**On Mac**

brew install cmake

brew install git-lfs

brew install itk

brew install cask

(brew install dgtal)

**General**

mkdir SGext

cd SGext

git clone https://github.com/phcerdan/SGEXT SGEXT-master

cd SGEXT-master

git pull

git fetch

cd ..

mkdir build\_dependencies

cd build\_dependencies

cmake ../SGEXT-master/dependencies -DCMAKE\_BUILD\_TYPE=Release

cmake --build .

cd ..

mkdir build (build should be parallel to build\_dependencies 🡪 SGext folder should have folders SGEXT-master, build\_dependencies, build)

cd build

cmake ../SGEXT-master -DCMAKE\_BUILD\_TYPE=Release -DDEPENDENCIES\_BUILD\_DIR=../build\_dependencies

make -j12

cd ../cpp-scripts

% END INSTALLATION

./create\_distance\_map --input=../Geometries/m1p2\_053107.nrrd --outputFolder=../Geometries/

./thin --input=../Geometries/m1p2\_053107.nrrd --select=dmax --skel=1isthmus --inputDistanceMapImageFilename=../Geometries/m1p2\_053107.nrrd --foreground=white --exportImage=../Geometries/

cd ../Geometries

create / m1p1\_053007\_SEG \_skel\_adj.nrrd  by copying the file / m1p1\_053007\_SEG \_skel.nrrd

Open/ m1p1\_053007\_SEG\_skel\_adj.nrrd. Change the space origin to (0,0,0) and save the adjusted skeleton file

cd ../cpp-scripts

./analyze\_graph --input ../Geometries/m1p2\_053107\_SKEL\_adj\_dmax\_isthmus1\_p0.nrrd -o ../Geometries/  --removeExtraEdges --mergeThreeConnectedNodes --mergeFourConnectedNodes --mergeTwoThreeConnectedNodes --exportGraphviz --verbose -d ../Geometries

mv ../Geometries/Segmentation5\_DMAP.nrrd ../Geometries/Seg\_DMAP.nrrd

mv ../Geometries/Segmentation5\_SKEL\_adj\_dmax\_isthmus1\_p0\_REDUCED\_sp\_c\_m\_data.txt ../Geometries/Seg\_adj\_SKEL.txt

mv ../Geometries/Segmentation5\_SKEL\_adj\_dmax\_isthmus1\_p0\_REDUCED\_sp\_c\_m.dot ../Geometries/Seg\_adj\_SKEL.dot

rm ../Geometries/Segmentation106\_SKEL\_adj\_dmax\_isthmus1\_p0.nrrd

**In Geometries/MATLAB**

*[arcs,nodes]=formatData('../segmentationControl\_SKEL\_adj\_dmax\_isthmus1\_p0\_REDUCED\_sp\_c\_m\_data.txt','../segmentationControl\_SKEL\_adj\_dmax\_isthmus1\_p0\_REDUCED\_sp\_c\_m.dot','../segmentationControl\_DMAP.nrrd');*

*[arcs,nodes]=formatData('../m1p4\_060407\_adj\_SKEL\_dmax\_isthmus1\_p0\_REDUCED\_sp\_c\_m\_data.txt','../m1p4\_060407\_adj\_SKEL\_dmax\_isthmus1\_p0\_REDUCED\_sp\_c\_m.dot','../m1p4\_060407\_DMAP.nrrd');*

*[arcs,nodes]=formatData('../segmentation5\_SKEL\_adj\_dmax\_isthmus1\_p0\_REDUCED\_sp\_c\_m\_data.txt','../segmentation5\_SKEL\_adj\_dmax\_isthmus1\_p0\_REDUCED\_sp\_c\_m.dot','../segmentation5\_DMAP.nrrd');*

**Nodes** is a matrix with node number, x,y,z coordinate and the node degree.

Nodes of degree 1 are terminal

Nodes of degree 3 are branch-points a node of degree 3 has

one vessel coming in and 2 vessels going out.

**Arcs** is a cell array that contains the points along each vessel. For each vessel the first row gives the from and to node IDs. Rows 2 to the end give x,y,z coordinates and the radius at that point.

*plotSlicerData(arcs,nodes,'b');*

Shows the graph with labeled nodes – don’t worry about the graph other nodes – be careful if root node is flipped!

Identify the root node number from plotSlicerData. In the example it is 6

Correct graph (provides the ability correct arcs and nodes manually)

The best way to see nodes and edges to be corrected is in slicer see commands below.

Save corrected arcs and nodes (Example – root node has ID 6)

*[path, arcsC, nodesC, correction\_log]=correctionEngine(arcs,nodes,6)*

*[path, arcsC, nodesC, correction\_log]=correctionEngine(arcs,nodes,126)*

This code removes erroneous vessels and generates the file path determining the path from the root to each terminal vessel.



**path:** A list of nodes transversed to get to any node in the network starting at the root node. From these paths we can see if the network has not been corrected appropriately.

*[orientation,newNetwork,connectivity,arcsC2,maxDaughters,maxParrents] = directedGraphInfo(arcsC,nodesC, path);*

**Orientation:** This code assigns a -1 to all vessels that are oriented the wrong way.



**newNetwork:** is a matrix with 6 columns stating 1: Vessel ID, 2: ID for starting node, 3: ID for end node, 4: IQM of the vessel radius, 5: Vessel length, 6: std of the vessel radius.

**connectivity:** first column start node, 2nd column in-degree – how many daughters, 3rd column out-degree – how many parents, columns 4-7 ID for daughter vessels, columns 8-9 ID for parent vessel.

**maxDaughters:** Max number of daughters

**maxParents:** Max number of parents

In newNetwork identify the root vessel as the one node with no parent. In this example node (6).

*ID = find(connectivity(:, 3)==0);*

*rootNodeIndex = connectivity(ID,1)*

*[vessel\_details] = extract\_geometry(newNetwork,connectivity,arcsC2,orientation);*

*This code returns vessel\_details for each vessel the nodes is oriented from the beginning to the end of the vessel. The code should add a column to arcsC (the second argument) that calculates the length between two nodes, this will give us the length of the vessel.*

*[T] = Export\_to\_Excel(vessel\_details,0,'Aorta')*

*This code generates an excel file, that is read in by R. Run the R-code LocateChangepoints.Rmd to identify the change-points.*

[details] = Import\_from\_excel('IMPORT.xlsx',vessel\_details);

*This code updates vessel\_details adding the changepoints*

changepoint\_location=get\_changepoint\_locations(details);

This code identifies the radius and x-value associated with each changepoint

vessel\_radii = get\_radii\_exponential(details, changepoint\_location, TaperID, Scale)

TaperID = “” or “1,2” (vessel IDs for tapered vessels)

Scale is the scaling factor from voxels to mm.

Plots all points along each vessel along with change point and estimated parameters. This code should construct the file that is needed by the c-code. Note – this code only finds radii in straight vessels.

Next we need to construct the input to the c-code

data = CreateFluidsCodeInput(vessel\_radii,details,maxDaughters)

A paper with a diagram and writing on it

Description automatically generated

