MVN-ANALYSIS: SOFTWARE TO CHARACTERIZE THE STRUCTURE AND FLUID FLOW FORCES WITHIN ENGINEERED MICROVASCULAR NETWORKS

by

Ryan Armstrong

Senior Honors Thesis

Department of Physics

University of North Carolina at Chapel Hill

April 2020

This thesis has been reviewed by the research advisors and the departmental coordinator for the honors program and has been found to be satisfactory.

Dr. Boyce Griffith, Advisor

Dr. William Polacheck, Co-Advisor

Dr. Lu-Chang Qin, Departmental Coordinator for the Honors Program

ACKNOWLEDGMENT

This thesis would not have been possible without the generosity of many wonderful people. I would like to thank my advisors, Dr. Boyce Griffith and Dr. Bill Polacheck, for providing me with the opportunity to learn through this project and their guidance along the way. Further, I owe a massive debt of gratitude to Dr. Simone Rossi, who not only provided the fluid solver used in this report, but also replied to a countless number of confused emails with unwavering patience and support.

Additionally, Crescentia Cho produced and imaged the microvascular networks used in the analysis with assistance from Joanna McDonald–without which, there would be no project. Similarly, Margaret Anne Smith donated her time to teach me the Trelis interface, which greatly increased my productivity.

Beyond the technical support, I have always been encouraged by my parents, Laura and Doug; my siblings, Cate, Holden and Harry; and my best friend, Andrea Brucculeri. I would also like to thank (or maybe blame) Samantha Pagan, Patrick Taylor, and Maggie Hildebrand for inspiring me to pursue a thesis. I miss the 2 a.m. walks from Phillips to Cosmic.

Abstract

Microvascular networks consist of interconnected blood vessels $\lesssim 150$ μm in diameter and are responsible for transporting metabolites and nutrients to the body's cells. New microvascular blood vessels are formed from existing vessels through the process of angiogenesis. While many chemical gradients controlling angiogenesis are known, the role of hemodynamic forces–such as wall shear stress and pressure–are less understood. Controlling angiogenesis is clinically relevant and could be uses in the treatment of cancer, ischemic heart disease, peripheral arterial disease, and others. However, the small physical scale and large topological complexity of microvascular networks makes researching the effects of fluid flow forces on angiogenesis difficult. Therefore, we created software capable of characterizing the structure and fluid flow forces of engineered microvascular networks. Specifically, we demonstrate feasibility to obtain computational volume meshes of the network lumens which can be used in existing fluid solvers to obtain velocity, pressure, and wall shear stress distributions for a microvasular network. Further, the developed software generates physiologically relevant graphical (nodes and edges) and numerical outputs. These outputs describe a network's segment lengths, radii, surface areas, volumes, contraction factors, fractal dimensions, connectivity, anisotropy, vessel density, and distance of non-vascularized areas to a vessel wall. Using these outputs, we compared microvasular networks grown in low-nutrient and static storage conditions to networks grown in standard nutrient and dynamic storage conditions. Within the low-nutrient and static networks, we observed a decrease in vessel density but a potential maintenance of support to nearby tissue. Further, the analysis yielded a linear relationship between segment radii and nutrient concentration that could be used to manufacture micrvascular networks at desired average segment radii. Ultimately, the software is designed to relate fluid flow forces to angiogenesis and inform engineering decisions when creating microvascular networks.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter One

Introduction

The cardiovascular system serves as the body's scale-linking mechanism. It connects the respiratory, digestive, endocrine, and excretory systems to individual cells by transporting gases (oxygen and carbon dioxide), nutrients, enzymes, hormones, and heat [1]. The actual exchange of these molecules and energy primarily occurs at the microvascular level where diameters of blood vessels are $\lesssim$ 150 μm [2]. Further, these exchanged metabolites and nutrients only diffuse a few hundred microns within tissue. Therefore, viable, metabolically active cells must have nearby access to microvascular blood vessels [3].

Creation of new microvascular vessels occurs through the process of angiogenesis. During this process, already formed vessels either sprout (sprouting angiogenesis) or split (intussusceptive angiogenesis) to create new vessels. Many biochemical factors and co-factors are known to stimulate or inhibit the process, especially for sprouting angiogenesis. For example, the fibroblast growth factor (FGF) and vascular endothelial growth factor (VEGF) proteins promote angiogenesis [4]. Therefore, hypoxic cells–which up-regulate FGF receptors–stimulate new vessel growth, while molecules that bind to VEGF–such as the drug Bevacizumab–inhibit angeiogenesis [5, 6]. However, for both sprouting and intussusceptive angiogenesis, hemodynamic forces, such as wall shear stress (WSS) and pressure, also influence levels of vessel formation. Though, these factors are less understood than their chemical counterparts. Most studies report increases in angiogenesis with increases in shear stress [7–9], however a

minority of studies report the opposite and claim decreases in angiogenesis with increases in shear stress [10].

Understanding the factors, including the hemodynamic factors, that control angiogenesis is clinically relevant. Stimulation of angiogenesis could be therapeutic to ischemic heart disease, peripheral arterial disease and wound healing. Alternatively, inhibiting angiogenesis could limit cancer metastasis, ophthalmic conditions, and rheumatoid arthritis [11].

Motivated by these applications, the Polacheck lab developed a novel microfluidic platform capable of supporting the growth of perfusable, three-dimensional microvascular networks (MVNs) of human umbilical vein endothelial cells (HUVECs).[1] Compared to *in vivo* platforms, the Polacheck networks offer better control over the known chemical gradients that drive angiogenesis, and therefore can better isolate mechanical causes. However, the network vessels are too small to directly measure the fluid forces and too topologically complex to manually quantify structurally. Therefore, with the ultimate goal of investigating the relationship between fluid flow forces and angiogenesis, we developed a software suite (`mvn-analysis`) capable of characterizing the structure and fluid flow forces within the Polacheck microvascular networks.

1.1 Polacheck Mircofluidic Devices

The work presented in this thesis is focused on the development and use of computational tools and not on the biology or development of the microvascular networks. However, a general understanding of the Polacheck microfluidic devices helps motivate the need for these computational tools.

Broadly, the Polacheck microfluidc devices are PDMS-on-glass chambers containing physiologic extracellular matrix (ESM) seeded with HUVECs and human lung fibroblasts. During

[1]The methods paper for the Polacheck microvascular network platforms used in this study is not yet published. Though, see Polacheck *et al.* (2019) for similar [12].

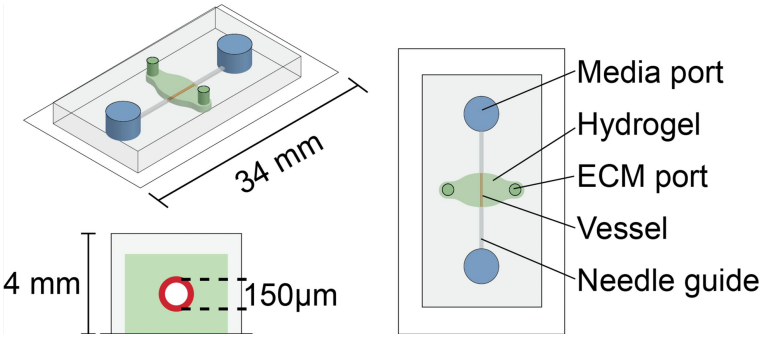


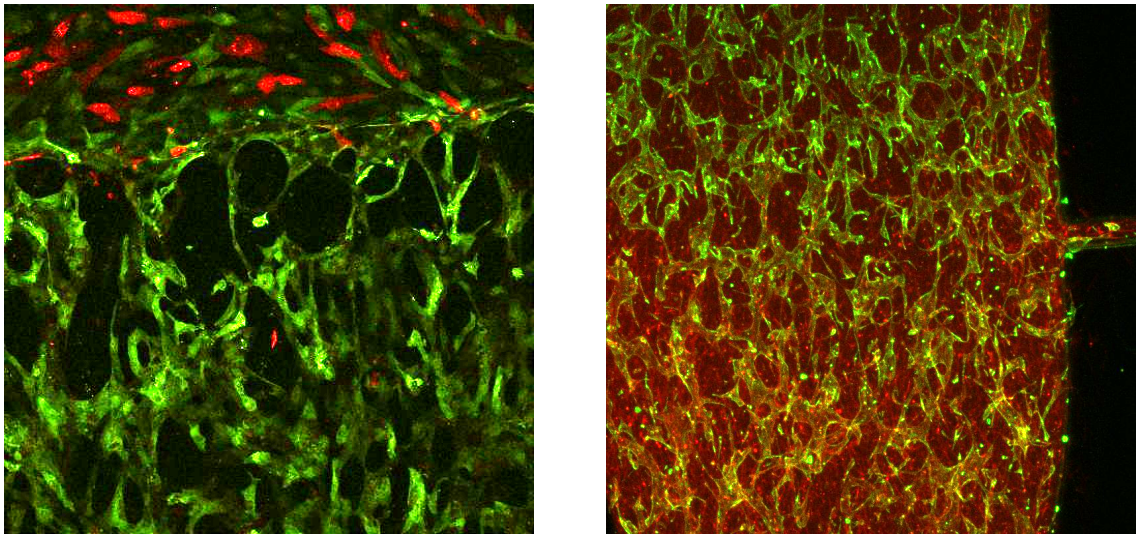
Figure 1.1 A microfluidic device similar to the once used in this study.

the construction of the devices, sterile needles are used to negatively cast two parallel channels in the ESM. These channels, referred to as parent channels, are seeded with HUVECs and irrigated with cell culture media by a rocker. Figure 1.1 shows a single channel variant of a similar device from the work published in Polacheck *et al.* (2019) [12].

Within successful devices, the ECM HUVECs create a network with open lumens. These networks undergo self directed growth and are expected to connect to the parent channels via anastomosis. Figure 1.2 shows examples of microvascular networks created in these types of devices.

The diameter, separation, and distance from the glass of the parent channels are all device variables currently being explored. However, the diameters are on the order of $\sim 150 - 300$ μm, the separations are on the order of $10^1 - 10^3$ mm and the distance from the glass is in order of 10^2 μm. In addition, the composition of the ECM and the irrigation conditions (rocker settings) are also being investigated.

As can be seen from Figure 1.2, the MVNs are structurally complex. Therefore, changes in the device parameters are difficult to relate to changes in the networks. While the ultimate goal of this research is to relate fluid flow forces to angiogenesis, any computational methods able characterize the MVNs would be beneficial to standardizing their manufacturing parameters. This application is explored in Chapter 4, where `mvn-analysis` is used to compare the networks generated by devices varying in ECM composition and level of irrigation.



(a) 10× magnification. **(b)** 4× magnification.

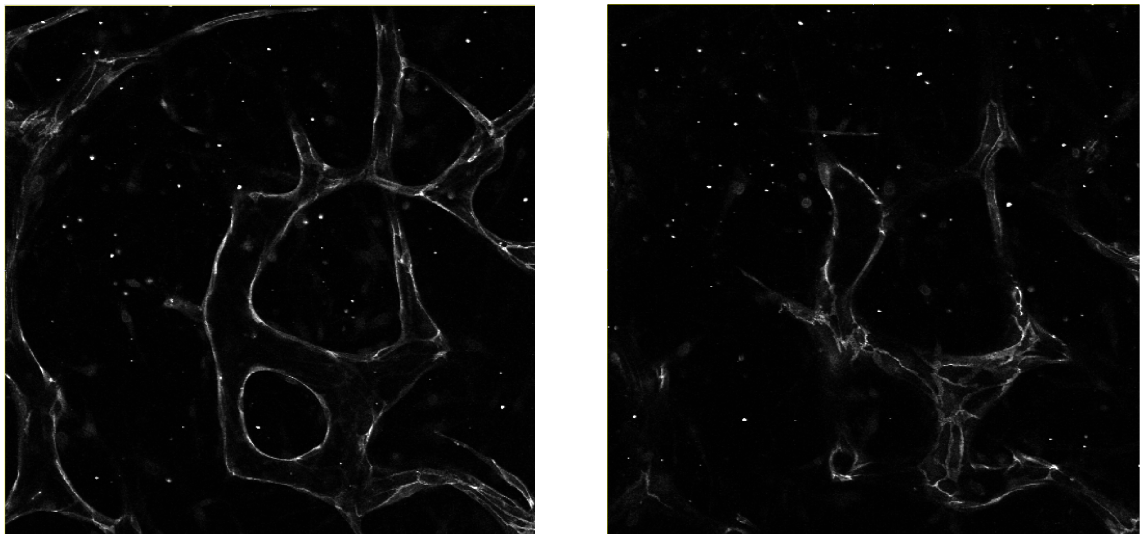
Figure 1.2 MVNs created by the Polacheck lab in devices similar to the ones described. Horizontal parent channels are shown at the top of (a) and through the center of (b). HUVECs are labeled green.

1.2 Image Segmentation

Within the context of this research, image segmentation relates to distinguishing between pixels that represent vessel lumens and pixels of non-lumenized space.

1.2.1 MVN Imaging

The HUVEC used in the Polacheck devices were stably transduced using lentiviral transduction particles to constitutively express GFP (green) and Ruby (red). The cells of of the microvascular network express GFP and therefore are shown as green in Figure 1.2. Imaging of the MVNs was conducted using an Olympus FV3000 confocal microscope to capture discrete parallel planes along the z-axis.



(a) **(b)**

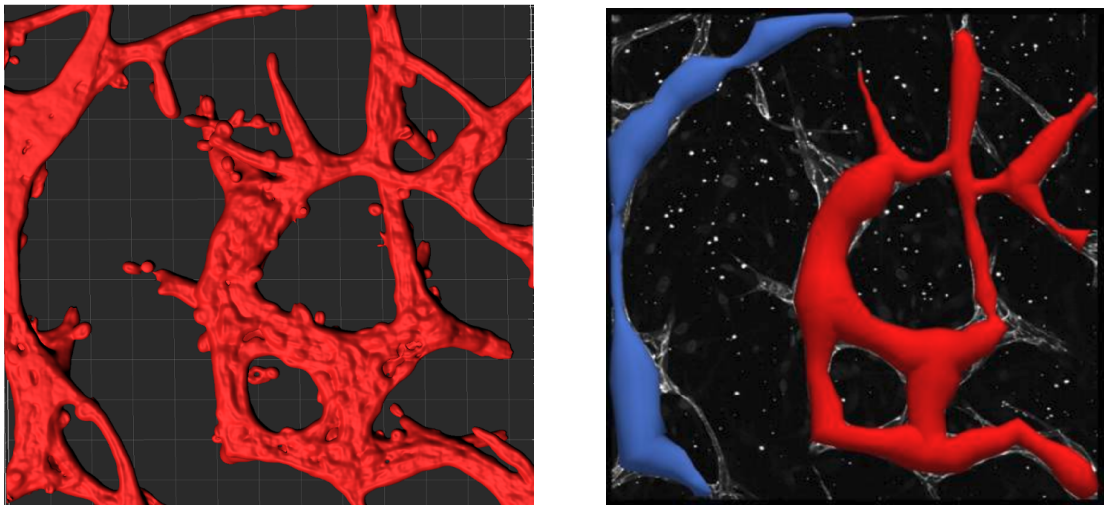
Figure 1.3 Example z-slices prior to any image processing.

1.2.2 Segmentation Challenges

Segmenting the MVNs may initially appear trivial, however accurate, three dimensional reconstructions of large scale microvascular networks continues to be a challenge. As tubular objects at the micron scale, MVN segmentations are prone to distortions and artifacts [13]. Further, because the network is highly connected, structural modeling errors at one location may yield effects on distant parts of the network.

Additionally, to obtain the z-sliced images, we are only able to image the network fluoresce. This fluoresce comes from the GFP expressed in the HUVECs. Therefore, images only represent the cells (not the lumems). Further, GFP is not expressed uniformly across a cell's surface or between different cells. Together, this means image segmentation of the microvascular networks consists of determining lumen locations from images of the lumen surfaces that are discontinuous in all dimensions and of varying intensity. For reference, sample z-slices are shown in Figure 1.3

There exist open source and enterprise software with functionalities designed to create



(a) Imaris. **(b)** Vesselucida 360

Figure 1.4 Third party software 3D segmentations.

these segmentations. Trials using Imaris and Vesselucida 360 are shown in Figure 1.4.[2] Both the Imaris and Vesselucida segmentations have positive qualities. Imaris was able to keep vessels distinct that appear to cross but are on different z-planes. Vesselucida produced a completely smooth output with no artefacts. However, the Imaris segmentation is not smooth and the lumen contains holes. Similarly, the Vesselucida segmentation neglects many smaller branches.

It is possible that with extra manual labor (pre-processing and post-processing), third party software could have provide the segmentations required for this project. However, the demonstrations with Imaris and Vesselucida show that the problem of segmenting microvascular networks is not yet completely solved.

[2]The open source software, ImageJ produced results similar to Imaris.

1.3 Necessary Fluid Dynamics

1.3.1 Reynolds Number

The Reynolds number is a dimensionless quantity that can be used to predict fluid flow patterns. Specifically, it is the ratio between inertial flow forces and viscous flow forces. Therefore, high Reynolds number flows are dominated by inertial forces and are expected to be turbulent while low Reynolds number flows are dominated by viscous forces and are expected to be laminar. The Reynolds number is defined as,

$$\mathrm{Re} = \frac{\rho u D}{\mu} \tag{1.1}$$

where ρ is the density of the fluid, u is the fluid velocity, D is the diameter of the tube, and μ is the fluid's viscosity. Thus, for fluids of constant density and viscosity, the Reynolds number is maximized by maximizing velocity and diameter.

Initial fluid experiments with the Polacheck MVNs plan to flow cell culture media. The cell culture media is primarily water and therefore is expected to be incompressible with an approximate density of $992.2\frac{\mathrm{kg}}{\mathrm{m}^3}$ and viscosity near $0.653 \times 10^{-3}\frac{\mathrm{kg}}{\mathrm{ms}}$ at 37 C [14]. Further, *in vivo* experiments have measured blood velocity within the microcirculation to an upper range of ~ 2 mm/s [15]. Finally, the maximum diameters observed in the Polacheck MVNs are $\lesssim 300$ μm. Therefore, within the Polacheck networks, the maximum Reynolds number should be $\sim 1 \times 10^{-4}$.

The recognized transition between laminar and turbulent flow occurs near Re = 2300. Therefore the flow within the network is clearly laminar. Further, as the flow satisfies the limit $\mathrm{Re} \longrightarrow 0$, the Stokes equations are a valid representation of the fluid dynamics [16].

1.3.2 Stokes Flow

As stated, Stokes flow occurs when $\mathrm{Re} \longrightarrow 0$. Under such conditions, the inertial flow forces are negligible in comparison to the viscous flow forces. Therefore, for an incompress-

ible, Newtonian fluid, such as cell culture media (water), the incompressible Naiver Stokes equations reduce to incompressible Stokes equations,

$$\mu\nabla^2\mathbf{u} - \nabla p + \mathbf{f} = 0 \tag{1.2}$$

$$\nabla \cdot \mathbf{u} = 0 \tag{1.3}$$

where, p is pressure, $\mathbf{u}$, is the fluid velocity vector, and $\mathbf{f}$ is an applied force. Equation 1.2 represents the momentum balance of the fluid and Equation 1.3 represents the incompressibility of the fluid. Unlike the full Naiver Stokes equations, the Stokes equations are linear in velocity and pressure. Therefore, they are more attractive for computational modeling [16].

1.3.3 Wall Shear Stress

Fluid flow in microvascular networks have been observed to obey a no-slip boundary on the vessel walls [8]. Therefore, the gradient of the fluid velocity must be non-zero and the vessel wall must experience a shear stress. The shear stress at the vessel wall can be calculated by subtracting the normal component of the viscous stress tensor from the total viscous stress tensor [17],

$$\mathcal{T} - (\mathcal{T} \cdot \mathbf{n})\mathbf{n} \tag{1.4}$$

where $\mathbf{n}$ is the unit normal vector to the vessel wall and $\mathcal{T}$ is the viscous stress tensor,

$$\mathcal{T} = \mu(\nabla\mathbf{u} + (\nabla\mathbf{u})^T) \cdot \mathbf{n} \tag{1.5}$$

Therefore, fluid flow solutions that obtain velocity fields through the lumen can be used to calculate wall shear stress for a given viscosity. Further, for Newtonian fluids (constant viscosity), relative wall shear stress can be calculated from only the velocity profile, as WSS is linearly dependant on viscosity.[3]

[3]This paper focuses on the development of a process to determine wall shear stress within microvasculature. For more information on the biological pathway causing mechanically stimulated angiogenesis, see Polacheck *et al.* (2017), de la Paz *et al.* (2012), and Egginton *et al.* (2016) [9, 18, 19].

1.4 Open Source Image Processing

Many of the steps in the developed image processing pipelines utilize third party software, such as `scikit-image` [20] and `SciPy` [21]. These packages are excellently documented, so the following descriptions focus on usage and effects rather than their implementations. The filters and computations described below will be referenced later in the discussion of the custom code elements. The euclidean distance transform and Gaussian blur use `SciPy`'s implementation. The rest use `scikit-image`.

Those familiar with common image processing filters and transformations can skip this section (1.4) without loss of continuity.

Contrast Enhancement

Each pixel in a single channel image has a finite range of possible intensity values. In general, we expect the vascularized regions to have higher pixel intensities than regions of empty space. Therefore, enhancing the contrast between background and vessel pixels is beneficial for segmentation.

`scikit-image` provides three methods to alter the distribution of pixel intensities within an image. Contrast stretching scales the image pixel values to include all intensities which fall within a percentile range (often 2 to 98). Histogram equalization scales the image pixel such that the cumulative distribution of intensities is linear. Finally, adaptive histogram equalization behaves as a local version of histogram equalization (linearizing cumulative distributions within subsections of the image) [20].

Opening and Closing

Morphological opening is the process of eroding (shrinking bright regions and expanding dark regions) followed by dilation (expanding bright regions and shrinking dark regions). Opening has the effect of eliminating small bright areas and opening thin dark cracks. Morphological

closing represents the opposite process (dilation followed by erosion). It has the effect of connecting nearby bright objects and closing thin dark cracks [20]. In particular, opening is beneficial for removing noise in the vessel segmentation while closing is useful for repairing broken vessel connections.

Reconstruction via Dilation

Reconstruction uses the dilation operation to isolate connected regions in an image [20]. This can be effective for segmenting images that have particularly sparse fluoresce, however aggressive use of the filter over connects the vessel segments.

Minimum Cross-Entropy Thresholding

Cross-entropy thresholding returns the intensity value that minimizes cross-entropy between the foreground and the foreground mean (lumens), and the background and the background mean (non-lumens). This value can then be used to convert the image to binary. The implementation of this method tests every possible threshold [22]. The benefit of cross-entropy thresholding is it requires no user input.

Li Thresholding

Li thresholding operates under the same theory as cross-entropy, however uses an iterative approach to finding the entropy minimum (rather than testing all possible values). Therefore, this implementation is quicker, but may return a local minimum [23].

Random Walker Segmentation

Random walker segmentation allows the user to label an image with known areas background (non-lumen) and foreground (lumen). The unknown regions are then assigned to the two groups by solving an anisotropic diffusion equation such that diffusion is difficult across high

gradients [20, 24]. This is particularly useful as a backup segmentation method when cross-entropy produces poor results. The disadvantage is the requirement of user inputs (although `mvn-analysis` helps provide these inputs).

Sauvola Local Thresholding

Sauvola local thresholding determines thesholding values for images within a user defined window. The smaller the window, the more local the threshold and the more detail (and artefacts) will be extracted. This is especially useful in 3D segmentation as information from adjacent z-plane can be utilized [25].

Skeletonization

Skeletonization reduces a binary objects to a 1 pixel wide representation. It is useful for extracting the center line of segmented vessels [26].

Euclidean Distance Transform

The euclidean distance transform sets the intensity of pixels to their minimum euclidean distance (in pixels) to the background. Used on lumen segmentations, this transformation is useful for tracking cell widths. Used on the inverse of a lumen segmentation (lumens are dark, non-lumens are light), the euclidean distance transform can be used to determine distance from a vessel (i.e. distance from an nutrient source).

Gaussian Blur

This blurs the image. It has the effect of smoothing and therefore can limit artefacts when generating a skeleton. To a certain degree, it can also be used to to fill holes prior to segmentation.

Marching Cubes

The marching cubes algorithm extracts a 2D surface from a 3D volume. This algorithm is used to generate the surface mesh OBJs of the lumens [20].

Chapter Two

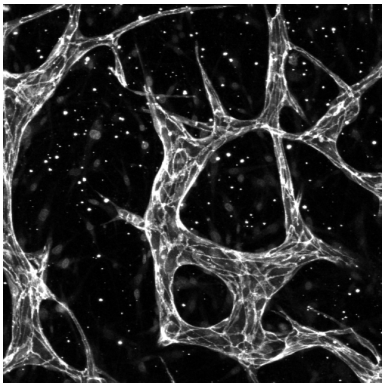
Methods

The following processes were developed specifically for images produced by the Polacheck lab. However, the software is freely available (`github.com/Ryan-A-Armstrong/mvn-analysis`) and accepts any single-channel, multi-page, `.tif` image.

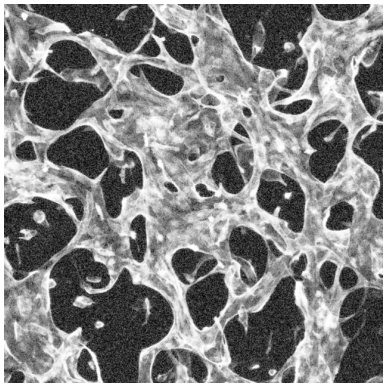
Further, the described computations were run locally on a laptop with 8 GB of RAM and an Intel i7 2.8 GHz Quad Core processor. When possible, the software parallelizes computations across all available processing units. Additionally, the software automatically limits the computation resolution in an attempt to keep memory usage under $\sim$ 7 GB. [1] For instructions on how to use the software, see the documentation on GitHub.

Throughout this chapter, two example data sets will be routinely used to demonstrate the software's major functionalities and outputs. Figure 2.1 shows these data sets as flatted, contrast enhanced, two dimensional images. Sample A is section of a network that is relatively simple and planar. Sample B is a section of a network that is relatively more complex and three dimensional. These samples were chosen due to their differences, however they alone do not represent all networks. Therefore, the GitHub repository contains major outputs of additional samples. I recommend exploring these outputs to better understand the performance of the software on a more diverse range of inputs.

[1] The 7 GB maximum usage is only for `mvn-analysis` memory usage and does not protect against *total* memory usage. i.e. running other programs along with `mvn-analysis` may cause memory overages.



(a) Sample A



(b) Sample B

Figure 2.1 Flattened and contrast enhanced images of Sample A and Sample B.

Finally, the input files used to generate the outputs of Samples A and B are included in Appendix A.

2.1 Summary of Major Outputs

Code written for `mvn-analysis` has two major goals. Goal 1 is to extract a volume mesh representing the lumens that can be used in fluid dynamics simulations. Goal 2 is to opportunistically collect as much data along the way that could be used to characterize or compare MVNs. The following serves as a reference for the major data types a user has access to.

2D Segmentation

- **2D Binary Mask**: A two dimensional binary image where vessel lumens are stored as high values and the background is stored as low values.
- **2D Skeleton**: A binary image representing the centerlines of the vessel lumens.
- **2D Distance Transform**: A grayscale image where the intensity of a pixel is the euclidean distance of the pixel from the nearest background pixel.

2D Weighted Graphs

- **NetworkX Graph**: A graph representation (nodes and weighted edges) of the two dimensional microvascular network segmentation. Nodes represent vessel branches and termini. Edges represent the connections between the nodes and carry weights related to the physical structure of the network.
- **Graph Characteristics**: Quantification and distributions of vessel segment lengths, volumes, surface area, average radii, contraction factor, fractal dimensions, and node connectivity.
- **Graph Comparisons**: Histogram and normal fit comparisons of characteristics between two or more separate networks (different networks, the same network over time, or the same network generated with graphs of different parameters).
- **Anisotropy Analysis**: Empirical measures and qualitative visualizations for the directionality (vessel alignment) within a network.

25D Surface Meshes

- **Surface Mesh**: Watertight OBJ mesh representing the lumen surface. This mesh is based on the two dimensional segmentation and assumes ellipsoidal lumens.

3D Segmentation and Surface Meshes

- **3D Binary Mask**: A three dimensional binary image where vessel lumens are stored as high values and the background is stored as low values.
- **Inverted 3D Distance Transform**: A three dimensional image where pixel values represent the distance to the nearest vessel wall (from outside the vessel).
- **3D Skeleton**: A three dimensional binary image representing the centerlines of the vessel lumens.
- **Honest-Scaling 3D Surface Mesh**: OBJ mesh representing the lumen surface. This

mesh is based on analyzing each slice of the input image and is z-scaled to the physical image acquisition parameters.

- **Rounded-Scaling 3D Surface Mesh**: OBJ mesh representing the lumen surface. This mesh is based on enforcing rounded lumens to the honest scaling representation.

Volume Meshes

- **Tetrahedral Meshes**: Each described OBJ can be converted to a tetrahedral mesh representing the lumen volume. This mesh is guaranteed to have no internal holes and can also be converted to ExodusII format for flow simulations.

Numerical Outputs

- **Volume Analysis**: Each three dimensional segmentation can also calculate volume based data. This data includes total lumen volume, vessel density, and average distance to the nearest vessel wall (from empty space).

2.2 Image Pre-Processing

The Polacheck lab imaged the microvascular networks using fluorescent confocal microscopy. The output images captured vessel walls at parallel planes along the z-axis and were saved in Olympus' proprietary multi-page, multi-channel, `.oir` file format.

ImageJ was used to open the `.oir` files in grayscale and convert them to `.tif` format. Additionally, ImageJ was used to read the metadata and record the scaling information for the sample (microns / pixel in the $x, y,$ and z directions).

Then, if the original image was multi-channeled, the `Stack To RBG` command was used to merge the channels. Finally, the `Image Type` command was used to convert from RBG to 8-Bit grayscale. The resulting single-channel, multi-page, `.tif` image was saved to be used as the data input to `mvn-analysis`.

2.3 `mvn-analysis` Computations

The following processes all occur within the `mvn-analysis` software. For more information on the effects of specific sub-processes and setting parameters, see the documentation on GitHub.

2.3.1 2D Segmentation

While the ultimate goal of the the software is to generate three-dimensional segmentations, generating two-dimensional segmentations has merit. For networks that are largely planar (centerlines exist on the xy plane), a 2D representation of the network captures structure and connectivity fairly well. Further, for all networks, a 2D segmentation describes the furthest extent of vessels in the xy direction, therefore providing critical information on known areas of no-vascularization. This information can then be used as a computational prior for the eventual 3D segmenatations.

Reading `.tif` Stacks and Setting the Global Unit

Each page of the input `.tif` is converted to a two dimensional NumPy array. The array size is then assessed to determine if any memory saving actions are required. Specifically, the maximum length of an image axis is set to 512 pixels and 750 microns. Violations of the pixel length maximum will lead to down-scaling the image axis to 512 pixels. Violation of the physical length maximum will yield an increase in global unit size (the micron / pixel scale of `mvn-analysis` outputs[2]). These limits are set to keep memory usage under 7 GB throughout the entire analysis pipeline, not just the 2D segmentation. The downscale factor

[2] `mvn-analysis` defaults to producing outputs of 1 micron/pixel, which I refer to as the global unit. For large inputs, this unit is increased in integer amounts until the memory requirements are met. i.e. a sample that spanned 500 microns would produce outputs at 1 micron/pixel while a sample that spanned 1200 microns would produce outputs at 2 microns/pixel.

and the unit conversion are stored as parameters for later calculations in order to maintain the proper physical representation. Finally, the stack of image arrays is flattened to a single 2D representation by summing the pixels along the z-axis.

Generating and Scaling the Outputs

The contrast of the flattened image is increased via one of the three methods described in section 1.4 (user chosen). Then, small holes in the vessel ares are identified by taking the difference between the morphological closing and the original contrast enhanced image. These dark spots (now represented as light spots) are added back to the enhanced image. Non-vessel light spots are filtered by taking the morphological opening of the working image. The background and vessels are then segmented using dilation reconstruction. From there, the image is thresholded by minimum cross-entropy or random-walker segmentation (user defined).

The resulting binary image has opening and closing performed (to remove artefacts and connect broken vessel segments). If specified by the user, the image is separated into connected regions and retains only the largest segment. The binary image is then up-scaled such that each pixel physically represents the global unit discussed earlier. The resulting mask is smoothed by a Gaussian blur to reduce surface roughness that can contribute to skeletonization artefacts.

Finally, the binary mask is converted to a skeleton and a euclidean distance transform. The binary mask, skeleton, and distance transform are all stored as inputs to later processes. Figure 2.2 shows an high-level review of the segmentation process for Sample A and Figure 2.3 shows a summary of the major outputs of the 2D segmentation when performed on Sample A and Sample B.

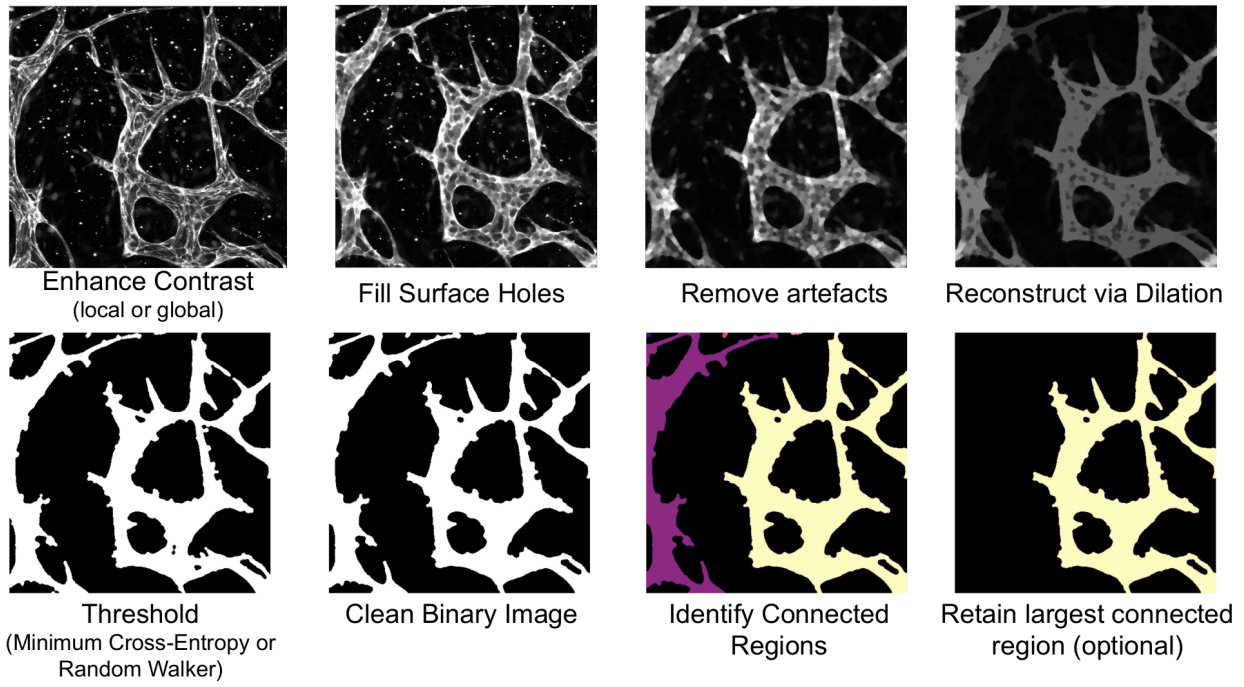
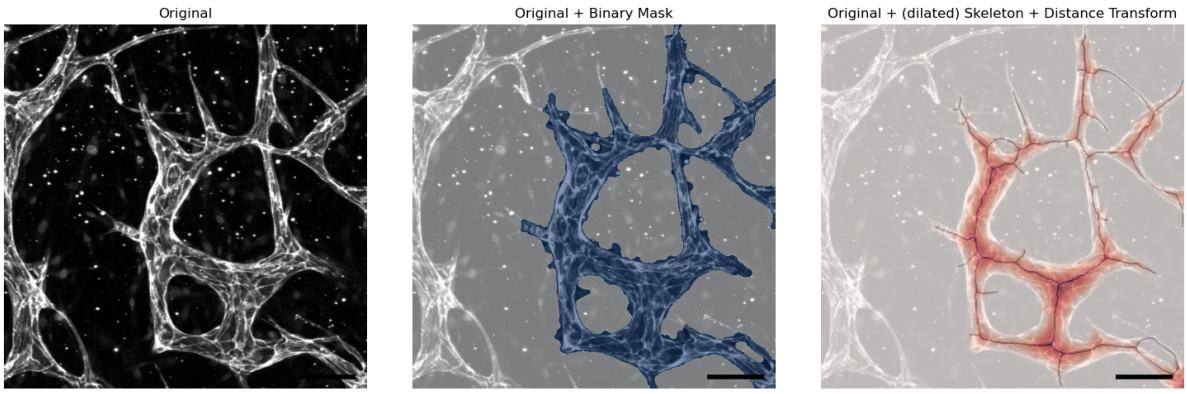


Figure 2.2 High-level overview of 2D segmentation process on Sample A.

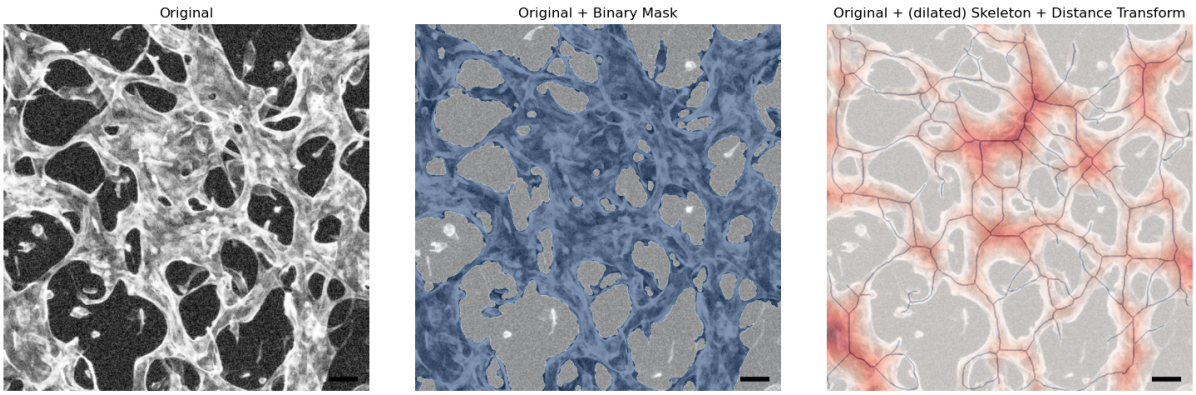
2.3.2 2D Weighted Graphs

Graphs are a data structure composed of nodes and edges. Within the Polacheck networks, nodes are the locations of vessel branches and termini (ends). Edges represent the connections between the nodes and are weighted by parameters such as segment length, radius, volume, surface area, contraction factor, and fractal dimension. Representing the Polacheck networks as graphs is motivated by the need for empirical measures which can be used to compare separate networks or a single network over time.

The the graph representations in `mvn-analysis` utilizes the NetworkX package. Graphs in the NetworkX data format have immediate compatibility with a large library of graph analysis algorithms. Therefore, generating NetworkX graphs allows for simple extendability of data analysis even after the graphs are constructed [27].



(a) Sample A: Scale 100 μm



(b) Sample B: Scale 100 μm

Figure 2.3 Major outputs from the 2D Segmentation step. Only the largest connected regions have been retained. *Left:* Enhanced contrast. *Middle:* Binary Mask. *Right:* Distance Transform and Skeleton.

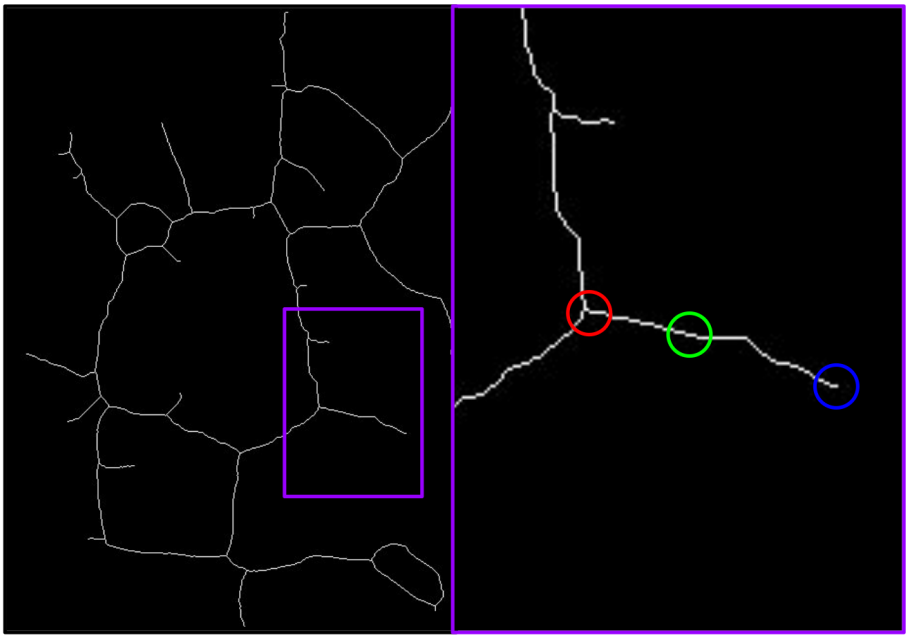


Figure 2.4 Example of branch (red) and end (blue) location for Sample A's skeleton. The vessel path (green) is not a node.

Identifying Ends and Branches

Branch and end points are determined for the network graph by analyzing the skeleton produced in the 2D segmentation step. Specifically, the skeletonization is represented as a binary array of 0s and 1s. For every pixel of value 1, the sum of the surrounding 3×3 pixel area is taken (excluding the center pixel). If the sum equals 1, then the center pixel must be an end point. If the sum is 2, the pixel is a point along a vessel path and is ignored. Else, the sum is greater than 2 and the pixel must be a branch point. This intuition can be verified for Sample A in Figure 2.4.

Generating the Weighted Graph

With the ends and branch locations known, it is possible to generate the graph nodes and edges. First, every end and branch location is represented as a NetworkX node with its euclidean coordinate as an accessible attribute. Then, a copy of the skeleton is made with the node locations removed (set to zero). Starting at each branch location, a recursive

algorithm (referred to as *SkeletonWalker*) steps along all paths in the skeleton. For each step, the *SkeletonWalker* algorithm adds the distance traveled (the global unit times 1 or $\sqrt{2}$ for a straight or diagonal step), notes the vessel radius (the global unit time the value of the distance transform at the current location), and removes the pixel from the copied skeleton.

SkeletonWalkers which reach pixels that have no neighbors are adjacent to, or at, the location of a node. The identity of this node can be determined by checking the ends and branches sets for existence of the coordinates in the immediate 3×3 neighborhood of the the *SkeletonWalker*. Once determined, a NetworkX edge can be created between the origin node of the *SkeletonWalker* and the node it reached. The edge is weighed with the parameters the *SkeletonWalker* accumulated throughout its journey, such as total segment length and the estimated segment volume and surface area. The process described can be seen as pseudo code in Procedure 1. Remember, Procedure 1 would be instantiated for every branch node in the network.

The use of a skeleton that gets destroyed (set to zero) during calculation is advantageous. Every pixel in the skeleton can only be counted once, and at the end of the processes, a completely empty image verifies every pixel was counted.

Cleaning the Graph

To meaningfully compare MVNs using the graphs, they must have similar (or reduced) noise. Therefore, generated NetworkX graphs are cleaned based on user parameters. For example, nodes are given a minimum euclidean separation and edges have a minimum length. For nodes within the minimum separation, a new node is created at their midpoint. The new node is passed the edges of the old nodes and adjusts the weights to account for the new position. Edges that do not meet the minimum length are simply removed. Additionally, edges with positive length but zero displacement (self loops) are also removed. Like the image masks, the user can decide to retain only the largest connected network. Additionally, the user can allow disconnected graphs, but specify a minimum number of nodes for a viable

Procedure 1 SkeletonWalker

Input: skeleton, distTransform, originNode, totLength, totVolume, totSurfaceArea

Input: The first call of SkeletonWalker from any origin node sets all parameter totals to 0

erodedSkeleton = skeleton − current pixel location

neighbors = non zero elements in surrounding 3x3 area

for all neighbors **do**

move to neighbor

d = distance to neighbor

r = value of distTransform at current location

totLength = totalLength + d

totVolume = $d * \pi r^2$

totSurfaceArea = $d * 2\pi r$

Call SkeltonWalker with updated location and weights

end for

if there are no neighbors **then**

Determine what destination node has been reached by searching ends and branches

Create NetworkX edge between originNode and currently reached node with the current SkeletonWalker weights

end if

network, thereby removing artefacts.

The graph cleaning process for Sample A is shown in Figure 2.5 for various euclidean distance thresholds (and requiring a connected graph with a minimum edge length of 3 μm). Further, the nodes and edges of the graphs created for Samples A and B are shown in Figure 2.6. For visual validation, the nodes are overlayed on the flattened sample image at their stored euclidean coordinates. If the graph generation process was successful, the node locations should correspond with the branches and ends observed in the image.

Numerical Analyses

In addition to setting the nodes and edges, constructing the network graphs generates a set of numerical characteristics for each sample. Specifically, each vessel segment is has a length, volume, surface area, average radius, contraction factor, and fractal (Hausdorff) dimension, where

$$\text{contraction factor} = \frac{\text{displacement}}{\text{length}}, \tag{2.1}$$

and [28],

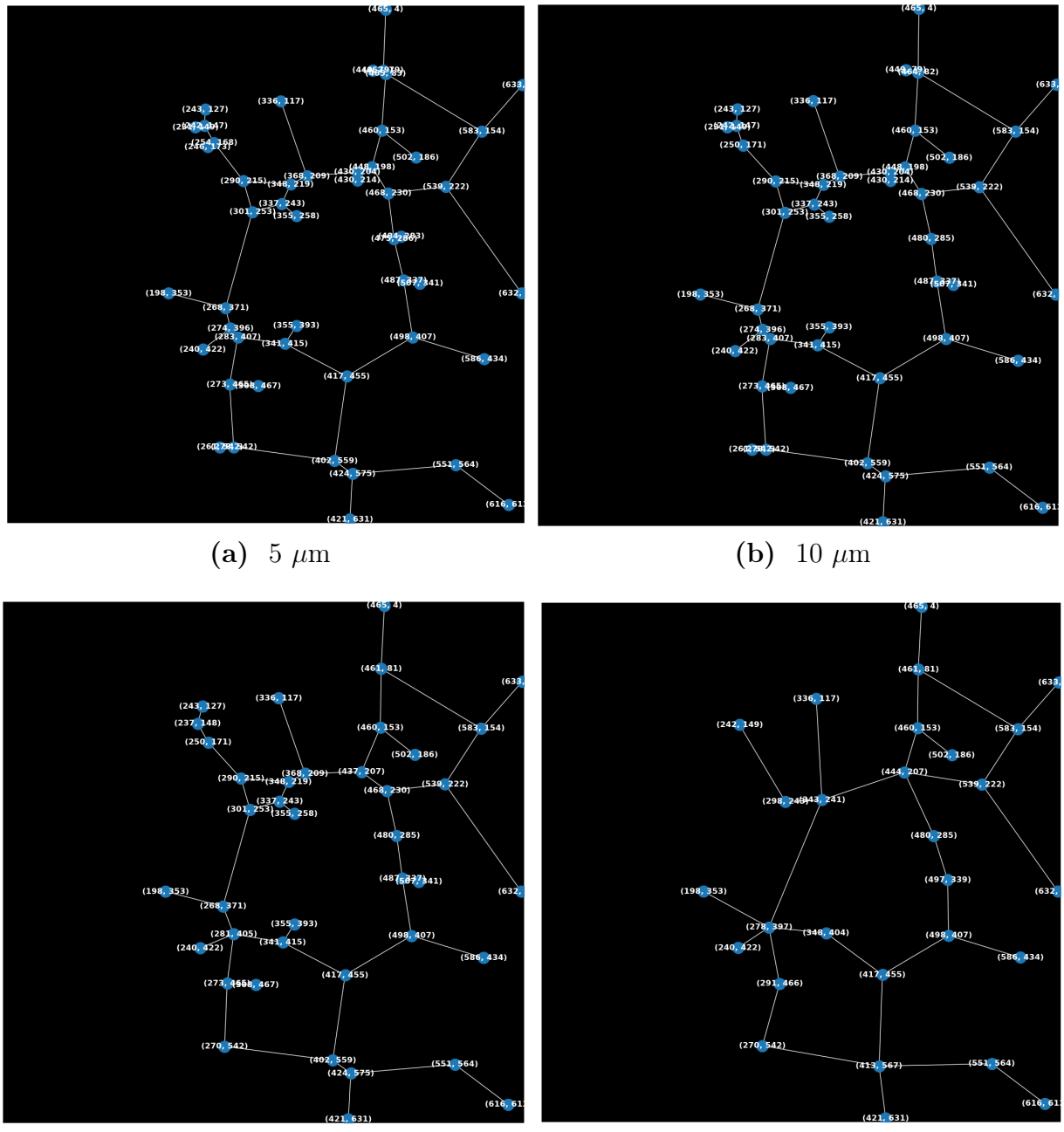
$$\text{fractal dimension} = \frac{\ln(\text{length})}{\ln(\text{displacement})} \tag{2.2}$$

These segment values are combined (summed or averaged) to provide network-wide characteristics. The characteristics for Samples A and B are summarized in Table 2.1. Note, "node connectivity" is defined later in **Graph Analyses**.

In addition to the network-wide values shown, the distribution of individual segment values can be compared between networks. This process is automated within `mvn-analysis` and explored in Chapter 4.

Anisotropy

Frequently, members of the Polacheck lab have postulated that certain MVNs–or certain sections of MVNs–have their vessels oriented in a similar direction. However, this "direc-



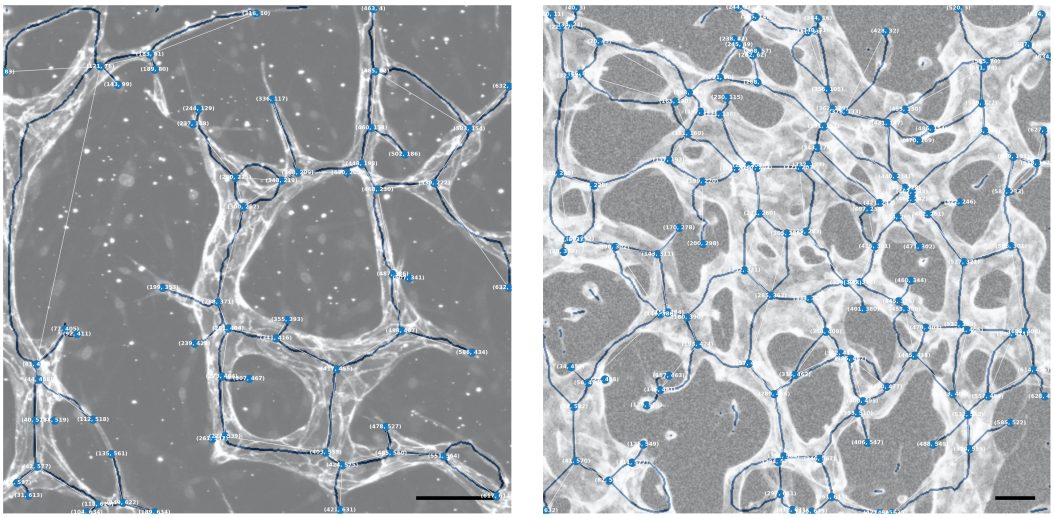
(a) 5 μm **(b)** 10 μm

(c) 20 μm **(d)** 40 μm

Figure 2.5 Output graphs for Sample A after imposing the labeled minimum euclidean distance separations between nodes.

Table 2.1 Graph summary values generated for Samples A and B.

Graph Characteristic	**Sample A**	**Sample B**
Number of branch points	34	103
Number of end points	27	33
Total length μm	4783.2	18804.1
Total surface area μm^2 **(assumes circular vessels)**	477455.3	4450488.0
Total volume μm^3 **(assumes circular vessels)**	5057376.9	122078184.2
Average branch length μm	75.9	108.7
Average branch surface area μm^2	7578.7	25725.4
Average branch volume μm^3	80275.8	705654.2
Average branch radius μm	17.3	45.0
Average fractal dimension	1.023	1.0213
Average contraction factor	0.9106	0.9144
Average node connectivity	1.1949	1.6339



(a) Sample A: Scale 100 μm **(b)** Sample B: Scale 100 μm

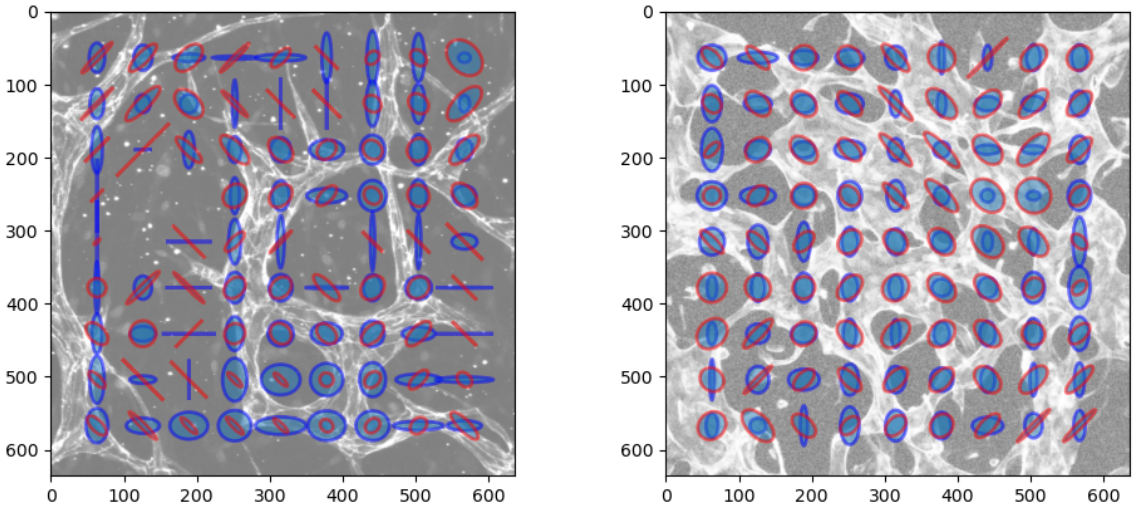
Figure 2.6 Graph representation (nodes and edges) of Sample A and B overlayed on their respective flattened images. The blue lines show the skeleton used the generate the graph and the white lines show the graph edges. The original image can be found on GitHub if there is difficulty viewing the graph edges.

tionality" characteristic is too qualitative for robust comparisons between MVNs.

Therefore, the *SkeletonWalker* code is leveraged to create an empirical measure of directionality. At each location, the *SkeletonWalker* can only move in one of four directions: along the x axis, the y axis, the $y = x$ axis, and the $y = -x$ axis. During the edge creation process, the number of moves along each axis is recorded (moves along the x and y axes increase their counts by 1, moves along the diagonals increase their counts by $\sqrt{2}$). Thus, at the end of the edge formation, the graph outputs the directionality percentages for each axis. A second version of this process weights each move based on the the current radius of the vessel. Outputs of these calculations are shown in Table 2.2. Additionally, local directionality can be displayed as a series of ellipses that have their major and minor axes scaled to the ratios between the directionality axes. See Figure 2.7 for an example.

Table 2.2 Anisotropy scores for Samples A and B.

Direction	Sample A Unweighted	Sample A Weighted	Sample B Unweighted	Sample B Weighted
Percent y-axis	0.3188	0.3334	0.2716	0.2701
Percent x-axis	0.2514	0.2421	0.2067	0.2025
Percent y=-x axis	0.234	0.2052	0.269	0.2873
Percent y=x axis	0.1958	0.2192	0.2527	0.2400
Max:	0.3188	0.3334	0.2716	0.2873



(a) Sample A **(b)** Sample B

Figure 2.7 Local directionality. Blue ellipses show x, y distribution while red ellipses show y=x, y=-x distribution.

Graph Analyses

Generating the graph using NetworkX provides access to a large library of graph analysis algorithms. Currently the only algorithm used by `mvn-analysis` is the `allPairsNodeConnectivity` algorithm. For every combination of nodes in the graph, the `allPairsNodeConnectivity` algorithm determines the fewest number of edges that would need to be cut to completely disconnect the two nodes. Similar to the characteristics described in the **Numerical Analyses** section, the graph connectivity is another way to compare MVNs. However, once flow tests begin, connectivity may serve as a predictor for network pruning. For end-user data exploration, `mvn-analysis` save the network object such that it can be opened in a separate IPython environment.

2.3.3 2.5D Surface Mesh

As previously mentioned, the 2D segmentation provides information on the maximum xy extent of the vascular cells. Therefore, transforming this data into three-dimensions provides approximate information on the maximum extent of the vessels within the entire imaged volume.

For this report, a 2.5D object is a three dimensional object that is generated only from a two dimensional representation. Specifically, the euclidean distance transform generated in the 2D segmentation step can be used to create a 2.5D surface and volume mesh representation of the lumens.

Imposing the Assumption of Round, Symmetric Vessels

The euclidean distance transform is a representation of the the lumen binary mask where pixel intensities equal the shortest euclidean distance to a non-lumen area. Further, the skeletonization of the lumen binary mask represents the vessel center-lines. Therefore, the product of the distance transform and skeleton yields a skeletonization with non-zero ele-

ments equal to the value of the distance transform at the same location.

If vessel cross sections are circular and the MVNs are completely planar (all vessel centerlines exist on the xy plane), then the weighted skeleton would describe the local radius of the vessels. Thus, an accurate, three dimensional representation of the network could be created by projecting the radii into three dimensions and creating circular tube surfaces.

Obviously, the MVNs do not have perfectly circular cross sections nor are they completely planar. However, enforcing these conditions allows us to follow similar logic and robustly generate surface and volume meshes that share characteristics with the real, three dimensional geometries.

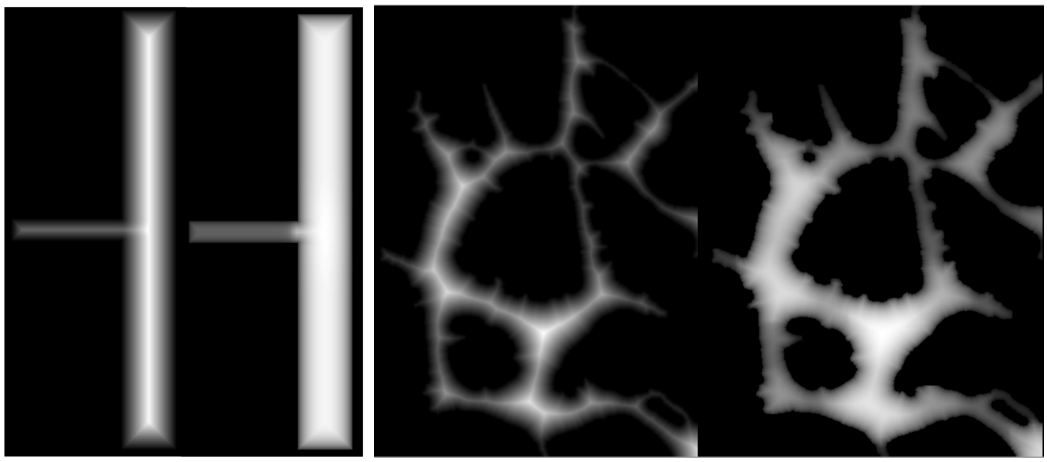
Projecting Into 3-Space

To produce a 2.5D volume, we fill voxels above and below the plane of the distance transform according to their intensity. For example, if a pixel has intensity 5, then the 5 voxels above and 5 voxels below the xy plane are filled. However, doing this procedure directly on the euclidean distance transform would create diamond lumen cross sections (the values the distance transform are linear with respect to the perpendicular distance from the vessel centerline). Therefore, prior to the voxel filling procedure, the distance transform is scaled to have a quadratic relationship with respect to perpendicular distance from the vessel centerline.

At each location along the skeleton, the distVal $\times$ distVal neighborhood of the distance transform is sampled (where distVal is the distance transform value at the current location). Each pixel within the neighborhood is then assigned a scaled value according to,

$$\text{scaled} = \sqrt{(\text{max} - \text{min})^2 - (\text{max} - \text{original})^2} + \text{min} \tag{2.3}$$

where "original" represents the original pixel value and "max" and "min" are the maximum and minimum values in the neighborhood. Once every skeleton location has been sampled, the final transformed image is represented by the average of each pixel's scaled values. The



(a) Artificial data **(b)** Sample A

Figure 2.8 Transformation of the euclidean distance transform (*left*) to the quadratic distance transform (*right*).

procedure has the effect of converting the euclidean distance transform to a quadratic transform and smoothing the result. From there, voxel filling can proceed. Figure 2.8 shows the effect of this transformation on both the euclidean distance transform of artificial rectangular data and Sample A.

Extracting Surface Representation

The marching cubes algorithm is used to extract the 2D surface from the 3D binary image. PyMesh is used to clean and convert the extracted triangle representation to an OBJ file.

Z-Scaling and Examples

The described procedure will create circular lumen cross sections. However, general ellipsoidal cross sections can be obtained by pre-scaling the distance transform. For example, by multiplying the distance transform by 0.5, the resulting lumen cross sections will have xy axes twice the magnitude of their z-axes. When selected by a user, `mvn-analysis` will automatically choose this scaling parameter to force the 2.5D volume to have z-depth equal to the z-depth of the sample (based on image metadata). Figure 2.9 shows the 2.5D representation

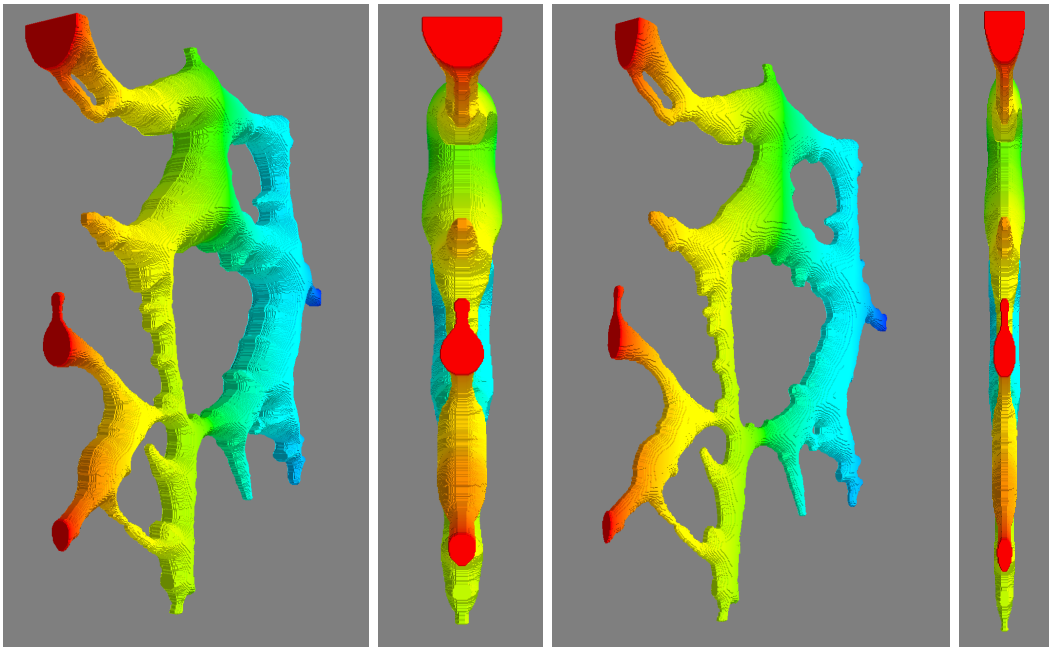
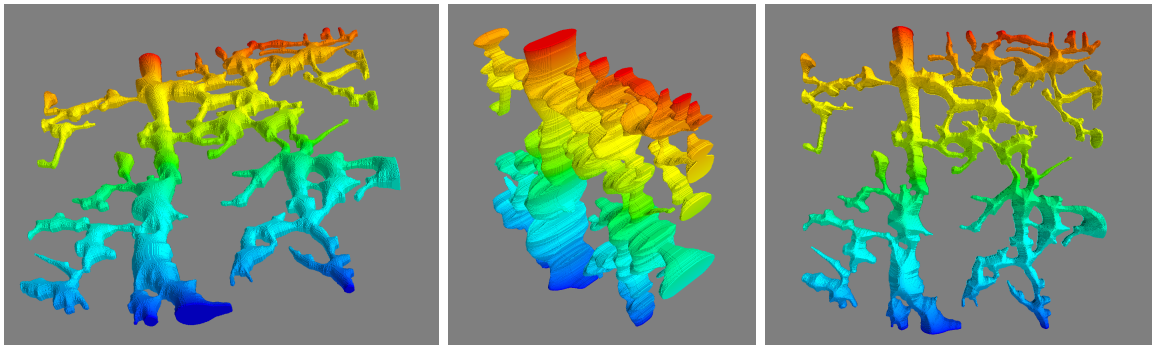


Figure 2.9 Comparison between enforced circular lumens (a, b) and enforced z-depth (c, d) for a highly planar network (Sample A).

of Sample A using both circular lumens and automated ellipsoidal lumens.

As can be seen, enforcing proper z-depth decreases the height of the lumen cross sections. We assume for sample A, this re-scaling more correctly fits the physical network as the circular lumens have a z-depth greater than than the total image acquisition depth. However, using the automated z-scaling is not always beneficial. For example, Figure 2.10 shows 2.5D representations of a more three dimensional MVN.

As can be seen, the automated z-scaling generates unrealistic lumen cross sections. This occurs because the sample is highly three dimensional, and therefore image depth is a poor proxy for vessel thickness. Finally, Figure 2.10 (c) shows the z-fitted 2.5D mesh looking directly along the z-axis. From this view, the mesh looks identical to the circular lumen mesh, demonstrating that only the z-components are altered during these procedures.



(a) (b) (c)

Figure 2.10 Comparison between enforced circular lumens (a) and enforced z-depth (b, c) a highly three dimensional network.

Uses

As shown in the comparisons of Figure 2.9 and 2.10, the degree to which the 2.5D mesh represents the MVN is dependent on individual MVN characteristics. Beyond losing information about vessel z-location, the 2.5D mesh representations will have connections between vessels that appear to cross but exist on different z planes. Therefore, the procedure is not high fidelity. Nevertheless, the generation of 2.5D meshes is robust and lightweight. The meshes generated are guaranteed to be water tight, smooth, and provide flat surfaces at inflow-outflow locations. Therefore, they are easily adapted to fluid flow models.

2.3.4 3D Segmentation and Surface Mesh

Because the MVNs grow in three dimensions, a three dimensional segmentation is required to fully capture the structural data. The following sections discuss methods developed in an attempt to overcome the challenges of 3D segmentation described in Section 1.2.2.

Thresholding Image Planes

The 3D segmentation process takes in the original `.tif` image represented as a stack of NumPy arrays. Each array has its contrast enhanced and noise removed. Then the entire

(a) 1 **(b)** 14 **(c)** 21 **(d)** 31 **(e)** 35

Figure 2.11 Examples of the binary images at slices (n/39) through the Sample A 3D reconstruction

stack (represented as a single 3D image) has threshold values calculated using Sauvola local thresholding. Somewhat uniquely, the calculated thresholds are not used to convert the image to binary, but are rather used as the input images to Li thresholding (which ultimately ends in conversion to binary). Morphological opening and closing are performed, however unlike the 2D segmentation, disconnected regions are always retained.

At this point, the MVN is represented as a stack of binary images. Figure 2.11 shows examples of the binary slices produced in this step. As predicted from Section 1.2.2, the areas of lumen are not full and are often not connected. Therefore, the following sections are focused on filling these errors.

Lumen Filling via the Ellipsoid Octant Method

The first procedure we developed to fill the lumen holes is referred to as the ellipsoid octant method. The motivation behind the method is the assumption that dark voxels that have a high ratio of light voxels in a given neighborhood are more likely to be part of the lumen. Therefore, the ellipsoid octant method creates a three dimensional binary mask that represents an ellipsoid of a given xy dimension and z height. The ellipsoid is then broken into 8 equal sections (hence the octant) and centered on dark voxels within the 3D binary array that represent the current MVN segmentation.

Each octant then sums the product of itself and the segmentation, and divides by its

own sum. This yields the percentage of the voxels spanned by the octant that are bright. A threshold percentage is provided to the method, and an octant is considered "on" if it reaches the threshold.

Additionally, the method is provided an acceptance rule. If the *total ellipsoid* acceptance rule is used, the dark voxel being investigated is set to bright (included in the lumen) if the total percentage of bright neighbors for the whole ellipsoid is greater than the threshold. If instead, the *total octant count* acceptance rule is used, the dark voxel is set to bright if the number of "on" octants is greater than a given threshold. Finally, it the *opposite pairs count* condition is used, the voxel is set to bright if the total number of pairs of octants that are spatially furthest from each other and both "on" is greater than a threshold. The different acceptance methods–along with their respective thresholds–are designed to provided various levels of rigger to the filter.

Importantly, the ellipsoid octanct method uses the 2D segmentation mask as a prior. Specially, the method only tests voxels that are dark in the 3D segmentation, but bright in the 2D mask. As the 2D segmentation captures the greatest extent of all vessels, any area not included in the 2D mask also should not be included in the 3D segmentation. This qualification both increases the speed of the algorithm (by reducing the search space) and prevents the method from filling areas that are outside the lumen but densely vascularized.

The method starts with an ellipsoid of a given size and iterates over the search space. Then the ellipsoid size is decreased and the processes is repeated until the ellipsoid is only searching adjacent voxels. The ellipsoid octant method has a finite reach and the main goal of the filter is to close gaps in the vessel walls.

Lumen Filling via the Ray Casting Method

The ellipsoid octant method returns a new binary array representation of the MVN segmentation. Ideally, the gaps in the vessel walls will have been sealed. However, large holes in the lumen usually persist. Therefore, a ray casting method was developed to fill remaining

lumen holes.

Once again starting from dark voxels that are bright in the 2D segmentation, the ray casting method generates three dimensional trajectories (the traditional ϕ and θ angles of spherical coordinates). The method then propagates the rays through the binary image and notes the values of the voxels encountered. Rays that originate in the lumen should never reach the array edges without passing through a bright pixel. Therefore, the ray tracing method calculates the percentage of rays which reach the array edge. This percentage can then be used as an acceptance criteria for the voxel.

The total number of rays–as well as their distribution (uniform, random, or xy planar)–can be controlled by the user. The ray casting method has the advantage of acting at infinite distance. The filter's success is not dependent on the size of the lumen hole, but rather on how well sealed it is. Figure 2.12 demonstrates the effects of the ellipsoid octant method and ray casting method.

Lumen Filling via Iterative Meshing

Examining Figure 2.12 shows (a, b) to successfully fill the holes in the lumen. However, a closer examination of (c) shows holes can still persist (look near x=350, y=375). Therefore, one final method is used to fill the lumens.

The marching cubes algorithm was mentioned previously in reference to extracting a 2D surface from a 3D volume image. Intuitively, these surfaces occur where regions of white meet regions of black. Therefore, holes in the lumen can be represented as surfaces within the lumen.

To take advantage of this representation, the 3D binary image is first separated into connected regions and the largest section is retained. Then, this connected image is converted to surface meshes using the marching cubes algorithm. The largest output mesh (measured by number vertices) is assumed to represent the exterior lumen surface. Therefore, all other meshes represent holes within the lumen.

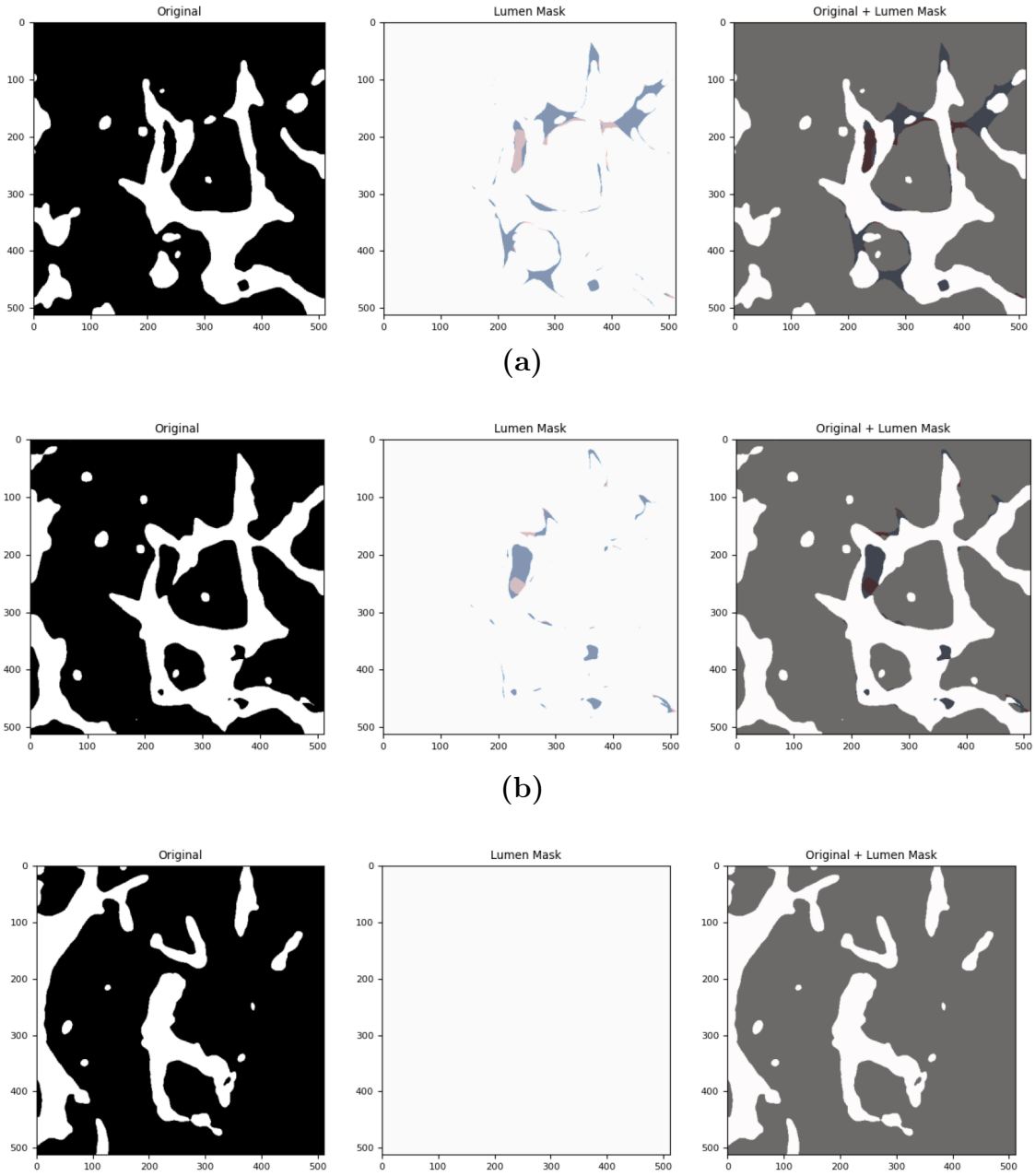


Figure 2.12 Performance of the lumen filling methods on Sample A. The blue represent the filling achieved by the ellipsoidal octant method and the red represents the filling achieved by ray casting.

Bounding boxes are generated for the sub-surfaces. Voxels of the 3D binary array that are within the bounding box coordinates are identified as holes and have their dark values set to light values. The updated image is then re-meshed and re-examined for holes. The process continues until meshing the 3D binary image returns a single surface. The resulting binary array represents the largest connected region of lumen.

Scaling and Interpolating

The final major step in segmenting the lumen involves interpolating between z-planes. The MVN images are taken in discrete z-steps that are often up to 6× the size of the distance between pixels.

Within `mvn-analysis` two binary image planes are interpolated by first representing them as signed euclidean distance transforms (the euclidean distance transform of the image minus the euclidean distance transform of the inverted image). Then, the weighted average of the two signed transforms is taken (weighted on how close the interpolation should be to each plane). Finally, the interpolated plane is taken to be the values greater than 0 in the weighted average.

Interpolated planes are added to the 3D binary MVN segmentation such that each z-step represents the global unit. Then, each plane in the image is scaled in the x and y axes such that their respective steps also equal the global unit. The final result is a physiologically relevant binary representation of the largest connected lumen.

3D Surface Meshes

Using the marching cubes algorithm on this binary array yields the "honestly scaled" mesh mentioned in Section 2.1. However, while the procedures used to fill lumen holes are relatively robust, the 3D segmentation still struggles to accurately capture the entire lumen. In particular, the upper and lower sections of the vessels are often missing (leading to flat surfaces on the top and bottom of the honest mesh). This occurs for two reasons. First,

the upper and lower sections of the lumens are represented by thin lines when observed as z-slice. Therefore, they are often excluded in the threshold (as their inclusion would also bring artefacts). Second, as mentioned, large z-stepping while imaging can simply skip these areas.

To attempt to reconstruct these missing areas, the process used to generate the 2.5D meshes is applied to each plane in the binary array. This enforces curved vessels on the flat planes. Pre-scaling the distance transforms–as discussed in Section 2.33–is used to mitigate the effect of the smoothing.

Figure 2.13 demonstrates the effect of the filter. As can be seen, the rounding filter reestablishes curvature in the top and bottom planes of the network. Further, Figure 2.14 shows the 3D segmentation is able to have vessels cross in separate z-planes without being joined. Finally, Figure 2.15 shows the 3D mesh of Sample B compared to its 2.5D representation. Taken together, these figures show how segmenting in three dimensions has large impact on vessel connectivity.

2.3.5 Surface Meshes to Volume Meshes

The surfaces generated in the following sections are saved as Wavefront (OBJ) triangle meshes. However, the Stokes flow solver used later in the analysis requires volume meshes. For this, the TetWild software package is used to convert the OBJ to a tetrahedral volume MSH file [29].

2.3.6 Numerical Outputs

Within `mvn-analysis`, three dimensional binary arrays of the lumens are constructed in three separate ways: 2.5D transformation, honest scaling 3D segmentation, and rounded 3D segmentation. During each of these processes, the 3D binary array is used to calculate volumetric characteristics of the network. The outputs of this data is shown in Table 2.3

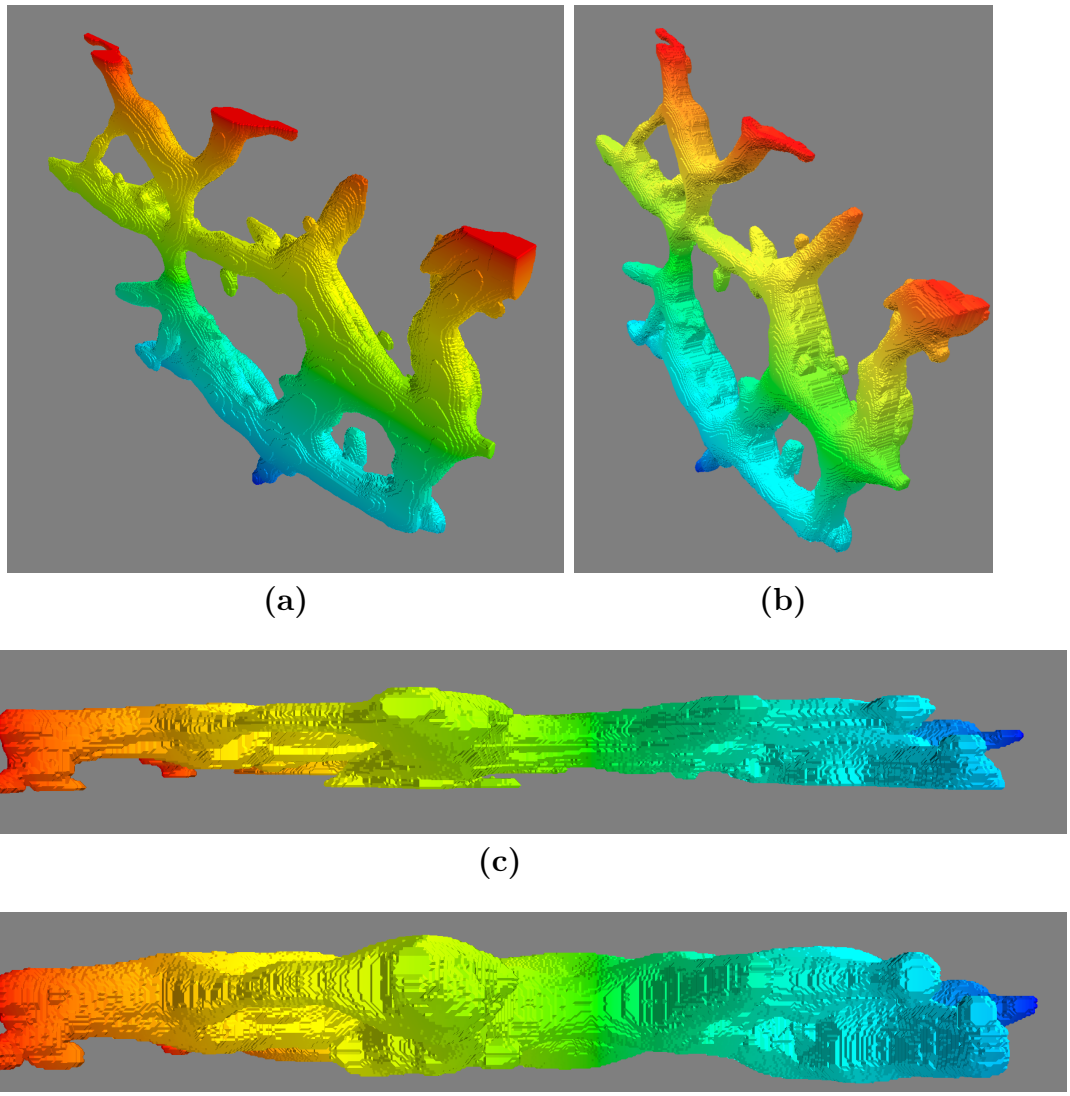


Figure 2.13 "Honest" scaling 3D mesh (a, c) and "rounded" scaling 3D mesh (b, d) for Sample A. For the rounded scaling, 0.5 prefactor was applied to the distance transforms.

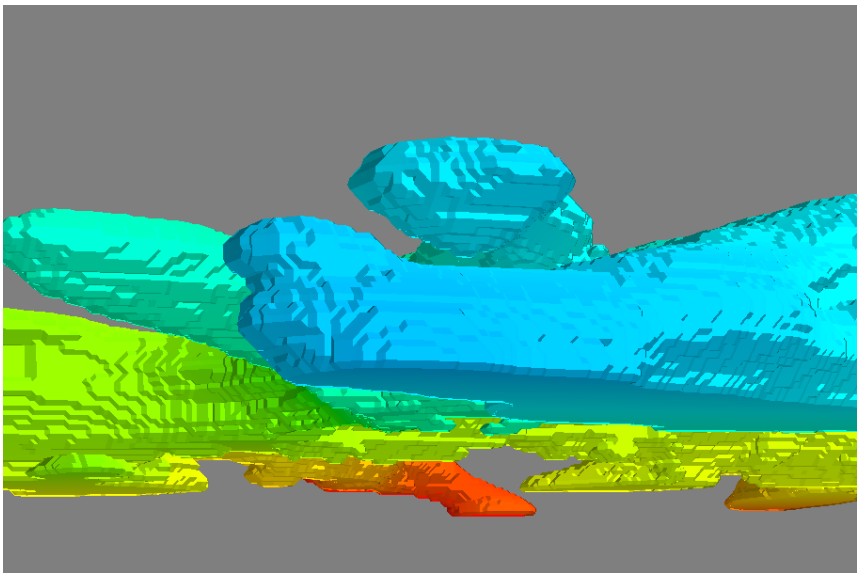
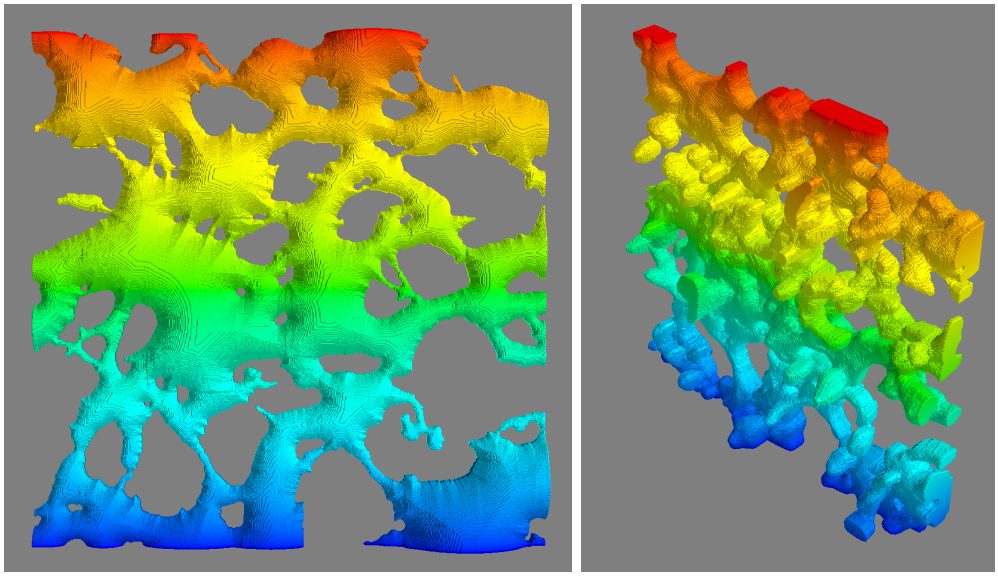


Figure 2.14 Demonstration of vessels crossing in different z-planes. These the vessels shown are connected in the 2D and 2.5D representations of the network.



(a) **(b)**

Figure 2.15 2.5D (a) and 3D (b) segmentations for Sample B.

Table 2.3 Volumetric data generated for Sample A.

Source	**25D**	**Honest 3D**	**Rounded 3D prefactor-0.5**
Connected	True	True	True
Bounding Box (z, x, y) (μm)	(43, 635, 446)	(40, 635, 469)	(69, 635, 469)
Total volume μm^3	2003356	2688945	3391810
Vascular density ($\frac{VesselVolume}{TotalVolume}$)	0.1645	0.2257	0.1650
Avg dist from vessel wall μm	26.7	27.2	29.9
Max dist from vessel wall μm	132.6	143.2	145.3
Avg max dist from vessel wall per z-plane μm	131.6	140.8	141.6

and Table 2.4

Importantly, the vascular density and average distance from vessel wall calculations are independent on the size of of network (physical or imaged) and rather dependent on the network morphology. Therefore, these characteristics are useful for comparing MVNs. Further, they are biologically relevant, as vascular density is related to the volume of blood being transported by the network while the distance from a vessel wall characterizes how well the network is able to provide nutrients to non-vascularized regions.

Table 2.4 Volumetric data generated for Sample B.

Source	**25D**	**Honest 3D**	**Rounded 3D prefactor-0.5**
Connected	True	True	True
Bounding box (z, x, y) (μm)	(238, 1270, 1270)	(232, 1270, 1270)	(302, 1270, 1270)
Total volume μm^3	91363744	101779440	109411016
Vascular density ($\frac{VesselVolume}{TotalVolume}$)	0.2380	0.2651	0.2246
Avg dist from vessel wall μm	45.7	46.4	48.2
Max dist from vessel wall μm	183.0	216.8	230.5
Avg max dist from vessel wall per z-plane μu	162.0	166.3	172.5

2.4 Performance Notes

2.4.1 Speed

Many operations within `mvn-analysis` require iterating over large, three dimensional arrays. Therefore, whenever possible, these processes utilize the parallel Numba just-in-time compiler to pre-compile Python loops to machine code and execute the process in parallel. Additionally, information gathered from the 2D segmentation process (fast) is used to optimize the 3D segmentation process. For example, the algorithms that fill internal lumen holes do not search all space, but rather only search the space that is spanned by the 2D image mask (as the 2D lumen will represent the extreme values of the lumen). Nevertheless, an obvious speed increase could come from transferring the project from Python to C++.

2.4.2 Memory

The major memory saving parameters have already been discussed (increasing the global unit and thereby decreasing the image and mesh resolutions). However, the software could be further optimized to limit memory usage by transferring from a NumPy image representation to a SciPy sparse array representation. Even the most vessel dense image samples were majority empty space. Further, many of the operations are embarrassingly parallel. Therefore, large image sets could be broken into smaller sub samples, processed, and recombined.

Chapter Three

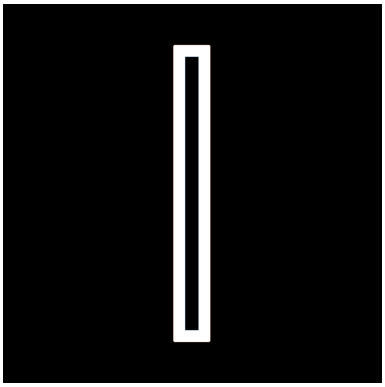
Preliminary Validation

The segmentations and calculations produced by `mvn-analysis` are understood to be imperfect. However, quantification of this error and proof of algorithmic validity is critical if the software is to be used in a research setting.

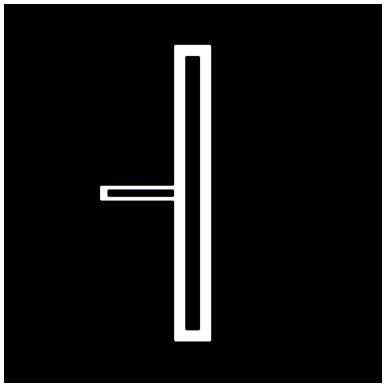
3.1 T-Junction Artificial Data

In order to determine the accuracy of the the segmentation process, a simple artificial data set was created. In the data set, a $100 \times 20 \times 10$ hollow box makes a T-Junction with a $50 \times 400 \times 50$ hollow box. The artificial data is provided as a `.tif` stack with z-slices at every other voxel. Therefore, to accurately return volume parameters, `mvn-analysis` must accurately fill the lumen holes and interpolate in the z-direction. The major results of the test are shown in Figure 3.1.

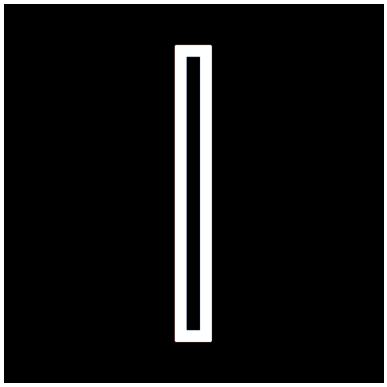
The volume analysis returned a value of 977980 pixels^3, underestimating the known value of 1020000 pixels^3 by approximately 4 percent. When the analysis was re-run using the entire sample resolution (i.e., not skipping z-planes), the error was reduced to approximately 2 percent, indicating that some–but not all–of the discrepancy comes from data lost in the discrete z-steps. While this test far from validates the software as a whole, it does support the validity of the core segmentation process.



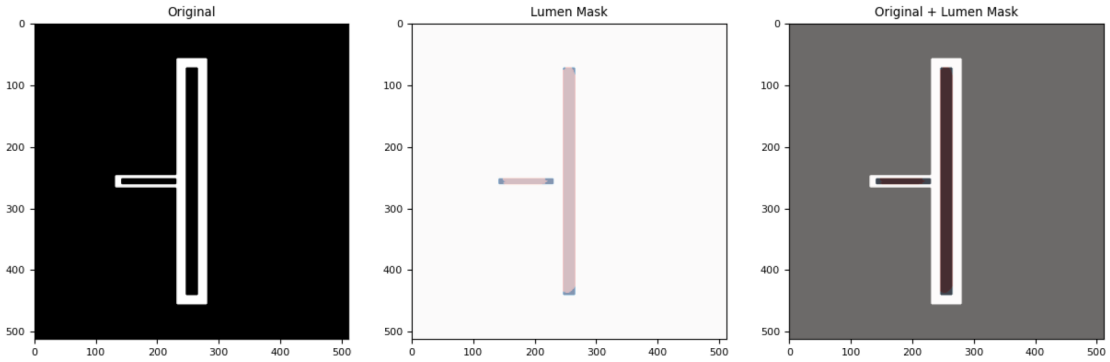
(a) Slice 5



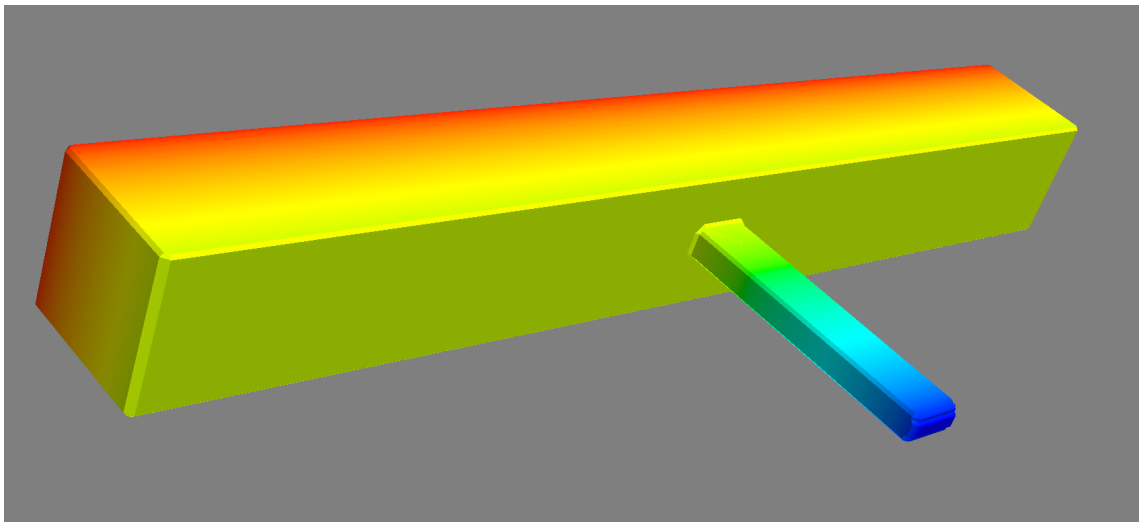
(b) Slice 25



(c) Slice 45



(d) Lumen filling



(e) Resulting 3D Segmentation

Figure 3.1 Review of artificial data segmentation. (a,b,c) show binary slices, (d) shows an example of lumen filling, and (e) shows the "honest scaling" surface mesh segmentation.

Table 3.1 Sample B volume calculations.

Source	**2.5D**	**Honest 3D**	**Rounded 3D**	**Graph**
Volume	91363744	101779440	109411016	122078184
Mean	106158096		**Std.**	11203528

3.2 Cross-Method Validation

Within the segmentation and analysis, total vessel volume was calculated in four separate ways: 2.5D transform, honest 3D segmentation, rounded 3D segmentation, and graph generation. Comparing these values gives a high-level check for consistency across the software (but not of absolute accuracy).

The graph based volumes in Table 2.1 retained smaller, disconnected networks, while the volumes in Tables 2.3 and 2.4 were generated only from the largest connected network. Therefore, the comparison is only carried out for Sample B, as there is a negligible difference between its connected and unconnected representations. The volume data is reproduced in Table 3.2, along with the standard deviations of the set.

The standard deviation reported in Table is ~ 10 percent the value of the mean. Once again, these values are not expected to exactly match, however the reasonably similar values indicates a general consistency of volume calculations across methods. Taken together, the T-Junction and cross-validation results support the validity of the `mvn-analysis` processes, however, more representative artificial data needs to be produced and tested on all sub-processes before full confidence can be achieved.

Chapter Four

Stokes Flow Simulations

As discussed, the ultimate research goal is the relation of fluid flow forces to angiogenesis. Therefore, this chapter demonstrates how the outputs of `mvn-analysis` can be used to visualize the pressure, velocity, and wall shear stress distributions for a MVN segment. This demonstration uses the 2.5D lumen segmentation of Figure 2.10.

4.1 Computation Pipeline

4.1.1 Mesh Pre-Processing

At the end of the `mvn-analysis` pipeline, MVN lumens are represented as an ExodusII tetrahedral volume mesh. In order to run fluid computations on the mesh, boundary conditions must be assigned to the surfaces. Therefore, the third party software, Trelis, is used to label groups of surfaces that shared boundary conditions.

4.1.2 `BeatIt`

`BeatIt` is Dr. Simone Rossi's suite of "c++ code for heart biomechanics and more," and is available on GitHub (`github.com/rossisimone/beatit`). In particular, `BeatIt` provides a finite-element solution to the Stokes flow equations. Within `BeatIt`, surfaces can be prescribed pressure boundary conditions, no-slip boundary conditions, and zero-traction bound-

ary conditions. `BeatIt` then outputs velocity and pressure fields based on the geometry and prescribed boundary conditions.

4.1.3 Visualizing Wall Shear Stress

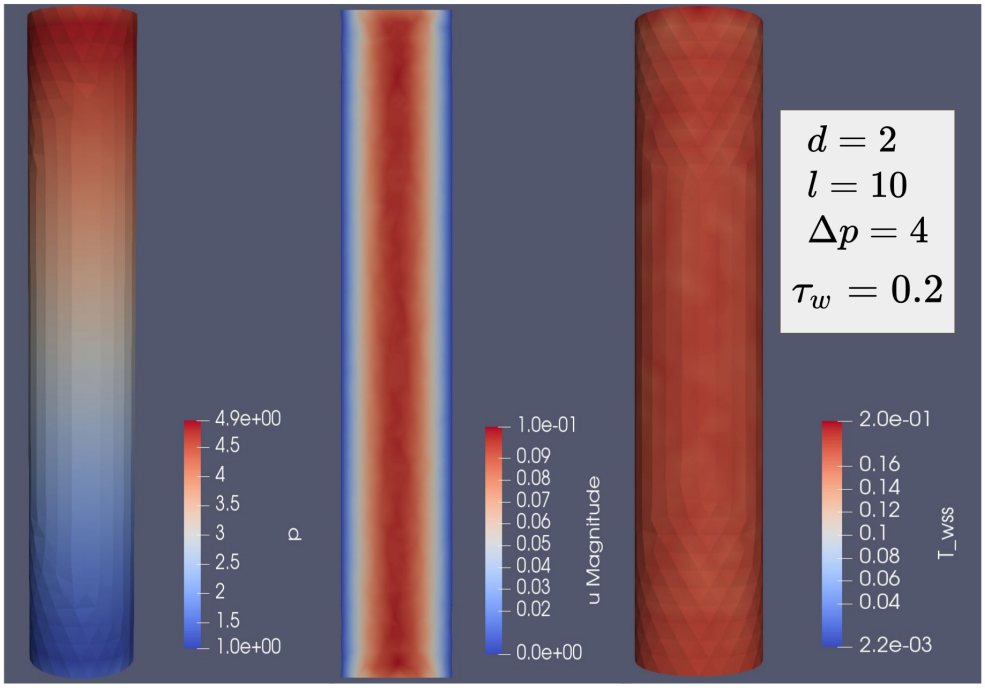
The third party software, ParaView, can be used to calculate the gradients of the the velocity fields generated by `BeatIt` and the normal vectors of the surfaces generated by `mvn-analysis`. Therefore, theses velocity gradients and normals (along with the known viscosity of the fluid) can be applied to Equations 1.4 and 1.5 to generate wall shear stress vectors and magnitudes. These calculations were implemented as a `Programmable Filter` in ParaView (filter also available from GitHub) and are used to visualize wall shear stress for the simulations.

4.2 Validation

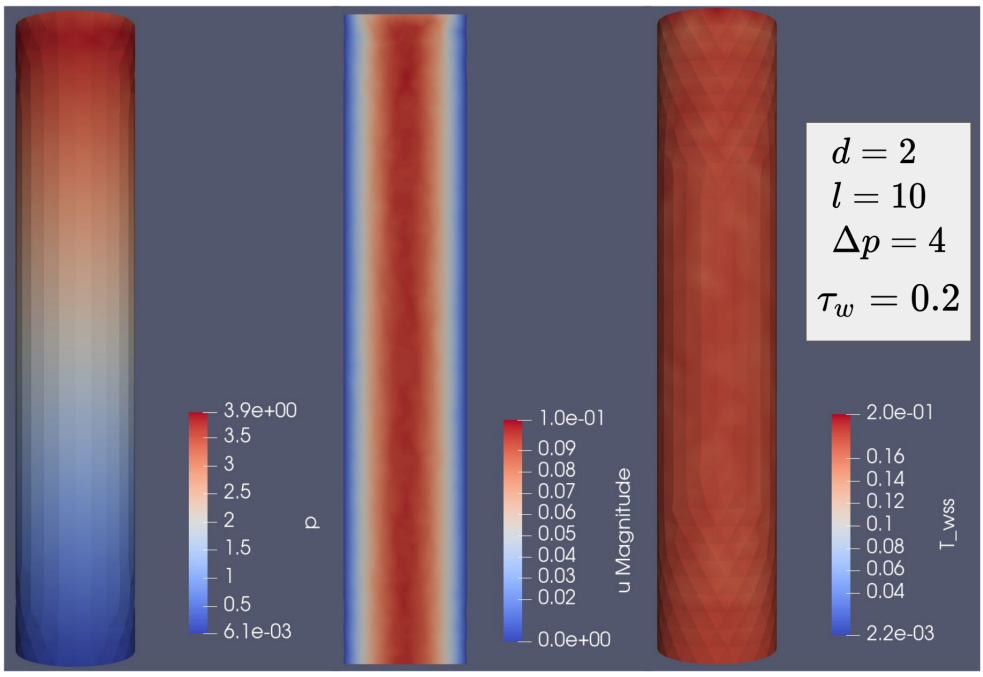
To validate the procedure described, flow simulations were conducted on a cylindrical pipe 10 units in length and 2 units in diameter. The ends of the pipe were set to a pressure differential of 4 units (using both a pressure to pressure boundary scheme and a pressure to zero traction scheme). The walls of the pipe were assigned no-slip boundary conditions. Under such conditions, the wall shear stress is known to be,

$$\tau_{wss} = \frac{d\Delta p}{4l} \tag{4.1}$$

where d is the diameter of the pipe (2 units), Δp is the pressure difference along the pip (4 units), and l is the length of the pipe (10 units). Therefore, the simulations described predict wall shear stresses of 0.2 units. Figure 4.1 show the results of the simulations matching this prediction exactly.



(a) Pressure (5 units) → Pressure (1 units)



(b) Pressure (4 units) → Zero Traction

Figure 4.1 Validation of WSS calculations using `BeatIt` and ParaView post-processing. The scales are in relative units. *Left*: pressure, *center*: velocity, *right*: wall shear stress.

4.3 Usage on an Example MVN

Having validated the method on synthetic data, the 2.5D mesh from Figure 2.10 was used represent the flow simulations within MVN geometries. Flow tests have not yet been conducted on the MVNs, so the input conditions were chosen arbitrarily to produce results in the physiological range (mm/s velocity scale). Specifically, a 5.0 Pa pressure was applied to one end of the parent channel and a 3.0 Pa pressure was applied to the other end. All other inlet-outlets on the image xy boundary were assigned zero-traction conditions. No-slip boundary conditions were applied to the vessel walls.

The geometry and pressure distribution is shown in Figure 4.2. The resultant velocity distribution (at the center z-plane) and wall shear stress distribution (assuming flowing water) are shown in Figure 4.3. While the results of this particular simulation are meaningless (the boundary conditions were not based on observed data), the procedure demonstrates the feasibility of using `mvn-analysis` and `BeatIt` to determine wall shear stress values within MVNs once proper boundary conditions are known.

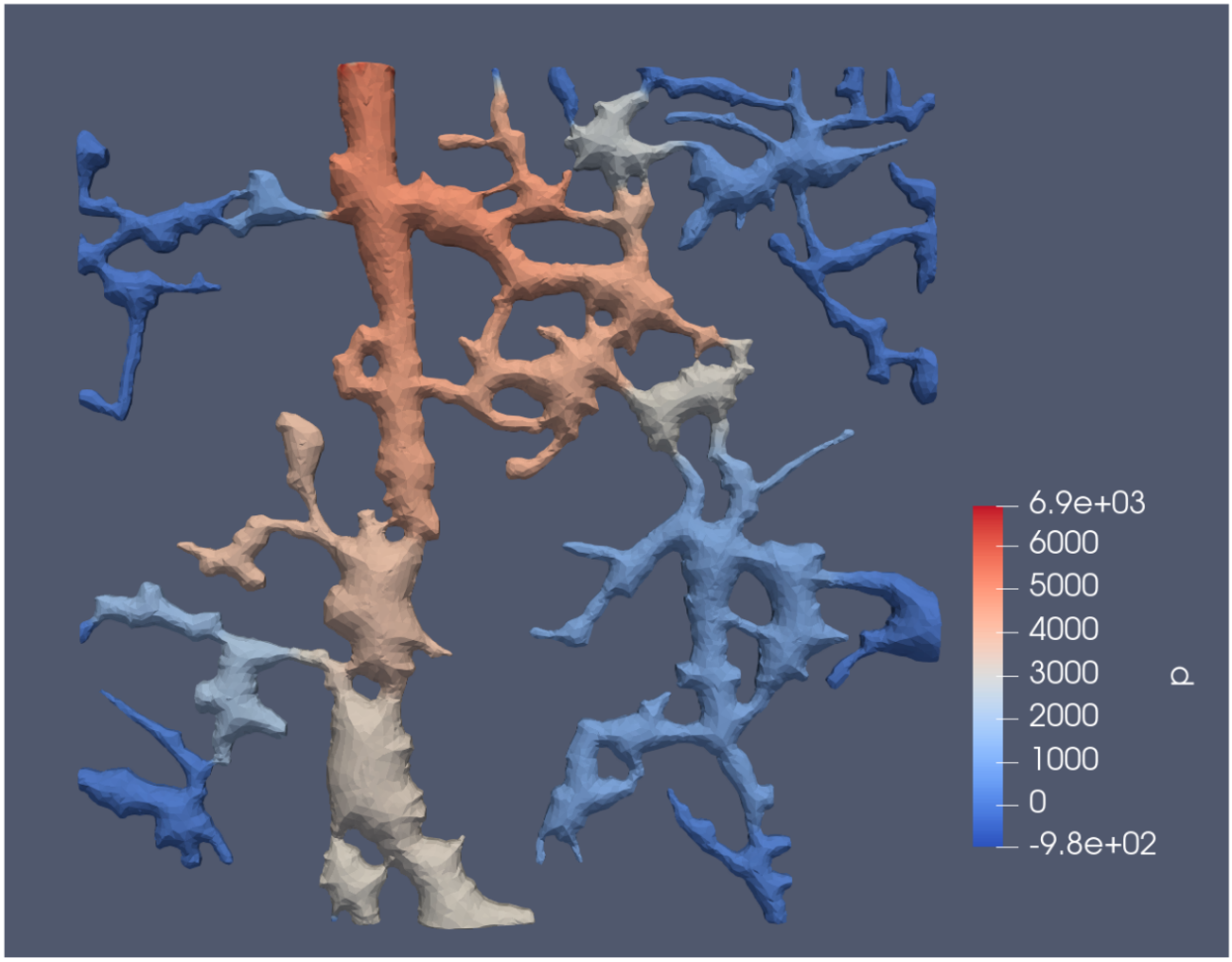
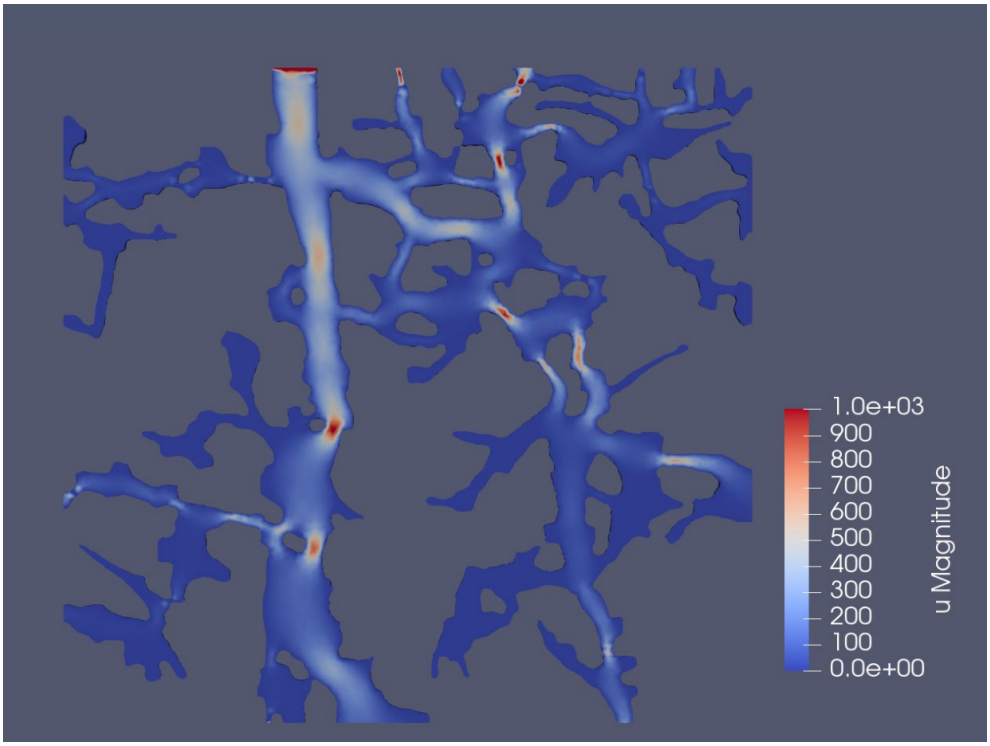
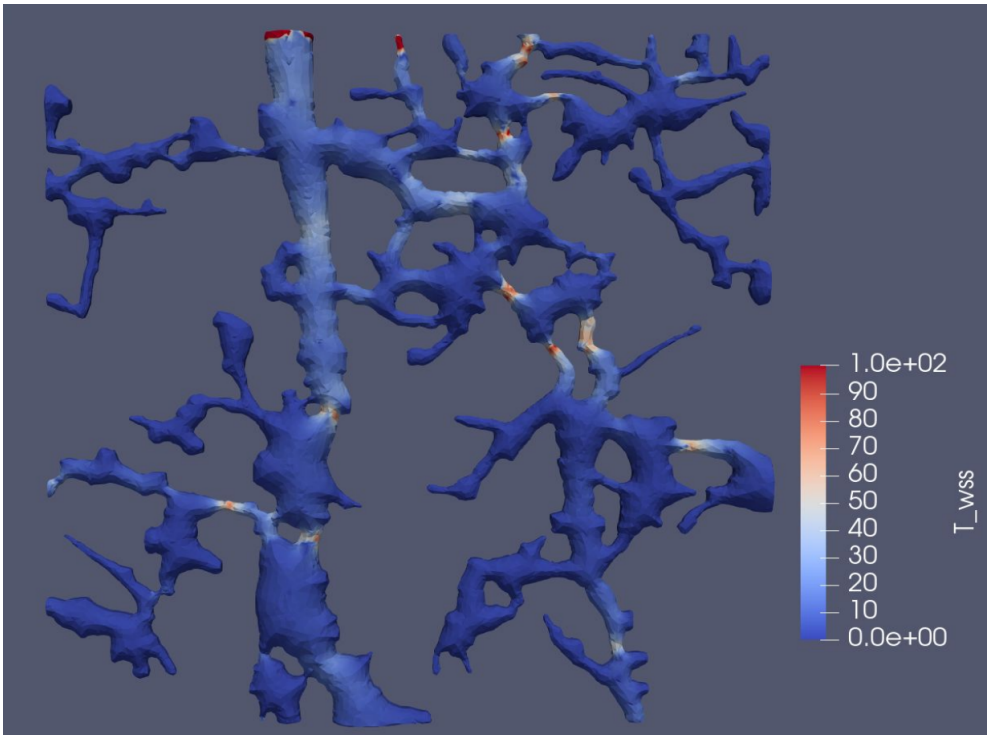


Figure 4.2 Pressure field and geometry for the example Stokes flow simulation (pressure in mPa).



(a) Velocity (μm/s))



(b) Wall Shear Stress (mPa)

Figure 4.3 Velocity and wall shear stress distributions.

Chapter Five

Preliminary Data Exploration

In this chapter, 2D graphs are generated for MVNs grown under varying conditions in order to relate MVN manufacturing variables to morphological changes in the networks.

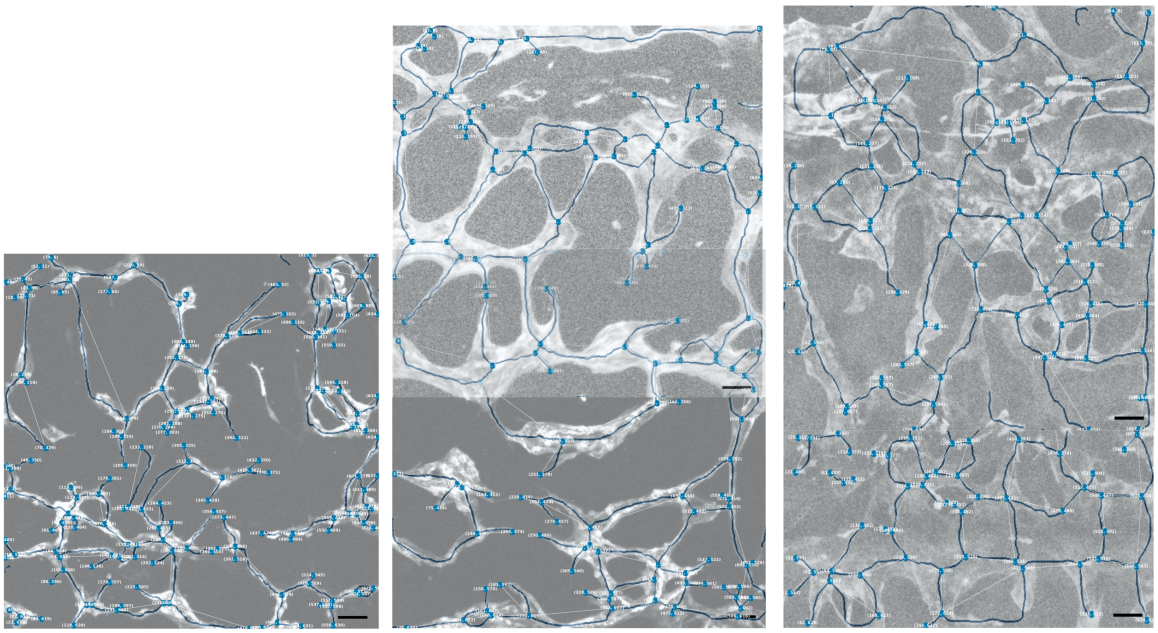
5.1 Nutrient Concentration

The Polacheck lab conducted an experiment to determine if low nutrient conditions influence vascular network morphology. The MVNs are traditionally grown using EGM2 media from Promocell with added nutrients and growth factors. We refer to this as "regular" media. In contrast, "basal" media has no added nutrients and growth factors.

For the experiment, three sets of MVNs were grown.

- 50.0% serum depleted media – cells cultured in 50.0% regular media + 50.0% basal media
- 25.0% serum depleted media – cells cultured in 25.0% regular media + 75.0% basal media
- 12.5% serum depleted media – cells cultured in 12.5% regular media + 87.5% basal media

For clarity, the 50.0% serum depleted media has the greatest concentration of nutrients and growth factors.



(a) 12.5% **(b)** 25.0% **(c)** 50.0%

Figure 5.1 2D graph generation for MVNs produced using serum at the labeled percentages.

5.1.1 Graph Generation

2D graphs were generated for the networks described in the previous section. Each allowed for disconnected regions of greater than 3 nodes, required edges to have minimum length of 3 μm, and required nodes to be separated by at least 15 μm. The resultant graphs are shown in Figure 5.1.

5.1.2 Graph Analysis

The the segment characteristics (described in Section 2.3.2) for the nutrient concentration experimental samples are summarized in Table 5.1. In addition to the raw values, a linear fit was applied to each characteristic (with respect to serum percentage). The R and R^2 correlation factors are also displayed in Table 5.1.

With the exception of directionality measures, all characteristics were at least weakly correlated ($|R| > 0.5$) to percent serum. Further, segment radii and connectivity were highly

Table 5.1 Summary of average segment characteristics for the MVNs created at varying nutrient concentrations.

Avg. Segment Characteristic	**12.5% Serum**	**25% Serum**	**25% Serum**	**50% Serum**	**50% Serum**	R	R^2
Length	91.0	145.0	135.6	138.8	133.7	0.583	0.340
Surface Area	9055	31035	23433	37302	33122	0.862	0.743
Volume	961223	739003	456894	984040	846587	0.892	0.796
Radius	18.7	29.5	31.6	47.8	44.6	0.978	0.975
Fractal Dimension	1.0251	1.0222	1.0203	1.0191	1.0205	-0.819	0.671
Contraction Factor	0.9042	0.9025	0.9128	0.9158	0.9124	0.753	0.568
Connectivity	1.1350	1.3002	1.0838	1.6582	1.6383	0.936	0.877
Unweighted Directionaliy	0.2667	0.2721	0.3504	0.2764	0.2895	-0.041	0.002
Weighted Directionality	0.2619	0.2888	0.366	0.2713	0.3004	-0.054	0.003

Table 5.2 Summary of volume based calculations for the 2.5D segmentations of the MVNs grown under varying nutrient concentrations.

Avg. Volume Characteristic	**12.5% Serum**	**25% Serum**	**25% Serum**	**50% Serum**	**50% Serum**	R	R^2
Vessel Density	0.1207	0.1994	0.1723	0.2862	0.2589	0.973	0.946
Avg. Distance from a Vessel Wall	38.8	50.5	47.3	38.2	46.1	-0.093	0.009

correlated ($|R| > 0.9$) to percent serum. Specifically, increases in nutrients increased vessel radii and average connectivity.

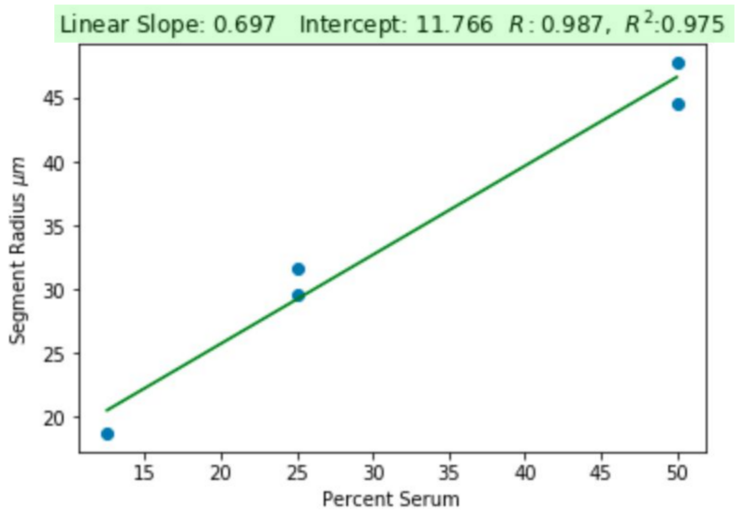
In particular, the correlation between vessel radii and percent serum was the strongest ($R = 0.978, R^2 = 0.975$). At such high correlation, it is potentially feasible to use serum percentage (on the range $[12.5\%, 50.0\%]$) to manufacture MVNs with desired average segment radii. This linear relationship is shown in Figure 5.2 along with the distribution of the individual segment radii.

As can been seen from the distribution in Figure 5.2 (b), the positive correlation between segment radii and percent serum is dominated by a shifting of the radii distribution rather than a reshaping. This leads to the hypothesis that with greater nutrient concentration, all vessels increase in radius (rather than simply pruning thin vessels).

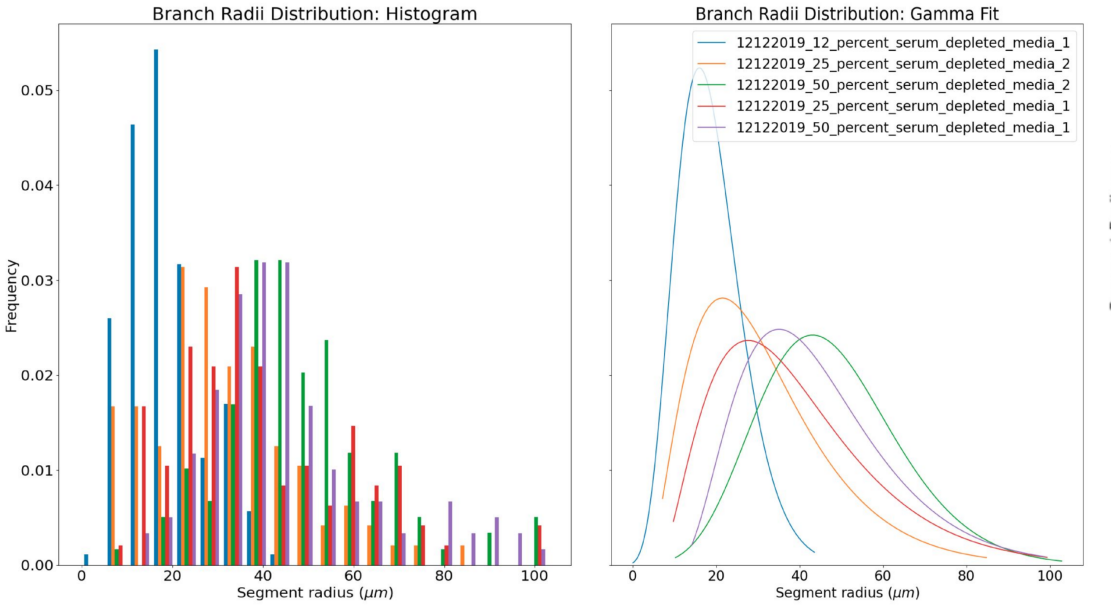
5.1.3 2.5D Volume Analysis

In addition to the graphs discussed, the 2D segmentations of the nutrient concentration samples were used to generate 2.5D volume segmentations. These segmentations were used to compare vessel density and average distance from a vessel wall. These results are summarized in Table 5.2.

Interestingly, vessel density is shown to be highly correlated to serum percentage ($R = 0.973, R^2 = 0.946$) while average distance from a vessel wall is shown to have no correlation



(a) Linear Fit



(b) Segment Distribution

Figure 5.2 Linear relationship between vessel segment radii and percentage serum (a) and distribution of vessel segment radii (b).

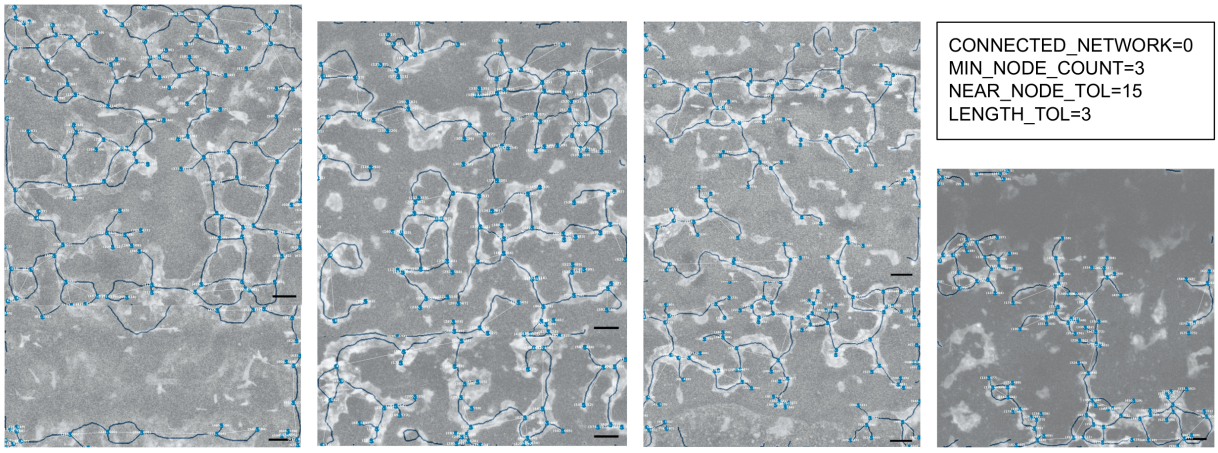
to serum percentage ($R = -0.093, R^2 = 0.009$). These results indicate that the volume of fluid (blood in physiological networks) and the surface area of cells within the MVNs decrease with decreases in nutrients. However, non-vascularized areas do not experience a decrease in access to the network. If the low nutrient networks are actually perfusable, they may represent an increase in vascular efficiency (i.e. the same volume of tissue is supported with lower HUVEC mass). However, this hypothesis will require future flow tests to further explore.

5.2 Static Storage

In addition to nutrient concentration, the Polacheck lab conducted an experiment to test storage conditions for the MVNs. Traditionally, the parent channels are connected to cell culture media reservoirs and the microfluidic devices are stored on a rocker. This "dynamic" storage creates a flow of media through the parent channels. In the following experiment, this storage technique (the "control") was compared against "static" storage of the MVNs (connecting the parent channels to the media reservoirs but not placing the devices on the rocker).

5.2.1 Graph Generation

2D graphs were generated for the networks described in the previous section. Each allowed for disconnected regions of greater than 3 nodes, required edges to have minimum length of 3 μm, and required nodes to be separated by at least 15 μm. Additionally, the parent channels were not included in the generated graphs. The resultant graphs are shown in Figure 5.3.



(a) Control **(b)** Static Storage

Figure 5.3 2D graph generation for MVNs stored in dynamic (a) and static (b) conditions. The empty section of (a) is the parent channel and was excluded from the network segmentation.

5.2.2 Graph Analysis

The segment characteristics for the control and static experimental samples are summarized in Table 5.3. In addition to the raw values, a two-sample Z-test was applied to each characteristic. For the Z-test, the null hypothesis was the average segment characteristics were equivalent. The alternative hypothesis was the the characterises were not equivalent. The Z-scores and probabilities, P, of the null hypothesis are also shown in Table 5.3.

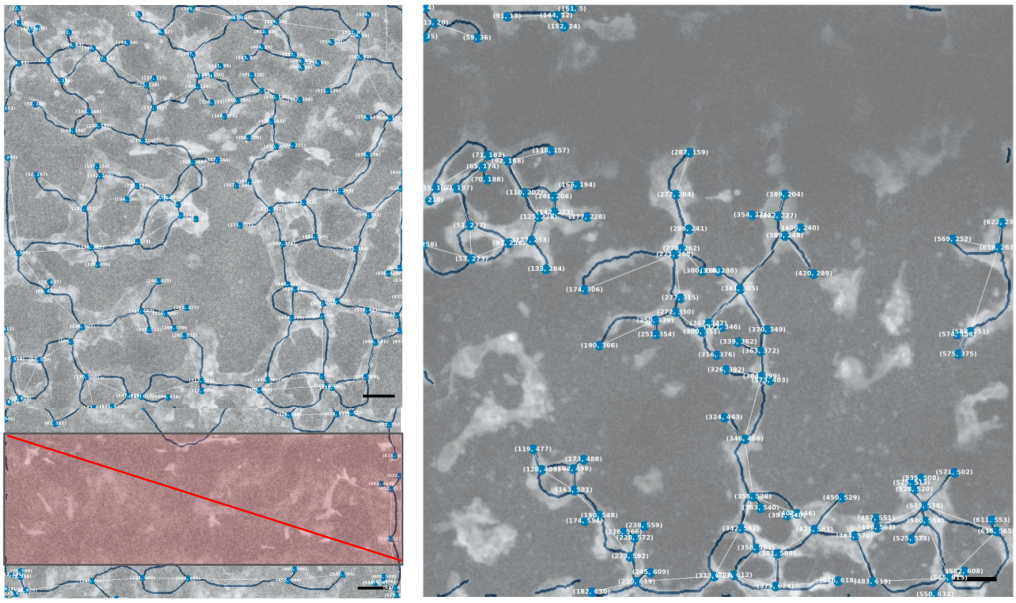
At the $\alpha = 0.10$ level, the differences in segment length, surface area, and volume were significant between the control and static MVNs. Further, at the $\alpha = 0.05$ level, the differences in the segment radius, contraction factor, and connectivity were significant between the control and static MNVs. Specifically, static storage conditions saw significant decreases in each of these characteristics.

5.2.3 2.5D Volume Analysis

As with the nutrient concentration experiment, the 2D segmentations of the storage experiment were used to create 2.5D volume segmentations. These segmentations were then

Table 5.3 Summary of the graph characteristics for dynamic and static storage conditions.

Average Segment Characteristic	Control	Static	Z	P
Length	116.6	101.0	1.656	0.0977
Surface Area	24218	15856	1.939	0.0525
Volume	543415	252100	1.856	0.0634
Radius	36.5	26.8	3.141	0.0017
Fractal Dimension	1.0229	1.0229	0.000	1.0000
Contraction Factor	0.9037	0.8872	2.3172	0.0205
Connectivity	1.2502	1.0404	2.5972	0.0094
Unweighted Directionality (Max)	0.29175	0.27862	0.8751	0.3815
Weighted Directionality (Max)	0.30085	0.27622	0.9731	0.3305



(a) Parent Channel Volume Exclusion

(b) Static Outlier

Figure 5.4 (a) Representation of the parent channel volume excluded from density and distance calculations (red). (b) The static condition outlier (poorly segmented).

used to determine average vessel density and distance from vessel wall characteristics for the control and static storage devices.

As computation notes, these calculations did not consider the volume occupied by the parent channel (as it is not representative of the network). Further, theses calculations were carried both with and without an outlier, which was poorly segmented. However, the conclusions reached are the same both with and without the outlier. The parent channel exclusion and the outlier are shown in Figure 5.4. A summary of the volume characteristics and corresponding Z-test values is shown in Table 5.4

Similar to the low nutrient MVNs, we observed with high certainty ($Z = 12.762, P < 0.00001$) that the MVNs stored in static conditions show a decrease in vessel density. Further like the low nutrient MVNs, we conclude no significant change ($Z = 0.1976, P = 0.8434$) in the static MVN's average distance from a vessel wall. Once again, this change indicates that the volume of fluid and the surface area of cells within the MVNs decrease in static storage,

Table 5.4 Summary of volume based calculations for the 2.5D segmentations of the MVNs grown under varying storage conditions.

Avg. Volume Characteristic	**Control**	**All Static**	**Z**	**P**
Vessel Density	0.2431	0.1449	6.3175	< 0.00001
Avg. Distance from Vessel Wall	38.9	47.38	0.9260	0.3544
Avg. Volume Characteristic	**Control**	**Removed Outlier Static**	**Z**	**P**
Vessel Density	0.2431	0.1596	12.762	< 0.00001
Avg. Distance from a Vessel Wall	38.9	39.956	0.1976	0.8434

however non-vascularized areas do not experience a decrease in access to the network.

5.3 Summary

Both low-nutrient and static networks displayed decreased vessel density with maintained average distance to a vessel wall. Additionally, for both low-nutrient and static networks, the percent decrease in segment length was less than the percent change in segment radius (perceived segment elongation), the segment contraction factor decreased (increased segment curvature), and connectivity decreased (rough increase in hierarchy). These changes could be consistent with the networks existing at different stages of development or expressing the same stage differently.

Chapter Six

Current and Future Work

6.1 3D Graph Representation

Representing the MVNs as NetworkX graphs is not restricted to two dimensions. In fact, a three dimensional representation is preferred as it more accurately reflects the true connectivity of the network. The 2D graph generation process can be converted to a 3D graph generation process simply by having the *SkeletonWalker* code search the immediate $3 \times 3 \times 3$ area. The only delay on this implementation is the creation of clean 3D skeletons to provide as inputs to the *SkeletonWalker*. The skeletonization process is sensitive to surface noise, which is far greater in the 3D segmentations than the 2D segmentations. I am currently working on 3D skeleton filtering procedures and should release the 3D graph generation to GitHub shortly.

6.2 Validation

While T-Junction and cross-method validation described in Chapter 3 provides general confidence in the segmentation techniques, more artificial data needs to be created to validate each sub process in the analysis pipeline. Further, the output characteristics should be compared against literature values in similar research.

6.3 Error Analysis

Beyond pure validation (obtaining results fitting expectations) I am especially interested in understanding how non-biologic factors impact the final results of calculations. For example, I would like to investigate differences in graph outputs when the same MVN is imaged at different magnifications and when different user parameters are used to generate the segmentations.

6.4 Blood-based Considerations

Initial flow experiments with the Polacheck MVNs plan to perfuse cell culture media (water). Therefore, the incompressible Stokes flow and Newtonian fluid-based wall shear stress calculations presented are valid. However, as a future consideration, blood at the microvascular scale does not behave as an incompressible, Newtonian fluid. Therefore, adjustments to the fluid modeling may be necessary.

The following sections take a brief look into the literature regarding modeling blood and WSS at the microvascular scale. As these are distant considerations for the project, this section (6.4) can be skipped without a loss of continuity.

6.4.1 Red Blood Cell and Non-Newtonian Effects

Blood primarily consists of a suspension of compressible red blood cells (RBCs) within an aqueous plasma. In large vessels (> 500 μm), such as arteries, blood behaves as a Newtonian fluid with constant viscosity [30]. However, at the microvascular scale, blood separates into two distinct, immiscible fluid layers as described by the Farheus-Lindqvist effect [31]. RBCs concentrate at the center of the vessel and exhibit compressible, shear thinning, non-Newtonian characteristics while a cell-free layer (CFL) of Newtonian plasma remains in contact with the vessel wall. [30]. Current fluid models capable of calculating

WSS approach this flow dynamic by either modeling RBCs as discrete bodies or by applying non-Newtonian corrections to the viscosity values based on vessel diameters and shear rates.

Discrete RBC Modeling and WSS

Balogh & Bachi (2019) provides the most advanced *in silico* model of realistic, RBC resolved microvasculature flow [32]. Using the model, the paper investigated the effects of the RBCs on WSS by running the computations both with RCBs present (RBC flow) and without RBCs present (plasma flow). Importantly, the non-Newtonian effects of the blood were captured by the inclusion of RBCs, so the plasma flow represented incompressible Newtonian flow.

When the model used pressure based boundary conditions to compare RBC flow and plasma flow (such as in the example in Section 4.3), Balogh & Bachi observed differences in fluid velocities throughout the network. Therefore, changes in WSS were due to changes in velocity of the fluid and not directly relatable to the RBCs. However, bounding the model by flow rate yielded WSS of the RBC flow up to three times greater in magnitude than for the plasma flow.

More specifically, under flow-rate boundary conditions, the paper found RBCs increased WSS most drastically in venules (blood vessels of diameter $\sim 10 - 100$ μm) and on the side of a vessel where WSS was originally lower in plasma flow. Further, when considering the spatial gradients of WSS, the circumferential component was greater than the axial component in the RBC flow and the reverse was observed in plasma flow. Finally, the RBC flows created fluctuations in WSS with time while plasma flows did not.

These conclusions are important for future versions of the project on two fronts. First, the Balogh & Bachi paper demonstrated RBCs impact flow velocities in microvascular networks. The conclusion is further supported in Schmid *et al.* (2019) which provided the mechanism for the velocity discrepancies by demonstrating RBCs support equating outflow velocities at divergent capillary bifurcations [33]. Therefore, the pressure bound models currently used for the networks would yield incorrect fluid velocities and therefore incorrect WSS. However

the paper also showed bounding the model by fluid velocity would still predict incorrect magnitudes and spatial/temporal distributions of the WSS if RBCs are not included in the model.

Taken together, the experiments of Balogh & Bachi appear to advocate modeling RBCs for future versions of the Polacheck microvascular networks, especially considering the majority of vessels fall in the venule size classification (which feel the greatest effects of RBCs). However, modeling RBCs is more computationally expensive than modeling cell-free fluid.

6.4.2 Continuum Model with Non-Newtonian Corrections

The other advanced approach to modeling the non-Newtonian and Farheus-Lindqvist aspects of blood is to assume a phase-separated, incompressible fluid. The inner layer represents the concentration of RBCs and is subject to non-Newtonian flow, where the effective viscosity is calculated by constitute equations, and the outer fluid behaves Newtonianly and represents the cell-free plasma layer [34]. Specifically, Marcinkowska *et al.* (2007) supports the use of the Quemada model to estimate the effective viscosity for the RCB core [35].

Using the Quemada model and the approach described above, Sriram *et al.* (2014) shows modeling microvascular blood flow as a mixture of non-Newtonian and Newtonian fluids accurately reproduced experimental observations of fluid velocities within the RBC core and CFL. Further, the paper showed models that assumed Newtonian cores or homogeneous fluids yielded lower correlations with the experimental data and would significantly underestimate WSS [34].

The ability to use a continuum model would provide computational time advantages over a RBC tracking model. However, the modeling method of Sriram *et al.* does not directly translate to the Polacheck microvascular networks. Sriram *et al.* validated their model with the results of experiments of blood flow in straight glass capillaries reported in Long *et al.* (2004) [36]. These experiments do not account for bifurcations which, as discussed in

relation to Schmid *et al.* (2019), alter fluid velocities in the presence of RBCs. Therefore, it is unknown how the model would react to the highly curved and bifurcated Polacheck networks.

6.4.3 *In Vivo* Relevance

The previous sections provided model based conclusions linking RBCs and non-Newtonian flow to changes in WSS magnitude and location. However, for clinical applications and relevance to angiogenesis, the effects of these factors need to be observed on the biological scale. Thus, Xu *et al.* (2017) performed an *in vivo* experiment to determine the importance of RBC presence in microvascular flow by applying both plasma only (no cells) and blood (plasma with RBCs) flow to rat venules [37] (similar to the experiment conducted *in silico* in the Balogh & Bachi paper).

Notably, WSS increased nitric oxide (NO) signalling in the endothelial cells for both the plasma flow and RBC flow. However, the RBC flow (and not the plasma flow) also increased calcium ion (Ca^{2+}) concentrations in the endothelial cells. Mechanistically, the Ca^{2+} observations were triggered by interactions with shear stress induced releases of ATP from the RBCs, and therefore, will only be observed in the RBC flow [37].

Importantly, NO is associated with angiogenesis, so the result showed that both plasma and RBC flows are potentially capable of triggering angiogenesis through WSS [38]. However, the association may be caused by NO's interaction with VEGF, so controlling for VEGF in the Polacheck networks could potentially eliminate this effect [39]. Further though, NO is a vasodilator and Ca^{2+} increases gap formation and permeability of the endothelium [37, 40]. Therefore, both flow conditions change physical constraints to a computational model. The nitric oxide creates topological changes the the vessels for plasma flow and RBC flow, while the calcium ions alter vessel properties for the RBC flow.

6.4.4 Overall Impacts

As stated, the initial experiments with the Polacheck microvascular networks plan to flow media rather than blood. Therefore, the current use of the incompressible Stokes flow solver accurately represents the *in vitro* experiments and the RBC discussions are only relevant to future experiments.

However, the discussions of this section demonstrate that the media flow experiment may not reflect *in vivo* dynamics, which could affect clinical applicability and experiments with blood may require changes to the model for continued accuracy. Immediate adjustments could use the viscosity of plasma to calculate WSS (respecting the CFL) or could use the three-dimensional distance transform to adjust viscosity values use based on vessel diameter (to respect the effective viscosity). Furhter, micro particle image velocimetry (μPIV) of could be used to obtain velocity boundary conditions (rather than pressure). Distant adjustments could include discrete RBC tracking or modeling a multi-phase fluid. Ultimately, each adjustment will increase the complexity of the computations and needs to be justified experimentally.

REFERENCES

[1] C Alberto Figueroa, Charles A Taylor, and Alison L Marsden. "Blood Flow". In: *Encyclopedia of Computational Mechanics Second Edition* (2017), pp. 1–31.

[2] Alfons JHM Houben, Remy JH Martens, and Coen DA Stehouwer. "Assessing microvascular function in humans from a chronic disease perspective". In: *Journal of the American Society of Nephrology* 28.12 (2017), pp. 3461–3472.

[3] Mark A Skylar-Scott et al. "Biomanufacturing of organ-specific tissues with high cellular density and embedded vascular channels". In: *Science advances* 5.9 (2019), eaaw2459.

[4] Thomas H Adair and Jean-Pierre Montani. "Angiogenesis". In: *Colloquium series on integrated systems physiology: from molecule to function*. Vol. 2. 1. Morgan & Claypool Life Sciences. 2010, pp. 1–84.

[5] Y Ganat et al. "Chronic hypoxia up-regulates fibroblast growth factor ligands in the perinatal brain and induces fibroblast growth factor-responsive radial glial cells in the sub-ependymal zone". In: *Neuroscience* 112.4 (2002), pp. 977–991.

[6] Brian I Rini. "Vascular endothelial growth factor–targeted therapy in renal cell carcinoma: current status and future directions". In: *Clinical Cancer Research* 13.4 (2007), pp. 1098–1106.

[7] Peter A Galie et al. "Fluid shear stress threshold regulates angiogenic sprouting". In: *Proceedings of the National Academy of Sciences* 111.22 (2014), pp. 7968–7973.

[8] Siavash Ghaffari, Richard L Leask, and Elizabeth AV Jones. "Flow dynamics control the location of sprouting and direct elongation during developmental angiogenesis". In: *Development* 142.23 (2015), pp. 4151–4157.

[9] William J Polacheck et al. "A non-canonical Notch complex regulates adherens junctions and vascular barrier function". In: *Nature* 552.7684 (2017), p. 258.

[10] Guillaume Chouinard-Pelletier, Espen D Jahnsen, and Elizabeth AV Jones. "Increased shear stress inhibits angiogenesis in veins and not arteries during vascular development". In: *Angiogenesis* 16.1 (2013), pp. 71–83.

[11] József Tımár et al. "Angiogenesis-dependent diseases and angiogenesis therapy". In: *Pathology Oncology Research* 7.2 (2001), pp. 85–94.

[12] William J Polacheck et al. "Microfabricated blood vessels for modeling the vascular transport barrier". In: *Nature Protocols* 14.5 (2019), p. 1425.

[13] Pol Kennel et al. "Toward quantitative three-dimensional microvascular networks segmentation with multiview light-sheet fluorescence microscopy". In: *Journal of biomedical optics* 23.8 (2018), p. 086002.

[14] Kerry J Howe et al. *Principles of water treatment*. John Wiley & Sons, 2012.

[15] KP Ivanov, MK Kalinina, and Yu I Levkovich. "Microcirculation velocity changes under hypoxia in brain, muscles, liver, and their physiological significance". In: *Microvascular research* 30.1 (1985), pp. 10–18.

[16] Brian J Kirby. *Stokes Flow*. URL: http://www.kirbyresearch.com/index.cfm/wrap/textbook/microfluidicsnanofluidicsch8.html#x53-1690008.

[17] Chet Miller. *Wall Shear Stress*. Jan. 2016. URL: https://physics.stackexchange.com/questions/250227/wall-shear-stress.

[18] Nathaniel G dela Paz et al. "Role of shear-stress-induced VEGF expression in endothelial cell survival". In: *J Cell Sci* 125.4 (2012), pp. 831–843.

[19] Stuart Egginton et al. "Shear stress-induced angiogenesis in mouse muscle is independent of the vasodilator mechanism and quickly reversible". In: *Acta Physiologica* 218.3 (2016), pp. 153–166.

[20] Stéfan van der Walt et al. "scikit-image: image processing in Python". In: *PeerJ* 2 (June 2014), e453. ISSN: 2167-8359. DOI: 10.7717/peerj.453. URL: https://doi.org/10.7717/peerj.453.

[21] Pauli Virtanen et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17 (2020), pp. 261–272. DOI: https://doi.org/10.1038/s41592-019-0686-2.

[22] Chun Hung Li and CK Lee. "Minimum cross entropy thresholding". In: *Pattern recognition* 26.4 (1993), pp. 617–625.

[23] CH Li and Peter Kwong-Shun Tam. "An iterative algorithm for minimum cross entropy thresholding". In: *Pattern recognition letters* 19.8 (1998), pp. 771–776.

[24] Leo Grady. "Random walks for image segmentation". In: *IEEE transactions on pattern analysis and machine intelligence* 28.11 (2006), pp. 1768–1783.

[25] Jaakko Sauvola and Matti Pietikäinen. "Adaptive document image binarization". In: *Pattern recognition* 33.2 (2000), pp. 225–236.

[26] TY Zhang and Ching Y. Suen. "A fast parallel algorithm for thinning digital patterns". In: *Communications of the ACM* 27.3 (1984), pp. 236–239.

[27] Aric Hagberg, Pieter Swart, and Daniel S Chult. *Exploring network structure, dynamics, and function using NetworkX*. Tech. rep. Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

[28] Tilmann Gneiting, Hana Ševčiková, and Donald B Percival. "Estimators of fractal dimension: Assessing the roughness of time series and spatial data". In: *Statistical Science* (2012), pp. 247–277.

[29] Yixin Hu et al. "Tetrahedral meshing in the wild." In: *ACM Trans. Graph.* 37.4 (2018), pp. 60–1.

[30] Prosenjit Bagchi. "Mesoscale simulation of blood flow in small vessels". In: *Biophysical journal* 92.6 (2007), pp. 1858–1877.

[31] Robin Fahraeus and Torsten Lindqvist. "The viscosity of the blood in narrow capillary tubes". In: *American Journal of Physiology-Legacy Content* 96.3 (1931), pp. 562–568.

[32] Peter Balogh and Prosenjit Bagchi. "Three-dimensional distribution of wall shear stress and its gradient in red cell-resolved computational modeling of blood flow in in vivo-like microvascular networks". In: *Physiological reports* 7.9 (2019).

[33] Franca Schmid et al. "Red blood cells stabilize flow in brain microvascular networks". In: *PLoS computational biology* 15.8 (2019), e1007231.

[34] Krishna Sriram, Marcos Intaglietta, and Daniel M Tartakovsky. "Non-Newtonian flow of blood in arterioles: consequences for wall shear stress measurements". In: *Microcirculation* 21.7 (2014), pp. 628–639.

[35] Anna Marcinkowska-Gapińska et al. "Comparison of three rheological models of shear flow behavior studied on blood samples from post-infarction patients". In: *Medical & biological engineering & computing* 45.9 (2007), pp. 837–844.

[36] David S Long et al. "Microviscometry reveals reduced blood viscosity and altered shear rate and shear stress profiles in microvessels after hemodilution". In: *Proceedings of the National Academy of Sciences* 101.27 (2004), pp. 10060–10065.

[37] Sulei Xu et al. "New insights into shear stress-induced endothelial signalling and barrier function: cell-free fluid versus blood flow". In: *Cardiovascular research* 113.5 (2017), pp. 508–518.

[38] Lucia Morbidelli, Sandra Donnini, and Marina Ziche. "Role of nitric oxide in the modulation of angiogenesis". In: *Current pharmaceutical design* 9.7 (2003), pp. 521–530.

[39] Dai Fukumura et al. “Predominant role of endothelial nitric oxide synthase in vascular endothelial growth factor-induced angiogenesis and vascular permeability”. In: *Proceedings of the National Academy of Sciences* 98.5 (2001), pp. 2604–2609.

[40] Andrew Maiorana et al. “Exercise and the nitric oxide vasodilator system”. In: *Sports Medicine* 33.14 (2003), pp. 1013–1035.

APPENDIX

Appendix A

Input Files for Sample Outputs

A.1 Sample A

input.txt

```
# MVN Image Analysis Input File Sample A
===============
Required Fields
===============
# Input image and scale values um / pixel
TIF_FILE=/home/ryan/Desktop/mvn-analysis/data/original_sample.tif
SCALE_X=1.2420
SCALE_Y=1.2420
SCALE_Z=1.1400
#### Piplelines to Run ####
SEGMENT_2D=1
MESH_25D=1
SEGMENT_3D=1
MESH_3D=1
SKEL_3D=0
VOLUME_ANALYSIS=1
NETWORK_2D_GEN=1
NETWORK_2D_COMPARE=0
```

```
NETWORK_3D_GEN=0
NETWORK_3D_COMPARE=0
#### Save Parameters #####
OUTPUT_DIR=/home/ryan/Desktop/mvn-analysis/outputs/
SAVE_2D_MASK=1
SAVE_2D_SKEL=1
SAVE_2D_DIST=1
SAVE_2D_DISPLAY=1
SAVE_2D_REVIEW=1
SAVE_25D_MESH=1
SAVE_25D_MASK=1
GENERATE_25D_VOLUME=1
SAVE_3D_MASK=1
SAVE_3D_SKEL=1
SAVE_3D_MESH=1
GENERATE_3D_VOLUME=1
SAVE_3D_MESH_ROUND=1
GENERATE_3D_ROUND_VOLUME=1
SAVE_2D_NETWORK=1
SAVE_3D_NETWORK=0
#### Display Parameters #####
PLOT_ALL_2D=0
REVIEW_PLOT_2D=1
PLOT_25D_MESH=1
PLOT_3D_THRESH_SLICES=1
PLOT_LUMEN_FILL=1
PLOT_3D_MESHES=1
PLOT_3D_SKELS=0
```

```
PLOT_NETWORK_GEN=1
PLOT_NETWORK_DATA=0
====================
Adjustable Parameters
====================
#### 2D Analysis #####
# original, rescale, equalize, adaptive
CONTRAST_METHOD=rescale
# entropy, random-walk
THRESH_METHOD=entropy
RWALK_THRESH_LOW=0.08
RWALK_THRESH_HIGH=0.18
BTH_K_2D=3
WTH_K_2D=3
DILA_GAUSS=0.333
OPEN_FIRST=0
OPEN_K_2D=0
CLOSE_K_2D=5
CONNECTED_2D=1
SMOOTH=1
#### 25D Analysis ####
H_PCT_25D=-1
CONNECTED_25D_MESH=1
CONNECTED_25D_VOLUME=0
#### GRAPH Analysis #####
CONNECTED_NETWORK=0
MIN_NODE_COUNT=3
NEAR_NODE_TOL=15
```

```
LENGTH_TOL=3
#### 3D Analysis #####
# Output parameters
CONNECTED_3D_MASK=1
CONNECTED_3D_MESH=1
CONNECTED_3D_SKEL=1
### Thresholding parameters
# original, rescale, equalize, adaptive
SLICE_CONTRAST=original
# sauvola, none
PRE_THRESH=sauvola
WINDOW_SIZE_X=15
WINDOW_SIZE_Y=15
WINDOW_SIZE_Z=7
BTH_K_3D=3
WTH_K_3D=3
CLOSE_K_3D=1
### Lumen filling parameters
# octants, pairs, ball
ELLIPSOID_METHOD=octants
MAXR=15
MINR=1
H_PCT_R=0.5
THRESH_PCT=0.15
THRESH_NUM_OCT=5
THRESH_NUM_OCT_OP=3
MAX_ITERS=1
# uniform, random
```

```
RAY_TRACE_MODE=uniform

# sweep, xy, exclude_z_pole

THETA=exclude_z_pole

N_THETA=6

N_PHI=6

MAX_ESCAPE=3

PATH_L=1

FILL_LUMEN_MESHING=1

FILL_LUMEN_MESHING_MAX_ITS=3

ENFORCE_ELLIPSOID_LUMEN=1

H_PCT_ELLIPSOID=0.5

### Skeletonizaiton Parameters

SQUEEZE_SKEL_BLOBS=1

REMOVE_SKEL_SURF=1

SKEL_SURF_TOL=5

SKEL_CLOSING=1
```

A.2 Sample B

input.txt

```
# MVN Image Analysis Input File Sample B
===============
Required Fields
===============
# Input image and scale values um / pixel
TIF_FILE=/home/ryan/Desktop/mvn-analysis/data/Shared/09092019_gel_region_10x.tif
SCALE_X=2.4859
SCALE_Y=2.4859
SCALE_Z=6.00
#### Piplelines to Run ####
SEGMENT_2D=1
MESH_25D=1
SEGMENT_3D=1
MESH_3D=1
SKEL_3D=0
VOLUME_ANALYSIS=1
NETWORK_2D_GEN=1
NETWORK_2D_COMPARE=0
NETWORK_3D_GEN=0
NETWORK_3D_COMPARE=0
#### Save Parameters #####
OUTPUT_DIR=/home/ryan/Desktop/mvn-analysis/outputs/
SAVE_2D_MASK=1
SAVE_2D_SKEL=1
SAVE_2D_DIST=1
```

```
SAVE_2D_DISPLAY=1
SAVE_2D_REVIEW=1
SAVE_25D_MESH=1
SAVE_25D_MASK=1
GENERATE_25D_VOLUME=1
SAVE_3D_MASK=1
SAVE_3D_SKEL=0
SAVE_3D_MESH=1
GENERATE_3D_VOLUME=1
SAVE_3D_MESH_ROUND=1
GENERATE_3D_ROUND_VOLUME=1
SAVE_2D_NETWORK=1
SAVE_3D_NETWORK=0
#### Display Parameters #####
PLOT_ALL_2D=0
REVIEW_PLOT_2D=1
PLOT_25D_MESH=1
PLOT_3D_THRESH_SLICES=1
PLOT_LUMEN_FILL=1
PLOT_3D_MESHES=1
PLOT_3D_SKELS=0
PLOT_NETWORK_GEN=1
PLOT_NETWORK_DATA=0
=====================
Adjustable Parameters
=====================
#### 2D Analysis #####
# original, rescale, equalize, adaptive
```

```
CONTRAST_METHOD=equalize
# entropy, random-walk
THRESH_METHOD=random-walk
RWALK_THRESH_LOW=0.45
RWALK_THRESH_HIGH=0.48
BTH_K_2D=3
WTH_K_2D=1
DILA_GAUSS=0.0
OPEN_FIRST=0
OPEN_K_2D=0
CLOSE_K_2D=0
CONNECTED_2D=1
SMOOTH=2
#### 25D Analysis ####
H_PCT_25D=-1
CONNECTED_25D_MESH=1
CONNECTED_25D_VOLUME=0
#### GRAPH Analysis #####
CONNECTED_NETWORK=0
MIN_NODE_COUNT=3
NEAR_NODE_TOL=15
LENGTH_TOL=3
#### 3D Analysis #####
# Output parameters
CONNECTED_3D_MASK=1
CONNECTED_3D_MESH=1
CONNECTED_3D_SKEL=1
### Thresholding parameters
```

```
# original, rescale, equalize, adaptive
SLICE_CONTRAST=original
# sauvola, none
PRE_THRESH=sauvola
WINDOW_SIZE_X=13
WINDOW_SIZE_Y=13
WINDOW_SIZE_Z=9
BTH_K_3D=1
WTH_K_3D=1
CLOSE_K_3D=1
### Lumen filling parameters
# octants, pairs, ball
ELLIPSOID_METHOD=octants
MAXR=15
MINR=1
H_PCT_R=0.5
THRESH_PCT=0.15
THRESH_NUM_OCT=5
THRESH_NUM_OCT_OP=3
MAX_ITERS=1
# uniform, random
RAY_TRACE_MODE=uniform
# sweep, xy, exclude_z_pole
THETA=exclude_z_pole
N_THETA=6
N_PHI=6
MAX_ESCAPE=1
PATH_L=1
```

```
FILL_LUMEN_MESHING=1

FILL_LUMEN_MESHING_MAX_ITS=3

ENFORCE_ELLIPSOID_LUMEN=1

H_PCT_ELLIPSOID=0.5

### Skeletonizaiton Parameters

SQUEEZE_SKEL_BLOBS=1

REMOVE_SKEL_SURF=1

SKEL_SURF_TOL=5

SKEL_CLOSING=1
```
