Solution to Homework One

Cong Li(PB21081606)

1. Optimal length of a dendritic arbor

The design is that now our neurons can link to each other by their axons, dendrites and the spines(in biology, we call all of these as neurites) on the dendrites. So the probability density of a given neuron's neurites appear at a given point in the function column will be:

$$\rho_n = \frac{L\frac{\pi s^2}{4}}{R^3} \propto \frac{Ls^2}{R^3} \tag{1}$$

L means the total length of the neurites a neuron has.

Attension: I've made a little change to what the teacher taught in class. The teacher said in Design 4, just replace the 'd' with 's' was okey, just as I do in equation 1. But we should notice that only dendrites have spines on them, don't the axons, that's to say, ρ_a is actually not equal to ρ_d. However, that's a little defect in our discussion. So I haven't change the form of ρ_a and ρ_d, but change their name for rigor.

Thus, the probability density of the neurite of one neuron and that of another neuron appear at the same point will be (we assume thatρ_n is independent to which neuron it is describing and annotate it as P)

$$P = \rho_n^2 \tag{2}$$

Because only those neurites occupy the same voxel with a volume d^3 can form a synapse. So the total number of the synapses between two given neurons n will be,

$$n = P \times (\frac{R}{s})^3 \tag{3}$$

In our design, there only be one synapse between two given neurons in a function column. That is,

$$n = P \times (\frac{R}{s})^3 = 1 \tag{4}$$

According to Eqs.1-2,4,

$$R \propto L^{2/3} s^{1/3} \tag{5}$$

As the volume R^3 is approximately the total volume of neurons and we can ignore the volume of the somas, we have,

$$NL\frac{\pi}{4}d^2 \sim R^3 \quad that \quad is \quad NLd^2 \sim R^3 \tag{6}$$

According to Eqs.5-6,

$$R \propto \frac{N^{2/3} d^{4/3}}{s^{1/3}} \tag{7}$$

According to Eqs.6-7,

$$L = \frac{Nd^2}{s} \tag{8}$$

2. Quantitative Analysis of Dendritic Morphology

I have made a python program to finish the quantitative analysis. See annex for the jupyter notebook. The following are the results and after that I'll make a brief introduction of my program.

2.1 Visualization of the neuronal 3D arbor shape

a) *pyramidal arbor shape*:

In Fig.1, the red ball represents the soma, and those red lines(actually cylinders) represents the arbor of the pyramidal neuron. It's clear that the pyramidal neuron has a sparse arbor which is distributed across the 3D space.

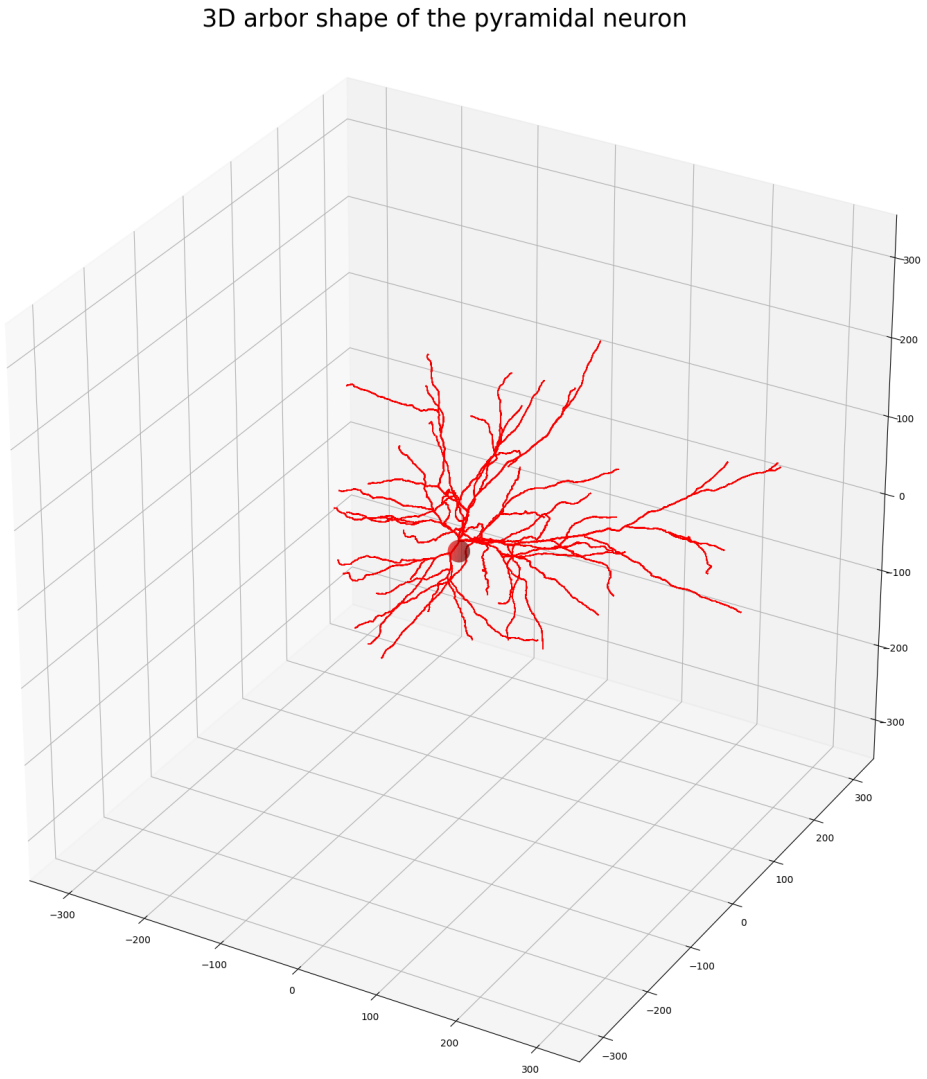

Figura 1. Pyramidal arbor 3D shape

b) *Purkinjie arbor shape*

As is shown in the Fig.2, the arbor of Purikinjie neuron is less sparse than that of pyramidal neuron. Its arbor is packed on a plane.

c) *larval zebrafish arbor shape*

Fig.3 shows an arbor shape of a neuron from the larval zebrafish's brain.

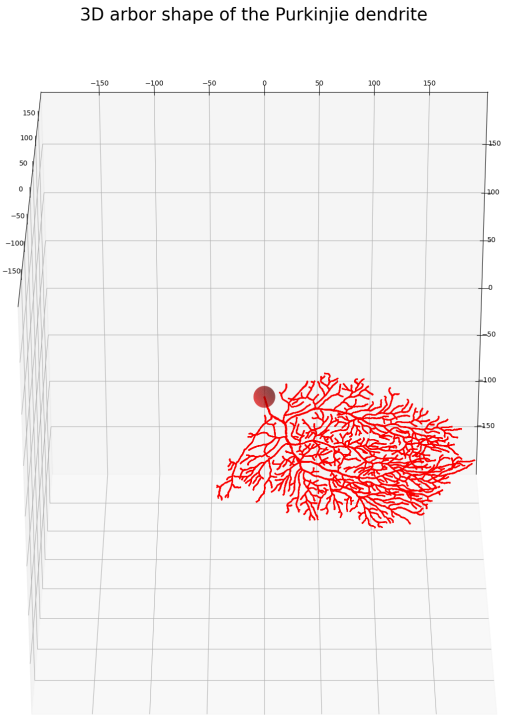

Figura 2. Purikinjie arbor 3D shape

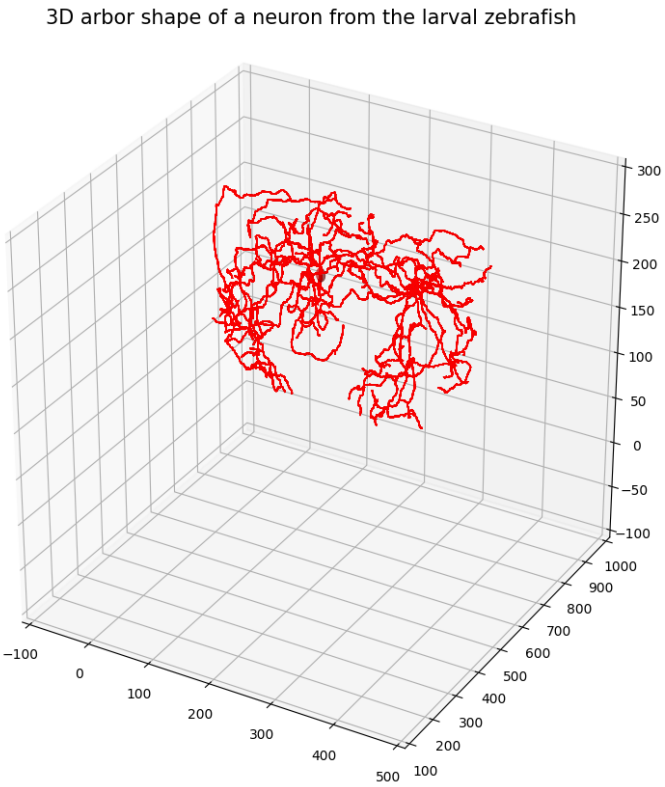

Figura 3. larval zebrafish arbor 3D shape

2.2 calculation of the mean path length

In my program, I assume the path length from a dendrite segment to the soma to be the sum of all the height of the segments encountered in this path. The Table 1 shows the calculation results.

Tabela 1. mean path length from a dendrite segment to soma

	pyramidal neuron	Purikinjie neuron	larval zebrafish's neuron
mean path length(μm)	164.819	144.060	405.229

2.3 Comparsion between the total spine reach zone area with the arbor area

a) *2D projection of the arbor*

It's obvious that the compution method of the arbor area is similar to that of the spine reach area(just change the segment radius).

So first of all, I tried to creat a 2D projection of the arbor in the 3D space. Taken into consideration that the radius of segment is significantly less than the height of the segment, I assume the projection of the segment in the 3D space to be a rectangle rather than a rounded rectangle for simplicity. In this way, I can get the projection image of the arbor of pyramidal neuron and Purikinjie neuron(Figs.4-5)

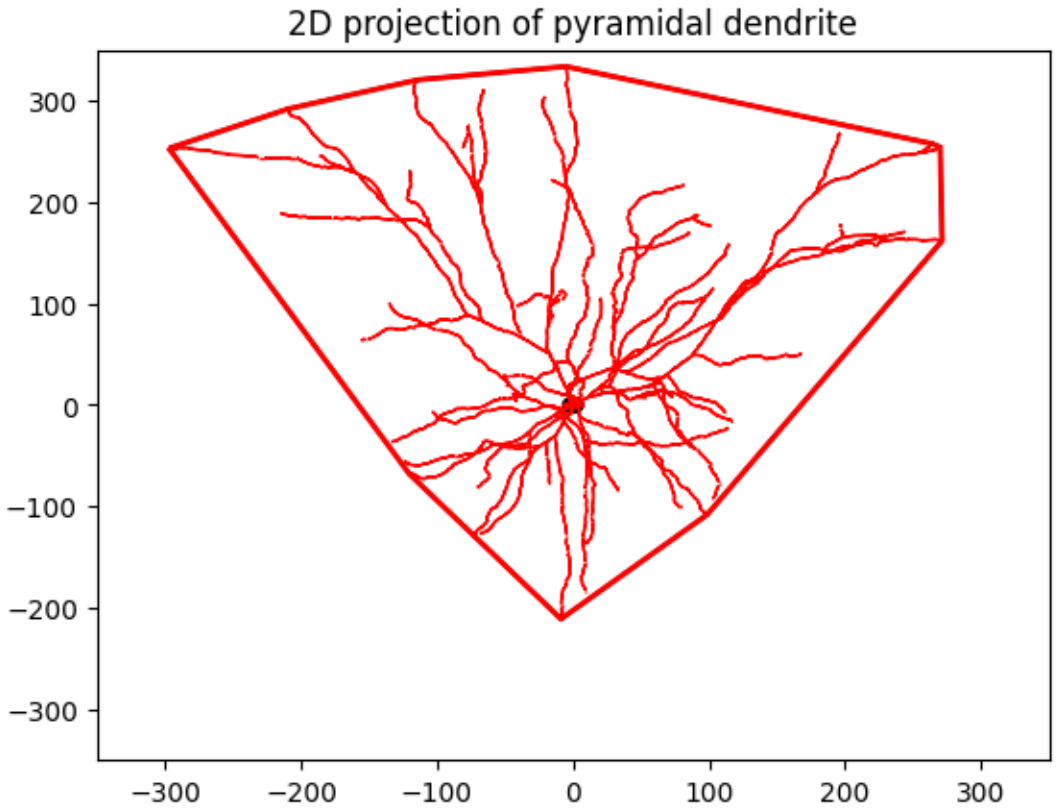

Figura 4. 2D projection of the arbor of pyramidal neuron

In Fig.4, I projected the arbor onto the xy plane.

In Fig.5, in order to ensure the projection plane is the growth plane of the arbor of Purikinjie neuron, I performed a principal component analysis(PA) before projection.

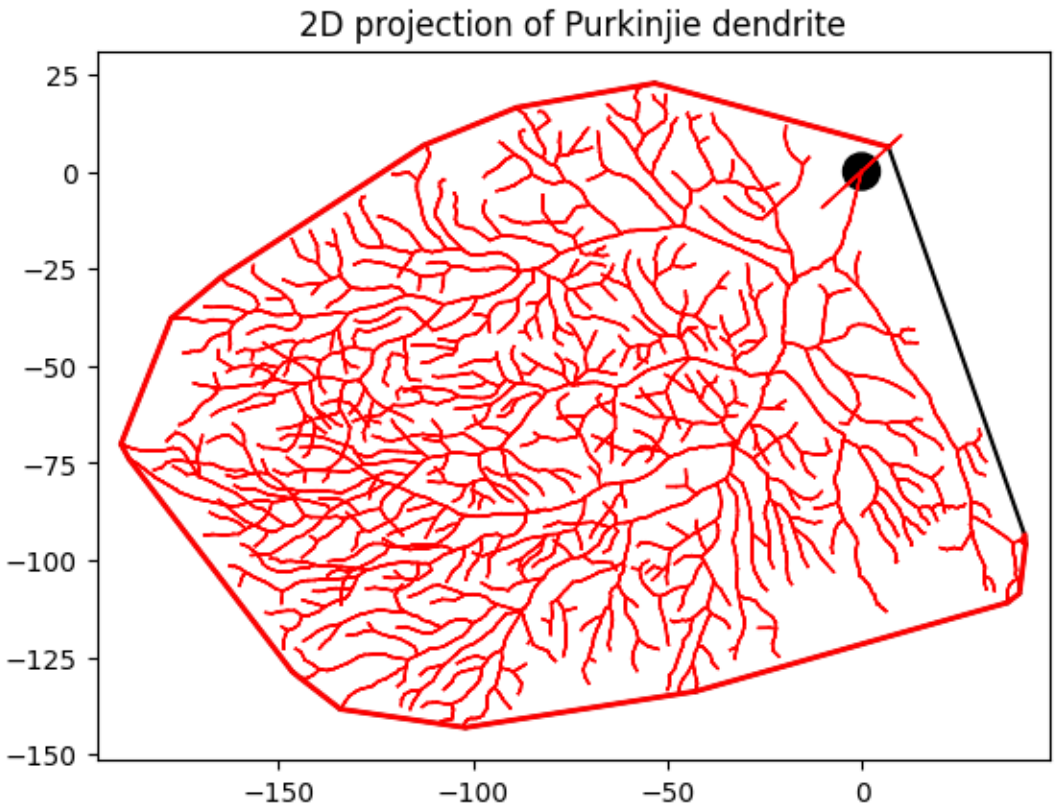

Figura 5. 2D projection of the arbor of Purikinjie neuron

By changing the segment radius to $s = 2\mu m$, I got the 2D projection of pyramidal/ Purikinjie dendrite(with spines) and these images can be seen in the jupyter notebool in annex.

b) *calculation the ratio between spine reach area and arbor area*

The calculation method of arbor area is similar to that of spine reach area. I summed up all the area of the rectangles that represents the projected segments and deleted the overlap area. And the Table 2 shows all the calculation results.

Tabela 2. spine reach/arbor area of pyramidal/Purikinjie neuron

	pyramidal neuron	Purikinjie neuron
arbor area(μm^2)	181666.3650	27740.5951
spine reach area(μm^2)	28422.2373	24877.2665
arbor area/spine reach area	6.3917	1.1151

3. Optimal Orientation Map

This question is actually an optimization problem, with the total wires's length being the target function and the distribution of neurons with different orientation selectivity being the decision variable.

3.1 Case of rule 1

According to rule 1, each neuron will establish equal numbers of connections(defined as n) with neurons of all preferred orientations. So no matter what distribution pattern all the neurons in the 2D lattice

take, the neuron at a given point on the 2D lattice will have the same wiring length, which is proportional to the sum of the distances from this point to all the other points on the 2D lattice and n. And it means that no matter what distribution pattern the neurons take, the total wires' length will be the same, which results that any pattern would be the solution to the optimization problem. From a perspective of evolution, this means each neuron with specific orientation selectivity has the same probability to appear at any point on this 2D lattice. This leads to a *Salt and Pepper* arrangement.

In mathmatics, the description above can be expressed by the following formula:

$$L = \sum_i n_i = \sum_i n \times \sum_{i \neq j} l_{ij} = n \sum_i \sum_{i \neq j} l_{ij}$$

In Eq.9, the L is total wires' length and l_{ij} is the distance between ith neuron and jth neuron.

3.2 Case of rule 2

According to rule 2, the connection number between one neuron and another will monotonically decreasing with the increase of difference between their favorite orientation. Now let's make some necessary definitions:

- p_i: the ith point on the 2D lattice
- lp_{ij}: the distance between p_i and p_j
- n_{ij}: the connection number between p_i and p_j
- $c_{\Delta\theta}$: the connection number between two neurons with the difference between their favorite orientation to be $\Delta\theta$
- N: the total number of points on the 2D lattice

Then we have,

$$L = \sum_{i \neq j} lp_{ij} n_{ij}$$

And we know that, lp_{ij} takes its value from $[lp_{12}, lp_{13}, lp_{14}, ..., lp_{(N-1)N}]$ and the latter is a specific list. Also n_{ij} takes its value from $[c_{\Delta\theta_1}, c_{\Delta\theta_1}, ..., c_{\Delta\theta_m}]$ in which the $c_{\Delta\theta_i}$ will occur n times if there are n pairs of neurons with the difference between their favorite orientation to be $\Delta\theta_i$ and m is the total number of types of neurons with different orientation selectivity.

And we notice that the two lists shown above both have a length of $N(N-1)$ and are specific. It turns out that L is the sum of the products of all pairs consisting of two elements from each list.

According to the **rearrangement inequality**, for two sequences $a_1 \leq a_2 \leq ... \leq a_n$ and $b_1 \leq b_2 \leq ... \leq b_n$,

$$a_1 b_1 + ... + a_n b_n \geq a_1 b_{\pi(1)} + ... + a_n b_{\pi(n)} \geq a_1 b_n + ... + a_n b_1$$

where $\pi(1), \pi(2), ...\pi(n)$ is any permutation of $1, 2, ...n$.

So in order to minimize the total wires' length, we need those neurons with more connection number to be more closer to each other, as is shown in the *Icecube* arrangement.

4. Acknowledgement

The successful completion of the work would be impossible without the help of ChatGPT and the following blog post. calculation of overlap area of two rectangles