Supplementary Methods

Automatic Skeleton Correction
To remove a significant proportion of small erroneous branches from the skeleton, all singly-connected branches of less than 10 µm in length were automatically deleted. In an effort to repair vessel-gap errors, we combined several skeletonizations of the data. Multiple skeletons were produced, employing a series of progressively lower thresholds at the segmentation step. Starting with the highest (most conservative) threshold skeleton, unconnected branch endpoints within the volume were found and labeled, by determining all skeleton voxels with exactly one neighbor within their locally-connected 3×3×3 neighborhood. The software then attempted to resolve the unconnected endpoints systematically. Firstly, the skeleton was compared with an alternate skeleton that was obtained by applying a lower threshold to the original binary mask. In general, the use of a lower threshold results in a skeleton with many additional branches, of which some may be erroneous, while others may represent the true vasculature. The software searched this alternate skeleton to find any branches that would connect two endpoints if they were transplanted to the original skeleton. Similarly, a search was performed for branch segments that would connect an endpoint to another point on the skeleton. Any branches meeting either of these criteria were combined with the original skeleton to produce a new, corrected skeleton. This process was then repeated using additional lower-threshold skeletonizations. In all, five separate thresholds were used, yielding a final corrected skeleton with an augmented branch structure and fewer unconnected endpoints.

Manually-Guided Skeleton Correction Using Graphical User Interface (GUI)
The automated procedures described above removed some of the errors in the skeletonization. The remaining errors were resolved interactively, using a custom-made software GUI in MATLAB®. This GUI allowed the user to graphically overlay skeleton branches and endpoints on the image data in a maximum intensity projection (MIP), where the orientation and depth of the projection was manually chosen. The user could systematically resolve unconnected endpoints, redundant skeleton branches, and other errors in the data. For each singly-connected branch, the user could easily remove the branch, join it to another singly-connected branch via the endpoints, or join it to an arbitrary point elsewhere on the segmented skeleton (creating a new branch point in the process). Two separate software methods were used for the manually-assisted correction: shortest-path computation and extrapolation of endpoint trajectories.

Shortest-Path Computation
Once the GUI user identified two points on the skeleton for which a connecting branch was required (either two endpoints, or an endpoint and a skeleton point), the software attempted to draw a bridging strand, computed as the shortest path between the two points, based on Dijkstra’s algorithm¹. The path entries were weighted according to the value of the local enhanced image data at each voxel. The resulting bridging strands were then incorporated into the corrected skeleton.
Extrapolation of Endpoint Trajectories

For endpoints to which the shortest-path computation could not be applied (e.g., in cases where there was significant drop-out of vessel contrast), an alternative method based on the projection of vessel path trajectory was employed. Using the geometry of the skeleton near the endpoint, the local curvature of the image structure (obtained via the eigenvector corresponding to the smallest eigenvalue of the Hessian matrix), and the surrounding enhanced image information, we applied a model-based extrapolation method. For each unresolved skeleton endpoint, we iteratively extrapolated the unconnected branch, starting from its endpoint. At each trajectory point, the next step was determined by finding the local curvature associated with all voxels within the local $3 \times 3 \times 3$ neighborhood, applying a weighting based on each voxel's associated enhanced image intensity, and then selecting the updated trajectory point according to the voxel location that maximized this weighted curvature. The extrapolated trajectories were evaluated for a maximum of 50 steps. If the trajectory was found to intersect with another point on the skeleton, the projected path was presented via the GUI to the user, who was then prompted to select whether or not to incorporate it into the corrected skeleton as a bridging strand, based on a visual assessment of its validity.

Vascular Layer Delineation

The boundaries between the vascular layers were delineated using a manually-guided interpolation approach. The data volume was split into successive XZ slices through the data volume, each spanning 10 frames in the Y-direction. Maximum intensity projections of the angiogram data were then computed over the Y-direction, yielding a series of 2-D representations of XZ slices of the data. For each of these, a user traced (using piecewise curves established by mouse-clicks) two boundary lines based on the angiogram information. One boundary separated the superficial and intermediate layers, while the other separated the intermediate and deep layers. These boundary lines on the successive XZ slices were then converted to surface boundaries, using 3-D linear interpolation. Each branch and branch point was then associated with a particular vascular layer (or layers), based on the co-ordinates of its constituent voxels in relation to the boundary surfaces.

Vectorization of Vascular Skeleton

The vectorization proceeded as follows: first, the branch points on the skeleton were found. This was achieved by first determining all voxels on the skeleton whose deletion would result in the number of distinct objects in the local $3 \times 3 \times 3$ neighborhood increasing from one to three. Second, a separate routine determined the skeleton voxels with more than two neighboring voxels within their $3 \times 3 \times 3$ neighborhood. The two sets of voxels identified using these respective methods were combined by a logical OR operation, giving a binary map of the branch points equal in size to the skeleton data volume.

Next, individual branches were labeled by deleting the branch points, thus breaking the skeleton into distinct branch components that could each be assigned an individual label. A sorted list of the voxels comprising each individual branch was determined. The branch points were also labeled, independently from the branches. The graph connectivity information was found by searching the local neighborhood of each labeled branch point in turn, and identifying the labels of branches that contacted this neighborhood. In this manner, each labeled branch point was
associated with one or more labeled branches. Similarly, branches could then be associated with their respective connected branch points. Each branch point was assigned to a particular retinal vascular layer, based on its location. A branch was labeled as a connector between two particular layers, if each layer contained one of its two associated branch points. The above information was stored in indexed data structures. The criteria used for branch point detection in some cases returned branch points that consisted of more than one voxel. In such occurrences, the multi-voxel structures were considered as single branch points, and the connections of these constituent voxels were merged.

Branch Order

We assigned an order of “1” to major supplying and draining arteries and veins, and subsequently used the vectorized graph to determine the order of all other branches in the network. Arterioles and venules that branch from the major supplying vessels were assigned an order of “2”, and capillaries were associated with orders of “3” or higher. Accordingly, if the major retinal vessels are labeled as either arteries or veins, this labeling can be used in conjunction with the connectivity information provided by the graph in order to distinguish between arterioles and venules. Supplementary Movie 3 illustrates labeling of major arteries and arterioles (red), major veins and venules (blue), and capillaries (green).

Computation of Branch Diameter and Length

A 3-D parametric representation $\langle x(t), y(t), z(t) \rangle$ of each branch was obtained using 3rd order polynomial fitting, where $t$ is the sorted voxel index. An estimate $\hat{l}$ of the length of each branch over the parameterized space $\mathcal{S}$ was computed from the parametric representation via

$$
\hat{l} = \int_{t \in \mathcal{S}} \sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2 + \left(\frac{dz}{dt}\right)^2} \, dt
$$

Vessel branch diameter was then estimated at multiple locations along each branch. Firstly, the branch was divided into overlapping nine-voxel segments, with a parametric representation computed for each. At the center voxel of each segment, parameterized as $t_c$, we constructed a line $L$ in the xy-plane that was orthogonal to the segment at $t_c$, along which cross-sectional grayscale image information could be used to estimate the diameter. The slope of the orthogonal line was given by

$$
m(t_c) = \left[ -\frac{dx}{dy} \right]_{t_c} = \left[ -\frac{dx}{dt} \left(\frac{dy}{dt}\right)^{-1} \right]_{t_c}
$$

We then used this slope to obtain the co-ordinates of the line $L$ through $t_c$, and computed grayscale image data values along $L$ using interpolation. These grayscale values comprised a cross-sectional profile of the vessel branch at $t_c$. To estimate the vessel diameter, we fitted a Gaussian model to the cross-sectional profile, of the form
\[ f(r; \theta) = \theta_1 e^{-\frac{(r-\theta_2)^2}{2\theta_3^2}} + \theta_4 \]  

where \( r \) denotes location along \( L \). The computed diameters were corrected to account for the theoretical point-spread function of the imaging system\(^3\), as well as loss of resolution due to scanning during the acquisition. Supplementary Figure 2 illustrates the computation of vessel diameter estimates. In Supplementary Figure 2A, the measurement site is depicted; the cross-sectional profile is obtained along a line (green) that has slope perpendicular to the XY projection of the skeleton branch (red) at the measurement site. Supplementary Figure 2B shows the measured line profile, as well as the Gaussian fit used to estimate the vessel diameter.

The potential effect of branch diameter errors on flow modeling is significant, as conductance is proportional to the fourth power of diameter, and exhibits critical behavior for vessels with diameter on the scale of smaller capillaries\(^4\). Ascertaining the actual vessel boundaries from angiogram data may be confounded by several factors, including red blood cell location and orientation effects\(^5\). Also, while the transverse point spread function (7.2 \( \mu \)m full-width at half maximum) was accounted for directly in determining true diameters, the implicit deconvolution process is noisy. To mitigate these effects, we computed average diameters for each vessel branch by taking the mean of all diameter estimates along the branch, excluding values for which the fitting error was deemed high.

**Theoretical Flow Reconstruction**

We chose to model flow in the network by way of an analogous representation based on lumped electrical parameters. Though this type of model is not suitable for predicting hemodynamic parameters for individual vessel segments, the distribution of flow can be estimated with reasonable precision\(^4\). Vessel branches can be represented as branches of a resistive network, with a resistance term determined by the branch length and radius. We used the topological information provided by the vascular graph to derive a simple model of blood flow in the inner retina. Similarly to the approach taken by Blinder et al. for cortical vasculature\(^6\), we employed a circuit analogy based on Kirchoff’s current law to compute values of blood flow. As illustrated in Supplementary Figure 3A, the sum of the flows directed outward from a branch point \( m \) is considered to be zero, i.e.,

\[ \sum_{k \in N_A} \frac{P_m - P_k}{\rho_{mk}} = 0 \]  

where \( P_m \) is the “pressure” at node \( m \), \( N_A \) is the set of adjacent branch point nodes, and \( \rho_{mk} \) is the “resistance” of the branch connecting nodes \( m \) and \( k \).

Since blood comprises a suspension of blood cells in plasma, the flow of blood is more complex than that of a pure fluid. There are several major effects that are known to influence flow in microvasculature, including the Fåhræus effect, the Fåhræus-Lindqvist effect, and the phase separation effect. These phenomena have significant implications for the distribution of blood...
viscosity and pressure, as has been described in previous work\textsuperscript{4, 7}. For our simulations, the resistance value of a particular branch was modelled as a function of its length and radius, under the assumption of a modified Hagen-Poiseuille law\textsuperscript{6, 8}.

**Boundary Conditions**

In order to emulate realistic hemodynamics within the retinal vasculature, we applied flow and pressure boundary conditions derived from experimental data. Firstly, for endpoints corresponding to large arteries and veins at the optic nerve head, the boundary flow values were assigned based on Doppler OCT measurements of blood flow. For branch points that were directly connected to an endpoint (as depicted in Supplementary Figure 3B), a boundary condition was applied. This is achieved by modifying Eq. 4, such that it takes the form

\[
\sum_{k \in N_A} \frac{P_m - P_k}{\rho_{mk}} = Q_{meas}
\]

(5)

where \(Q_{meas}\) is the measured flow into the endpoint. At the edge of the field, the flows in these same vessels were estimated using the assumption that flow is apportioned evenly across the retina; the magnitude of flow in each vessel was assumed to decrease between the optic nerve head and the edge of the field-of-view by a fraction equal to the ratio of the vessel’s supply zone in the field-of-view to the vessel’s assumed supply zone in the whole vascularized retina. The assumed supply zones in the field-of-view are illustrated in Supplementary Figure 4. Secondly, we imposed node pressures for major vessels near the optic nerve head as hard constraints, using pressure values reported for arteries and veins of commensurate diameter in the cat mesentery\textsuperscript{9}. Realistic flow and pressure values within the solved network were further ensured by modifying branch resistance values to incorporate an increase in apparent viscosity at lower vessel diameter, according to the method described by Pries et al.\textsuperscript{4}.

The equations for the \(M\) branch points can be assembled and written as a matrix equation of the form

\[
Cp = q
\]

(6)

where \(C\) is an \(M \times M\) matrix, \(p = [P_1, P_2, ..., P_M]^T\) is the vector of unknown node pressures, and \(q\) is a vector representing the sum of branch flow rates at each node. Eq. 6 was solved in the least-squares sense, in order to find approximate values of the node pressures, i.e.,

\[
\hat{p} = \arg\min_p \| Cp - q \|
\]

(7)

An implementation of Eq. 7 was performed in MATLAB\textsuperscript{®} using a linear least-squares algorithm. Using the estimated node pressures, an estimate of flow rate \(Q_{m,n}\), i.e., the flow in the branch that joins branch points \(m\) and \(n\), can be obtained via the relation

\[
Q_{m,n} = \frac{P_m - P_n}{\rho_{mn}}
\]

(8)
Absolute values of flow and pressure are not necessarily accurate in such a model, as they depend on assumptions relating to the boundary conditions and blood viscosity. However, the distributions of these parameters are largely independent of these assumptions, and thus flow simulations based on the model can be used to assess the vascular interconnectivity. Arterioles (defined here as first-order branches from vitreal arteries) and capillaries were singled out for simulated dilation, as they are known or hypothesized to be involved in the regulation of blood flow\textsuperscript{10}. When examining the effects of vessel dilation on the surrounding network, dilation was modeled as a fixed increase in the vessel diameter. For each simulated dilation experiment, Eq. 7 was solved once at baseline and again after dilating selected vessels by modifying the $C$ matrix. Proportionate changes in flow following dilation, relative to baseline, were determined.
References