Appendix 1 – MATLAB code for analysing *B. subtilis* septa pores

% read the image into MATLAB and convert it to grayscale

RGB=imread(‘image\_name.jpg'); %enter name of AFM image within ' '

imshow(RGB);

Grey = rgb2gray(RGB);

imshow(Grey); %show AFM image on Matlab windows

%detect circle (drawn by hand in ImageJ to be a perfect circle)

Rmin=100;

Rmax=800;

[centre,radius]=imfindcircles(Grey,[Rmin,Rmax],'Sensitivity',0.99);

%show circle and centre

viscircles(centre,radius);

plot(centre(:,1),centre(:,2),'yx','LineWidth',1);

%convert values to nm

nm=1.9230769;%This conversion number is nm of image/pixels

centreNM = nm\*(centre);

radiusNM = nm\*radius;

%Read cartesian coordinates of all holes exported from ImageJ

XY = readtable('Name\_file.txt'); %enter name of data file within ' '

%Separate them in two columns (X and Y)

X=XY(:,1);

Y=XY(:,2);

%Divide the centre vector into two

x\_zero = centreNM(:,1);

y\_zero = centreNM(:,2);

%move the centre from the top left corner to the centre of the septum

hold on;

Size = size(X);

L=Size(:,1);

X\_polar = X;

Y\_polar = Y;

for i = 1:L %Change the coordinates(X)values respect x\_zero as centre

if (X.(1)(i) < x\_zero)

X\_polar.(1)(i)= (x\_zero-X.(1)(i));

end

if (X.(1)(i) > x\_zero)

X\_polar.(1)(i)= (X.(1)(i)-x\_zero);

end

i=i+1; %make for advance to the next iteration

end

for i = 1:L %Change the coordinates(Y)values respect y\_zero as centre

if (Y.(1)(i) < y\_zero)

Y\_polar.(1)(i)= (y\_zero-Y.(1)(i));

end

if (Y.(1)(i) > y\_zero)

Y\_polar.(1)(i)= (Y.(1)(i)-y\_zero);

end

i=i+1; %Make for advance to the next iteration

end

%Calculate each radius for all the X\_polar Y\_polar values

pore\_radius = X; %create a the variable equal to X just because it has the right amount of rows

for i = 1:L

pore\_radius.(1)(i)=sqrt((X\_polar.(1)(i))\*(X\_polar.(1)(i))+(Y\_polar.(1)(i))\*(Y\_polar.(1)(i)));

i=i+1;

end

%Calculate the distance from each pore to the leading edge

Distance\_edge = X;

for i = 1:L

Distance\_edge.(1)(i) = pore\_radius.(1)(i)-radiusNM;

i=i+1;

end

% create results table with the distance to the leading edge and the distance to the centre (pore radius)

writetable(Distance\_edge);

writetable(pore\_radius);

Appendix 2– FIJI code for analysing mature mesh pores

**Code 1:**

run("Close All"); //close all images

//define directory location of your file

dir1 = getDirectory("Choose Source Directory");

dir2 = getDirectory("Choose Destination Directory");

//write the exact name of your FILE WITH EXTENSION

name1= getString("File name.extension", "SAC\_.tif");

//define in how many slides you want to cut the image

number\_slides=getNumber("number\_of\_slides", 255);

//This is to avoid seeing all the actions the system does

setBatchMode(true);

/\*Use the loop to repeat the same process X times, The maximum number is 255 which is the number of times we can divide

this image in binary using the threshold.

\*/

for(i=0;i<number\_slides;i++){

//Open image already transformed into grey-scale

open(dir1+name1);

//threshold image

setAutoThreshold("Otsu");

//Here we are selecting the threshold area and define variables

a = 0;

b = number\_slides-i;

setThreshold(a, b);

//apply the threshold, select false to put holes in black and ture to put them in white

setOption("BlackBackground", true);

run("Convert to Mask");

//clean up the binary image

run("Fill Holes");

run("Open");

//convert to stack

run("Images to Stack", "name=Stack2 title=[] use");

//write the name of the stack and the stack sequence WITHOUT EXTENSION

name2 = getString("File name", "SAC\_3\_");

//Save stack as .tiff and .gif

saveAs("tiff", dir2+name2);

saveAs("gif", dir2+name2);

//Save the stack as Image Sequence for the analysis with **Code 2**

dir3 = getDirectory("Choose a Directory");

run("Image Sequence... ", "format=TIFF name="+name2+" digits=3 save=["+dir3+"]")

**Code 2:**

//Put the source folder where the binary stacks are

//Choose the destination folder where to put the results

dir1 = getDirectory("Choose Source Directory");

dir2 = getDirectory("Choose Destination Directory");

list = getFileList(dir1);

//Ask user number for the nm to pixel conversion of the image and the number of slices

conversion=getNumber("conversion nm/pixel=", 0.769230769);

number\_of\_repetitions = getNumber("Number of slice to analyse = ", 255);

//This is to avoid seeing all the actions the system does

setBatchMode(**true**);

//This is to create the array that will hold the final result and the file where it's going to be stored

HCFA = newArray(number\_of\_repetitions);

headline = "Depth\_slice\_number, HCFA"; //File headers

File.append(headline, dir2+"Final"+".csv"); //Create blank file

//Create vectors to count the pores by different size and create file to save the data

Yellow = newArray(number\_of\_repetitions);

Green = newArray(number\_of\_repetitions);

Magenta = newArray(number\_of\_repetitions);

Blue = newArray(number\_of\_repetitions);

Total\_count = newArray(number\_of\_repetitions);//Count all pores/slice

headline2 = "Depth\_slice\_number, Yellow\_pores, Green\_pores, Magenta\_pores , Blue\_pores, Total\_number\_pores"; //File headers

File.append(headline2, dir2+"ColourCount"+".csv"); //Create blank file

//This for is to repeat the same process for the number of slices chosen by the user

**for**(j=0;j<=number\_of\_repetitions;j++){

//Clean up the roi manager

**if** (roiManager("count")>0) {

roiManager("Deselect");

roiManager("Delete");

}

run("Close All"); //close all images

run("Clear Results"); //Clear results table

//Open folder with the binary stack of images from the AFM image

open(dir1+list[j]);

//Convert image pixel to calculate it diving image size (nm) / pixels

run("Properties...", "channels=1 slices=1 frames=1 unit=nm pixel\_width="+conversion+" pixel\_height="+conversion+" voxel\_depth="+conversion+"");

//threshold image (to convert it to the right format for analysis

setAutoThreshold("Otsu dark");

//apply the threshold, select false to put holes in black and ture to put them in white

setOption("BlackBackground", **true**);

run("Convert to Mask");

//Obtain the table using RoiManager, Measure

run("Analyze Particles...", "size=2-Infinity display include add");

resetThreshold();

//Visualize nothing to be able to highlight later certain areas

roiManager("Show None");

//if there are no rois, do not try to calculate anything

**if** (roiManager("count")>0) {

number\_of\_rois = roiManager("count");

// variables to count pores in each slice of different colours

yellow\_rois = 0;

green\_rois = 0;

magenta\_rois = 0;

blue\_rois = 0;

//Classify rois by area and count how many pores correspond to colours

**for** (i = 0; i < number\_of\_rois; i++) {

roiManager("select", i); //select a ROI

Area = getResult("Area", i);

//colour code pores depends on their area

**if**(Area <= 20){

Roi.setStrokeColor("yellow");

Roi.setStrokeWidth(5);

yellow\_rois = yellow\_rois + 1;

}

**if**(Area > 20 && Area <= 500){

Roi.setStrokeColor("green");

Roi.setStrokeWidth(5);

green\_rois = green\_rois + 1;

}

**if**(Area > 500 && Area <= 1500){

Roi.setStrokeColor("magenta");

Roi.setStrokeWidth(5);

magenta\_rois = magenta\_rois + 1;

}

**if**(Area > 1500) {

Roi.setStrokeColor("blue");

Roi.setStrokeWidth(5);

blue\_rois = blue\_rois + 1;

}

Overlay.addSelection(); //Add it to the overlay

} //close for from line 82

saveAs("jpeg", dir2+list[j]); //Save processed image

//Allocate the count of pores to the colour vectors created at the beggining and the total count, will be different for each slice

Yellow[j]=yellow\_rois;

Green[j]=green\_rois;

Magenta[j]=magenta\_rois;

Blue[j]=blue\_rois;

Total\_count[j]=number\_of\_rois;

//Save all the count values from all the pores in ("ColourCount.csv") that we created at the beginning

contentline2= ""+ j +","+Yellow[j]+","+Green[j]+","+ Magenta[j]+","+ Blue[j]+","+ Total\_count[j];

File.append(contentline2, dir2+"ColourCount"+".csv");

//Create an array and allocate all values of Area

to\_sort = newArray(number\_of\_rois);

Total=0;

**for** (i = 0; i < number\_of\_rois; i++) {

roiManager("select", i); //select a ROI

//Allocate values to new arraw

Area = getResult("Area", i);

to\_sort[i]=Area;

//Calculate total area (sum of all numbers)

Total = Total + Area;

}

//Sort the values of area from smaller to larger

Sorted = Array.sort(to\_sort);

//Define arrays needed to calculate cumulative fraction of total area per each slice

cum\_value = newArray(number\_of\_rois+1);

cum\_fraction = newArray(number\_of\_rois+1);

cum\_value[0]=Sorted[0];

cum\_fraction[0]=cum\_value[0]/Total;

**for** (i = 1; i < number\_of\_rois; i++) {

//Calculate cumulative value

cum\_value[i]=cum\_value[i-1]+Sorted[i];

//Calculate cumulative fraction of total area

cum\_fraction[i]=cum\_value[i]/Total;

//Calculate half of the cumulative fraction of total area (HCFA)

**if** (cum\_fraction[i]>0.40 && cum\_fraction[i]<0.60) {

HCFA\_in = i;

HCFA[j] = Sorted[HCFA\_in];

}

} //Close for from line 151

//Create final results table with the raw Area, the area sorted and the cumulative fraction for each individual slice

table1="Results\_table";

Table.create(table1);

**for** (i = 0; i < number\_of\_rois; i++) {

Area = getResult("Area", i);

Table.set("Raw Area",i, Area);

Table.set("Area sorted",i, Sorted[i]);

Table.set("Cumulative fraction",i, cum\_fraction[i]);

}

saveAs("Results",dir2+File.nameWithoutExtension+".csv");

//Save all the HCFA values from all the repetitions in one single file ("Final.csv") that we created at the beginning

**if** (HCFA[j]>0){

contentline= "" + j + "," + HCFA[j];

File.append(contentline, dir2+"Final"+".csv");

}

//Close random table that keeps appearing at each iteration

name = File.nameWithoutExtension+".csv";

x=isOpen(name);

**if**(x==**true**){

selectWindow(name);

run("Close");

}

}//Close the main for from **line 37**

//Close Results table at the end

name2 = "Results";

y=isOpen(name2);

**if**(y==**true**){

selectWindow(name2);

run("Close")