ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Joyce Abou-Jaoude

Internship at Blue Brain: 18/06/2007 – 22/09/2007

Introduction:

The main concern of my internship was to improve the placement of neurons in their respective layers, in a way that satisfies the multiple biological constraints. I also made comparative plots for unrepaired, unraveled and repaired cells, to observe how the repair is affecting the placement. Moreover, I plotted histograms that show the differences between repaired neurons and clones, to see again how clones affect the placement. Finally, a consequential goal was to find the most precise layer boundaries.

Documentation:

Comments on Matlab Functions written:
function newData
provides four plots: plot 1: plot of layer boundaries (given by Sonia and
Rodrigo on September 6 2007)
plot 2: same plot with layer boundaries interpolated,
and with additional lines showing the
orientation that needs to be taken to
calculate layer thicknesses
plot 3: shows correlation between layer thicknesses,
and the correlation with the total height
plot 4: Spearman correlation

function orthogonalLine
returns slope of "best" orthogonal line to a set of lines
In this case set of lines is bottom line and layer1 upper boundary

function getCorrelation
returns an image showing the correaltion between the layer thicknesses
returns a second image showing the Spearman Correlation

function getIntersection
Returns intersection between layer and all orthogonal lines
Variable used under the form y-mx=b
Coord obtained under the form= [y x]

function getThickness
Finds points of intersection between layer boundaries and orthogonal
lines
Returns layer thicknesses

function lineEQ
given coordinates of two points, it returns the equation of the line joining those two
points

function getPlacementHintDnewrep
Reads the morphology parameters file and extracts the parameters from it and
then assigns an index to the neuron in the neuronDB.dat file. The index
varies between 0 and 1 when an index can be assigned otherwise it is left
to -1
The placement is done based on the dendrite height.

function aboveConstraintNeuronsD
identifies neurons with dendrites crossing their respective upper boundary
Format of output file: name   layer   type   excess

function axoneAbove
This function prints in a file the neurons which have an axon higher than the dendrites
The output file is as follows : neuronName, type, difference between Axon and Dendrite
heights

function belowConstraintNeuronsDnew
identifies neurons with dendrites that do not reach their respective lower boundary
Format of output file: name   layer   type   difference

function binPlacement
function that will return the positions of the somas, i.e. the bin in which the neuron
should be placed

function drawLayers
function that draws the layer boundaries

function getConstraint
returns the pia, now defined as the highest dendrites of Layer 5 cells, with their somas placed at the bottom of the layer, plus a bin height.

function getCorrelationAxonDendrite
pass as input the Morphology file and obtain a plot of the Dendrite Height
versus the Axon Height, and a second plot with a different color for each
layer cells

function getDendriteHeigths
Function that plots histograms showing the maxDendriteHeights for each type of neurons
Done to compare dendrites heights for clones and repaired cells

function getDifferenceAxonDendrite
function that plots histograms showing the difference between Axon
Heights and Dendrite Heights for each type of neurons

function getLayerDefinition
defines Layer Boundaries

function getMaxBin
returns the maximum possible bin number of neurons depending on the location of upper boundary

function getMinBinnew
returns the minimum possible bin number of neurons depending on the location of lower boundary

function getMorphIndices
returns the corresponding index in the morphology file for any index in the neuronDB file

function getMorphParameters
Writes the Morphology Parameters of all h5 files in a directory to a file
Function takes as a parameter the directory path where the h5 files are (sourcePath) , and then loops for it to get all the h5 files, does the analysis on the neurons and stores the

results in the file
MorphParameters.txt in the directory specified by outputPath
The format of the output file is as follows
fileName maxHeightAxon maxDepthAxon maxHeightDendrite maxDepthDendrite maxRadiusAxon maxRadiusDendrite maxDiameterDendrite maxDiameterAxon

function myupdatefcn
allows displaying name of neuron after clicking on point of graph

function myupdatefcntwo
allows displaying name of cell after clicking on a point in figure, used for figure with multiple simultaneous plots

function location2
returns neuron indices of specific type in specific layer

function getTypes
function that returns all neuron types

Placement Hints:

Assumptions used : - Each layer is divided into 10 bins.

- Cells used are the ones unraveled and repaired by Haroun (old repair)

Related functions: getPlacementHintDnewrep , aboveConstraintNeuronsD , belowConstraintNeuronsDnew, binPlacement, drawLayers, getConstraint, getLayerDefinition, getMaxBin, getMinBinnew, getMorphIndices, myupdatefcntwo, location2, getTypes.

Placement that was done initially:

- Pia is equal to maximum Dendrite heights of Layer 1 cells.
- Only constraint on neurons is that their AXONS should not cross the pia --> determine maximum allowed bin in layer to satisfy this condition
- Neurons are then placed randomly in allowed bins
- Placement only for excitatory cells

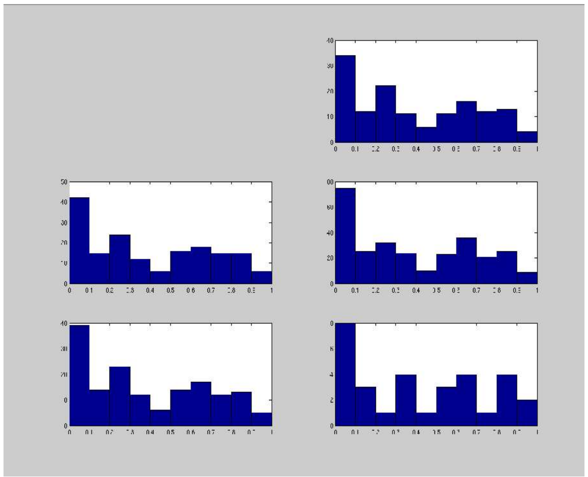
Steps done to improve placement:

Improvement 1:

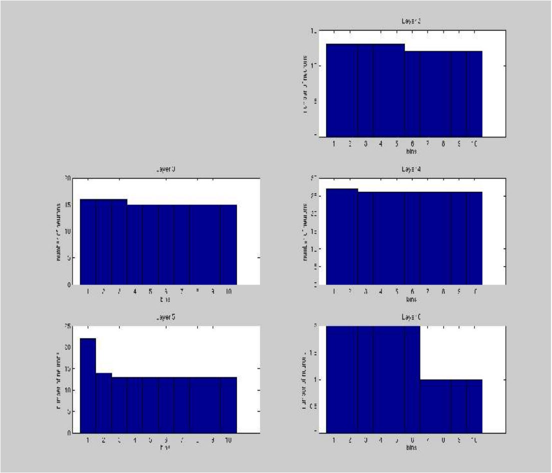
- Additional Constraint: Neurons should have a uniform distribution
- Introduced a counter for each bin, that keeps track of the number of neurons in each bin --> place neurons in the allowed bin which has the smallest counter -->counters updated accordingly
- Remark: I placed the neurons in ascending order of their maxBin to insure the most uniform distribution possible (ie place the neurons with the smallest maxBin first).
- Changes in the binPlacement function.

Comparative plots: Plots showing the neuron distribution in Layers 2 to 6.

Random placement



Uniform placement



Note: There are no excitatory cells in Layer 1.

Improvement 2:

- Change pia: equal to maximum Axons of Layer 1 cells.
- place neurons so that their DENDRITES do not cross the pia.
- Add black lines to indicate what neurons do not satisfy the maximum height constraint.
- Neurons that do not satisfy the maxBin Constraint are saved in a file called aboveConstraintDNeuronDB.txt, in which we find the name of the neurons, their layer number, their type, and by how much they are above the pia. (via the aboveConstraintNeuronD function)
- change in the getPlacementHint function.

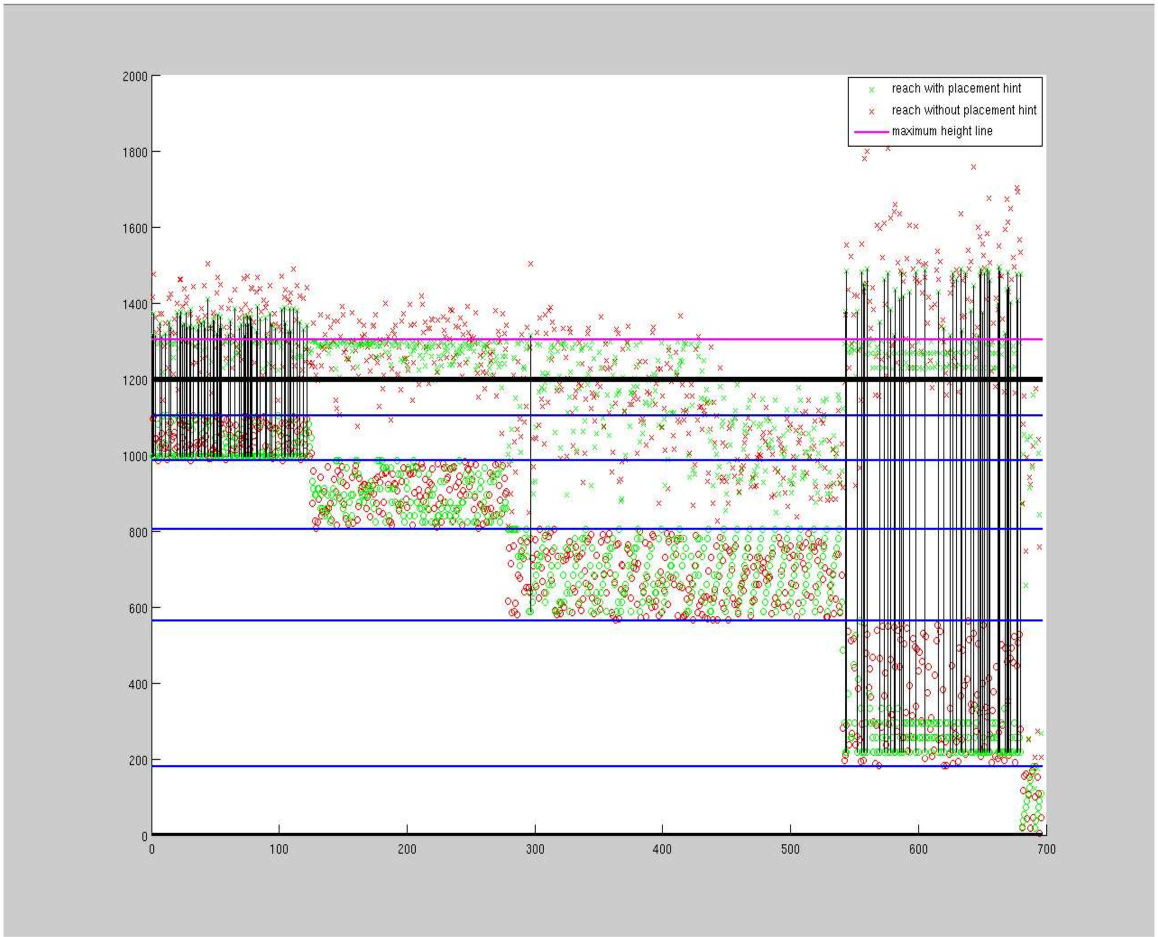


Figure showing the placement of the somas before the placement hints (red circles), after the placement hints (green circles), the maximum dendrite reach before the placement hints (red 'x'), and after the placement hints (green circles).

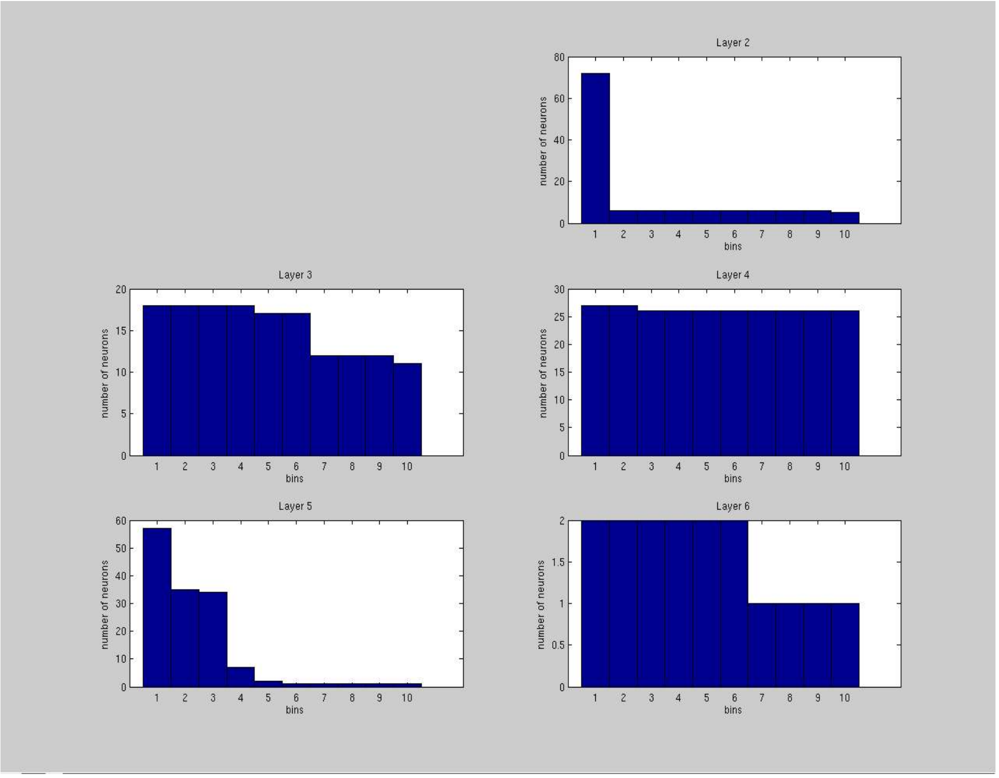
xaxis : arbitrary

yaxis: height

Magenta line represents the pia.

Blue lines represent the Layer Boundaries.

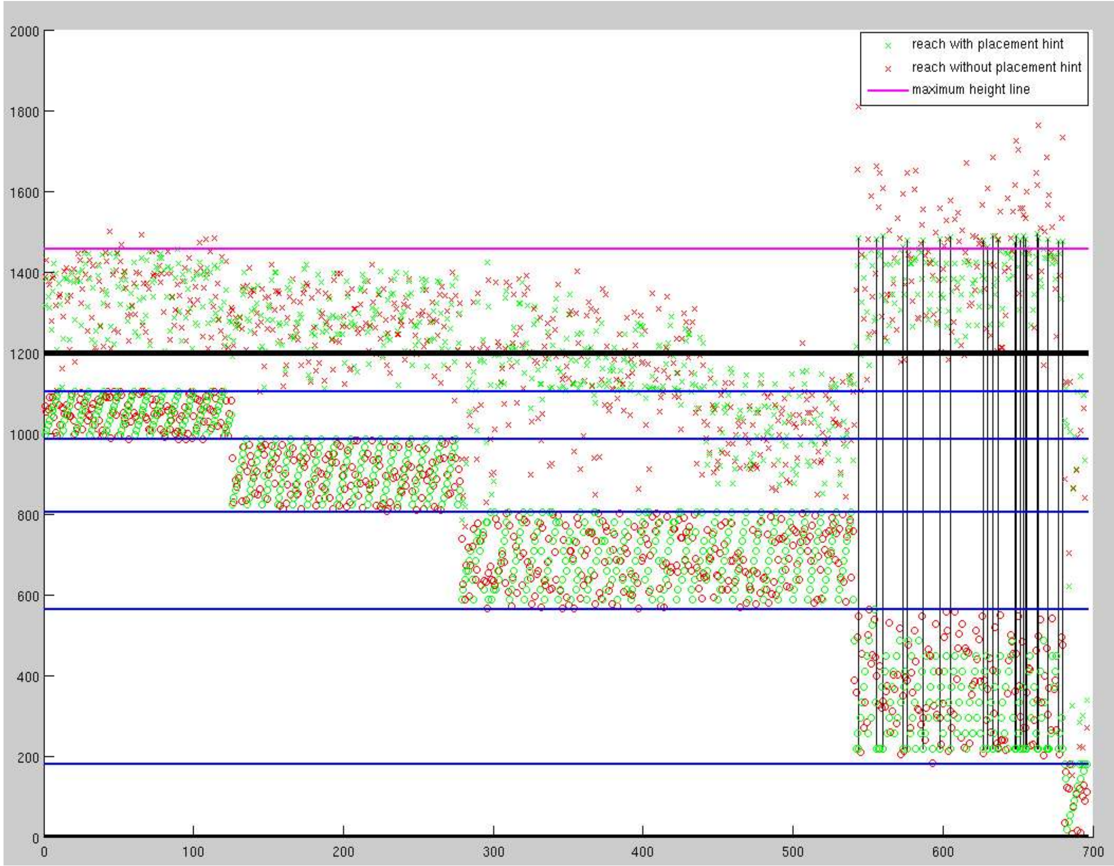
Histogram showing the distribution of neurons:

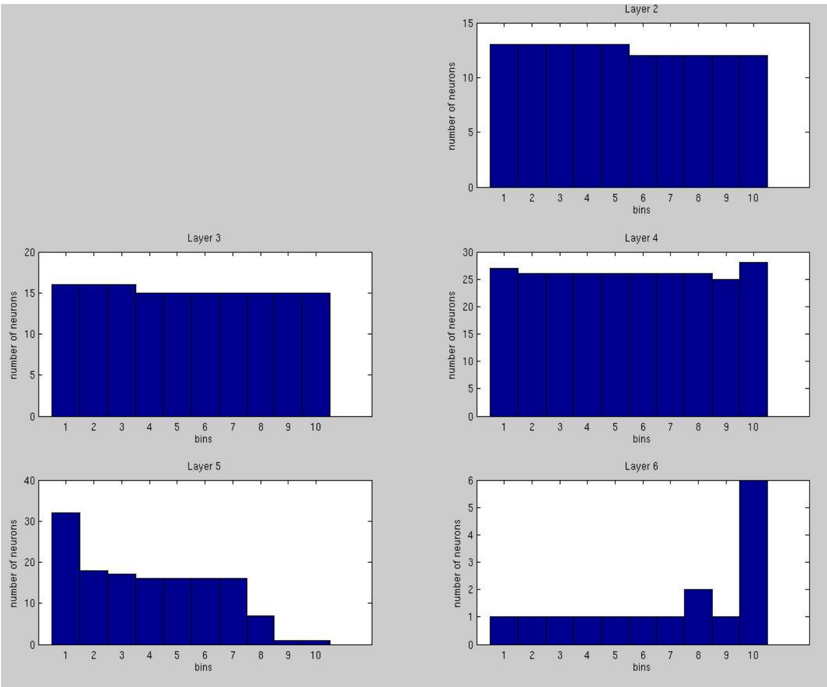


After obtaining Correlation plots (discussed later) and a phone conference with Ms.Wang, we identified new constraints to be incorporated in the code.

Improvement 3:

- Modify Pia: equal to Lower boundary of Layer 5 plus the maximum dendrite height of the Layer 5 excitatory cells -->higher pia.
- include new Constraint: for 6 types of neurons (L2PC, L3PC, L4PC, L5CHPC, L5CSPC and L6CCPC), their maximum dendrite height should lie between the lower boundary of Layer 1 and the pia.
- changes in getPlacementHint function, and in binPlacement (introducing a minBin)
- getMinBinnew and getMaxBin functions updated

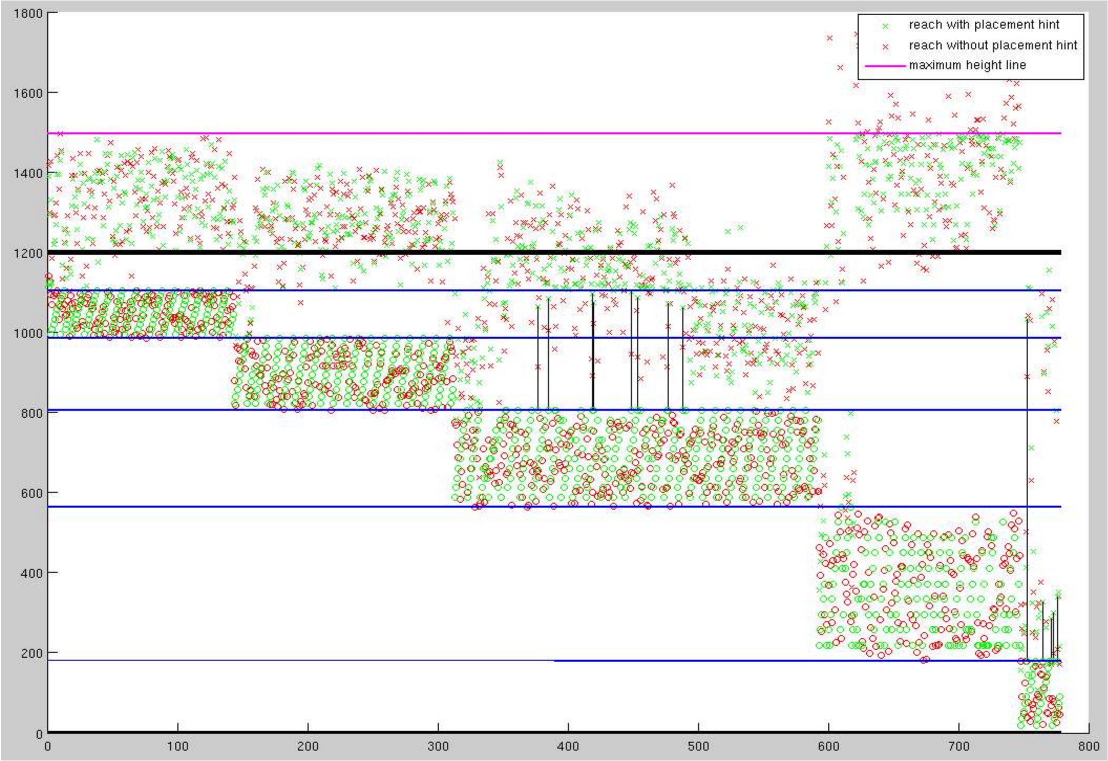

reach with placement hint
reach without placement hint
maximum height line
2000
1800
1600
1400
1200
1000
800
600
400
200
0
0
100
200
300
400
500
600
700

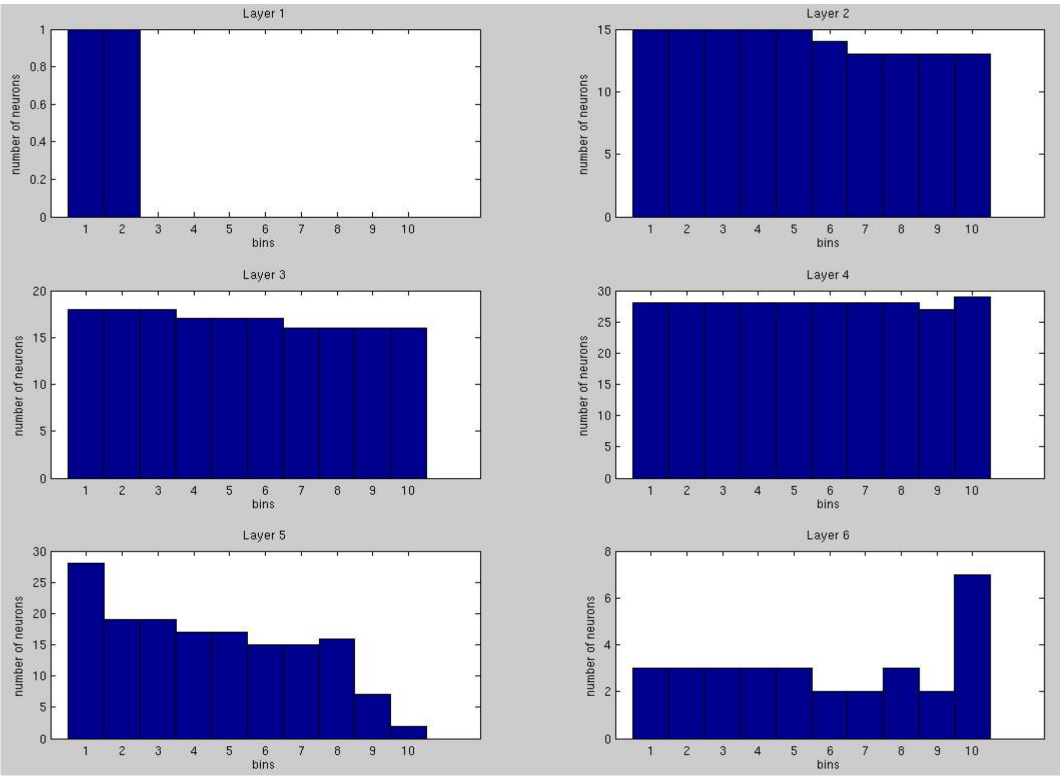

Layer 2
Layer 3
Layer 4
Layer 5
Layer 6
number of neurons
bins
1
2
3
4
5
6
7
8
9
10

Improvement 4:

- update pia to insure that no dendrites cross the maxHeight Constraint: pia equal to Lower boundary of Layer 5 plus the maximum dendrite height of all Layer 5 cells plus one bin Height.
- allow name of neuron to be displayed when clicking on the soma after placement hints (via myupdatefcntwo)
- use black lines to show which cells are below their constraint (cells belonging to one of the six types mentioned)
- placement for ALL cells (excitatory and inhibitory) such that their dendrites do not cross the pia.
- Neurons that do not satisfy the minBin constraint are saved in a file called belowConstraintDNeuronDB.txt, in which we find the name of the neurons, their layer number, their type, and by how much they are lower (via the belowConstraintNeuronsDnew function)
- changes in getPlacementHints function.

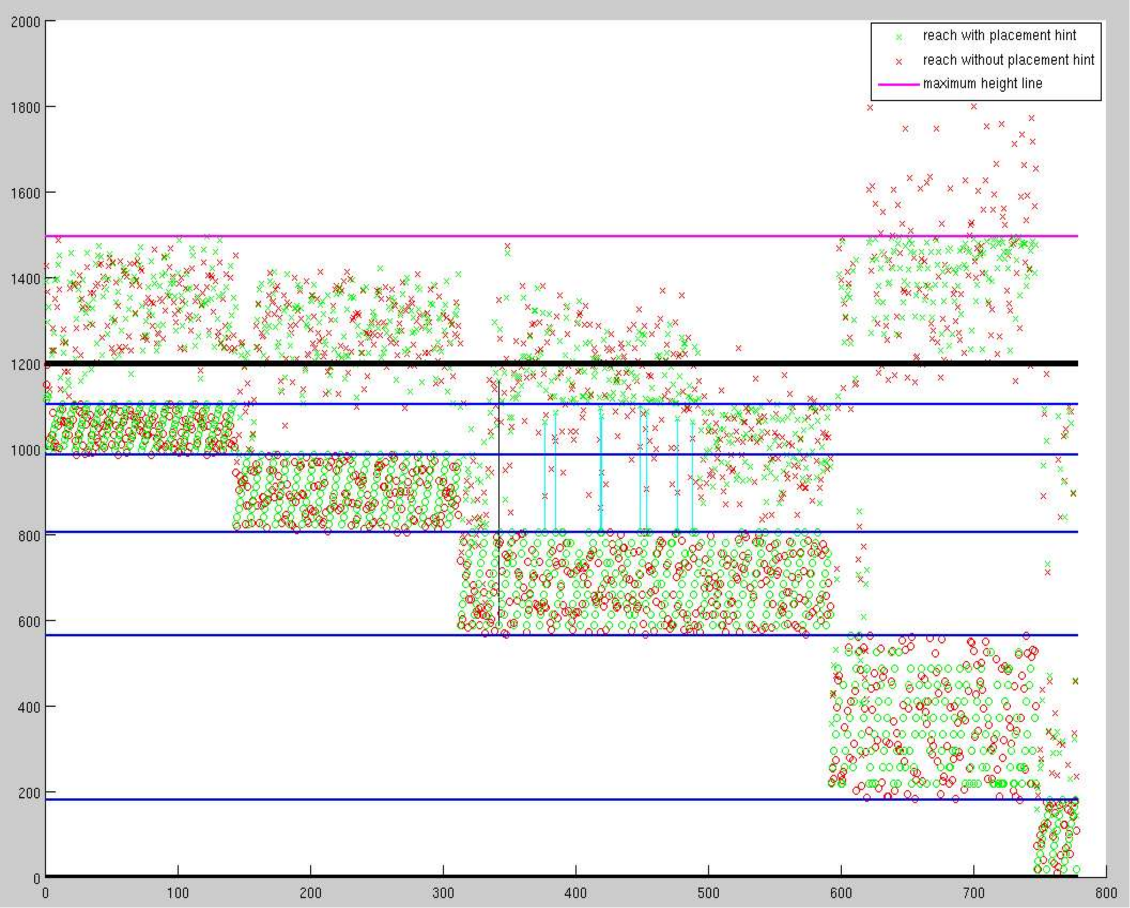
Assumption: A biological constraint implies that axons of all cells should not cross the pia too. However, this constraint was incorporated in the code only temporarily, then it was removed. The reason behind that is that the repair of the axons is still incomplete, it makes them too long, and this affected the placement by putting more cells in the bottom of each layer.

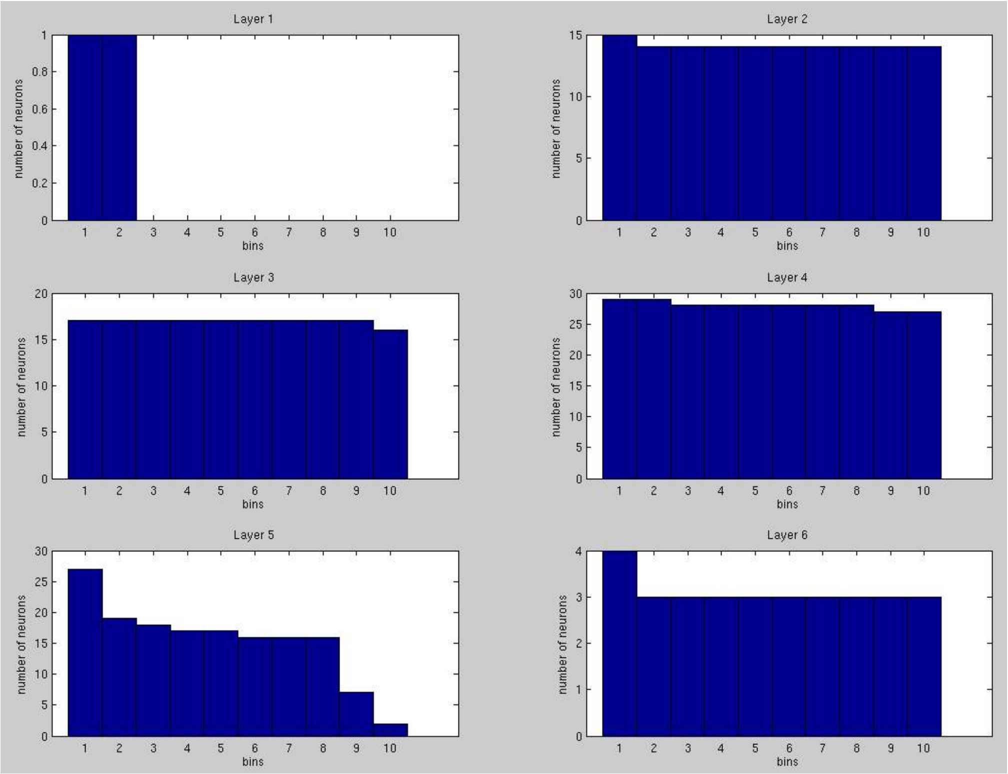

reach with placement hint
reach without placement hint
maximum height line
1800
1600
1400
1200
1000
800
600
400
200
0
0
100
200
300
400
500
600
700
800


Layer 1
Layer 2
Layer 3
Layer 4
Layer 5
Layer 6
number of neurons
bins
1
2
3
4
5
6
7
8
9
10

Improvement 5:

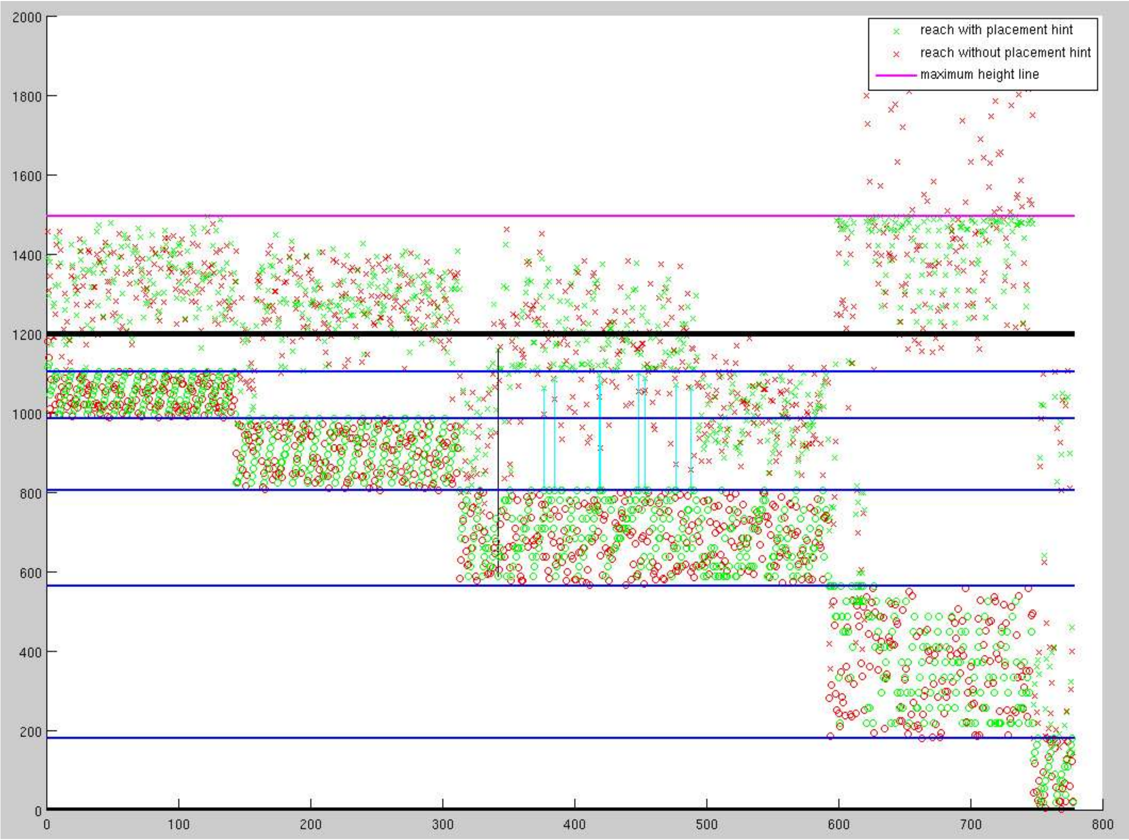
- New constraint: dendrites of L4SP neurons should lie between the lower boundary of layer 3 and the upper boundary of layer 2.
- Modification of previous constraint: the L6CCPC dendrites should be between lower boundary of Layer 5 and lower boundary of Layer 1. (For the other five types mentioned, the constraint did not change)
- Black lines represent above constraint neurons, cyan lines represent below constraint neurons.
- changes in the getPlacementHint function

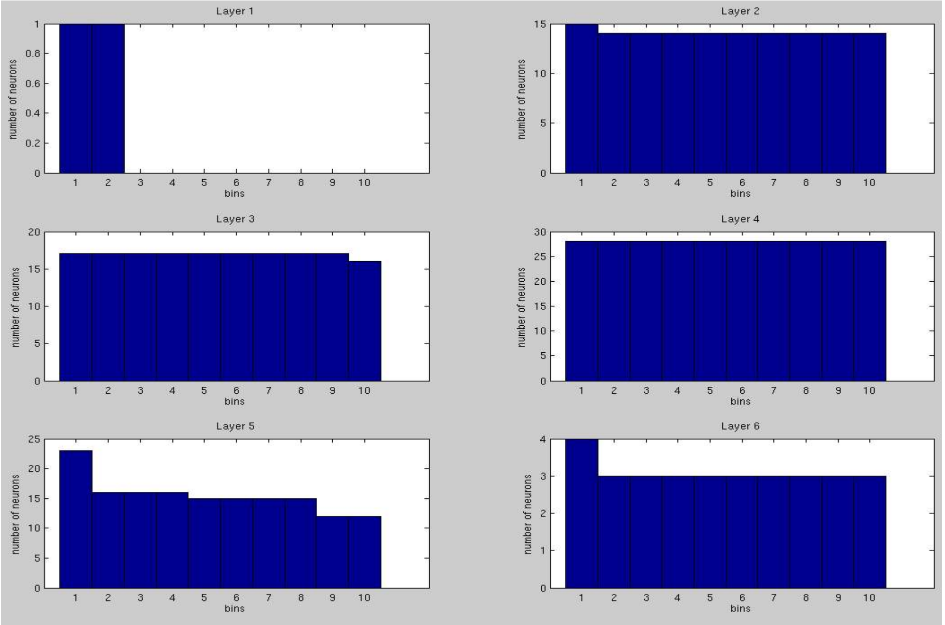




Improvement 6:

- Start by placing neurons who have the smallest range allowed to insure the best uniform distribution.
- binPlacement modified

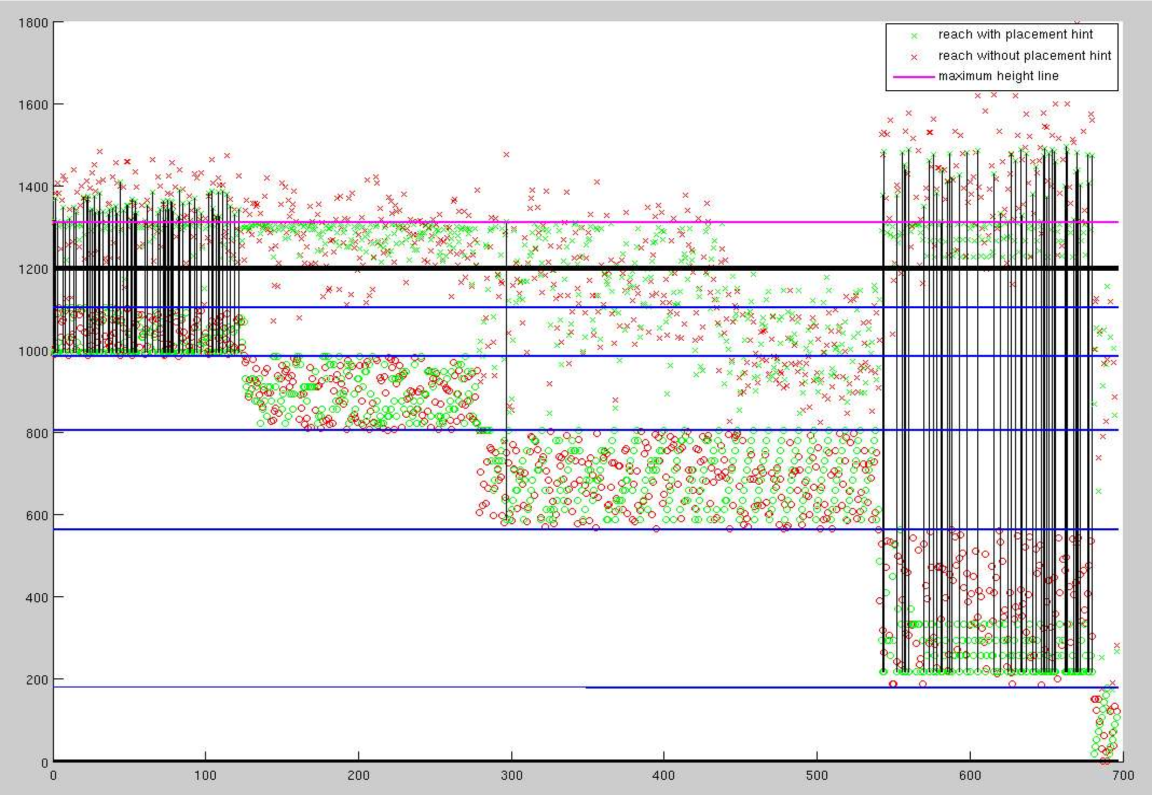

reach with placement hint
reach without placement hint
maximum height line
2000
1800
1600
1400
1200
1000
800
600
400
200
0
0
100
200
300
400
500
600
700
800

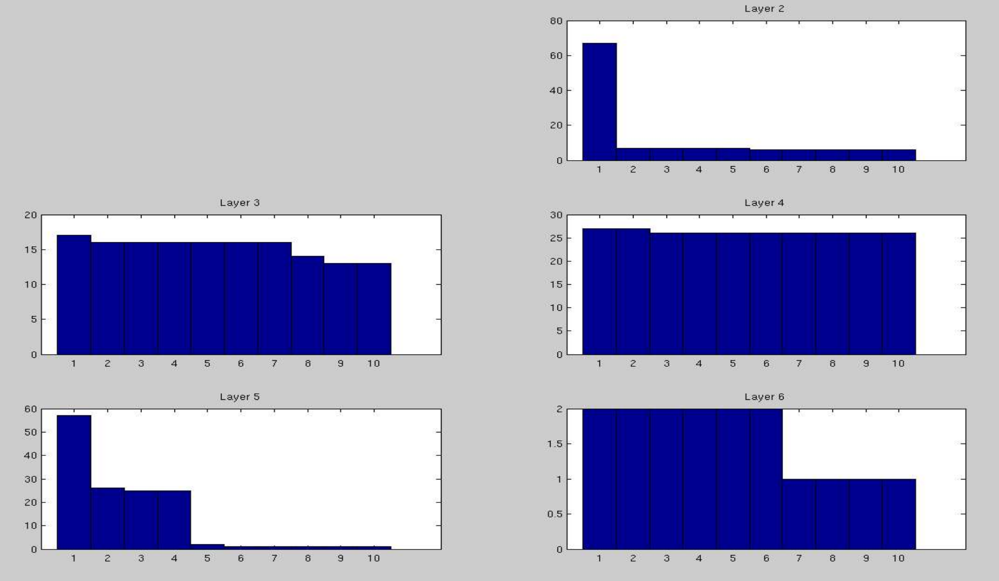

Layer 1
Layer 2
Layer 3
Layer 4
Layer 5
Layer 6
number of neurons
bins
1
2
3
4
5
6
7
8
9
10

Other trials:

Unsuccessful Trial:

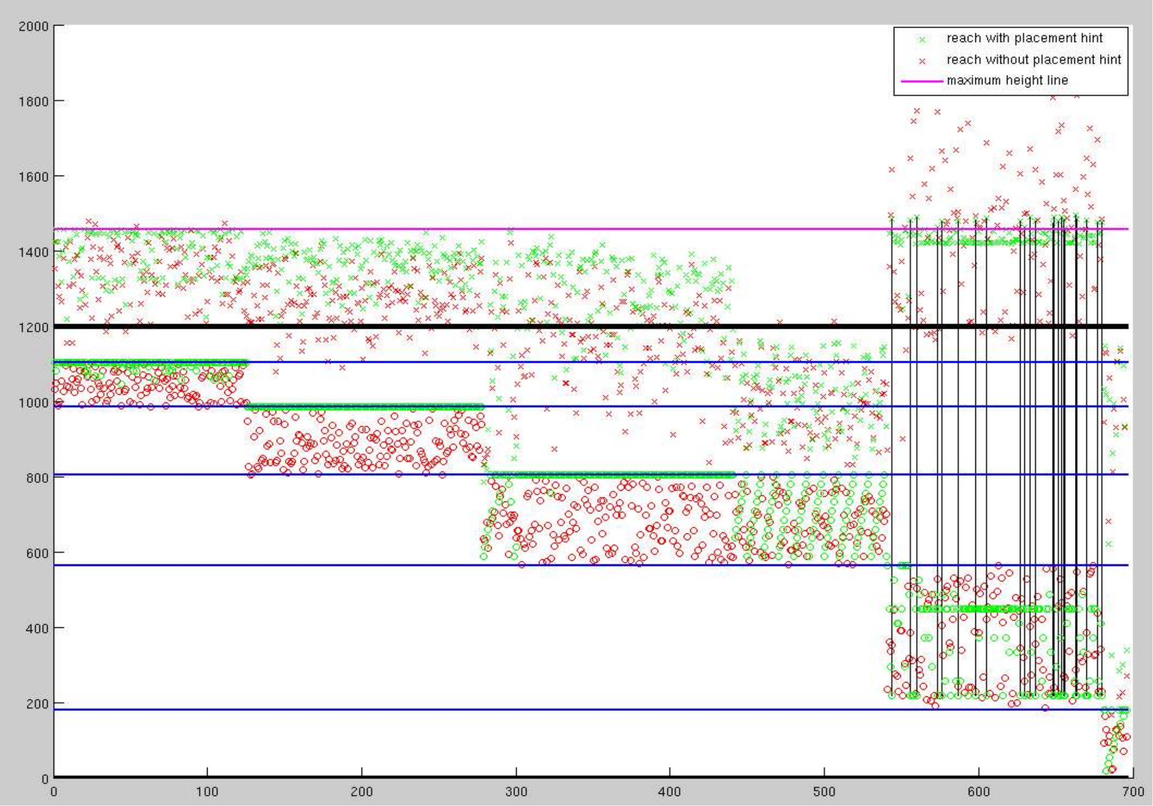
In order to obtain an average maximum height or pia, we placed all neurons at the middle of each layer, and calculated the constraint as the average of the sum of the maximum axon height and the layer height divided by two: (OLD: done last week of June)

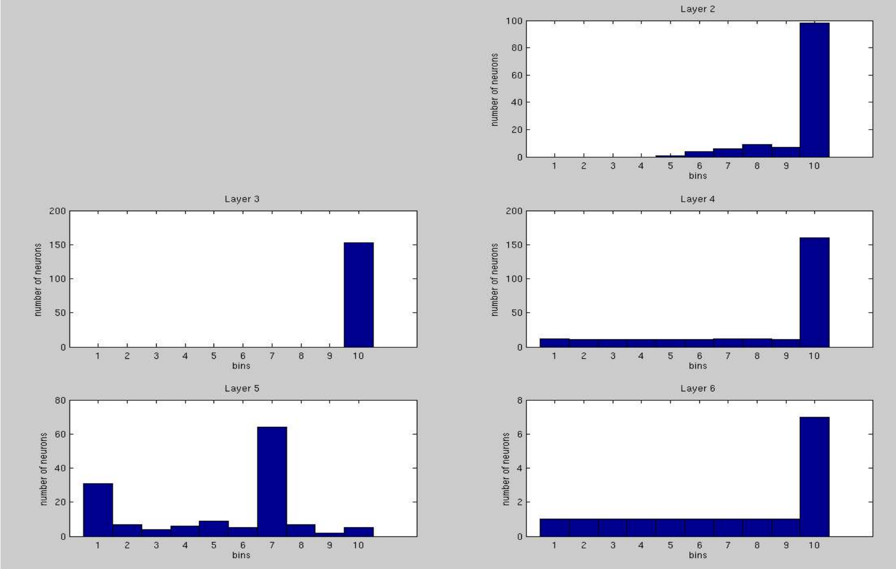




Wrong trial:

- Results obtained when the following constraint was incorporated: dendrites of 6 types of neurons must reach the pia: L2PC, L3PC, L4PC, L5CHPC, L5CSPC and L6CCPC.

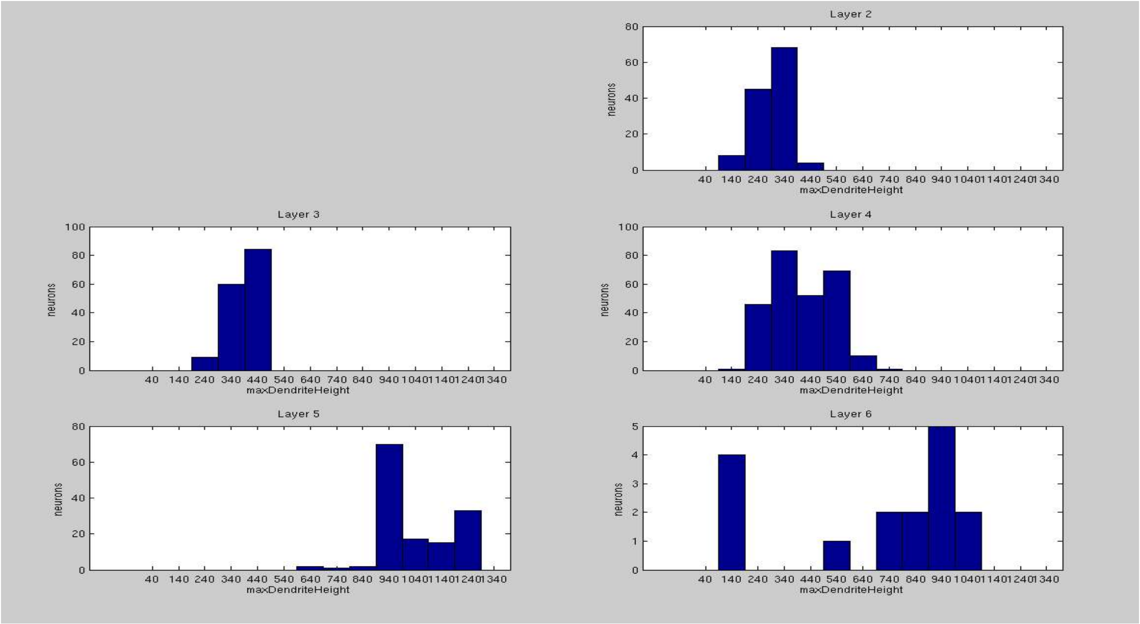




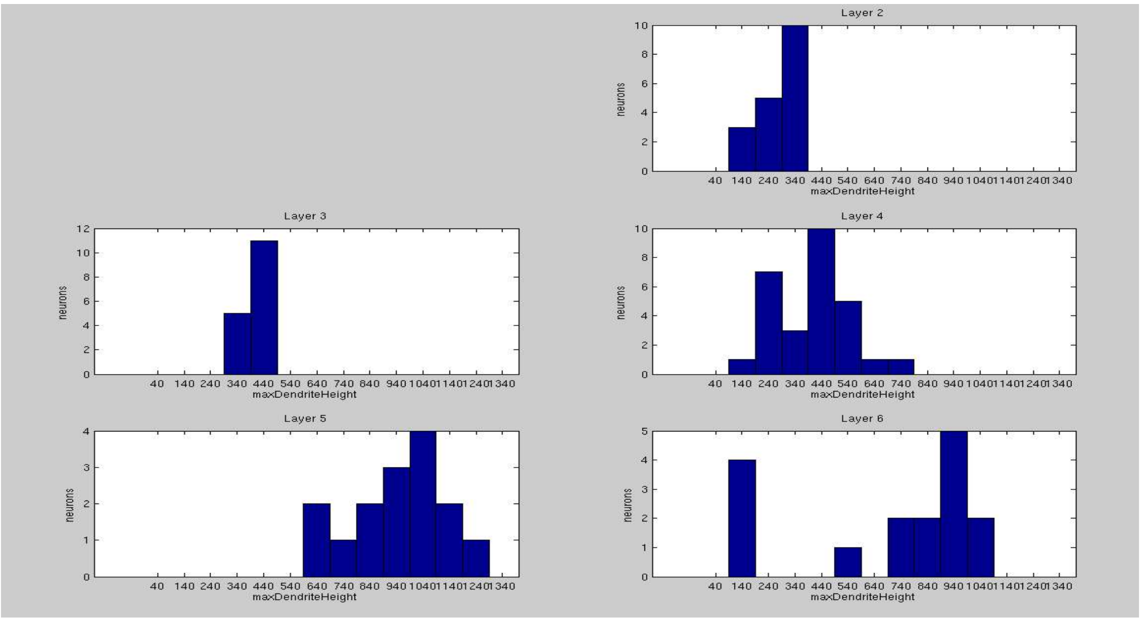
Clones vs repaired cells:

It was useful to see if the clones were the ones who were affecting the placement. First, i plotted histograms of dendrite heights for excitatory cells, with and without clones. Related function: the getDendriteHeigths function.

With clones:



Without clones:

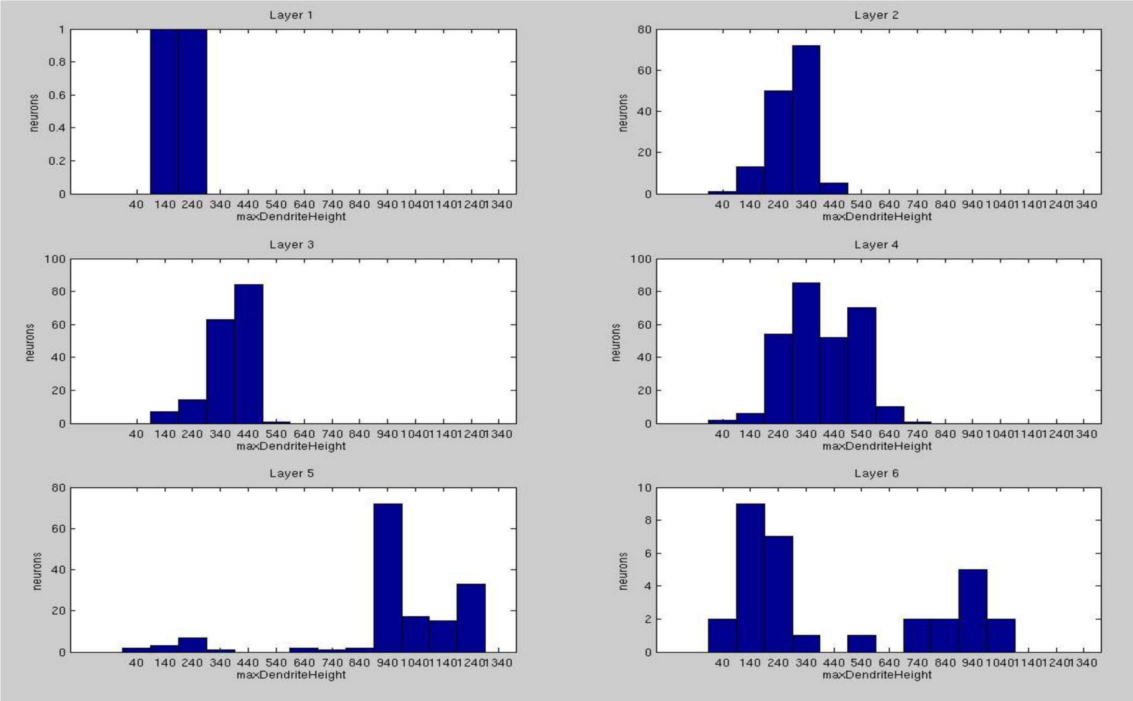


Remarks: - Most high dendrites in layer 5 are due to clones.

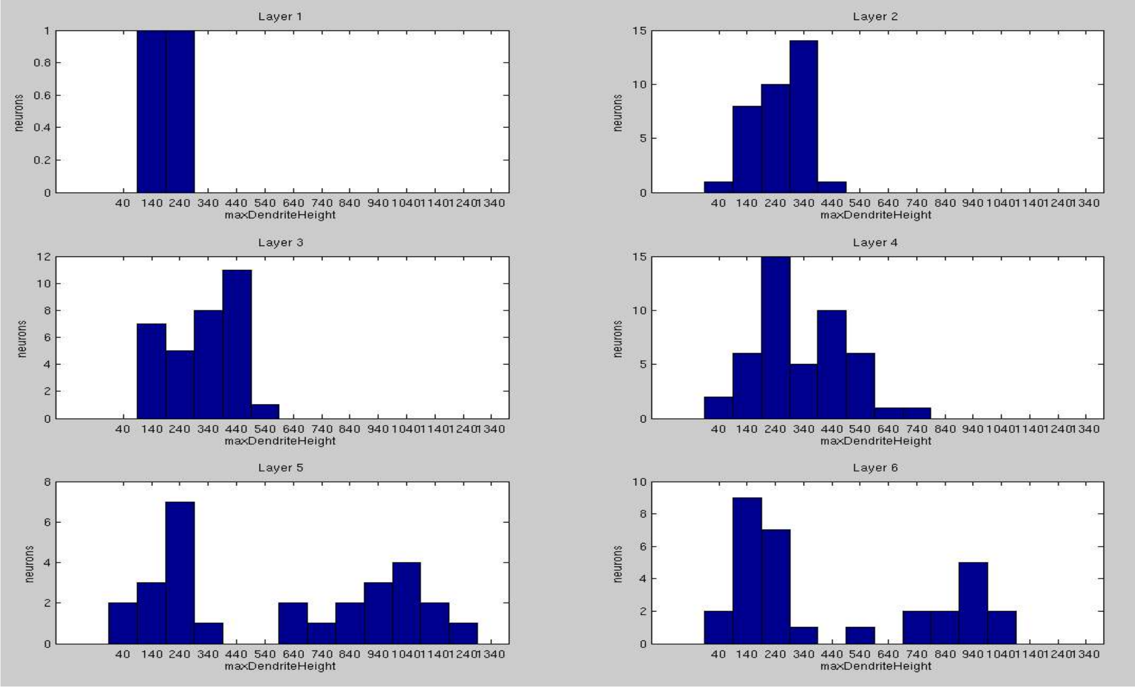
- Some short dendrites of layer 4 are also due to clones

For all cells: (excitatory and inhibitory)

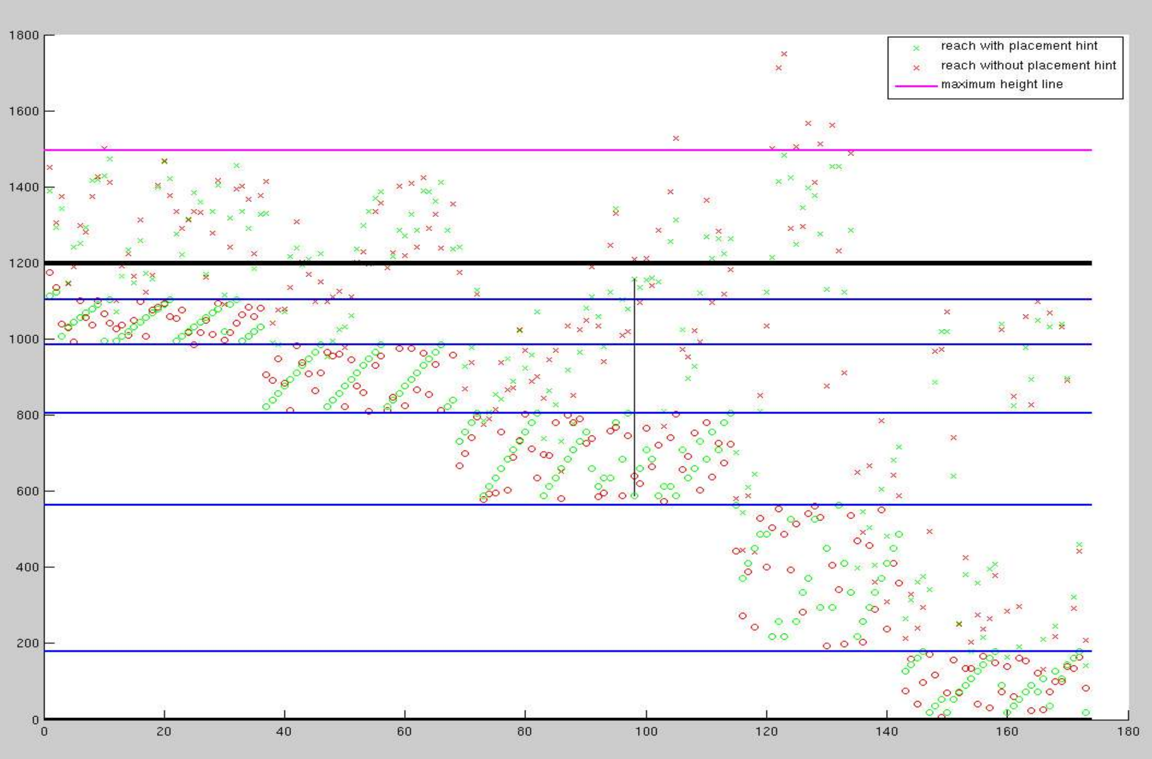
With clones:

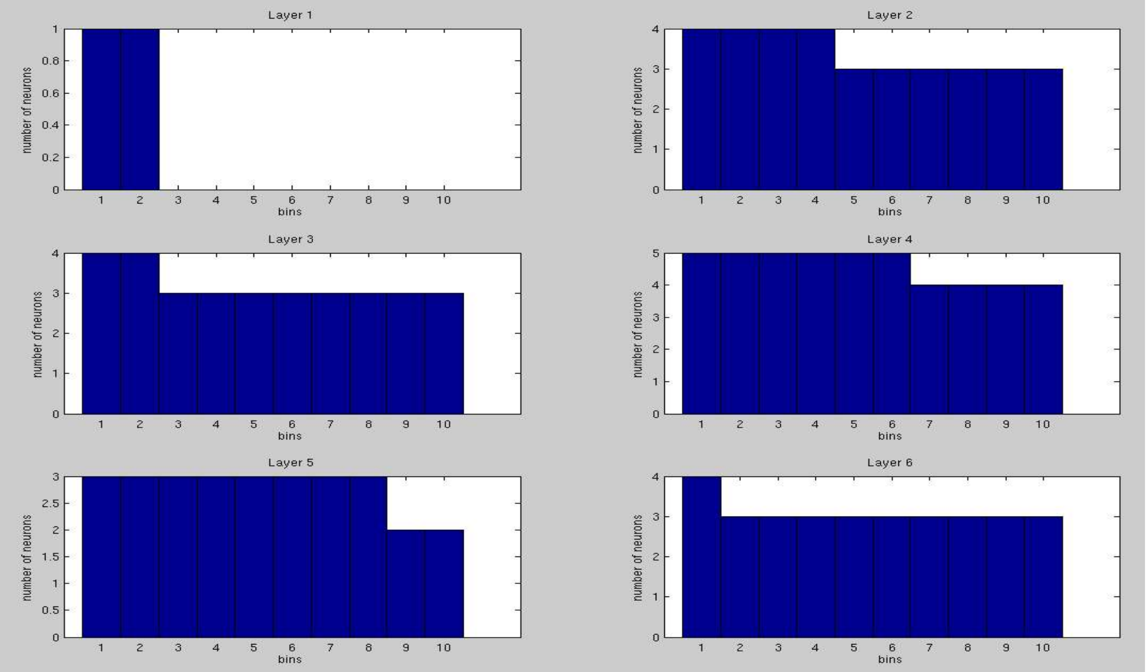


No clones:



Consequently, all previous graphs of the placement hints were plotted from cells without clones, to observe the new pia height. The following figures show the placements of all cells without clones for the case in which all constraints are included (Improvement 6)



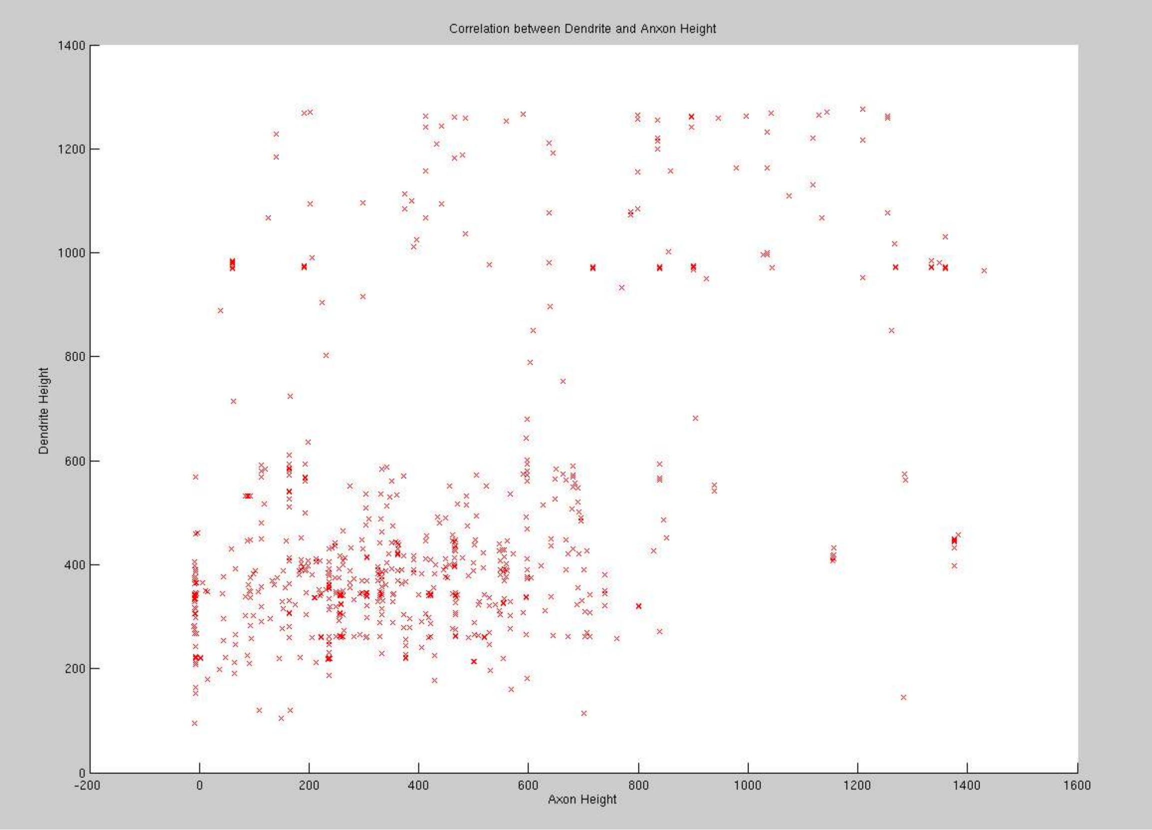


Remark: only one L4SP cell do not fit to its constraint and is above its upper boundary (lower boundary of Layer 1) --> black line

Repaired vs Unraveled vs Unrepaired:

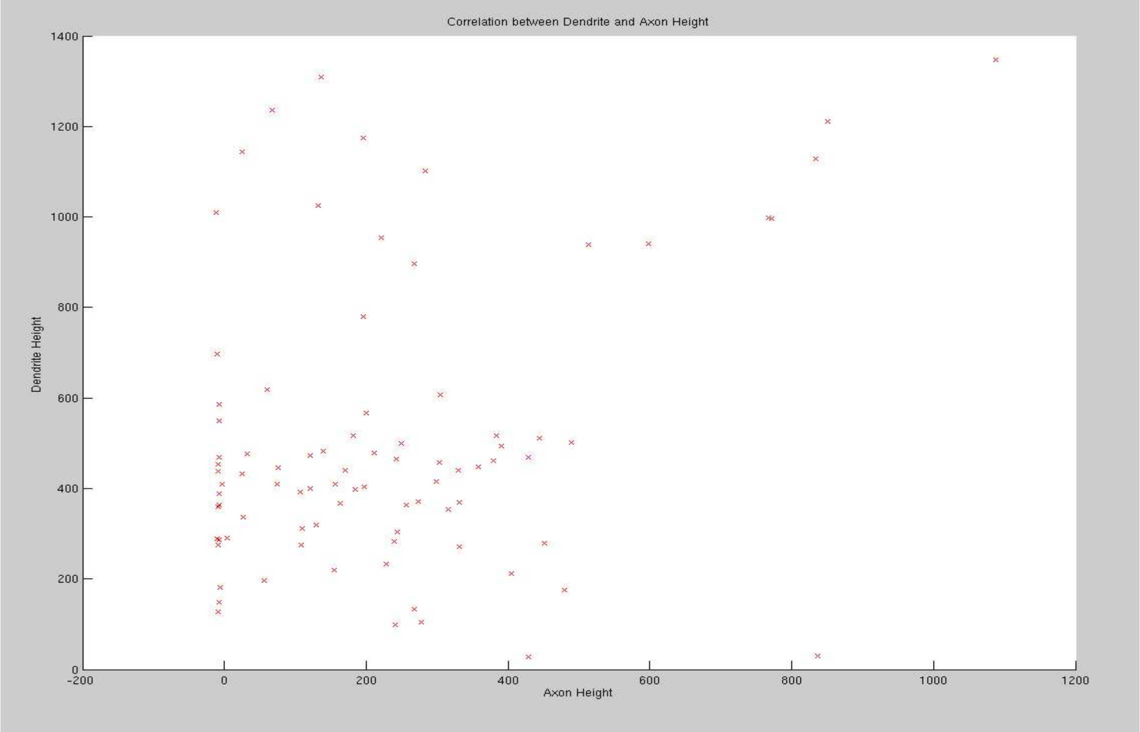
It was also useful to study the effect of the unraveling and the repair process on dendrite heigths, axon heights and consequently on the placement.
Related functions: getCorrelationAxonDendrite, myupdatefct, getDifferenceAxonDendrite, axonAbove.
First, I plotted correlation graphs showing the correlation between axon and dendrite heights for unrepaired, unraveled and repaired cells. Neither me nor Karthik had a complete database of the unraveled neurons. I used blueRepair to unravel neurons and save them in a database (via the getCorrelationAxonDendrite function).

Correlation for repaired cells:

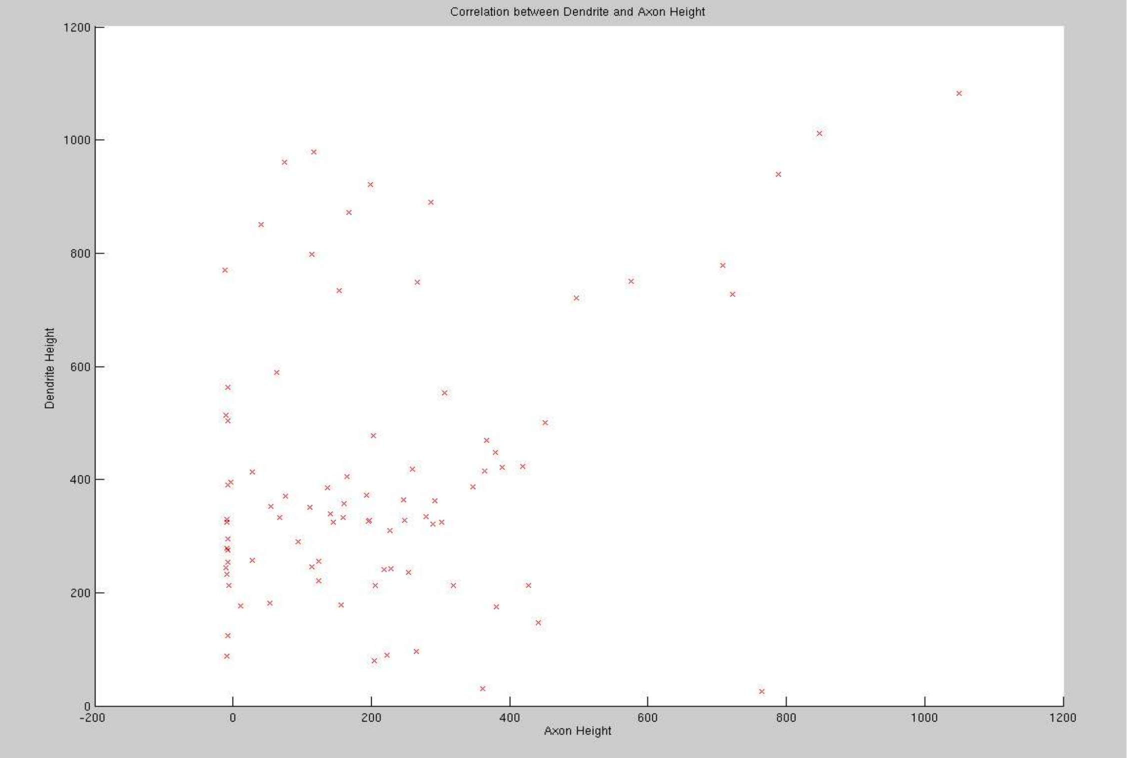


Same but with a different color for each Layer:

Correlation for unraveled:

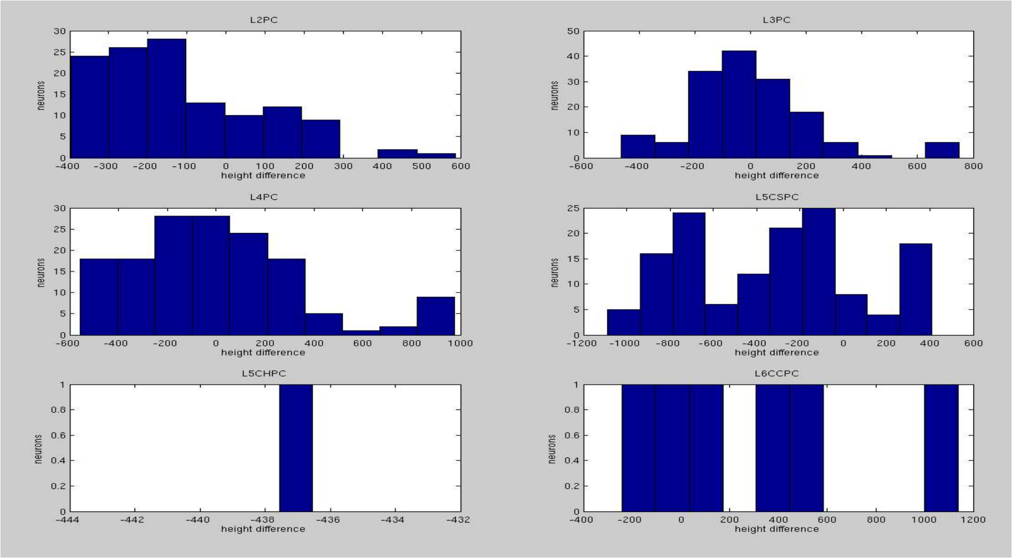


Correlation for unrepaired:

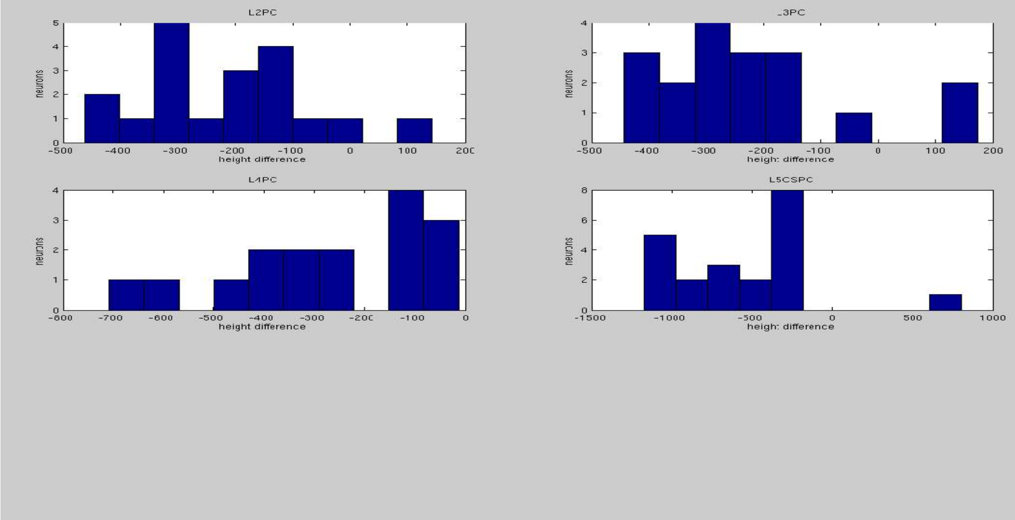


Using myupdatefct. it is possible to click on a point (neuron) in the figure and have its name displayed.

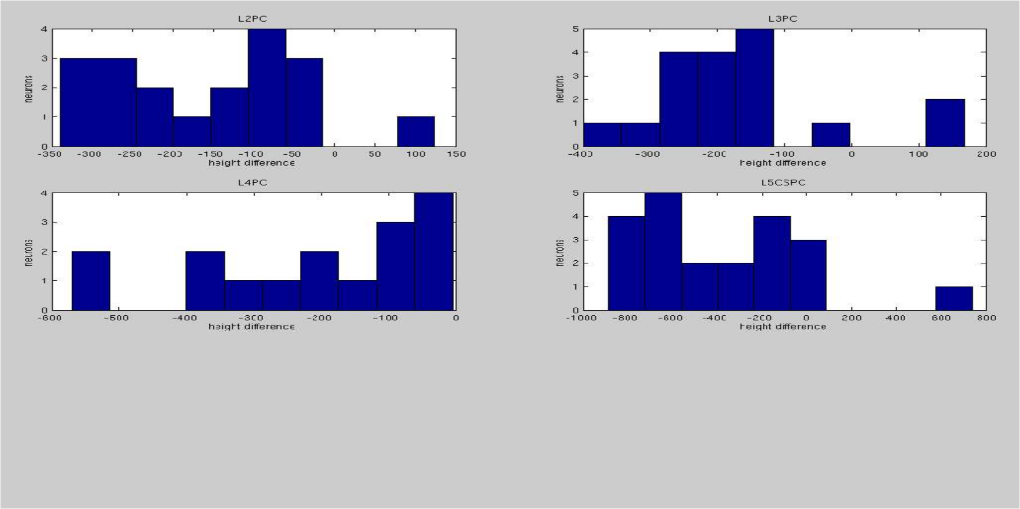
Remark: After the repair process, we find that some neurons have huge axons compared to their dendrites.
To verify this fact, we plotted the difference between axon and dendrite heights for the three categories (via the getDifferenceAxonDendrite function)
difference= Axon height – Dendrite height
For repaired:



For unraveled:



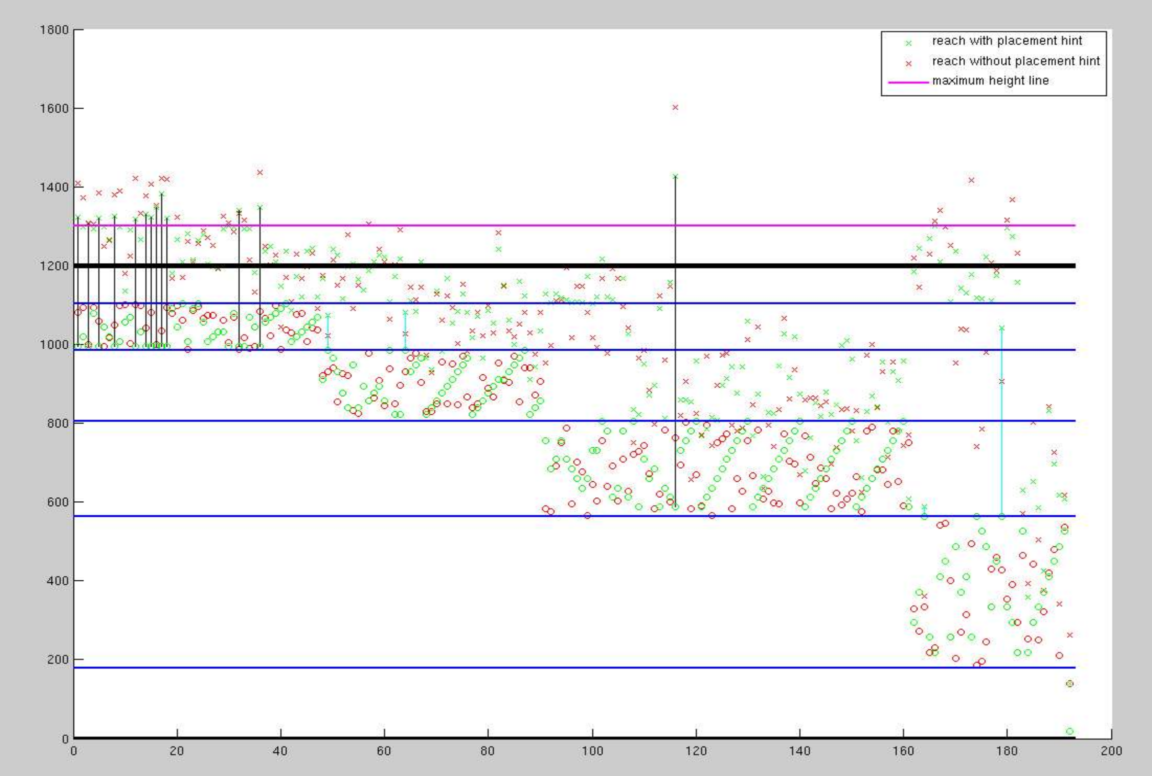
For unrepaired:

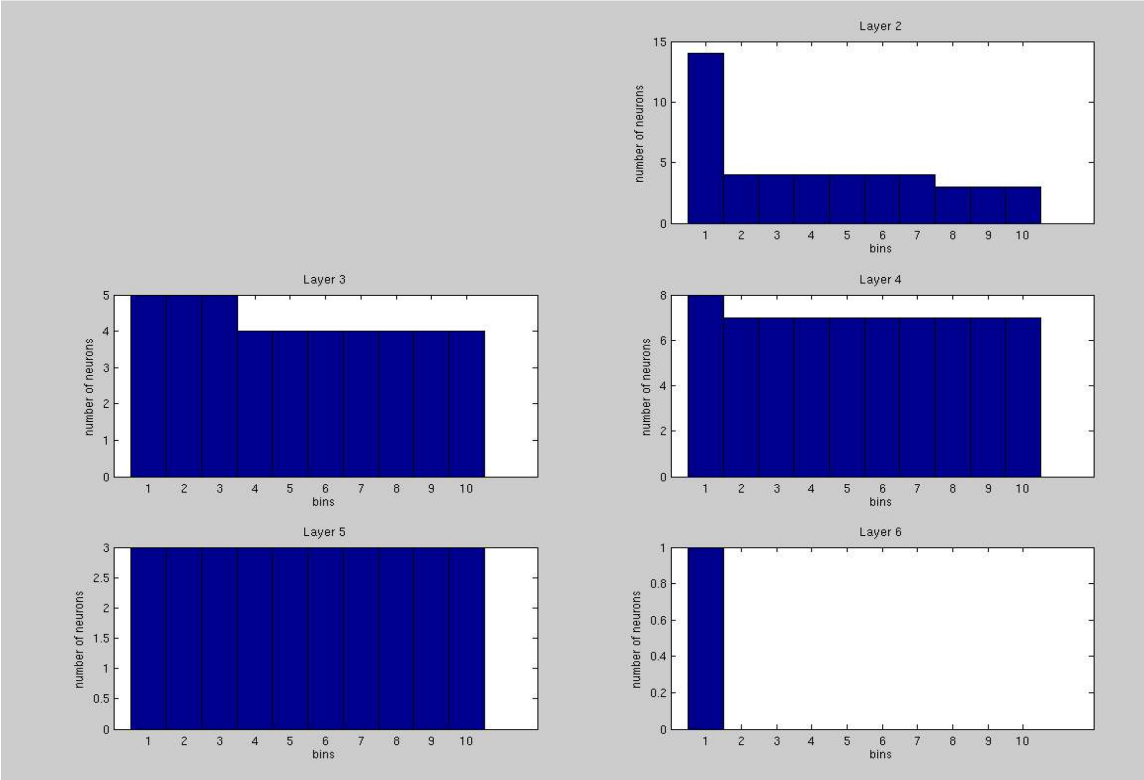


Remark: - As expected, very few neurons have a positive difference in both the unraveled and unrepaired case. This is not true however for the repaired case, in which we find many neurons with axons higher than dendrites.

- printed on 3 different files the names of neurons with axon higher than dendrite, with their type and by how much they are higher (via the axonAbove function)

Finally, we plotted the same placement hints figure for unrepaired cells. Again, after improvement 6:

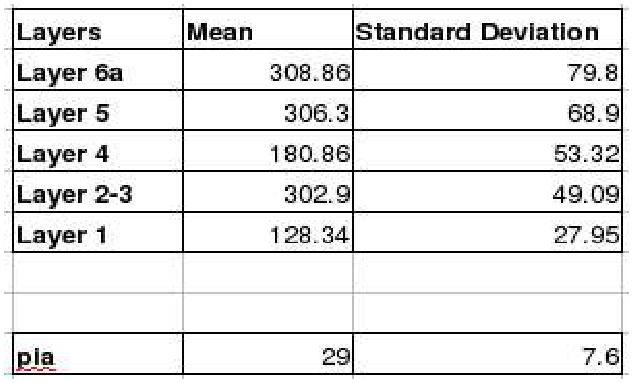




Remark: Pia is lowered considerably.

Layer Boundaries:

Related functions: newData, orthogonalLine, getCorrelation, getIntersection, getThickness, lineEQ.
We first obtained thickness measurements for all layers ,for the pia and for the total heights from Sonia.
After reading the excel file using matlab, I calculated the mean and standard deviation each layer and obtained the following table:



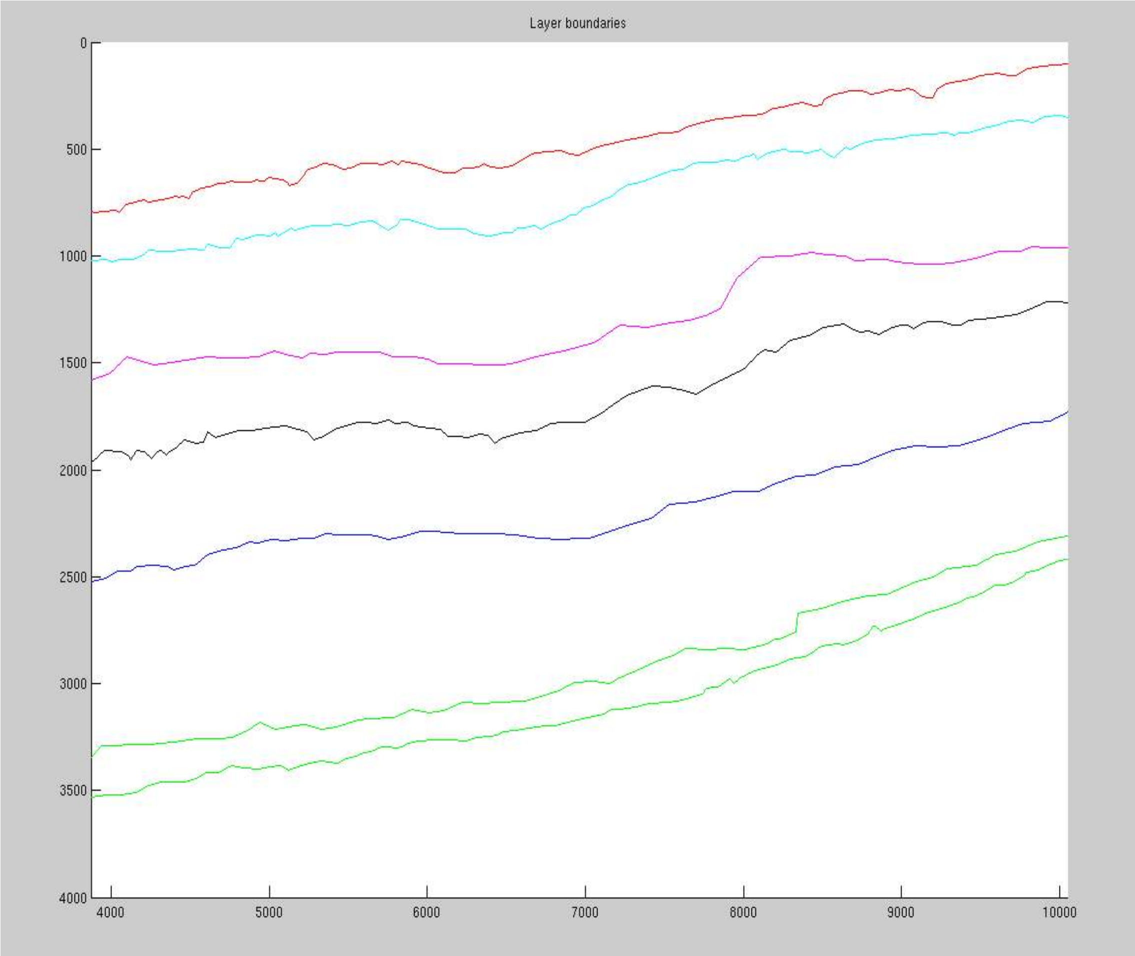
Layers	Mean	Standard Deviation
Layer 6a	308.86	79.8
Layer 5	306.3	68.9
Layer 4	180.86	53.32
Layer 2-3	302.9	49.09
Layer 1	128.34	27.95
pia	29	7.6

We then realized that the measurements taken were independent from each other (ie measurements for a layer not related to other layers) and thus incomplete.

Later, Sonia and Rodrigo gave us seven text files: bottom, upper boundary of Layer 6b, upper boundary of Layer 6a, upper boundary of Layer 5, upper boundary of Layer 4, upper boundary of Layer 3/2, upper boundary of Layer 1.

In each file, we find the xy coordinates of consecutive points on the boundaries.
The start and end points weren't the same for all layers, this is why I needed to take the values corresponding to the shortest layer only.

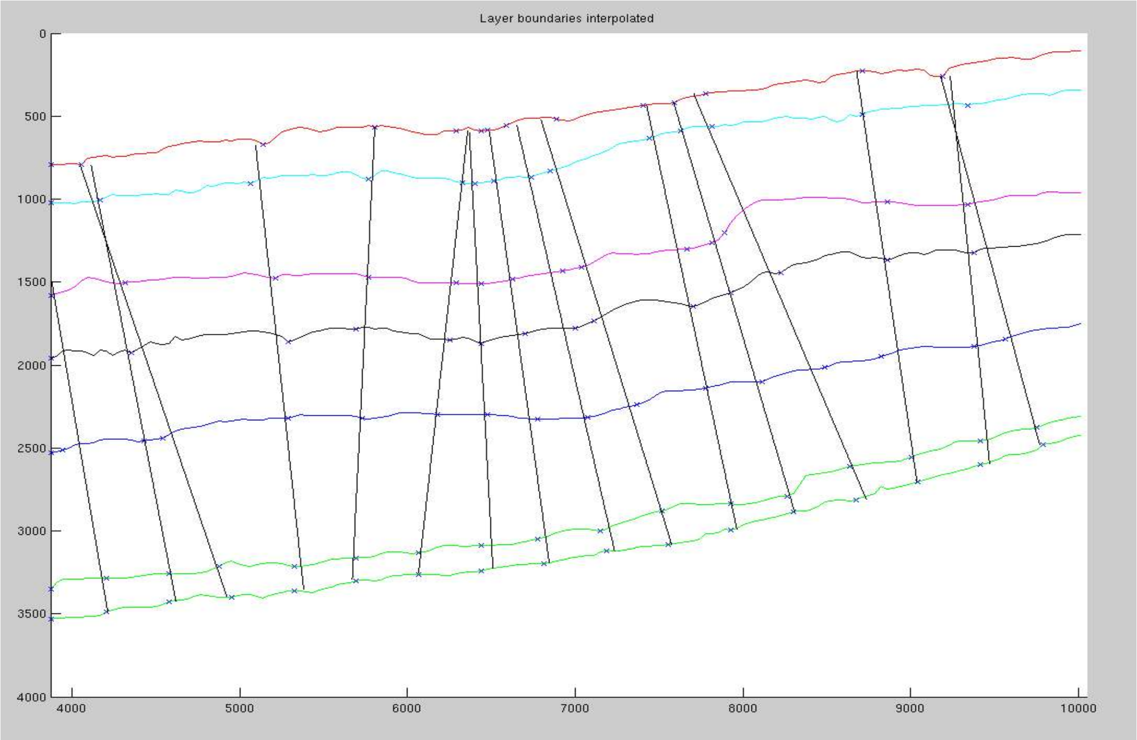
I plotted the boundaries and obtained the following (via the newData function):



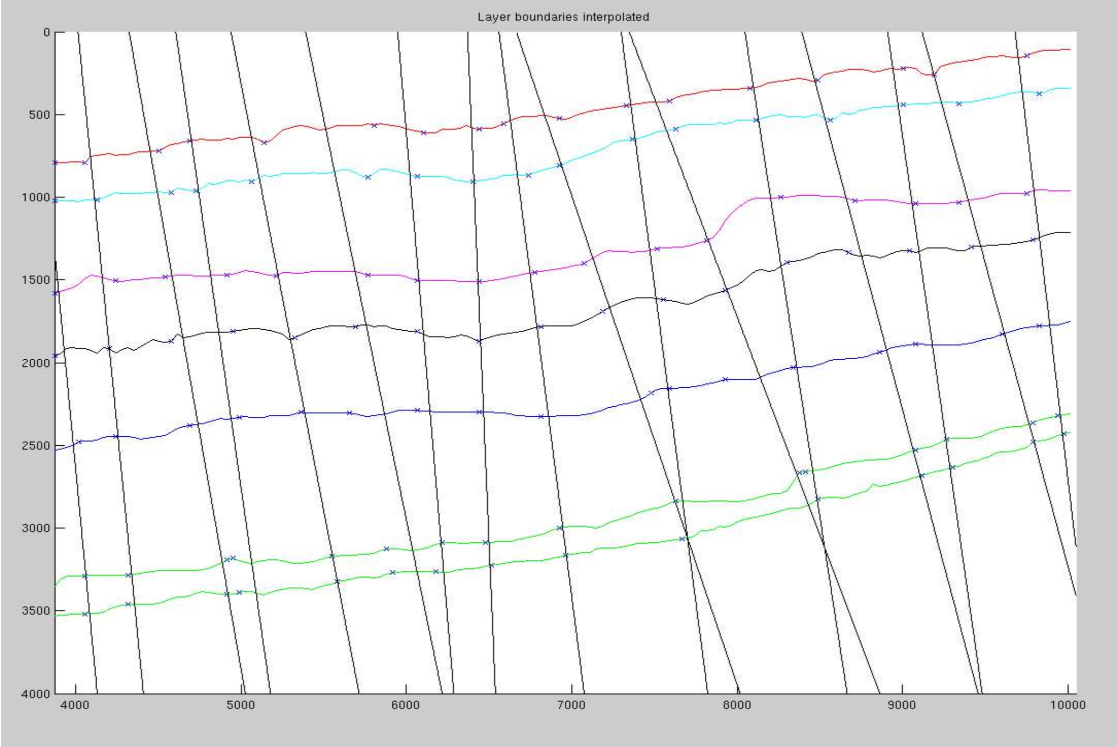
The next step was to do a cubic interpolation of those values, and to calculate layer thicknesses. Multiple ways were used to see in what way or orientation should the layers be “cut”.

The first way was to take equally distanced points on the bottom boundary, and to choose the point on the next layer as being the closest to this point, then repeating this process to all layers (ie finding the minimum distance from the points on the next layer to a specific point a a given layer). Then the points would be fitted linearly(via the newData function).

The figure obtained was:

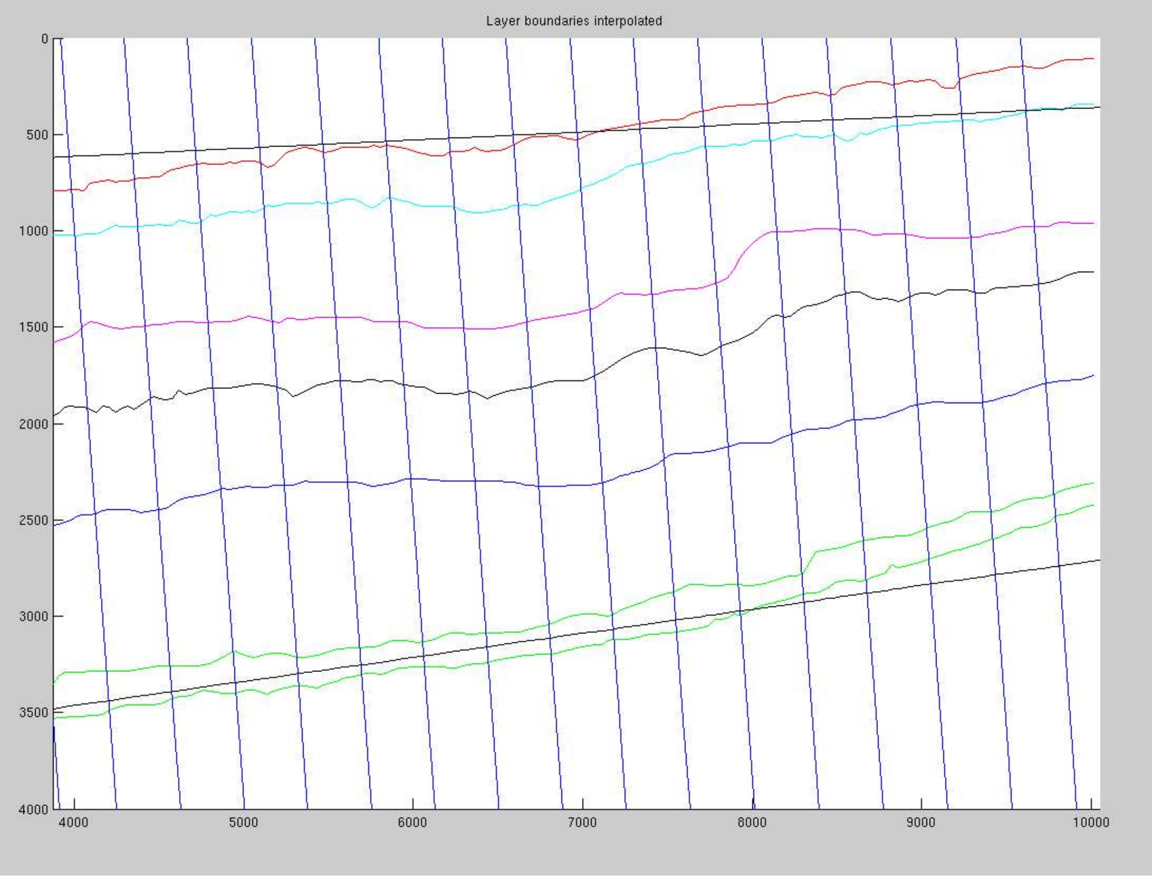


A second trial was to do the same process, except that the initial points are taken on the upper boundary of layer 5 instead of the bottom.



A set of other methods such as choosing points on the other layers as being the closest to the initial one only (and not to the point on the previous layer) were tried, but didn't bring any satisfactory results.

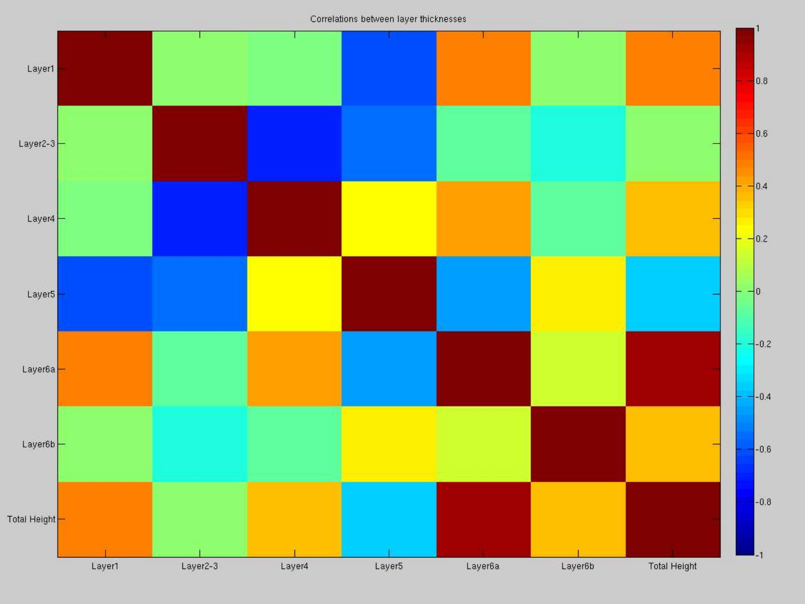
The last method used was chosen in a way to obtain lines that are not affected with the small variations ("bumps"...), ie lines that are radially oriented, in a parallel manner. A fitting of the bottom and upper boundary of layer 1 was conducted, and the "best" orthogonal line to both those lines was found, relying on the angles of the fitted lines (via the orthogonalLine function : computed angles for each one of the fitted lines, found a mean angle, found a "common" slope equal to the tan of the mean angle, found the slope of orthogonal lines).



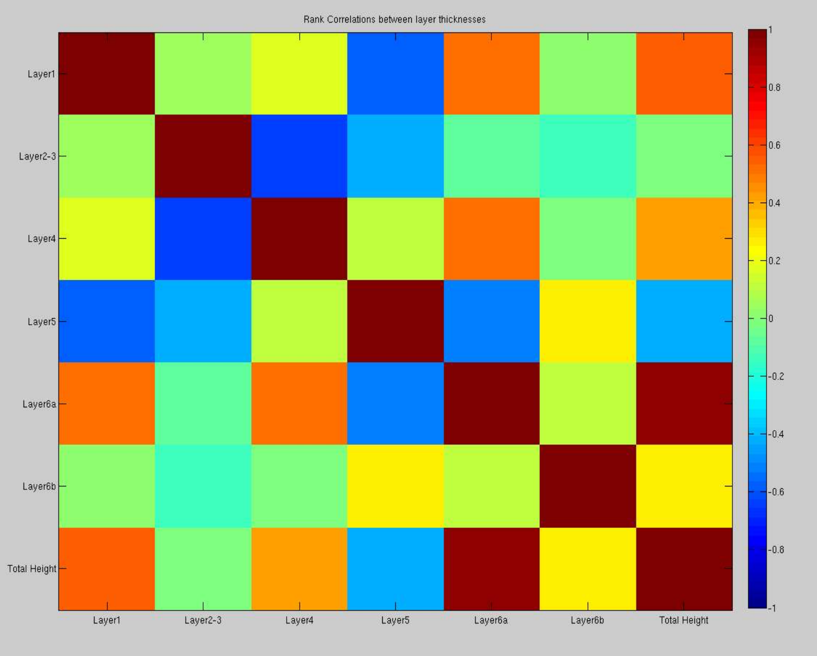
Then next step was to find the intersection of those orthogonal lines with the boundaries, in order to obtain the thicknesses.
Because the equation of the boundaries are unknown, I did a linear interpolation for each boundary with a very small step size, took each small segment alone (used lineEQ

function),found the intersection with the lines. If the intersection belongs to the segment, I saved it, else I ignored it. (via the getIntersection function)

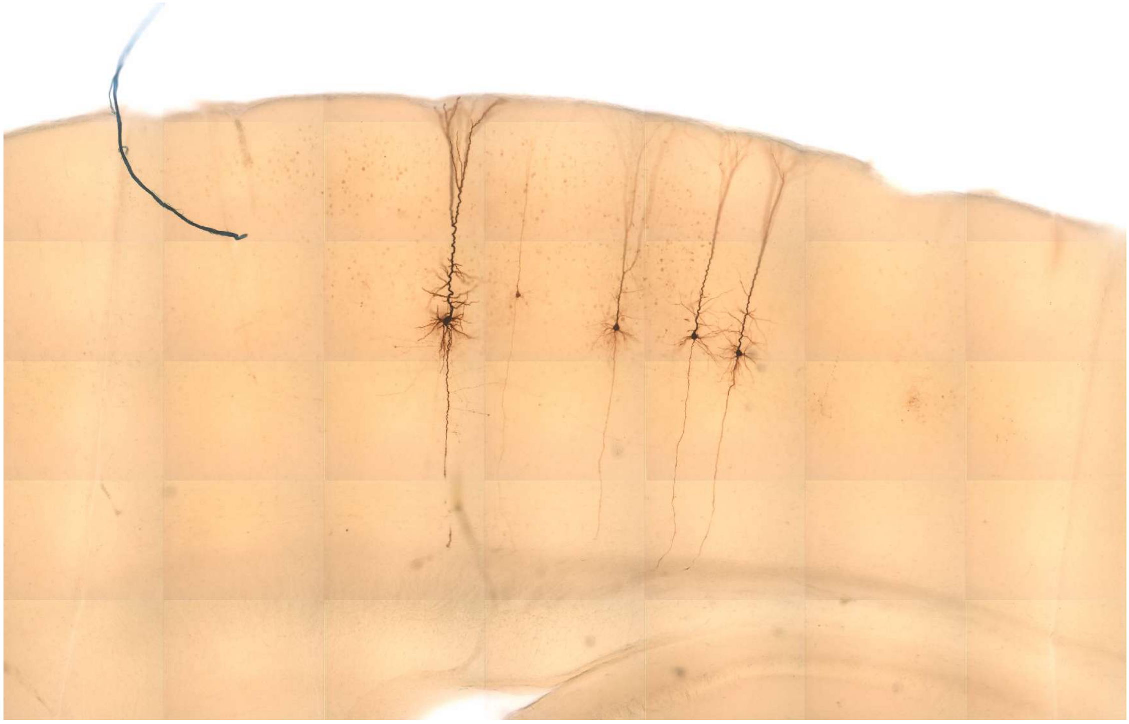
After having the points of intersection, I computed the thicknesses by computing the distance between points on consecutive layers (via the getThickness function) I calculated the total height each time, and obtained the following correlation plot (via the getCorrelation function):



The Spearman Correlation:



However, I was waiting for an image showing the orientation of the apicals from Sonia, to have a more precise idea, because what was done above is incomplete and might be an approximation for very small portions only.
The following picture was the last staining obtained from Rodrigo and Sonia. However it is still incomplete, due to two main facts: layer boundaries are not represented and the staining is not for all the desired portion.



The next step would then be to obtain those biological images.

Code guidelines:

Main functions:

getPlacementHintDnewrep:

inputs: - filePath corresponds to the morphoParameters file in the morphology folder

- path corresponds to the neuronDB file in the placementAlgorithm folder

Pseudocode:

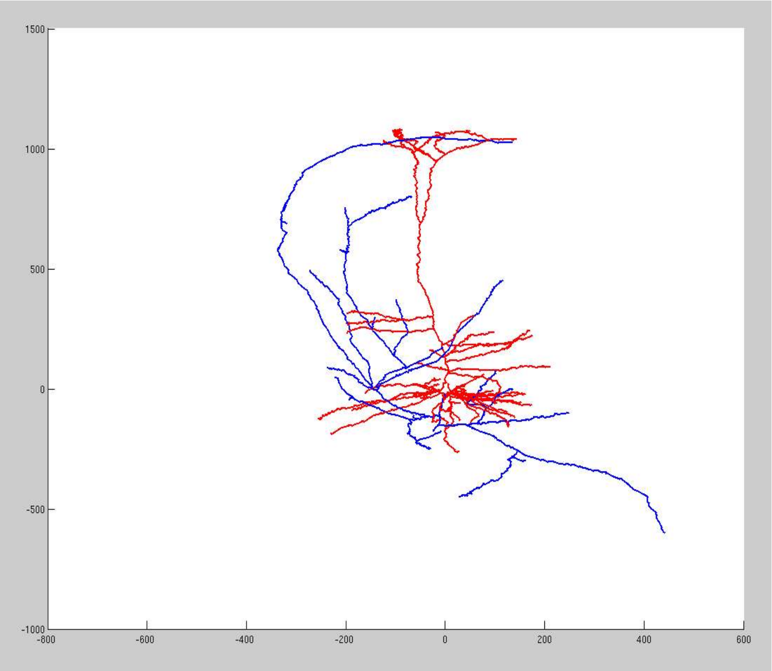
```
read h5 files
define pia = max dendrite height of all Layer 5 cells + lower boundary of Layer 5 +one
bin height.
LOOP1: for each layer
    calculate the bin height
    calculate the dendrites' maximum reach for the neurons in that specific layer
    LOOP2: for each type present in the layer
        getMinBin: [ cond1: if current type is L2PC or L3PC or L4PC or
                                L5CSPC or L5CHPC
                            define lower constraint as being the lower boundary of
                            Layer1
                            else if current type is L4SP
                           define lower constraint as being lower boundary of
                            Layer3
                           else lower constraint is 0.
                     end cond1
                    loop1: for each bin in the layer (starting with highest bin)
                        find the neurons which do not satisfy their lower
                            constraint when placed at that bin
                        cond2: if its the tenth bin, assign to them a minBin of 10
                            else assign a minBin of current bin+1
                        end cond2
                    end loop1
                    for the remaining neurons which always satisfy their lower
                    constraint, assign a minBin of 1 ]
        getMaxBin: [ cond1: if current type is L4SP or L6CCPC or L6CTPC or
                                L6CSPC
                                define upper constraint as upper boundary of Layer2
                        else define upper boundary as the pia
                    end cond1
                    loop1: for each bin (starting with the first one)
                        find the neurons which do not satisfy their upper
                          constraint when placed at that bin
                      cond2: if its the first bin, assign to them a maxBin of 1
                          else assign a maxBin of current bin-1
                      end cond2
                  end loop1
                  for the remaining neurons which always satisfy their upper
                  constraint, assign a maxBin of 10 ]
    end LOOP2
end LOOP1

LOOP3: for each layer
     find neurons in this layer
     binPlacement: [ declare ten counters for each bin and initialize them to 0
                  loop1: for each bin (starting with the smallest)
                      loop2: for each neuron
                          calculate the allowed range as the difference
                          between the maxBin and minBin + 1 for this
                          neuron
                          cond1: if the range allowed is equal to current bin
                              (insure neurons with smallest range are
                              placed first)
                              loop3: for each counter between minBin
                                  and maxBin
                                  find the minimum of those counters
                                  increment minimum counter by 1
                                  assign soma position of this neuron
                                  to the bin with the corresponding
                                  counter.
                              end loop3
                          end cond1
                      end loop2
                  end loop1 ]
    save in placement index all soma positions returned divided by 10.
end LOOP3


aboveConstraintNeuronsD: [ loop1: for each type
                               define upper constraint as above (depending on
                               current type)
                               loop2: for each neuron
                                   dendrite reach = lower boundary of layer +
                                   maxBin * binHeight + max dendrite height
                                   excess= dendrite reach – upper constraint
                                   cond1: if excess is positive
                                       print it in a file
                                   end cond1
                               end loop2
                           end loop1 ]
belowConstraintNeuronsDnew: [ loop1: for each type
                               define lower constraint as above (depending on
                               current type)
                               loop2: for each neuron
                                   dendrite reach = lower boundary of layer +
                                   minBin * binHeight + max dendrite height
                                   difference= dendrite reach – lower constraint
                                   cond1: if difference is negative
                                       print it in a file
                                   end cond1
                               end loop2
                           end loop1 ]
LOOP4: used for plotting
end LOOP4

LOOP5: generate the file to be given for building the circuit.
end LOOP5
```

Unresolved issues:

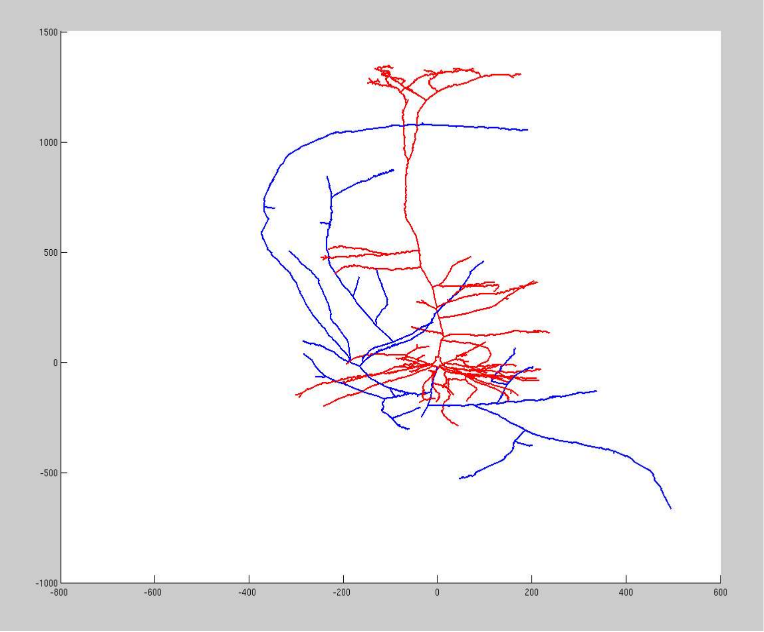
- shrinkage of layers not taken into consideration
- The repair process affects the placement a lot. When the problems with the repair are resolved, a new morphology file should be used as an input to the placement hints function. Then, we will be able to observe the progression from unrepaired cells, to unraveled ones, to repaired ones. (Currently, this comparison is not possible, due to the fact that we are using the new unraveling done by Karthik for the unraveled database, but the old one for the repaired database).

Here are some figures obtained that confirm the fact that we are using two different unraveling processes.

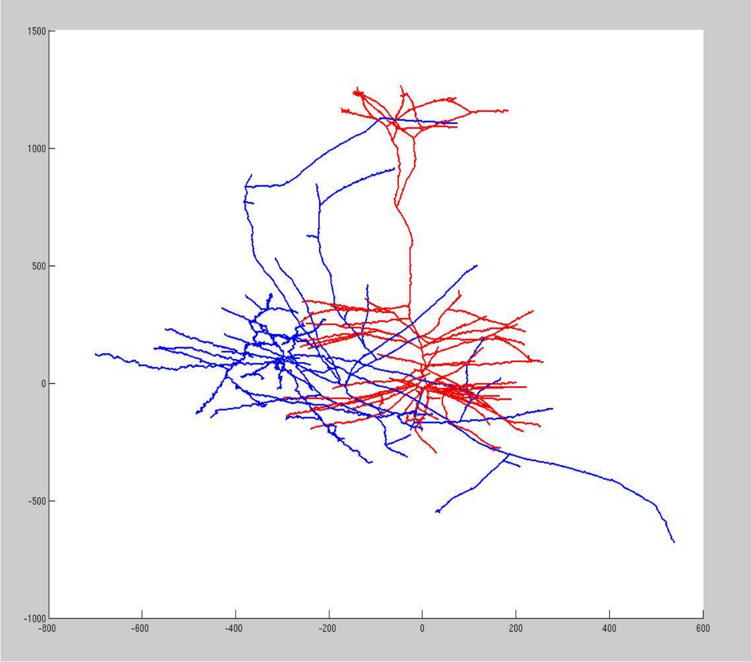
Unrepaired and Raveled:



Unraveled:



Repaired:



- Some clones are not very representative and might be misleading (such as Layer 5 clones with huge dendrites).
- It is still not clear how we should include in the code the orientation of the apicals, need more biological data.

Thank you!