Pattern Recognition Letters 74 (2016) 24–29

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Using a novel clumpiness measure to unite data with metadata: Finding common sequence patterns in immune receptor germline V genes☆

Gregory W. Schwartz[a], Ali Shokoufandeh[b], Santiago Ontañón[b], Uri Hershberg[a,c,*]

[a] Department of Biomedical Engineering, Science & Health Systems, Drexel University, 3141 Chestnut Street, Philadelphia, PA 19104, USA
[b] Department of Computer Science, Drexel University, 3141 Chestnut Street, Philadelphia, PA 19104, USA
[c] Department of Microbiology and Immunology, Drexel University College of Medicine, 2900 W. Queen Lane, Philadelphia, PA 19129, USA

ARTICLE INFO

Article history:
Received 20 June 2015
Available online 8 February 2016

Keywords:
Hierarchical clustering
Aggregation
Tree analysis
Immune receptor repertoire
Adaptive immunity
Multiscale analysis

ABSTRACT

When finding relationships in biological systems, we often describe hierarchies based on one facet of the data. However, when using this hierarchy to elucidate relationships between metadata, the distribution of metadata labels within the hierarchy may exhibit different levels of aggregation—uniform, random, or clumped. As of now, there exists no measure for finding the level of aggregation, or "clumpiness", between labels distributed among the leaves of a hierarchical container. We propose a clumpiness measure to aid in the quantification of relationships between metadata. We validated our measure with random trees and found that the measure is resistant to changes in the tree size, label size, and the number of types of labels, compared to the closest alternative measures. We used our clumpiness measure to quantify the relationships between light and heavy chains in human and mouse B cell and T cell receptor V genes based on their motifs. We found that the B cell heavy chains were the most aggregated while the T cell chains were the least aggregated and that the IGL chain was clumped the most with the T cell chains out of all of the B cell chains.

1. Introduction

Biological systems are often described through hierarchical relationships of different labels. Due to the complex nature of biological systems, we often describe these hierarchies using only a subset of the available information about each element in the dendrogram. For instance, we can create a phylogeny of different species based on their genes while at the same time retaining other metadata, or labels, about their behavior, survival, and phenotypes. However, unlike the gene data, these metadata labels may be distributed randomly, uniformly, or clumped throughout the hierarchical structure. In this paper, we present a novel measure to quantify the extent that a hierarchical structuring of data describes a relationship of aggregation, or "clumpiness", between the metadata labels with which its components are categorized. In this fashion, we can unite the two levels of the structure—the information from the data and the categorical information from the metadata.

Let us consider the adaptive immune system as a general example of a multi-scale biological system. This system is comprised of several repertoires of immune cells with individual receptors of unique specificity for different antigens in the environment. In order to cover a wide range of antigens, the body generates a vast and diverse pool of differently responding cells called the immune repertoire of the body. Under specific conditions, these antigens can trigger competitive proliferation, mutation, and death in only a subset of the cells. The successful recognition of an antigen by a cell's receptor leads to the cell dividing and producing its own lineage of cells responding to similar antigens. The resulting hierarchical structure is associated with metadata labels such as the tissue where one of the descendant cells is found, the function of that cell in the immune response (such as an effector cell or a memory cell), or its fate—death or division. Because the metadata labels are of a different scale than the data (in this scenario the pattern of mutations in a given cell), it is possible to have the labels widely dispersed in the container (here a hierarchical data structure) but be close together in small clumps as opposed to

☆ This paper has been recommended for acceptance by Qian Xiaoning.

* Corresponding author at: Department of Biomedical Engineering, Science & Health Systems, Drexel University, 3141 Chestnut Street, Philadelphia, PA 19104, USA. Tel.: +1 215 895 1698, fax: +1 215 895 4983.
E-mail addresses: gregory.schwartz@drexel.edu (G.W. Schwartz), uri.hershberg@drexel.edu (U. Hershberg).

http://dx.doi.org/10.1016/j.patrec.2016.01.011
0167-8655/

being randomly or uniformly distributed. These scenarios appear throughout the biological domain.

Another hierarchical container of data we may use, in order to capture the different possible "behaviors" of the cells, is to cluster the cells by their common gene expression patterns [1]. In this case, the labels describe a common progenitor (ancestor) cell or varying levels of mutation in its DNA. Finally, we will look specifically in this paper at the hierarchical clustering of sequence fragments. Both in our example and in other biological examples, it is commonly considered that motifs can be indicative of binding capability and interaction potential. In this case, we hypothesize that a group of cells with similar behaviors are motivated by a subset of common motifs with related structures. As binding of the receptor and survival of the cell depend on the receptor structure, in order to look at the relationships between the different parts of the receptors we need to account for the relationship between sequence fragments. As such, we must find the relationship (container) between each region of the receptor (data) before quantifying the overall distribution and degree of aggregation of chains (label).

As shown in these examples, although the data is clustered together as the result of the tree, we can ask additional questions about the relationship between the metadata labels within the tree. More specifically, we would like to quantify the degree of aggregation, or "clumpiness", between the labels by using the structure of the tree generated by the pairwise relationship of the data. While there is a wide range of metrics to measure aggregation, they are focused on the spatial distribution in two dimensions [2–9]. As of now, there exists no measure for the quantification of aggregation in a distribution within hierarchical trees. Furthermore, previous studies attempting to find patterns between metadata in hierarchical structures are based on grouping similar sections of the container rather than finding the impact of dispersion on the metadata and are heavily focused on visualization [10–13]. In this paper, we will demonstrate the power of such an analysis by focusing on the last example, where we can find the relationship between immune receptors by their sequence fragments.

In order to look at the distribution of labels, we need new tools. We propose our clumpiness measure as a way to measure the degree of aggregation between labels in a hierarchical container. Our measure is robust to the container size, data size, and label size, and thus is scale invariant. In addition, our measure is generalizable to more than two labels and is efficiently computable and maintainable. In this paper, we will (1) describe the measure, (2) show the generalization, (3) demonstrate the use of the measure to find the relationship between receptor chains, and (4) quantify the response of the measure to noise and sizes.

2. Notations and definitions

Suppose we have a rooted binary tree with a set of vertices V. Let us now call $I \subseteq V$ the set of non-leaf and non-root vertices of the tree, and $T \subseteq V$ all of the leaf vertices whose parent is in I (thus, this includes all the leaf vertices, except the leaves that are children of the root, since the root is not in I). Now, let us assume $M \subseteq T$ to be the subset of leaves of interest, our "relevant" leaves, and $L = \{L_1, L_2, \ldots, L_n\}$ to be a partition of M (i.e., $M = \bigcup_{i=1}^{n} L_n$). This partition can represent, for example, a set of labels that we care about in our application domain. We call these labels "relevant" as they contain our relevant leaves. We can now transform the data from our domain into a hierarchical container.

We specifically focus on the domain of immunology in this study. The B and T cells are white blood cells with cell surface receptors, the B cell receptor (BCR) and T cell receptor (TCR) respectively, that bind to antigen which can invoke an immune response. These receptors are quite diverse and each B and T cell express just one type of this receptor. The BCR is composed of a heavy chain (IGH) and a light chain (either IGK or IGL), while the analagous chains on the TCR are the β chain (TRB) and the α chain (TRA). As we want to compose our hierarchical container from structural units, we use subregions of these receptor genes in our clustering.

These subregions, we call "protein fragments", are 20 amino acid long sequences taken from an overlapping sliding window across an amino acid receptor sequence. Then our hierarchical clustering generates clusters that are each a group of protein fragments with similar sequences (further explained in Section 5.1). The leaves in the hierarchical container represent these clusters, where each parent contains the union of the children's protein fragments. In this way we have completed the transformation of the data into a hierarchical container of relationships.

3. Clumpiness measure

3.1. Definition

The clumpiness of the set of leaves M when partitioned according to L in a k-ary tree is defined as

$$C(L) = \frac{1}{n}\left(\prod_{i=1}^{n} \frac{x}{y_i}\right)^{1/n} \tag{1}$$

That is, the geometric mean of x weighted by the frequency of each label, y_i. The result is set between 0 and approximately 1 by normalizing by the total number of labels, n. The numerator x is intuitively the weighted number of viable vertices in I weighted by y_i, resulting in

$$x = \frac{1}{|I|}\sum_{v \in I} \delta(v) w(v) \tag{2}$$

$$y_i = \frac{|L_i|}{|T|} \tag{3}$$

We say that a non-root vertex v is "viable" if $\delta(v) = 1$, meaning that v has at least one vertex of each label in its descendant leaves. So,

$$\delta(v) = \begin{cases} 0 & : \bigvee_{i=1}^{n} |D(v) \cap L_i| = 0 \\ 1 & : \text{otherwise} \end{cases} \tag{4}$$

where $D(v)$ is the set of descendant leaves of vertex v contained in M, our relevant leaves. We then weigh the vertex if it is viable by the number of vertices of each relevant label and how far away they are from the vertex in question

$$w(v) = \sum_{i \in D(v)} \left(\prod_{j \in E(v,i)} \frac{1}{c(j)} \right), \tag{5}$$

where $E(v, i)$ is the set of vertices on the shortest path from (and including) v to (but not including) the relevant leaf i and $c(j)$ is the number of children of j. We weigh by the number of children as we want the maximum value of our vertex of interest to be 1, so we keep dividing the values of the descendant vertices based on branching.

If we want to find the clumpiness of a label L_i with itself we need to change our approach: the more clumpy L_i is with other labels, by definition the less clumpy L_i is with itself. Using this property, we can then have L contain two sets—those leaves in L_i and all other leaves. Then the clumpiness of L_i with itself becomes $1 - C(L)$. For the sake of simplicity, we will focus on the case of a rooted full binary tree containing 2 labels.

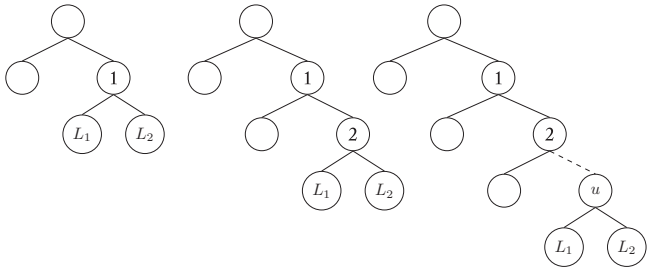

Fig. 1. Example for the case of finding the maximum $C(L)$ through the minimum y_i. The trees grow from the rightmost vertex, with the parent of the rightmost vertex being labeled u.

3.2. Exclusions

We are excluding two types of vertices in the clumpiness measure calculation. This vertex will always be counted in every tree and every analysis, so it's inclusion is not informative and prevents a value of 0.

We also remove any leaves attached to the root vertex from the analysis. If we were to have the majority of the leaves attached to the root vertex (assuming the tree is not binary), which is highly unlikely in a hierarchical clustering scenario, we would just be manufacturing a smaller and smaller y_i value for no apparent reason, inflating the value of the measure. As such, we remove these vertices from the calculation.

3.3. Bounds

In this section, we will describe the bounds of the clumpiness measure defined above. In order to do so, we will assume a rooted full binary tree with $L = \{L_1, L_2\}$. First, if $0 \leq (x/y_i) \leq n$, then we can see that the clumpiness measure is bounded as $0 \leq C(L) \leq 1$. Thus, let us see the range of x and y_i. $0 \leq x \leq 1$ as $w(v)$ has a maximum value of 1 in a binary tree and so the numerator of x has a maximum value of $|I|$. $0 < y \leq 1$ as $0 < |L_i| \leq |M| \leq |T|$.

Next, assume $x = 0$. If we have no viable inner vertices, then the numerator of x is 0 and $C(L) = 0$. Now assume $x = 1$. If all inner vertices are viable, then $x = 1$. Then the inner vertices nearest to the leaves must also be viable, so each of those inner vertices must have leaves from L_1 and L_2, so $|L_1| = |L_2|$ and thus $y_i = 0.5$, so $(x/y_i) = 2$ and thus $C(L) = 1$. Assume $y_i = 1$. First, we note that $\delta(v)$ is 0 or 1. Furthermore, $0 \leq w(v) \leq 1$. Then the numerator of x is between 0 and $|I|$. Therefore, we find that $0 \leq x \leq 1$. Finally, if $y_i = 1$, then $0 \leq C(L) \leq 1$.

Now, let us assume a minimum y_i. The smallest y_i we could have is $(1/|T|)$, where a label is assigned to only one leaf. Let us consider a binary tree with 5 vertices—2 inner vertices and 3 leaves as in Fig. 1. The non-root inner vertex has two children: one from L_1 and one from L_2. Then $x = 1$, $y_i = (1/2)$, and $C(L) = 1$. Now, when we add on additional vertices, we are increasing $|I|$ by 1, decreasing $|T|$ by 1 and then adding two additional leaves increasing $|T|$ by 2 resulting in a net increase of $|T|$ by 1.

Let us call this growth $u = 1$, where u is the number of inner vertices excluding the root. We can see an example of these trees in Fig. 1. Then $u + 1$ is the number of leaves whose parent is not the root. Then we can rewrite (x/y_i) as

$$\frac{x}{y_i} = \frac{\left(\frac{\sum_{j=0}^{u-1} 2^{-j}}{u}\right)}{\left(\frac{1}{u+1}\right)} \tag{6}$$

$$= \frac{\left(\frac{2-2^{1-u}}{u}\right)}{\left(\frac{1}{u+1}\right)} \tag{7}$$

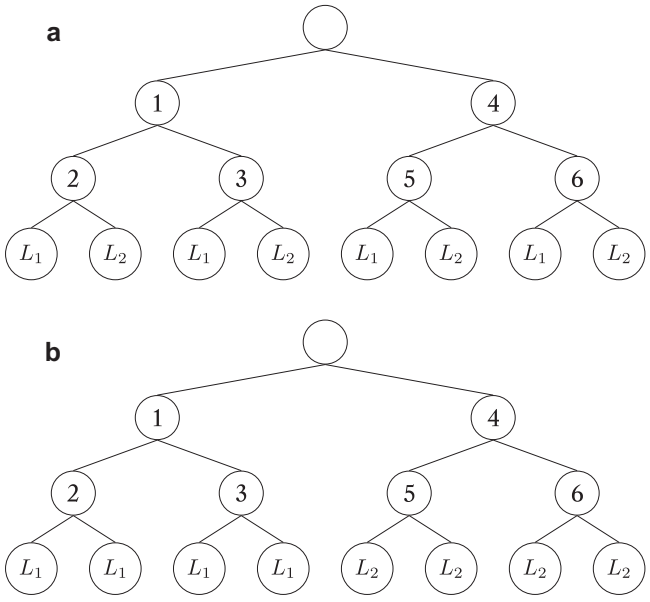

Fig. 2. Example rooted perfect binary trees for use with the clumpiness measure. A rooted perfect binary tree where each leaf is labeled either L_1 or L_2 . (a) Alternating labels of the leaves result in $C(\{L_1, L_2\}) = 1$, $C(\{L_1, L_1\}) = 0$, and $C(\{L_2, L_2\}) = 0$. (b)Like labels grouped on either side of the tree resulting in $C(\{L_1, L_2\}) = 0$, $C(\{L_1, L_1\}) = 1$, and $C(\{L_2, L_2\}) = 1$.

We consider two cases. If we continue to add vertices to the side of the binary tree that is irrelevant to x, as they do not have any associated labels in their leaves, the overall numerator becomes $(1/u)$ and the formula converges to 1 and $C(L)$ converges to 0.5. The maximum value here is at $u = 1$, where we end up with $C(L) = 1$. However, if we were to add on vertices which are relevant to x, we increase $|I|$ and $|T|$ by 1 and we increase the numerator of x as well. Here, $w(v)$ grows based on u as all inner vertices are relevant. The numerator of x is a geometric series that converges to 2 as u approaches infinity. The maximum of this equation occurs at $u = 4$, i.e., $C(L) = 1.17$.

Assigning each label to every other leaf in order to increase x, we see that x increases at a decaying rate while the numerator of y_i increases at a constant rate, so we would never go above the aforementioned value of 1.17, which is the absolute maximum of $C(L)$.

3.4. Example

In order to fully illustrate the process by which the measure works, we here provide an example using two binary trees (Fig. 2). In both trees, we first compute the clumpiness of L_1 with L_2 and, for illustration purposes, we will then compute the clumpiness of L_1 with itself.

We will start with finding the clumpiness between L_1 and L_2. The measure is as follows: in the tree from Fig. 2(a), we ignore the root vertex. Starting at the lower left with vertex 2, we see that it has at least one L_1 and one L_2 in its descendant leaves, so it is a viable vertex. Its value is 1 as vertex 2 has 2 children on the shortest path to 2 relevant leaves, so $w(v_2) = (1/2) + (1/2) = 1$. Then vertices 3, 5, and 6 also have a value of 1 for the same reason. For vertex 1, we note that again it has at least one label of each type in its descendant nodes, but they have a shortest distance of 2 away. Then vertex 1 has 4 relevant descendant leaves, so by getting the number of children at each step we end up with $(1/2)(1/2) + (1/2)(1/2) + (1/2)(1/2) + (1/2)(1/2) = 1$. The same is true for

vertex 4. Then our x in the clumpiness measure is equal to $(6/6) = 1$. Our y_i would be $y_{L_1} = (4/8) = (1/2)$ for L_1 and $y_{L_2} = (4/8) = (1/2)$ for L_2, as both labels take up the same fraction of the leaves. It's also important to note here that we would exclude any leaves attached to the root vertex in this measure as stated in Section 3.1. Then our clumpiness becomes

$$\begin{aligned} C(\{L_1, L_2\}) &= \frac{1}{2}\left(\frac{1}{\frac{1}{2}} + \frac{1}{\frac{1}{2}}\right)^{1/2} \\ &= \frac{1}{2}(4)^{1/2} \\ &= \frac{1}{2}2 \\ &= 1 \end{aligned}$$

If we look at the tree in Fig. 2(b), we see that the only node with descendant leaves from both labels is the root vertex—however, we exclude the root vertex from the measure. As a result, our x value in the measure is 0 and so $C(\{L_1, L_2\}) = 0$.

Now let us look at the clumpiness of L_1 with itself. In Fig. 2(a), we want to find the clumpiness of L_1 with any other label that is not L_1. That is, each label that is not L_1 is lumped together into some generic label, G. Since we only have one other label in this tree, L_2, we will just rename all L_2 to G. Then $C(\{L_1, L_2\}) = C(\{L_1, G\}) = 1$. In order to find the clumpiness of L_1 with itself, we just take $C(\{L_1\}) = 1 - C(\{L_1, G\}) = 0$. For the tree in Fig. 2(b), we have $C(\{L_1\}) = C(\{L_1, G\}) = 1 - C(\{L_1, L_2\}) = 1$.

The two trees in Fig. 2 illustrate the theoretical concept of clumpiness—we would say that the tree in Fig. 2(a) is maximally clumpy with L_1 and L_2 but minimally clumpy with L_1 and itself and vice versa with Fig. 2(b).

There are cases for which the value of clumpiness exceeds 1 (to a maximum of 1.17) as explained in Section 3.3. These cases have extremely biased subtrees expanding in one direction. While the maximum appears in a very small tree, there is still the possibility of having a value greater than 1 if the tree grows in a fashion as seen in Fig. 1.

4. Simulations

In order to verify that the measure measures clumpiness and aggregation of a distribution as we intended, we artificially generated random trees with known clumpiness distributions. In addition, we also compared the clumpiness measure to its closest equivalents using the diversity of each vertex as described below. Last, we observed the degree of noise and consistency in each case in different sized trees.

4.1. Random tree generation

We generated a collection of random trees that simulate a clumpy environment. We made random rooted full binary trees of size $s \pm 10$ vertices. We began from a root vertex and randomly choose either 0 or 2 children, stop if the size of the tree is met, or otherwise recursively iterate this process on the leaves. Trees that were not of the wanted size with the variation were discarded. We labeled the leaves of the trees with L_1 and L_2 (and L_3 where applicable) in the following way: we generated a random list l of the leaves in the tree. Given a neighborhood of size t, we chose the first leaf $v \in l$, assigned the nearest t leaves that label, and subsequently removed v from l, repeating this cycle until all leaves were labeled. We did not label a leaf if the leaf was already labeled. This process eventually gave us a random rooted full binary tree of size $s \pm 10$ with labeled leaves.

4.2. Measure quantifications

We used three measures on these randomly generated trees for sizes 500, 1000, 1500, and 2000 for neighborhood values t from 1 to 20. For each size and neighborhood, we generated three types of trees: (1) trees with approximately even amounts of L_1 and L_2, (2) trees with approximately even amounts of L_1, L_2, and L_3, and finally (3) trees with three times the amount of L_1 compared to L_2 or L_3. We compared our clumpiness measure with two other measures: (1) the arithmetic mean diversity of each inner node in the tree, and (2) the geometric mean diversity of each inner node in the tree [14]. We ran these measures on 100 randomly generated trees of each type, resulting in 24,000 analyses.

Diversity of order 1 [14,15], as used in this paper, is the exponential function of Shannon entropy [16,17]. The entropy of a vertex, in our case, is defined as the Shannon entropy between the two labels in question for the measure as opposed to the traditional entropy of a certain height for hierarchical networks [18]. We can then say that the diversity of a vertex is in [1, 2], as we are finding the diversity only of the relevant leaves, or leaves with labels pertaining to the clumpiness of what we want. The diversity will either tell us that we have 1 type of label, 2 types of labels, or somewhere in between. Then the arithmetic mean of all of the inner vertices should tell us something about the clumpiness between two labels.

Likewise, the geometric mean of the diversity in the tree is using the same geometric mean as our clumpiness measure, where x in this case is the arithmetic mean diversity of the inner nodes. We use this measure to try to normalize the value based on sample sizes.

4.3. Results

For each tree, we performed "inter-clumping" comparisons, where we measured the degree of clumpiness between different pairs of labels, and "intra-clumping" comparisons, where we measured the degree of clumpiness for a label with itself. We compared our measure with diversity by applying the arithmetic mean diversity and the geometric mean diversity of each vertex of each tree (Fig. 3). For all trees of all sizes, the three measures produced higher values for the intra-clumping comparisons and lower values for the inter-clumping comparison of L_1 and L_2. For trees that had approximately equal quantities of L_1 and L_2, the intra-clumping comparisons were perfectly overlapping in our measure by definition, while the diversity measures were overlapping but not perfectly (Fig. 3(a)).

In addition, our measure has the inter- and intra-clumping comparisons meeting while the diversity measures do not (Fig. 3(a)). Furthermore, we see that our measure increases and decreases the same amount, while the diversity measures have the inter-clumping comparison decreasing less than the intra-clumping comparisons increase (Fig. 3(a)). In the trees with approximately equal quantities of L_1, L_2, and L_3, we see that our measure maintains the same range as before whereas the diversity measures do not (Fig. 3(b)). In the trees with three times as many L_1 leaves as L_2 or L_3, our measure remains stable in its clumpiness with respect to the number of neighbors assigned, while the diversity measures are inconsistent with the previous types of trees and even within intra-clumping comparisons as the L_1 comparison is always smaller than the L_2 comparison (Fig. 3(c)).

From these simulations, we see that our clumpiness measure was more consistent than the other measures when more labels or different sample sizes of labels are introduced. Furthermore, the measure was more descriptive about the distribution of the labels than the other two measures and more in line with our concept of clumpiness and aggregation.

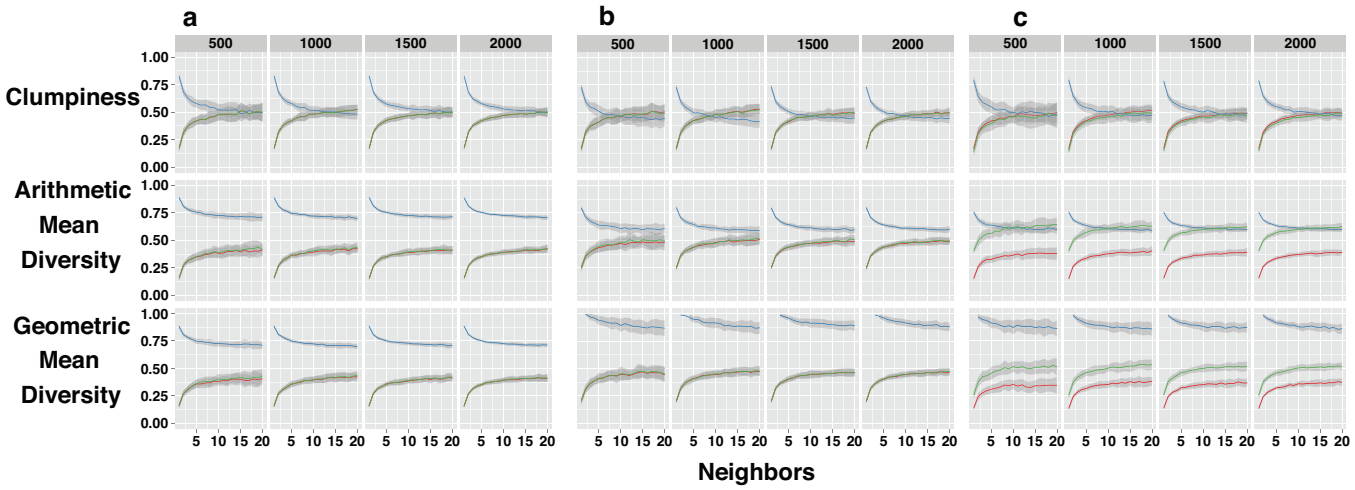

Fig. 3. Values of potential clumpiness measures for labels L_1, L_2, and L_3 of different populations in different size trees. Each facet within the plots represents values from random rooted full binary trees of size 500, 1000, 1500, and 2000 vertices (with a variation of ± 10 vertices). The x-axis is the number of nearest leaves to a randomly chosen leaf that all become the same label, unless a leaf is already assigned a label. The y-axis represents either our own clumpiness measure, the arithmetic mean diversity of the tree, or the geometric mean diversity of the tree (we exclude the root vertex in all instances). Each point in a line represent the median of the measure from 100 random trees of that type, while the gray shading represents the median absolute deviation. There are three lines in each plot, one representing the measure of L_1 with itself (red), L_1 with L_2 (blue), and L_2 with itself (green). (a) The clumpiness of trees with approximately equal quantities of L_1 and L_2. (b) The clumpiness of trees with approximately equal quantities of L_1, L_2, and L_3. (c) The clumpiness of trees with three times as many L_1 leaves as L_2 or L_3. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4.4. Simulations validate the clumpiness measure

We say that clumpiness in our simulated trees is defined by how many surrounding neighbors are assigned the same type as a randomly selected leaf. By this definition, we see that our measure did capture clumpiness in the random trees (Fig. 3). That is, the more neighbors assigned closest to a random leaf, the less clumpy our inter-clumping comparison was and the more clumpy our intra-clumping comparisons were. This effect was not only consistent across tree sizes, but the values were as well. In trees with approximately equal quantities of L_1 and L_2, we see that all of the measures generally withstand the size of the tree (Fig. 3(a)). Even when there are three labels in the tree of approximately equal quantities, we still see a decrease in all inter-clumping comparisons and an increase in all intra-clumping comparisons (Fig. 3(b)). The diversity measures, however, have very different ranges from before and even within the same comparisons. They also have different ranges between the inter- and intra-clumping comparisons within this type of tree, seemingly due to the addition of a third label influencing the sample (Fig. 3(b)). When calculating the clumpiness in a tree with varying sample sizes, however, we see that our measure is able to withstand the biases while the diversity measures do not (Fig. 3(c)). For this reason, we propose that our measure is stable across tree sizes and sample sizes and can be used for finding the relationships between "wholes" in hierarchical clusterings of "parts of a whole" analyses.

5. Biological application

With our clumpiness measure, we can now answer questions which require the union of different levels of the hierarchical container. We here sought the relationship between heavy and light chains of the germline immune receptors based on their sequence fragments. These relationships can reveal the evolutionary pathways of the receptors and can identify similarities between the BCR and the TCR.

5.1. Hierarchical clustering

We analyzed 517 human and 677 mouse light chain and heavy chain V gene protein sequences based on the entire known B cell and T cell receptor repertoires in the ImMunoGeneTics (IMGT) database [19]. Taking an overlapping sliding window across all 1194 alleles, we split the sequences into fragments consisting 20 residues. We used hierarchical spectral clustering to cluster together all fragments into a binary tree, where the leaves of the tree are clusters [20]. The clustering algorithm in [20] uses fuzzy metrics to define similarity between strings by counting the number of "qgrams" within a string. Here, we used qgrams with $q = 3$ to generate the tree. The algorithm stops when the Newman–Girvain modularity for a clustering step is less than or equal to 0, leaving us a with a leaf that contains a collection of fragments that we call a cluster [20]. From this tree, we labeled each cluster for its originating chain type (light or heavy) and receptor type (BCR or TCR). Clusters that contained multiple labels were not measured for clumpiness, but remained in the tree as irrelevant vertices in order to maintain the structure.

5.2. Results

The clumpiness between chains from human and mouse BCR and TCR light chain and heavy chain V gene protein sequence fragments using our measure can be found in Fig. 4. Each chain was highly related with itself, with IGH clusters being the most clumpy while TRB clusters were the least clumpy. We found that the next highest relationships were the TCR chains with themselves and the light chain BCRs with themselves. Lastly, we found that IGL clusters had the closest relationship with the TCR chains.

5.3. IGL fragments clump more with TCRs than IGH or IGK

Using our clumpiness measure and hierarchical spectral clustering, we sought the relationships between chains in human and mouse BCR and TCR V gene protein sequences (Fig. 4). We found that heavy chain BCRs had the highest similarity out of every

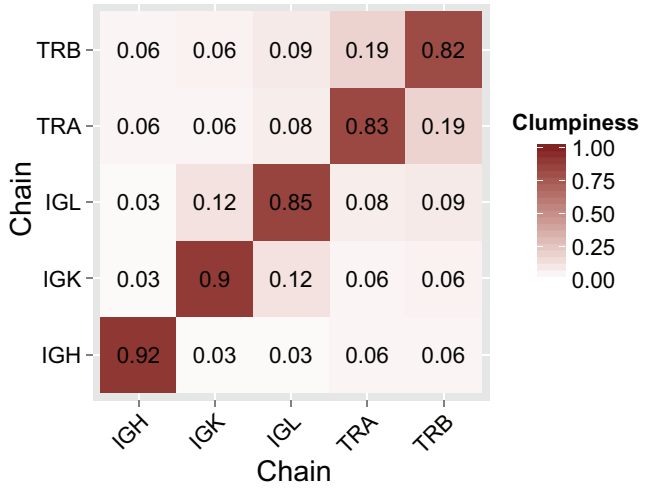

Fig. 4. The heatmap of clumpiness values between chains in the hierarchical spectral clustering of human and mouse BCR and TCR protein sequence fragments. The value in each cell is the value of the clumpiness measure and the color corresponds to each value, with white as 0 and red as 1. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

chain, followed by IGK, IGL, TRA, and TRB. We have previously found TCRs to be more diverse than their BCR counterparts, so this finding makes biological sense [17]. We also saw that the relationship between TRA and TRB was stronger than between IGK and IGL. In fact, IGH was very much unlike IGK or IGL (Fig. 4). Furthermore, we found that IGL had the highest relationship out of all BCR chains to the TCR chains (Fig. 4). This finding supports previous evidence stating that IGL was the first chain to branch off of its TCR ancestors [21].

It is worth noting that neither the arithmetic mean diversity nor geometric mean diversity measures were able to capture this information due to the low sample size of IGL compared to the other chains (results not shown).

6. Conclusions

We have demonstrated the robustness of this clumpiness measure to differing tree sizes and sample sizes and have shown its utility in a real biological application. Using this measure, we are now able to link the relationship of the data creating a hierarchical container with the metadata categorizing the data. With this unification, we were able to show that there exists a relationship between the BCR and TCR receptor chains through IGL using short sequence fragments or motifs from the receptor genes. The generalization for this method should provide the discovery of previously unseen patterns within hierarchical containers.

Acknowledgments

Gregory W. Schwartz is funded by the U.S. Department of Education Graduate Assistance in Areas of National Need (GAANN) program, CFDA Number: 84.200. Research reported in this publication was supported by the National Institute Of Allergy And Infectious Diseases of the National Institutes of Health under Award Number P01AI106697 and the National Science Foundation Information & Intelligent Systems under Award Number 1551338. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health or the National Science Foundation.

The clumpiness metric was implemented in the Haskell programming language and is available at https://github.com/DrexelSystemsImmunologyLab/clumpiness. An application for finding the clumpiness using this library is also available at https://github.com/DrexelSystemsImmunologyLab/find-clumpiness.

References

[1] L.J. Mcheyzer-Williams, P.J. Milpied, S.L. Okitsu, M.G. Mcheyzer-Williams, Class-switched memory B cells remodel BCRs within secondary germinal centers, Nature Immunol. 16 (February) (2015) 1–12, doi:10.1038/ni.3095.
[2] P.J. Clark, F.C. Evans, Distance to nearest neighbor as a measure of spatial relationships in populations, Ecology 35 (4) (1954) 445, doi:10.2307/1931034.
[3] H. Li, J.F. Reynolds, A new contagion index to quantify spatial patterns of landscapes, Landscape Ecol. 8 (3) (1993) 155–162, doi:10.1007/BF00125347.
[4] J.N. Perry, Spatial analysis by distance indices, J. Anim. Ecol. 64 (3) (1995) 303–314, doi:10.2307/5892.
[5] J.N. Perry, Measures of spatial pattern for counts, Ecology 79 (3) (1998) 1008–1017, doi:10.1890/0012-9658(1998)079[1008:MOSPFC]2.0.CO;2.
[6] H.S. He, B.E. DeZonia, D.J. Mladenoff, An aggregation index (AI) to quantify spatial patterns of landscapes, Landscape Ecol. 15 (7) (2000) 591–601, doi:10.1023/A:1008102521322.
[7] K. McGarigal, S.A. Cushman, E. Ene, FRAGSTATS v4: Spatial pattern analysis program for categorical and continuous maps. Computer software program produced by the authors at the University of Massachusetts, Amherst (2012). Available at: http://www.umass.edu/landeco/research/fragstats/fragstats.html.
[8] B. Li, L.V. Madden, X. Xu, Spatial analysis by distance indices: an alternative local clustering index for studying spatial patterns, Methods Ecol. Evol. 3 (2012) 368–377, doi:10.1111/j.2041-210X.2011.00165.x.
[9] B. Parresol, L. Edwards, An entropy-based contagion index and its sampling properties for landscape analysis, Entropy 16 (4) (2014) 1842–1859, doi:10.3390/e16041842.
[10] J. Seo, B. Shneiderman, Interactively exploring hierarchical clustering results, Computer 35 (7) (2002) 80–86, doi:10.1109/MC.2002.1016905.
[11] N. Elmqvist, J.-d. Fekete, Hierarchical aggregation for information visualization: overview, techniques, and design guidelines, IEEE Trans. Vis. Comput. Graphics 16 (3) (2010) 439–454.
[12] J. Heinrich, C. Vehlow, F. Battke, G. Jager, D. Weiskopf, K. Nieselt, iHAT: interactive hierarchical aggregation table for genetic association data, BMC Bioinf. 13 (Suppl 8) (2012) S2. Available at: http://bmcbioinformatics.biomedcentral.com/articles/10.1186/1471-2105-13-S8-S2.
[13] J. Bernard, N. Wilhelm, B. Krüger, T. May, T. Schreck, J. Kohlhammer, MotionExplorer: exploratory search in human motion capture data based on hierarchical aggregation, IEEE Trans. Vis. Comput. Graphics 19 (12) (2013) 2257–2266, doi:10.1109/TVCG.2013.178.
[14] L. Jost, Entropy and diversity, Oikos 113 (2006) 363–375. http://onlinelibrary.wiley.com/doi/10.1111/j.2006.0030-1299.14714.x/abstract.
[15] M. Hill, Diversity and evenness: a unifying notation and its consequences, Ecology 54 (2) (1973) 427–432, doi:10.2307/1934352.
[16] C.E. Shannon, A mathematical theory of communication, Bell Syst. Tech. J. 27 (1948) 379–423, doi:10.1002/j.1538-7305.1948.tb01338.x.
[17] G.W. Schwartz, U. Hershberg, Conserved variation: identifying patterns of stability and variability in BCR and TCR V genes with different diversity and richness metrics., Phys. Biol. 10 (3) (2013) 035005, doi:10.1088/1478-3975/10/3/035005.
[18] F. Emmert-Streib, M. Dehmer, Information theoretic measures of UHG graphs with low computational complexity, Appl. Math. Comput. 190 (2007) 1783–1794, doi:10.1016/j.amc.2007.02.095.
[19] M.P. Lefranc, IMGT®, the International ImMunoGeneTics information system® for immunoinformatics: Methods for querying IMGT® databases, tools, and web resources in the context of immunoinformatics, Mol. Biotechnol. 40 (2008) 101–111, doi:10.1007/s12033-008-9062-7.
[20] L. Shu, A. Chen, M. Xiong, W. Meng, Efficient SPectrAl Neighborhood blocking for entity resolution, 2011 IEEE 27th International Conference on Data Engineering (2011) 1067–1078, doi:10.1109/ICDE.2011.5767835.
[21] M.H. Richards, J.L. Nelson, The evolution of vertebrate antigen receptors: a phylogenetic approach, Mol. Biol. Evol. 17 (2000) 146–155.