Universidad Zaragoza

1542

Master Thesis

Application of a statistical inference model to the prediction of antibody affinity from sequence analysis

Author:

David Luna Cerralbo

Supervised by:

Dr. Pierpaolo Bruscolini

Dr. Sergio Pérez Gaviro

FACULTAD DE CIENCIAS

2021

Contents

1 Introduction

In the last years, the general interest in therapeutic antibodies has grown quickly due to their huge potential application in a large number of diseases. The technologies have grown to higher and higher level of sophistication, starting from rough mouse antibodies, to chimeras, to humanized antibodies, and finally to fully human antibodies, produced by phage display or directly by humanized animals (i.e., mice, rabbits, chicken, lamas, ... that have undergone genetic engineering to produce human antibodies). However, new antibodies are still difficult to design, they face stability problems, have a low probability to survive the pre-clinical stages, and also the successful ones, often show long-term failures, prompting for immunogenicity in the patient, i.e., an immune response neutralizing the action of the antibody drug.

Even if progress in the fields are fast, a rational design of good antibodies is still out of reach, for several reasons: we cannot design, with the computer, the structure of an antibody with the necessary accuracy to reach the required affinity to the antigen; we cannot predict precisely its equilibrium properties, and on the other hand, we do not know with sufficient precision the mechanisms of antibody maturation and selection, that transform a "naive" antibody, generated by the random combination of copies of the V, (D), and J genes, into a specialized tool for recognizing and binding a particular antigen, without causing damages to the "self".

In recent years, thanks to the fast growth of the available databases of antibody sequences [12], together with the availability of tools to manipulate [10] and align them [7], some different, sequence based, approaches have been developed, addressing different relevant issues as, for instance, the generation of the naive sequences starting from different copies of the genes [3], the characterization of the statistical distribution of the human sequences, and the definition of a humanization score based on such distribution [4] (in particular, the Multivariate Gaussian Model [6]), or the use of the same approach to predict which sequences will have a high affinity for the antigen [1].

In particular, in the latter article, the authors distinguish two clusters in the patient's antibody repertoire, which they interpret as the germline and hypermutated cluster, and learn a Multivariate Gaussian Model from the hypermutated cluster, to see to which extent the statistical score can be used as a predictor of the affinity of the antibody for its target antigen. Moreover, they use a simple "regularizer" for the covariance matrix, that is equivalent to consider a prior distribution with uniform probability of finding any residue at any position. However, in their Figure 7, three high density zones are observed, not just two. This could indicate that, despite the most robust solution for the clustering algorithm was two clusters, there could be clearly identifiable sub-clusters. To confirm this, we identified the sequences that make up each of the high-density zones and how they are divided into the germline and hypermutated clusters found in [1].

In this work, we address some different, but related questions: is it possible to describe the evolution, in the sequence space, that leads the sequences from the germline to the hipermutated cluster? Does this evolution provides a better prior when inferring the distribution of the hypermutated sequence? Is the hypermutated cluster an ensemble of fully optimized sequences, or rather

the antibody maturation has stopped after reaching a collection of sequences that have evolved enough to grant sufficient affinity, but that are suboptimal with respect to a final attractor, of even greater affinity? In particular, in agreement with the multivariate gaussian nature of our model, we describe the evolution in the sequence space as a Ornstein-Uhlenbeck (OU) process, that is, a stochastic Markov process, where the sequences undergo a drift towards an attractor of high affinity, while they also diffuse randomly in sequence space. We do so for the sake of the simplicity of the process, being aware that the actual mechanism of antibody maturation involves different agents at different scales, so that a thorough description on fundamental grounds is ruled out.

In this work, the antibody maturation process is described as a OU process.

Introducing some approximation and simplifying hypotheses, we are able to perform a bayesian inference of the attractor and final variance, using a prior that accounts for the distribution of the germline ensemble and on the (much bigger) statistics of possible naive antibodies. In the following, it is discussed how the OU model is implemented, as well as the whole pre-treatment of the experimental data, including alignment and codification of the sequences in order to make possible a gaussian description, identification of the different sub-clusters of sequences that have survived the maturation and pruning process, the choice of a restricted number of positions along the sequences, to ease the computational burden. The rest of this thesis is organized as follows: in the rest of the introduction, we briefly describe some general and structural properties of the antibodies, and their generation and maturation process. General features of the OU process will also mentioned, and the specific goals and scope of the work will be stated. Then, in the Methods section, the whole workflow that we have implemented to pursue our goals will be described. The following section will be presented the results that justify our way of approaching the description of the evolution, and those related to the efficacy of the method. Finally, some conclusions will be drew some conclusions, with special emphasis on the possible further developments of this approach.

1.1 Antibody structure and function.

Antibodies are a globular plasma proteins that are synthesized in the B-cells. Their function is to neutralize what the body does not recognize as its own such as viruses or bacteria, and also elicit the response of the immune system.. As it can be seen in Figure 1, in an antibody exits two regions with two main functions: the antigen binding fragment (Fab), which recognizes the antigen, and the crystallizable fragment (Fc), which interacts with other elements of the immune system. Structurally, antibodies consist of four polypeptide chains joined by disulfure bonds: two light chains, which can be also divided into two type (the κ and the λ light chains), and two heavy chains.

The Fab region is is composed of one constant and one variable domain from each heavy and light chain of the antibody. The variable region is in charge of the antigen recognition function and can be also divided into four Framework Regions (FRs) and three Complementary Determining Regions (CDRs). The FRs have a structural function and are very conserved, whereas the CDRs

are also known as hypermutated regions due to their high variability.

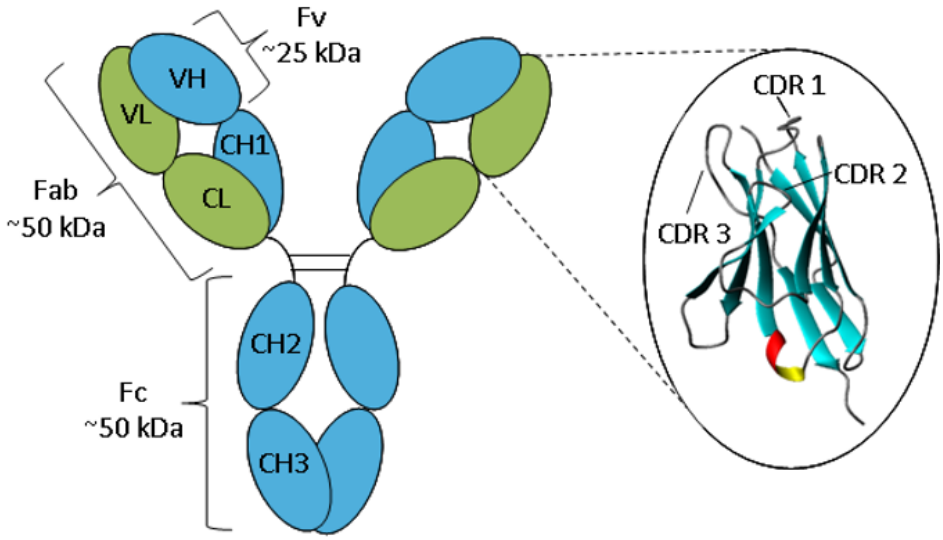


Figure 1: Antibody structure.

Such variability is justified by the fact that the CDRs are regions especially devoted to antigen recognition and binding, with great affinity and specificity. Due to the enormous variety of possible chemical structures of the antigen, a "brute force" strategy, where the correct sequences for all necessary antibodies is "hardcoded" at the gene level, is impossible, since it would imply the use of an enormous quantity of genes. As explained in the next subsection, the strategy actually implemented includes the introduction of stochastic recombinations and mutations at different stages, as well as a selection based on a "fitness" of the antibody, that includes its affinity for the antigen and its safety for the organism itself. The complexity of such process is what makes rationale in-silico design of antibodies so difficult: it requires extremely precise force-fields to be able to reproduce correctly the affinity of a sequence to the antigen, especially in the loops of CDR3. This is also the reason why the most common methods of antibody development are intrinsically biological, letting phages or animals to develop high affinity antibodies, and then trying to "humanize" them if required, preserving the activity of the CDRs.

1.2 Affinity maturation

Affinity maturation is a process by which B cells produce antibodies of increasing affinity for an antigen during the course of an immune response. With repeated exposures to the same antigen, the host will produce antibodies of successively higher affinities. This affinity maturation is composed by two different processes: the somatic recombinations and the clonal selection.

A somatic recombination is an alteration of the somatic cell DNA that is inherited by its daughter cells. Specifically, the V(D)J recombination is performed in the early stages of T and B cell maturation that involves several copies of the V, D and J genes, from which only one copy of each gene is chosen. Thanks to this process the immune system is able to generate an almost unlimited number of different light and heavy chains in a remarkably economical way, by joining separate gene segments together before they are transcribed. So a huge repertoire of antibodies with unique variable regions is available.

This V(D)J recombination is different for light and heavy chains as it can be seen in Figure 2. For light chains variable V and a joining J gene segment in the genomic DNA are joined to form a complete light-chain V-region exon, whereas the heavy-chain V regions are constructed from three gene segments. First, the diversity D and J gene segments join, afterwards the V gene segment joins to the combined DJ sequence, forming a complete V_H exon [2]. Also, notice that the in the light chain case the constant region C is encoded in only one exon meanwhile in the heavy chain case it is encoded in several ones.

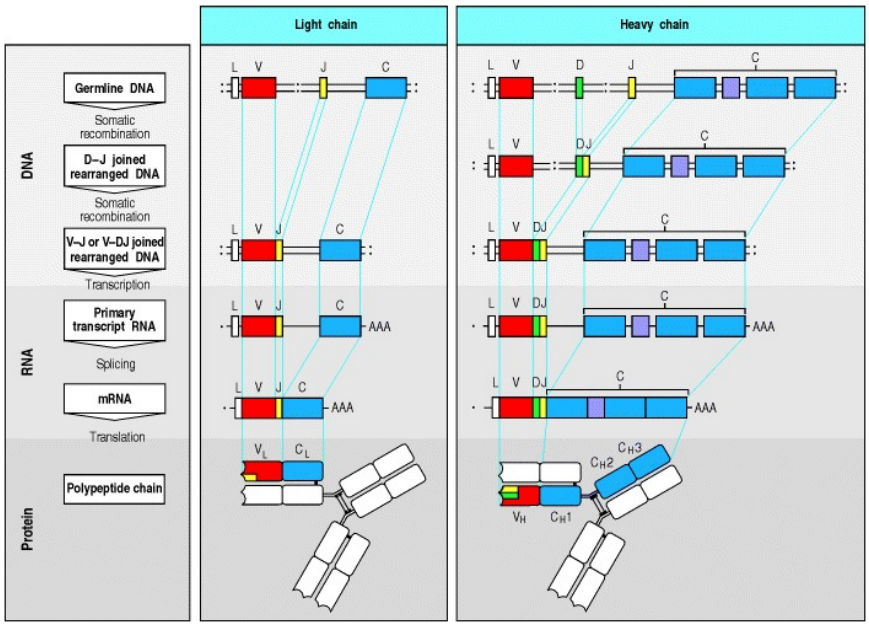


Figure 2: V(D)J recombination. Picture taken from [2].

The sequences generated by the V(D)J recombination are known as 'naive' sequences because they have not been yet in contact with a pathogen, inducing a process of mutation and selection (maturation). These naive sequences are interesting because its statistical distribution can be used as a priori distribution for the maturated sequences. To infer this naive distribution a large data base is required.

Luckily enough, such large data bases need not to be acquired experimentally, and can be build computationally, thanks to a recent software, IGoR [3]. This is a C++ software released in 2018

that is designed to infer V(D)J recombination related processes from sequencing data such as recombination model probability distribution or generation probabilities of sequences . It has three functionalities or modes:

- Learning mode: IGoR learns recombination statistics from data sequences.
- Analysis mode: IGoR outputs detailed recombination scenario statistics for each sequence.
- Generation mode: IGoR produces synthetic sequences with specified recombination statistics.

In the generation mode, V(D)J recombination selects two or three germline segments from the germline library, depending on whether it is being considered the variable domain or with the constant one. Then, IGoR assembles them while making some pair deletion or insertion at the junctions. So the creation of large naive sequences databases is possible. Interestingly, IGoR allows to deal with user-defined libraries, so that it is possible to use the gene libraries provided in [1].

On the other hand, the clonal selection theory postulates that each antigen will stimulate the lymphocyte or group of lymphocytes which have receptors on their membrane that are able to recognize and bind specifically to it, resulting in their proliferation and differentiation into cells with the same recognize characteristics as the original lymphocytes. According to this theory, these original lymphocytes exhibit a single type of antibody on their surface which has been generated by V(D)J recombination. In the first phase of maturation, lymphocytes with specific antibodies to self molecules are destroyed. All those that survive this first phase become inactive lymphocytes. From this kind of lymphocytes, those that never encounter a matching foreign antigen are discarded, but those that are activated produce many clones of themselves, completing the maturation process.

1.3 Hypothesis and goals

In this project, the aim is to improve the scoring function described in [1] by taking a more realistic a priori distribution and reinterpreting the germline and the hypermutated clusters as clusters linked by a temporal evolution, instead of as independent ones. This evolution will be modeled following an Ornstein-Uhlenbeck process, from which we extract the links between the two clusters and the attractors of the dynamic process. These attractors allow us to see whether the hypermutated cluster is the end of this maturation process or just an intermediate cluster with sufficient affinity to face the antigen. However, this OU process does not support a conjugate distribution whose analytical expression could be used to simplify the maths. For this reason, different sub-clusters of sequences will be identified, abandoning the two indivisible clusters scheme presented in [1]. Each of these sub-clusters will be assumed to have a unique evolution time and a Gaussian distribution. In this way, we have a system in which the OU process has been replaced by a set of temporary linked Gaussian distributions.

To identify the sub clusters, we do not try to divide directly the clusters found in [1], but we realign the sequences from KABAT scheme to the AHo one, and then a new clustering analysis of all the sequences is performed. Once the sub clusters are identified, they are compared with the two clusters considered in [1] to identify what is our new germline cluster. They are also compared with the 3 high density zones of Figure 7 of [1] (after translating the nucleotides to amino acids) to check their composition.

Once the new germline cluster has been identified, the multivariate Gaussian model described in [4] is inferred from it and used to check for compatibility between the germline cluster and the IGoR sequences, which will provide 'in silico' statistics to complement the experimental statistics provided by the germline cluster. This new germline cluster, together with an a priori distribution inferred from the IGoR sequences, will be the starting point of the OU process. A new challenge arises from this, because given the importance of this a priori distribution, a minimum number of IGoR sequences is needed to be able to find reliable results. This minimum size of the IGoR data set will be estimated by observing the evolution of different parameters, that will be defined later, depending on the size of the data set from which they are inferred.

2 Methods

2.1 Sequence alignment

The initial point is to consider a multiple sequence alignment (MSA). The goals of performing this multiple sequence alignment its to identify regions of similarity that may be a consequence of functional, structural, or evolutionary relationships between the sequences and also to unificate the length of all them in the whole data base. To achieve these goals, the so-called 'gaps' are introduced. These 'gaps' between amino acids are represented with '-', and the position in which they will appear in the sequences and the length of them depends on which alignment scheme is used. An example of the use of these 'gaps' can be seen in Figure 3. In this project the AHo scheme was used. In this scheme, the natural length for the VH sequences is 149 amino acids, as it can be seen in the example showed in Figure 4.

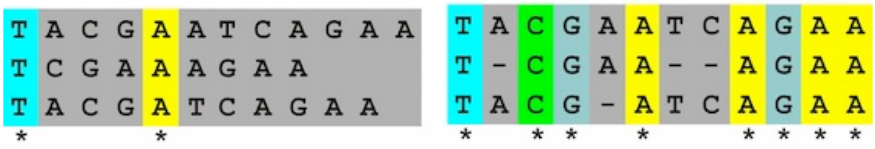


Figure 3: Picture taken from [11]. Example of sequence alignment. The length of the sequences has been reduced to 12 amino acids to facilitate the visualization of the alignment process. It can be seen that before the alignment (left panel) there are only 2 positions containing the same amino acid in the 3 sequences. However, after the alignment (right panel), 8 positions are containing the same amino acid in all 3 sequences.

```
QVQLVQS-GAELKKPGASVKVSCEASG-YSFTD-----HHIHWVRQAPGQGLEWLGWMNPV---TGAVNYARNFQGRVSF
YRTRELGIAYMDLRNLRFDDTAVYFCARKTAGDVSG-------------DNRGFFFDLWGRG-------
```

Figure 4: Example of a sequence aligned following the AHo scheme.

The sequences generated by IGoR are not aligned, so ANARCI [2] was used to perform the alignment. On the other hand, the sequences taken from [1] were aligned following the KABAT scheme instead of the AHo one. For this reason they had a length of 225 amino acids instead of 149. To solve this, all the gaps in the sequences were removed, obtaining unaligned sequences, and they were realigned following the AHo method, for which again ANARCI will be necessary [3].

[2] ANARCI is a tool to classify and number antibody and T-cell receptor amino-acid variable domain sequences. It can align sequences with the five most popular numbering schemes: Kabat, Chothia, Enhanced Chothia, IMGT and AHo.

[3] The python code used to remove the gaps from all the sequences in a data set and realign them using ANARCI can be found in the Annex 6.1

2.2 Sequence codification

Once all the sequences of a data set have been aligned, that data set becomes a set of M sequences of L amino acids each. These amino acids are encoded as characters, so the method presented in [6] will be used to go from characters to numbers. In this transformation, the residues are defined by a binary alphabet. Following this procedure, every site in the sequence is transformed into a Q length binary vector, being Q the number of different amino acids considered. So that binary vector will be filled by 0's excepting the position that the amino acid occupies in the amino acid alphabet that will be setted to 1. If that sequence site was occupied by a gap then the binary vector will be a full 0's vector. An example can be seen at Figure 5.

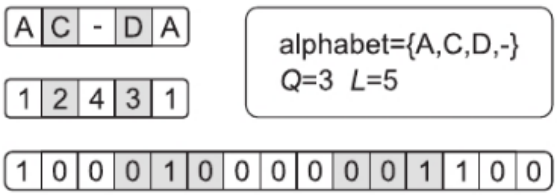


Figure 5: Binary sequence transformation example taken from [6]. To simplify the process, the binary amino acid dictionary considered is reduced (Q = 3) and also the sequence length (L = 5).

Note that, once this binary transformation has been applied to a data set of sequences, whole columns of 0's can appear. For that, it is sufficient that a certain amino acid j does not appear at position *i-th* in the whole data set. These null columns can be interpreted as an over representation of the phase space, which will allow the specificity of a data set to be studied. The greater the number of null columns, the more specific the data set will be.

The inverse problem of going from a binary sequence of $N = 20L$ bits to the corresponding sequence of characters is trivial, if the vector complies with the links related to the impossibility of having two active positions in the same block of 20 bits representing a single position, this would correspond to having two amino acids in the same position of the sequence.

As will be seen in the Section 2.4, one of the methods will consist of generating sequences in which each of their components follows a Gaussian distribution, as shown in the equation (12). However, these sequences, in general, do not comply with the links explained before. We have therefore to face with the problem of starting from a real vector and obtaining a binary vector that corresponds to an admissible sequence. That is, we need the generated string to be composed of 0's or 1's, but never of intermediate values, and only having one active position in the same block. To achieve this, *Límite* parameter is defined as shown in [9]. This parameter works as a threshold, so that if the component of our generated sequence is greater than *Límite* it is transformed into 1, being 0 otherwise. The numerical value of this threshold parameter will be taken from [9] and corresponds to 0.367. The use of this parameter as a threshold implies a discretization of the sequences that may lead to a loss of information, so a detailed study will be necessary.

2.3 The Multivariate Gaussian Model.

In Ref. [4] a Multivariate Gaussian Model was proposed, as a tool to infer an approximate statistical distribution of the human sequences, in order to perform an in-silico humanization, alternative to standard approaches like, e. g. "CDR grafting". In that model all the sequences in a data set are assumed to be distributed following a normal distribution:

$$f(x;\mu,\Sigma) = N(\mu,\Sigma) \tag{1}$$

being μ the mean and Σ the covariance matrix extracted from a sequence data base taken from [6].

Denoting the i-th sequence of a data set by x_i, when the empirical average and covariance are calculated in [4] an associated weight σ_i is introduced. These weights were introduced because in [4] the reference database is experimentally obtained. The goal of weighting the sequences is to reduce the possible experimental bias due to the fact that these sequences are not statistically independent of each other. In this project we do not have such a reference experimental database, so all sequences will be weighted uniformly. After all, the empirical average and covariance expressions are the following:

$$\bar{x}_i = \frac{1}{M}\sum_{m=1}^{M} x_i^m \tag{2}$$

$$C_{ij} = \frac{1}{M}\sum_{m=1}^{M}(x_i^m - \bar{x}_i)(x_j^m - \bar{x}_j) \tag{3}$$

The covariance matrix is not invertible so a Bayesian inference method is needed. This method requires a prior distribution for μ and Σ so a Normal Inverse Whisart distribution is taken as prior for μ and Σ.

$$p^{pr}(\mu,\Sigma) = N(\mu|\eta,\frac{\Sigma}{\kappa})IW(\Sigma|\Lambda,\nu) \tag{4}$$

Knowing that, the posterior distribution of μ and Σ given a data base X is :

$$p^{post}(\mu,\Sigma|X) \propto p(X|\mu,\Sigma)p^{pr}(\mu,\Sigma) = N\left(\mu|\eta',\frac{\Sigma}{\kappa'}\right)IW(\Sigma|\Lambda',\nu') \tag{5}$$

then, it can be deduced that:

$$<\mu>_{post} = \lambda\eta + (1-\lambda)\bar{x} \tag{6}$$

$$<\Sigma>_{post} = \lambda U + (1-\lambda)\bar{C} + \lambda(1-\lambda)(\bar{x}-\eta)(\bar{x}-\eta)^T \tag{7}$$

As can be seen in equations (6) and (7), the factor λ represents the proportion of prior to be included in the posterior one. That prior distribution, following again the steps of [6], will be taken as a flat distribution.

To set the optimal value of λ, the method proposed in [9] is followed. In this way, two new parameters d_1 and d_2 are defined as:

$$d_1 = \frac{|\Sigma_{cad} - \Sigma_{ref}|}{|\Sigma_{ref}|} \tag{8}$$

$$d_2 = \frac{|\mu_{cad} - \mu_{ref}|}{|\mu_{ref}|} \tag{9}$$

where Σ_{ref} and μ_{ref} are obtained from the experimental sequences taken from [5] but, as said before, weighing all the sequences equally. On the other hand, Σ_{cad} and μ_{cad} are obtained from the data set to be considered.

The optimal value of λ is the one that simultaneously minimizes the values of d_1 and d_2. It must be taken into account that the discretization of the sequences when going from continuous values to 0's and 1's could affect the simultaneous minimization of d_1 and d_2 indicating a different optimal value of λ for each parameter. This problem will be addressed in the following sections.

In order to quantify the humanness of any given sequence, a Multivariate Gaussian (MG) score was defined in [4]. The higher the MG score, the higher the humanness. Mathematically the MG score is defined as the logarithm of $p(y|X)$ being y the evaluating sequence and X the learning data base. In [4], $p(y|X)$ is defined as:

$$p(y|X) = t_N\left(\frac{M}{1-\lambda} + 2, \langle\mu_{post}\rangle, (1 + \frac{1-\lambda}{M})\langle\Sigma_{post}\rangle\right) \tag{10}$$

where t_N is the multivariate t-distribution probability density defined as:

$$t_p(\rho, \mu, S) = \frac{\Gamma(\frac{\rho+p}{2})}{\Gamma(\frac{\rho}{2})(\rho\pi)^{\frac{p}{2}}}|S|^{-\frac{1}{2}}\left(1 + \frac{1}{\rho}(y-\mu)^T S^{-1}(y-\mu)\right)^{-\frac{\rho+p}{2}} \tag{11}$$

A score similar to the one defined in the equation (11) is defined in [1]. This score was used to, once their Gaussian Model was inferred using their cluster of hypermutated sequences, predict the affinity towards an antigen. In this work, the procedure is the same, the score of the equation (11) is used to predict the affinity towards an antigen when the model is inferred using the hypermutated cluster obtained from [1] and compared it with what is obtained when using the results of the OU process.

2.4 Data set generation.

In this project two types of data sets are generated: those containing sequences generated following the Multivariate Gaussian Model [4] and those generated with IGoR [3].

In the Multivariate Gaussian data sets, each component of the generated sequences is distributed following a Gaussian distribution. Thus, the distribution to be followed by the sequences is:

$$p(x|\mu,\Sigma) = \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}} exp(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)) \tag{12}$$

where Σ and μ will be inferred from the model developed in [4] obtained from the experimental database. This data set will be used as a comparison benchmark, while learning the gaussian distribution that best approximates the sequences generated by IGoR.

The other type of data sets has been generated using IGoR. To generate this kind of data sets also ANARCI [7] and Biopython [10] are needed. The procedure is as follows:

- First, we generate sequences with IGoR. These sequences are nucleotide chains that have not been translated yet. In addition, before generating the sequences it is necessary to provide IGoR with the germline genes, which will be obtained from [1].
- Once the sequences have been generated, they must be translated into amino acids. For this, the Biopython library is used.
- The next and last step is to align those translated sequences. In this step, as already said in subsection Section 2.1, ANARCI is used.

In this case, we will work with the VH domain. That is why all the sequences generated by IGoR will have an ASCII length of 149 amino acids and a binary length of 2980, considering Q = 20.
The python script that performs all the generation process can be found at Annex 6.2.

2.5 Data base optimal size estimation.

One of the key issues when working with a data set, is its size. It has to be large enough so that the parameters estimated from the data set, such as the mean and the covariance, have an acceptable error, but small enough so it is computationally possible. So the goal is to find the optimal data set size using the combination of three different criteria: the parameters d_{ant} and d_{inv}, the covariance norm and the number of null columns of the data set.

- **The parameters d_{ant} and d_{inv}.**
  The parameters d_{ant} and d_{inv} have been defined in the following way:

$$d_{ant} = \frac{|\Sigma_i - \Sigma_{i-1}|}{|\Sigma_{i-1}|} \tag{13}$$

$$d_{inv} = \frac{|\Sigma_i - \Sigma_{inv}|}{|\Sigma_{inv}|} \tag{14}$$

  [These parameters are similar to d_1 and d_2 defined in (8) and (9)] On one hand, d_{ant} represents the difference in the covariance, when estimated before, Σ_{i-1}, and after, Σ_i, adding ΔN new sequences. When this parameter tends to zero means that the ΔN new sequences added have not contributed with any new relevant information to the covariance of the data set and

therefore, it does not make sense to keep adding sequences.

On the other hand, d_{inv} represents the difference in the covariance between the step at which the covariance matrix was first invertible, Σ_{inv}, and the i-th step Σ_i. In this case, the indication that adding more sequences to the data set will not add any more relevant information will be that this parameter becomes constant.

- **The covariance norm.**

  The second indicator will be the Frobenius norm of the covariance matrix, defined as follows:

$$C_{norm} = \sum_{i,j=1}^{L\cdot Q} |C_{ij}|^2 = \sum_{i,j=1}^{L\cdot Q} \left| \left(\frac{1}{M} \sum_{m=1}^{M} (x_i^m - \bar{x}_i)(x_j^m - \bar{x}_j) \right) \right|^2 \tag{15}$$

  being M the number of sequences in the data set, L the sequence length that will be 149 in this case and Q the number of amino acids in our amino acid alphabet.

  Every time that ΔN sequences are added to the data set, the covariance norm is calculated. As in the d^{inv} previous case, the indication that adding more sequences to the data set will not add more relevant information will be that the covariance norm stabilizes.

- **Null columns of the data set.**

  The last indicator is the number of null columns in the data set. As it was explained in the Section 2.2, a null column is a column full of 0's when the sequences are translated to binary. The fact that the number of null columns remains fixed means that the last sequences added to the data set have not contributed to new amino acids different from those already present in each column. This indicator does not make sense on its own, since the number of different amino acids in each column could remain constant but their percentage of occurrence varies each time that new sequences are added. However, it will be useful when combined with the other indicators explained previously.From a technical point of view, a null column implies that the covariance matrix will not be invertible, so that some kind of regularizing prior is needed.

To reduce the computational cost, the procedure when estimating the data base optimal size will be as follows[4]:

- Consider a data set of n sequences.
- Calculate the mean, the covariance and the null columns of the previous data set.
- Generate ΔN new sequences and add them to the initial data set.

[4]The python code used to estimate the mean and the covariance of a large data set following this method can be found in the Annex 6.4.

- Recalculate the mean, the covariance and the null columns of the new N ($N = n + \Delta N$) data set. In the case of the mean and covariance we do not recalculate them from scratch. They will be recalculated from the n sequences data set in an iterative process, as follows:

$$\bar{x}_N = \frac{n}{N}\bar{x}_n + \frac{\Delta N}{N}\bar{x}_{\Delta N} \tag{16}$$

$$C_N = \frac{n}{N}C_n + \frac{\Delta N}{N}\left((\bar{x}_{\Delta N} - \bar{x}_N)(\bar{x}_{\Delta N} - \bar{x}_N)^T + \frac{1}{\Delta N}\sum_{m=1}^{\Delta N}(x_i^m - \bar{x}_{\Delta N,i})(x_j^m - \bar{x}_{\Delta N,j})\right) \tag{17}$$

  being:

  - C_N and $\bar{x}_N$ the covariance and the mean of the N data set.
  - C_n and $\bar{x}_n$ the covariance and the mean of the n data set.
  - $C_{\Delta N}$ and $\bar{x}_{\Delta N}$ the covariance and the mean of the ΔN data set.

- Apply the three size evaluating criteria: the parameters d_{ant} and d_{inv}, the covariance norm and the number of null columns of the data set.

- Repeat all the previous steps until satisfied.

Note that this method is not only necessary in order to increase the efficiency of the algorithm, but also to reduce the memory cost of the process, because this method only considers delta N sequences at a time, so the computer's RAM memory does not limit the size of the data set.

2.6 Ornstein−Uhlenbeck process.

An Ornstein-Uhlenbeck process is a stochastic process that is used in this project as a noisy relaxation process: from the naive sequences generated by IGoR to the highest MG score sequences. Thus, the objective is to model the maturation process using this OU process even if slightly modified, to take into account that we do not know exactly the naive sequence that represents the starting point of the evolution (as it would be implicitly required by the standard OU process, where the variance is null at the beginning).
This modeling allows to estimate the links that join the initial state and cluster of sequences with the highest MG score, and also to characterize statistically the latter, by estimating its mean and covariance. From now on, this target mean and covariance will be referred to as attractors and will be denoted by θ and ω respectively. Notice that these attractors θ and ω are considered unique.
However, the Ornstein-Uhlenbeck process unfortunately does not support a conjugate distribution whose analytical expression can be used to simplify the maths. For this reason, we introduce an approximate scheme, where clusters of observed sequences are considered as multivariate gaussian snapshots of the evolution, and separate Normal Inverse Wishart distributions are provided as prior distributions for such snapshots. That is, we assume that all antibodies have a common birth time

τ, identifying clusters of sequences which will be assumed to have evolved at the same time and adopt a scheme in which the ordinary OU process is replaced by a system of temporally linked gaussian distributions inferred from these clusters, which are obtained in the Section 3.3.
To perform well this modified Ornstein-Uhlenbeck process an initial state is needed which is unknown and therefore it can be assume to be its attractor, with no evolution. In this way, it is defined by a Normal Inverse Whisart (NIW) distribution that is defined as the combination of a normal distribution and a prior defined by another NIW distribution. That normal distribution is inferred from the germline cluster whereas the NIW prior distribution is inferred from a data set generated by IGoR.

$$p(\theta_0, \omega_0|X_0) = \frac{\mathcal{N}(\theta_0, \omega_0)p^{pr}(\theta_0, \omega_0)}{\int d\omega_0 d\theta_0 \mathcal{N}(\theta_0, \omega_0)p^{pr}(\theta_0, \omega_0)} \tag{18}$$

and the priori distribution as a Normal Inverse Whisart:

$$p^{pr}(\theta_0, \omega_0) = \mathcal{NIW}(\eta_0, \gamma_0, \psi_0, \nu_0) \tag{19}$$

so (18) is still a NIW distribution, with

$$\begin{aligned}
\eta_0' &= \frac{n_0\overline{X}_0 + \gamma_0\eta_0}{n_0 + \gamma_0} \\
\gamma_0' &= \gamma_0 + n_0 \\
\psi_0' &= \psi_0 + n_0\overline{C}_0 + \frac{n_0\gamma_0}{n_0 + \gamma_0}(\overline{X}_0 - \eta_0)(\overline{X}_0 - \eta_0)^T \\
\nu_0' &= \nu_0 + n_0
\end{aligned} \tag{20}$$

where

$$\begin{aligned}
\overline{X}_0 &= \frac{1}{n_0}\sum_{c\in\pi_0^*} X^c \\
\overline{C}_0 &= \frac{1}{n_0}\sum_{c\in\pi_0^*} (X^c - \overline{X}_0)(X^c - \overline{X}_0)^T
\end{aligned} \tag{21}$$

Now, it will be taken as posterior values for θ_0, ω_0 the mode for the $\mathcal{NIW}$ above.

$$\begin{aligned}
\theta_0^{post} &= \eta_0 \\
\omega_0^{post} &= \psi_0/(\nu_0 + G + 2)
\end{aligned} \tag{22}$$

where G is the dimension of the sequences after the binary transformation described in Section 2.2. Notice that not all the sequences of the IGoR data set will be used to infer the NIW distribution. First, the K-means algorithm based on the Hamming distances will be applied to the IGoR data base to verify in how many different clusters the data set can be divided. Then, the Multivariate Gaussian model described in Section 2.3 will be inferred using the from each of the different clusters, and using the MG score the compatibility between each of the IGoR clusters and the germline

cluster can be quantify. Only the IGoR cluster with the higher compatibility with the germline cluster will be use to infer the a priori distribution of the initial state.

The rest of the clusters considered in the modified OU process, which are obtained in the Section 3.3, will be indexed as $j = (1, \ldots, N_{cl})$ where N_{cl} is the number of clusters considered in the maturation process.
Taking into account the assumptions done at the beginning of the OU section, the probability of observing a state X at time t is given by the modified Ornstein-Uhlenbeck probability, defined as:

$$p(X|\alpha, \theta, \omega) = p^{MOU}(X^c|\Delta t_j, \alpha, \theta, \omega, \theta_0, \omega_0) = \mathcal{N}(\mu(\Delta t_j), \Sigma(\Delta t_j)) \tag{23}$$

where from now on it can be set:

$$\begin{aligned} \mu_j &= \mu_j(\Delta t_j^*)) = \theta - E_j(\theta - \theta_0) \\ \Sigma_j &= \Sigma(\Delta t_j^*)) = \omega - E_j(\omega - \omega_0)(E_j)^t \end{aligned} \tag{24}$$

with

$$E_j = E_j(\Delta t_j)) = \exp\left(-\alpha \Delta t_j\right) \tag{25}$$

where Δt_j represents the time elapsed from the moment that the naive antibody is synthesized to the measurement time and α the relaxation matrix. Notice that this matrixα is supposed to be block diagonal and also symmetric in order to make the problem computationally affordable. This implies real eigenvalues (i.e., multi-exponential relaxation), and an evolution dynamics where each position mutates independently from the others, and just the present species affects the mutated one.
In this way, the probability of certain trajectory in the phase space for the maturation process:

$$\mathcal{P} = \prod_{j=1}^{N^{cl}} \left(\mathcal{N}(X|\mu_j, \Sigma_j) p^{pr}(\mu_j, \Sigma_j)\right) \tag{26}$$

Assuming now that $p^{pr}(\mu_j, \Sigma_j) = \mathcal{NIW}(\mu_j, \Sigma_j|\eta_j, \gamma_j, \psi_j, \nu_j)$, being:

$$\begin{aligned} \eta_j' &= \frac{n_j \overline{X}_j + \gamma_j \eta_j}{n_j + \gamma_j} \\ \gamma_j' &= \gamma_j + n_j \\ \psi_j' &= \psi_j + n_j \overline{C}_j + \frac{n_j \gamma_j}{n_j + \gamma_j} (\overline{X}_j - \eta_j)(\overline{X}_j - \eta_j)^T \\ \nu_j' &= \nu_j + n_j \end{aligned} \tag{27}$$

where

$$\overline{X}_j = \frac{1}{n_j}\sum_{c\in j}^{n_j} X^c$$
$$\overline{C}_j = \frac{1}{n_j}\sum_{c\in j}^{n_j}(X^c - \overline{X}_j)(X^c - \overline{X}_j)^T \tag{28}$$

At his point, following the maths done in [6] and in [4], it is possible reduce the number of parameters just choosing:

$$\gamma' = \nu' + G - 1 \tag{29}$$

In the equation (27) the parameters η_j and ψ_j represents the connection between the cluster j and the initial state, inferred from the germline cluster. Both parameters can be written in the following way:

$$\begin{aligned}
&\eta_j = \eta^{ref} - E_j(\eta^{ref} - \eta_0) = \eta^{ref} - E_j(\eta^{ref} - \theta_0^{post}) \\
&\psi_j = \psi^{ref} - E_j(\psi^{ref} - \psi_0)E_j^t = \psi^{ref} - E_j(\psi^{ref} - \omega_0^{post}(\nu_0 + G + 2))E_j^t \\
&\gamma_j = \gamma \ , \ \forall j = 1, \dots, N^{cl} \\
&\nu_j = \nu \ , \ \forall j = 1, \dots, N^{cl}
\end{aligned} \tag{30}$$

where η^{ref} and ψ^{ref} are taken from a uniform distribution.
The goal of the (30) is to reintroduce a coupling between the different distributions learned from the different clusters: instead of finding an analytic form of a conjugate prior for the OU parameters θ_j, ω_j, that we are not able to find, we propose independent NIW priors for all the snapshot of the OU process corresponding to the same-time-t_j-clusters, but then we couple the hyper-parameters of such NIW distributions, to impose that the only difference among them is related to the different evolution moment as ruled by E_j.
Now, the probability 26 can be maximized on the Σ_j, μ_j variables just by taking the mode of the NIW distribution:

$$\begin{aligned}
\mu_j^{post} &= \mu_j^{mode} = \eta'_j \\
\Sigma_j^{post} &= \Sigma_j^{mode} = \frac{\psi'_j}{\nu'_1 + G + 2}
\end{aligned} \tag{31}$$

This μ_j^{post} and Σ_j^{post} choice corresponds to the maximum of the NIW:

$$\mathcal{M}_j = \frac{(\gamma')^{\frac{G}{2}}(\nu' + G + 2)^{G(\frac{\nu'+G+2}{2})}}{(2\pi)^{\frac{G}{2}} 2^{\frac{\nu' G}{2}} \Gamma_G(\frac{\nu'}{2}) \det(\psi'_j)^{\frac{G+2}{2}}} e^{-\frac{G}{2}(\nu'_{j,a}+G+2)} \tag{32}$$

The next step, is to substitute this maximum in 26 and maximize on the variables α and ΔT_j. These minimized values are considered the optimal values for both variables and make possible to find the attractors θ and ω according to the following expression:

$$\mu_j^{post} = \theta^{post} - E_j^{opt}(\theta^{post} - \theta_0^{post}) \tag{33}$$

$$\Sigma_j^{post} = \omega_j^{post} - E_j^{opt}(\omega^{post} - \omega_0^{post})(E_j^{opt})^T \tag{34}$$

Notice that we are using as attractor the mean of the fully maturated sequences cluster, θ^{post}, what is a vector of length G whose components are continuous variables evaluated on a range of 0 to 1. Therefore, when translated into amino acids it must be discretized following the procedure of Section 2.2. Because of this, in (3.1) we check whether this discretization leads to a loss of information and whether this information is sufficient for the discretized mean to cease to be an attractor.

Once the attractors θ^{post} and ω^{post} are obtained, since they are assumed to characterize the fully matured sequences cluster, it can be checked, using the MG score described in Section 2.3 if the sequences of the hypermutated cluster obtained from [1] are fully maturated or they were just maturated until their virus affinity was high enough. If they are not fully maturated sequences it means that a data base of sequences with a higher affinity than the hyper mutated ones can be generated using the gaussian method described in Section 2.4.

3 Results and discussion

In this section we present and discuss the results obtained.
In Section 3.1 we test whether the discretization of sequences leads to a loss of information.
As mentioned in Section 2.6, we are going to use a NIW distribution inferred from an IGoR data set as the prior distribution for the initial state of the modified Ornstein-Uhlenbeck process. Therefore, in the (3.2) we worked with the parameters defined in Section 2.5 to estimate which is the optimal size of the IGoR data set to use. In addition, we also compare that result with the one obtained by using a Gaussian data set.
On the other hand, the use of different clusters of sequences to perform the modified OU process was also explained in the Section 2.6 section. These clusters will be obtained in the Section 3.3 together with the approximations needed due to the huge computational cost of the OU process. Finally, Section 3.4 contains the results obtained from the simulation of the modified OU process, together with an interpretation of the obtained relaxation matrix and the maturation times of each cluster. We also work with the attractors obtained, checking if the sequences of the hypermutated cluster of [1] are really fully matured.

3.1 Sequence discretization.

The goal of this section is to check whether the discretization of sequences leads to a loss of information. Furthermore, we also quantify this loss by checking whether the choice of optimal parameters changes when the Multivariate Gaussian model described in Section 2.3 is used or whether the mean is no longer a good attractor once it is discretized.
As mentioned in Section 2.2, going from ASCII to binary sequences is a trivial process, but the reverse process is not. In the process of going from binary sequences to ASCII sequences it was necessary to discretized the sequences using the variable *Lìmite* that acted as a threshold. This discretization could lead to a loss of information, as well as the fact that the variables d_1 and d_2, defined in (8) and 9 respectively, used to find the optimal value of λ, are not minimized simultaneously. In this section several tests are performed to study this discretization process thoroughly.
Using the scripts described in Annex 6.2, two data sets of Gaussian sequences are generated. In one of them the sequences are discretized before calculating μ and Σ and in the other one they are not. In this way, what are the minimum values of d_1 and d_2 for each lambda value in both data sets, see Figure 6 if there are not one but two predictions in the discrete case and it can be checked if the lambda predictions are the same in both cases.

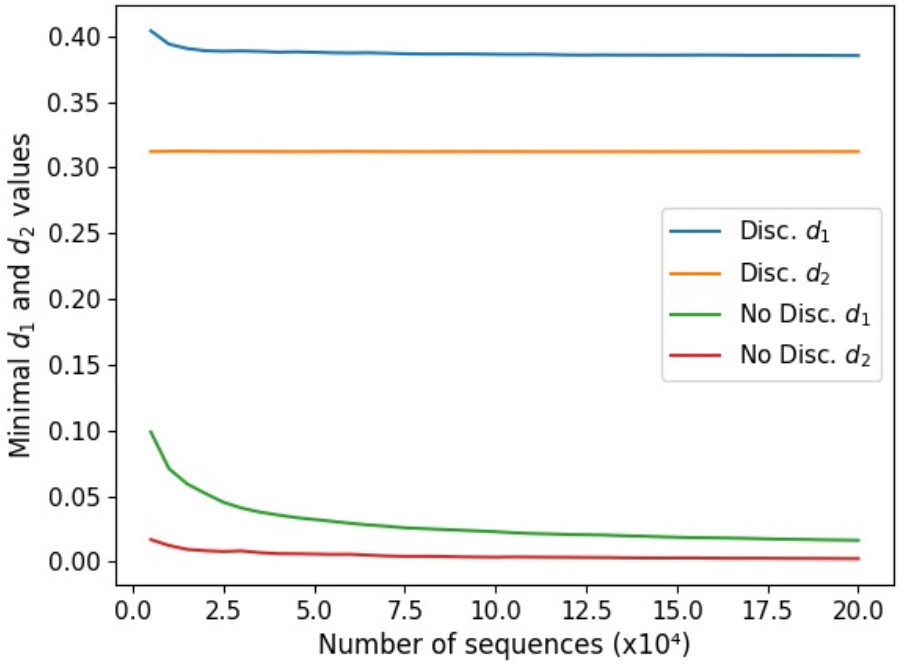


Figure 6: Minimal d_1 and d_2 values.

As can be seen in Figure 6, the minimum values of d_1 and d_2 are not the same whether we discretized the sequences or not. If it is looked at d_1 it can be observed that, in the discrete case, it starts with a value similar to 0.4 and, although it starts to decrease as the number of sequences increases, it remains constant from twenty-five thousand sequences onwards. On the other hand, in the continuous case, d_1 begins with a much lower value than in the discrete case, namely 0.1 and, although this time it decreases slightly as the the size of the data set increases without becoming completely constant.

If we now look at d_2 it can observed that in the continuous case its value starts already very close to 0 and practically does not vary with the number of sequences. However, in the discrete case, it starts with a value of 0.32, remaining stable regardless of the data set size.

The fact that the minimum values of both parameters, d_1 and d_2, are shorter in the continuous case than in the discrete case means that, regardless of the size of the data set, it is obtained a mean and a covariance closer to the reference ones in the continuous case. This fact reflects that there is indeed a loss of information in the discretization process.

The next step is to check if this loss of information is big enough to cause the predictions of the optimal lambda when inferring the Multivariate Gaussian model between the discrete case and the non-discrete case to not coincide. In this case it is checked that different values of λ are obtained depending on whether the sequences are discretized or not but they do not depend on the data set

size once it is bigger that two thousand sequences. In the non-discrete case, it is also check that the prediction of the optimal value of λ is the same for d_1 and d_2. However, in the discrete case it is obtained different optimal λ values from d_1 and d_2, i.e. both parameters are not being minimized simultaneously.

Finally, it has to be checked if the discretized mean is still a good enough attractor. To do this, we take the human and murine [5] databases and also a database generated with IGoR. Once this is done, we infer the Multivariate Gaussian model using each of these databases and check if the discretized mean of each one obtains a high enough score to be considered an attractor. Notice that when inferring the MG model a value of $\lambda = 0.1$ is used as in [4], fixing instead $\Omega = 1$ since we are weighting all the sequences equally. The results are the following:

	Human data base	Murine data base	IGoR data base
MG score	18723.99	22532.30	9077.39

Table 1: MG scores of the discretized mean of a human, a murine and an IGoR data bases, learning the MG Model from each of them.

The the maximum score found in [4] taking $\lambda = 0.1$ and $\Omega = 0.498$ was 13105.79. This maximum score could vary when changing the Ω value, but we can be sure that the discretized mean still have a very high score. Note that the human and murine data bases contain sequences consisting of a heavy chain and a light chain, so they have a length of 298 amino acids when aligned with ANARCI instead of 149 as in IGoR. For this reason the IGoR score is lower, because the maximum score in their case is also lower.

As it has been showed the discretization of sequences entails a loss of information, as can be seen in Figure 6. To quantify this loss of information, we have checked whether the optimal value of λ when the MG model is inferred from a data base of sequences that has been discretized and another that has not and also whether the score of the discretized mean is high enough so that it ceases to be an attractor. In the first case we find that not only are different values obtained depending on whether the data base has been discretized or not, but we also observe that different predictions are obtained for the two parameters d_1 and d_2. However, this loss of information is not large enough to prevent the discretized mean from remaining an attractor.

3.2 Optimal data set size study.

In the Section 2.6, it is stated that a NIW distribution inferred from an IGoR data set is to be used as the a priori distribution for the initial state of the modified Ornstein-Uhlenbeck process. In this section we estimate the minimum size that the IGoR data set must have in order for the results estimated from it to be reliable. Therefore, we work with the parameters defined in Section 2.5 , i.e. the parameters d_{ant} and d_{inv}, the covariance norm and the number of null columns of the data

set. In addition, we also compare that result with the one obtained by using a Gaussian data set.

3.2.1 d_{ant} and d_{inv} parameters.

Firstly we evaluate the parameters d_{ant} and d_{inv} described by the equations (13) and (14) to estimate the optimal size of a data set. The experiment was performed both with one data set generated with IGoR and another one generated following the equation (12) explained in Section 2.2. Performing double test, we will be able to check if the optimal size of the data set varies depending on both the type of sequences that compose it and also how d_{ant} and d_{inv} vary, depending on both the number of sequences and the type the of data set. Once both data sets have been generated and the parameters d_{ant} and d_{inv} have been evaluated following the steps described in Section 2.5, the results obtained can be seen in Figure 7.

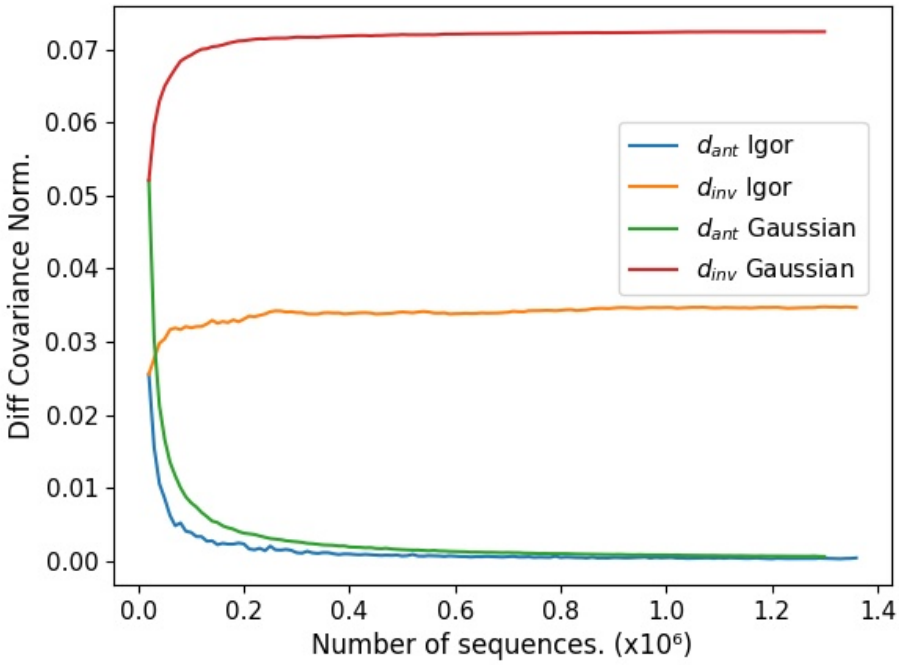


Figure 7: d_{ant} and d_{inv} dependence with the type and size of a data set.

It can be seen that d_{inv} has greater values in the Gaussian data set case than in the IGoR data set one. On the other hand, the tendency to stabilize in both parameters in both cases is rather fast. In the IGoR data set case it seems that with four hundred thousand sequences, both parameters have already stabilized, so this number of sequences would be the minimum that this criterion would require. In the case of Gaussian sequences, a slightly larger data set of about five hundred thousand sequences would be necessary.

3.2.2 Frobenius norm of the covariance.

The second indicator to estimate the optimal data set size is the Frobenius norm of the covariance following the (15). As in the previous Section 3.2.1, two data sets are considered, one generated by IGoR and another one generated following the equation (12). Again, once both data set have been generated and the norm of the covariance has been evaluated following the steps described in Section 2.5 the results shown in Figure 8 are obtained.

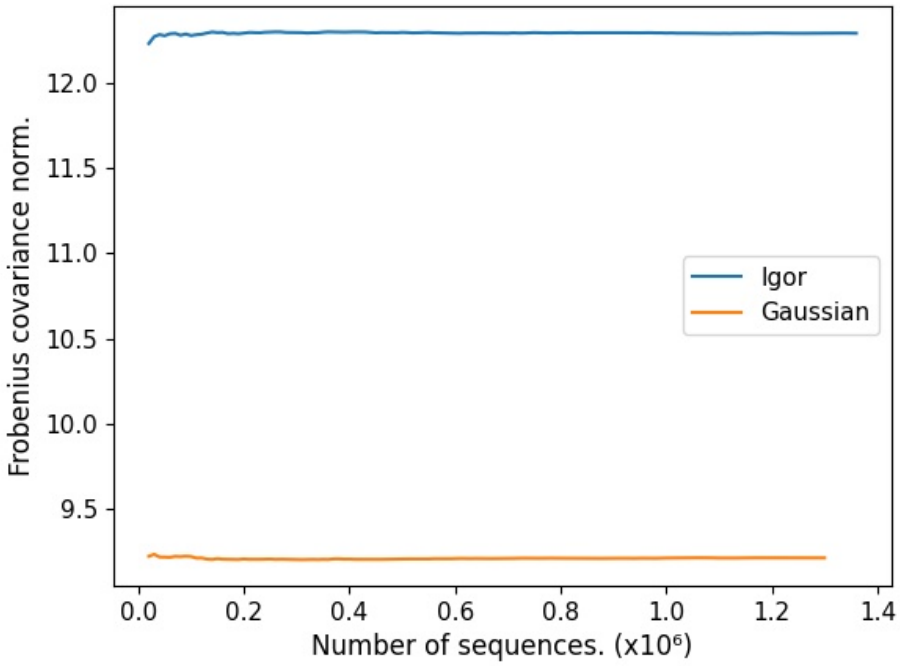


Figure 8: Frobenius covariance norm dependence with the type and size of a data set.

As it can be seen in this case, the covariance norm stabilities for about two hundred thousand sequences, regardless of whether the data set is Gaussian or IGoR-generated, establishing then this minimum for data set size. Notice Figure 8 that the values of the covariance norm for the gaussian data set case are lower than the IGoR ones, what its the opposite case than in the Figure 7.

3.2.3 Null columns indicator.

Finally, the third indicator to estimate the optimal data set size is the number of null columns. As it was explained in the Section 2.2, a null column is a column full of 0's when the sequences are translated to binary. Thus, these null columns imply that an amino acid j does not appear in position i-th in the whole data set. Following then the procedure explained in Section 2.4, if from a certain size of the data set the number of null columns remains stable, it will indicate that the minimum size of the data set has been reached. Then, in the Figure 9 it can be seen the number of

null columns as a function of the size of the data set.

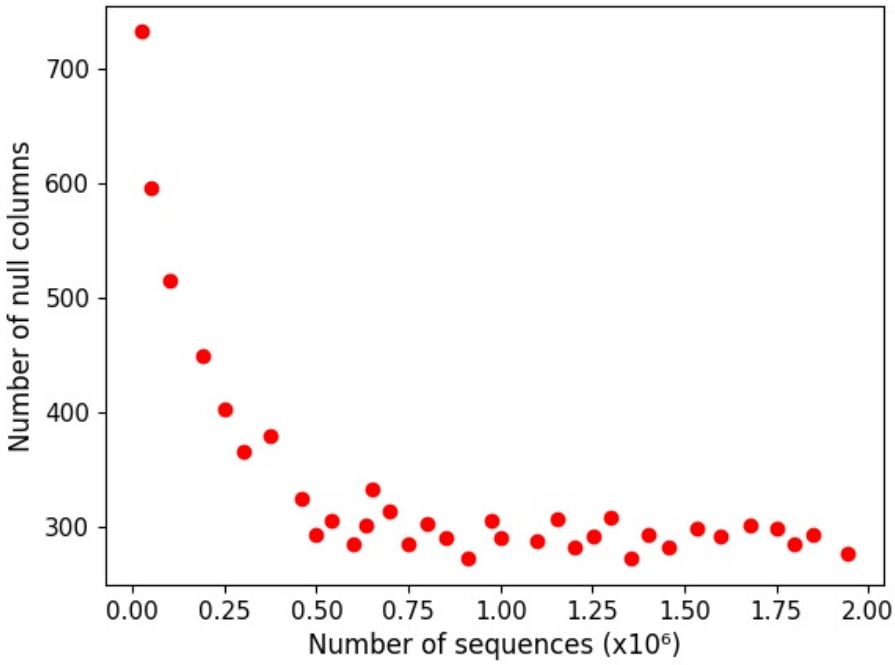


Figure 9: Number of null columns in an IGoR data set as a function of the data set size.

It can be observed that the number of null columns in an IGoR data set depends on the size of it up to 10^5 sequences, when it stabilizes, so this will be the value for this indicator.
[5] The number of null columns can be also used to check if the maturation process is a process of sequence specification, starting from the sequences generated by IGoR to those corresponding to the hypermutated cluster in [1]. Notice that the latter are sequences of antibodies that have undergone a maturation process. On the other hand, the sequences generated by IGoR are the so-called naive sequences, sequences that have not been matured yet.

To check if there is a sequence specification process when starting from the sequences generated by IGoR to those taken from [1], we will calculate the number of null columns in both data sets and also check if the null columns in IGoR are also null in the reference data set. This is because null columns can be seen as an over representation of the space phase. Therefore, if the number of null columns in the maturated data set is higher than in the IGoR one and also the IGoR null columns are contained in the reference null column set, it can be said that there is a specification process.

What was found is that the number of null columns in the data set taken from [1] is 2044.

[5]The python code to calculate the number of null columns can be found in the Annex 6.3.

Taking into account that these are sequences of length 149 with 20 possible amino acids in each position and therefore with 2980 binary columns, the fact that there are 2044 null columns means that 68.5% of the binary columns are null. This is an extremely specific data set.

On the other hand, continuing with the question of whether or not there is a sequence specification process, the next step is to see if these null columns shown in Figure 9 are also null in the reference data set. From now on, the null columns in the IGoR data set that are not null in the reference data set will be called *Non-matching null columns*. Since the number of null sequences depends on the size of the IGoR data set, it is assumed that the number of non-matching null columns also depends on the data set size. Thus, the results obtained can be seen in the Figure 10.

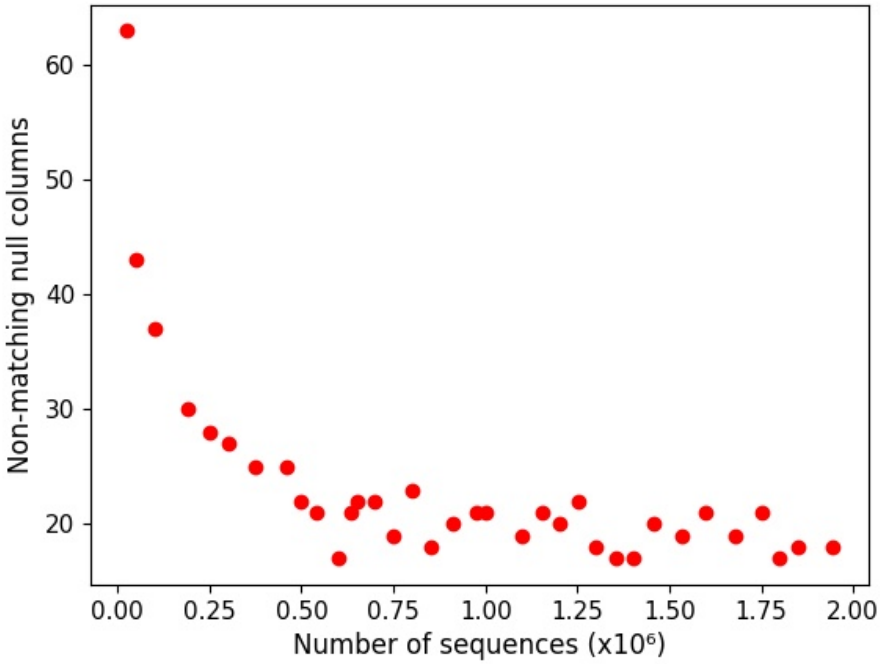


Figure 10: Number of non-matching columns between the IGoR data set and the reference data set as a function of the IGoR data set size.

As can be seen in Figure 10, the number of non-matching sequences depends on the size of the IGoR data set. It can be also observed that, exists a behavior similar to the observed in the Figure 9. Previously, it has been said that to confirm that we were dealing with a sequence specification process, the null columns in the IGoR data set should also be null in the data set taken from [1], which was used as a reference. However, looking at the Figure 10 it can be seen that it is impossible for the number of non-matching null columns to go down to zero, because when the data set reaches a size of about six hundred thousand sequences the number of non-matching columns stops going down, starting to oscillate around a value of 20. Despite this, the specification process is not yet discarded. The next step is to check the variability of these mismatched columns in the

reference database. If their variability is too high, this could imply an important loss of information, while if it is too low, it could be neglected. In this way, 3 sufficiently large data sets are generated with IGoR, namely containing six hundred thousand (data set 1), seven hundred thousand (data set 2) and nine hundred thousand sequences (data set 3). Then, it can be seen in the Figure 11 the variability in the reference data base of their mismatching columns.

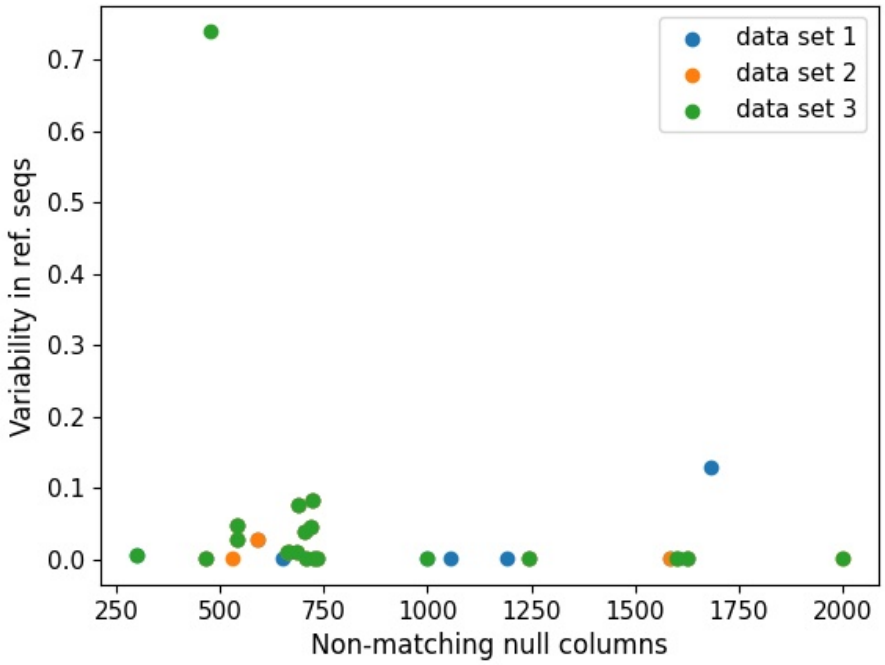


Figure 11: Null non-matching columns variability in the reference data set.

As can be seen, except for one column in the data set 1, the variability in the reference data set of the non-matching null columns is at most 15%. In fact, if we look at the specific case of the data set 3, it is observed that no column exceeds 3% variability. Therefore, it can be ensured that, excepting specific cases such as that one column with 75% variability in the 600k sequences data set, the null columns in the IGoR data set are either also null in the reference data set or their variability is low enough so as not to imply a significant loss of information. This confirms that we are indeed dealing with a process of sequence specification.

3.3 Clustering analysis.

As mentioned in Section 2.6, the modified OU process consists of a series of temporary linked Gaussian distributions, which each of them inferred from independent sequence cluster identified in the data base obtained from [1]. In this section, we realigned the sequences of this data base and applied the K-means algorithm to divide them into different clusters. The objective is to check if there are different sub-clusters that can be used in the modified OU process.

The data set of 3212 sequences taken from [1] was aligned following the KABAT scheme and divided in two clusters:

- A hypermutated cluster (S1) composed of 1578 more similar to the broadly neutralizing antibody VRC-PG04.
- A germline cluster (S2) composed of 1634 centered on the IGHV1-2*02 and IGHJ2*02 germline genes.

Both of them were obtained using the shallow tree clustering algorithm based on the Hamming distance between sequences. We perform our own clustering algorithm, but these two clusters S1 and S2 will be taken as a reference to identify which sequences form each of our own clusters.

3.3.1 Preprocessing of sequence data sets.

In the figure 7 of [1], Asti et al presented a density plot of the identity/divergence analysis between the productive sequences with inferred germline IGHV1-2 gene and the IGHV1-2 and VRC-PG04 sequences. They identify the high-density zone in the upper right zone of the figure with the hypermutated cluster, due to the large divergence from the germline gene IGHV1-2 and the similarity with VRC-PG04, and the high-density zone in the bottom left zone with the germline gene, due to the high similarity from the germline gene IGHV1-2 and the divergence with VRC-PG04. However, in this figure does appear a third high-density zone at the bottom right region. It can be seen that this high-density zone has a great divergence with the germline gene IGHV1-2 but also with VRC-PG04. The hypothesis is that the sequences that make up this third high-density zone are still undergoing maturation. That is, they would have started the maturation process, but they would not have completed it, so they could belong to a third cluster, different from the germline and the hypermutated ones.

In order to identify the sequences that make up each of these three high-density zones, the steps followed in [1] are repeated. The nucleotide sequences of the 3212 sequences composing the clusters S1 and S2 are obtained from [1][6] and also the nucleotide sequences of the IGHV1-2*02 gene and the VRC-PG04 antibody are obtained from [12]. Then, two alignment tests of the 3212 nucleotide sequences are performed, one setting as subject the IGHV1-2*02 gene sequence and the other setting as subject the sequence of the VRC-PG04 antibody. For this step it was used the tool BLASTn [13]. From this two tests it is obtained the necessary data to reproduce the figure 7 of [1]. After that, a clustering experiment is carried out using the K-Means algorithm, taking 3 as the number of clusters. In this way it is obtained the nucleotide sequences that make up each of the three high density zones Z1, Z2 y Z3 . Finally, all the sequences are translated using Biopython and realigned from KABAT to AHo using ANARCI. Here must be noticed that ANARCI is unable to realign

[6]The python code to obtain the nucleotide sequences can be found in Annex 6.7

some of the sequences, so from now on our data set will only have 3050 sequences (see Figure 12.

The next step is to check which sequences make up the clusters S1 and S2 with respect to these 3 high density areas Z1, Z2 and Z3.

- The S1 (hypermutated) cluster is composed by 1429 sequences: 1115 sequences of the upper right zone (Z1), 8 of the bottom left zone (Z2) and 306 of the bottom right zone (Z3).
- The S2 (germline) cluster is composed by 1621 sequences: 1555 of the bottom left zone (Z2) and 66 of the bottom right zone (Z3).

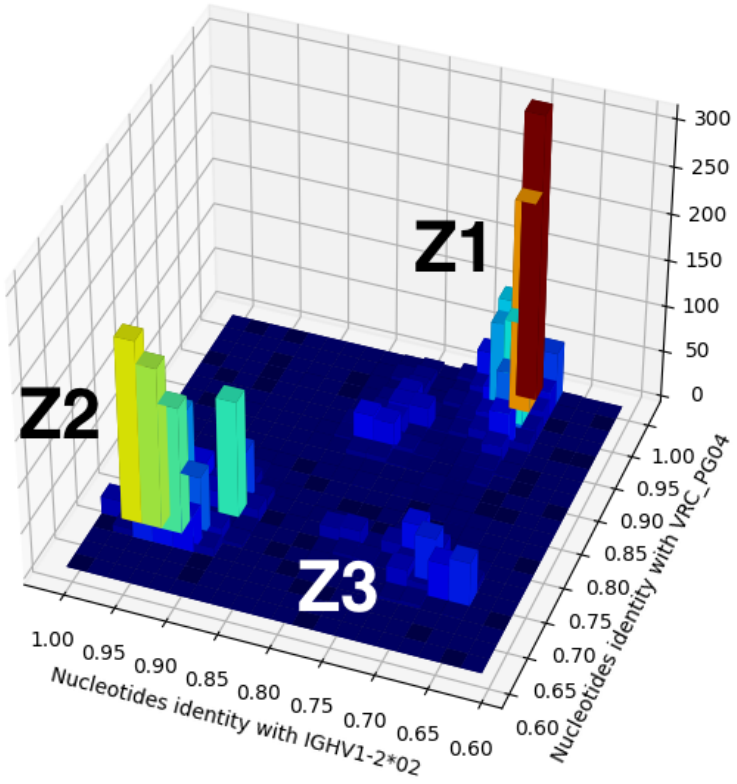
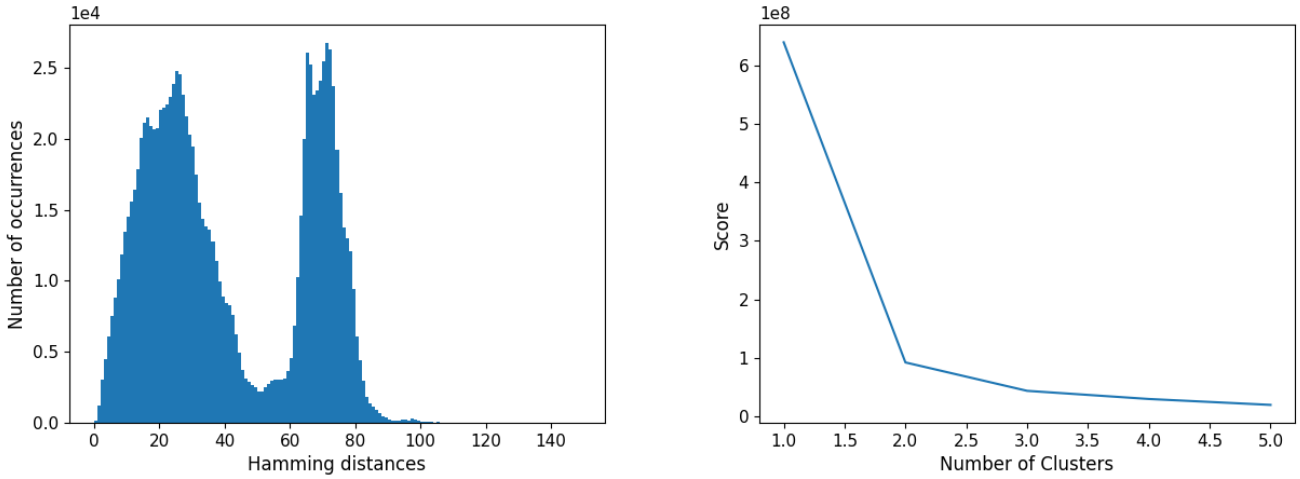


Figure 12: Reproduction of the Figure 7 of [1], with 3050 sequences, where are indicated the high density zones Z1, Z2 and Z3.

It seems that there is indeed a very good correlation between the upper right zone and the S1 cluster, and the bottom left zone and the S2 cluster. So they called hypermutated zone (Z1) and germline zone (Z2). It can also be seen that the bottom right zone (Z3) has been divided between S1 and S2, although not equally. We are interested in the S1 cluster, so we apply the K-means algorithm on it to see if it composed by several sub-clusters. In this way, the Figure 13 is obtained.



(a) Hamming distance distribution.

(b) Elbow curve.

Figure 13: The hypermutated cluster (S1).

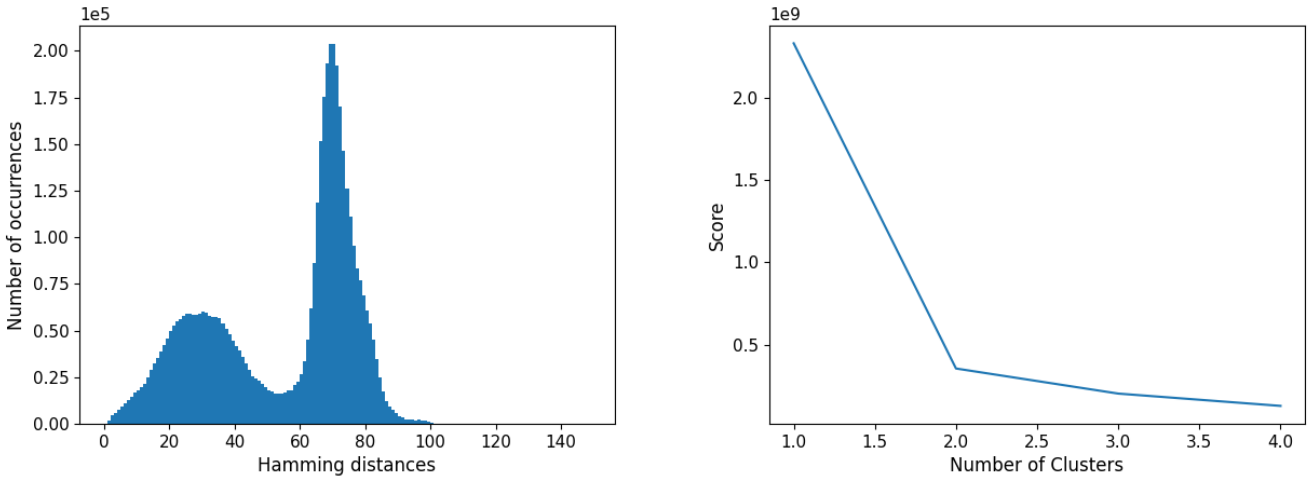
Looking at the Figure 13, both the elbow curve (right panel) and the Hamming distance distribution (left panel), it seems clear that we can split the cluster S1 into two sub-clusters S1.1 and S1.2. Once split, we check its composition with respect to Z1, Z2, and Z3.

- The S1.1 sub-cluster is composed by 1108 sequences: 1103 sequences of Z1, 2 of Z2 and 3 of Z3.
- The S1.2 sub-cluster is composed by 312 sequences: 12 of sequences of Z1, 6 of Z2 and 303 of Z3.

There is indeed a very good correlation between S1.1 and Z1, and between S1.2 and Z3. It seems that it is possible to separate the Z1 sequences from the Z3 sequences if we divide the S1 cluster into two sub-clusters, although when K-Means is applied to the whole 3050 sequences data set, the most robust solution was two clusters and not three.

3.3.2 Deeper clustering search.

In this section are presented the results of our own clustering of the data base obtained from [1]. The K-means algorithm is applied to the 3050 sequences, obtaining the Figure 14:



(a) Hamming distance distribution.

(b) Elbow curve.

Figure 14: The 3050 sequences data set realigned using ANARCI.

As it can be seen in 14 the most robust solution is to divide the data set into two clusters. In this way we divided the 3050 sequences into two clusters C1 and C2. Now, this new two clusters are compared with the clusters S1 and S2, finding that:

- The cluster C1 is composed by 1117 sequences: 1117 sequences of S1 y 0 of S2.
- The cluster C2 is composed by 1933 sequences: 311 sequences of S1 y 1622 of S2.

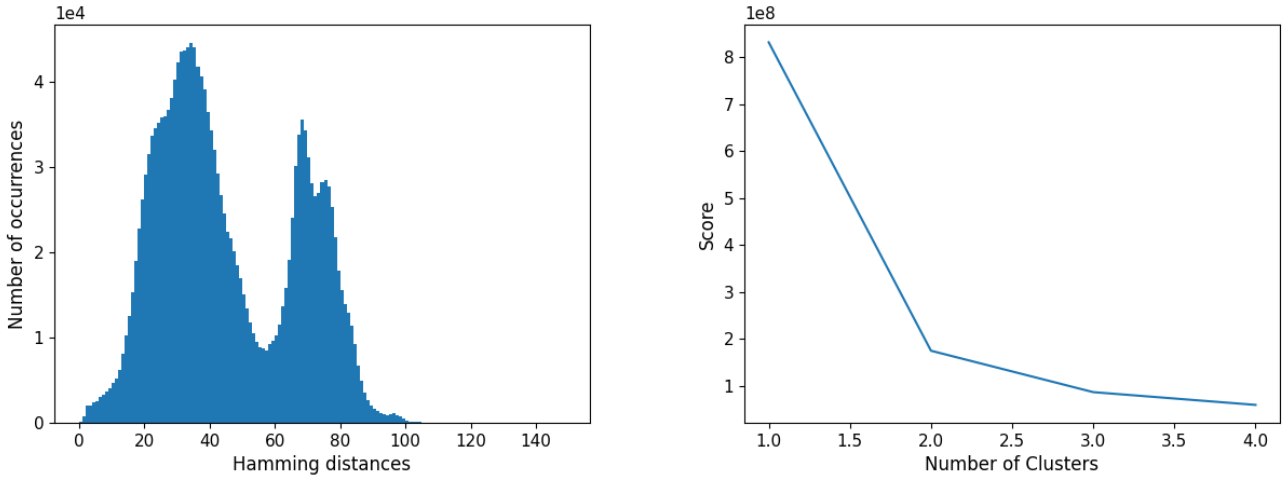
Also, we compare C1 and C2 with Z1 (hipermutated zone), Z2 (germline zone) and Z3 (intermediate zone) it is found that:

- The cluster C1 is composed by 1117 sequences: 1104 sequences of Z1, 8 of Z2 and 5 of Z3.
- The cluster C2 is composed by 1933 sequences: 10 sequences of Z1, 1556 of Z2 and 367 of Z3.

Looking at these last two comparisons, it can be seen that C1 is only composed of sequences belonging to S1 and that most of them also belong to Z1. On the contrary, it is seen that C2 is composed of 311 sequences from S1 and the rest from S2, most of which belong to Z2 and Z3. If we compare these results with those previously obtained when comparing S1 and S2 with Z1, Z2 and Z3, it is observed that the 367 that form Z3 were classified as hypermutated, assigned to cluster S1, in [1] and yet in our case they are assigned to cluster C2, together with the sequences of cluster S2

germline.

At this point we search for sub-clusters in C1 and C2, so we apply the K-means algorithm again to each of them. In the case of C1 it is found that there is not any sub-clusters. For this reason, and as it is composed only of sequences from S1 and Z1, it is considered as our hypermutated cluster. On the other hand, in the case of C2, when applying K-means the results shown in the Figure 15 are found.



(a) Hamming distance distribution.

(b) Elbow curve.

Figure 15: The 1932 sequences that form the C2 cluster.

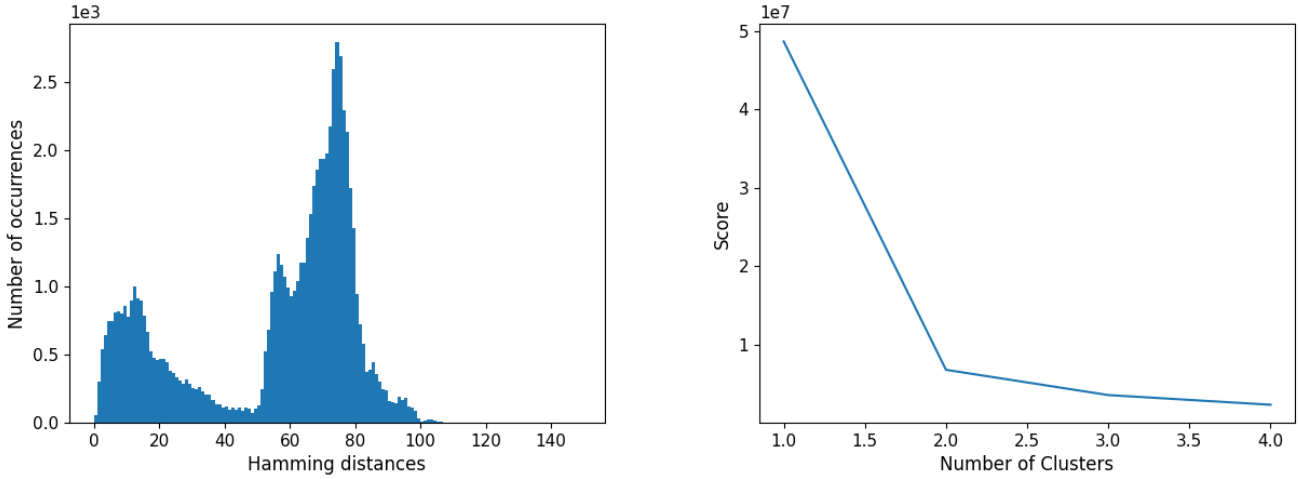
Several interesting things can be observed in this Figure 15. The first is that there do indeed appear to be two sub-clusters in cluster C2. Another thing is that if you look at the hamming distance distribution, there are two peaks centered at about 30 and 70. It can be seen that the second peak is right in the same position as the second peak in the figure Figure 13 where cluster S1 was represented. This seems to confirm the hypothesis that the Z3 sequences that were assigned to S1 in [1] haven assigned to our germline cluster.
The resultant two clusters are called C2_1 and C2_2. If we compare them with S1 and S2 and with Z1, Z2 and Z3 the following results are found:

- The cluster C2_1 is composed by 1556 sequences: 1 sequence of S1 y 1555 of S2.
- The cluster C2_2 is composed by 377 sequences: 310 sequences of S1 y 67 of S2.

Also, we compared C2_1 and C2_2 with Z1 (hipermutated zone), Z2 (germline zone) and Z3 (intermediate zone):

- The cluster C2_1 is composed by 1556 sequences: 0 sequences of Z1, 1493 of Z2 and 63 of Z3.
- The cluster C2_2 is composed by 377 sequences: 10 sequences of Z1, 63 of Z2 and 304 of Z3.

Looking at these two comparisons, it can be seen that the C2_1 cluster is composed, except for one sequence, of sequences from the germline S2 cluster and that most of these sequences belong to the Z2 zone (identified in [1] as germline). However, in the case of C2_2, it happens the opposite: 310 sequences belong to S1 and also 304 belong to Z3.
Taking these data into account, we searched again for sub-clusters within C2_1 and C2_2 using K-Means. In the case of C2_1 we did not find any more sub-clusters. Since it consists almost entirely of S2 sequences and also belongs to Z2, it is considered our germline cluster. On the other hand, in the case of C2_2, when applying K-means the results shown in the Figure 16 were found:



(a) Hamming distance distribution.

(b) Elbow curve.

Figure 16: The 375 sequences that form the C2_2 cluster.

It seems that indeed, C2_2 can be divided again into two sub-clusters which are called C2_2_1 and C2_2_2. As in the two previous cases we compared these two new sub-clusters with S1 and S2 and also with Z1, Z2 and Z3.

- The cluster C2_2_1 is composed by 58 sequences: 1 sequence of S1 y 57 of S2.
- The cluster C2_2_2 is composed by 319 sequences: 309 sequences of S1 y 10 of S2.

Also, we compared C2_1 and C2_2 with Z1 (hipermutated zone), Z2 (germline zone) and Z3 (intermediate zone):

- The cluster C2_2_1 is composed by 58 sequences: 0 sequences of Z1, 52 of Z2 and 6 of Z3.
- The cluster C2_2_2 is composed by 319 sequences: 10 sequences of Z1, 9 of Z2 and 298 of Z3.

Looking at the comparisons with S1 and S2 and with Z1, Z2 and Z3, it is clear that the sequences that make up Z3 have been isolated, thus confirming that they were indeed a well defined sub-cluster. Furthermore, due to its composition, cluster C2_2_2 is identified as an intermediate cluster.

In addition, it should be noted that neither C2_2_1 nor C2_2_2 can be split into further sub-clusters. At this point, the clustering process is finished.
In the following, a summary of the clustering study performed is shown

- The first step was to divide the entire data base of 3050 sequences into two clusters, as this was the most robust solution according to the K-means algorithm. These two clusters were named C1 and C2 and they overlapped with S1 and S2, except for some 300 sequences, later identified as belonging to Z3, which instead of being classified in C1 as hypermutated were being classified as germline, so they were in C2.

- The cluster C1 turned out to be indivisible and was called hypermutated because it was a subset of cluster S1. However, the cluster C2 could be divided into two other sub-clusters: C2_1 and C2_2.

- A further attempt was made to re-split these two new clusters C2_1 and C2_2. In this case C2_1 could not be split and was identified as germline because it was mostly formed by sequences from cluster S2. On the other hand, C2_2 could be split into two other sub-clusters, called C2_2_1 and C2_2_2.

- The two clusters C2_2_1 and C2_2_2 could not be re-divided into sub-clusters. C2_2_2 is called an intermediate cluster because most of its sequences belong to Z3. C2_2_1 is identified as another intermediate cluster but closer to the germline cluster than C2_2_2 since all his sequences belong to Z2 and to S2.

Notice that, as found in [1], when considering all the sequences, the most robust solution was two sub-clusters, as it was shown in the Figure 14. However, by applying an iterative process in which at each step we try to divide again the sub-clusters resulting from the previous step, we have been able to identify several sub-clusters. In this way, the germline cluster, C2_1, and three other clusters, C1, C2_2_1 and C2_2_2, have been identified. These last three, as explained in the Section 2.6, are considered same-time-t_a-clusters, being t_a the time that the sequences of this cluster have been subjected to the maturation process.

3.3.3 The hypermutated cluster as affinity predictor.

In [1] Asti et al found that learning a Multivariate Gaussian Model from their hypermutated cluster S1, to see to which extent the statistical score can be used as a predictor of the affinity of the antibody for its target antigen. As test data set they used 30 antibodies which the neutralization titer has been experimentally assessed.
Since in the Section 3.3.2 we have redefined the hypermutated cluster, we check if this hypermutated cluster C1 is a better predictor of the affinity than the hypermutated cluster S1 defined in [1]. In order to check that, we took from [1] the 30 antibodies sequences that they used as test data

set and calculated their MG scores learning the MG model from three data sets: S1, C1 and C1 learning the prior distribution from the germline cluster C2_1. Notice that, as it was mentioned in the Section 2.3, the portion of prior distribution considered is determined by λ. That parameter optimal value is taken from [4] and it is 0.1. The results obtained can be found in the Table 2.

	MG score		
Sequence id	S1	C1	C1 germ prior
9815	5526	5677	6603
10731	5134	5312	5207
17720	5833	5998	8633
18278	6047	6266	9028
24972	5410	5482	6163
31458	4844	4934	5900
43567	4858	4969	6014
47890	3282	3109	4087
53821	5333	5489	6491
57729	4232	4333	5099
61048	5413	5521	6489
69713	3595	3411	3004
71632	5205	5305	6265
86277	5418	5571	6537
86984	4617	4714	5001
95589	4455	4521	4752
96298	5628	5772	7853
120119	4359	4460	5060
127586	3231	3124	2631
135083	5454	5607	6558
149590	5398	5547	6515
149768	4257	4347	4764
151901	6075	6290	9056
164202	5852	6071	8795
165478	4931	5067	5242
179500	5060	5072	4961
186275	5089	5220	6276
186640	5689	5852	8376
195462	4244	4231	4657
196147	4951	5047	5845

Table 2: MG scores of the 30 test sequences taken from [1] learning the MG Model from S1, C1 and C1 with a prior distribution inferred from the germline cluster.

With the data presented in the Table 2, the average score and its dispersion in each case are calculated in Table 3.

	S1	C1	C1 germ prior
Average MG score	4981 ± 742	5072 ± 866	6028 ± 1644

Table 3: Average MG scores calculated from the data exposed in the Table 2.

As it can be seen in the Table 3, our hypermutated cluster obtain a higher average MG score than the hypermutated cluster S1 taken from [1] when considering this 30 sequences test data set. Also notice that since the C1 cluster is a subset of the S1 cluster, as it was mentioned in the previous Section 3.3.2, the average score is similar in both cases. However, when a more realistic prior distribution is used, the average MG score grow up, clearly outperforming the two other cases.

3.3.4 Ten position alternative clustering analysis.

In the Section 2.6 it was mentioned that, in order to obtain the attractors of the dynamics, it is necessary to find the optimal value of the relaxation matrix α and of Δt_j being $j = 1, ..., N_{cl}$ the number of clusters considered in the process, as it can be seen in the equation (33). These optimal values were obtained by maximizing the equation (32) which, in spite of assuming the diagonal alpha matrix by boxes and symmetrical, implies a high computational weight. For this reason, in order to reduce the computational weight, it was decided to carry out the modified Ornstein-Uhlenbeck process considering only ten positions of the sequences and not all 149.

To select these 10 positions of the sequences, the Kullback-Leibler divergence will be used[7]. This divergence is a measure of how one probability distribution, $p_i(a)$, is different from a second reference probability distribution, $q_i(a)$. In this case, $q_i(a)$ represents the probability of appearance, inferred from germline base data C2_2, of an a amino acid at position *i-th* and $p_i(a)$ will be equivalent inferred from the hypermutated cluster C1 or from the intermediate cluster C2_1.
Once that $q_i(a)$ and $p_i(a)$ have been defined, the Kullback-Leibler curve follows the equation (35). The positions selected will be those that maximize this Kullback-Leibler curve:

$$KL_i = \sum_{aa=1}^{21} p_i(a) log(p_i(a)/q_i(a)) \quad i = 1, ..., 149 \tag{35}$$

In this equation it is observed that if $q_i(a) = 0$ and $p_i(a)$ does not cancel, this curve presents a divergence. This would represent that the amino acid aa does not appear in the position *i-th* in the germline data set but it does in the data set from which it is inferred $p_i(a)$. This case is very similar to the one studied in Section 3.2.3 in which it was found that there were null sequences in the IGoR data set that were not null in the S1 cluster. To avoid divergence-related problems, if $q_i(a) = 0$ but $p_i(a)$ does not cancel out, a value of:

$$q_i(a) = 1/n_{germ} \tag{36}$$

[7]The python code to calculate the Kullback-Leibler divergence between two data sets can be found at Annex 6.8

will be assumed, where n_{germ} represents the number of sequences in the germline data base C2_1. In this way, following the equation (35) the Figure 17 is obtained.

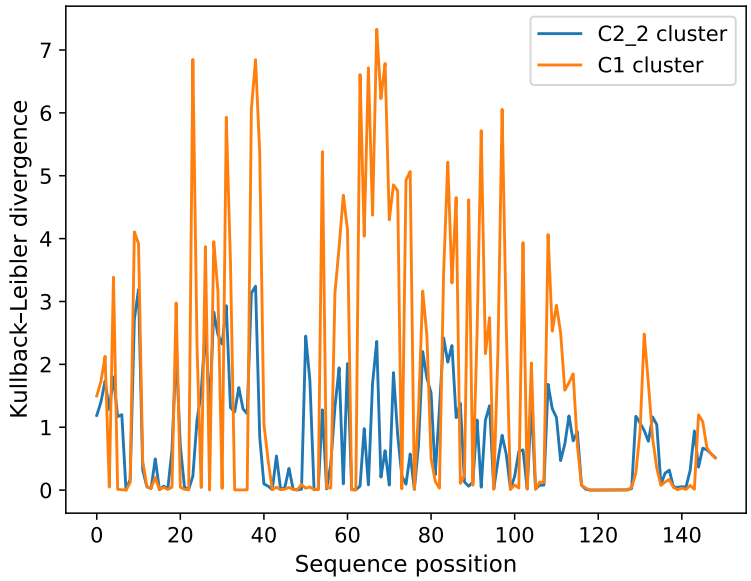


Figure 17: Kullback-Leibler divergence considering the 149 sequence positions.

It can be seen in Figure 17 that there is an area from position 63 to 73 that has a Kullback-Leibler divergence. The Figure 17 can be seen enlarged centered on these positions in the Figure 18.

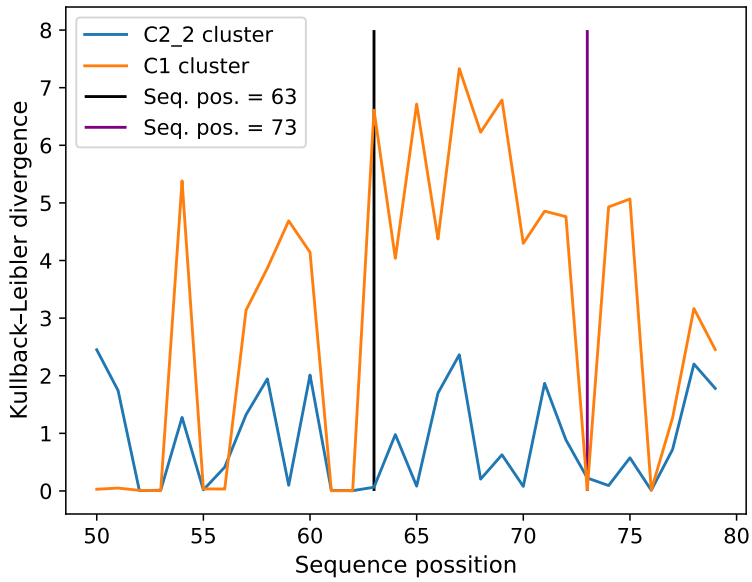
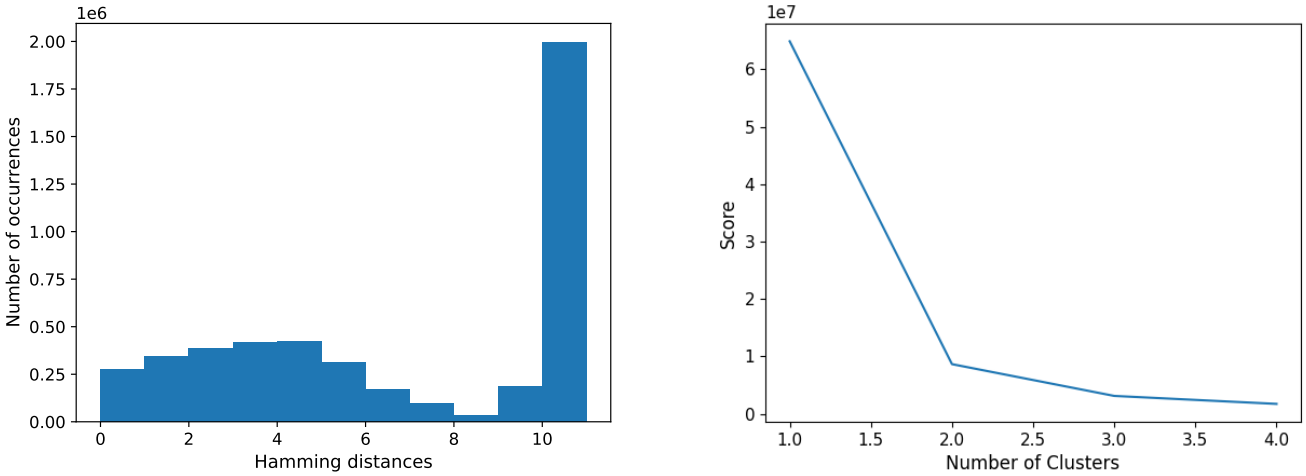


Figure 18: Kullback-Leibler divergence considering only position from 63 to 73.

It is observed that in these selected positions, from 63 to 73, not only the KullBack-Leibler divergence of cluster C_1 is high, but cluster C2_2 also shows some divergence. This will be useful, as it indicates that in these positions there are differences between the germline cluster and the cluster C2_2, which is supposed to be intermediate between C2_1 and C1.

Now, a clustering process similar to the one performed in the Section 3.3.2 section is performed considering only these 10 positions selected. The objective is to find sub-clusters with which to perform the modified OU process while checking if the splitting of these sub-clusters is similar to the one obtained in the Section 3.3.2, which would be an indication that the columns chosen using the Kullback-Leibler divergence are the correct ones.

As in section Section 3.3.2, the starting point are the 3212 sequences obtained from ASTI, of which ANARCI is only able to realign 3050, which are the ones that are finally considered. Thus, the k-means algorithm is applied to these 3050 sequences, obtaining the results shown in Figure 19.



(a) Hamming distance distribution.

(b) Elbow curve.

Figure 19: All 3050 sequences obtained from [1] and realigned by ANARCI, but considering just the sequence positions from 63 to 73.

In Figure 19 it is observed that, as in Section 3.3.2, the set of 3050 sequences can be divided into two sub-clusters, called CC1 and CC2. Once these two sub-clusters have been obtained, studied which sequences make up each of them.
First they will be compared with the clusters C1 and C2 obtained in the Section 3.3.2:

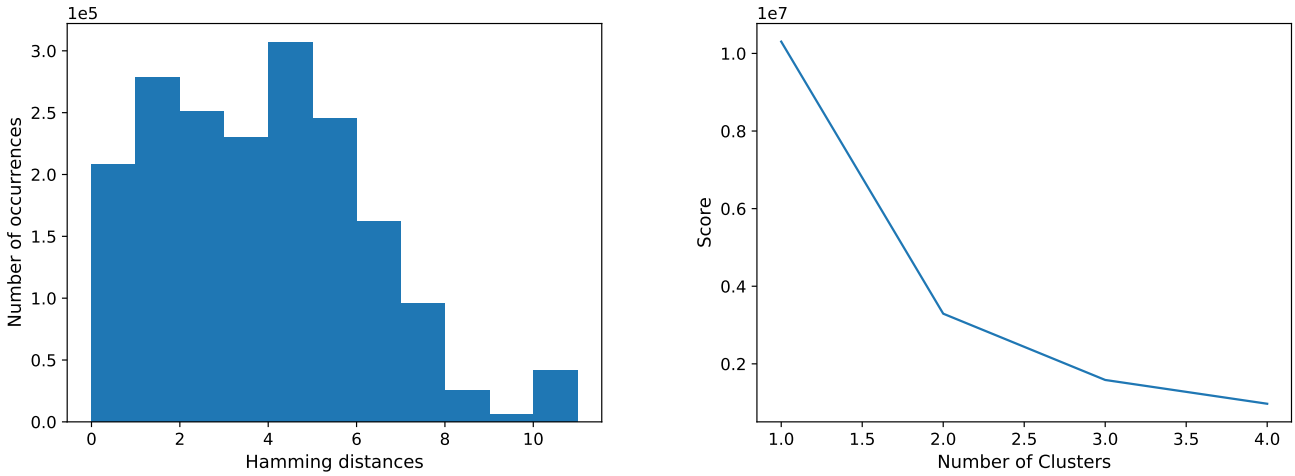
- The cluster CC1 is composed by 1118 sequences: 1109 sequences of C_1 and 9 of C_2.
- The cluster CC2 is composed by 1932 sequences: 3 sequences of C_1 and 1929 of C_2.

and with the S1 and S2 clusters:

- The cluster CC1 is composed by 1118 sequences: 1118 sequences of S_1 and 0 of S_2.
- The cluster CC2 is composed by 1932 sequences: 311 sequences of S_1 and 1621 of S_2.

It is observed that the distribution of the sequences is practically the same as that obtained in the Section 3.3.2, when the complete sequences were considered. On the other hand, from the comparison with S1 and S2, it is observed that CC1 only contains sequences from the S1 file and CC2 contains a mixture of S1 and S2, although S2 predominates. Thus, it is concluded that this first division in two clusters of the 3050 sequences, but only considering positions from 63 to 73, is practically identical to the one achieved in Section 3.3.2, when all the positions of the sequences were considered.

The next step is to check whether CC1 and CC2 can be split again into more sub-clusters, as was the case in the Section 3.3.2 section. To do so, the k-means algorithm is applied to them. In this case, it is observed that cluster CC1 cannot be re-split. Considering that it is practically a subset of C1, CC1 is the cluster that will be considered as hypermutated in the modified OU process. On the other hand, in the case of the cluster CC2 the Figure 20 is obtained.



(a) Hamming distance distribution.

(b) Elbow curve.

Figure 20: K-Means applied to the 1932 sequences of CC2.

It is observed in this Figure 20 that although the distribution of Hamming distances in this case is more homogeneous than in the case of C2, the elbow curve indicates that CC2 can be divided into two sub-clusters, called CC2_1 and CC2_2.

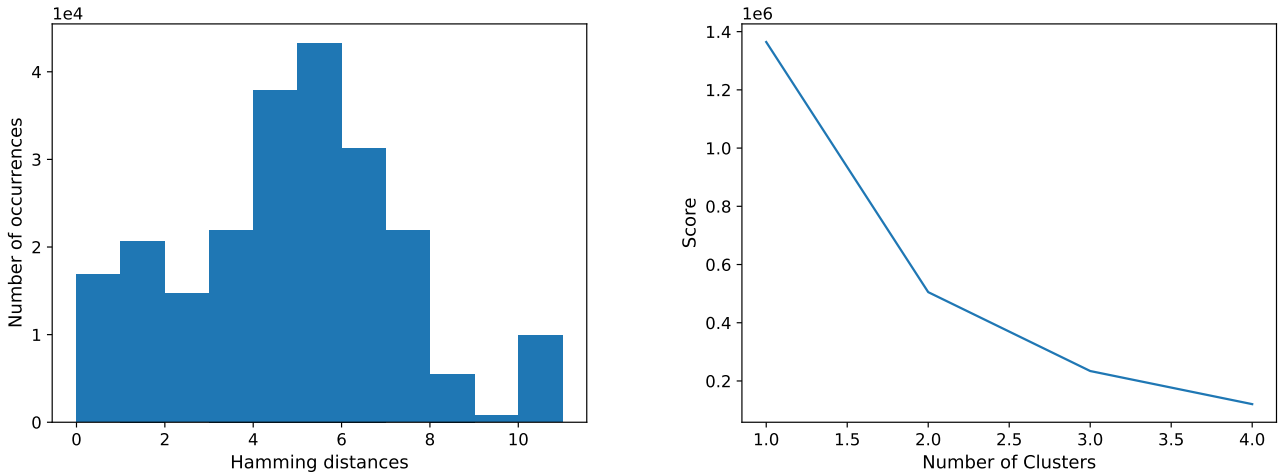
First they are compared with the clusters C2_1 and C2_2 obtained in the Section 3.3.2, so the following results are obtained:

- The cluster CC2_1 is composed by 1261 sequences: 1246 sequences of C2_1 and 15 of C2_2.
- The cluster CC2_2 is composed by 671 sequences: 319 sequences of C2_1 and 352 of C2_2.

and with the S1 and S2 clusters:

- The cluster CC2_1 is composed by 1261 sequences: 1 sequence of S1 and 1260 of S2.
- The cluster CC2_2 is composed by 671 sequences: 304 sequences of S1 and 367 of S2.

It is deduced that CC2_1 is almost entirely made up of sequences from the C2_1 cluster that are also part of S2. However, CC2_2 is formed by an almost homogeneous mixture of sequences from C2_1 and C2_2, which also appears when compared to S1 and S2. If we try again to split both clusters, CC2_1 and CC2_2 into more sub-clusters, what we found is that CC2_1 cannot be divided into further sub-clusters while CC2_2 can. Since CC2_1 is mostly made up of sequences from cluster C2_1, which was referred to as a germline cluster in section Section 3.3.2 when considering the complete sequences, we consider this CC2 cluster as the germline for the modified OU process. On the other hand, CC2 does seem to be able to be split into two other sub-clusters, as shown in the figure Figure 21.



(a) Hamming distance distribution.

(b) Elbow curve.

Figure 21: K-Means applied to the 1932 sequences of CC2.

Following the same procedure that in the previous cases, the cluster CC2_2 is splitted into two sub-clusters, called CC2_2_1 and CC2_2_2, and again both sub-clusters are compared with the C2_1 and C2_2 clusters.

- The cluster CC2_2_1 is composed by 270 sequences: 232 sequences of C2_1 and 38 of C2_2.
- The cluster CC2_2_2 is composed by 401 sequences: 87 sequences of C2_1 and 314 of C2_2

Also, we compare them with the S1 and S2 clusters:

- The cluster CC2_2_1 is composed by 270 sequences: 2 sequences of S1 and 268 of S2

- The cluster CC2_2_2 is composed by 401 sequences: 302 sequences of S1 and 99 of S2

We observe that by splitting the cluster CC2_2, which was formed by sequences of C2_1 and C2_2, we obtain two sub-clusters, CC2_2_1 and CC2_2_2, which form roughly a subset of C2_1 and C2_2, respectively. However, if we look at the comparison with S1 and S2, we see that we have the opposite case to the comparison with C2_1 and C2_2, as CC2_2_1 is mostly formed by sequences from S2 and CC2_2_2 by sequences from S1. Given this situation, we compare the composition of these last two clusters CC2_2_1 and CC2_2_2 with the high density zones found in Figure 7 of [1] and which were denominated in the previous Section 3.3 as: Z1 (hypermutated), Z2 (germline) and Z3 (intermediate).

- The cluster CC2_2_1 is composed by by 270 sequences: 1 sequence of Z1, 259 of Z2 and 10 of Z3.
- The cluster CC2_2_2 is composed by by 401 sequences: 11 sequences of Z1 and 96 of Z2 and 294 of Z3.

Notice that we have not achieved exactly the same separation as in Section 3.3.2 since C2_2_1 was more numerous than CC2_2_1. However both clusters are formed by sequences belonging to the germline zone Z2 and also to cluster S2. On the other hand, the composition of the CC2_2_2 cluster appears to be the same as that of C2_2_2 with some added sequences from the S2 germline cluster.
In this way, considering only ten positions of the sequences, the division into sub-clusters is really similar to the one obtained in the Section 3.3.2 where we considered the whole sequence length. The only really big difference is the magnitude of cluster CC2_2_1, which although it is entirely formed by sequences belonging to S2 and Z2 as in the case of C2_2_1 contains more sequences transposed from the germline cluster. This indicates that the role of the 10 positions chosen following the KL divergence criterion is relevant in determining which cluster each sequence belongs to, which validates the KL criterion.
Once the clustering process has been finished we can conclude that, considering these ten sequence positions, four sub-clusters can be identified starting from the 3050 sequences taken from [1]. These four sub-clusters are used in the OU modified process as explained in the Section 2.6

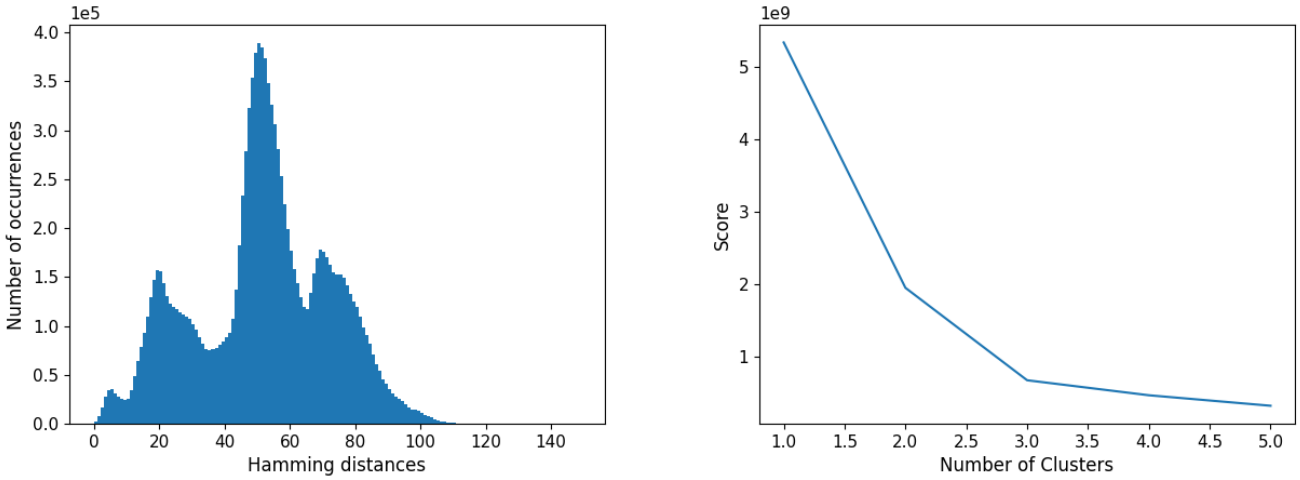
3.4 Ornstein−Uhlenbeck

In this section the results obtained by performing the Ornstein-Uhlenbeck process described in the Section 2.6 are presented.
To perform this OU experiment the sub-clusters obtained in the Section 3.3.4 are used. Specifically, the sub-cluster CC2_1 will be considered as the germline cluster, so it is used to , together with the data inferred from an IGoR data set, to estimate the initial state parameters following the equation (18). The other three sub-clusters: CC1, CC2_2_1 and CC2_2_2 are used to infer the temporally linked gaussian distributions.

3.4.1 Initial state of the Ornstein Uhlenbeck process

As it was mentioned in the Section 2.6, the initial state is inferred from the sub-cluster denoted in the Section 3.3.4 as germline, the cluster CC_2_1 , and a data base generated by IGoR following the procedure explained in Section 2.4.

The first step is to check whether the IGoR data base is composed of a single cluster or several sub-clusters. For this, we use the K-Means algorithm applied to this base data. The results obtained are shown in the Figure 22.



(a) Hamming distance distribution.

(b) Elbow curve.

Figure 22: Results of applying the K-Means algorithm to a fifty thousand sequences data base generated by IGoR.

Looking at the Hamming distances distribution it seems clear that the IGoR data base is composed by more than a unique cluster. Specifically, the elbow curve seems to indicate that there are three sub-clusters. That is why, before using all the data base sequences to infer the a priori distribution of the initial state following the equation (18), it must be checked if the sequences that form each of the three clusters have the same compatibility with the germline cluster. To check that, the MG score defined in (11) is used.
The procedure consists of separating the IGoR base data sequences into three sub-clusters and inferring the Gaussian model with each of them. For each case, the average score of the sequences forming the germline cluster CC2_1 is calculated. The IGoR cluster with the highest compatibility with the germline cluster is the one used to infer the a priori distribution of the initial state. Thus, it is found that when the Gaussian model is inferred from the cluster whose centroid is 71, the average MG score of the germline cluster sequences C2_2_1 is the highest, followed by the case in which the Gaussian model is inferred from the cluster whose centroid is 51.

The next step is to generate an IGoR data base large enough so that when extracting the sub-cluster used to infer the prior probability distribution of the initial state, we obtain a data set large enough to provide reliable results. In Section 3.2 the optimum size of the data set was discussed, resulting that it should be about five hundred thousand sequences. Once that this data set is generated we infer the initial state following equation (18). Also, notice that the parameters ν_0 and γ_0 that appear in the equation (20) are set both as the number of sequences that form the IGoR sub-cluster used to infer the prior.

3.4.2 Alpha minimization

Once that the initial state of the OU process and the different clusters to be used have been found, the objective, as mentioned in the OU, is to estimate the attractors of the dynamics θ and ω, which follow the equation (33) and (34). For this purpose, we must find the relaxation matrix α and the different Δt_j that maximize the equation (26) knowing that the maximum of a NIW distribution associated to each 'snapshot' follow the equation (32).
Since the only term in the equation (32) that depends on α and Δt_j is the determinant of ψ'_j, the procedure to maximize the (26) is to minimize that ψ'_j determinant[8]. Notice that the ψ'_j determinant and the rest of the equation (26) also depends on γ_j and on ν_j, which are related to each other by the equation (29). We considered just one unique γ, instead of three γ_j, and defined it as a parameter to be optimized together with α and the different Δt_j.

Although only ten sequence positions were considered, the computational cost is huge, even restricting ourselves to the block diagonal matrix mentioned in Section 2.6, that decouples mutations at different positions: we have 2100 independent elements in matrix α to optimize, plus the values of Δt_j for the clusters, and the parameter γ. Actually, since we set $\Delta t_0 = 0$ by definition, and we have no independently defined time scale, so that we can set $\Delta t = 1$ for the hypermutated cluster, and end with 2103 variables. In our attempts of optimization, the results obtained for the different Δt_j did follow the expected order, that is:

$$0 = \Delta t_0 < \Delta t_{CC2\_2\_1} < \Delta t_{C2\_2\_2} < \Delta t_{CC1} = 1 \tag{37}$$

that makes sense, since, if we look at the clustering process performed in the Section 3.3.4, we see that CC2_2_1 is formed by sequences that belonged to both C2 and S2, while CC2_2_2 was formed by sequences that also belonged to C2, although they did not belong to S2, but to S1. Both were supposed to be a temporary intermediate clusters between C1 and C2_1, but CC2_2_1 closer the the germline cluster than CC2_2_2. Despite this satisfactory result, the optimization of alpha was not possible, because the number of variables was too high, yielding a strong dependence on the initial condition.

[8]The python code to minimize the ψ' determinant can be found at Annex 6.9

In order to avoid a this problem, the relaxation matrix α was set to diagonal matrix, instead of bloc-diagonal as in the former case. With this approximation the number of variables was reduced to 203. In this simplified approach, the order of the Δt_j continued following the (37), as in the previous approach. However again we could not estimate a unique α matrix because we were not able to find the global minimum minimum since the result depended strongly on the initial conditions. On top of this, we noticed that, when trying to calculate the ω and θ attractors with different α corresponding to different initial conditions, the results did not make physical sense, since the attractors should have all their elements less than (or close to) one, to be reconverted to amino acid sequences, but the results where order of magnitudes away from that, since we have not imposed any constraints on ω and θ to fulfill this condition. As conclusion, although the order in which the different Δt_j obtained for both approaches are ordered as optimal is as expected, we have not been able to find a unique alpha matrix that maximizes the equation (26) and at the same time the attractors obtained from it have a physical sense.

4 Conclusions

In this work, we have attempted to improve the scoring function described in [1] as a predictor of the affinity of an antibody to the antigen given by the gp120 capsid protein HIV-1 virus, by taking a more realistic a priori distribution and reinterpreting the germline and the hypermutated clusters as clusters linked by a temporal evolution, instead of as independent ones. This temporal evolution was assumed to be described by an Ornstein-Uhlenbeck process, even if this assumption in more related to the fact that OU is possibly the simplest dynamical process which is intrinsically Gaussian, and that leads asymptotically to an attractor (which is something that one can expect when searching the sequence space for the sequences with the highest affinity to a target), rather than to a choice related to the details of the mutation and selection process.

First, in the Section 3.2.3 we compared a fully maturated data set, the hypermutated cluster taken from [1] and a data set composed of naive sequences generated by IGoR. Using the binary transformation, explained in the Section 2.2 and taken from [6], we saw that the data hypermutated data set is more specific than the naive sequences. Also we performed several test to confirm that the maturation process implies a sequence specification process, leading from a broader to a narrower variety of amino acids types at each position.

Then, in the Section 3.3 reinterpretation of the hypermutated (S1) and germline (S2) clusters taken from [1] was done. First, we saw that although the most robust solution according to K-Means algorithm was 2 clusters, S1 and S2, the cluster S1 could be re-splitted again into two other clusters. Furthermore, when we compared these three clusters with the high density zones shown in the Figure 12 (Z1, Z2 and Z3) we observed an almost perfect correlation. Then, we performed our own clustering analysis starting from the data base taken from [1] but realigned using ANARCI. We found four sub-clusters, which were identified based on their composition. In that way we found a hypermutated cluster (C1), a germline cluster (C2_1) and two intermediate clusters C2_2_1 and C2_2_1.

It is important to notice that our clustering (together with the choice of the sequence alignment scheme) outperforms that in [1], in the sense that the score of the test data set taken from [1], and composed of 30 antibodies of high affinity to gp120 (as measured by their IC50 value), is higher when considering a Multivariate Gaussian Distribution inferred from our C1 cluster (with uniform prior), than the one obtained from the S1 cluster in [1], again with uniform prior. The results are even more remarkable in the case we use the dataset C1 together with a prior obtained from the germline dataset.

In our attempt to relate the different clusters within a unique evolutionary trajectory (during the maturation phase) connecting the germline to the hypermutated cluster through the intermediate

ones, in Section 3.3.4 we repeated the clustering process but only considering these ten sequence position. We found mostly the same four clusters than in the Section 3.3 expecting some sequences that were transferred from the germline cluster to one of the intermediates. This confirms that the ten sequences positions are relevant in the maturation process and validates the result obtained in the Section 3.3.4.

Once that these sub-clusters were obtained, we saw in the Section 3.4 the Ornstein-Uhlenbeck process. First, in the section Section 3.1, we checked that the discretization of the sequences implied a lost of information. Then, in the Section 3.4.1 the we defined the initial state of the OU process as a combination of the germline cluster and a data set generated by IGoR following the equation (18). To ensure the reliability of the results, we generated a IGoR data set, following the procedure explained in the Section 2.4, large enough based on the result obtained in the Section 2.5. Then we tried to estimate the relaxation matrix α and the different maturation times Δt_j of each sub-cluster considered in the OU process. First we tried assuming α to be symmetrical and block diagonal, however although the times distribution of the clusters were coherent, we could not estimate α reliablydue to the high number of variables to optimize. Then we tried assuming a full diagonal α matrix. Again, we found a coherent Δt_j results, however the resulting α matrix depended on the initial conditions and the attractors calculated from it had no physical sense.

In the future, starting from the results found in this project, several refinements can be done. The main problem here, preventing the OU approach from succeeding was the requirement of the optimization of the relaxation matrix, to find the attractors and also the lack of constrains on the attractors themselves, to ensure a meaningful result. In this sense, a procedure to estimate the attractors without minimizing α will be essential. Notice that this totally different approach is needed since we can not simplify the α matrix more than we have done the α optimization. Also, it would be important to connect the α matrix to the detailed microscopic mechanisms of random mutation and selection that lead from the naive sequences to the high-affinity, hypermutated ones.

5 Bibliography

References

[1] Asti L, Uguzzoni G, Marcatili P, Pagnani A. Maximum-Entropy Models of Sequenced Immune Repertoires Predict Antigen-Antibody Affinity. PLoS Comput Biol 12(4): e1004870. (2016) doi:10.1371/journal.pcbi.1004870

[2] Alberts B, Johnson A, Lewis J, et al. Molecular Biology of the Cell. 4th edition. New York: Garland Science; 2002. The Generation of Antibody Diversity. Available from: https://www.ncbi.nlm.nih.gov/books/NBK26860/

[3] Marcou, Q., Mora, T. Walczak, A.M. High-throughput immune repertoire analysis with IGoR. Nat Commun 9, 561 (2018). doi:10.1038/s41467-018-02832-w

[4] Clavero-Alvarez, A., Di Mambro, T., Perez-Gaviro, S. et al. Humanization of Antibodies using a Statistical Inference Approach. Sci Rep 8, 14820. (2018) doi.10.1038/s41598-018-32986-y

[5] Pilar Calvo Príncipe, Refinement of a statistical model for antibody humanization, TFM.(September 2019)

[6] Baldassi C, Zamparo M, Feinauer C, Procaccini A, Zecchina R, Weigt M, et al. Fast and Accurate Multivariate Gaussian Modeling of Protein Families: Pre-dicting Residue Contacts and Protein-Interaction Partners. PLoS ONE 9(3): e92721. (2014) doi:10.1371/journal.pone.0092721

[7] Dunbar J, Deane CM. ANARCI: antigen receptor numbering and receptor classification. Bioinformatics. 2016 Jan 15;32(2):298-300. Epub 2015 Sep 30. PMID: 26424857; PMCID: PMC4708101. doi: 10.1093/bioinformatics/btv552

[8] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.K-means.html

[9] David Luna Cerralbo, Validation and refinement of a statistical model for antibody classification and humanization, TFG.(June 2020)

[10] https://biopython.org/docs/1.75/api/Bio.Seq.html

[11] https://www.investigacionyciencia.es/blogs/medicina-y-biologia/28/posts/ste-es-el-rbol-de-tu-vida-10631

[12] Giudicelli, V., Duroux P., Ginestoux C., Folch G., Jabado-Michaloud J., Chaume D., Lefranc M.-P., Nucleic Acids Res., 34:D781-D784 (2006).

[13] Madden T. The BLAST Sequence Analysis Tool. 2002 Oct 9 [Updated 2003 Aug 13]. In: McEntyre J, Ostell J, editors. The NCBI Handbook [Internet]. Bethesda (MD): National Center for Biotechnology Information (US); 2002