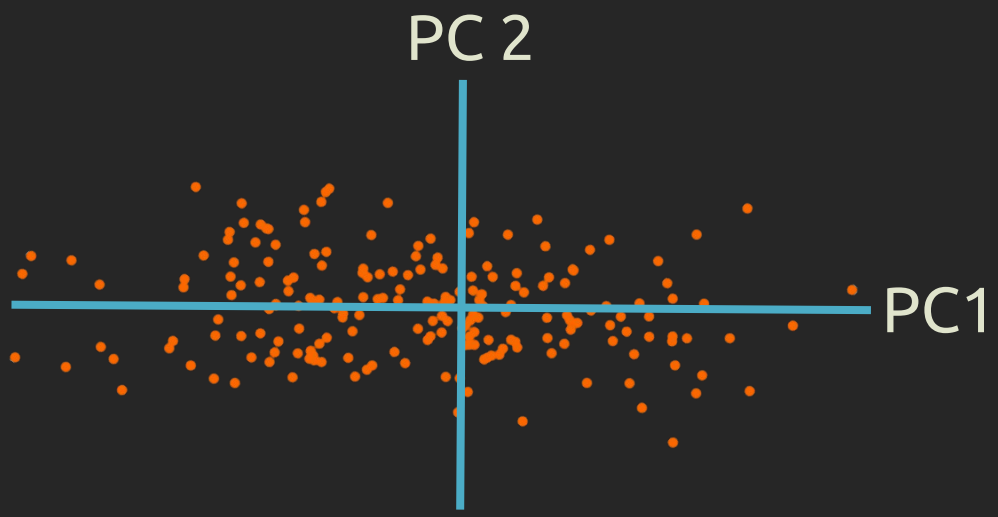
TL;DR:

PCA = rotating and stretching your data to maximise variance covariance

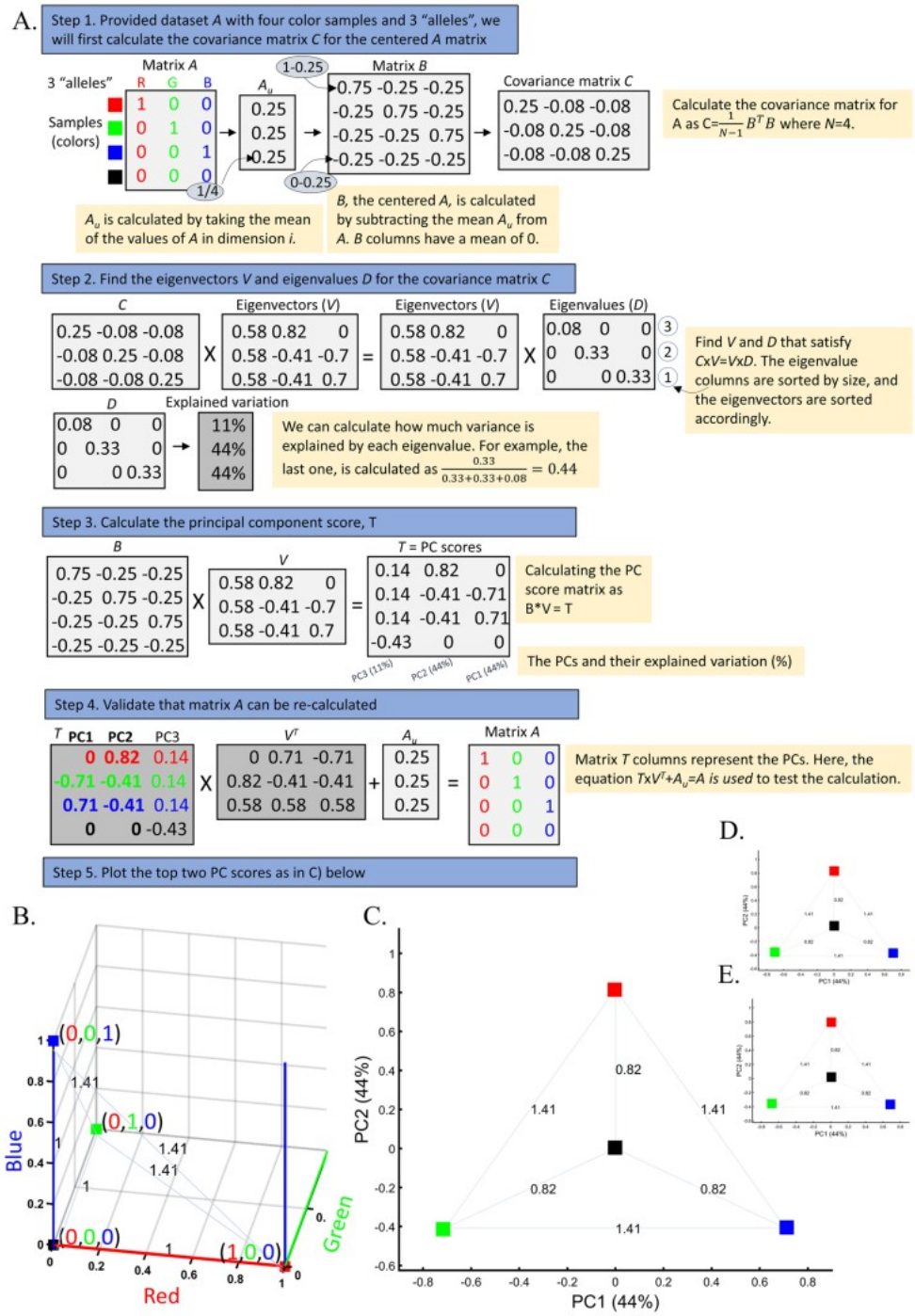


TL;DR:

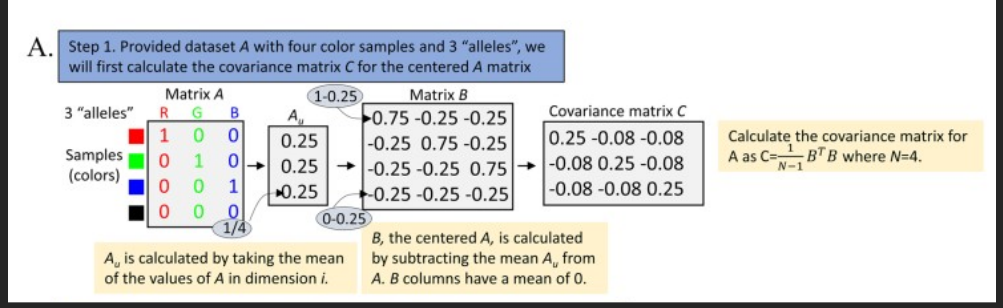
PCA = rotating and stretching your data to maximise variance covariance



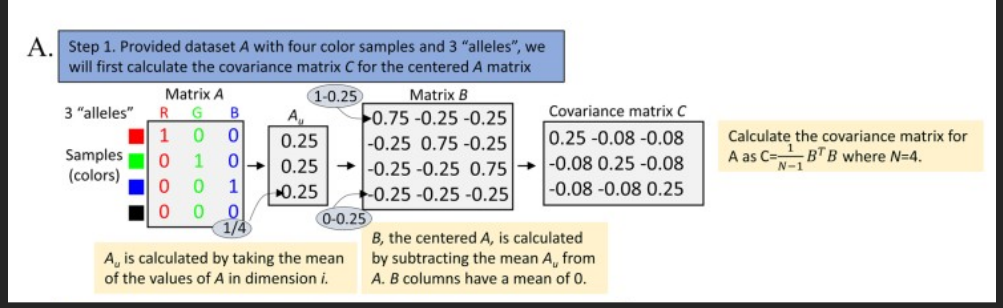
The black box



Elhaik, E. Principal Component Analyses (PCA)-based findings in population genetic studies are highly biased and must be reevaluated. Sci Rep 12, 14683 (2022). https://doi.org/10.1038/s41598-022-14395-4

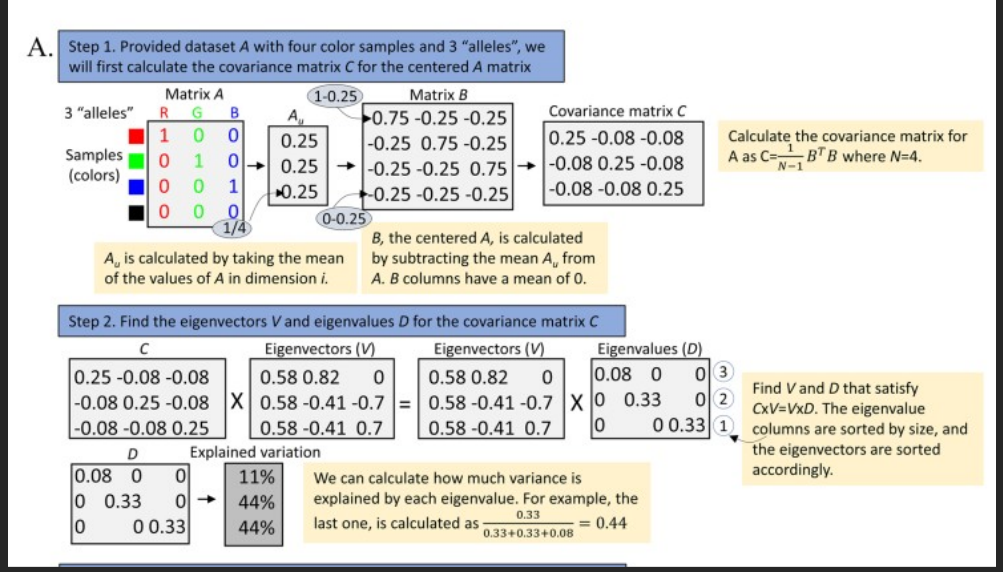


Step 1: **centre** the matrix
Step 2: measure the variance covariance of the centred matrix



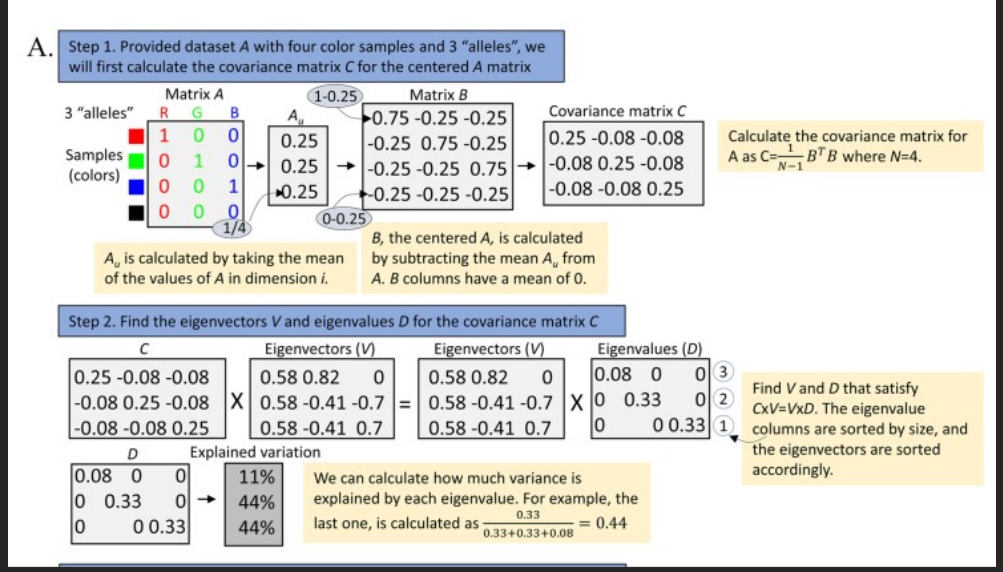
Keep this in a corner of your head, we'll get back to this!

Step 1: **centre** the matrix
Step 2: measure the variance covariance of the centred matrix



Step 3: do a eigen decomposition.
Basically satisfy the equation:

VCV matrix * eigenvector = eigenvector * eigenvalue.



Step 3: do a eigen decomposition.
Basically satisfy the equation:

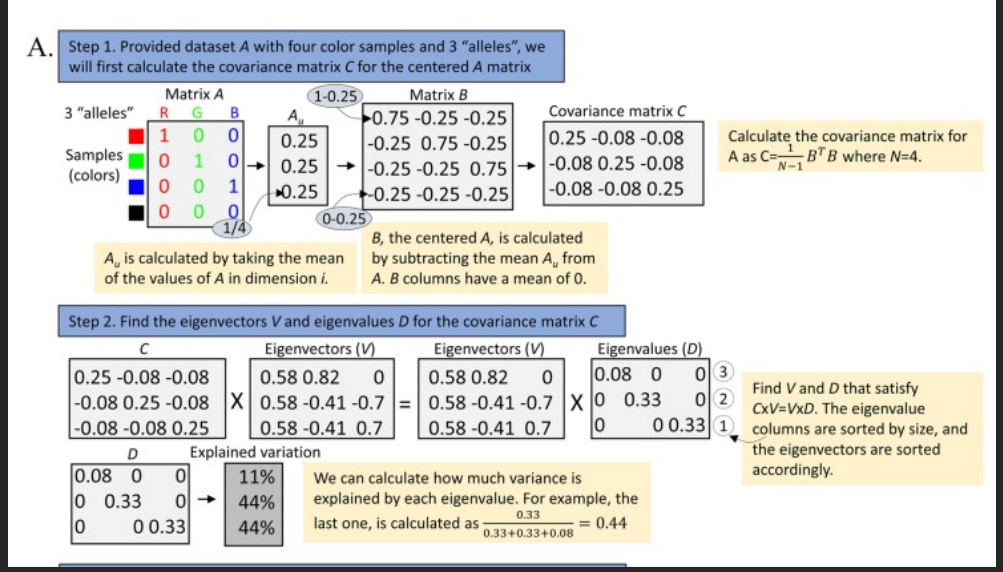
VCV matrix * eigenvector = eigenvector * eigenvalue.

The variance-covariance between traits

The vector (direction) of change

The vector (direction) of change

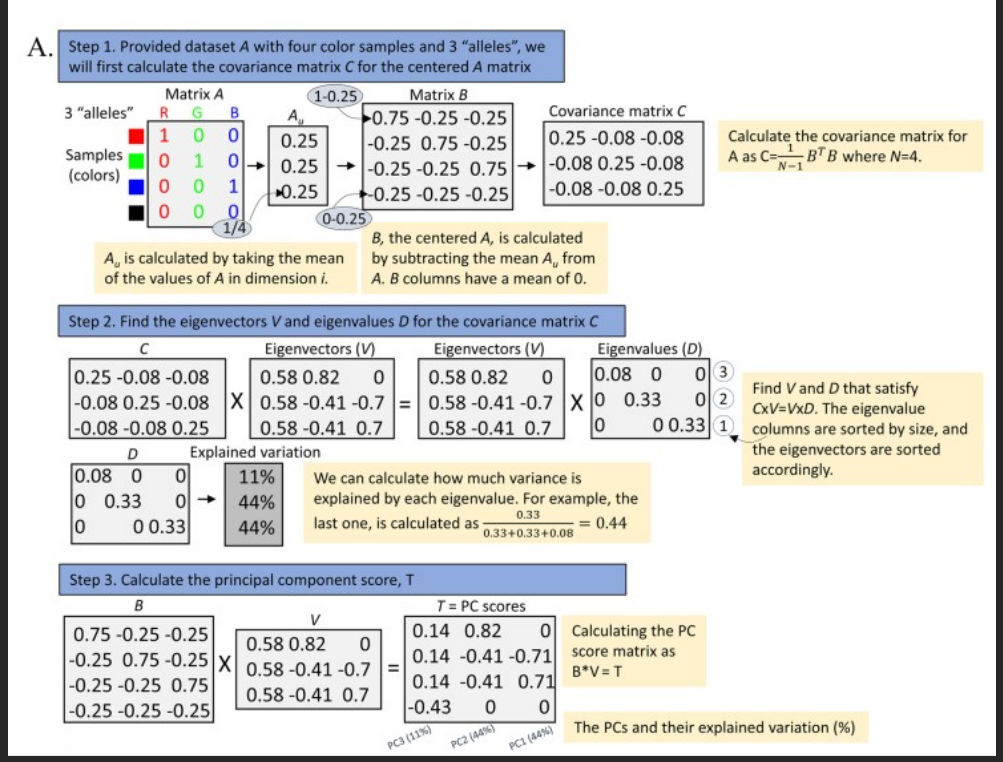
The strength (length) of change



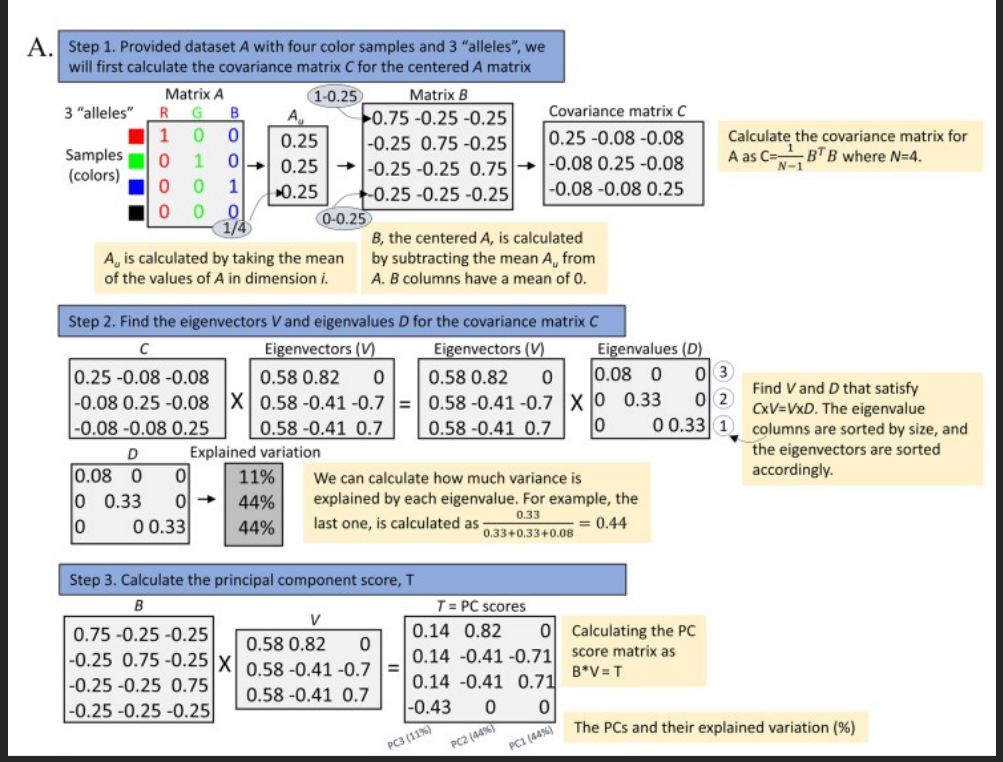
Step 3: do a eigen decomposition.
Basically satisfy the equation:

VCV matrix * eigenvector = eigenvector * eigenvalue.

This method is the core of the PCA. I think it's OK to treat it as a black box since it varies between algorithms. R default's is LAPACK but EISPACK or other algorithms can also be used. Also, these algorithms are explicitely approximations :"All you can hope for is a solution to a problem suitably close to x." (`base::eigen`). This can explain differences between ordinations of the same data.



Step 4: multiply the centred matrix by the eigenvector



Step 4: multiply the centred matrix by the eigenvector
Step 5: that's it.

A.

Step 1. Provided dataset *A* with four color samples and 3 "alleles", we will first calculate the covariance matrix *C* for the centered *A* matrix

Matrix *A*

3 "alleles"	R	G	B
Samples (colors)	1	0	0
	0	1	0
	0	0	1
	0	0	0

1/4

A_u

0.25
0.25
0.25

1-0.25

Matrix *B*

0.75 -0.25 -0.25
-0.25 0.75 -0.25
-0.25 -0.25 0.75
-0.25 -0.25 -0.25

0-0.25

Covariance matrix *C*

0.25 -0.08 -0.08
-0.08 0.25 -0.08
-0.08 -0.08 0.25

Calculate the covariance matrix for A as $C=\frac{1}{N-1}B^T B$ where *N*=4.

A_u is calculated by taking the mean of the values of *A* in dimension *i*.

B, the centered *A*, is calculated by subtracting the mean A_u from *A*. *B* columns have a mean of 0.

Step 2. Find the eigenvectors *V* and eigenvalues *D* for the covariance matrix *C*

C

0.25 -0.08 -0.08
-0.08 0.25 -0.08
-0.08 -0.08 0.25

X

Eigenvectors (*V*)

0.58 0.82 0
0.58 -0.41 -0.7
0.58 -0.41 0.7

=

Eigenvectors (*V*)

0.58 0.82 0
0.58 -0.41 -0.7
0.58 -0.41 0.7

X

Eigenvalues (*D*)

0.08 0 0 (3)
0 0.33 0 (2)
0 0 0.33 (1)

Find *V* and *D* that satisfy *CxV=VxD*. The eigenvalue columns are sorted by size, and the eigenvectors are sorted accordingly.

D

0.08 0 0
0 0.33 0
0 0 0.33

Explained variation

11%
44%
44%

We can calculate how much variance is explained by each eigenvalue. For example, the last one, is calculated as $\frac{0.33}{0.33+0.33+0.08} = 0.44$

Step 3. Calculate the principal component score, T

B

0.75 -0.25 -0.25
-0.25 0.75 -0.25
-0.25 -0.25 0.75
-0.25 -0.25 -0.25

X

V

0.58 0.82 0
0.58 -0.41 -0.7
0.58 -0.41 0.7

=

T = PC scores

PC3 (11%)	PC2 (44%)	PC1 (44%)
0.14	0.82	0
0.14	-0.41	-0.71
0.14	-0.41	0.71
-0.43	0	0

Calculating the PC score matrix as B*V = T

The PCs and their explained variation (%)

Step 4. Validate that matrix *A* can be re-calculated

T

PC1	PC2	PC3
0	0.82	0.14
-0.71	-0.41	0.14
0.71	-0.41	0.14
0	0	-0.43

X

V^T

0 0.71 -0.71
0.82 -0.41 -0.41
0.58 0.58 0.58

+

A_u

0.25
0.25
0.25

=

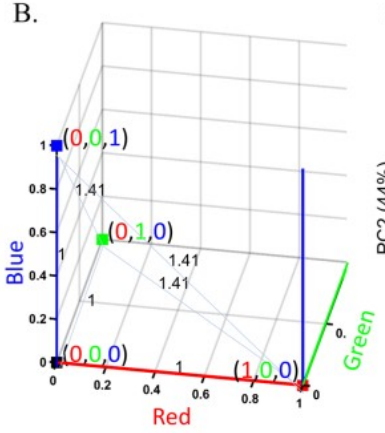
Matrix *A*

1 0 0
0 1 0
0 0 1
0 0 0

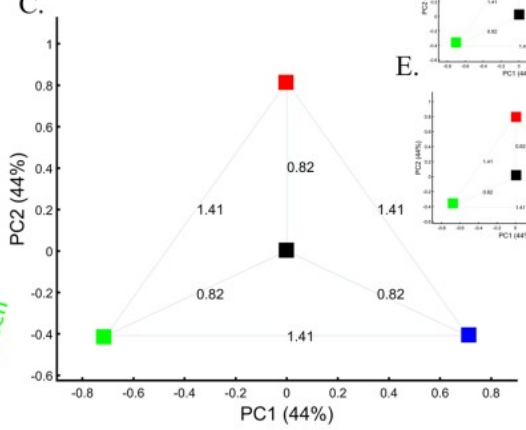
Matrix *T* columns represent the PCs. Here, the equation $TxV^T+A_u=A$ *is used* to test the calculation.

Step 5. Plot the top two PC scores as in C) below

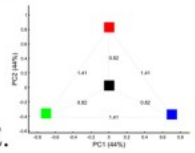
B.



C.



D.



E.

PCA: what it’s good at?

- *Ordinating* your data (i.e. ranking all your variables and making them independent and orthogonal).

PCA: what it’s good at?

- *Ordinating* your data (i.e. ranking all your variables and making them independent and orthogonal).

- Reducing dimensionality (to some extend – e.g. going from 200D to 10D).

PCA: what it’s good at?

- *Ordinating* your data (i.e. ranking all your variables and making them independent and orthogonal).

- Reducing dimensionality (to some extend – e.g. going from 200D to 10D).

- Creating a “true” mathematical space (that contains all the possible trait combinations).

PCA: what it’s bad at (in my opinion)?

- Being interpreted by humans in 2D.

PCA: what it's bad at (in my opinion)?

- Being interpreted by humans in 2D.

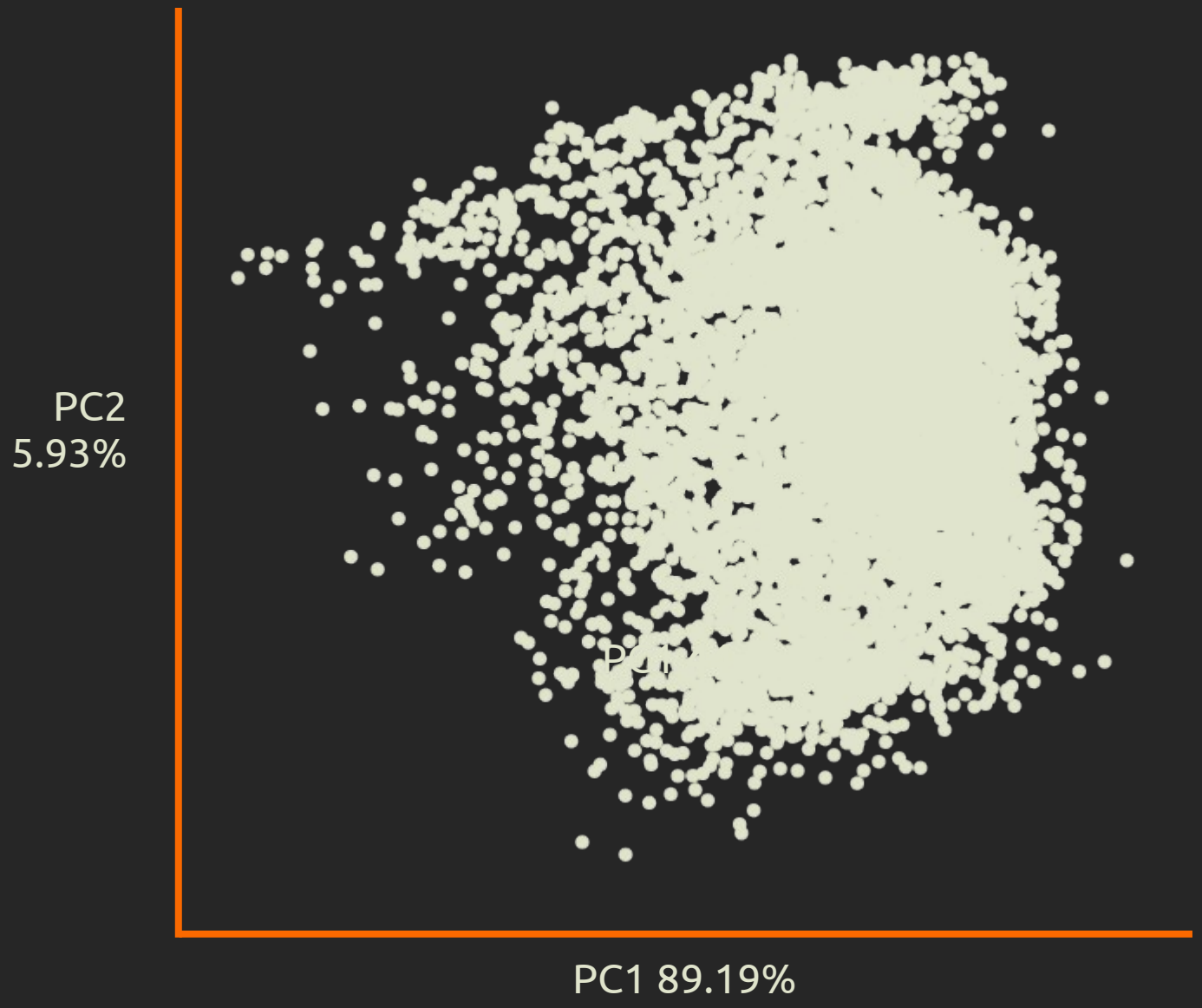
- Spreading data (the centre of your space is the average data value, not anything biological).

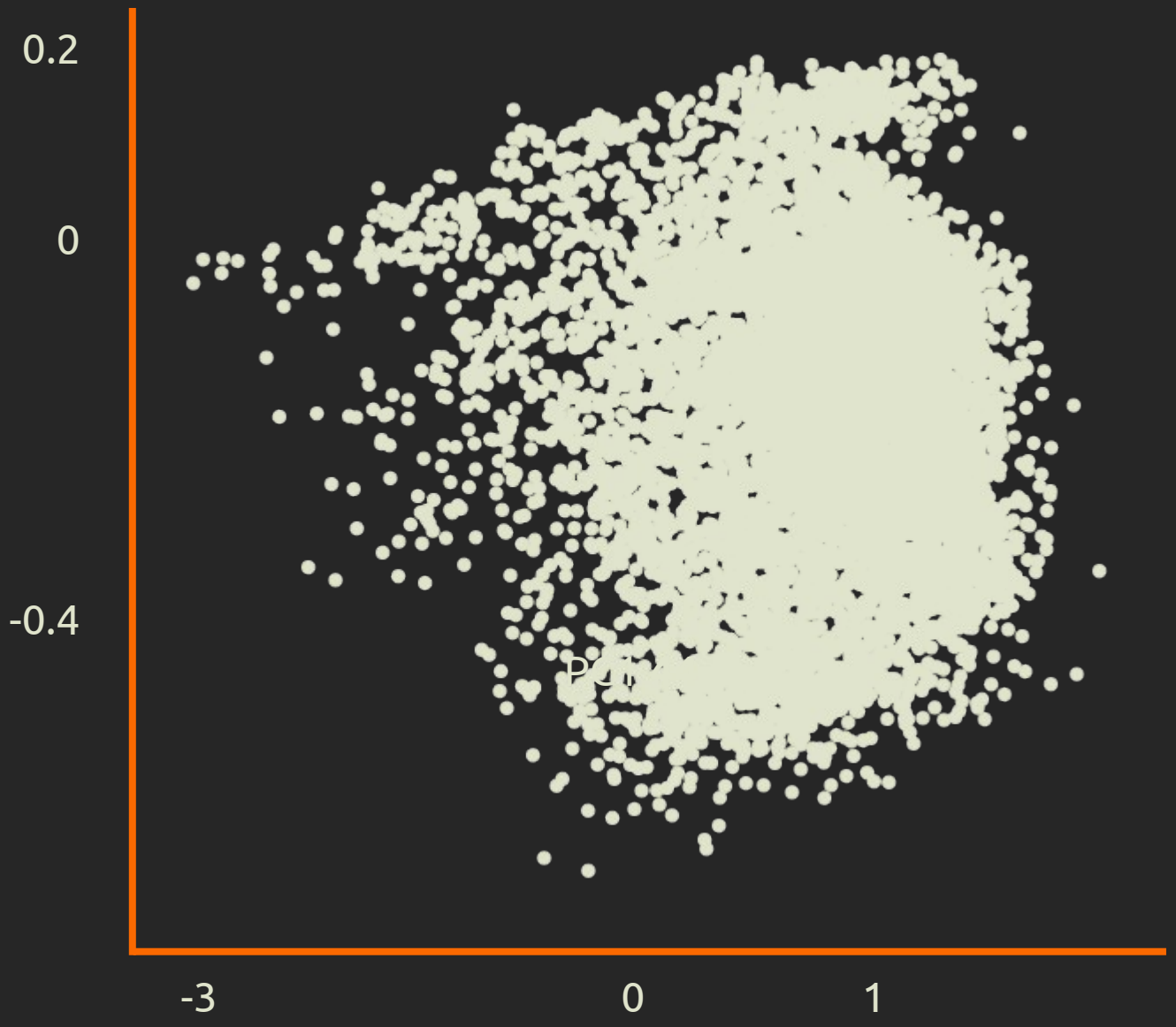
PCA: what it's bad at (in my opinion)?

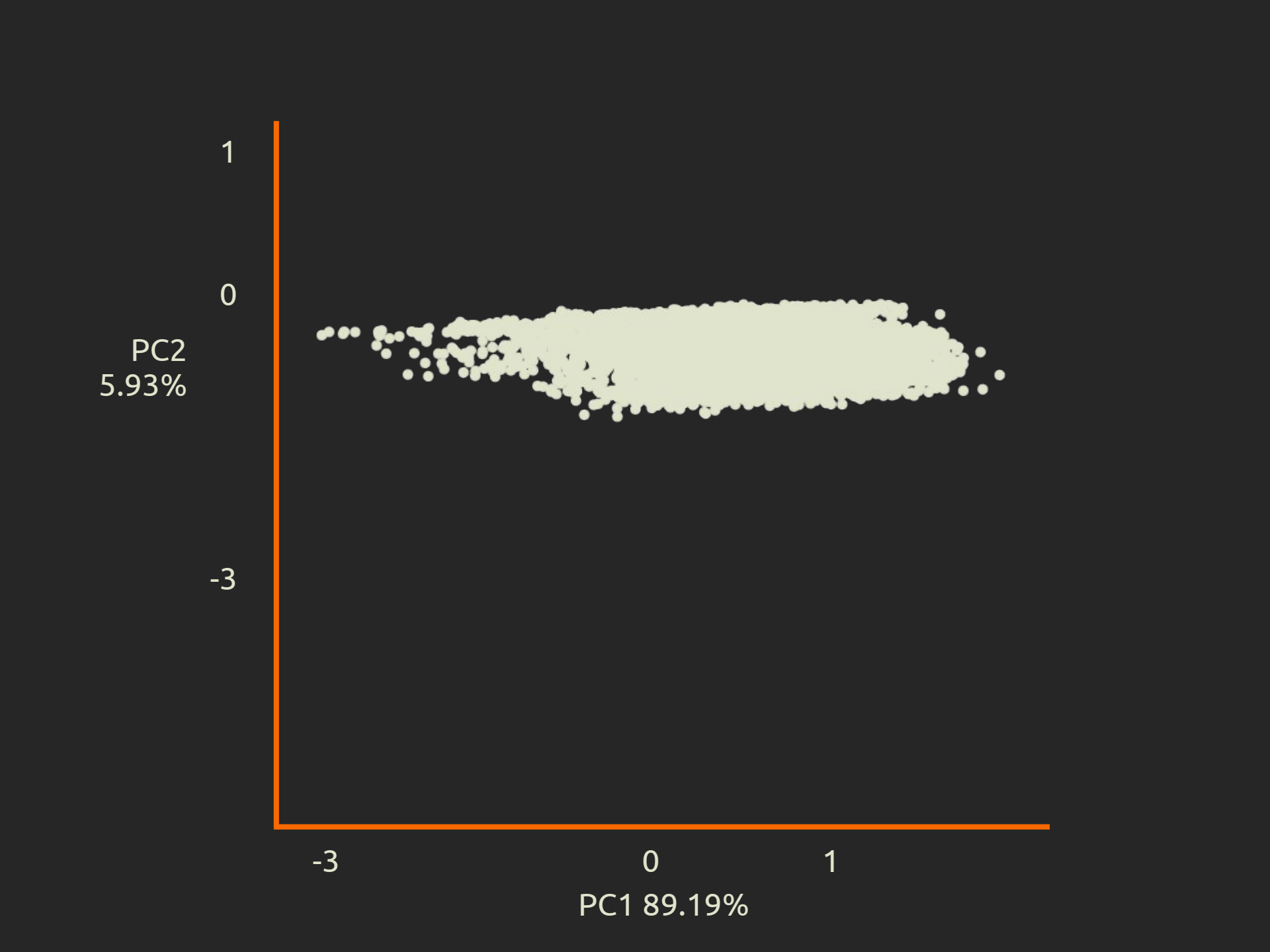
- Being interpreted by humans in 2D.

- Spreading data (the centre of your space is the average data value, not anything biological).

- Creating dimensions that are easy to interpret (e.g. PC1 = correlation between n-variables decided by the algorithm – these can sometimes map to biological things, sometimes not!).


PC2
5.93%
PC1 89.19%


0.2
0
-0.4
-3
0
1


1
0
-3
PC2
5.93%
-3
0
1
PC1 89.19%