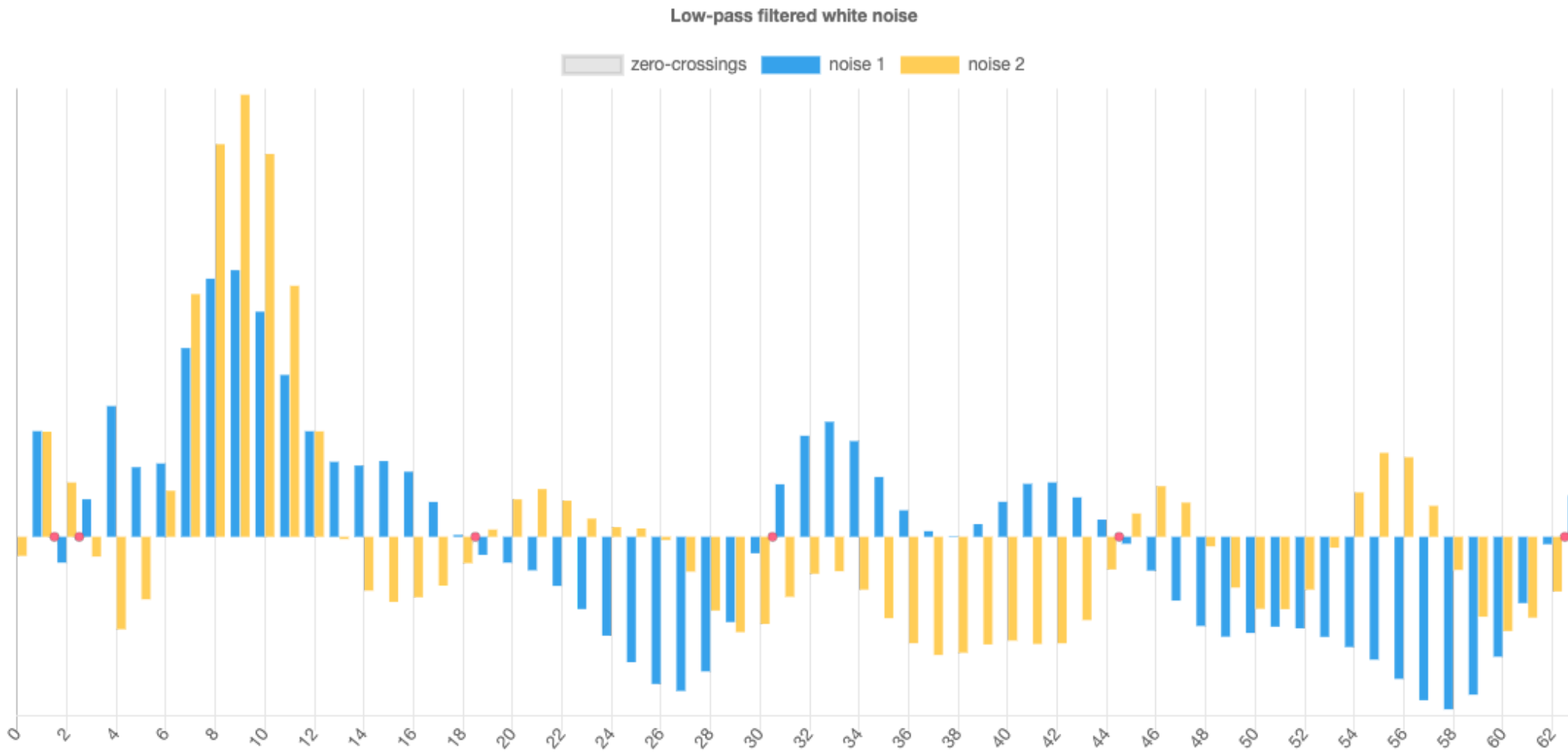
What can('t) we do with low-pass filtered white noise?

Here an illustration of the vessel generation process. Following Edwin Abbott's Flatland, in order to see what is going on we can start with a simple 1-D case. We create an array with 64 random numbers between -0.5 and +0.5. The bars in the graph below represent the values at 64 line element positions. For positive values the bar goes up, for negative values the bar goes down. Each time the random number is close to zero the bar is shortest. The red dots represent positions on the line each time a bar is followed by another bar with the opposite sign. These points indicate the zero-crossings of our random list of numbers.



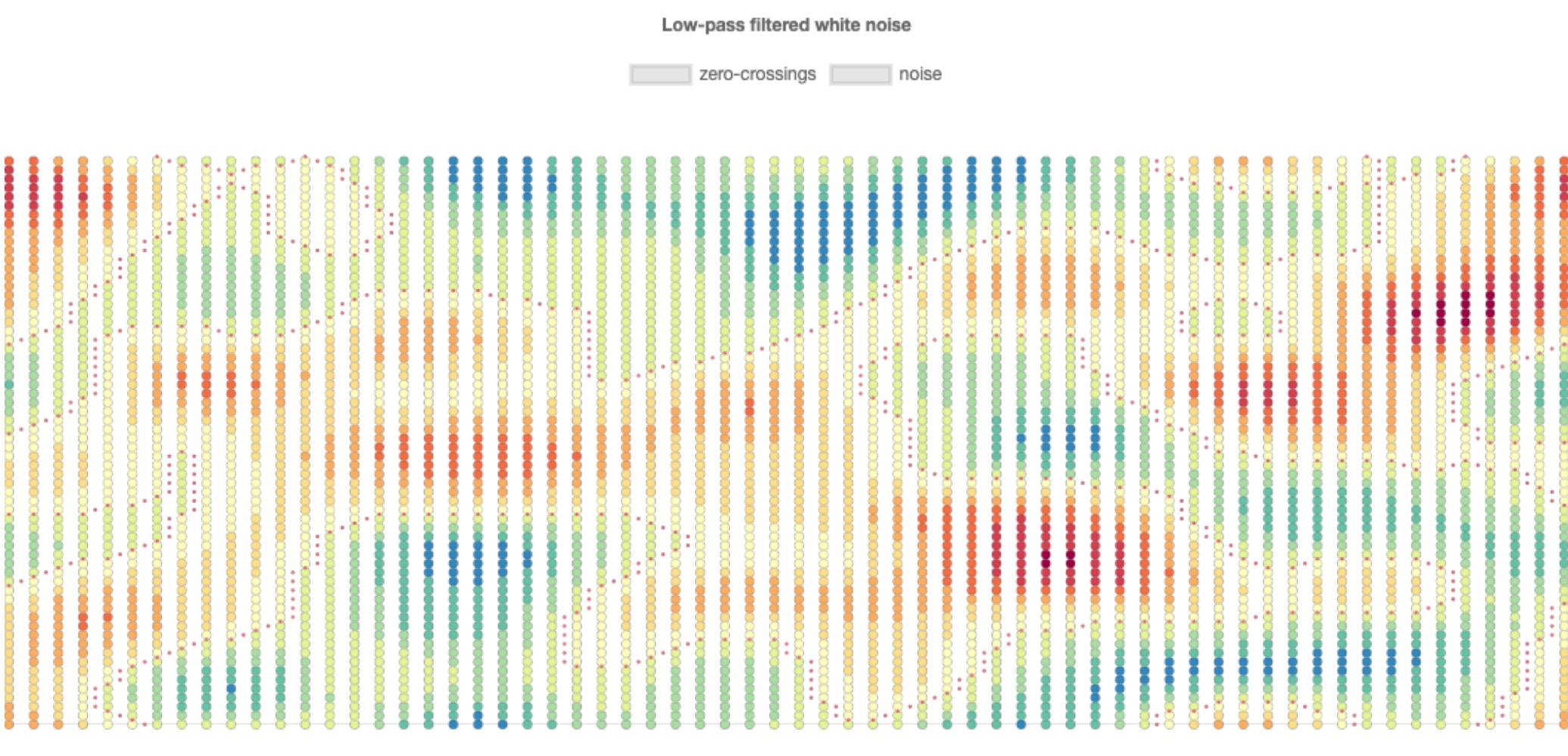
Low-pass filter (iteration: 17)

Each time we click on the "Low-pass filter" button the initially random list of numbers is smoothed by averaging three consecutive values. As this step is repeated only low frequency variations in the data are retained. High frequency components are removed. In order to show the effect each click applies the filter operation 7 times. As the number of filtering steps increases the values become smoother with larger structures and with fewer zero-crossings.

There is a second (hidden) random array of numbers. If you enabled the display of the array in the figure you can see that similar observations apply to this independently smoothed array. It also becomes smoother and different positive and negative regions are established.

Summary: In 1-D a low-pass filtered white noise array has zero-crossings that are points (have dimension 0). As more smoothing steps are applied fewer points are generated.

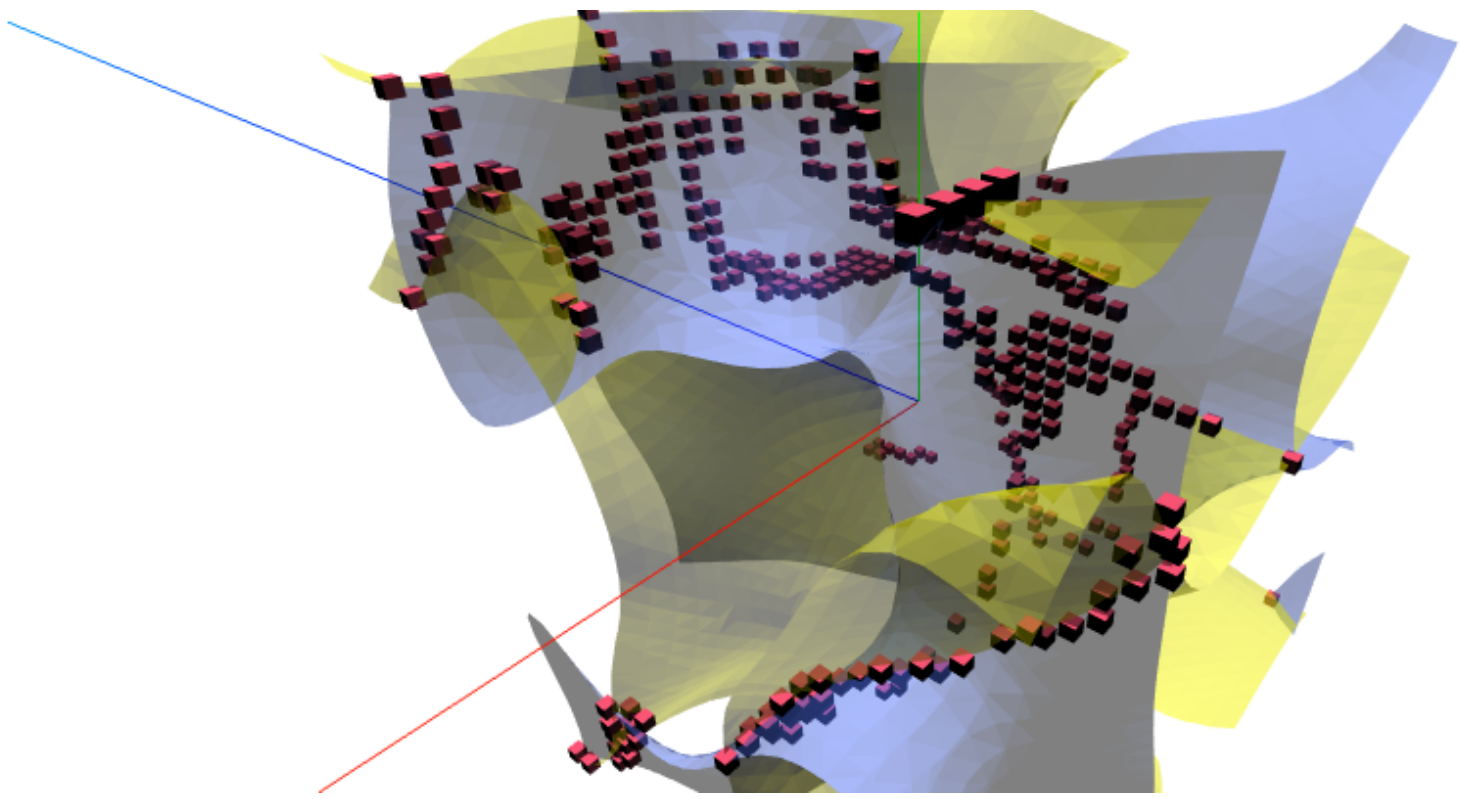
We now repeat the same process with one additional dimension in 2-D. We will see that instead of points this process will generate lines as zero-crossings.



Low-pass filter (iteration: 29)

Each time we low-pass filter the grid the structures become larger. Enable the display of the zero-crossings in the graph after a couple of smoothing steps. It is easy to see that an independently filtered second field would generate different zero-crossings. The intersection of those two line segments is again a set of points. This pattern continues in dimensions greater than 2.

One more dimension! To not clutter our screen we will skip the display of the random values and only show the zero-crossings. We will do this twice on two independently generated and filtered white noise volumes. The zero-crossings are calculated using a marching cubes algorithm so we can visulize them as surfaces inside our data cube, one in blue for the second random field in yellow. In 3D this results for each random field in a 3-1=2D manifold (surface). Additionally to the two surfaces we also mark the locations in our 32x32x32 volume where the zero-crossings of both manifolds intersect, or more correcly, where they are both sufficiently close to zero as red cubes.



Low-pass filter (iteration: 89)

After many low-pass filter steps you can see that the intersections of the two manifolds are forming 3-1-1=1D manifolds - lines.

Summary: Each filtered random field of dimension N has a zero-crossing of dimension N-1. The intersection of two zero-crossings of dimension N-1 has dimension N-2.

Ok, so how can we use this? We like to generate de-novo data that in same respects resemble microvascular tissue. The zero-crossings of a low-pass filtered 3-D volume is a 2-D surface. The zero crossings of two low-pass filtered 3D volumes are objects of 2 dimensions less - so instead of surfaces we get 3 - 2 = 1D lines. Can we interpret those as standins for blood vessels? Here an short movie of what these lines in 3D can look like if we increase the resolution of the volume.

Link to the Movie

This volume was generated using the FakeBloodVessel program that implements this procedure for 3D data. There is more structure that we can get from this setup. The space in-between the intersecting zero-crossings can be described by the pair of signs of the low-pass filtered random fields. There are four possible combinations of signs ++, +-, -+, and -- so we get a random assortment of blobs that will never intersect with our lines. Labels for these void regions are created by the application FakeBloodVessels if the '-w' option is used. The options gets another argument that specifies how far away from the lines the regions are. The space in between is 'empty'.

Ok, what next? We can now place objects into the different regions. We have i) empty space, ii) vessel like structures, and iii) four different void regions.

Notes

We call it low-pass filtering, why not band-pass filtered white noise? Oh, so yeah, so we start with all kind of frequencies even high-frequency variations in intensity, but by averaging we are retaining only the low-frequency variations. Each time we average we loose some of the high-frequencies. .

Implementing a low-pass filter using averaging on a 6-neighborhood stencil and applying it multiple times is not a good idea in practice. Here we do this as an illustration only. It tends to still show the features discussed. The FakeBloodVessel application is using larger filter kernls to reduce the tendency of the smoothed data to be less smooth in diagonal directions. Applying the smoothing operation once in frequency space is another (future) todo.