TCR distance calculation and clustering: TCRcluster

Zhu Yule, Tang Jiayi, Huang Yutong, Hu Zihao, Lu Jianzhang

Group 1

2022.12.15

ZJE 浙江大学爱丁堡大学联合学院
ZJU-UoE INSTITUTE

TCR recognition

- Epitope, also called antigenic determinant, is a site or region on an antigen that is recognized by an antibody as a foreign body.
- T cell receptors (TCRs) recognize antigens presented by major histocompatibility complex (MHC) on antigen-presenting cells (APC).

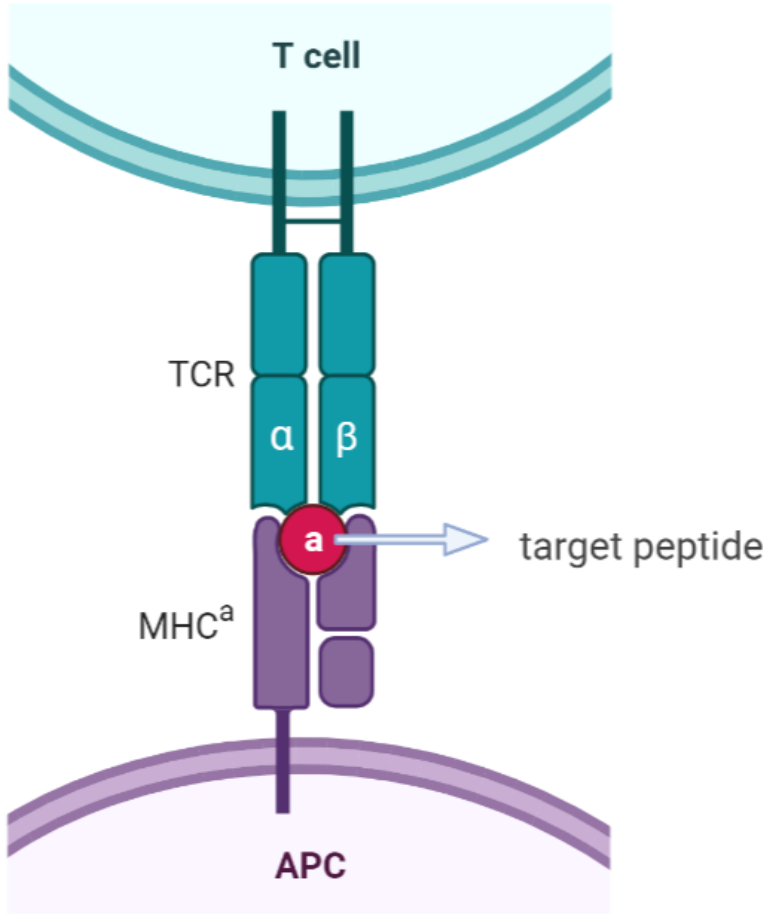


Figure: TCR recognization (created by biorender.com)

CDR3

- CDR3 regions on β chain may largely decide the recognition of TCR-pMHC
- **Project aim:** Cluster TCR sequences according to their antigen-binding affinity.

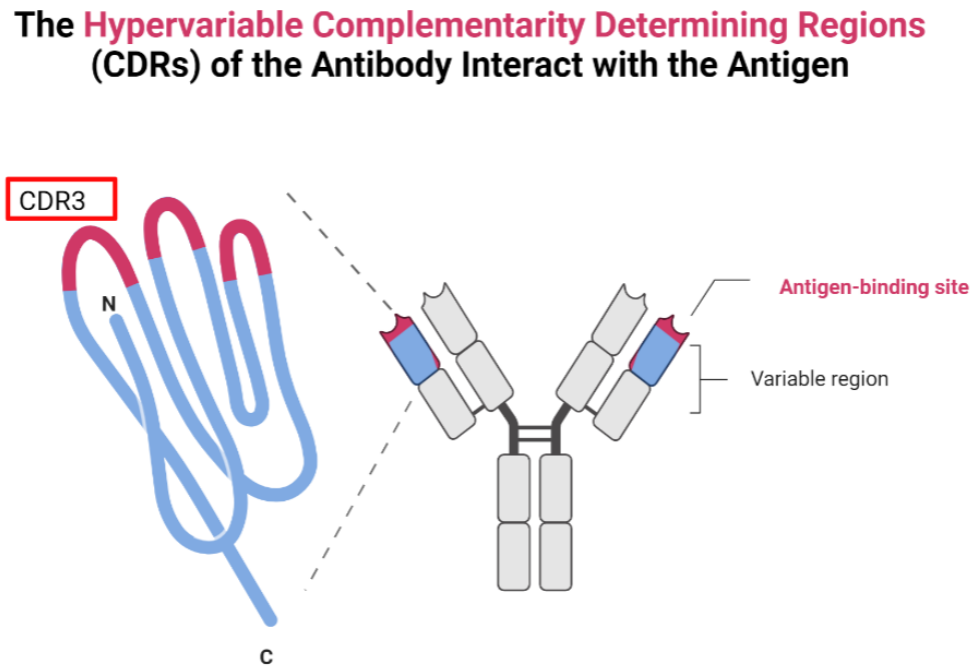
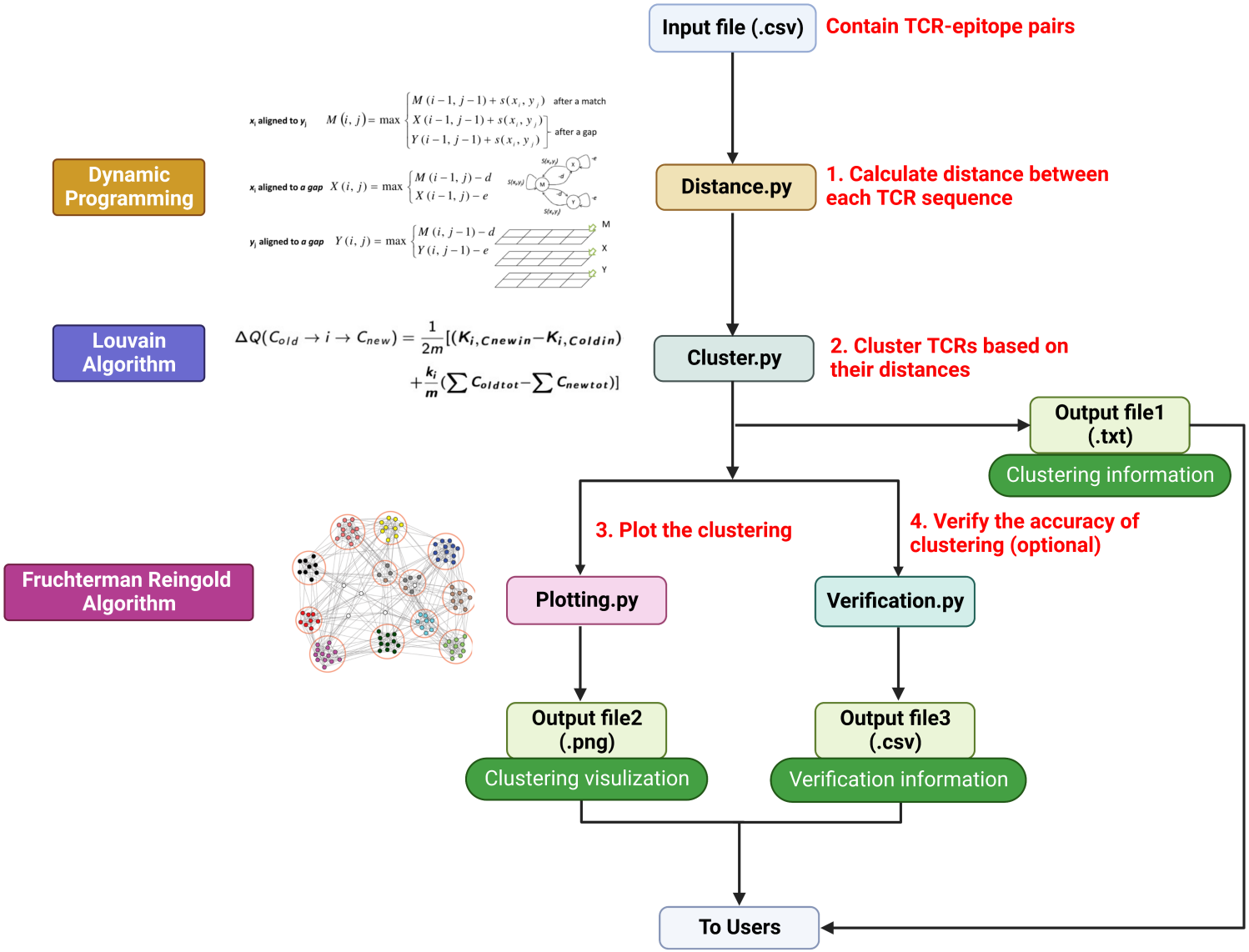


Figure: CDR3 region

Flow Chat


Input file (.csv)
Contain TCR-epitope pairs
Dynamic Programming
Distance.py
1. Calculate distance between each TCR sequence
Louvain Algorithm
Cluster.py
2. Cluster TCRs based on their distances
Output file1 (.txt)
Clustering information
3. Plot the clustering
4. Verify the accuracy of clustering (optional)
Fruchterman Reingold Algorithm
Plotting.py
Verification.py
Output file2 (.png)
Clustering visulization
Output file3 (.csv)
Verification information
To Users

The similarity (distance) between each TCR sequence

Workflow

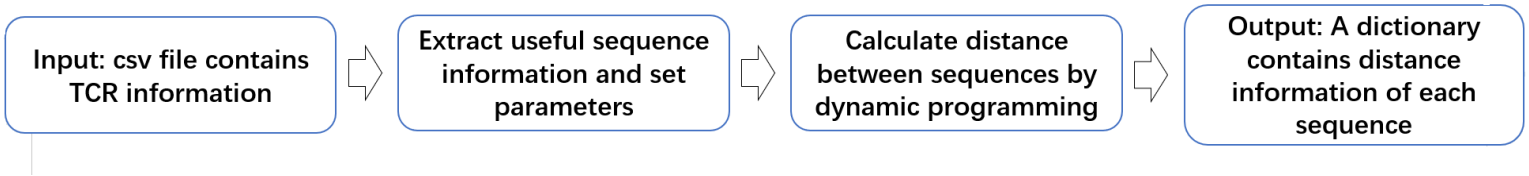


Figure: The workflow for calculating distance

Input

	A	B	C	D	E	F	G
1		CDR3a	CDR3b	HLA	peptide	epitope	MHC
2	seq0	CAAAAGNTGKLIF	CASSRLGASAETLYF	-	HGIRNASFI	-	-
3	seq1	CAAAAYNQGGKLIF	CATSDPAGMTGGWHGYTF	A03:01	KLGGALQAK	IE-1_CMV	YFAMYQENVAQTDVDTLYIIYRDYTWAELAYTWY
4	seq2	CAAADDKIIF	CASSQGTISYEQYF	A02:01	LLWNGPMAV	-	YFAMYGEKVAHTHVDTLYVRYHYYTWAVLAYTWY
5	seq3	CAAADNYGQNFVF	CAWSSGEGTDTQYF	A11:01	AVFDRKSDAK	EBNA-3B_EBV	YYAMYQENVAQTDVDTLYIIYRDYTWAAQAYRWY
6	seq4	CAAAETSYDKVIF	CAGGGSQGNLIF	A02:01	GILGFVFTL	Flu-MP_Influenza	YFAMYGEKVAHTHVDTLYVRYHYYTWAVLAYTWY
7	seq5	CAAAETSYDKVIF	CAGPPVAGANNLFF	B08:01	RAKFKQLL	BZLF1_EBV	YDSEYRNIFTNTDESNLYLSYNYYTWAVDAYTWY
8	seq6	CAAAETSYDKVIF	CASSFSGNTGELFF	B08:01	RAKFKQLL	BZLF1_EBV	YDSEYRNIFTNTDESNLYLSYNYYTWAVDAYTWY
9	seq7	CAAAETSYDKVIF	CASSFVDRAETQYF	A11:01	IVTDFSVIK	EBNA-3B_EBV	YYAMYQENVAQTDVDTLYIIYRDYTWAAQAYRWY
10	seq8	CAAAETSYDKVIF	CASSFVDRAETQYF	B08:01	RAKFKQLL	BZLF1_EBV	YDSEYRNIFTNTDESNLYLSYNYYTWAVDAYTWY
11	seq9	CAAAETSYDKVIF	CASSLKGGRHEQYF	A03:01	RLRAEAQVK	EMNA-3A_EBV	YFAMYQENVAQTDVDTLYIIYRDYTWAELAYTWY
12	seq10	CAAAETSYDKVIF	CASSLQSQGAPYEQYF	B08:01	RAKFKQLL	BZLF1_EBV	YDSEYRNIFTNTDESNLYLSYNYYTWAVDAYTWY

Figure: The input information

Dynamic programming (preparation)

- Three states to record the sequences' situation when matching.

X, Y and M states

X state means to open a gap in the column direction;
Y state means to open a gap in the line direction;
M state means two sequences are matched in this position.

0	-inf	-inf	-inf	-inf
-2	0	0	0	0
-3	0	0	0	0
-4	0	0	0	0
-5	0	0	0	0

Figure: X state

0	-2	-3	-4	-5
-inf	0	0	0	0
-inf	0	0	0	0
-inf	0	0	0	0
-inf	0	0	0	0

Figure: Y state

0	-inf	-inf	-inf	-inf
-inf	0	0	0	0
-inf	0	0	0	0
-inf	0	0	0	0
-inf	0	0	0	0

Figure: M state

Dynamic programming (formula)

- Dynamic programming for the transformation between states.

Transformation of three states

$$M(i,j) = \max \begin{cases} M(i-1,j-1) + s(x_i, y_j) & \text{after a match} \\ X(i-1,j-1) + s(x_i, y_j) \\ Y(i-1,j-1) + s(x_i, y_j) \end{cases} \text{after a gap}$$

AT
TC

AT
-C
-T
TC

$$X(i,j) = \max \begin{cases} M(i-1,j) - d \\ X(i-1,j) - e \end{cases}$$

$$Y(i,j) = \max \begin{cases} M(i,j-1) - d \\ Y(i,j-1) - e \end{cases}$$

Figure: The iterative process between states

Dynamic programming (output)

- Calculating the distance between every two sequences:

The distances for each sequence

```
{Node1:{Node2:dis(1,2),Node3:dis(1,3)...},
Node2:{Node1:dis(1,2),Node3:dis(2,3)...},
...}
```

Figure: The output for the sequence distances

The complexity of the algorithm: $O(n^2k^2)$
n: the number of the sequences
k: the length of one sequence

Clustering

- Greedy algorithm O(n log n) running time
- According to Louvain algorithm, introduce **Modularity**.
- Modularity Q: The overall quality of how the network is segmented by communities (**clusters**)

$$Q = \frac{1}{2m} \sum_{ij} [A_{ij} - \frac{k_i k_j}{2m}] \delta(c_i, c_j)$$

A_{ij}: The edge weight between nodes i and j

k_i and k_j: The sum of the weights of the edges respectively from nodes i and j

$2m$: The sum of all of the edge weights in the network

c_i and c_j: The communities of the nodes

δ: An indicator function $\delta(c_i, c_j) = 1$ if $c_i = c_j$ else 0

- $Q \propto \sum_c$ [(Real edges within c) - (Expected edges within c)]

Step 1: Modularity optimization

- Only make local changes to the memberships of nodes-communities.
- At first, each node corresponds to one community.

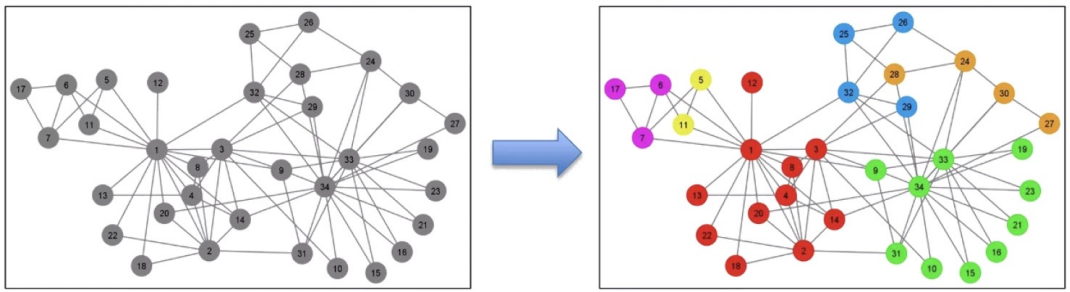


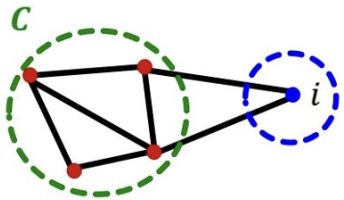
Figure: Modularity optimization step

- Every node $\rightarrow$ the community of its neighbor.
- A change in the modularity of the partitioning marked as ΔQ
- Δ Q $> 0 \rightarrow$ Gain or improve the quality
- $C' = argmax_c$, $\Delta Q(C \rightarrow i \rightarrow C')$ Iterate until no node moves.

Step 1: How to calculate ΔQ?

Derive formula: $\Delta Q(i \to C)$

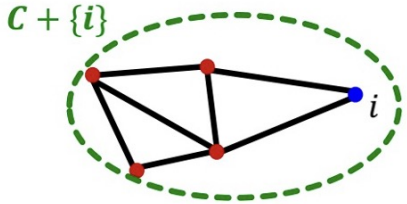
Before merging



$$Q_{\text{before}} = Q(C) + Q(\{i\})$$

$$= \left[\frac{\Sigma_{in}}{2m} - \left(\frac{\Sigma_{tot}}{2m}\right)^2\right] + \left[0 - \left(\frac{k_i}{2m}\right)^2\right]$$

After merging



$$Q_{\text{after}} = Q(C + \{i\})$$

"Σ_{in}" of $C + \{i\}$ "Σ_{tot}" of $C + \{i\}$

$$= \frac{\Sigma_{in} + k_{i,in}}{2m} - \left(\frac{\Sigma_{tot} + k_i}{2m}\right)^2$$

$$\Delta Q(i \to C) = \frac{1}{2m}\left(\frac{\sum_{tot} k_i}{m} - K_{i,i_n}\right)$$

$$\Delta Q(C_{old} \to i \to C_{new}) = \frac{1}{2m}\left[\left(K_{i,Cnewin} - K_{i,Coldin}\right) + \frac{k_i}{m}\left(\sum C_{oldtot} - \sum C_{newtot}\right)\right]$$

Step 2: Aggregation

The first step → Attain the current stability

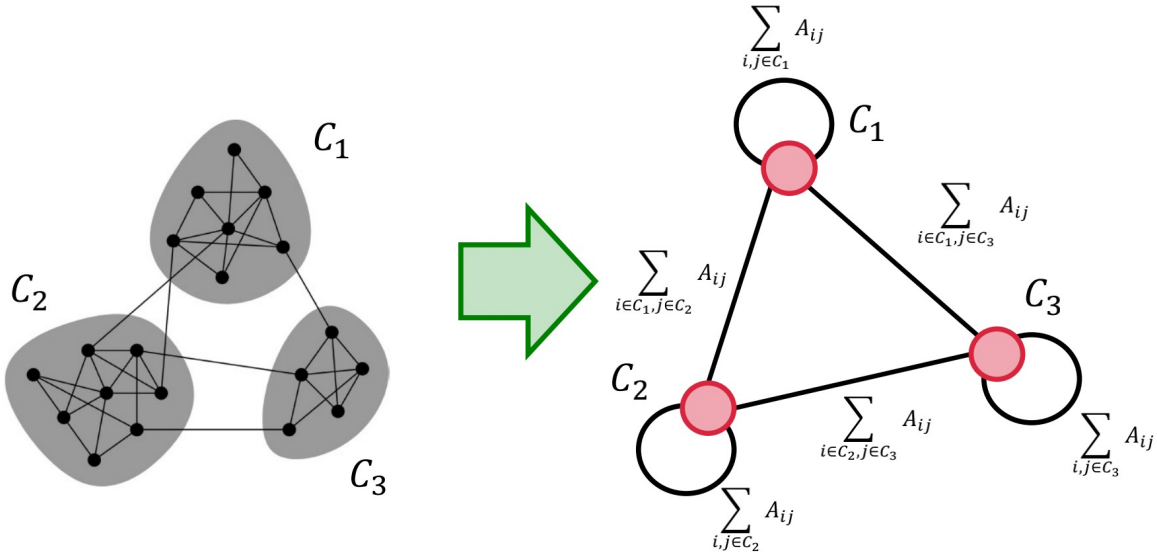


Figure: Aggregation step

- The communities obtained in the first phase are aggregated into super-nodes.
- The edges within the original community → self-loop.
- All edges between the communities → weights between super-nodes.

Clustering: Summary

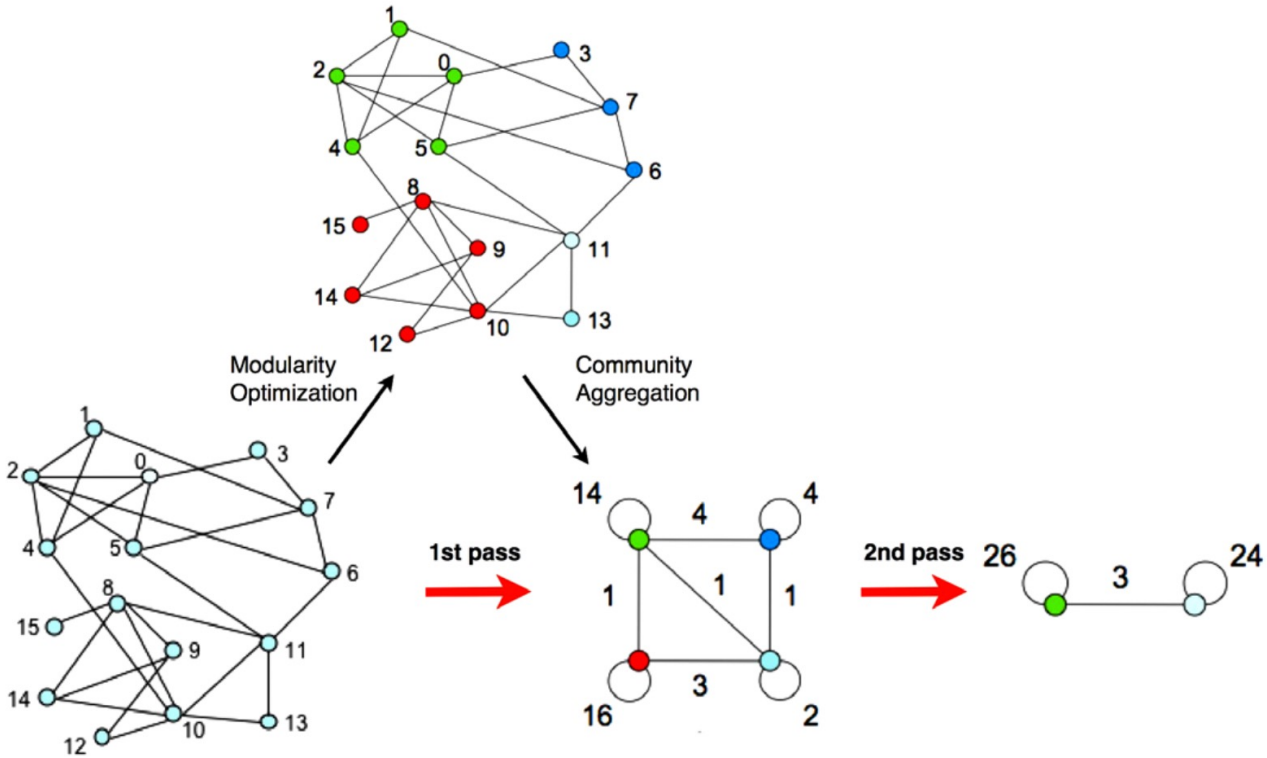


Figure: Summary of the clustering method

- The passes are repeated constantly until no increase of modularity can be gained.

Visualization: Work Flow

- To preliminarily test the accuracy of our clustering process
- Use **networkx** and **FR algorithm** to do the plotting.

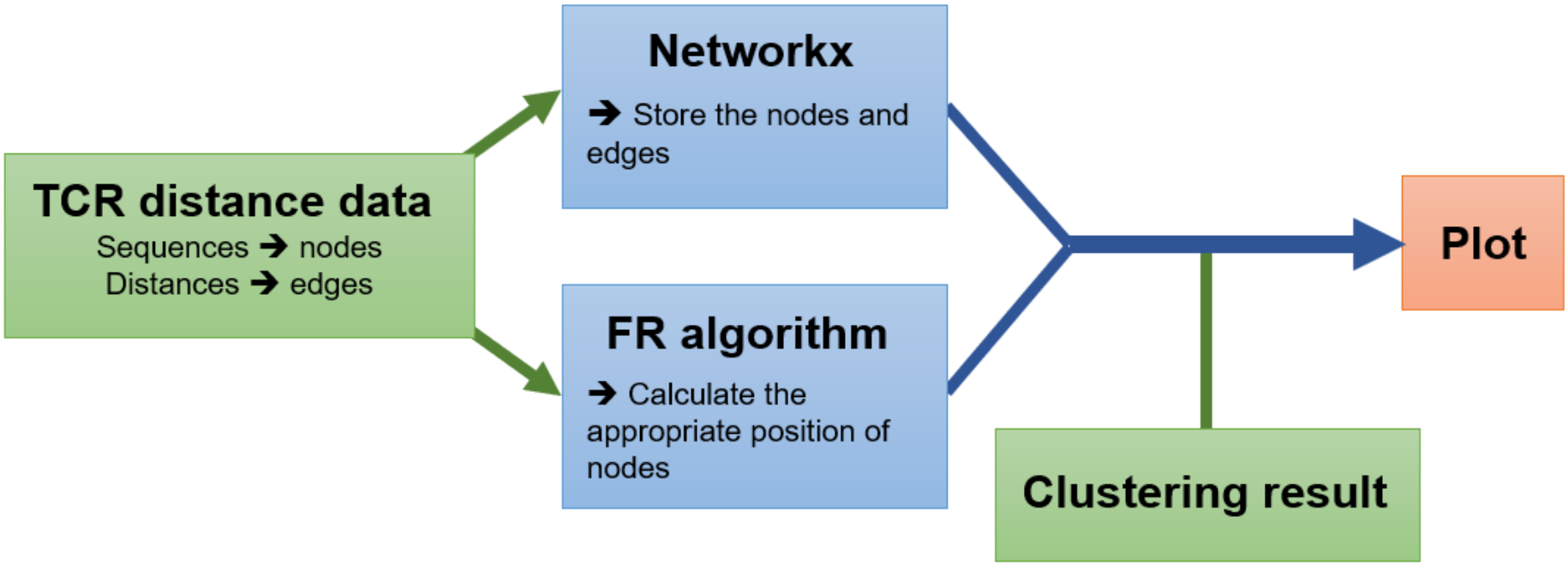


Figure: The visualization work flow

Visualization: FR algorithm

- **Fruchterman-Reingold algorithm: O(n^2)**

Treat all nodes as charged particles. In this way, there are two forces act on each node: the **coulomb force** and **tensile force**.

1) **Coulomb forces**: $F_q = \frac{k_q \cdot q^2}{r^2}$ keep nodes separated.
2) **Tensile forces**: $F_k = k \cdot x$ promote nodes with high similarity to get closer.

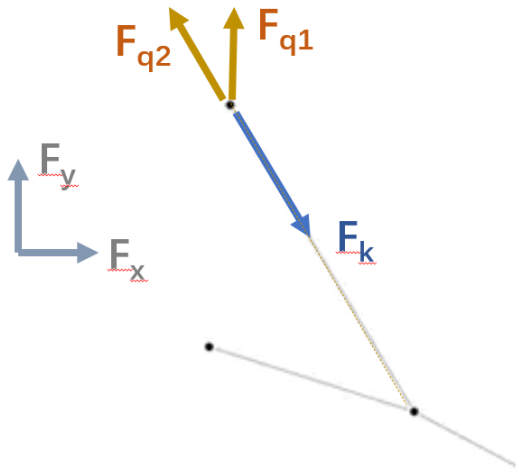


Figure: Force condition analysis in FR algorithm

Visualization: Result

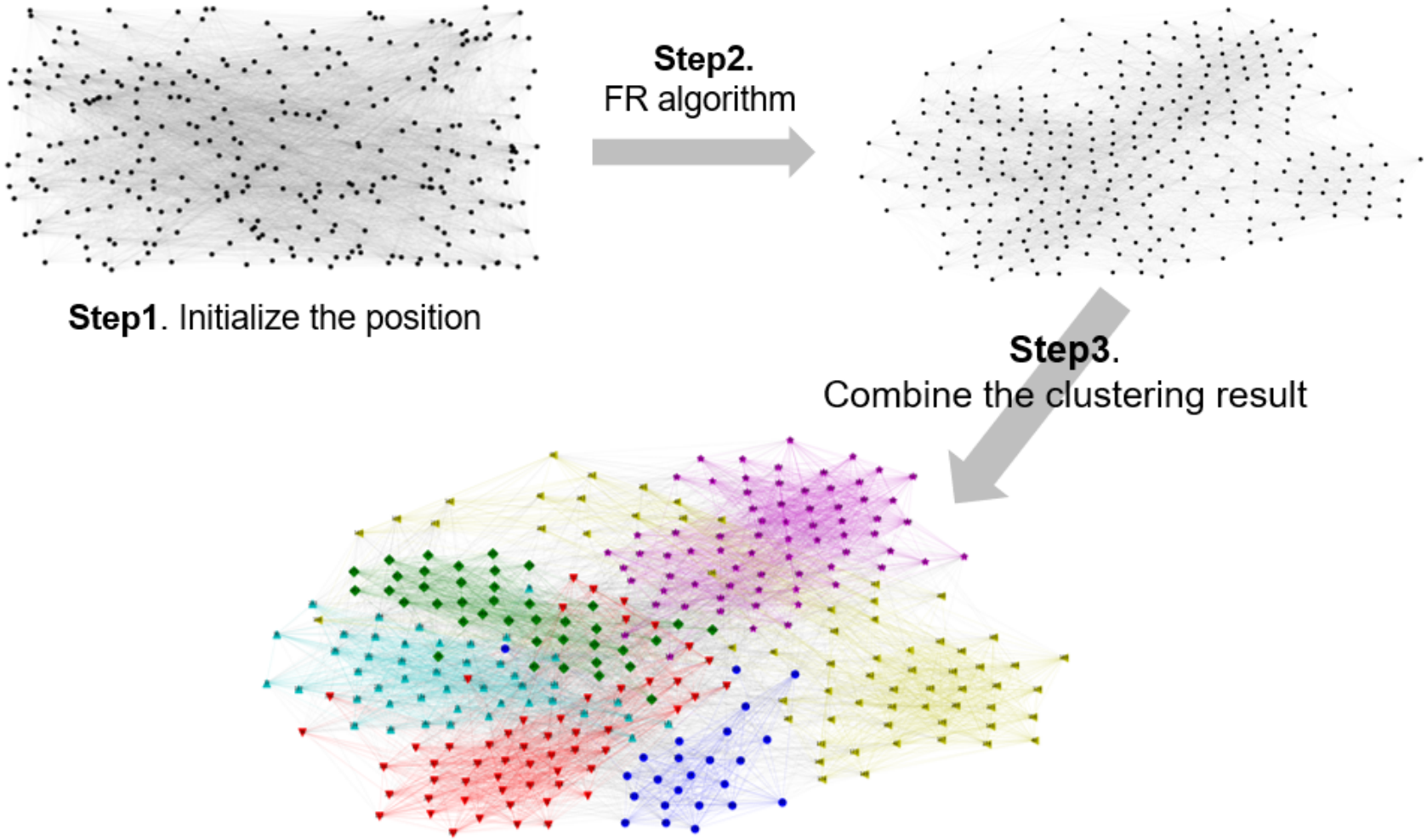
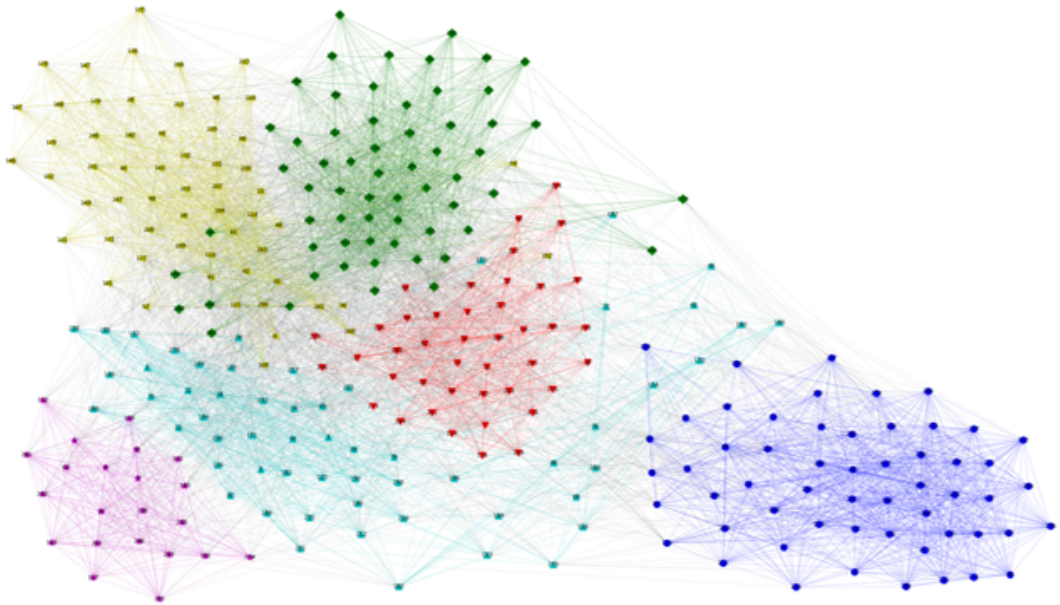


Figure: The visualization process

Output file

Community 0: 2, 11, 12, 14, 24, 29, 44, 60, 66, 70,
Community 1: 1, 3, 5, 8, 9, 16, 30, 46, 50, 52, 63, 6
Community 2: 0, 13, 32, 35, 39, 43, 49, 51, 54, 55,
Community 3: 19, 22, 25, 27, 33, 61, 65, 71, 74, 90
Community 4: 15, 20, 28, 40, 41, 42, 45, 48, 59, 86
Community 5: 4, 6, 7, 10, 17, 18, 21, 23, 26, 31, 34

Output1: clustering information



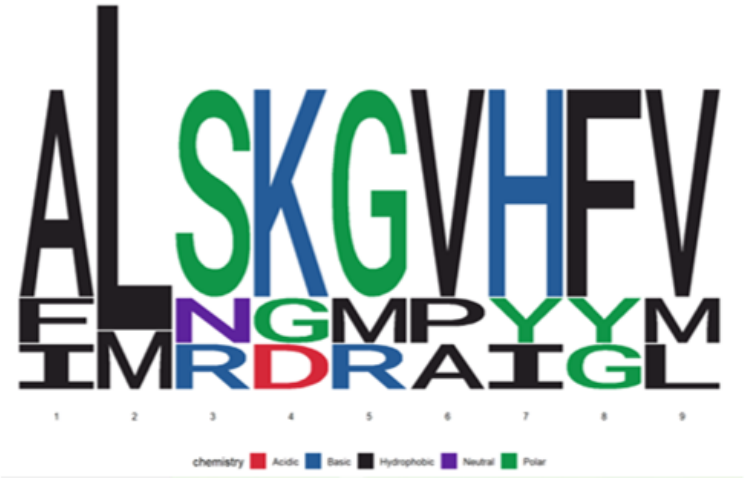
Output2: visualization

Verification

- Provide consensus motif for each cluster
- **Ideal results:** same or similar antigens in the same cluster

	Communit	CDR3b	Peptide	Consensus
0	0	CASSIDRGSEAFF	ALSKGVHFV	ALSKGVHFV
1	0	CASSLYGLGTEAFF	ALSKGVHFV	ALSKGVHFV
2	0	CASSTEGGTEAFF	ALSKGVHFV	ALSKGVHFV
3	0	CASSRPGGGSTEAFF	ALSKGVHFV	ALSKGVHFV
4	0	CASSLEGQLNTEAFF	ALSKGVHFV	ALSKGVHFV
5	0	CASSSYPIWDYTEAFF	ALSKGVHFV	ALSKGVHFV
6	0	CASRGDRGRATEAFF	ALSKGVHFV	ALSKGVHFV
7	0	CAVNRDAGKSTF	FLRGRAYGL	ALSKGVHFV
8	0	CAVNRDAGKSTF	FLRGRAYGL	ALSKGVHFV
9	0	CASSLGQNTEAFF	FLRGRAYGL	ALSKGVHFV

Output3: consensus motif



Motif logo of the red cluster

Advantage and limitation of the distance calculation:

Advantage:

$Distance = (D_{CDR3\beta}) * weight + (D_{CDR3\alpha}) * (1 - weight)$

CDR3β is required, while CDR3α also plays a role in antigen recognition.

Limitation:

Very different (low similarity) TCRs may recognize the same epitope.

Possible Solution:

Algorithms based on *tcrdist3* or machine learning.

	CDR3a	CDR3b	HLA	peptide
seq17027	CAVGGPI	CASTARALNTGELFF	A02:01	LLDFVRFMGV
seq17403	CAVHGYGQNFVF	CVVGEFYF	A02:01	LLDFVRFMGV

Figure: Two TCRs with low alignment score recognize the same peptide

Advantage and limitation of the clustering:

Advantage:

1. Do not require setting the number of clusters beforehand.
2. Louvain algorithm greedily maximizes modularity. Our codes run much faster than the standard package.

Limitation:

1. Unstable result: randomization of the iteration order of the nodes.
2. May yield badly connected communities.

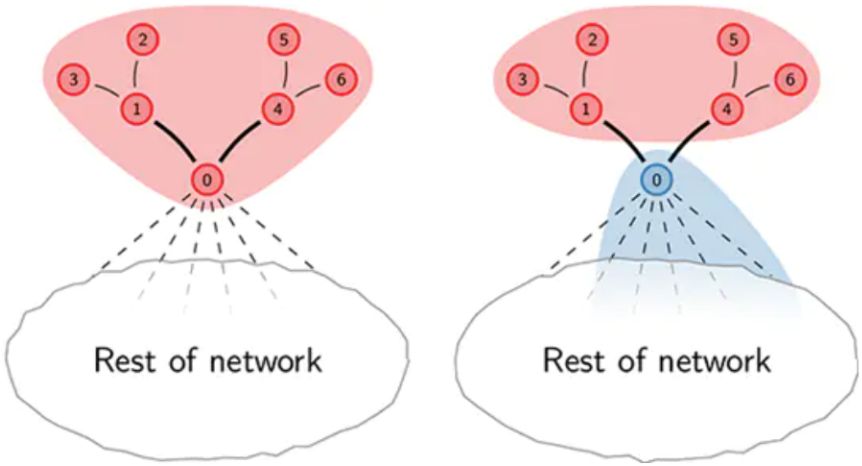


Figure: A reason to yield bad communities

Advantage and limitation of the plotting:

Advantage:

FR algorithm is based on the theory of particle physics: easy to understand.

Limitation:

1. Need to change the position of each node every 0.3 seconds for about 700 times (300 sequences $\sim$ 7 min).
2. Randomization of the initial position of each sequence.

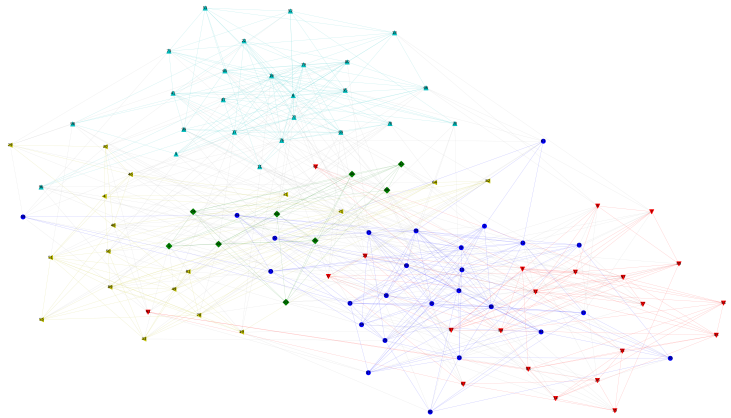


Figure: A bad plotting under 100 sequences

Significance and application:

Diagnosis:

1. Reflect the function and immune response state of T cells and design personalized treatments for patients.
2. Detect autoimmune diseases such as type I diabetes.
3. Predict the recognized antigens of new TCRs.

Novel treatments:

1. Vaccines conferring T-cell immunity against specific cancer mutations: injection of transcribed mRNA coding for antigens.
2. TCR-T cell therapy: Use CRISPR-Cas9 system to knock out endogenous TCR α and β genes and replace them with tumour-specific TCR sequences.

References:


Blondel, V.D. et al. (2008) Fast unfolding of communities in large networks, Journal of Statistical Mechanics: Theory and Experiment, 2008, p. 10008.


Dash, P. et al. (2017) Quantifiable predictive features define epitope-specific T cell receptor repertoires, Nature, 547(7661), pp. 8993.


Foy, S.P. et al. (2022) Non-viral precision T cell receptor replacement for personalized cell therapy, Nature [Preprint].


Fruchterman, T.M.J. and Reingold, E.M. (1991) Graph drawing by force-directed placement, Software: Practice and Experience, 21(11), pp. 11291164.


Glanville, J. et al. (2017) Identifying specificity groups in the T cell receptor repertoire, Nature, 547(7661), pp. 9498.


Traag, V.A., Waltman, L. and Van Eck, N.J. (2019) From Louvain to Leiden: guaranteeing well-connected communities, Scientific Reports, 9, p. e5233.


Tusup, M. et al. (2021) mRNA-Based Anti-TCR CDR3 Tumour Vaccine for T-Cell Lymphoma, Pharmaceutics, 13(7), p. 1040.


Stanford. Stanford cs224w: Community detection in networks. 2021.

Thanks for listening