

The background of the slide is a dark blue field filled with various chemical structures and equations in a lighter blue color. These include organic molecules like benzene rings, carboxylic acids, and complex polymers, as well as inorganic structures like silicates and nitrates. Chemical equations such as  $2\text{KNO}_3 + \text{H}_2\text{CO}_3 \rightarrow \text{K}_2\text{CO}_3 + 2\text{HNO}_3$  and  $\text{SiO}_2 + 2\text{H}_2\text{O} \rightarrow \text{H}_4\text{SiO}_4$  are also visible. The text is centered over this background.

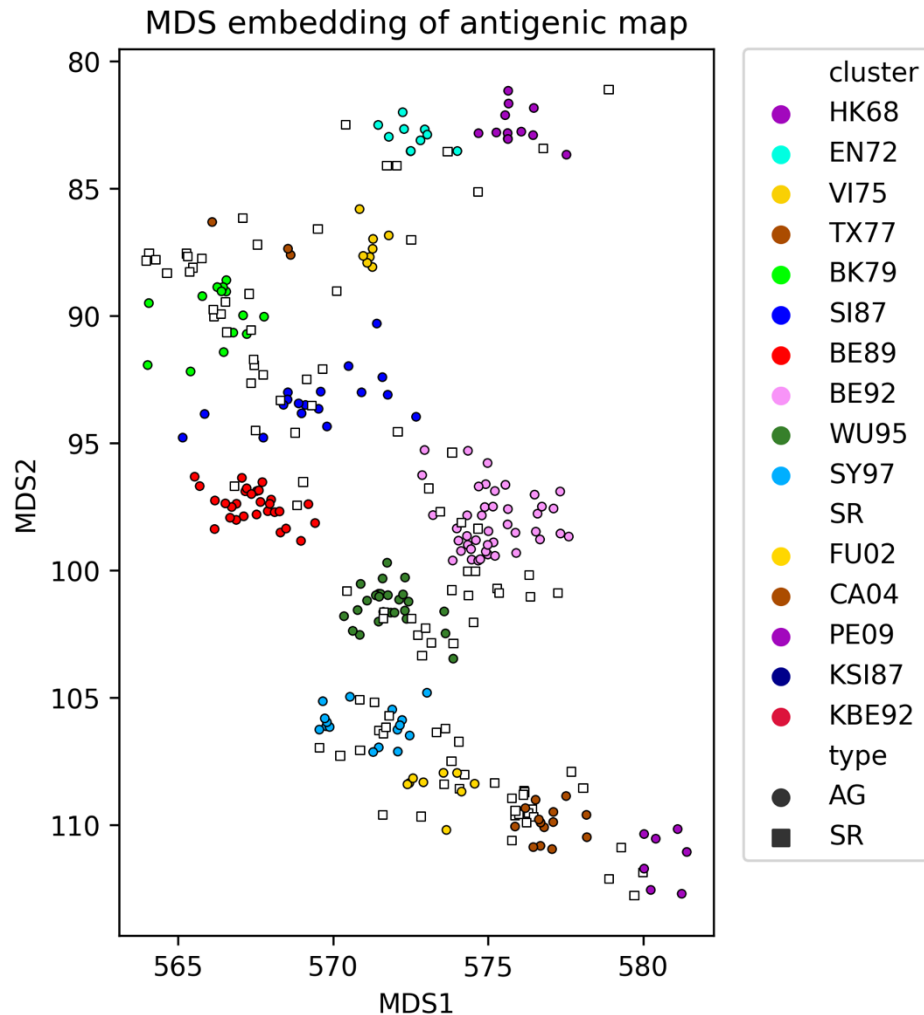
# Data Science for genomics protein language models and virus evolution

Francesco Durazzi

*DIFA, University of Bologna*

8 July 2025

# Today's case study: Influenza virus antigenic maps



Over the years, the evolving influenza virus has been tested against many antibodies, to describe the changes in immune response over time

E.g.: old vaccines were effective against old flu variants, but not against the most recent ones

Antigenic Maps: the experimental antibody response of each influenza variant is condensed into a 2D plot, where variants close together share a similar response to the same antibodies/vaccines

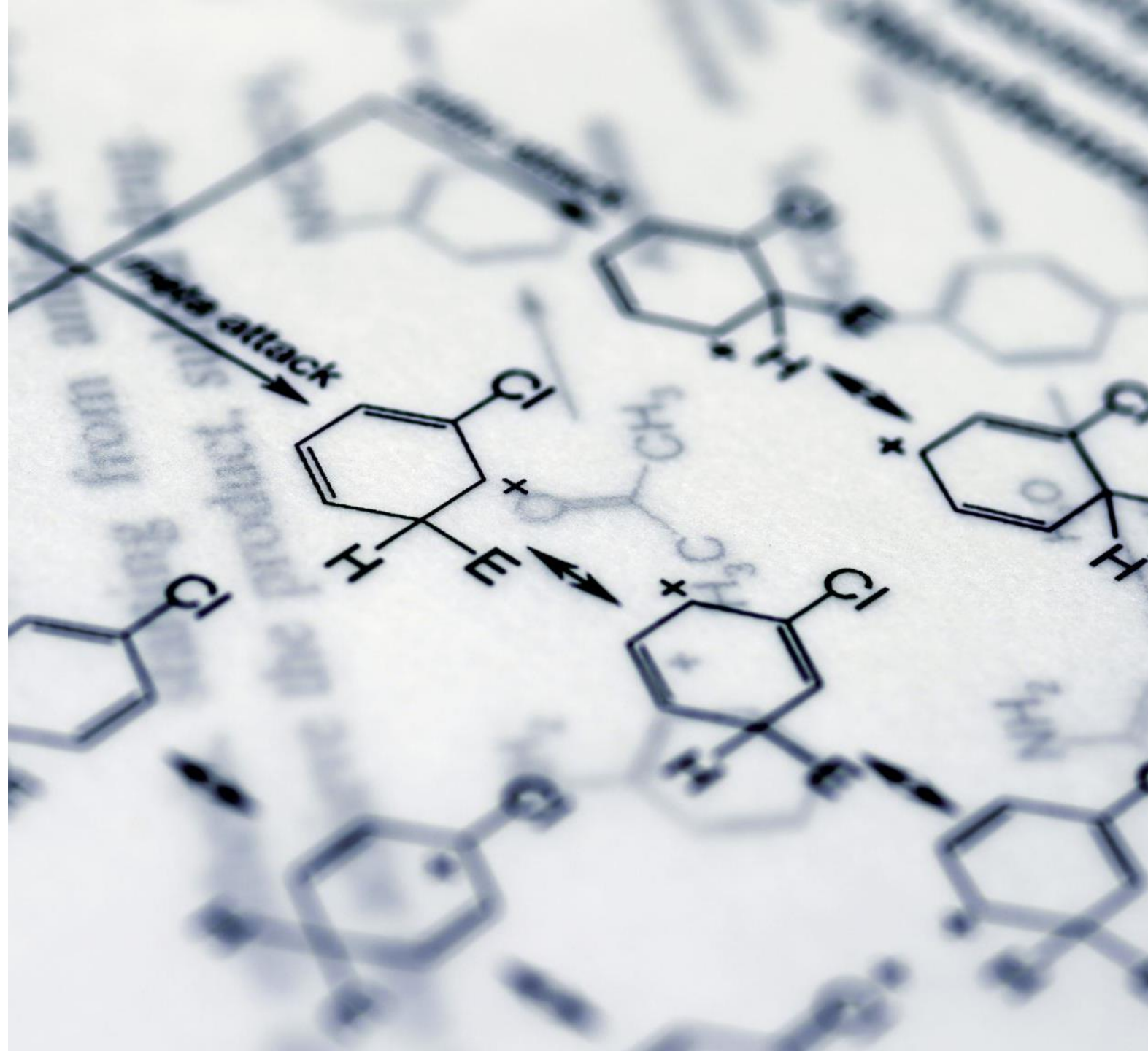
13 antigenic clusters emerged over time (from top to bottom in the figure)

# Today's case study: Influenza virus antigenic maps

- Suppose a new influenza strain emerges in the population...
- How do we know if the existing vaccines provide immunity for it? A lot of experiments would be required to test it...
- The challenge is:

**Can we predict the antigenic properties of the new strains from the protein sequence of the virus alone?**

N.B. Sequencing is way cheaper than antibody binding essays

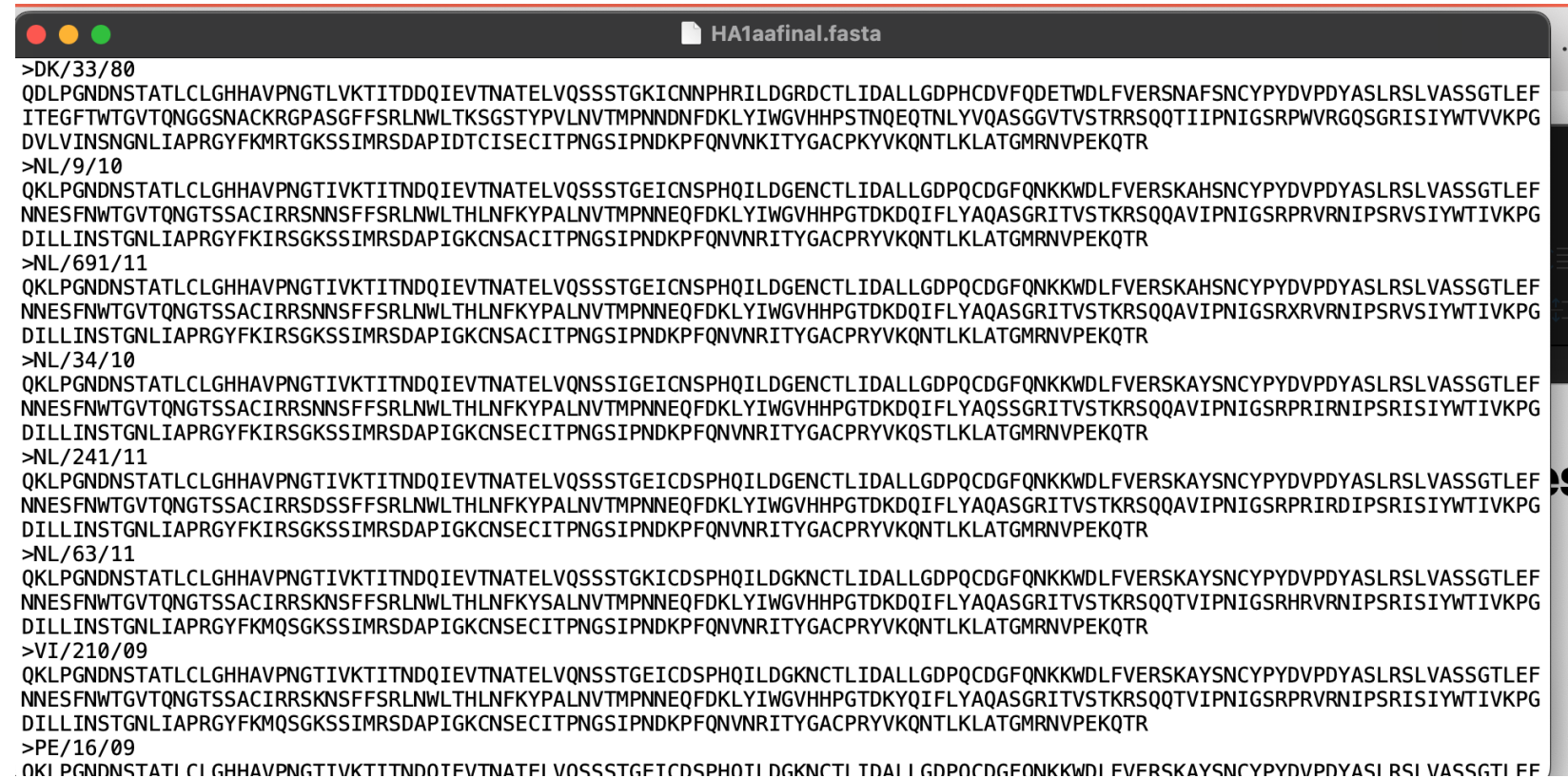


# Genomic data: fasta files

Sequencing outputs  
are strings of text

Each sequence may  
have a different length  
and is composed by a  
list of symbols

For proteins, there are  
20 symbols  
representing the  
possible amino acids

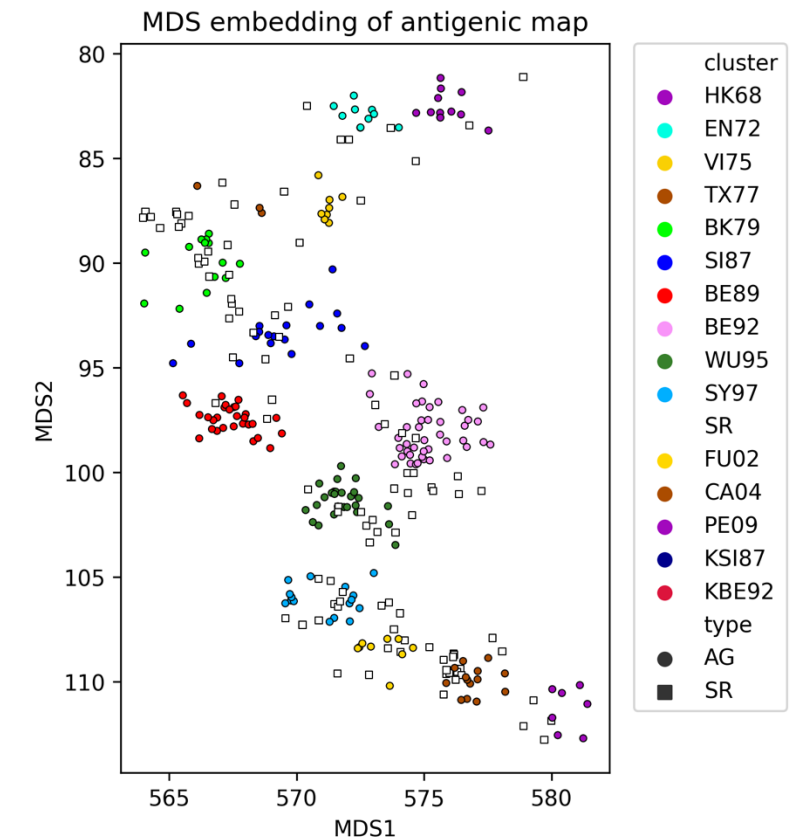


```
HA1aafinal.fasta
>DK/33/80
QDLPGNDNSTATLCLGHHAVPNGTLVKTITDDQIEVTNATELVQSSSTGKICNPNHRILDGRDCTLIDALLGDPHCDVFQDETWDLFVERSNAFSNCYPYDVPDYASLRSLVASSGTLEF
ITEGFTWTGVTQNGGSNACKRGPASGFFSRLNWLTKSGSTYPVLNVTMPNNDNFDKLYIWGVHHPSTNQEQTNLYVQASGGVTVSTRSQQTIIIPNIGSRPWVRGQSGRISYWTIVKPG
DVLVINSNGNLIAPRGYFKMRTGKSSIMRSDAPIDTCISECITPNGSIPNDKPFQNVNKITYGACPKYVKQNTLKLATGMRNVPEKQTR
>NL/9/10
QKLPGNDNSTATLCLGHHAVPNGTIVKTIITNDQIEVTNATELVQSSSTGEICNSPHQILDGENCTLIDALLGDPQCDGFQNKKWDLFVERSKAHSNCYPYDVPDYASLRSLVASSGTLEF
NNESFNWTGVTQNGTSSACIRRSNNSFFSRLNWLTHLNFKYPALNVTMPNNEQFDKLYIWGVHHPGTDKDQIFLYAQASGRITVSTKRSQQAVIPNIGSRPRVRNIPSRVSIYWTIVKPG
DILLINSTGNLIAPRGYFKIRSGKSSIMRSDAPIGKCNSECITPNGSIPNDKPFQNVNRITYGACPRYVKQNTLKLATGMRNVPEKQTR
>NL/691/11
QKLPGNDNSTATLCLGHHAVPNGTIVKTIITNDQIEVTNATELVQSSSTGEICNSPHQILDGENCTLIDALLGDPQCDGFQNKKWDLFVERSKAHSNCYPYDVPDYASLRSLVASSGTLEF
NNESFNWTGVTQNGTSSACIRRSNNSFFSRLNWLTHLNFKYPALNVTMPNNEQFDKLYIWGVHHPGTDKDQIFLYAQASGRITVSTKRSQQAVIPNIGSRPRVRNIPSRVSIYWTIVKPG
DILLINSTGNLIAPRGYFKIRSGKSSIMRSDAPIGKCNSECITPNGSIPNDKPFQNVNRITYGACPRYVKQNTLKLATGMRNVPEKQTR
>NL/34/10
QKLPGNDNSTATLCLGHHAVPNGTIVKTIITNDQIEVTNATELVQSSSTGEICNSPHQILDGENCTLIDALLGDPQCDGFQNKKWDLFVERSKAYSNCYPYDVPDYASLRSLVASSGTLEF
NNESFNWTGVTQNGTSSACIRRSNNSFFSRLNWLTHLNFKYPALNVTMPNNEQFDKLYIWGVHHPGTDKDQIFLYAQSSGRITVSTKRSQQAVIPNIGSRPRIRNIPSRISYWTIVKPG
DILLINSTGNLIAPRGYFKIRSGKSSIMRSDAPIGKCNSECITPNGSIPNDKPFQNVNRITYGACPRYVKQNTLKLATGMRNVPEKQTR
>NL/241/11
QKLPGNDNSTATLCLGHHAVPNGTIVKTIITNDQIEVTNATELVQSSSTGEICDSPHQILDGENCTLIDALLGDPQCDGFQNKKWDLFVERSKAYSNCYPYDVPDYASLRSLVASSGTLEF
NNESFNWTGVTQNGTSSACIRRSNNSFFSRLNWLTHLNFKYPALNVTMPNNEQFDKLYIWGVHHPGTDKDQIFLYAQASGRITVSTKRSQQAVIPNIGSRPRIRNIPSRISYWTIVKPG
DILLINSTGNLIAPRGYFKIRSGKSSIMRSDAPIGKCNSECITPNGSIPNDKPFQNVNRITYGACPRYVKQNTLKLATGMRNVPEKQTR
>NL/63/11
QKLPGNDNSTATLCLGHHAVPNGTIVKTIITNDQIEVTNATELVQSSSTGKICDSPHQILDGKNCTLIDALLGDPQCDGFQNKKWDLFVERSKAYSNCYPYDVPDYASLRSLVASSGTLEF
NNESFNWTGVTQNGTSSACIRRSKNSFFSRLNWLTHLNFKYPALNVTMPNNEQFDKLYIWGVHHPGTDKDQIFLYAQASGRITVSTKRSQQTVIPNIGSRPRVRNIPSRISYWTIVKPG
DILLINSTGNLIAPRGYFKMQSGKSSIMRSDAPIGKCNSECITPNGSIPNDKPFQNVNRITYGACPRYVKQNTLKLATGMRNVPEKQTR
>VI/210/09
QKLPGNDNSTATLCLGHHAVPNGTIVKTIITNDQIEVTNATELVQSSSTGEICDSPHQILDGKNCTLIDALLGDPQCDGFQNKKWDLFVERSKAYSNCYPYDVPDYASLRSLVASSGTLEF
NNESFNWTGVTQNGTSSACIRRSKNSFFSRLNWLTHLNFKYPALNVTMPNNEQFDKLYIWGVHHPGTDKYQIFLYAQASGRITVSTKRSQQTVIPNIGSRPRVRNIPSRISYWTIVKPG
DILLINSTGNLIAPRGYFKMQSGKSSIMRSDAPIGKCNSECITPNGSIPNDKPFQNVNRITYGACPRYVKQNTLKLATGMRNVPEKQTR
>PE/16/09
QKLPGNDNSTATLCLGHHAVPNGTIVKTIITNDQIEVTNATELVQSSSTGEICDSPHQILDGKNCTLIDALLGDPQCDGFQNKKWDLFVERSKAYSNCYPYDVPDYASLRSLVASSGTLEF
```



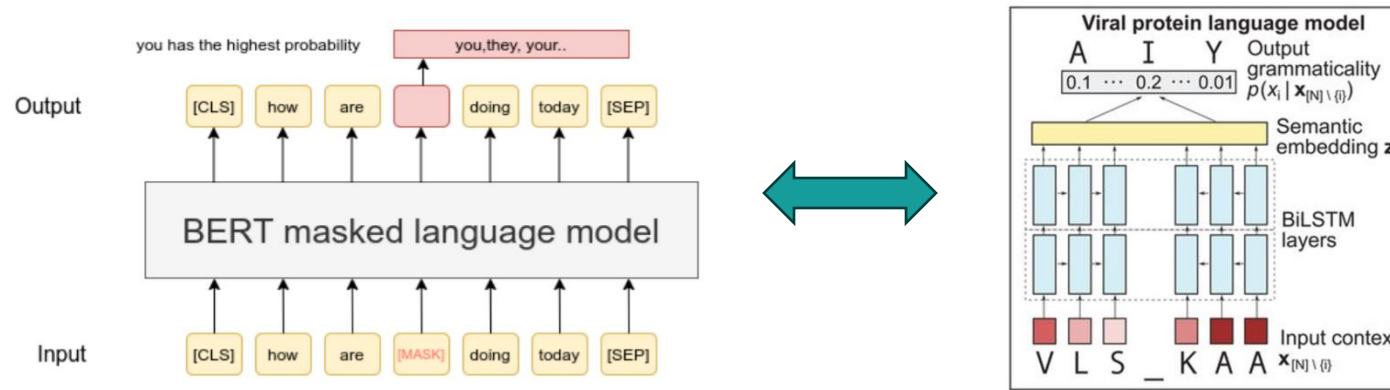
# Geno2pheno: mapping sequences into phenotype properties

```
HA1aafinal.fasta
>DK/33/80
QDLPGNDNSTATLCLGHAVPNGTIVKTIITNDQIEVTNATELVQSSSTGKICNNPHRILDGRDCTLIDALLGDPQCDVFDQETWDLFVERSNFASNCYPYDVPDYASLRSLVASSGTLEF
ITEGFTWTGVTQNGGSSACKRGSPASGFFSRLNLWTKSGSTYPVLNVTMPNNDNFDKLYIWGVHHPSTNQEQTNLYVQASGGVTSTRRSQTTIIPNIGSRPWVRGQSGRISYWTIVKPG
DVLVINSNGNLIAPRGYFKMRTGKSSIMRSDAPIDTCISECITPNGSIPNDKPFQNVNRITYGACPRYVKQNTLKLATGMRNVPEKQTR
>NL/9/10
QKLPNDNSTATLCLGHAVPNGTIVKTIITNDQIEVTNATELVQSSSTGEICNSPHQILDGENCTLIDALLGDPQCDGFGQNKKWDLFVERSKAHSNCYPYDVPDYASLRSLVASSGTLEF
NNESFNWGTGVTQNGTSSACIRRSNNSFFSRLNLWTLNFKYPALNVTMPNNEQFDKLYIWGVHHPGTDKQDIFLYAQASGRITVSTKRSQQAVIPNIGSRPRVRNIPSRVSIYWTIVKPG
DILLINSTGNLIAPRGYFKIRSGKSSIMRSDAPIGKCNSECITPNGSIPNDKPFQNVNRITYGACPRYVKQNTLKLATGMRNVPEKQTR
>NL/691/11
QKLPNDNSTATLCLGHAVPNGTIVKTIITNDQIEVTNATELVQSSSTGEICNSPHQILDGENCTLIDALLGDPQCDGFGQNKKWDLFVERSKAHSNCYPYDVPDYASLRSLVASSGTLEF
NNESFNWGTGVTQNGTSSACIRRSNNSFFSRLNLWTLNFKYPALNVTMPNNEQFDKLYIWGVHHPGTDKQDIFLYAQASGRITVSTKRSQQAVIPNIGSRPRVRNIPSRVSIYWTIVKPG
DILLINSTGNLIAPRGYFKIRSGKSSIMRSDAPIGKCNSECITPNGSIPNDKPFQNVNRITYGACPRYVKQNTLKLATGMRNVPEKQTR
>NL/34/10
QKLPNDNSTATLCLGHAVPNGTIVKTIITNDQIEVTNATELVQSSSTGEICNSPHQILDGENCTLIDALLGDPQCDGFGQNKKWDLFVERSKAYSNCYPYDVPDYASLRSLVASSGTLEF
NNESFNWGTGVTQNGTSSACIRRSNNSFFSRLNLWTLNFKYPALNVTMPNNEQFDKLYIWGVHHPGTDKQDIFLYAQSSGRITVSTKRSQQAVIPNIGSRPRVRNIPSRVSIYWTIVKPG
DILLINSTGNLIAPRGYFKIRSGKSSIMRSDAPIGKCNSECITPNGSIPNDKPFQNVNRITYGACPRYVKQNTLKLATGMRNVPEKQTR
>NL/241/11
QKLPNDNSTATLCLGHAVPNGTIVKTIITNDQIEVTNATELVQSSSTGEICNSPHQILDGENCTLIDALLGDPQCDGFGQNKKWDLFVERSKAYSNCYPYDVPDYASLRSLVASSGTLEF
NNESFNWGTGVTQNGTSSACIRRSNNSFFSRLNLWTLNFKYPALNVTMPNNEQFDKLYIWGVHHPGTDKQDIFLYAQASGRITVSTKRSQQAVIPNIGSRPRVRNIPSRVSIYWTIVKPG
DILLINSTGNLIAPRGYFKIRSGKSSIMRSDAPIGKCNSECITPNGSIPNDKPFQNVNRITYGACPRYVKQNTLKLATGMRNVPEKQTR
>NL/63/11
QKLPNDNSTATLCLGHAVPNGTIVKTIITNDQIEVTNATELVQSSSTGEICNSPHQILDGENCTLIDALLGDPQCDGFGQNKKWDLFVERSKAYSNCYPYDVPDYASLRSLVASSGTLEF
NNESFNWGTGVTQNGTSSACIRRSNNSFFSRLNLWTLNFKYSALNVTMPNNEQFDKLYIWGVHHPGTDKQDIFLYAQASGRITVSTKRSQQAVIPNIGSRPRVRNIPSRVSIYWTIVKPG
DILLINSTGNLIAPRGYFKMQSGKSSIMRSDAPIGKCNSECITPNGSIPNDKPFQNVNRITYGACPRYVKQNTLKLATGMRNVPEKQTR
>VI/210/09
QKLPNDNSTATLCLGHAVPNGTIVKTIITNDQIEVTNATELVQSSSTGEICNSPHQILDGENCTLIDALLGDPQCDGFGQNKKWDLFVERSKAYSNCYPYDVPDYASLRSLVASSGTLEF
NNESFNWGTGVTQNGTSSACIRRSNNSFFSRLNLWTLNFKYPALNVTMPNNEQFDKLYIWGVHHPGTDKQDIFLYAQASGRITVSTKRSQQAVIPNIGSRPRVRNIPSRVSIYWTIVKPG
DILLINSTGNLIAPRGYFKMQSGKSSIMRSDAPIGKCNSECITPNGSIPNDKPFQNVNRITYGACPRYVKQNTLKLATGMRNVPEKQTR
>PE/16/09
OKI PGNDNSTATI CL GHAVPNGTIVKTIITNDQIEVTNATELVQSSSTGETCNSPHQILDGENCTLIDALLGDPQCDGFGQNKKWDLFVERSKAYSNCYPYDVPDYASLRSLVASSGTLEF
```



# Step 1

## embedding with protein Language Model



Hie et al., *Learning the language of viral evolution and escape*, 2021

LM process:

- 1 – Each of the 20 aminoacids is assigned with a numerical vector (*word embedding*), learned during training together with the LM weights.
- 2 – Each word embeddings are refined with the *attention mechanism* which looks at surrounding aminoacids: context dependent embeddings (e.g. “*We have the right to vote.*”  $\neq$  “*That’s right!*”)
- 3 – Refined word embeddings are processed and aggregated by several neural network layers up to the final level: prediction of the aminoacid *masked out*.

In this example, we will use ProtBERT, trained on 216M proteins from all the tree of life. Many LMs (for natural language, but also proteins) are hosted online in the HuggingFace repository.

# Code for today



## Jupyter Notebook on Google Colab (Python)

Google Colab offers a cloud computational server to execute Python code. You only need a Gmail account to access it.

N.B. Depending on availability, GPUs are often available, which increase by far the speed of running deep learning models (for today's task: 20s vs 25min).

You can modify as you want the notebook, but remember to save it on your Google Drive or download it at the end of the lecture. Otherwise you will lose your access!

<https://colab.research.google.com/drive/18snicSSIkWX1Z9UjOXrebHIQTppvdFIQ?usp=sharing>

# Step 2

## dimensionality reduction: PCA

### Principal Component Analysis

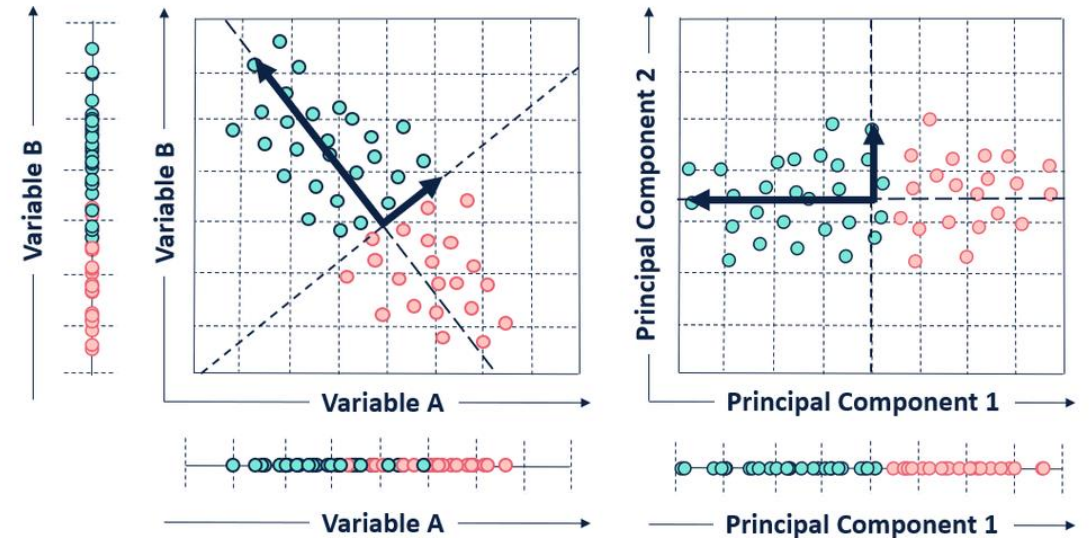
PCA rotates/projects your axis on a new reference system maximizing the variance of your data. Removes redundancies and correlations between the features.

Steps:

- 1- Create covariance matrix (correlation between features).
- 2- Diagonalize cov matrix. It means finding the eigenvectors: axes of maximal variance of the data: more variance  $\propto$  more information.
- 3- Project data into the eigenvectors.
- 4- (optional) Project into less eigenvectors  $\rightarrow$  lose some information, but reduce dimensionality.

**Covariance Matrix**

$$\begin{bmatrix} \text{Var}(x_1) & \dots & \text{Cov}(x_n, x_1) \\ \vdots & \ddots & \vdots \\ \text{Cov}(x_n, x_1) & \dots & \text{Var}(x_n) \end{bmatrix}$$





# Step 3

## linear (Ridge) regression

Supervised method to predict a target (y) from a data features (X)

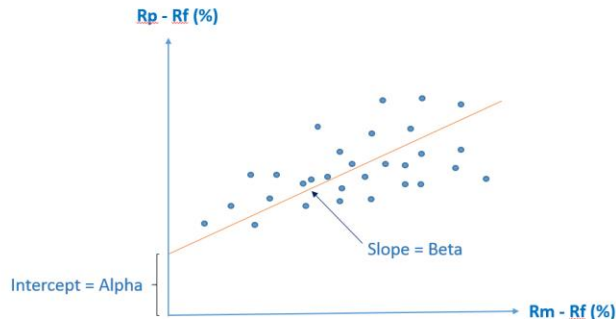
y = 2D antigenic map coordinates  
X = 1024D embedding

### Linear regression

$$y = X\beta + \epsilon = \beta_0 + \beta_1 X_1 + \dots + \beta_N X_N$$

Find  $\beta$  that minimizes the error

$$\beta = \operatorname{argmin}((y - X\beta)^2)$$



# Step 3

## linear (Ridge) regression

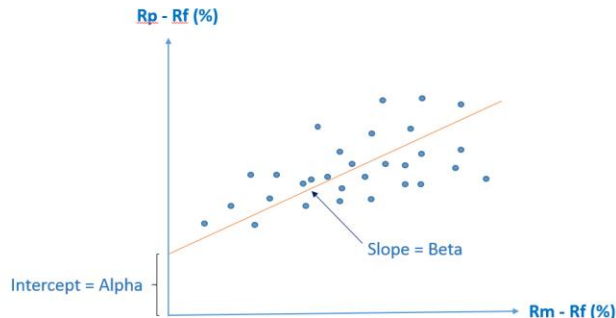
Supervised method to predict a target (y) from a data features (X)

y = 2D antigenic map coordinates  
X = 1024D embedding

### Linear regression

$$y = X\beta + \epsilon = \beta_0 + \beta_1 X_1 + \dots + \beta_N X_N$$

Find  $\beta$  that minimizes the error  
 $\beta = \operatorname{argmin}((y - X\beta)^2)$



### Ridge regression (penalized)

$$y = X\beta + \epsilon = \beta_0 + \beta_1 X_1 + \dots + \beta_N X_N$$

Same as linear model, but with different minimization objective:

Find  $\beta$  that minimizes the error **and the coefficients value**

$$\beta = \operatorname{argmin}((y - X\beta)^2 + \lambda\beta^T\beta)$$

By constraining  $\beta$  to be near zero, you try to use less features, selecting only the most important ones and reducing risk of overfitting (N.B. you can fit perfectly N points with a polynomial of N-1 degrees, but probably it won't generalize well)