MARK FRICKER

PHYSARUM NETWORK ANALYSIS

PLANT SCIENCES, OXFORD
FIRST EDITION

PUBLISHED BY PLANT SCIENCES, OXFORD
FIRST EDITION

First printing, March 2017

Current version, March 2017

Contents

Acknowledgements

This work was supported by a Visiting Fellowship at The Institute of Advanced Studies in Durham, The Leverhulme Trust (RPG-2015-437) and The Human Frontier Science Program (RGP0053/2012). My thanks to many colleagues who have been involved in network research on slime moulds that have contributed experimental resources during development of this software.

The *Physarum* network analysis package was based on the phase congruency analysis for fungal networks developed with Boguslaw Obara, University of Durham, available at:
`http://community.dur.ac.uk/boguslaw.obara/research/software/`

This package also uses the following matlab implementations:

Bioformats package for file import:
`http://www.openmicroscopy.org/site/products/bio-formats`

The phase congruency and skeleton extraction by Peter Kovesi, University of Western Australia:
`http://www.peterkovesi.com/matlabfns/`

The anisotropic second-order Gaussian kernels by Carlos Lopez-Molina, Universidad Pública de Navarra, Pamplona:
`http://www.kermit.ugent.be/software.php?navigatieId=0&categorieId=17`

The Frangi 'Vesselness' filter by Marc Schrijver and D. Kroon, University of Twente:
`https://uk.mathworks.com/matlabcentral/fileexchange/24409-hessian-based-frangi-vesselness-filter`

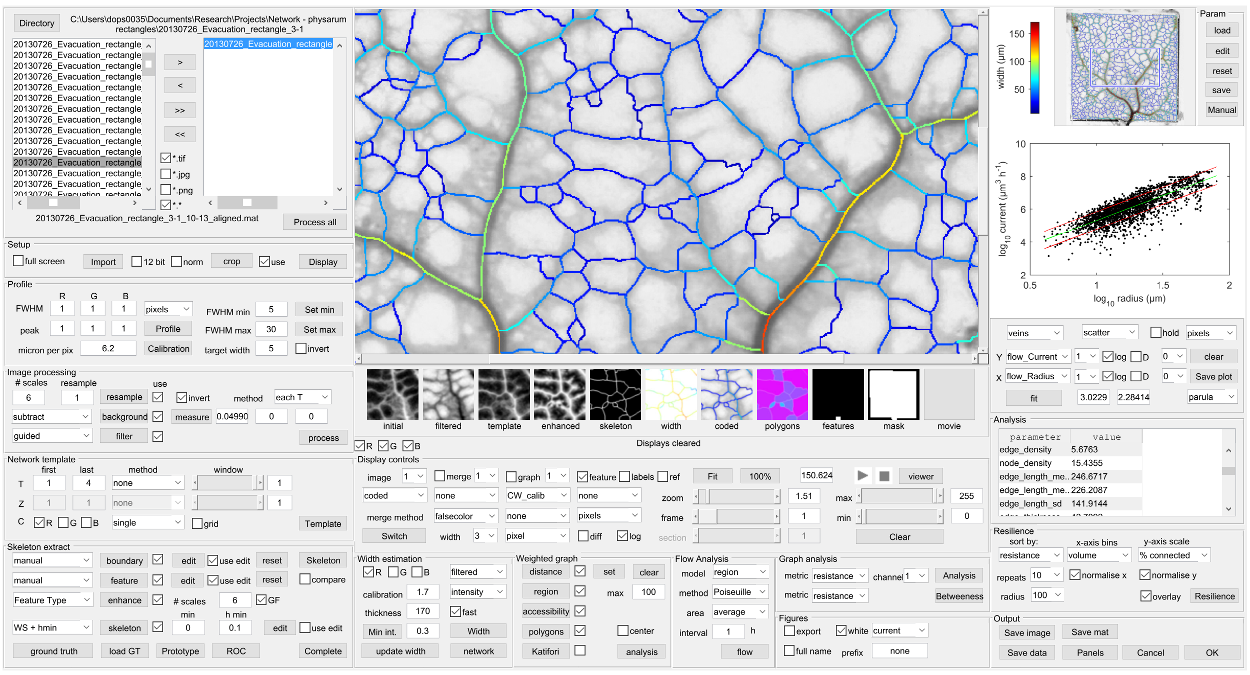
The export fig package by Oliver Woodford and Yair Altman:
`https://uk.mathworks.com/matlabcentral/fileexchange/23629-export-fig`

The ImageJ implementation of the Steger algorithm by Thorsten Wagner, University of Applied Sciences and Arts Dortmund, with a Matlab wrapper by Phil Jackson, University of Durham:
`https://github.com/thorstenwagner/ij-ridgedetection/tree/master/src/main/java/de/biomedical_imaging/ij/steger`

The Matlab implementation of the Boost Graph Library by David Gleich:
`https://uk.mathworks.com/matlabcentral/fileexchange/10922-matlabbgl`

The Bresenham line drawing algorithm by Aaron Wetzler, Haifa Technion, Israel:
`https://uk.mathworks.com/matlabcentral/fileexchange/28190-bresenham-optimized-for-matlab`

1
Introduction



Analysis

parameter	value
edge_density	5.6763
node_density	15.4355
edge_length_me..	246.6717
edge_length_me..	226.2087
edge_length_sd	141.9144

Figure 1.1: The GUI interface for the Physarum network analysis program

1.1 Overview

This manual describes a high-throughput automated image analysis system to detect and characterize large complex biological networks (Fig. 7.1). It was originally developed to analyse macroscopic networks such as fungal mycelia[1], but the same approaches can be applied to any planar 2-D curvi-linear structure, including *Physarum* vein networks As most biological networks have filaments, tubes or vessels that span a range of scales, with differing contrast and noise levels depending on the imaging method, we provide a number of different curvilinear feature enhancement methods, segmentation and skeletonisation algorithms to extract the network structure as a single-pixel wide binary skeleton. The skeleton is then converted to a graph representation, with nodes at the junctions or branch points, linked by edges that capture the

[1] B. Obara, V. Grau, and M. D. Fricker. A bioimage informatics approach to automatically extract complex fungal networks. *Bioinformatics*, 28:2374–81, 2012

curvi-linear structures. Once in a graph format, a wide range of graph theoretic measures can be calculated, or the graph can be used as the input to mathematical models that calculate flows on the network[2,3]

[2] L. L. M Heaton, E López, P. K Maini, M. D Fricker, and N. S Jones. Growth-induced mass flows in fungal networks. *Proc. R. Soc. B*, 277:3265–3274, 2010

[3] L. L. Heaton, E. Lopez, P. K. Maini, M. D. Fricker, and N. S. Jones. Advection, diffusion, and delivery over a network. *Phys Rev E*, 86:021905, 2012

1.2 *The network extraction work flow*

A flow diagram of the complete network extraction and analysis workflow is given in Fig. 1.2. An outline of each of the main sections is given below. A detailed explanation of the interface, algorithms and parameters used is given in the subsequent Chapters.

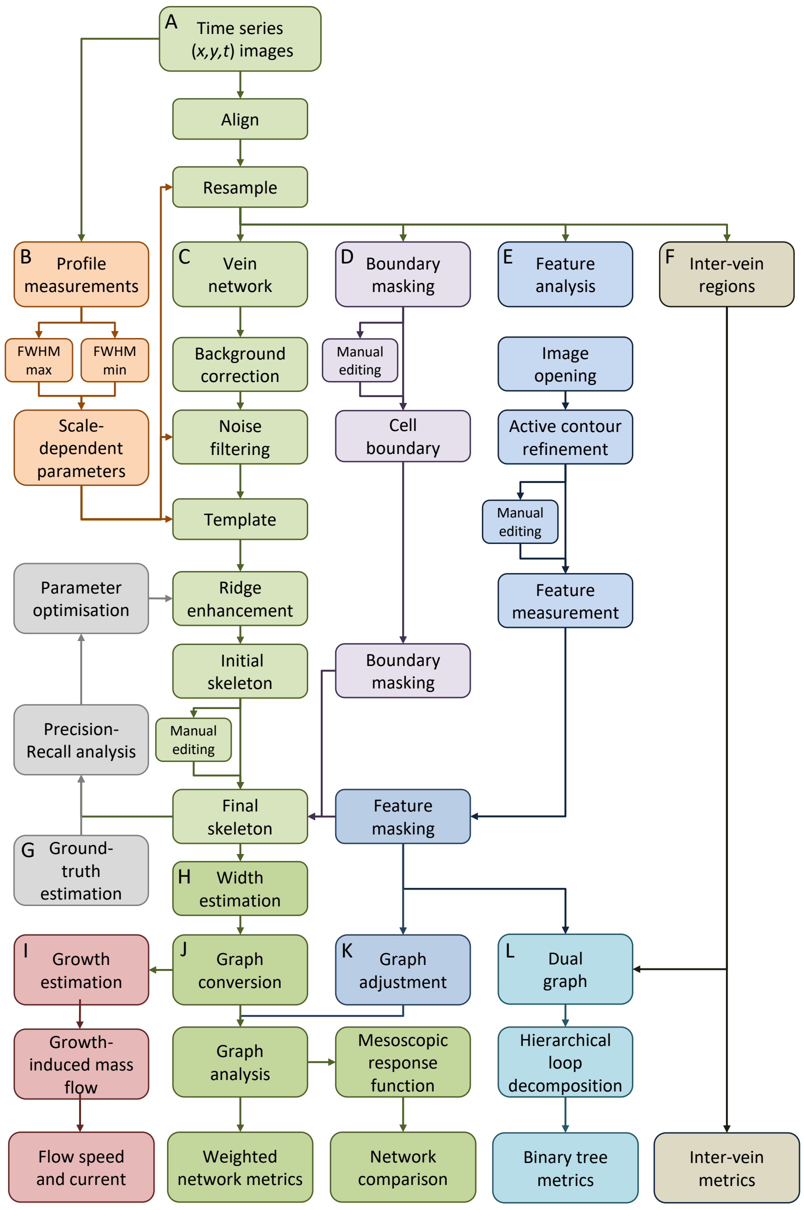


Figure 1.2: Flow diagram of the main steps in automated extraction and analysis of Physarum networks

1.2.1 *Image import*

A typical processing pipeline (Fig. 1.2A) starts with a single image or time-series during network formation (Fig. 1.3). A wide range of images can loaded directly into the program using the bioformats package, or using the import GUI (see Chapter 9), which has facilities to crop, filter and align images by cross-correlation.

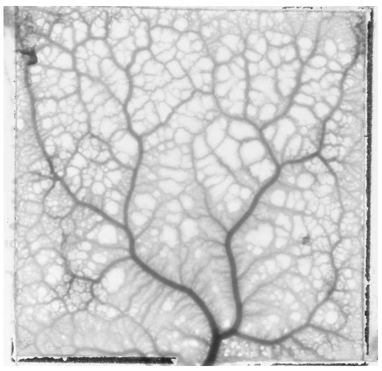


Figure 1.3: Typical image of a developing network in Physarum

1.2.2 *Setting the scale*

The minimum vein size is estimated as the full-width half-maximum (FWHM) from a user-defined profile drawn across the image (Fig. 1.2B). The value of the FWHM is then used to resample the image to ensure that the minimum vein size is consistent (typically 5 pixels wide), and matched to subsequent filtering operation.

1.2.3 *Extracting the vein network*

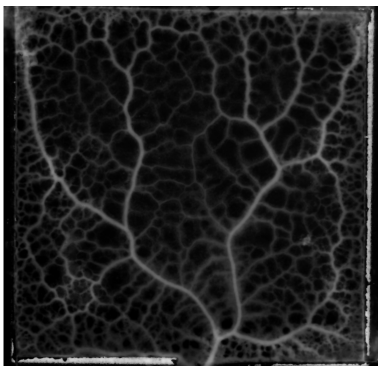


Figure 1.4: Template image after filtering and background subtraction Physarum

Images are inverted to give a white-on-black image, background corrected, and smoothed with a guided filter, using a 5×5 kernel size (Fig. 1.2C). The median intensity over time can be used to construct a single a template image, or each image in the time-series can be used as its own template.

The simplest method to identify the veins automatically is intensity-based segmentation of a bright-field transmission image to give a binary image, with ones representing the vein structure and zeros for the background. Typically this is followed by thinning of the binary image to give a single-pixel wide skeleton. However, the initial segmentation is critically dependent on the value for the threshold used, and it is rare that a single threshold provides adequate segmentation without either losing thinner, dimmer venules if it is set too high, or artificially expanding and fusing adjacent regions if it is set too low.

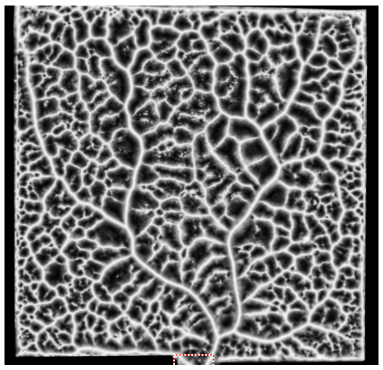


Figure 1.5: Ridge enhancement using the 'Feature Type' output from the intensity-independent phase congruency algorithm

This is particularly problematic in developing networks of *Physarum* as the veins vary in size and absorbance over an order of magnitude, and the interstitial areas between the veins also contain plasmodial sheet-like regions that are biologically significant, but not normally included as part of the network, as the intensity-based segmentation algorithms fail in these regions. Local contrast enhancement, adaptive or hysteresis thresholding may improve the initial segmentation. However, these approaches do not deal well with developmental time-series which contain a range of vein diameters, growing margins, and intervening plasmodial sheets. Thus, a number of ridge enhancement methods are provided that are typically applied over a range of scales and orientations to selectively highlight curvi-linear features (Fig. 1.5).

These include:

- 'Vesselness' based on the classic Frangi algorithm[4], that uses the eigenvalues and eigenvectors of the image Hessian matrix to highlight blood vessels.

[4] A. F. Frangi, W. J. Niessen, K.L. Vincken, and M.A. Viergever. *Multiscale vessel enhancement filtering*, pages 130–137. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998

- 'Neuriteness' based on the modified Hessian proposed by Meijering *et al.* (2004)[5] for neuronal tracing.
- Second-order anisotropic Gaussian kernels (SOAGK) developed by Lopez-Molina et al. (2015)[6] to enhanced fungal mycelial network images.
- The 'Steger' curvilinear detector proposed by Steger (1998)[7] for analysis of road networks from aerial photos.
- Phase-congruency, including the 'Feature-Type' image, developed by Kovesi (1999)[8] as a general, intensity-independent feature detector.

[5] E. Meijering, M. Jacob, J. Sarria, P. Steiner, H. Hirling, and M Unser. Design and validation of a tool for neurite tracing and analysis in fluorescence microscopy images. *Cytometry*, 58: 167 – 176, 2004

[6] C. Lopez-Molina, G. V. D. de Ulzurrun, J. M. Baetens, J. Van den Bulcke, and B. De Baets. Unsupervised ridge detection using second order anisotropic gaussian kernels. *Signal Processing*, 116:55–67, 2015

[7] C. Steger. An unbiased detector of curvilinear structures. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 20:113–125, 1998

[8] P. Kovesi. Image features from phase congruency. *Videre: Journal of Computer Vision Research*, 1:1–26, 1999

The Steger algorithm automatically yields a single-pixel wide skeleton. For the other methods, a single-pixel wide skeleton is extracted by segmenting the enhanced image. The skeletonisation process can be problematic as traditional skeletonisation algorithms over-segment the binary image and introduce many additional spurs depending on the roughness of the binary outline, that then require pruning. Thinning algorithms perform better, and yield skeletons that match the centreline of the network if the original binary image is symmetrical. Alternatively, watershed algorithms map the centreline directly and yield a fully connected skeleton (Fig. 5.9).

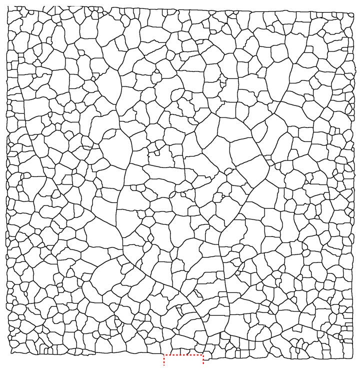


Figure 1.6: Watershed segmentation of the 'Feature Type' image to give a single pixel-wide binary skeleton.

1.2.4 *Setting boundary masks and features*

The boundary of the arena can also be used to define a binary mask that is imposed on the template (Fig. 1.2D), whilst additional components such as food sources can be defined as 'features' and also used to set the identity of the node in the graph representing the exit point of the arena (Fig. 1.2E). A number of automatic segmentation options are available within the main interface. Alternatively, manual segmentation can be selected, which opens a new window for binary editing that is described in Chapter 8.

1.2.5 *Analysis of polygonal regions*

The polygonal regions between the veins are also segmented from the complement of the pixel skeleton (Fig. 1.2F), which allows various morphological metrics to be calculated for each inter-vein region, including the area, shape and Euclidean distance from the skeleton. The volume of any plasmodial sheet in the polygonal regions can also be attributed to the nearest edge in the skeleton when calculating the flows (see Section 6.4.3).

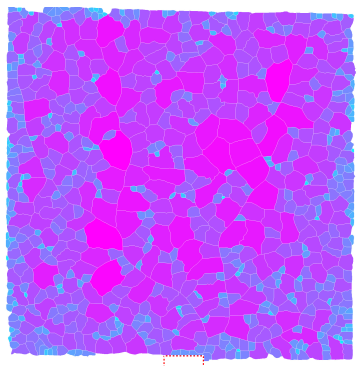


Figure 1.7: Analysis of the areas of the polygonal inter-vein regions.

1.2.6 *Comparison with a manually digitised ground-truth*

Pixel skeletons for each ridge enhancement-skeletonisation combination chosen can be evaluated against a manually digitised ground-truth (Fig. 1.2G), with a tolerance of half the minimum vein width for true positives (TPs), true negatives (TNs), false positive (FPs), and false negatives (FNs). This uses a separate GUI interface described in Chapter (7).

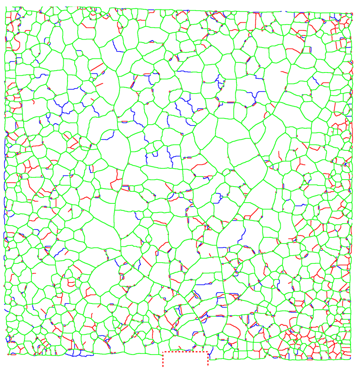


Figure 1.8: Precision-Recall analysis showing pixels that match the ground-truth in green, false negatives in red and false positives in blue.

1.2.7 *Estimation of vein width*

Once the skeleton has been extracted, it can be used as a template to interrogate the image locally to provide an estimate of the width of the veins (Fig. 1.2H). If the original binary image correctly covers the complete vein width, the Euclidean Distance Map (EDM) or the segmented area to length ratio, can be used to estimate the width. Alternatively, the original intensity image can be interrogated using greyscale granulometry techniques, or calibrated measurements of light transmission (Fig. 1.9). For example, the width of the veins and intervening plasmodial sheet at each pixel can be estimated using the Lambert-Beer law as $k \log_{10}(I_0/I)$, where I_0 is the incident light intensity, I the transmitted light intensity, and k a calibration coefficient based on the absorbance of the filled arena at the start where the path length is known.

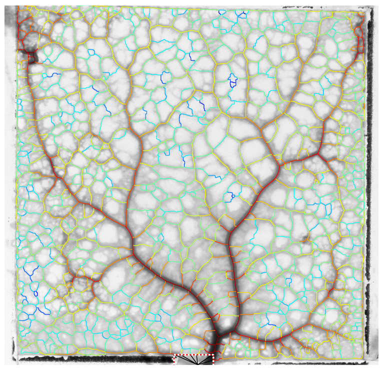


Figure 1.9: Estimation of network vein widths using the Lambert-Beer law. Veins are colour coded by $\log_{10}$ width in microns.

1.2.8 *Conversion to a weighted graph*

The pixel skeleton is then converted to a graph G, (Fig. 1.2J) with N nodes and M edges, where edge ij represents a vein segment between two junctions or a junction and free end, denoted as node i and node j. The graph can be adjusted to accommodate any 'features', such as food sources, by placing a new node at the centroid of the feature, connected to each edge incident on the boundary (Fig. 1.2K).

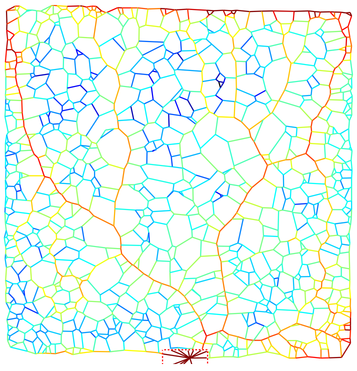


Figure 1.10: Conversion from a pixel-skeleton to a graph representation. Veins are colour coded by $\log_{10}$ width in microns.

1.2.9 *Measurement of graph metrics*

Each edge is associated with a vector of measurements, including the average intensity ($\bar{I}_{ij}$), length (l_{ij}), and average width (w_{ij}), determined by excluding pixels that overlapped with any larger veins at the end of the edge to give a centre-weighted estimate, more representative of the vein itself. The centre-weighted width is used to calculate the radius (r_{ij}), area ($a_{ij} = \pi r_{ij}^2$), volume ($v_{ij} = a_{ij} l_{ij}$), predicted resistance to flow or drag ($\theta_{ij} = l_{ij}/r_{ij}^4$).

The position of each node in the arena is used to calculate the Euclidean distance from a manually-defined reference point, denoted as node 0, (d_{i0}), and the hydraulic accessibility, measured as the path of minimum resistance, calculated using Dijkstra's algorithm. The edge betweenness centrality β_u for edge u is calculated as the proportion of all shortest paths (σ_{iuj}) between pairs of nodes i and j, that pass through u, to give a measure of the importance of a node or link to transport. Loss of the node or link with the highest betweenness centrality leads to the greatest increase in shortest path lengths.

Graphs can also be output for investigation using community detection algorithms that seek to partition the network into regions that have greater connectivity within the community than between communities. The way that different networks are structured can be compared using mesoscopic response functions [9,10].

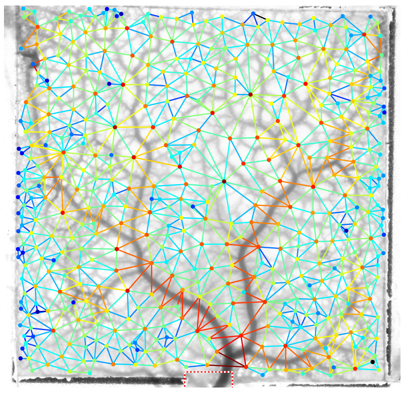
[9] S.-H. Lee, M.D. Fricker, and M.A. Porter. Mesoscale analyses of fungal networks as an approach for quantifying phenotypic traits. *J. Complex Networks*, 5:145, 2017

[10] J.-P. Onnela, D.J. Fenn, M.A. Reid, S.and Porter, P.J. Mucha, M.D. Fricker, and N.S. Jones. Taxonomies of networks from community structure. *Phys. Rev. E*, 86:036104, 2012

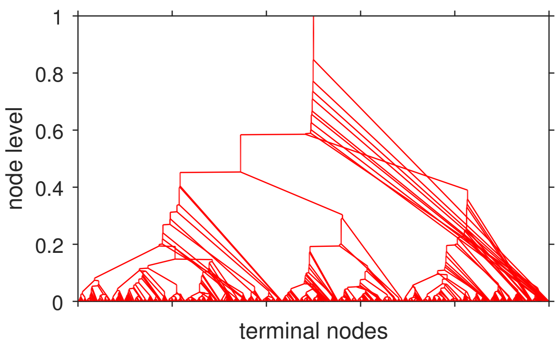
1.2.10 *Hierarchical network decomposition*

A dual-graph for the inter-vein regions can be constructed by linking each polygonal region to its neighbours, with the strength of the edge determined by the width of the intervening vein (Fig. 1.2L). In this implementation, the areas that are separated by the thinnest edge are fused first, then the areas separated by the second thinnest edge, and so on. Both the initial areas and the areas formed by fusions are represented as nodes in the dual-graph of the vein network, and these nodes are connected if the areas in question are formed by this hierarchical, fusion process (Fig. 1.11). Note that by construction, the dual graph of the vein network is a binary tree[11].

[11] E. Katifori and M.O. Magnasco. Quantifying loopy network architectures. *PLoS One*, 7:e37994, 2012



(a) Dual-graph



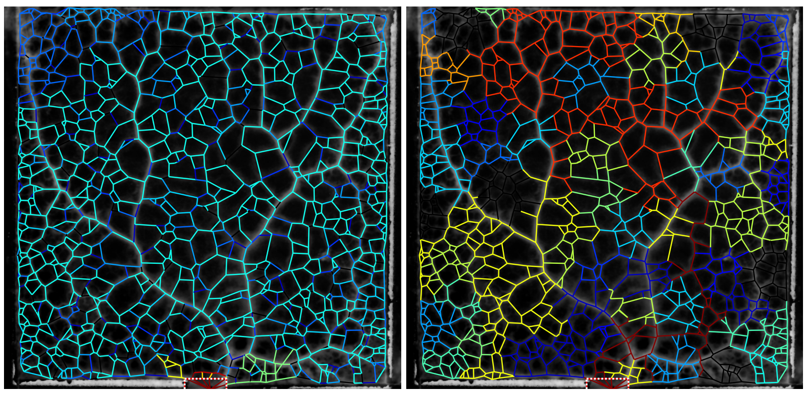
(b) Hierarchical loop decomposition

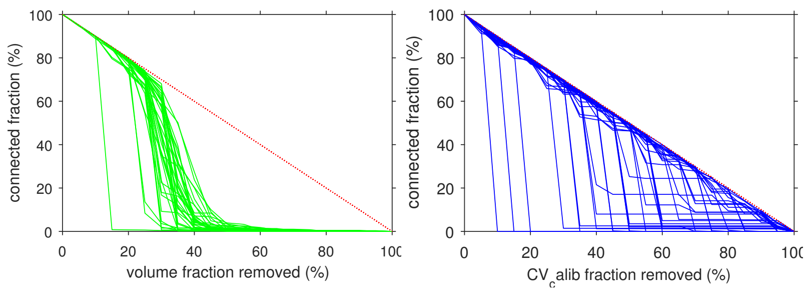
Figure 1.11: (a) Dual graph of the inter-vein with links colour-coded by intervening edge width, and nodes colour coded by area. (b) Binary tree from hierarchical decomposition of inter-vein dual graph.

The pattern of clustering in the binary tree provides an visual inication of the degree that the network forms a nested hierarchy (as is the case with *Physarum*), where there is some self-similarity moving up the node levels.

1.2.11 *Resilience*

In contrast to measures of network transport, there is no single standard measure to evaluate network resilience, as the extent of disruption depends on the type of damage inflicted. Nevertheless, robustness can be evaluated following removal of single edges, or as a function of successive edge removal in an ordered or random sequence (Fig. 1.12). In the case of the *Physarum* evacuation networks, the extent of 'damage' caused by vein removal can be assessed by the amount of the network that is still connected to the exit point, as a certain fraction of the veins are removed. As each iteration gives a unique pattern, multiple rns are required to build up an ensemble profile (Fig. 1.12 c,d).


(a) random vein removal (b) spatial vein attack



(c) random robustness profile (d) spatial robustness profile

Figure 1.12: Robustness analysis based on random removal of links (a,c) or spatially targetted removal of links (b,d). Colour-coding represents the level at which a vein was removed to help visualise the structure of the network during attack.

1.2.12 *Analysis of predicted flows*

The difference in vein volume between two time points can be used to estimate the volumetric current flow through the network exiting the arena (Fig. 1.2I). Veins that thin over time are the source of protoplasmic volume, while thickening veins are sinks (Fig. 1.13).

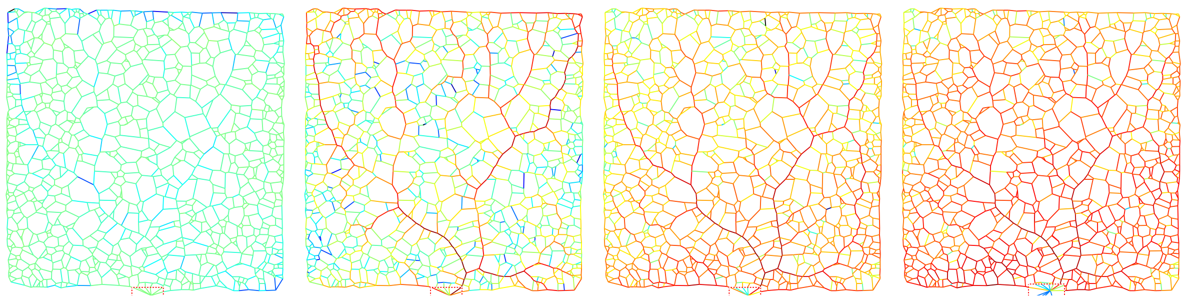

(a) Volumetric changes (b) $\log_{10}$ Current (c) $\log_{10}$ Speed (d) $\log_{10}$ Shear

Figure 1.13: Flow analysis based on volume changes as the plasmodium exits and enclosed arena.

To accommodate flows of material from any plasmodial sheet remaining in the inter-vein regions, the volume change for each pixel in the inter-vein region is allocated to the nearest edge in the network, based on the Euclidean distance map (EDM) from the pixel skeleton. The net difference in volume for all veins and inter-vein regions is assumed to exit the arena to conserve mass.

Given the net current at each node and the hydraulic conductance of each vein, we can calculate the unique current [12].

[12] L. L. M Heaton, E López, P. K Maini, M. D Fricker, and N. S Jones. Growth-induced mass flows in fungal networks. *Proc. R. Soc. B*, 277:3265–3274, 2010

2
Approaches to ridge enhancement and segmentation

2.1 *Introduction*

There are a wide range of different approaches that could be used to enhance the ridge-like structures in *Physarum* networks. This Chapter provides some of the theoretical background to the approaches implemented in the network analysis package.

2.2 *'Vesselness'*

One of the first methods to identify ridges exploited the local curvature of the intensity landscape as estimated from the Hessian (H_σ), comprising second-order partial derivatives, D_{aa} along direction a, of the intensity image (I), where the value of the standard deviation of the Gaussian kernel (σ) is varied over a range of scales that span the sizes of the underlying features[1]:

[1] A. F. Frangi, W. J. Niessen, K.L. Vincken, and M.A. Viergever. *Multiscale vessel enhancement filtering*, pages 130–137. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998

$$H_\sigma = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix} = \begin{bmatrix} \frac{\delta^2 I}{\delta x^2} * G_\sigma & \frac{\delta^2 I}{\delta x \delta y} * G_\sigma \\ \frac{\delta^2 I}{\delta x \delta y} * G_\sigma & \frac{\delta^2 I}{\delta y^2} * G_\sigma \end{bmatrix} \tag{2.1}$$

where

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}} \tag{2.2}$$

In the resultant scale-space representation, further information on ridge-like features can be extracted from the eigenvalues and eigenvectors of the Hessian, which show characteristic behaviour for a filamentous structure. By ordering the eigenvalues in terms of their absolute magnitude ($|\lambda_1| < |\lambda_2|$, for a 2D image), the smallest eigenvalue ($|\lambda_1|$) denotes the minimum change in intensity, with the corresponding eigenvector oriented along the centreline of the ridge, whilst the largest eigenvalue ($|\lambda_2|$) and eigenvector determine the orientation of the maximum curvature, normal to the ridge centreline. Prominent structures are distinguished from the background by relatively large values of the eigenvalues ($\sqrt{\lambda_1^2 + \lambda_2^2}$). In addition, the ratio $|\lambda_1|/|\lambda_2|$ gives an indication of how blob-like ($|\lambda_1| \approx |\lambda_2|$) or elongated and filament-like ($|\lambda_1| \ll |\lambda_2|$) the structure is at that point. Thus the 'Vesselness' (V_σ)

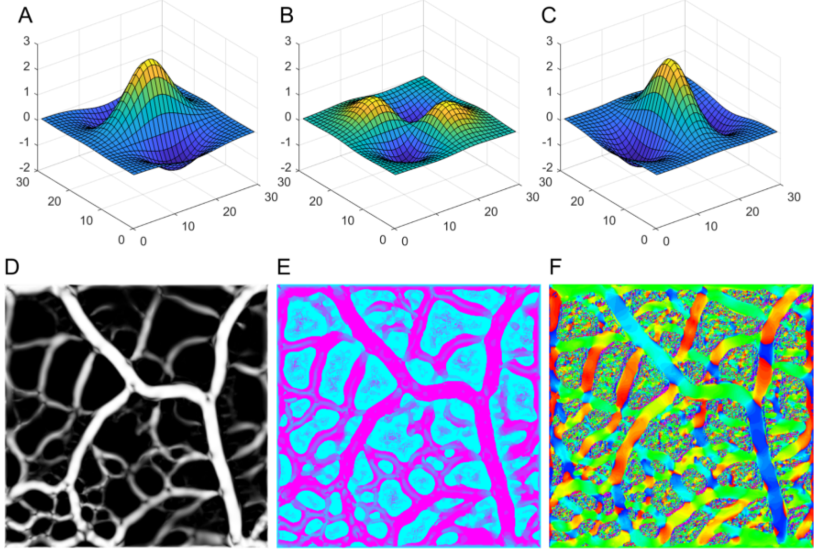


Figure 2.1: Frangi 'Vesselness' filter. (A-C) show surface plots of the D_{xx}, D_{xy} and D_{xy} filters at scale 7 that are used to calculate the second-order derivative of the image. The filters are shown inverted to highlight the shape of the ridge detector; (D) The 'Vesselness' output, calculated over 11 scales; (E) The scale at which the maximum response occurred; (F) the orientation of the maximum response.

measure (Frangi *et al.* 1998), defined by Equation (2.3), is large at those pixels that are part of a linear structure of scale (σ).

$$V_\sigma = e^{-\frac{\lambda_1^2}{2\beta^2\lambda_2^2}}\left(1 - e^{\frac{\lambda_1^2+\lambda_2^2}{2c^2}}\right) \tag{2.3}$$

Note that the relative contributions of the geometric ratio and the intensity components at a given scale (σ) are controlled by the coefficients β and c, respectively. Typically, β is set to 0.5 and c is set to half the maximum Hessian norm. Multi-scale 'Vesselness', for a given set of scales spanning the expected width of the vessels, can be computed as the maximum of the 'Vesselness' values calculated at each scale, and the eigenvectors at that scale used to define local orientation (Frangi *et al.* 1998).

2.2.1 *'Neuriteness'*

An alternative weighting of the eigenvalues of the Hessian matrix was proposed by Meijering *et al.* (2004)[2]:

[2] E. Meijering, M. Jacob, J. Sarria, P. Steiner, H. Hirling, and M Unser. Design and validation of a tool for neurite tracing and analysis in fluorescence microscopy images. *Cytometry*, 58: 167 – 176, 2004

$$H' = \begin{bmatrix} D_{xx} + \alpha D_{yy} & (1-\alpha)D_{xy} \\ (1-\alpha)D_{xy} & D_{yy} + \alpha D_{xx} \end{bmatrix} \tag{2.4}$$

Where α is set to be $-1/3$ such that the equivalent steerable filter[3] used in the calculation of the Hessian matrix is maximally flat in its longitudinal direction, effectively generating an anisotropic second order Gaussian filter. Conveniently, these kernels can be implemented as steerable filters constructed from a set of basis kernels (Freeman and Adelson, 1991). The 'Neuriteness' measure at scale σ, (N_σ) is determined from the modified eigenvalues as:

[3] W.T. Freeman and E.H. Adelson. The design and use of steerable filters. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 9:891–906, 1991

$$N_\sigma = \begin{cases} \frac{\lambda_\sigma}{\lambda_{\sigma,min}} & \text{if } \lambda_\sigma < 0 \\ 0 & \text{if } \lambda_\sigma \geq 0 \end{cases} \tag{2.5}$$

where λ_σ is the larger in absolute magnitude of the two modified eigenvalues, and $\lambda_{\sigma,min}$ is the smallest value of λ over all pixels such that:

$$\lambda_{\sigma,1}' = \lambda_{\sigma,1} + \alpha\lambda_{\sigma,2} \tag{2.6}$$

$$\lambda_{\sigma,2}' = \lambda_{\sigma,2} + \alpha\lambda_{\sigma,1} \tag{2.7}$$

$$\lambda_\sigma = \max\left(|\lambda_{\sigma,1}'|, |\lambda_{\sigma,2}'|\right) \tag{2.8}$$

$$\lambda_{\sigma,min} = \min_{\mathbf{p}\in I}(\lambda_\sigma) \tag{2.9}$$

$\lambda_{\sigma,1}, \lambda_{\sigma,2}$ are the eigenvalues of the Hessian matrix $H_\sigma(\mathbf{p})$, at pixel $\mathbf{p}$, for a given scale parameter σ.

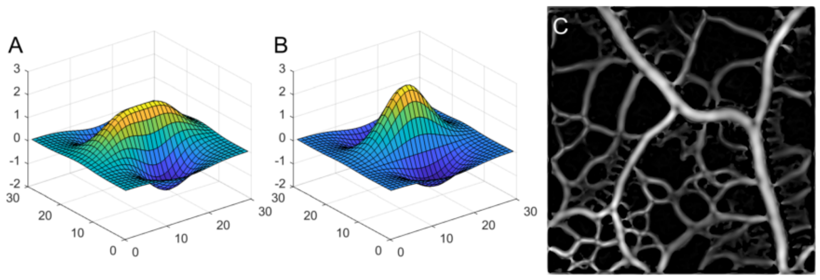


Figure 2.2: Meijering 'Neuriteness' filter. (A) A surface plot for the 'flattened' D_{xx} filter, shown inverted to highlight the shape of the ridge detector; (B) A normal D_{xx} filter for comparison; (C) The 'Neuriteness' output, calculated over 7 scales.

2.2.2 *Second-order anisotropic Gaussians SOAGK*

The use of second-order derivatives of anisotropic Gaussian kernels (SOAGKs) was developed further by Shui *et al.* (2012)[4] and Lopez-Molina *et al.* (2015)[5] to improve detection of ridge like elements, that are applied at a range of orientations to give anisotropic directional derivative (ANDD) filters at each scale (Fig. 2.3, A-C). On their own ANDD filters also generate extensions at the end of edge segments, termed edge-stretch, which has the benefit of improving local connectivity by filling in small gaps, particularly at junctions that occur in the 'Vesselness' filter for example, but with the disadvantage of adding spurious features at the end of edge segments. The latter errors can be minimised by using a fused detector that combines the ANDD filter with a small isotropic Gaussian as a geometric mean (Shui and Zhang, 2012). The enhanced edge image is taken as the maximum response at any scale and orientation (Fig. 2.3, E). These filters give strong responses when aligned to the dominant ridge at each scale, and provide estimates of the ridge intensity and ridge orientation without calculation of the eigenvalues and eigenvectors. In addition, the response at junctions is not attenuated to the same degree as the 'Vesselness' response because of the edge-stretch phenomena. Conversion to a single-pixel wide skeleton then uses local non-maximal suppression to identify key pixels on the ridge centerline, followed by hysteresis thresholding to identify pixels that form the connected skeleton.

[4] P.-L. Shui and W.-C. Zhang. Noise-robust edge detector combining isotropic and anisotropic gaussian kernels. *Pattern Recognition*, 45:806 – 820, 2012

[5] C. Lopez-Molina, G. V. D. de Ulzurrun, J. M. Baetens, J. Van den Bulcke, and B. De Baets. Unsupervised ridge detection using second order anisotropic gaussian kernels. *Signal Processing*, 116:55–67, 2015

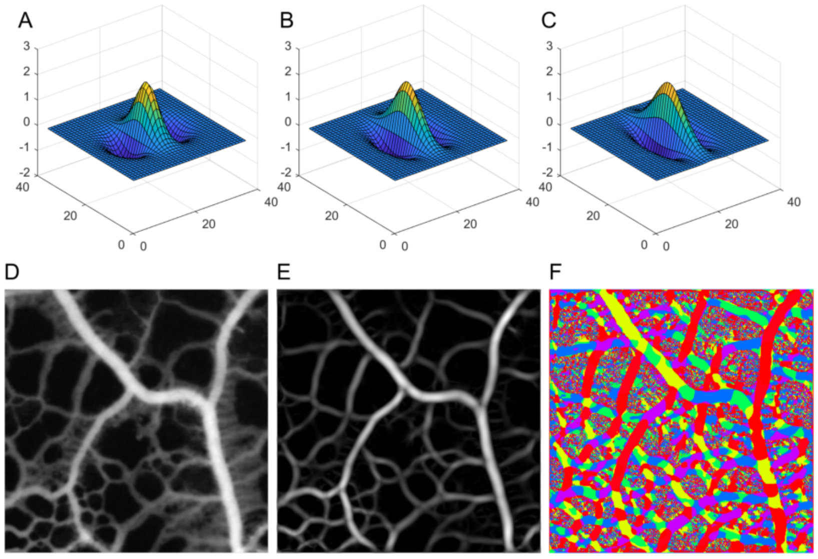


Figure 2.3: Second-order anisotropic Gaussian filter (SOAGK) after Llopez-Molin et al (2015). (A-C) Surface plots for the D_{xx} filter with 1,1.3 and 1.5 levels of anisotropy, shown inverted to highlight the shape of the ridge detector; (D) Original Physarum image (E) Output from the 'SOAGK' filter, calculated over 7 scales; (F) direction of the dominant ridge.

2.2.3 *The Steger curvi-linear detector*

An alternative strategy was proposed by Steger[6] to identify ridge centre-lines, following convolution with first and second order derivatives of (isotropic) Gaussian kernels without segmentation, from the coefficients of a second-order Taylor polynomial fit to the data (Fig. 2.4). Thus, the direction normal to the ridge is determined from the maximum absolute eigenvalue of the Hessian matrix, and the centre-line point determined from the first derivative of a quadratic polynomial along the normal line, with the strength of the ridge given by the second derivative. Using D_a and D_{aa} to represent the first and second partial derivatives along direction a, and (tn_x, tn_y) to represent the normal to the ridge centre-line at distance t, the centre-line point is given by:

[6] C. Steger. An unbiased detector of curvilinear structures. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 20:113–125, 1998

$$(p_x, p_y) = (tn_x, tn_y) \tag{2.10}$$

where

$$t = -\frac{D_x n_x + D_y n_y}{D_{xx} n_x^2 + 2D_{xy} n_x n_y + D_{yy} n_y^2} \tag{2.11}$$

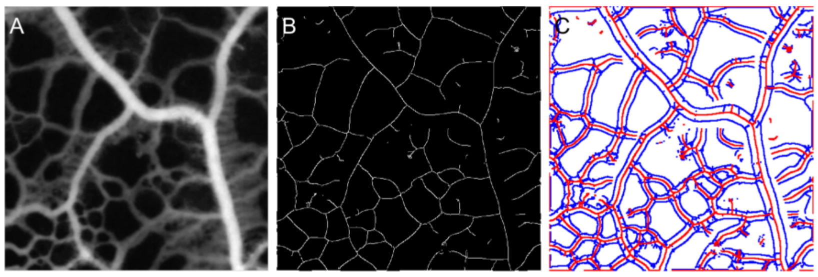


Figure 2.4: Steger filter. (A) Original image; (B) Center-line skeleton; (C) Centerline (red) and width estimate (blue).

2.2.4 *Intensity-independent enhancement using phase-congruency*

While ridge enhancement can be built on purely intensity-based filters, such as the Hessian or SOAGKs, these have the downside of being sensitive to changes in image contrast, which often leads to loss of a few pixels from the skeleton during the subsequent thresholding step, effectively disconnecting these edges. This can be ameliorated to some extent by inclusion of a local contrast equalisation step prior to enhancement (Shui and Zhang, 2012), or by the use of adaptive or hysteresis thresholding during segmentation (Lopez-Molina *et al.* 2015), or when linking points in the Steger algorithm (Steger, 1998). Nevertheless, in these approaches it is critical to establish a reliable, context-dependent threshold selection to achieve segmentation of a fully connected network.

Human observers face a similar challenge when trying to discriminate edges or ridges is a complex visual field. Morrone and Owens (1987)[7] proposed that human edge perception depends on the degree of phase congruency, independent of the brightness. Phase congruency can be estimated from the local energy determined by convolution of the image with Gabor filters at varying scale and orientation[8]. The phase congruency approach has been developed further as a generic means to extract a variety of image features by Kovesi[9]. Kovesi also introduced a range of improvements to the original measure to improve its overall utility, including log Gabor filters to increase the filter bandwidth, procedures for automated noise rejection, weighting to select against phase congruency of only a few frequencies, and improved spatial precision by including both the cosine and sine of the phase in the estimate. These provide good ridge enhancement, irrespective of image intensity, but also increase the number of parameters that can be tuned to achieve the best enhancement in any particular context.

[7] M.C. Morrone and R.A. Owens. Feature detection from local energy. *Pattern Recognition Letters*, 6:303 – 313, 1987

[8] S. Venkatesh and R. Owens. On the classification of image features. *Pattern Recognition Letters*, 11:339–349, 1990

[9] P. Kovesi. Image features from phase congruency. *Videre: Journal of Computer Vision Research*, 1:1–26, 1999

Following Kovesi, the local energy for a one-dimensional profile, $I(x)$ is given by:

$$E(x) = \sqrt{F^2(x) + H^2(x)} \tag{2.12}$$

where $F(x)$ is the signal with its DC component removed, and $H(x)$ is the Hilbert transform of $F(x)$, obtained by convolving the signal with a quadrature pair of log Gabor filters at scale n, and summing the even filter convolutions $(e_n(x))$ to give $F(x)$, and the odd filter convolutions $(o_n(x))$ to give $H(x)$. The phase congruency $PC(x)$, is normalised by the sum of the Fourier amplitudes $\sum_n A_n(x) \simeq \sum_n \sqrt{e_n(x)^2 + o_n(x)^2}$, with the addition of a small constant ε to improve stability at low Fourier amplitudes:

$$PC(x) = \frac{E(x)}{\sum_n A_n + \varepsilon} \tag{2.13}$$

The log Gabor filter bank is controlled by the minimum wavelength scale, the number of scales, the frequency bandwidth, and the number of filter orientations. Each of these requires some optimisation to achieve good enhancement for specific types of biological networks.

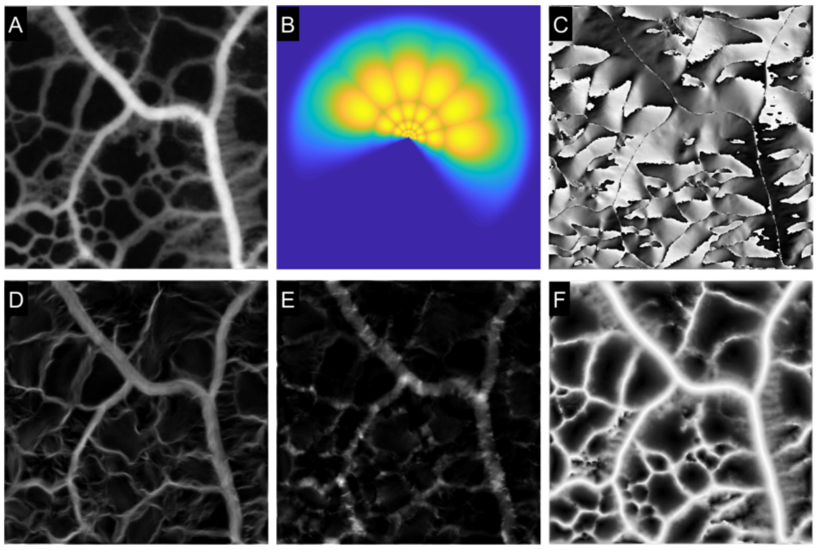


Figure 2.5: Phase Congruency filter. (A) Original image; (B) Set of log Gabor filters in the Fourier domain for 6 scales and 6 orientations; (C) Orientation image from the maximum response; (D) Maximum moment from the phase-congruency filter as a measure of edge strength; (E) Minimum moment from the phase-congruency response; (F) Local weighted mean phase angle ('Feature Type') output.

The next tuneable parameter controls the amount of noise rejection. To estimate the amount of noise adaptively from the image, Kovesi used a measure of the mean (μ_R) and variance (σ_R^2) of the Rayleigh distribution (R) describing the noise distribution at the smallest scale, with the assumption that ridges are relatively sparsely distributed in the image, so the mean will be dominated by background noise at this scale. Thus the noise threshold (T), is given by the mean noise response plus some number, k, of deviation units:

$$T = \mu_R + k\sigma_R \tag{2.14}$$

The local energy term $E(x)$ is therefore modified by subtracting the estimated noise (and setting any values below zero to zero).

The second set of tuneable parameters relate to the minimum spread of frequencies required to constitute a useful estimate of phase congruency. In the case of *Physarum* we are concerned with the detection of ridges, rather than lines or step functions. The expected power spectrum of a ridge falls off at $1/\omega^4$, where ω is the center frequency of the filter, which gives an expected distribution of frequency responses strongly skewed towards low-frequency end. The significance of $PC(x)$ can be down-weighted if the spread of frequencies is too narrow, however, in the case of ridge detection, this criterion should not be too harsh. Kovesi provides an estimate of the frequency spread by considering the normalised ratio of the sum of the Fourier amplitudes divided by the maximum response:

$$s(x) = \frac{1}{N}\left(\frac{\sum_n A_n(x)}{A_{max}(x) + \varepsilon}\right) \tag{2.15}$$

Where N is the total number of scales, $A_{max}(x)$ is the maximum filter response at x, and ε prevents division by zero. To penalise regions with few frequency components, the weighting function is constructed as a sigmoidal function:

$$W(x) = \frac{1}{1 + e^{\gamma(c - s(x))}} \tag{2.16}$$

where c is the cut-off value below which phase congruency values are penalised, and γ is a gain factor that controls the sharpness of the cutoff.

$$PC(x) = \frac{W(x)\lfloor E(x) - T \rfloor}{\sum_n A_n(x) + \varepsilon} \tag{2.17}$$

Where $\lfloor\ \rfloor$ denotes that $E(x) - T$ is equal to itself for positive values and zero otherwise.

The final amendment that Kovesi proposes is to include the information from both the cosine of the phase deviation, which should be large if phase congruency is high, and the absolute value of the sine of the phase deviation, which should be small (2.18):

$$\Delta\Phi_n(x) = \cos(\phi_n(x) - \bar{\phi}(x) - |\sin(\phi_n(x) - \bar{\phi}(x)| \tag{2.18}$$

This gives the complete estimate of phase congruency as:

$$PC(x) = \frac{\sum_n W(x)\lfloor A_n(x)\Delta\Phi_n(x) - T \rfloor}{\sum_n A_n(x) + \varepsilon} \tag{2.19}$$

The two dimensional extension of the phase congruency gives:

$$PC(x) = \frac{\sum_o \sum_n W_o(x)\lfloor A_{no}(x)\Delta\Phi_{no}(x) - T_o \rfloor}{\sum_o \sum_n A_{no}(x) + \varepsilon} \tag{2.20}$$

Where o is the index over orientations.

3 Installation

All the programs were written in MATLAB (The Mathworks, Nantick) and are packaged in a single compiled executable file for distribution as a standalone package. The program, manual and tutorial can be downloaded from:

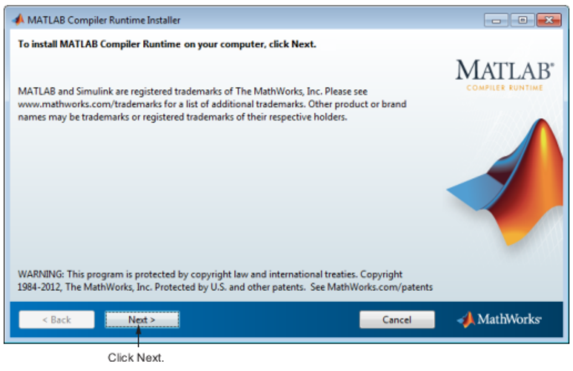
http://www.markfricker.org

The software has been tested on Windows 10, and requires a minimum screen resolution of 1600 x 900. In addition, an appropriate version of the MATLAB Compiler Runtime (MCR) is required to install the set of shared libraries that enables execution of the compiled MATLAB application. The MCR should automatically download from the MathWorks website when the program is installed for the first time. Alternatively MCR can be downloaded from the MathWorks Website:

http://www.mathworks.com/products/compiler/mcr.

Versions are also available as a MATLAB application that runs within the MATLAB 2017a environment and requires the image processing toolbox.

To install the MCR and run the program, double-click the compiled MATLAB self-extracting archive file. This extracts the MATLAB Runtime Installer from the archive, along with all the files that make up the deployed MATLAB environment. Once all the files have been extracted, the MATLAB Runtime Installer starts automatically. When the MATLAB Runtime Installer starts, it displays the following dialog box. Read the information and then click **Next** to proceed with the installation.

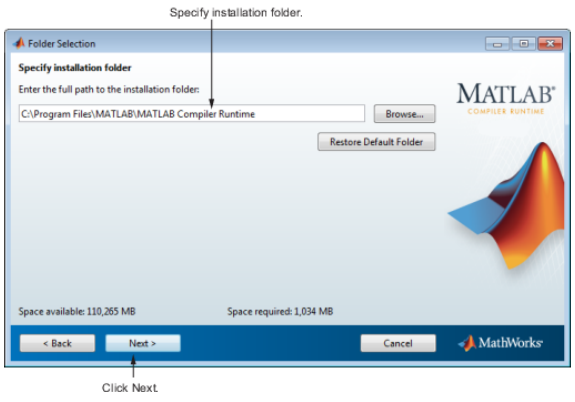


To install MATLAB Compiler Runtime on your computer, click Next.

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. Please see www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

WARNING: This program is protected by copyright law and international treaties. Copyright 1984-2012, The MathWorks, Inc. Protected by U.S. and other patents. See MathWorks.com/patents

Specify the folder in which you want to install the MATLAB runtime in the Folder Selection dialog box and click **Next**.



Specify installation folder

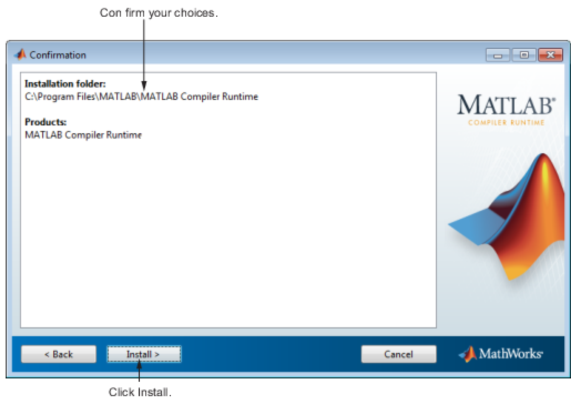
Enter the full path to the installation folder:

C:\Program Files\MATLAB\MATLAB Compiler Runtime

Space available: 110,265 MB Space required: 1,034 MB

Note: On Windows systems, you can have multiple versions of the MATLAB runtime on your computer but only one installation for any particular version. If you already have an existing installation, the MATLAB runtime Installer does not display the Folder Selection dialog box because you can only overwrite the existing installation in the same folder.

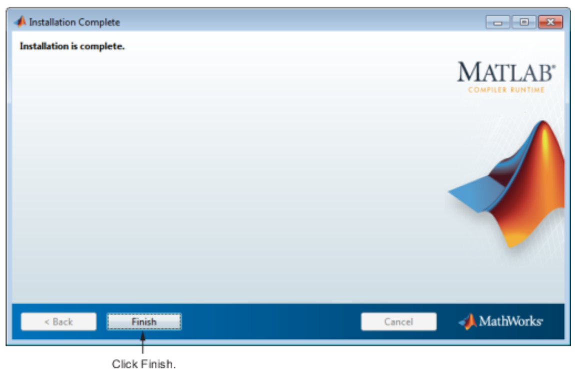
Confirm your choices and click **Install**. The MATLAB Runtime Installer starts copying files into the installation folder



Installation folder:
C:\Program Files\MATLAB\MATLAB Compiler Runtime

Products:
MATLAB Compiler Runtime

Click **Finish** to exit the installer.



Installation is complete.

MATLAB Runtime Installer Readme File: A readme.txt file is included with the MATLAB Runtime Installer. This file, visible when the MATLAB Runtime Installer is expanded, provides more detailed information about the installer and the switches that can be used with it.

A number of additional files needed to run the full suite of programs may also be installed at the same time as the main program. For example, the *bioformats* package (Linkert et al. 2010[1]) has been designed to read in images from different microscope manufacturers and store them in a standardised format. Full details are available on the open microscopy website:

`http://www.openmicroscopy.org/site/support/bio-formats4/`

The bioformats_packages.jar program needs to be available on the search path or installation directory of the matlab programs. bioformats_package.jar is available from:

`http://downloads.openmicroscopy.org/bio-formats/4.4.9/`

The latest version of Java needs to be installed, and is available from:

`http://www.java.com/en/`

Output of images at full resolution uses *export_fig.m* originally written by Oliver Woodford (2008-2014) and now maintained by Yair Altman (2015-). When exporting to vector format (PDF or EPS) this function requires that ghostscript is installed on your system. Ghostscript can be downloaded from:

`http://www.ghostscript.com`.

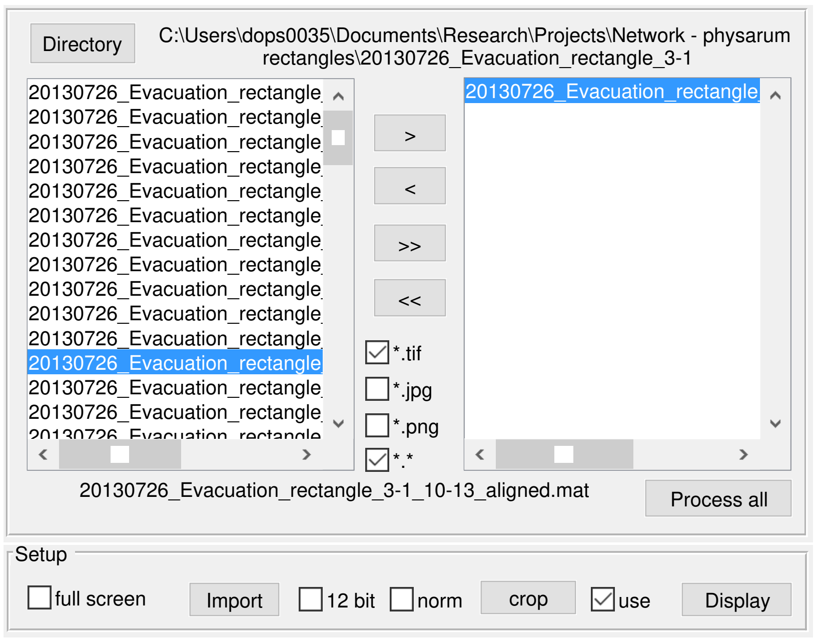
When exporting images to eps, *export_fig* additionally requires pdftops, from the Xpdf suite of functions. This can be downloaded from:

`http://www.foolabs.com/xpdf`

[1] M. Linkert, C.T. Rueden, C. Allan, J.-M. Burel, W. Moore, A. Patterson, B. Loranger, J. Moore, C. Neves, D. MacDonald, A. Tarkowska, C. Sticco, E. Hill, M. Rossner, K. W. Eliceiri, and J. R. Swedlow. Metadata matters: access to image data in the real world. *The Journal of Cell Biology*, 189:777–782, 2010

4
Loading images

The Physarum network GUI automatically displays any image files present in the current directory with the default *.tif extension when the program is started (Fig. 4.1). The working directory can be changed using the **Directory** button, and additional file types can be displayed using the ***.jpg**, ***.png** or ***.*** checkboxes. A wide range of images can loaded directly into the program using the bioformats package, or by clicking the **Import** button which opens the Import GUI (see Chapter 9), which has facilities to crop, filter and align images by cross-correlation.



C:\Users\dops0035\Documents\Research\Projects\Network - physarum rectangles\20130726_Evacuation_rectangle_3-1

20130726_Evacuation_rectangle
20130726_Evacuation_rectangle
20130726_Evacuation_rectangle
20130726_Evacuation_rectangle
20130726_Evacuation_rectangle
20130726_Evacuation_rectangle
20130726_Evacuation_rectangle
20130726_Evacuation_rectangle
20130726_Evacuation_rectangle
20130726_Evacuation_rectangle
20130726_Evacuation_rectangle
20130726_Evacuation_rectangle
20130726_Evacuation_rectangle
20130726_Evacuation_rectangle
20130726_Evacuation_rectangle

20130726_Evacuation_rectangle

[✓] *.tif
[ ] *.jpg
[ ] *.png
[✓] *.*

20130726_Evacuation_rectangle_3-1_10-13_aligned.mat

[ ] full screen [ ] 12 bit [ ] norm [✓] use

Figure 4.1: *Image load panel: Images in standard file formats can be imported, cropped and the relevant channel for processing selected.*

A single image can be selected for processing in the left-hand list box using a left mouse-click to highlight the name in the list followed by clicking the **>** arrow. Multiple files in any combination can be selected using *Ctrl + left click* to highlight the files, followed by the **>** arrow. Alternatively, all the files can be selected with the **>>** arrow and will appear in the right-hand list box. Individual files, or all the files can be removed using the **<** and **<<** arrows, respectively.

The default bit depth is 8-bit. If images are saved in 12-bit format, the **12-bit** checkbox needs to be ticked, as the file header will often report that the image is saved in 16-bit format and will not be displayed properly. Once imported, images are normalised to the range 0-1. The **norm** checkbox will also re-scale the input image between 0 and 1.

4.1 *Image display controls*

Once an filename is displayed in the right list-box, the image is automatically displayed in the main window (Fig. 4.2), or can be displayed at any time using the **Display** button. If multiple files have been selected, the last one in the sequence is shown. Any image can be viewed by clicking on it's filename in the right-hand listbox. At the same time, a thumbnail for the *initial* image is also displayed in the thumbnail shortcut bar immediately underneath the main display. Thumbnails for key steps in the processing sequence are displayed once the appropriate step has completed successfully, and can be used subsequently to switch rapidly between different images.

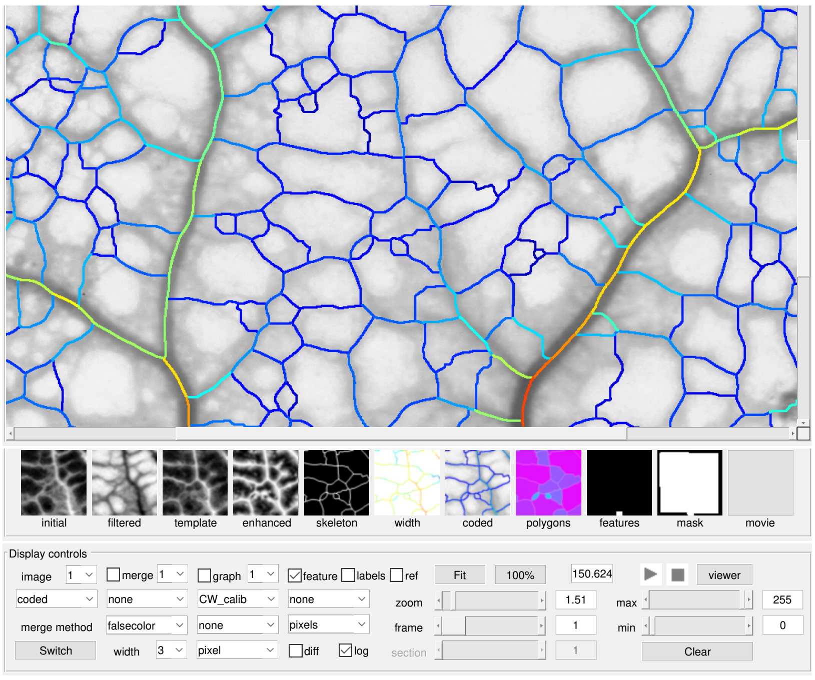


Figure 4.2: Main image display, thumbnail shortcut bar and associated controls to adjust zoom, image contrast and various annotation overlays

The image for display can be selected in the **image** drop-down menu for the channel shown in the adjacent drop-down menu. The default is for channel 1. To help the user decide on the best channel to process, the checkboxes underneath the **initial** thumbnail can be used to toggle display of the original **R**ed **G**reen and **B**lue channels of the original RGB image. A number of options to adjust the image displayed are available in the **Display controls** panel. The main controls that are relevant at this stage adjust the image size and contrast, through the **zoom**, **white level** and **black level** sliders. The **Fit** button resizes the image to ensure all of it is visible, whilst the **100%** button gives a 1:1 image:screen pixel scaling. The other commands in the **Display controls** panel will be covered at a later stage in the manual.

Other images can be merged with the image currently displayed (Fig. 4.3), using the merge drop-down menu, and the *'falsecolor'* option as the **merge method** in the **Display controls** panel.

The identity of the two images can be reversed using the **Switch** button. The method used to compare the two images can be selected from:

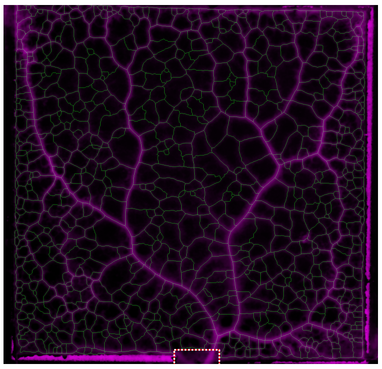


Figure 4.3: Illustration of the falsecolor merge function that for the template image and the skeleton image

- *'falsecolor'* : the first image is shown in magenta and the merge image is shown in green
- *'blend'* : The two images are alpha-blended retaining their original colors, but at 50% transparency.
- *'diff'* : the difference between the two images is displayed
- *'montage'* : the two images are shown side-by-side

4.2 *Cropping the initial image and selecting the channel*

The **crop** button will automatically show the original image, sized to fill the display window, and prompts the user to drag a rectangle on the image display using the mouse to enclose the desired region-of-interest (ROI). On completion of the rectangle, the image is cropped and the resulting sub-region displayed. If there is an error in the region selected, the process can be repeated by clicking on the filename in the listbox to reload the original image. The coordinates of the cropped region are stored in the parameter file associated with the image and will be re-applied everytime the image is loaded if the adjacent **use** checkbox is ticked.

5
Network extraction

5.1 Measurement of the approximate vein diameter

Images of the *Physarum* network may have been collected at different pixel resolutions, depending on the camera settings, and may span different width scales depending on the experimental treatment.

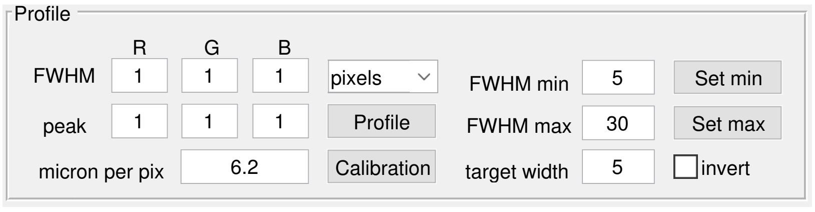

Profile
FWHM R 1 G 1 B 1 pixels
peak 1 1 1
micron per pix 6.2
FWHM min 5
FWHM max 30
target width 5
invert ☐

Figure 5.1: Image profile controls: these allow the user to measure the physical size of structures in the image from transects that automatically calculate the FWHM of the underlying feature. These are used to estimate the minimum and maximum diameters of the tubules to standardise all subsequent processing steps

To standardise all the subsequent processing steps, it is useful to define the expected minimum and maximum width of the veins using a transect drawn manually on the image, using the controls in the **Profile** panel (Fig. 5.1).

When either the **Set min** or **Set max** buttons are clicked, the user is prompted to draw a two-point transect on the image across a vein that, by eye, appears to be close to the smallest or largest tubule diameter, respectively. On completion of the second mouse click, a graph of the transect is displayed in the upper **graph** panel (Fig. 5.4), with the line colour reflecting the intensity values of the original RGB channels.

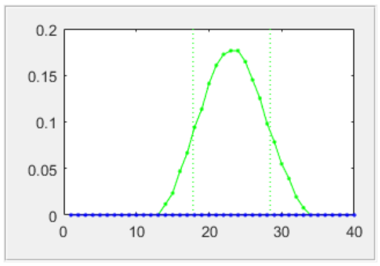


Figure 5.2: Profile measurements: the graph shows the intensity profile along a user-defined transect drawn on the image, along with the full-width half-maximum (FWHM) automatically calculated from the peak height, in this case for the green channel

In addition, the full-width at half-maximum (FWHM) peak intensity is automatically calculated and displayed as:

- two dotted vertical lines on the graph;
- the estimated pixel width in the **FWHM min** and **FWHM max** text boxes, respectively;
- the estimated pixel width and peak intensity (in normalised units) values in the **FWHM** and **peak** text boxes for the **R**,**G** and **B** channels;

Values for FWHM are given in pixels, whilst the peak intensity are given in normalised units ranging from 0 to 1. Additional profiles can be drawn at any stage without updating the **FWHM min** and **FWHM max** text boxes by using the **Profile** button.

The value of $FWHM_{min}$ is used to calculate a resampling factor needed to ensure that the minimum apparent vein width is at least 5 pixels wide to reduce pixelation errors later on. The actual target width is set by the **target width** box. Likewise, $FWHM_{max}$ is used as a guide to determine the number of scales to use in the subsequent steps to ensure that the largest veins are correctly segmented, and also that structures above this limit are identified as separate image features.

The **Calibration** button prompts the user to define the physical scale between two measurement points on the image, which then updates the adjacent textbox to give the pixel spacing in **micron per pixel**. Alternatively, the pixel size can be entered manually in the text box if the information is available from the original image file.

5.2 *Initial image processing*

The **Image processing** panel contains the controls used to resample the image according to the target minimum width, and allows for background subtraction and some image filtering (Fig. 5.3).

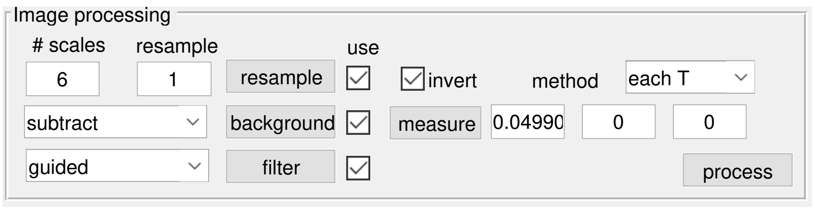


Figure 5.3: Image processing panel: These controls prepare the image for subsequent analysis by resampling, background subtraction and filtering

Each step has a number of options that can be selected from the adjacent drop down menu on the left. In addition, some steps allow the user to manually modify the image produced at a particular step, or to set the values for specific parameters. The **use** checkboxes enable the user to toggle particular steps on or off to explore the impact on the final result. If a particular step is not operational, the corresponding controls are greyed-out. The **process** button runs the entire sequence of steps on the currently selected image.

If any parameters are changed, a *parameters* file is automatically saved in the *parameters* sub-directory using the current filename appended with '_param' extension.

This *parameters* file is automatically re-loaded when the image is opened. In addition, many methods have additional parameters that are not available on the main interface. These can be edited using the **edit** button, which opens a text-editing table showing the defaut and current parameters. Values can be **reset** to the default parameters, or **saved** under a different name.

The **Process all** button in the **Image Load** panel runs the complete process and analysis sections for every selected file and also automatically saves the data and images.

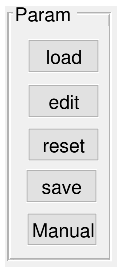


Figure 5.4: Control options to load, edit, reset or load the parameters used for processing and analysis

5.3 *Image resampling*

The value of $FWHM_{min}$ estimated from the **Profile** measurements is automatically used to calculate the resampling factor needed to ensure that the minimum vein is typically 5 pixels wide, although the actual target width can be set using the **target width** box. Likewise, the $FWHM_{max}$ value is used to determine the number of scales to use in the subsequent steps to ensure both that the largest veins are correctly segmented, and also that structures above this limit are identified as separate features. The resample **use** checkbox is active by default. If the box is unchecked, no resampling takes place, but it is possible that subsequent steps do not perform as expected, if for example, boundary masks or feature masks have been defined with resampling in place. Once the **resample** button is clicked, the initial image is resampled and displayed. The *resample* thumbnail is updated to show a small icon taken from the center of the resampled image.

5.4 *Background measurement and correction*

The **Background** checkbox activates a number of options to automatically or manually correct the background including:

- *'Subtract'* : Subtracts a constant value from the image. The *value* is set in the adjacent text boxes for each RGB channels (if present). The value can be input manually, or determined using the **measure** button which prompts the user to define a background ROI on the image for the measurement.

- *'Opening'* : the image is processed with an opening function using a disk-shaped kernel with the radius set by a fraction of **FWHM max / resample**. The default is 1.2× the radius of the largest feature expected. This removes any features smaller than $2 * FWHM_{max}$ and provides an estimate of the local background around each pixel. The opened image is subtracted from the original to correct for the local background.

- *'Surface fit'* : this finds all the local minima across the image and fits a surface to points in the 10-90% interval. The surface is converted to an image and subtracted from the original.

- *'Sub low pass'* : the image is filtered using a Gaussian kernel with a large radius sufficient to remove all the high-frequency information in the image. The standard deviation for the Gaussian kernel is calculated as $FWHM_{max}$, or approximately twice the size of the largest tubule diameter. The low pass image is subtracted from the original and the image re-normalised.

5.5 Filtering the image to improve signal-to-noise

Most simple noise reduction algorithms use isotropic kernels and smooth the noise in the image equally in all directions. This is not desirable when analysing networks, as the vein boundaries will become blurred. Instead a number of adaptive anisotropic filters are provided that smooth within the vein network structures, but do not spread across boundaries. In addition, the image intensities can subsequently be rescaled using local contrast-limited adaptive histogram equalisation (CLAHE).

The **filter** controls include:

- *'CLAHE'* : applies contrast-limited adaptive histogram equalisation[1] to the image to expand the contrast range over local regions.
- *'Coherence'* : applies an anisotropic diffusion filter written by Dirk-Jan Kroon[2] that smooths regions of low variance, but avoids blurring of object boundaries.
- *'Guided'* : applies an edge-preserving smoothing filter that is guided by the intensities in the original image[3]
- *'Coherence + CLAHE'* : applies the *'Coherence'* filter followed by histogram equalisation.
- *'Guided + CLAHE'* : applies the *'Guided'* filter followed by histogram equalisation.

[1] K. Zuiderveld. *Contrast limited adaptive histograph equalization*, pages 474–485. Graphic Gems IV. Academic Press Professional, San Diego:, 1994

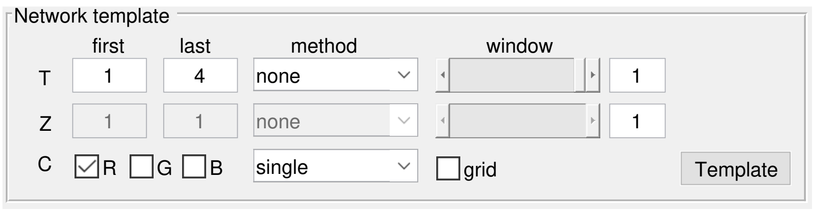
[2] D. J. Kroon, C. H. Slump, and T. J. Maal. Optimized anisotropic rotational invariant diffusion scheme on cone-beam ct. *Med Image Comput Comput Assist Interv*, 13:221–8, 2010

[3] K. He, J. Sun, and X. Tang. Guided image filtering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35:1397–1409, 2013

The filtering and contrast adjustment are applied to aid segmentation of the pixel skeleton. Quantitative measurements always refer back to the original image intensities. CLAHE is only needed if the subsequent segmentation step is dependent on image intensities, although visually it helps the user see detail across the whole image.

5.6 Setting up the network template

Each image in the time series can be used as it's own template for skeletonisation. Alternatively a single template image can be constructed from a projection of the data along the **T**ime dimension (or **Z** dimension if present) (Fig. 5.5).


Network template

	first	last	method	window	
T	1	4	none		1
Z	1	1	none		1
C	☑R ☐G ☐B		single	☐grid	Template

Figure 5.5: Template panel: These controls set the parameters for the template image prior to enhancement

Provided there is no lateral movement of the veins, this gives a single skeleton that can be applied to each image in turn to extract the same network. Each node and edge will therefore have a unique identity that is consistent across the whole image series. The images to combine for the template can be a subset defined by the **first** and **last** text-boxes, uisng a projection algorithm slected by the **method** drop-down menu that includes:

- *'none'* : no projection;
- *'single'* : uses a single image selected by the **first** text-box as the template;
- *'selected'* : uses the selected range of images;
- *'all'* : uses all the images, effectively the same as no projection;
- *'max proj'* : takes a maximum projection between the **first** and **last** images
- *'mean proj'* : takes an average projection between the **first** and **last** images
- *'med proj'* : takes a median projection between the **first** and **last** images
- *'max med'* : applies a rolling median filter of the **window** specified over the whole time-series and then takes a maximum projection between the **first** and **last** images
- *'max mean'* : applies a rolling averaging filter of the **window** specified over the whole time-series and then takes a maximum projection between the **first** and **last** images

5.7 *Setting up a boundary mask*

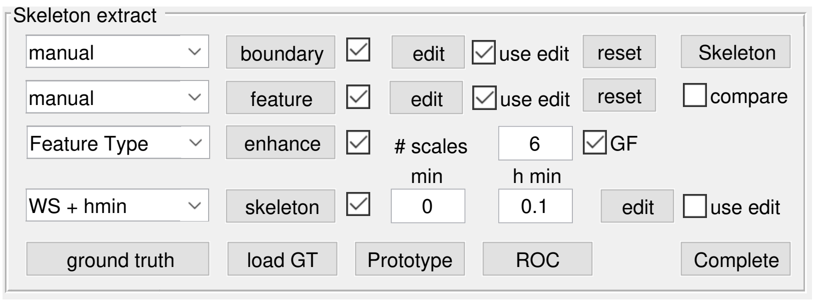


Figure 5.6: Skeleton extraction panel: These controls set the parameters to mask the template image and convert it to a single-pixel wide skeleton

In some instances it may be appropriate to define a boundary mask to exclude regions that should not be analysed, or to restrict the analysis to a specific region of the plasmodium. The default setting is not to use a boundary, but ticking the boundary **use** checkbox will enable the boundary controls.

The adjacent dropdown menu provides a number of options including:

- *'Holes'* : this segments the main fluorescent structures using an automatic threshold determined by Otsu's method[4], that minimizes the intraclass variance of the foreground and background distributions, but does not fill in any gaps or holes in the resulting binary image.

[4] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Trans. Systems, Man, Cyber.*, 9:62–66, 1979

- *'No holes'* : follows the same approach as the *'holes'* option, but also fills any internal holes in the binary image.
- *Background* : prompts the user to define a ROI in a background area of the image. The threshold is then calculated as the $background\ mean + 2 * SD\ units$.

Clicking the **Boundary** button runs the chosen method, displays the segmented binary image alongside the initial grayscale image to aid comparison, and updates the boundary thumbnail (Fig. 5.7). Care is needed with this step as the thresholding operation may introduce breaks in some dim veins, which are then not considered in subsequent processing steps.

It is possible that the automatic boundary settings do not provide the desired masking of unwanted information, requiring the user to define the mask manually. The **edit** button will open an additional window with a set of tools to allow manual adjustment of the binary image. Full details of the binary editing program are given in Chapter **??**. If a boundary image has been defined manually, the **use edit** checkbox is automatically activated and the manually defined boundary will be applied to subsequent processing steps.

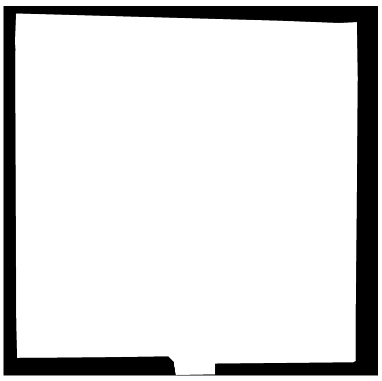


Figure 5.7: Boundary mask manually defined to match the experimental arena

5.8 Defining additional image features such as food sources

In addition to the plasmodial veins, the experiment may also include sheet-like regions of plasmodia or food sources. A different set of controls are available to allow the user to segment these structures independently. To enable the **Feature** controls, the feature **use** checkbox needs to be ticked. The adjacent dropdown menu then gives access to several different methods including:

- *'Auto'* : An automatic multi-threshold is used to partition pixel intensities into three bins, and the upper bin selected. This will extract the brightest objects, but it does not usually give a good unique segmentation of features.
- *'Opening'* : Uses image opening to calculate a modified image after grayscale opening to remove all objects smaller than $FWHM_{max}$, and then converts the resultant image to a binary mask using an automatic threshold. This is more effective at detecting larger objects, but the resultant binary image extends beyond the original boundary.
- *'Active contour'* : Performs the same image opening step as 'opening' above, but then 'shrinks' the features detected down

to match the underlying intensity profile using an active contour algorithm. This often performs better than a simple opening operation and is set as the default.

Clicking the **Feature** button runs the chosen method, updates the feature thumbnail and displays the segmented binary image superimposed on the initial grayscale image in falsecolour. Once the features have been defined, the **feature** checkbox in the **display** panel will superimpose the feature outline on any displayed image in a red-and-white border.

If no satisfactory segmentation of the features can be achieved with the automatic settings, the feature **edit** button opens the binary editing window to allow manual adjustment of the binary image. Full details of the binary editing program are given in Chapter **??**. If a feature image has been defined manually, the **use edit** checkbox is automatically activated and the manually defined features will be used in subsequent processing steps.

5.9 Enhancing the vein network

A number of options can be used to improve the relative contrast of veins prior to segmentation using kernels designed to pick-out 'ridge' like features that are applied over a range of scales and angles. These include:

- *'Frangi'* : This calls the Matlab implementation of the classic Frangi[5] 'vesselness' filter written by Marc Schrijver and Dirk-Jan Kroon and available from the Mathworks website (Fig. 5.8 (a)). This gives a strong response for bright features where the second-order derivative of the image (Hessian) shows a strong anisotropy.

[5] A. F. Frangi, W. J. Niessen, K.L. Vincken, and M.A. Viergever. *Multiscale vessel enhancement filtering*, pages 130–137. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998

- *'Neuriteness'* : Applies a version of the second-order anisotropic Gaussian kernel originally proposed by Meijering *et al.* (2004)[6] as part of their 'Neuriteness' detector. This uses a slightly flattened second-order Gaussian kernel at a range of scales and angles to give better discrimination of ridge-like structures, but does not include any additional information related to the anisotropy of the ridges (Fig. 5.8 (b)).

[6] E. Meijering, M. Jacob, J. Sarria, P. Steiner, H. Hirling, and M Unser. Design and validation of a tool for neurite tracing and analysis in fluorescence microscopy images. *Cytometry*, 58: 167 – 176, 2004

- *'SOAGK'* : Applies the multi-scale ridge detector developed by Lopez-Molina *et al.* (2015)[7] that uses anisotropic second-order Gaussian kernels. These can be configured flexibly in terms of size, orientation and anisotropy. This gives good ridge enhancement, but still retains the variation in local intensity along the tubules that can make subsequent segmentation more difficult. (Fig. 5.8 (c)).

[7] C. Lopez-Molina, G. V. D. de Ulzurrun, J. M. Baetens, J. Van den Bulcke, and B. De Baets. Unsupervised ridge detection using second order anisotropic gaussian kernels. *Signal Processing*, 116:55–67, 2015

- *'Feature Type'* : Uses the phase-congruency approach developed by Peter Kovesi[8] [9] to provide contrast-invariant ridge detection over a range of scales and angles. The MATLAB implementation[10] provides a number of outputs, including the level of phase congruency as a measure of the edge strength, but also the 'Feature Type', calculated as the weighted mean phase angle at every point in the image (Fig. 5.8 (d)). A value for the feature type of $pi/2$ corresponds to a bright line, 0 corresponds to a step and $-pi/2$ is a dark line. The feature type has proved to be one of the most robust and reliable outputs for subsequent segmentation, as all ridges, irrespective of their original intensity are identified with equal strength in the feature type image (Fig. 5.8 (d)).

[8] P. Kovesi. Image features from phase congruency. *Videre: Journal of Computer Vision Research*, 1:1–26, 1999

[9] P. Kovesi. Phase congruency: A low-level image invariant. *Psychological Research*, 64:136–148., 2000a

[10] P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing, 2000b. Available from: `http://www.peterkovesi.com/matlabfns/`

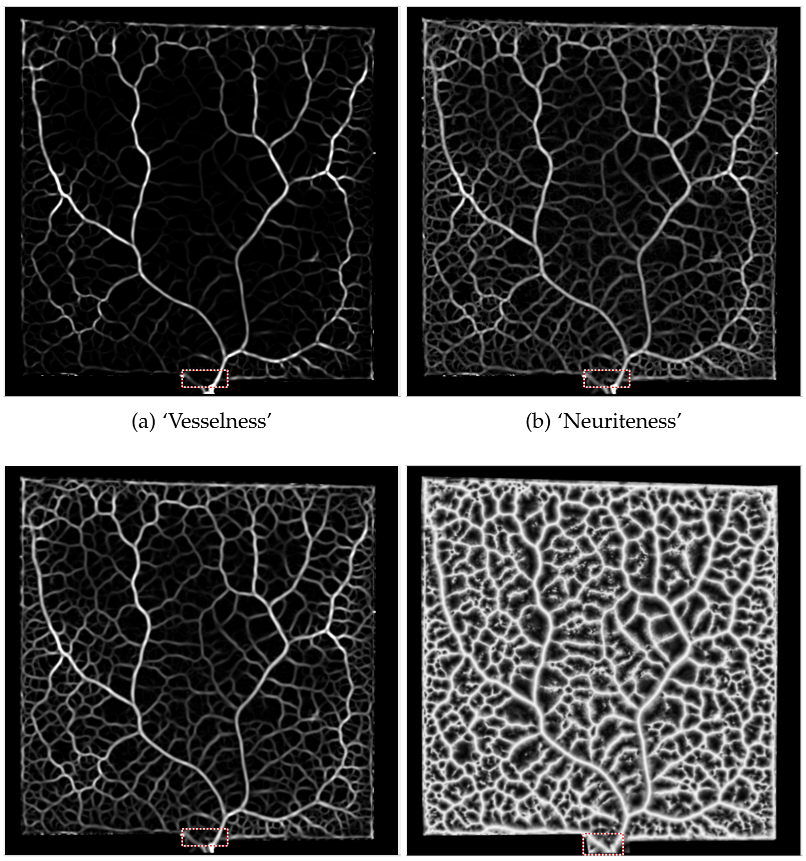

(a) 'Vesselness'
(b) 'Neuriteness'
(c) second-order anisotropic Gaussian (SOAGK)
(d) Phase congruency 'Feature Type'

Figure 5.8: Comparison of different multi-scale approaches to ridge enhancement. (a) the Frangi et al. (1998) 'Vesselness' algorithm based on the asymmetry of the isotropic second-order image gradient (note the 'gaps' that appear at the tubule junctions); (b) the Meijering et al. (2004) 'Neuriteness' algorithm; (c) the output of anisotropic second-order Gaussian filters according to Lopez-Molina et al. (2015); (d) the 'Feature Type' output of the phase congruency method according to Kovesi (1999, 2000). Note the strong edge response irrespective of the original tubule intensity;

In each case the number of scales is initially set by the $\#scales$ parameter, determined from the ratio of $FWHM_{max}/FWHM_{min}$, with a minimum bound of 3. However, this can be over-ridden using the adjacent text-box. This may be necessary to ensure that larger features are also processed prior to segmentation. The default number of orientations is set at 6. In the case of the SOAGK filters, the anisotropy is set at 1.1.

There is an additional **GF** check-box to apply a subsequent *'Guided'* filter to the enhanced image.

5.10 Skeletonization

The aim of the skeletonization step is to convert the enhanced image (Fig. 5.8 (a-d)) to a one-pixel wide skeleton along the centre-line of the tubule ridges. This is only an approximation to the true centre-line due to the pixel discretisation errors. There are two main approaches that can be used, namely 'hysteresis thresholding', that uses intensity information and some degree of pixel connectivity to provide an initial binary image. This then has to be thinned to give a single pixel skeleton. Alternatively, 'watershed thresholding' which follows connected ridges, irrespective of the absolute intensity and automatically generates a single-pixel wide skeleton.

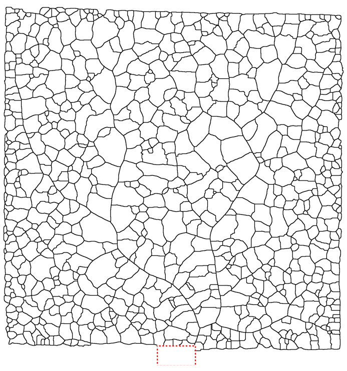


Figure 5.9: Skeleton following watershed thresholding of the 'Feature Type' image

The watershed is better at segmenting the centre-line of the ridges and can handle variations in intensity well. However, it does not include any tubules that have a free end, and has a tendency to over-segment regions with noise. Over-segmentation can be avoided by including additional an h-minimum filter (**hmin**), that suppress regions with small intensity fluctuations below the threshold set by the **hmin** text-box. This also helps to impose local minima to ensure adjacent structures are resolved.

Hysteresis thresholding starts with seed pixels above the upper threshold, and then propagates the initial segmentation as long as pixels remain above the lower threshold. The resulting binary image is then thinned to give the single-pixel skeleton. The value of the lower threshold is critical - too high and the network becomes disconnected; too low and large blocks of the image are included in the resultant binary image that may fuse separate tubules into a single object. When this block is thinned, the skeleton does not map onto the ridge centre-lines. In addition, as the thinning process is not guided by the intensities in the enhanced image, the skeleton does not necessarily converge on the expected pattern at junction points. These various options are selected from the drop down menu as follows:

- *'Hysteresis'* : applies hysteresis thresholding using a lower threshold set by the adjacent textbox. The upper threshold used to define the seed points is automatically set as $\textit{lower threshold} + 0.2$.
- *'Watershed'* : applies a watershed segmentation and then extracts the watershed lines as the pixel skeleton.
- *'Hist + hmin'* : applies an h-minimum transform to smooth out regions with low fluctuations in intensity, but then sets these regions as local minima to ensure that the surrounding ridges will be segmented individually and do not spread into the basin, even if the absolute values are above the lower hysteresis threshold.
- *'WS + hmin'* : uses the same h-minimum transform and imposes local minima before the watershed operation, to prevent over-segmentation of irrelevant ridges arising from noise within the

basins.

5.11 *Modifying the skeleton to accommodate features*

The initial pixel skeleton will include both veins and plasmodial sheets, even though a 'skeleton' is not necessarily a good representation of the larger features. Thus, if features have been segmented earlier in the sequence, these regions are punched out of the pixel skeleton so that they do not contribute to analysis of the vein network. During the analysis steps, described in (Chapter 6), the features are represented as a set of linear connections radiating out from the center to connect with the veins incident on the boundary. This ensures the overall connectivity of the network is retained.

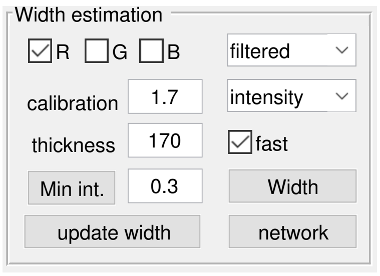
5.12 *Manual editing of the pixel skeleton*

If the automated methods to delineate the pixel skeleton are still incorrect, the skeleton can be edited manually using the **edit** button. This opens the binary editing program described in Chapter ?

5.13 *Estimation of the vein diameter*

Once the pixel-skeleton has been segmented satisfactorily, the first step in the analysis is to estimate the vein width. Although most *Physarum* images are single channel, there is an option to select any combination of channels to operate on using the **R**, **G**, and **B** checkboxes in the *Width estimation* panel (Fig. 5.10).

A number of different approaches are available that all provide some information on the vein diameter including:



[✓ R] G B
calibration 1.7
thickness 170
Min int. 0.3
filtered
intensity
[✓ fast]

Figure 5.10: Controls used to calculate the vein width

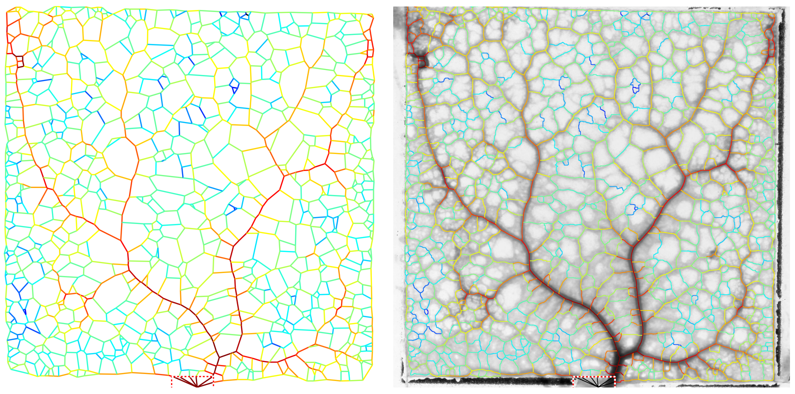
- *'Distance'* : This estimates the local FWHM from the original tubule intensity for each pixel in the skeleton. The peak height is estimated from the original intensity, whilst the distance is estimated from the distance transform of the pixel skeleton. The 50% threshold is estimated from where the pixel intensity falls below half the peak intensity, assuming a local background of zero.

- *'Integrated intensity granulometry'* : The intensity image is subject to a series of image openings (erosion followed by dilation) that successively remove structures as the size of the opening kernel exceeds the underlying object. This results in an intermediate (*x*,*y*,*s*) image, where *s* increases with the size of the disk-shaped kernel. The **fast** checkbox improves the speed of the granulometry calculation, although with less precision at low radii.

  The intensity of each pixel initially decreases slowly with *s* as the kernel samples more of the object, but then reduces dramatically once the boundary of the object is reached, and the kernel only samples the background. The integrated intensity under the granulometry curve is calculated and provides a measure of the width. The integral intensity cannot be directly related to the vein width without additional assumptions about the relationship between intensity and sampled volume. The **calibration** textbox can be used to input a conversion factor to convert relative intensity to width if it is possible to correlate between physical width and intensity, measured using the **Profile** controls (5.1), for example.

- *'Intensity'* The width of the veins and intervening plasmodial sheet at each pixel can be estimated using the Lambert-Beer law as $k \log_{10}(I_0/I)$, where I_0 is the incident light intensity, I the transmitted light intensity, and k a calibration coefficient based on the absorbance of a known thickness of plasmodium, measured from the image using the **Min int.** button, with the corresponding width in microns set in the **thickness** text box.

All three width estimates are calculated automatically by clicking the **width** button to allow comparison. However, subsequent operations will use the method set by the **width** drop down menu. The width is presented as a pseudo-colour coded version of the pixel skeleton that is scaled from $FWHM_{min}$ in blue to $FWHM_{max}$ in red (Fig. 5.11 (a)).


(a) Pseudo-color coded width

(b) Width skeleton superimposed on the filtered image

Figure 5.11: Pseudo-colour coded representation of the tubule width (a) ranging from blue ($FWHM_{min}$) to red ($FWHM_{max}$). (b) Result of blending the width image with the filtered image to visually inspect the performance of the overall segmentation and width analysis method

6
Network analysis

6.1 Conversion to a graph representation

The **network** button in the *Width estimation* panel (Fig. 5.10) will calculate convert the pixel-skeleton to a weighted graph representation. The nodes are defined at the junctions between the veins, or the end-points of free veins, which are connected by straight edges that preserve the topology of the network (Fig. 6.1). A number of metrics are associated with each edge including the Euclidean length of the underlying pixel skeleton, the average width calculated using each of the three methods, and the average width of the vein excluding the nodes (termed *'center width'*), which provides a more accurate measure of the tubule diameter itself.

If any features have been included, they are represented as a 'super-node' positioned at the intensity-weighted centroid position that is connected to all the veins that are incident on the feature boundary. Each of these veins is given an arbitrary value equal to the maximum width value.

Once the weighted graph has been calculated, it is overlaid on the colour-coded pixel skeleton with the nodes at the junctions connected by straight edges that match the colour-coding of the skeleton if the **graph** check-box is ticked.

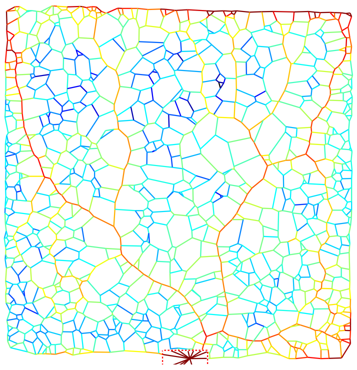


Figure 6.1: Conversion of the weighted pixel skeleton to a weighted graph. Junctions are represented as nodes linked by edges. The exit point of the arena is represented as a 'super-node' connected to all the incident veins on the boundary

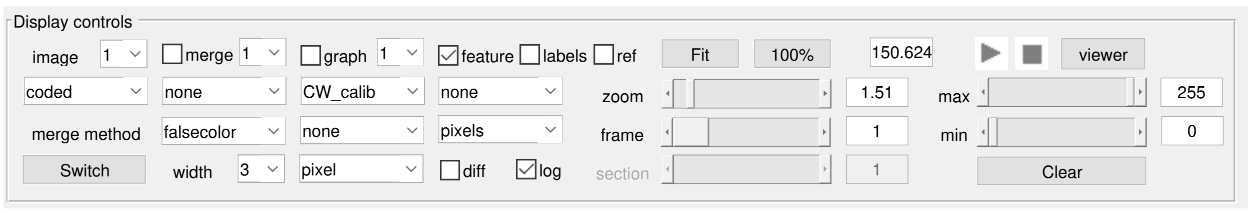

Display controls
image 1 | merge 1 | graph 1 | ✓ feature | labels | ref | Fit | 100% | 150.624 | viewer
coded | none | CW_calib | none | zoom | 1.51 | max | 255
merge method | falsecolor | none | pixels | frame | 1 | min | 0
Switch | width 3 | pixel | diff | ✓ log | section | 1 | Clear

Figure 6.2: Controls used to select the graph metric to overlay on the selected image

The **graph metric** displayed can be changed using the drop down menu (Fig. 6.2), and the colour placed on a log scale by ticking the **log** checkbox. The colour-coded limits run from the min and max of the selected metric. The **diff** check-box displays the difference in the selected metric between the next time-point and the current time-point.

The boundary of the features can also be highlighted using the

feature checkbox in the **Display controls** panel.

Text annotations for each edge can be displayed using the **labels** check-box (note: this adds a lot of information to the graph, which makes it difficult to read except at high zoom).

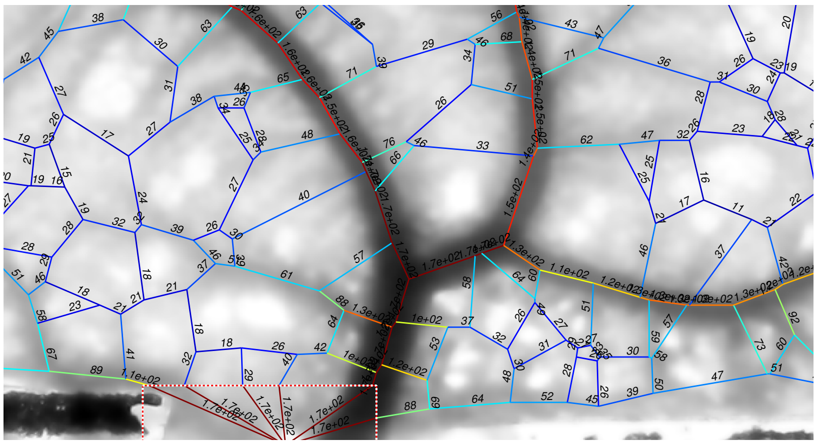


Figure 6.3: Graph of vein width overlaid on the image with edge labels included and the outline of the feature shown in a red-and-white dashed line.

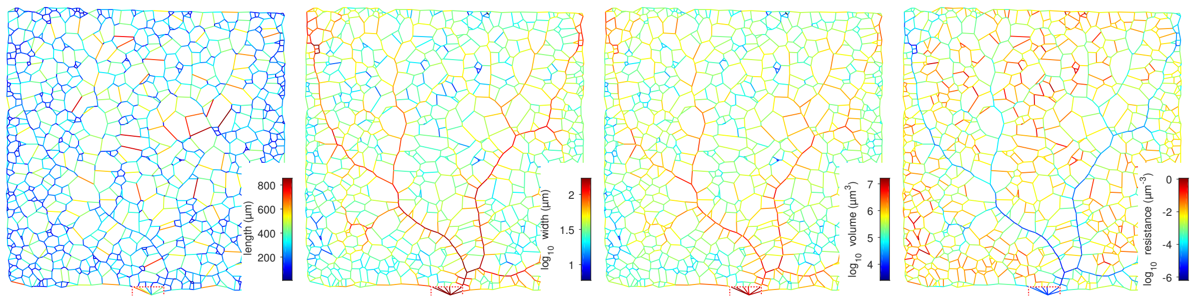
Alternatively, the information in the graph can be mapped back onto the pixel skeleton for each edge using the *pixel* option from the drop-down menu.

The graph overlay can be removed by clicking the **Clear** button.

6.2 *Graph edge metrics*

Each edge and node is associated with a vector of features including the average intensity ($\bar{I}_{ij}$), length (l_{ij}, Fig. 6.4(a)), and average width (w_{ij}), determined by excluding pixels that overlapped with any larger veins at the end of the edge, to give a centre-weighted estimate, more representative of the vein itself (Fig. 6.4(b)). The centre-weighted width is used to calculate the radius (r_{ij}), area ($a_{ij} = \pi r_{ij}^2$), volume ($v_{ij} = a_{ij} l_{ij}$, Fig. 6.4(c)), and predicted resistance to flow or drag ($\theta_{ij} = l_{ij}/r_{ij}^4$), Fig. 6.4(d)).

Figure 6.4: colour-coded maps for vein parameters



(a) Vein length (b) Vein width (c) Vein volume (d) Vein resistance

6.2.1 Graphical display

Each metric can be displayed as a *histogram* (Fig. 6.6) or *scatterplot* (Fig. 6.7) using the controls in the *display results* panels (Fig. 6.5). If the data is displayed as a histogram, the colormap reflects the time-point displayed, and can be changed using the **colormap** drop-down menu.

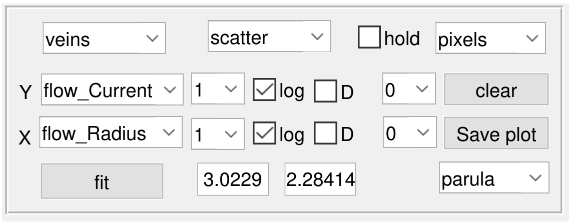


Figure 6.5: Controls used to display the results

The metrics available are selecting using the (*veins, nodes, polygons*) drop-down menu. The **X** and **Y** metrics are selected from the corresponding drop-down menu, for the channel shown.

Each variable can be plotted as a **log** using the check-box, or shown as a difference between the subsequent time-point and the current time-point using the **diff** check-box. The data is shown for the time-point selected by the **frame** slider in the *Display controls panel*. In addition, the data can be offset from the current time-point using the adjacent drop-down menu by +1 or -1 time-point. Plots can be overlaid using the **hold** check-box, and the axes scaled in *pixels, microns or mm* using the adjacent drop-down menu. The **fit** button will fit a linear regression to the data, and show the gradient and intercept in the adjacent text boxes (Fig. 6.7).

The **clear** button erases the plots. The **Save plot** saves a copy of the current plot.

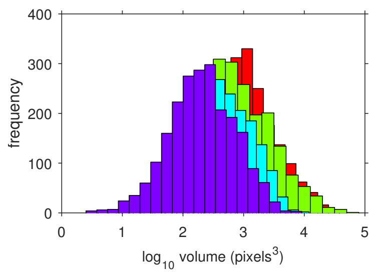


Figure 6.6: Scatter plot with fitted linear regression

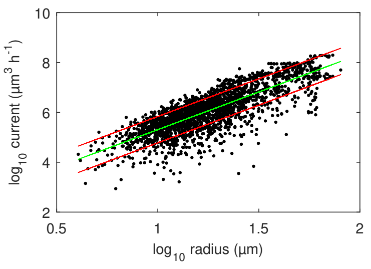


Figure 6.7: Scatter plot with fitted linear regression

6.3 Network metrics

Once the edge metrics have been calculated, they can be used to calculate a series of metrics for the entire graph (Fig. 6.8).

In each case, the control button calculates the specific parameter, and the adjacent check-box sets whether this will be calculated automatically. The **region** button assigns any volume of plasmodial sheet in the inter-vein regions to the nearest edge, within a limit set by the **max** text box. The **distance** button calculates the Euclidean distance from a reference point, positioned manually using the **set** button, or removed using the **clear** button. The reference might, for example, reflect the position of the exit point, denoted as node 0, (d_{i0}), (Fig. 6.9(a)). The position of the reference point can be displayed using the **ref** check-box in the *Display controls* panel.

The hydraulic accessibility (**accessibility**) to the exit point, measured as the path of minimum resistance, calculated using Dijkstra's algorithm (Fig. 6.9(b))

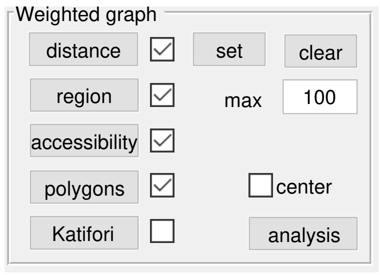
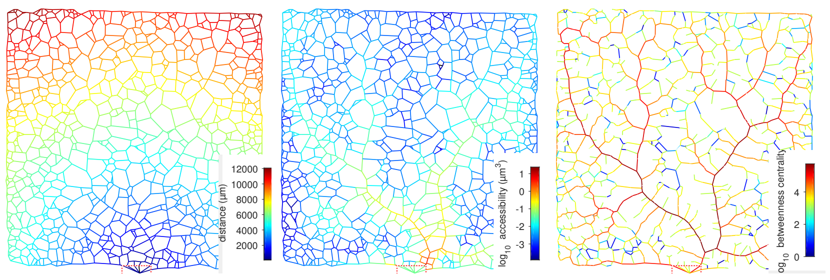


Figure 6.8: Controls to calculate network metrics



(a) Euclidean distance from the exit (b) Hydraulic accessibility (c) Betweenness centrality

Figure 6.9: color-coded maps for various network parameters

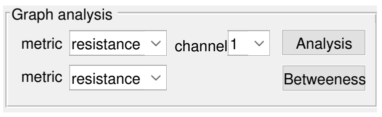


Figure 6.10: Controls to calculate graph metrics including Betweenness Centrality

Results for any shortest path calculations use the vein metric selected from *length, width, area, volume or resistance* in the *Graph analysis* panel. The edge betweenness centrality β_u for edge u was calculated as the proportion of all shortest paths (σ_{iuj}) between pairs of nodes i and j, that pass through u (Equation 6.1), to give a measure of the importance of a node or link to transport (Fig. 6.9(c)). Betweenness Centrality can be calculated using any one of the edge metrics (*length, width, area, or resistance*) using the drop-down menu in the *Graph analysis* panel (Fig. 6.10).

$$\beta_u = \sum_{ij} \frac{\sigma_{iuj}}{\sigma_{ij}} \tag{6.1}$$

6.4 *Summary statistics*

The edge morphology and network measures were used to calculate summary statistics for the network, including the route factor[1], defined as the average path length to the exit point divided by the Euclidean distance:

[1] M.T Gastner and MEJ Newman. Shape and efficiency in spatial distribution networks. *J. Stat. Mech.*, 2006: P01015, 2006

$$q = \frac{1}{N-1} \sum_{i=1}^{N-1} \frac{l_{i0}}{d_{i0}} \tag{6.2}$$

The global efficiency[2], defined as the mean reciprocal of the shortest paths, weighted by resistance, with the reciprocal for disconnected nodes defined as zero (Equation 6.3), along with the root efficiency (E_{root}) was calculated in a similar manner from the exit point to all other nodes.

[2] V. Latora and M. Marchiori. Efficient behavior of small-world networks. *Phys. Rev. Lett.*, 87:198701, 2001

$$E_{global} = \frac{1}{N(N-1)} \sum_{i \neq j \in G} \frac{1}{d_{ij}} \tag{6.3}$$

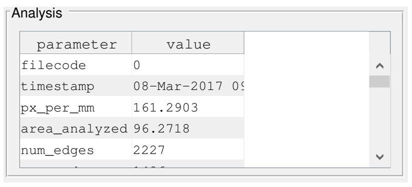
The α-coefficient[3] or meshedness[4] was used to measure the fraction of links present compared to a fully connected planar network, taking values from 0 to 1 to allow comparison of networks of different sizes:

[3] P Haggett and RJ Chorley. *Network Analysis in Geography (pp 74-76) Edward Arnold Publishers Ltd.* London, 1969

[4] J. Buhl, J. Gautrais, R.V Solé, P. Kuntz, J.-L. Valverde, S.and Deneubourg, and G. Theraulaz. Efficiency and robustness in ant networks of galleries. *Eu. Phys. J. B*, 42:123–129, 2004

$$\alpha = \frac{M - N + G}{2N - 5} \tag{6.4}$$

Results are presented in the analysis table (Fig. 6.11).



parameter	value
filecode	0
timestamp	08-Mar-2017 09
px_per_mm	161.2903
area_analyzed	96.2718
num_edges	2227

Figure 6.11: Results table for summary statistics from the graph analysis

6.4.1 Polygonal inter-vein statistics

The **polygons** button calculates a set of metrics for the polygonal inter-vein areas, including morphological properties such as *area, circularity, major axis length*, and the *maximum* and *mean distance* to the skeleton. Note: these metrics do not change between time-points if the pixel skeleton has been determined from a single template image.

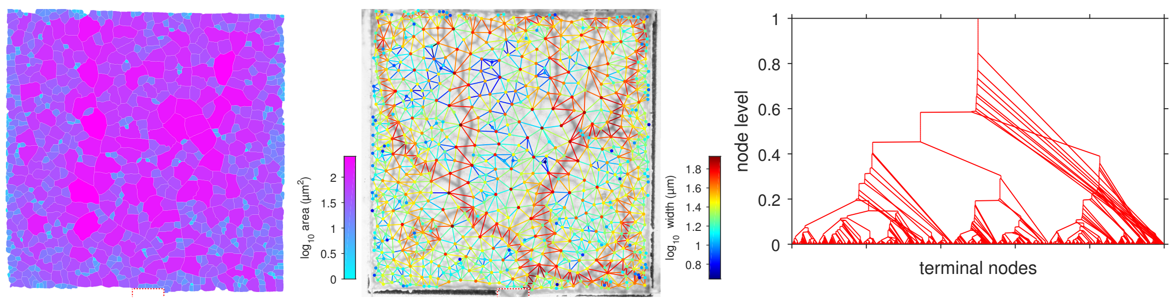
6.4.2 Hierarchical network decomposition

The overall structure of the *Physarum* network was investigated using hierarchical loop decomposition[5,6] of the dual-graph. In this implementation, the areas that were separated by the thinnest edge are fused first, then the areas separated by the second thinnest edge, and so on. Both the initial areas and the areas formed by fusions are represented as nodes in the dual-graph of the vein network, and these nodes are connected if the areas in question are formed by this hierarchical, fusion process. Note that by construction, the dual graph of the vein network is a binary tree (Fig. 6.12(c)).

[5] E. Katifori and M.O. Magnasco. Quantifying loopy network architectures. *PLoS One*, 7:e37994, 2012

[6] Y. Mileyko, H. Edelsbrunner, C.A. Price, and J.S. Weitz. Hierarchical ordering of reticular networks. *PLoS One*, 7:e36715, 2012

Figure 6.12: Hierarchical loop decomposition



(a) Polygon area (b) Dual-graph, color-coded by edge width (c) Predicted speed

Sequences of the area fusion events can be captured as a movie with 21 frames colour-coded to reflect the fractional size of the fused area at each time window (Fig. 6.13).

6.4.3 Analysis of predicted flows

Over short time intervals (hours), the spatial position of the tubes does not vary significantly, but their diameter changes in response to both short-term shuttle-streaming, and longer-term re-modelling of the network architecture as it exits the arena. It is therefore possible to track changes in each part of the network by re-applying the same network extraction routine to successive images using a

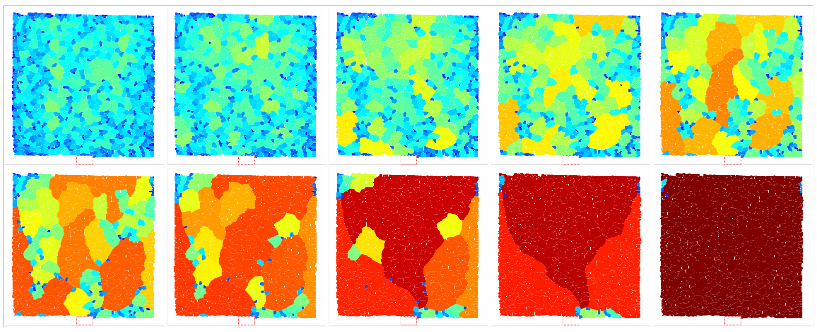


Figure 6.13: Sequence of polygon fusions as edges are removed in sequence

skeleton based on the median network present in the time window considered. Such time-dependent changes in tube volume can be used to predict the net mass flow through the tubular network[7]. Material flowing to or from the sheet-like regions between the veins or at the growing margin of the plasmodium is included in the flow models by allocating their change in volume to the nearest tube.

[7] L. L. M Heaton, E López, P. K Maini, M. D Fricker, and N. S Jones. Growth-induced mass flows in fungal networks. *Proc. R. Soc. B*, 277:3265–3274, 2010

The difference in vein volume between two time points is used to estimate the volumetric current flow through the network exiting the arena[8] (Fig. 6.15(a)). Veins that thin over time are the source of protoplasmic volume, while thickening veins are sinks.

[8] L. L. Heaton, E. Lopez, P. K. Maini, M. D. Fricker, and N. S. Jones. Advection, diffusion, and delivery over a network. *Phys Rev E*, 86:021905, 2012

If the *region* option is selected in the **model**, the volume change for plasmodia in the inter-vein region is also allocated to the nearest edge in the network, based on the Euclidean distance map (EDM) from the pixel skeleton. The net difference in volume for all veins and inter-vein regions was assumed to exit the arena to conserve mass. Assuming the volume of vein ij decreases from u_{ij} to v_{ij} over time t, the current flowing out of vein ij must be $(v_{ij} - u_{ij})/t$ greater than the current flowing into vein ij. As a simplifying assumption, half the net current is allocated to node i, and half to node j. To make an unbiased analysis of the relationship between current and changes in cross-sectional area, the current induced in vein ab by the changes in volume of the all the veins is calculated excluding vein ab itself. Thus, the net current flowing out of node i is defined as:

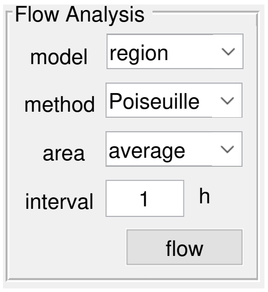

Flow Analysis
model: region
method: Poiseuille
area: average
interval: 1 h

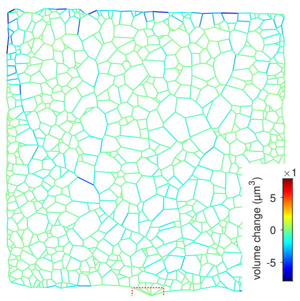
Figure 6.14: Flow analysis controls

$$q_i = \begin{cases} -\sum\limits_{j \neq i} q_j & \text{if node } j \text{ is the exit,} \\ \sum\limits_{ij \neq ab} \frac{u_{ij} - v_{ij}}{2t} & \text{otherwise.} \end{cases} \tag{6.5}$$

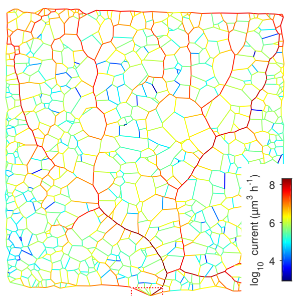
Note that the first sum is over the set of all nodes, while the second sum is over the set of all the veins ij directly connected to node i. The net current flowing out of each node and the conductance of each vein uniquely determine the pressure difference between any pair of nodes. Given a pressure drop ΔP between the end points of a vein of length l and radius r, the current Q through the vein follows equation (6.6), assuming Poiseuille flow is selected using the **method** drop-down menu, where ν is the dynamic viscosity of the cytoplasm:

$$Q = \frac{\pi r^4}{8 \nu l} \Delta P \tag{6.6}$$

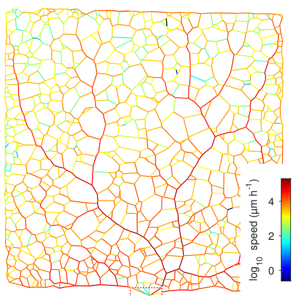
Given the net current at each node and the hydraulic conductance of each vein, the unique current in each vein that is consistent with equations (6.5) and (6.6) can be calculated (Fig. 6.15(b)). As the vein geometry is also know, the speed (Fig. 6.15(c)) and shear forces can be determined (Fig. 6.15(d)). The **area** drop-down menu allows the vein area used to reflect the vein radius for the *first* time point, the *last* time-point, or the *average*. The time **interval** is set in the appropriate text box.



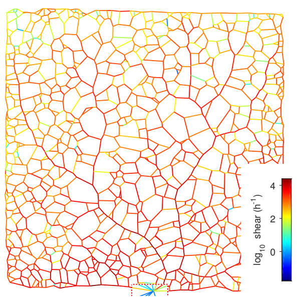
(a) Volume change



(b) Predicted current



(c) Predicted speed

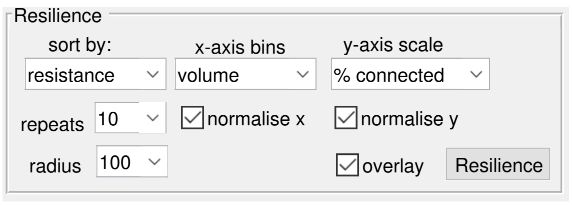


(d) Predicted shear

Figure 6.15: Flow analysis

6.5 *Network robustness*

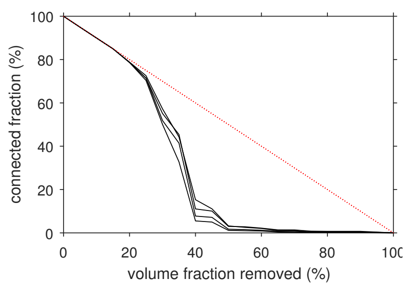
The robustness of the network is measured by removing edges in a particular order and measuring how much of the network remains as a single giant connected component or is still connected to the root node. Edges are ordered according to the **sort by** drop-down menu, with options that include *length, width, area, volume, resistance, random or spatial* (Fig. 6.16).


Resilience

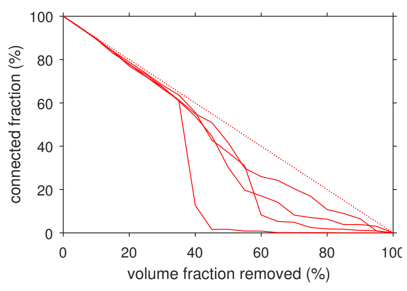
sort by: resistance
x-axis bins: volume
y-axis scale: % connected
repeats: 10
✓ normalise x
✓ normalise y
radius: 100
✓ overlay
Resilience

Figure 6.16: Robustness analysis controls

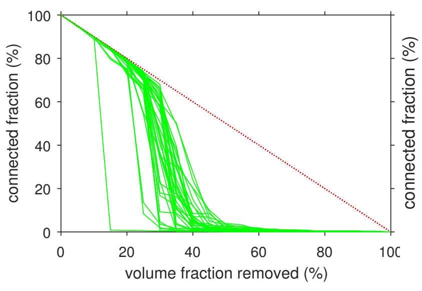
In the case of ordering by vein dimensions, there is one unique edge sequence for each network, with any ties resolved by the first occurrence, and so the robustness analysis is run once per time-point. If links are removed at *random*, there is the option to set the number of **repeats** from the drop-down menu to build up an ensemble profile of network robustness. The *spatial* option removes all edges within a **radius** in pixels, set by the drop-down menu, from randomly chosen initial pixel. The spatial attacks are also repeated multiple times set by the **repeats** option.



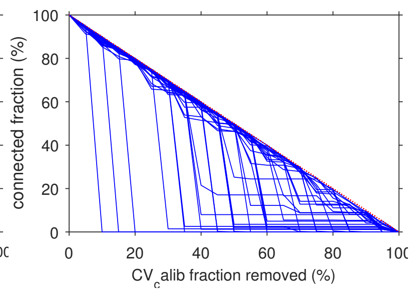
(a) Robustness ordered by length



(b) Robustness ordered by resistance



(c) Robustness based on random order

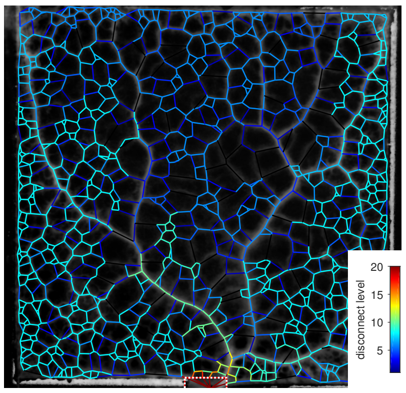


(d) Robustness based on spatial attack

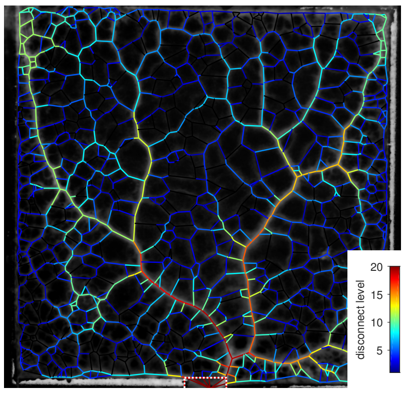
Figure 6.17: Robustness plots showing how much of the network remains connected to the root after a certain fraction of veins have been removed in order of length (a), resistance (b), at random (c) or following a spatial attack.

The results are displayed as a plot (Fig. 6.17), with the **x-axis** partitioned into 20 equal bins scaled by one of the edge morphology metrics *length, width, area, volume*, that can be expressed in absolute terms or normalised if the **normalise** check-box is ticked. The **y-axis scale** is expressed as the percentage of the volume still connected to the root (*% connected*), or the maximum size of the remaining component (*max size*), which can also be normalised if the **normalise** check-box is ticked. The **hold** check-box overlays successive plots.

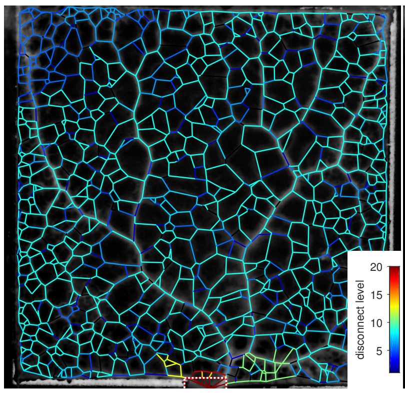
In addition, a colour-coded graph is displayed with edges coded either initial rank that each edge is assigned (*vein Rank*), or the level that the vein is actually disconnected (*vein Disconnect*), when the edge is no longer connected to the root (Fig. 6.18). For the *random* and *spatial*, the mean disconnect level (*vein meanDisconnect* or *spatial meanDisconnect*, respectively) summarises the results from multiple runs at each time-point.



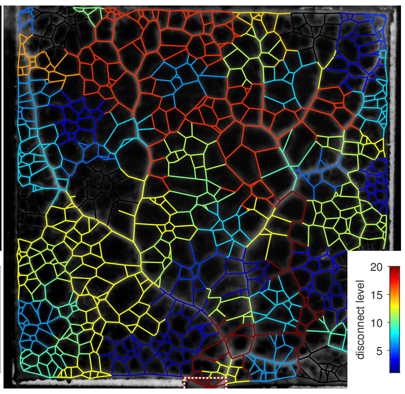
(a) Robustness ordered by length



(b) Robustness ordered by resistance



(c) Robustness based on random order



(d) Robustness based on spatial attack

Figure 6.18: Robustness analysis. Veins are color-coded to reflect the level at which they become dis-connected from the root

6.6 *Data output*

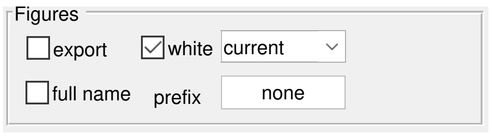


Figure 6.19: Figure output controls

The data and images from the analysis can be saved using the controls in the **Output** panel (Fig. 6.19). The **Save image** button saves a version of the image currently displayed, with all the annotations, in '*.png' and '*.pdf' format. The **Save data** button writes all the data on the veins, features, and polygonal regions to separate sheets in an Excel file. The **Panels** button saves a copy of the all the panels in the interface as '*.png' and '*pdf' files that can be used to tailor the illustrations in this manual to any specific application.

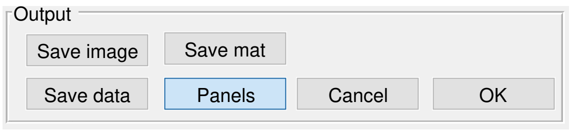
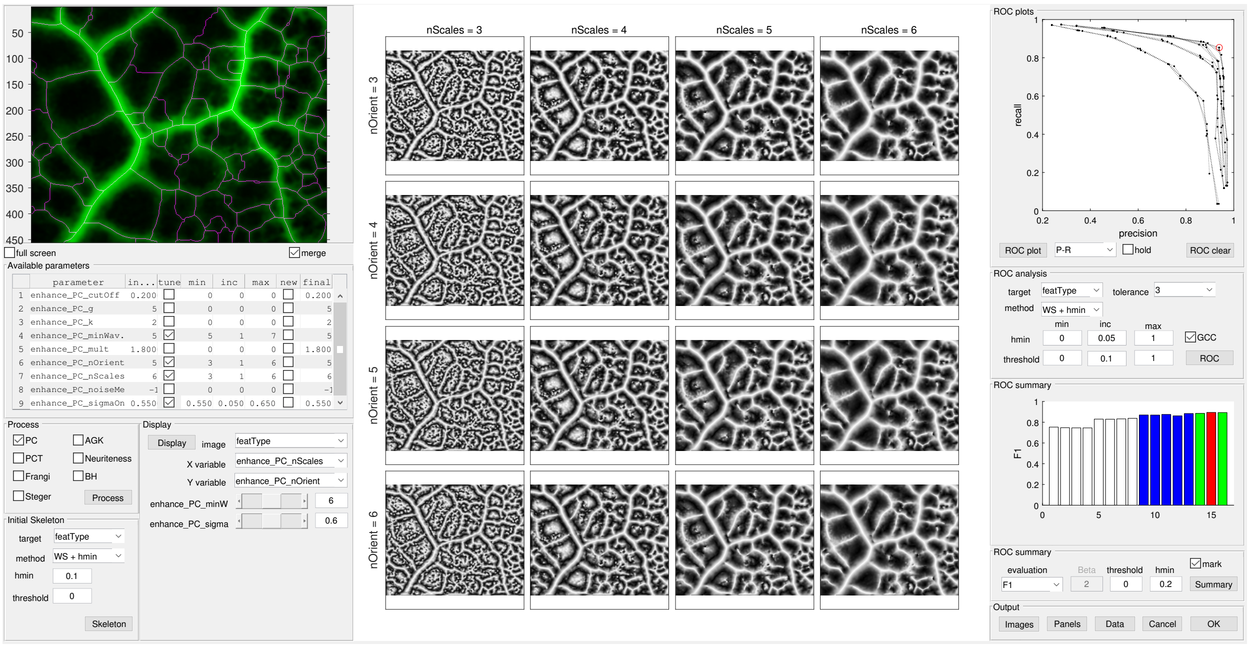


Figure 6.20: Output controls used to save data and images

7
Parameter selector



	parameter	in...	tune	min	inc	max	new	final
1	enhance_PC_cutOff	0.200	☐	0	0	0	☐	0.200
2	enhance_PC_g	5	☐	0	0	0	☐	5
3	enhance_PC_k	2	☐	0	0	0	☐	2
4	enhance_PC_minWav.	5	✓	5	1	7	☐	5
5	enhance_PC_mult	1.800	☐	0	0	0	☐	1.800
6	enhance_PC_nOrient	5	✓	3	1	6	☐	5
7	enhance_PC_nScales	6	✓	3	1	6	☐	6
8	enhance_PC_noiseMe	-1	☐	0	0	0	☐	-1
9	enhance_PC_sigmaOn	0.550	✓	0.550	0.050	0.650	☐	0.550

Display: image featType; X variable enhance_PC_nScales; Y variable enhance_PC_nOrient; enhance_PC_minW 6; enhance_PC_sigma 0.6

Initial Skeleton: target featType; method WS + hmin; hmin 0.1; threshold 0

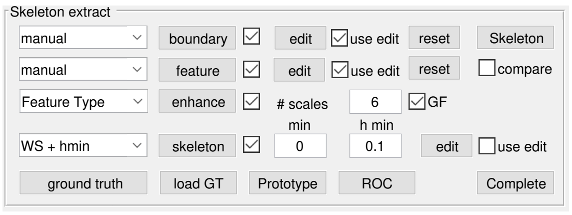
ROC analysis: target featType; tolerance 3; method WS + hmin; hmin min 0, inc 0.05, max 1; threshold min 0, inc 0.1, max 1; ✓ GCC

ROC summary: evaluation F1; Beta 2; threshold 0; hmin 0.2; ✓ mark

Figure 7.1: The GUI interface for the parameter selector program

7.1 Sensitivity analysis with factorial parameter combinations

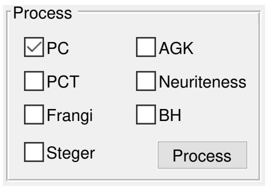
The parameter selector panel allows a factorial combination of parameters to be applied to a region-of-interest (ROI) from the image to compare with a manually-defined ground-truth. The program is called from the **Prototype** button in the *Skeleton extract* panel of the main interface (Fig. 7.2).



Skeleton extract: manual – boundary ✓; manual – feature ✓; Feature Type – enhance ✓, # scales 6, ✓ GF; WS + hmin – skeleton ✓, min 0, h min 0.1

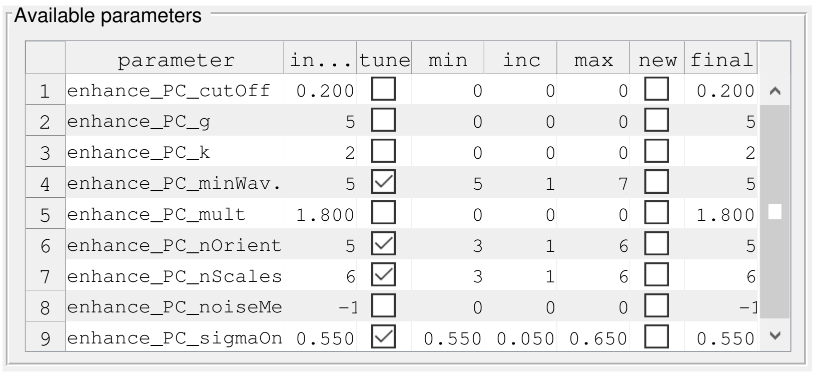
Figure 7.2: The **Prototype** button in the skeleton extract panel

The algorithm to apply to the ROI is chosen using the checkboxes in the *Process* panel. At the moment, only on algorithm should be selected at a time. In a later version, multiple choices are planned to aid comparison between methods. Once a method has been selected, the parameter options specific to that method are displayed in the *Available parameters* panel (Fig. 7.4).



Process	
✓ PC	☐ AGK
☐ PCT	☐ Neuriteness
☐ Frangi	☐ BH
☐ Steger	Process

Figure 7.3: Check-boxes to select the enhancement algorithm

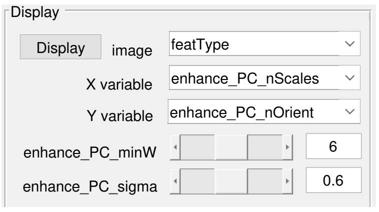


Available parameters

	parameter	in...	tune	min	inc	max	new	final
1	enhance_PC_cutOff	0.200	☐	0	0	0	☐	0.200
2	enhance_PC_g	5	☐	0	0	0	☐	5
3	enhance_PC_k	2	☐	0	0	0	☐	2
4	enhance_PC_minWav.	5	✓	5	1	7	☐	5
5	enhance_PC_mult	1.800	☐	0	0	0	☐	1.800
6	enhance_PC_nOrient	5	✓	3	1	6	☐	5
7	enhance_PC_nScales	6	✓	3	1	6	☐	6
8	enhance_PC_noiseMe	-1	☐	0	0	0	☐	-1
9	enhance_PC_sigmaOn	0.550	✓	0.550	0.050	0.650	☐	0.550

Figure 7.4: Table showing the parameters that can be tuned for the selected algorithm

the current value for each parameter is shown in the **initial** column, and cannot be edited. A minimum of two parameters must be selected using the **tune** check-boxes, and the range to explore set using the minimum value (**min**), the increment (**inc**), and the maximum value **max**. These two parameters are automatically set as the **X variable** and **Y variable** in the *Display* panel drop-down menus (Fig. 7.5). Each additional parameter selected for tuning, is displayed below the drop-down menus as a slider bar running between **Min** and **Max**. these slider are used to scroll through the images for display. In principle every parameter can be tuned in this way, but the number of combinations will increase very rapidly, so it is wise to explore a more limited sub-set of parameters and values in the first instance, to progressively hone in on the best combination.



Display

Display	image	featType
	X variable	enhance_PC_nScales
	Y variable	enhance_PC_nOrient
	enhance_PC_minW	6
	enhance_PC_sigma	0.6

Figure 7.5: Controls to display a sub-set of the processed images

If a value for a parameter has already been optimised, the **new** check-box can be ticked in the *Available parameters* table, and the value entered in the **final** column.

When the **Process** button is clicked, the ROI will be processed using the factorial combination of the parameter range set.

Once the ROI has been processed, the results are displayed as a 2-D grid (Fig. 7.6), labelled with the values for the first two parameters selected for tuning in the *Display* panel. Different parameters can be selected for the display grid using the drop-down menus and sliders. The type of image displayed can be selected using the **image** drop-down menu in the *Display* panel.

At the end of the processing step, the images are also automatically segmented using the values set in the *Initial skeleton* panel.

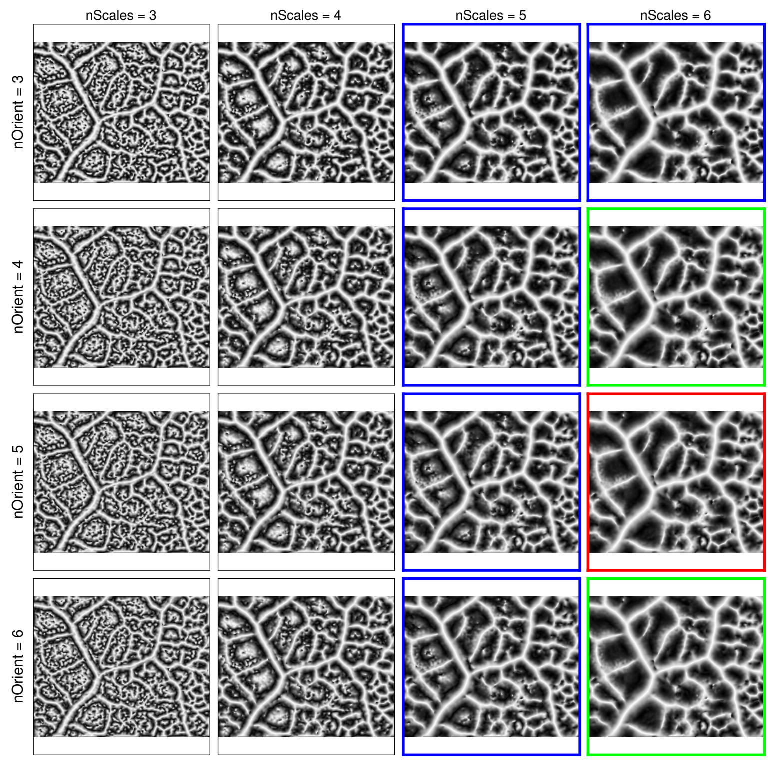


Figure 7.6: Grid showing results for the first two parameters in the sensitivity analysis

These follow the same options available in the main interface and process the enhanced image to a single-pixel wide skeleton. The skeleton can be displayed using the **image** drop-down menu. The skeletonisation parameters can be changed and just the skeletonisation step re-run on the set of enhanced images using the **Skeleton** button.

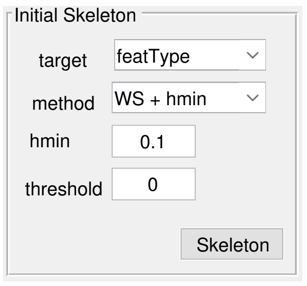

Initial Skeleton
target: featType
method: WS + hmin
hmin: 0.1
threshold: 0

Figure 7.7: Initial segmentation controls

7.2 *Comparison with a ground-truth skeleton*

Pixel skeletons for each ridge enhancement-skeletonisation combination can be scored against a manually digitised ground-truth (Fig. 1.2G), within a chosen tolerance, typically around half the minimum vein width, to classify each pixel as a true positive (TP), true negative (TN), false positive (FP), or false negative (FN). It is important that the ground-truth image is loaded into the main interface using the **Load GT** button (Fig. 7.8) prior to calling the *Parameter selector* GUI, so that the same ROI is extracted for comparison.

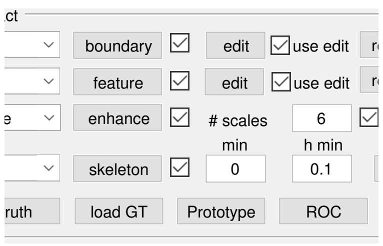

scales: 6
min: 0
h min: 0.1

Figure 7.8: The ***Load GT*** *button in the skeleton extract panel*

The target enhanced image for comparison is selected using the **Target** drop-down menu in the *ROC analysis* panel, along with the segmentation **method**, and the **tolerance** (in pixels) that will still be considered as a 'true positive' relative to the Ground-Truth skeleton.

If the *WS + hmin* option is chosen, the **hmin** parameter is applied

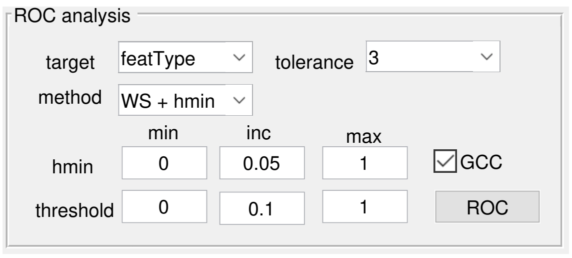


Figure 7.9: ROC analysis controls

over a range of values given by the **min**, increment (**inc**), and **max** text boxes. If *hysteresis* thresholding is selected, the **threshold** options are enabled, and can also be set to run from **min** to **max**, with a given increment (**inc**). The analysis can be restricted to the giant connected component (GCC), using the **GCC** checkbox, as a crude test of connectivity in the resultant skeleton.

The **ROC** button initiates the analysis, and results are presented as a *ROC curve*, *P-R plot*, *z-score* or *histogram*, depending on the setting of the **ROC plot** drop-down menu in the *ROC plots* panel.

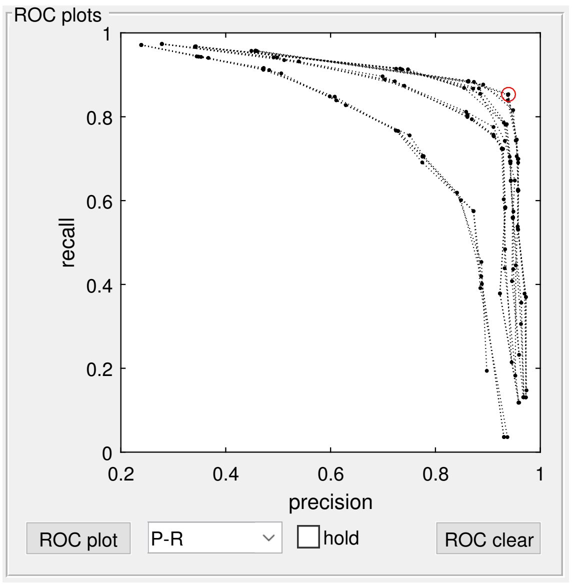


Figure 7.10: graphical output for a PRC curve

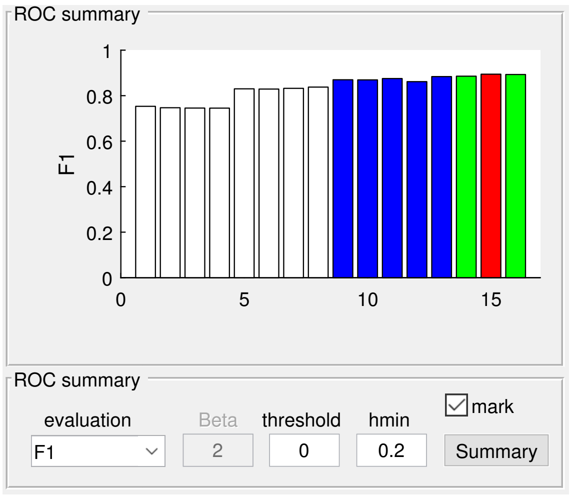
Precision-Recall (PRC) analysis, where Precision was calculated as TP/(TP+FP), and Recall as TP/(TP+FN) is used in preference to Receiver Operating Characteristic (ROC) plots, as the former are better suited to imbalanced datasets [1], when the number of true negatives (TNs) from the background is expected to be much greater than the true positives (TPs) from the skeleton.

[1] T. Saito and M. Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PLOS One*, 10:e0118432, 2015

Results for each run of the *hmin* or *threshold* parameter are connected by a dotted line. It should be noted that *hmin* and *threshold* do not behave quite as a conventional tuneable threshold in ROC or PRC analysis, as the output (a connected skeleton) does not necessarily correlate linearly with increasing or decreasing values of the parameter. For example, as the threshold is lowered, one might expect fewer FNs, more TPs, but also more FPs. However, if the threshold merges two adjacent ridges, the resultant skeleton my be incorrectly positioned mid-way between both when the binary image is skeletonised. This give a decrease in FPs and and increase in FNs and FPs.

The **ROC clear** button erases the plot, whilst the **hold** check-box allows multiple analyses to be superimposed.

In addition to the PRC or ROC plot, the best performance can be estimated by a summary statistic selected from the *ROC summary* panel (Fig. 7.11).



evaluation: F1, Beta: 2, threshold: 0, hmin: 0.2, ✓ mark

Figure 7.11: ROC summary controls and color-coded plot for the F_1 parameter

The best performance is assessed from the highest **evaluation** score, using a metric selected from the drop-down menu from *dprime*, F_1, *MCC*, F_2 or F_β. The F_β score represents a harmonic mean of recall and precision, where F_β, can be tuned to weight recall ($\beta = 2$) or precision ($\beta = 0.5$) more according to equation 7.1:

$$F_\beta = (1+\beta^2)\frac{\text{Precision} \times \text{Recall}}{(\beta^2 \times \text{Precision}) + \text{Recall}} \tag{7.1}$$

The maximum score is colour-coded red on the histogram output, values within 98% coloured green and 95% coloured blue. If the **mark** check-box is ticked, the same colour coding is applied to the border of the corresponding image in the image grid. In addition, the best score is highlighted on the PRC or ROC plot with a red circle.

If multiple parameters have been selected for the sensitivity analysis, the image with the best response may not be visible in the current grid. In this case, the image can be revealed by scrolling through the different parameter values using the sliders in the *Display* window. If the user is happy with the 'best' parameter combination, clicking on the preferred image will update the **final** parameter combination in the *Available parameters* table.

The *ROC* option in the *Display* panel, allows visualisation of the output of the ROC analysis in terms of TPs (green), FNs (red) and FPs (blue), for each of the parameter combinations visible (Fig. 7.12).

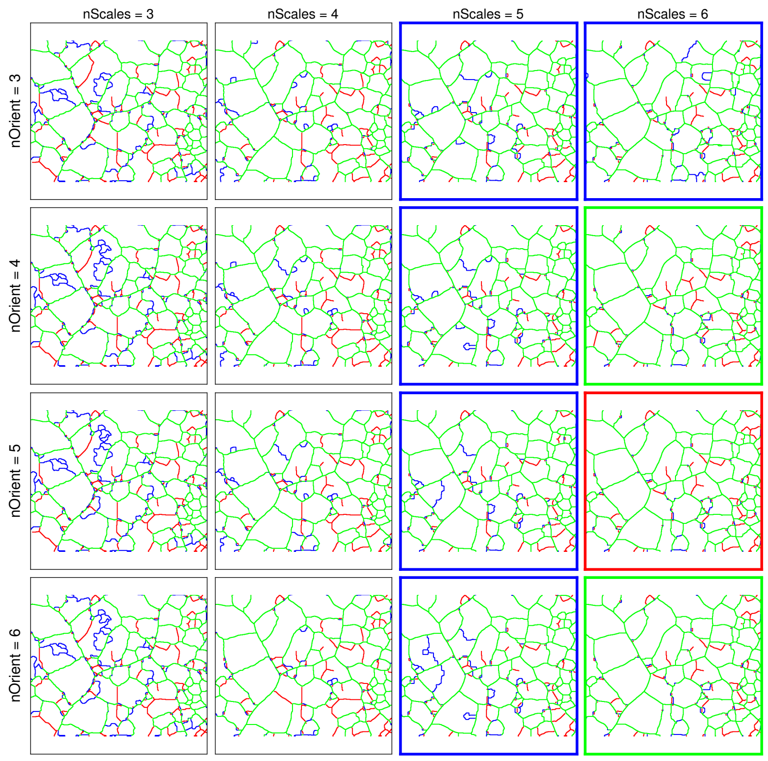


Figure 7.12: Grid showing TP (green), FN (red) and FP (blue) for skeletons derived from the first two parameters in the sensitivity analysis

7.3 *Output*

The image grid can be save using the **Images** button in the *Output* panel, whilst the complete set of summary statistics for each parameter combination can be saved to an Excel spreadsheet using the **Data** button. The **Panels** button saves each panel as a *.png and *.pdf image for inclusion in the manual.

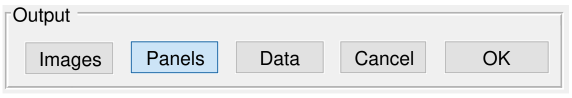


Figure 7.13: Output controls

When the *Parameter selector* GUI is closed the optimised parameters are returned to the main program. At this stage they are not automatically used to update the parameter settings to avoid accidentally corrupting the current parameters, thus if changes have been made, they need to be explicitly changed using the **edit** button in the *Param* panel.

8
Binary editing

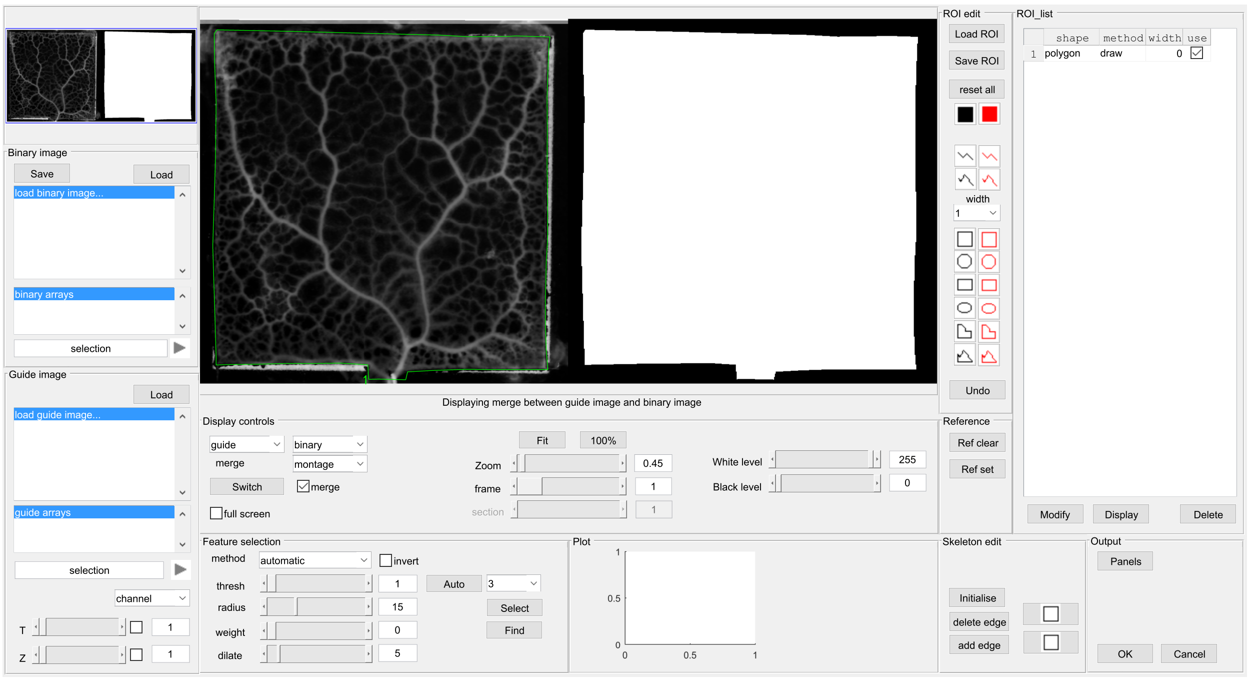


Figure 8.1: The GUI interface to edit binary images

8.1 Loading the images

Binary images are used to mask the boundary of the arena, delineate the food resources or other features, and represent the single-pixel wide skeleton. Whilst each of these can be generated automatically, it is often the case that they need to be edited manually. The *Binary editing* panel has tools to achieve this.

If the *Binary editing* window has been called from the *Physarum network* interface, by clicking the **edit** mask button for example, the template image and the current mask (if any) will automatically be displayed as a green-magenta merge when the window opens (Fig. 8.2).

The order of the merged image can be changed using the **Switch** button, the type of merge using the drop-down menu, or removed by un-checking the **merge** check-box (Fig. 8.3). The other display

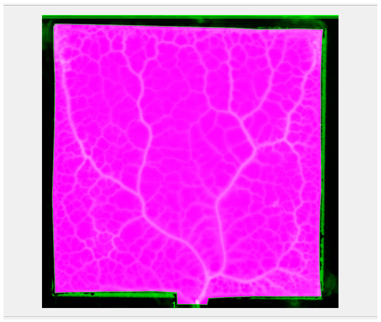


Figure 8.2: Main display window showing a merge between the guide image (green) and binary mask (magenta).

controls can be used to change the zoom, adjust the contrast, or, in multi-dimensional images, scroll through the sections and frames.

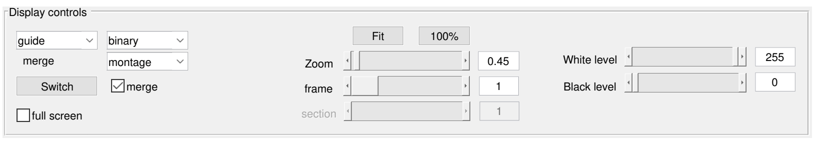


Figure 8.3: Main display window and controls to select the images to display, the type of merge, zoom and contrast.

In some instances, it may be desirable to load a previously constructed binary image, or a different guide image stored on disk. Available images (stored as MatLab arrays) can be loaded into the *Binary image* listbox (Fig. 8.4) using the **Load** button. The appropriate binary image from the arrays present in the matfile is selected by a single click, and displayed in the **selection** box. If the binary image is part of a structure within the matfile, it will be displayed in the **binary arrays** and has to be selected from this list-box. The adjacent right arrow is used to import it into the interface.

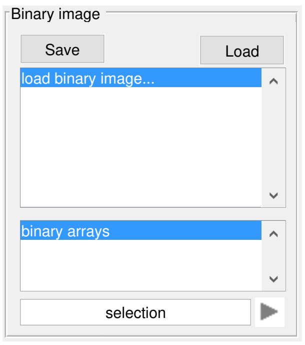


Figure 8.4: Controls to select and load a binary image (as a Matlab array).

In a similar manner, a separate guide image can be loaded from a MatLab file on disk using the controls in the *Guide image* panel. If the guide image is part of a MatLab structure, the names of the available array are shown in the **guide arrays** list-box, and has to be selected from here.

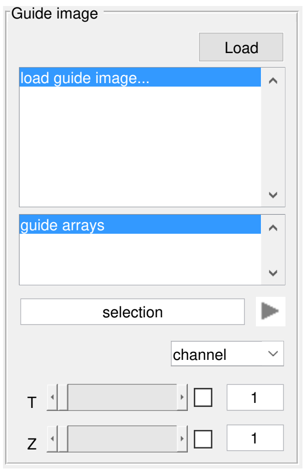


Figure 8.5: Controls to select and load a guide image (as a Matlab array).

The **channel** drop-down menu is used to choose the channel to import, whilst the **T** and **Z** sliders are used to select the frame and section, respectively. The check-boxes adjacent to the sliders give a maximum projection along that dimension. All changes are displayed in the main figure window. Once the most suitable combination has been chosen, the image is imported using the right arrow.

8.2 *Automatic feature selection*

The *Feature selection* panel is used to try to automatically segment the features of interest (Fig. 8.6).

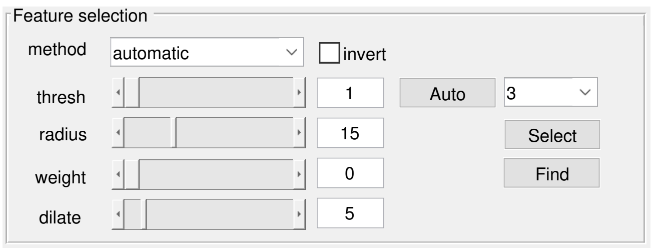


Figure 8.6: Controls to segment features in the image

The **method** drop-down menu provides access to a number of different thresholding strategies including:

- *manual:* The user chooses a threshold using the **threshold** slider.
- *automatic:* Automatically chooses a threshold when the **Auto** button is pressed using Otsu's criterion[1] for multiple thresholds,

[1] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Trans. Systems, Man, Cyber.*, 9:62–66, 1979

of which the lowest one is selected. The number of partitions is selected by the adjacent drop-down menu. A histogram of the image intensities is shown in the *plot* panel, with the position of the threshold(s) shown in green (Fig. 8.7).

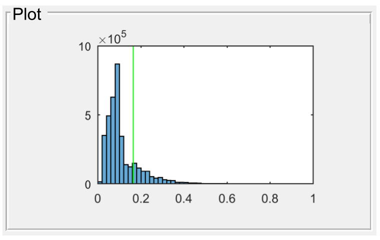


Figure 8.7: Histogram of piel intensities overlaid with the segmentation threshold

- *opening:* Applies a grayscale opening to the image, with the radius determined by the **radius** slider. The image is segmented automatically using the lowest threshold from multi-level threshold menu;
- *tophat:* Applies a tophat filter with the radius determined by the **radius** slider. The image is segmented automatically using the lowest threshold from multi-level threshold menu;
- *active contour:* Applies a grayscale opening to the image, with the radius determined by the **radius** slider. The opened image is then matched to the underlying objects using an active contour algorithm.
- *local mean:* The local mean is calculated using a circular filter with the radius set by the **radius** slider and used as a local threshold. An additional constant can be subtracted from the threshold using the **weight** slider.
- *local median:* The local median is calculated using a circular filter with the radius set by the **radius** slider and used as a local threshold. An additional constant can be subtracted from the threshold using the **weight** slider.
- *midgrey:* The mid-grey threshold is calculated as the average of the local maximum and minimum, calculated using morphological closing and opening operations, respectively, with the radius set by the **radius** slider. An additional offset k can be set using the **weight** slider.

$$T(x,y) = \frac{max + min}{2} - k \tag{8.1}$$

- *Bernsen*[2]*:* The method calculates the local contrast and mid-grey values in a similar manner to the mid-grey algorithm from morphological operations. If the local contrast is above the contrast value k set by the **weight** slider, the threshold is set at the local mid-grey value. If the local contrast is below the contrast threshold, the pixel is classified as part of the object if the mid-grey value is above 0.5, or background if below.
- *Niblack*[3]*:* The local threshold ($T(x,y)$) is calculated from the local mean ($m(x,y)$) and the local standard deviation ($s(x,y)$), with the radius set by the **radius** slider, weighted by a factor (k) set using the **weight** slider.

$$T(x,y) = m(x,y) + k * s(x,y) \tag{8.2}$$

where k is typically -0.2.

[2]

[3]

- *Sauvola*[4]*:* The local threshold ($T(x,y)$) is calculated in a similar manner to the Niblack algorithm from the local mean ($m(x,y)$) and local standard deviation ($s(x,y)$), with the radius set by the **radius** slider, but the weighting is calculated as:

$$T(x,y) = m(x,y)\left[1 + k\left(\frac{s(x,y)}{R} - 1\right)\right] \tag{8.3}$$

where $R = max(s)$ and k takes small positive values, typically in the range 0.2- 0.5, set by the **weight** slider.

4

8.3 *Manual Editing*

If automatic segmentation is not possible, or the results need further modification, The *ROI edit* controls can be used (Fig. 8.8). Each icon depicts the type of ROI that can be drawn in black, or erased in red. The sequence of ROIs can be saved (**Save ROI)**) and reloaded (Load ROI). In addition, any binary image created with the *Binary edit* called from another program is imported with the existing list of ROIs.

The **reset all** button clears all the ROIs and the segmented image. The solid black icon set the ROI to fill the image, whilst the solid red icon erases the full screen. These should only be used as the first ROI in the series. Each ROI is numbered and listed in the adjacent *ROI list* panel (Fig. 8.9), which indicates the *shape* of the ROI, the *method* (*draw* or *erase*), the line width (only relevant for polylines and freehand lines), and whether the ROI should be used (**use** check-box).

Examples of the different ROI shapes are shown in Fig. 8.3.

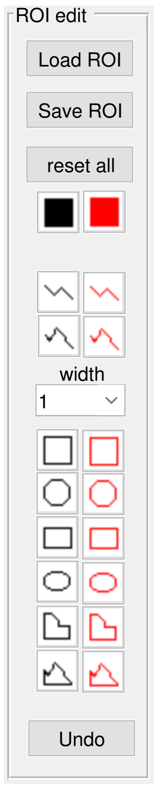
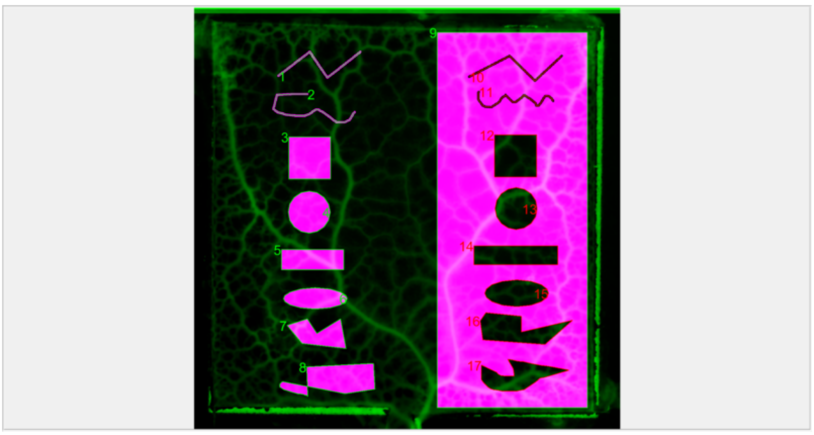
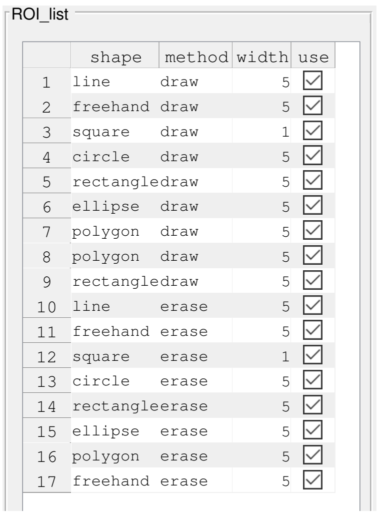


Figure 8.8: Manual editing controls



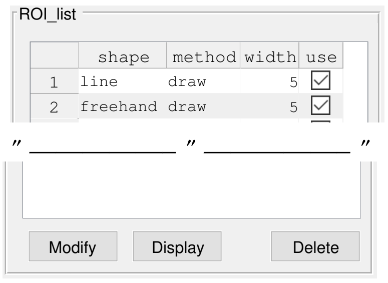

ROI_list

	shape	method	width	use
1	line	draw	5	✓
2	freehand	draw	5	✓
3	square	draw	1	✓
4	circle	draw	5	✓
5	rectangle	draw	5	✓
6	ellipse	draw	5	✓
7	polygon	draw	5	✓
8	polygon	draw	5	✓
9	rectangle	draw	5	✓
10	line	erase	5	✓
11	freehand	erase	5	✓
12	square	erase	1	✓
13	circle	erase	5	✓
14	rectangle	erase	5	✓
15	ellipse	erase	5	✓
16	polygon	erase	5	✓
17	freehand	erase	5	✓

Figure 8.9: List of current ROIs

For all the geometric shapes, a vertex is added with each left mouse click, whilst a double-click closes the shape. each vertex can be re-positioned by hovering over the vertex until a black circle appears and then dragging the circle to the new position. Additional vertices can be added along the lines using the **A** button on the keyboard. Once the shape is complete, it is plotted in green for an included region and red for an erased region, along with a unique label matching the row in the *ROI list*. In addition, the corresponding binary image is shown overlaid in magenta. For the polyline and freehand line buttons, the line width can be chosen in pixels from the drop-down menu, or adjusted in the *ROI list* box.

ROIs can be altered or deleted using the **Modify** button or **Delete** button, respectively, in the *ROI list* panel (Fig. **??**). The **Display** button can be used to show the ROI overlay and binary image at any time.



	shape	method	width	use
1	line	draw	5	✓
2	freehand	draw	5	✓

Figure 8.10: ROI list controls

8.4 Output

The **OK** button returns the adjusted binary iage to the calling program, along with the list of ROIs and their vertex co-ordinates. The **Cancel** button exits without saving any information. The **Panel** button saves a copy of each panel in *.png and *.pdf format for inclusion in the manual.

In addition, the binary image can be saved independently as a matlab file using the **Save** button in the *Load binary* panel (Fig. ref).

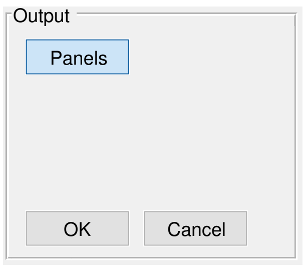


Figure 8.11: Output controls

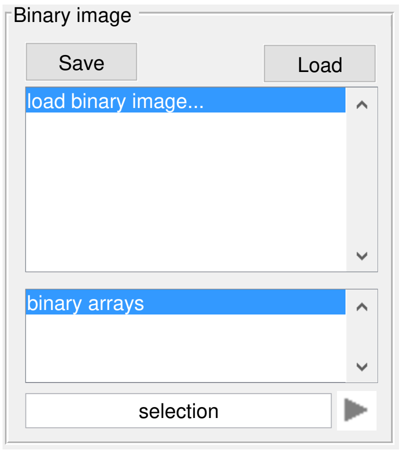
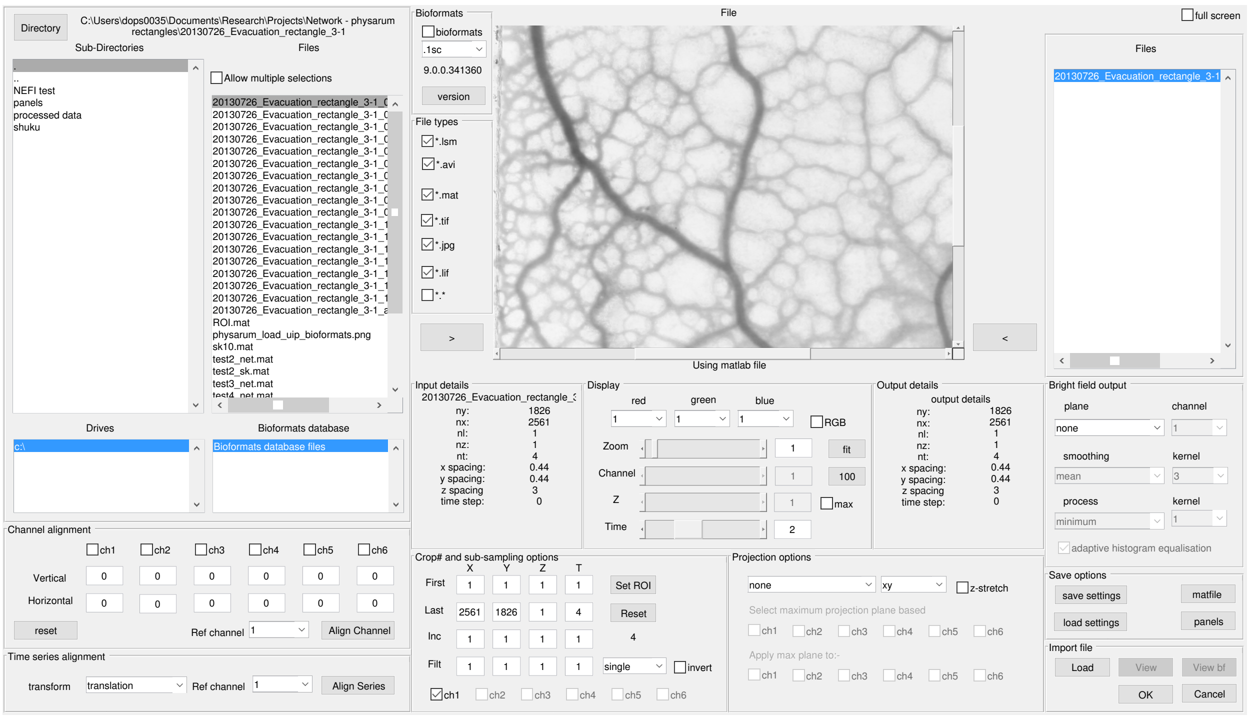


Figure 8.12: Option to save the adjusted binary image as a matfile

9
Advanced File Import


C:\Users\dops0035\Documents\Research\Projects\Network - physarum rectangles\20130726_Evacuation_rectangle_3-1

Sub-Directories: . .. NEFI test panels processed data shuku

Files: 20130726_Evacuation_rectangle_3-1_((×18) ROI.mat physarum_load_uip_bioformats.png sk10.mat test2_net.mat test2_sk.mat test3_net.mat test4_net.mat

Drives: c:\

Bioformats database: Bioformats database files

Bioformats: .1sc 9.0.0.341360

File types: [✓] *.lsm [✓] *.avi [✓] *.mat [✓] *.tif [✓] *.jpg [✓] *.lif [ ] *.*

Using matlab file

Files: 20130726_Evacuation_rectangle_3-1

Input details: 20130726_Evacuation_rectangle_3

```
ny:          1826
nx:          2561
nl:          1
nz:          1
nt:          4
x spacing:   0.44
y spacing:   0.44
z spacing    3
time step:   0
```

Display: red 1, green 1, blue 1; Zoom 1; Channel 1; Z 1; Time 2; 100

Output details

```
output details
ny:          1826
nx:          2561
nl:          1
nz:          1
nt:          4
x spacing:   0.44
y spacing:   0.44
z spacing    3
time step:   0
```

Bright field output: plane none; channel 1; smoothing mean; kernel 3; process minimum; kernel 1; [✓] adaptive histogram equalisation

Channel alignment

	ch1	ch2	ch3	ch4	ch5	ch6
Vertical	0	0	0	0	0	0
Horizontal	0	0	0	0	0	0

Ref channel 1

Time series alignment: transform translation; Ref channel 1

Crop# and sub-sampling options

	X	Y	Z	T
First	1	1	1	1
Last	2561	1826	1	4
Inc	1	1	1	1
Filt	1	1	1	1

4; single; [✓] ch1

Projection options: none; xy; Select maximum projection plane based; Apply max plane to:-

9.1 Introduction

This software was designed to import image stacks into Matlab. The import options allow some channel registration, time-series alignment, sub-sampling, smoothing and projection. A variety of different formats can be handlesd using the Bio-formats program (Linkert et al. 2010[1]):

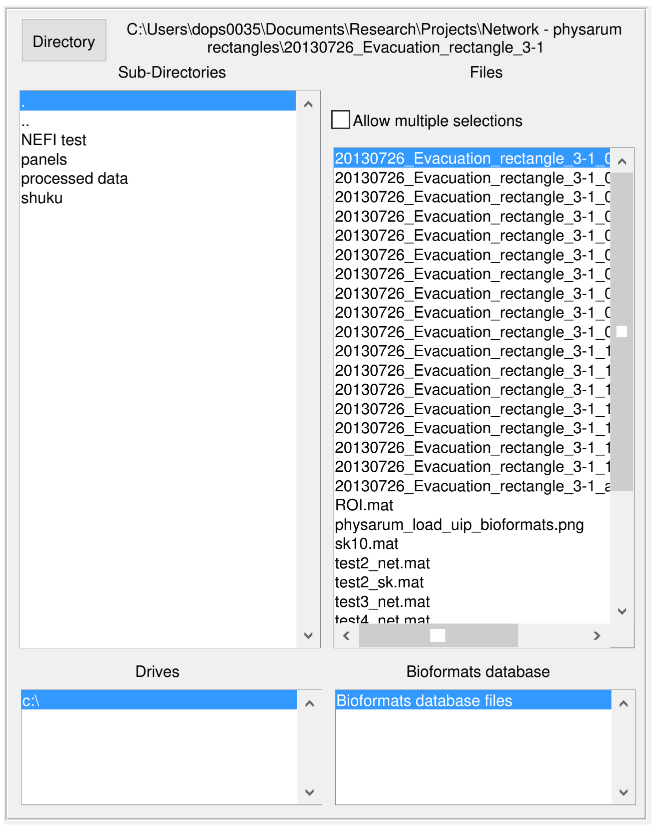
http://www.openmicroscopy.org/site/support/bio-formats4/
http://loci.wisc.edu/software/bio-formats

[1] M. Linkert, C.T. Rueden, C. Allan, J.-M. Burel, W. Moore, A. Patterson, B. Loranger, J. Moore, C. Neves, D. MacDonald, A. Tarkowska, C. Sticco, E. Hill, M. Rossner, K. W. Eliceiri, and J. R. Swedlow. Metadata matters: access to image data in the real world. *The Journal of Cell Biology*, 189:777–782, 2010

9.2 File Selection

The file selection panel shows:

- a **Directory** button that opens a standard dialog box to select a different directory
- sub-directories of the current folder
- individual files in the current directory
- the currently selected file
- the available drives
- image files within a bioformats database (if appropriate)



Directory

C:\Users\dops0035\Documents\Research\Projects\Network - physarum rectangles\20130726_Evacuation_rectangle_3-1

Sub-Directories

.
..
NEFI test
panels
processed data
shuku

Files

Allow multiple selections

20130726_Evacuation_rectangle_3-1_0
20130726_Evacuation_rectangle_3-1_0
20130726_Evacuation_rectangle_3-1_0
20130726_Evacuation_rectangle_3-1_0
20130726_Evacuation_rectangle_3-1_0
20130726_Evacuation_rectangle_3-1_0
20130726_Evacuation_rectangle_3-1_0
20130726_Evacuation_rectangle_3-1_0
20130726_Evacuation_rectangle_3-1_0
20130726_Evacuation_rectangle_3-1_0
20130726_Evacuation_rectangle_3-1_1
20130726_Evacuation_rectangle_3-1_1
20130726_Evacuation_rectangle_3-1_1
20130726_Evacuation_rectangle_3-1_1
20130726_Evacuation_rectangle_3-1_1
20130726_Evacuation_rectangle_3-1_1
20130726_Evacuation_rectangle_3-1_1
20130726_Evacuation_rectangle_3-1_a
ROI.mat
physarum_load_uip_bioformats.png
sk10.mat
test2_net.mat
test2_sk.mat
test3_net.mat
test4_net.mat

Drives

c:\

Bioformats database

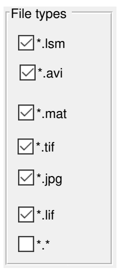
Bioformats database files

If the **Allow multiple selections** checkbox is ticked, a range of files can be loaded in one operation. This is useful to join a set of consecutive sequences from the same time-series experiment. The images must have the same dimensions in x and y, and have the same number of channels. Images are sorted in alphabetical order in the listbox and will be imported in this order. If a different order is required, each file has to be added in sequence manually.

A set of checkboxes (Fig. 9.1) are available to display a restricted set of file types or all files in the directory (*.*)

Once the required file(s) have been selected, the information on each file can be loaded by double clicking or using the **right-arrow**.

If the system can read the file format information, The file name(s) will then appear in the **output list** box, and the image details shown in both the **Input details** and **Output details** panels (see Fig. 9.2). If multiple files have been selected, only the details



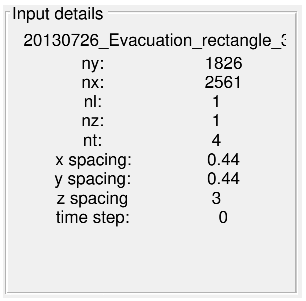
File types

[✓ *.lsm]
[✓ *.avi]
[✓ *.mat]
[✓ *.tif]
[✓ *.jpg]
[✓ *.lif]
.

Figure 9.1: Setting the file extensions

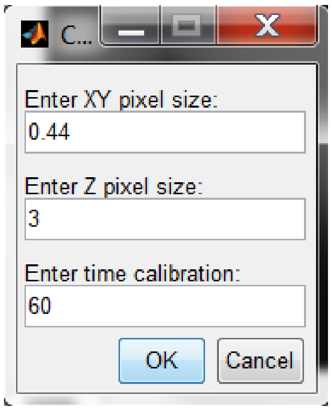
of the last file will be shown. Input and output details will be the same at this stage as no additional processing steps have happened.

If the system can not pick the pixel size information or time interval, a dialog box will appear with prompts for the user to enter the required information (see Fig. 9.3).



```
Input details
20130726_Evacuation_rectangle_3
         ny:        1826
         nx:        2561
         nl:           1
         nz:           1
         nt:           4
  x spacing:        0.44
  y spacing:        0.44
  z spacing            3
 time step:            0
```

Figure 9.2: The input details panel: displays of the size and calibration values for the current image



```
Enter XY pixel size:
0.44
Enter Z pixel size:
3
Enter time calibration:
60
```

Figure 9.3: Dialog box prompting for the x,y,z spacing and the time-interval

9.3 *Image display*

Once the images have been loaded, the first image of a time series will be displayed in the image window (Fig. 9.4).

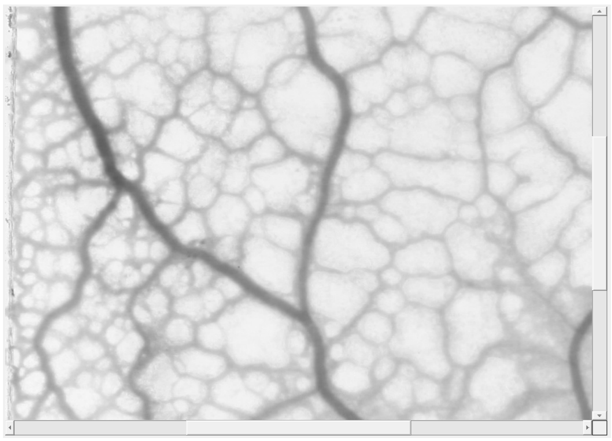


Figure 9.4: Image display panel: The image has been loaded in and an RGB version of the first three wavelength channels displayed. The red rectangle shows a user-defined region that will be cropped from the original file

If the image is a *z*-stack, the median plane will be displayed. If it is a multi-channel image, the median channel image will be displayed. If multiple separate image files have been loaded, a different file can be displayed by selecting the appropriate file from the output list box. A number of controls are available to alter how the image is displayed (Fig. 9.5).

For multi-channel images, different channels can be assigned to the **R**, **G** and **B** image planes using the drop-down menus to construct a RGB image, which will be displayed if the **RGB** checkbox is active.

The image size can be increased using the **Zoom** slider. If the image is larger than the display window, horizontal and vertical scroll bars will appear. The **fit** button maximises the size of the

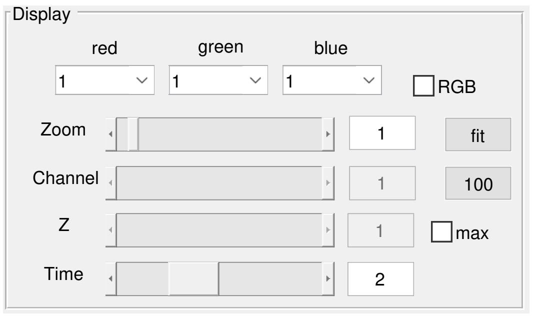


Figure 9.5: The display panel: Controls are provided to allow the user to select which channel, section or time-point to display, with further options to combine channels into an RGB image, or display a maximum projection of the z-stack (if present)

image to fit within the display window. The **100%** button displays the image at a 1:1 image pixel to display pixel size.

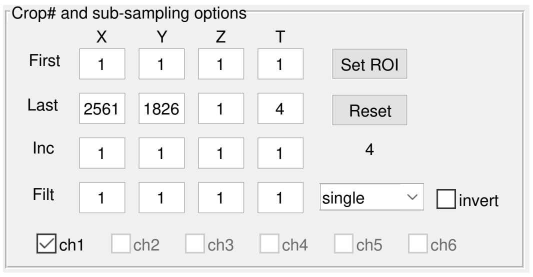
The **Channel** slider displays a single channel image (if the RGB checkbox is un-ticked).

The **Z** slider scrolls through *z*-sections if the image is a 3-D (*x*,*y*,*z*) stack. The **max** checkbox displays a maximum projection of the current *z*-stack.

The **Time** slider scrolls through each image in the time series.

9.4 *Image crop and sub-sampling options*

The image can be cropped by entering the **First** and **Last** pixel co-ordinates independently in the **X, Y, Z** or **T** text boxes (Fig. 9.6). Alternatively, a region-of-interest (ROI) can be selected in *x* and *y* using the **Set ROI** button. This prompts the user to draw a rectangular ROI on the image, which is then displayed in red. Completion of the ROI will update the values in the text boxes. Values can be reset using the **Reset** button if required.



	X	Y	Z	T
First	1	1	1	1
Last	2561	1826	1	4
Inc	1	1	1	1
Filt	1	1	1	1

Figure 9.6: The crop panel: Provides controls to crop, sub-sample and filter images

The **Inc** text boxes allow (integer) sub-sampling independently in each image dimension.

The **Filt** text boxes allow spatial or temporal averaging over the designated number of pixels.

The image class can also be changed at this point from **integer** (the default) to single or double precision.

A series of checkboxes (**ch1**. . .**ch6**) are available to select which channels are to be included in the output image. Usually the bright-field channel is not included in this selection, as it is processed separately (see Section 9.8 - Bright-field image processing).

The file details in the **Output details** panel should update to reflect the cropping and sub-sampling chosen. If multiple images have been loaded, the crop and sub-sampling options apply to all the files.

9.5 *Alignment options between wavelength images*

Options are available to correct slight mis-registration in *x,y* between individual wavelength images using the **Channel alignment** controls (Fig. 9.7). A reference channel, typically containing the best image selected using the **Z** and **T** sliders, is chosen using the **Ref channel** drop-down list.

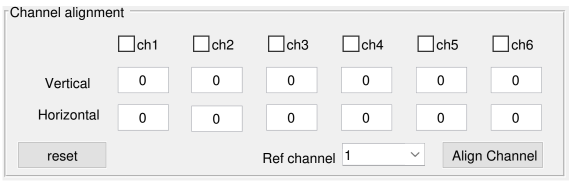


Figure 9.7: The channel alignment panel: Provides controls to allow sub-pixel registration between each channel and a selected reference channel

This acts as a template for cross correlation of any other channel, selected using the **ch1**...**ch6** checkboxes. The **Align Channel** button calculates the **Vertical** and **Horizontal** pixel offsets between the reference channel and the selected channels using cross-correlation across the whole image. These offsets are applied using bi-linear interpolation when the images are actually loaded.

The image display is modified during this process to display the before and after images in magenta and green.

9.6 *Alignment options over time*

Some level of correction for stage *x,y* drift or specimen movement can be achieved using the **Time series alignment** controls (Fig. 9.8). A reference channel is selected that has good contrast and features that are present in all images in the series, using the **Z** and **T** sliders, and, if necessary, a particular file if multiple files have been loaded simultaneously.

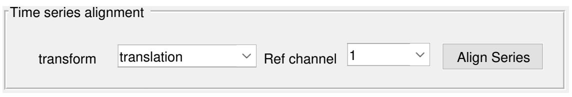


Figure 9.8: The Time series alignment panel: Provides controls to define a region for image alignment through the time-series, using a reference image

The **Align Series** button prompts the user to select a ROI on the target image that will be used as a template to calculate the *x,y* pixel offsets, rotation and scaling for the corresponding image in subsequent time-points using cross-correlation and bi-linear interpolation. The region used as a template must be within the red ROI outline, if this has been used to crop the image, and is highlighted in blue. The same offsets are applied to all channels and **z**-planes for each time point. The alignment only takes place when the images are actually loaded. A number of different transform types can be applied with increasing degrees of freedom, including *translation*, for (*x,y*) translation, *rigid*, for translation and rotation, *similarity* for translation, rotation, and scale and *affine*, for translation, rotation, scale, and shear.

9.7 *Image projection options*

It is possible to reduce the image dimensionality by projecting data along the *z*-axis using the drop-down menu in the **Projection options** panel (Fig: 9.9).

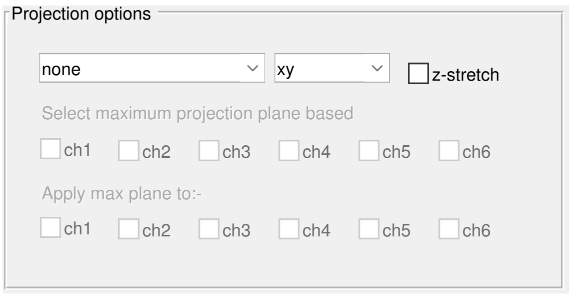


Figure 9.9: The projection panel: allows the user to reduce the image dimensionality by extracting a sub-set of the data in the z-dimension, or to re-orient the image stack to display xz *or* yz *views. A number of different projection options can be selected and applied to different combinations of channels*

- **Maximum:** displays the pixel with the maximum intensity in *z* for each channel (maximum intensity projection or MIP). This is a common approach to visualise data, but should not be used as a precursor to quantitative measurements, particularly when ratioing two channels, as it selects the 'noisiest' pixel at the extreme of the distribution along the *z*-axis, and pixels from different positions in *z* for different channels will appear in the projected image, making a nonsense of the ratio image.
- **Minimum:** displays the pixel with the minimum intensity in *z* for each channel. This is rarely useful for fluorescence images, but can be helpful for bright-field processing. Nevertheless, it is recommended that bright-field processing is handles separately (see Section 9.8 - Bright-field image processing)
- **Average:** gives an average brightness projection in *z*, which provides good noise reduction and may be useful for simple objects that do not overlap in the *z*-direction.
- **Max plane:** this gives the user the option of selecting the *z*-position of the brightest pixel in one-or-more channels and then extracting the same (*x*,*y*,*z*) pixel (voxel) from the other channels. It is recommended that the imaged is smoothed in *z* (as well as *x* and *y*), before this operation to ensure that the brightest pixel is more likely to correspond to the centre of the object of interest. This approach is required if different channels are going to be ratioed later on to ensure that information from the same (averaged) voxels are compared.
- **Mx + mx plane:** This provides an option to calculate the maximum plane projection, based on specific selected channel(s) that extracts the appropriate (*x*,*y*,*z*) voxel from a second set of channels, selected by the second set of checkboxes. The remaining

channels are processed using a simple maximum. This is useful for quantitative ratioing for the max plane images, which typically involve two wavelength channels and an autofluorescence channels for bleed through correction, and a morphological representation of the other channels from the (smoothed) maximum intensity projection.

9.8 *Bright-field image processing*

In microscope-based fluorescent imaging, a bright-field image is often collected simultaneously with the fluorescence channels using a (non-confocal) transmission detector. Bright-field images can be processed separately to accentuate more useful information using the **Bright field output** controls (Fig: 9.10). The simplest form of processing is a **single plane**, selected by the position of the *Z*-slider from the bright field channel (**Ch.**). An amount of noise filtering can be applied using the process selected by the **smooth** drop-down list (*mean*, *median* or *Wiener*) over a square *x*,*y* region specified by the **kernel** drop-down list.

If a projection option is chosen, the process required can be selected from the **process** drop-down menu. The algorithms available are designed to highlight pixels that might contain the most useful information within a local neighbourhood defined by the **kernel** drop-down menu.

Whether a projection of single plane option is chosen, the contrast of the resulting image can be improved using contrast-limited adaptive histogram equalisation (CLAHE) by checking the **adaptive histogram equalisation** box.

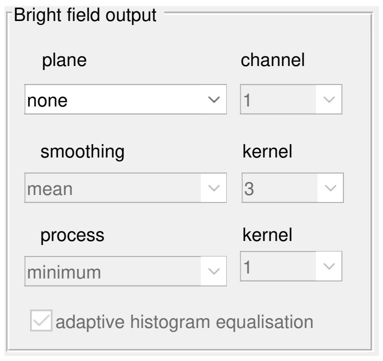


Figure 9.10: The bright field output panel: Provides controls to allow selection of single bright-field image planes or various algorithms to project the bright-field images, along with some contrast enhancement

9.9 *Saving the imported image and processing settings*

The **Save settings** button in the **Save options panel** saves the settings used for processing in a matlab file format, with the filename and a *settings* suffix (Fig: 9.11. The parameters can be re-loaded using the **Load settings** at a later stage. Note: when the settings are re-loaded, the user is prompted to re-set the alignment box for the time-series, and the cropping ROI is not available and must therefore be set-up again.

The **Save mat** button saves the processed fluorescence images and, if appropriate, bright-field images in a matlab file format. This can be re-loaded using the same interface at a later stage.

The **Save panels** button saves "png" images of each of the panels in the image and can be used to update this manual for any specific applications.

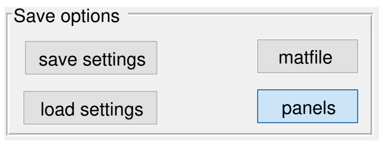


Figure 9.11: The Save options panel: allows the user to save the loaded image and the processing settings

9.10 *Importing the selected files*

Once all the processing steps have been completed, the selected files can be imported using the **Load** button (Fig: 9.12). If this is successful, the **View** button will be enabled and, if a separate bright field image has been processed, the **View bf** button. Clicking the **View** button will open a separate window with a video **Viewer** (see Chapter **??** - Viewer Program). If the image files have been loaded satisfactorily, clicking the **OK** button will return to the main program. The **Cancel** button will return to the main program, but without exporting the processed image file.

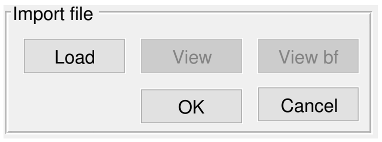


Figure 9.12: The import file panel: The Load button imports the selected image(s) and applies all the alignment, sampling, smoothing and projection parameters chosen. Once the fluorescence and bright-field images have been processed, they can be viewed with the appropriate button