MATH590: C3examples, topological clustering of blood cells

In this module homework you will use qualitative topological methods to identify red blood cells.

Read data and carry out visual examination of images

Isolate erythrocytes

Modify the function used to isolate leukocytes from the C2examples.nb to return a mask for the erythrocytes (RBCs).

In[7]:=
```
LeukocyteMask[im_, t_] := Module[{imR, imG, imB, mask,
   MaxCells, MinPixels, cellsB, Lmask, FLmask, Bmask, Rmask, FRmask},
  {imR, imG, imB} = Map[ImageAdjust[#] &, ColorSeparate[im]];
  mask = ColorNegate[Binarize[imG]];
  MaxCells = 100;
  MinPixels = Floor[Apply[Times, ImageDimensions[mask]] / MaxCells];
  cellsB = ImageMultiply[mask, imB];
  Lmask = FillingTransform[Binarize[cellsB, t]];
  FLmask = Binarize[ColorConvert[
     Colorize[SelectComponents[Lmask, #Count > MinPixels &]], "Grayscale"]];
  Bmask = FillingTransform[Binarize[ColorConvert[
      Colorize[SelectComponents[mask, #Count > MinPixels &]], "Grayscale"]]];
  Rmask = ImageMultiply[Bmask, ColorNegate[FLmask]];
  FRmask = FillingTransform[Binarize[ColorConvert[
      Colorize[SelectComponents[Rmask, #Count > MinPixels &]], "Grayscale"]]];
  Return[FRmask];
 ]
```

Here is an example of a mask isolating erythrocytes in a portion of an image

In[8]:=
```
im = ImageTake[Import[f[[1]]], {1, 160}, {1, 100}];
Rmask = LeukocyteMask[im, 0.6]
```

Out[9]=

And these are the erythrocytes

In[10]:=
```
rbc = ImageMultiply[Rmask, im]
```

Out[10]=

Revisit the problem of isolating individual erythrocytes by introducing a simplicial complex to represent the image data, and then isolating convex parts of dimension 2. First, find the edge of the erythrocyte domain

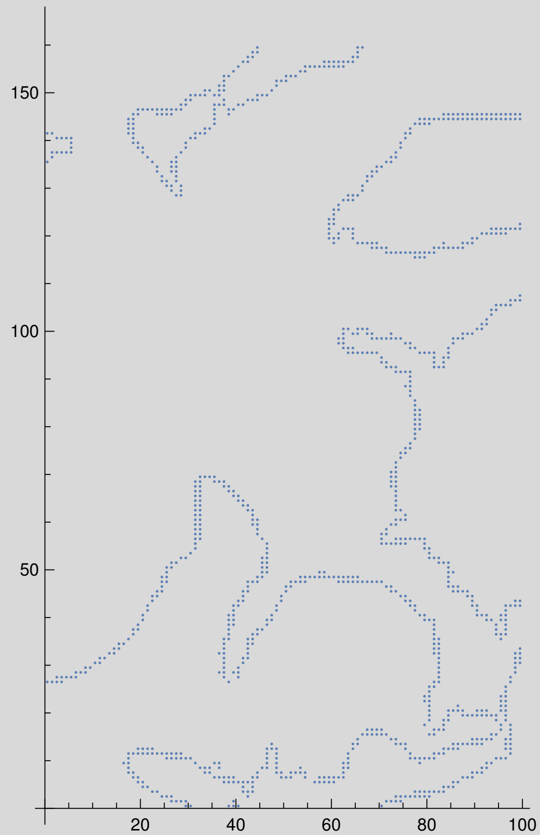
In[11]:=
```
edges = EdgeDetect[Rmask, 1]
```

Out[11]=

Construct a list of the edge points

In[12]:=
```
edgePoints = ImageValuePositions[edges, White];
ListPlot[edgePoints, AspectRatio → Automatic]
```

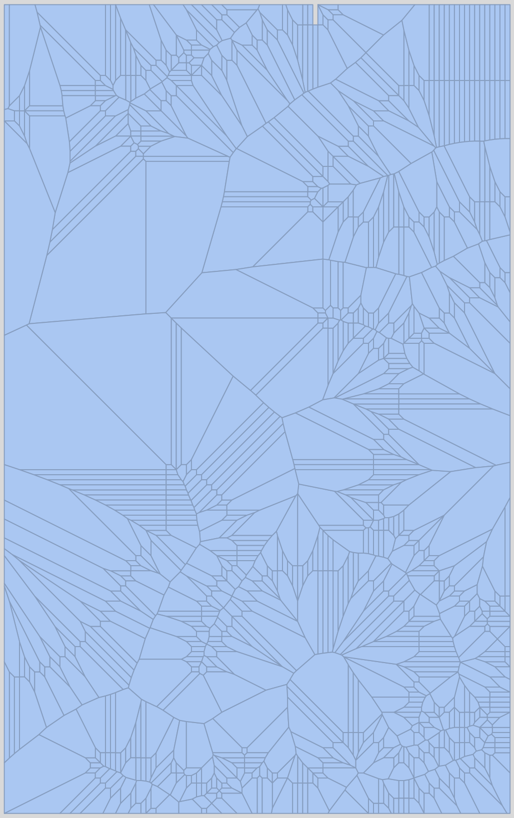
Out[13]=



Construct a Voronoi diagram based on the edge points, clipped to image size

In[14]:=
```
imgBounds = Transpose[{{0, 0}, ImageDimensions[Rmask]}];
vm = VoronoiMesh[edgePoints, imgBounds]
```

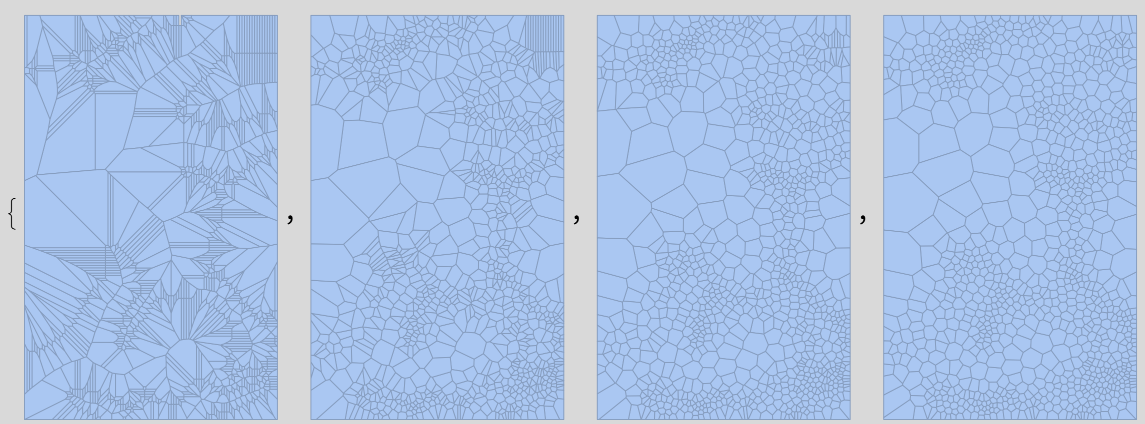
Out[15]=



The aspect ration of the Voronoi cells is quite large, and be reduced by constructing a mesh based upon the centers of the cells of the previous mesh

In[16]:=
```
vml = NestList[VoronoiMesh[Mean @@@ MeshPrimitives[#, 2], imgBounds] &, vm, 3]
```

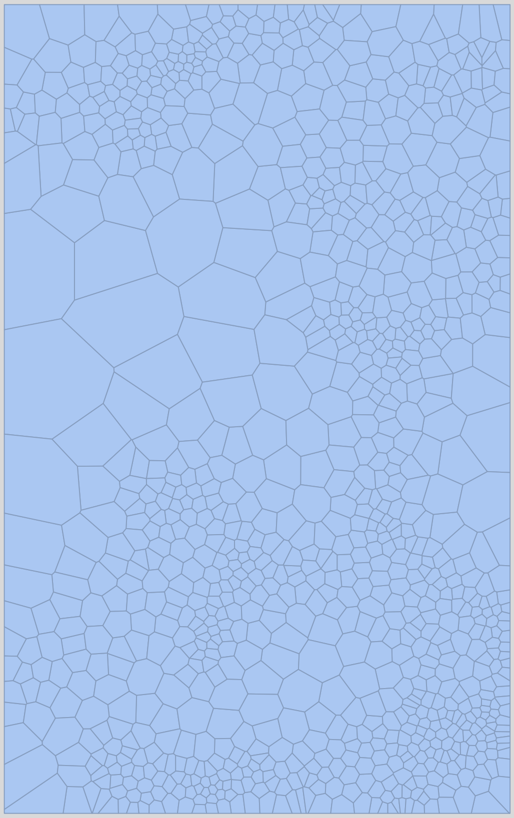
Out[16]=



Define the final, regularized Voronoi mesh

In[17]:=
```
vm = Last[vml]
```

Out[17]=



Extract the polygons of the final Voronoi mesh

In[18]:=
```
vmpoly = MeshPrimitives[vm, 2]
```

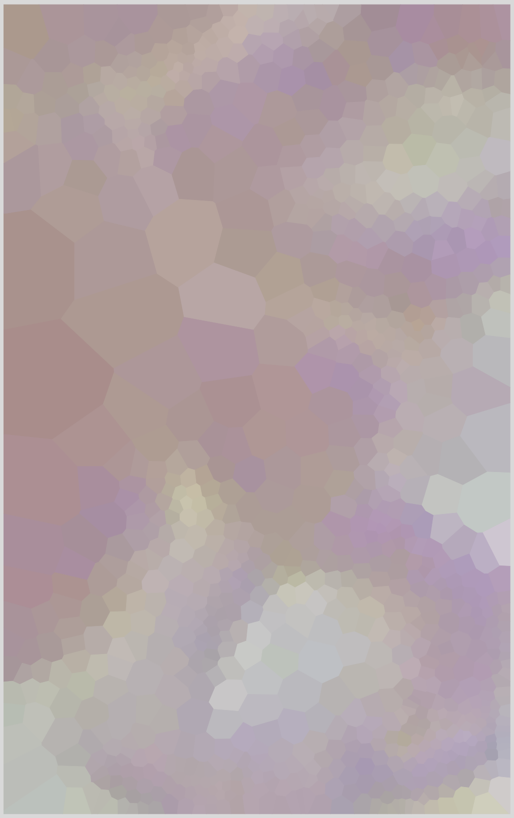
Out[18]=
```
{Polygon[{{62.9463, 160.}, {65.0694, 156.881}, {66.322, 160.}}],
 … 1016 …, Polygon[{{31.5407, 27.7119}, … 7 …, {29.7786, 23.7529}}]}
```

large output | show less | show more | show all | set size limit...

Associating each polygon with the RGB color value at its center gives a useful representation that combines the color data with the mesh.

In[19]:=
```
Graphics[Table[{RGBColor[ImageValue[im, Mean @@ p]], p}, {p, vmpoly}]]
```

Out[19]=



Find the polygon center positions and the image RGB values

In[20]:=
```
centers = Table[Mean @@ p, {p, vmpoly}];
```

In[21]:=
```
RGBs = Map[ ImageValue[im, #] &, centers];
```

Associate to each polygon center a third coordinate that corresponds to the difference between the Red and Blue intensity values (scaled by the image size)

In[22]:=
```
sc = Max[ImageDimensions[im]];
centerRmB[cen_, RGB_] := Append[cen, sc (RGB[[1]] - RGB[[3]])];
data = MapThread[centerRmB, {centers, RGBs}];
```

The interest is to link polygons that are close both in center position and in R-B values. Construct a function that returns a list of edges of points closer than distance d

In[25]:=
```
dist = Compile[ {{a, _Real, 3}, {b, _Real, 3}}, Norm[a - b]]
```

Out[25]=
```
CompiledFunction[ Argument count: 2
Argument types: {{_Real, 3}, {_Real, 3}} ]
```

In[26]:=
```
FindEdges[data_, d_] := Module[{i, j, l = {}, xy = {}, n},
  n = Length[data];
  Monitor[
   For[i = 1, i ≤ n, i++,
    For[j = i + 1, j ≤ n, j++,
     If[dist[data[[i]], data[[j]]] ≤ d,
      AppendTo[l, {i, j}];
      AppendTo[xy, {data[[i, 1 ;; 2]], data[[j, 1 ;; 2]]}]]
    ]
   ], {i, j}];
  Return[{l, xy}]]
```

As an example, compute the edges linking {x,y,R-B} values that are closer than 5 units, 10 units

In[27]:=
```
Off[CompiledFunction::cfta];
```

In[28]:=
```
{edgeij05, edgexy05} = FindEdges[data, 5];
```

In[29]:=
```
{edgeij10, edgexy10} = FindEdges[data, 10];
```

Define a function to superimpose upon the image the computed edges

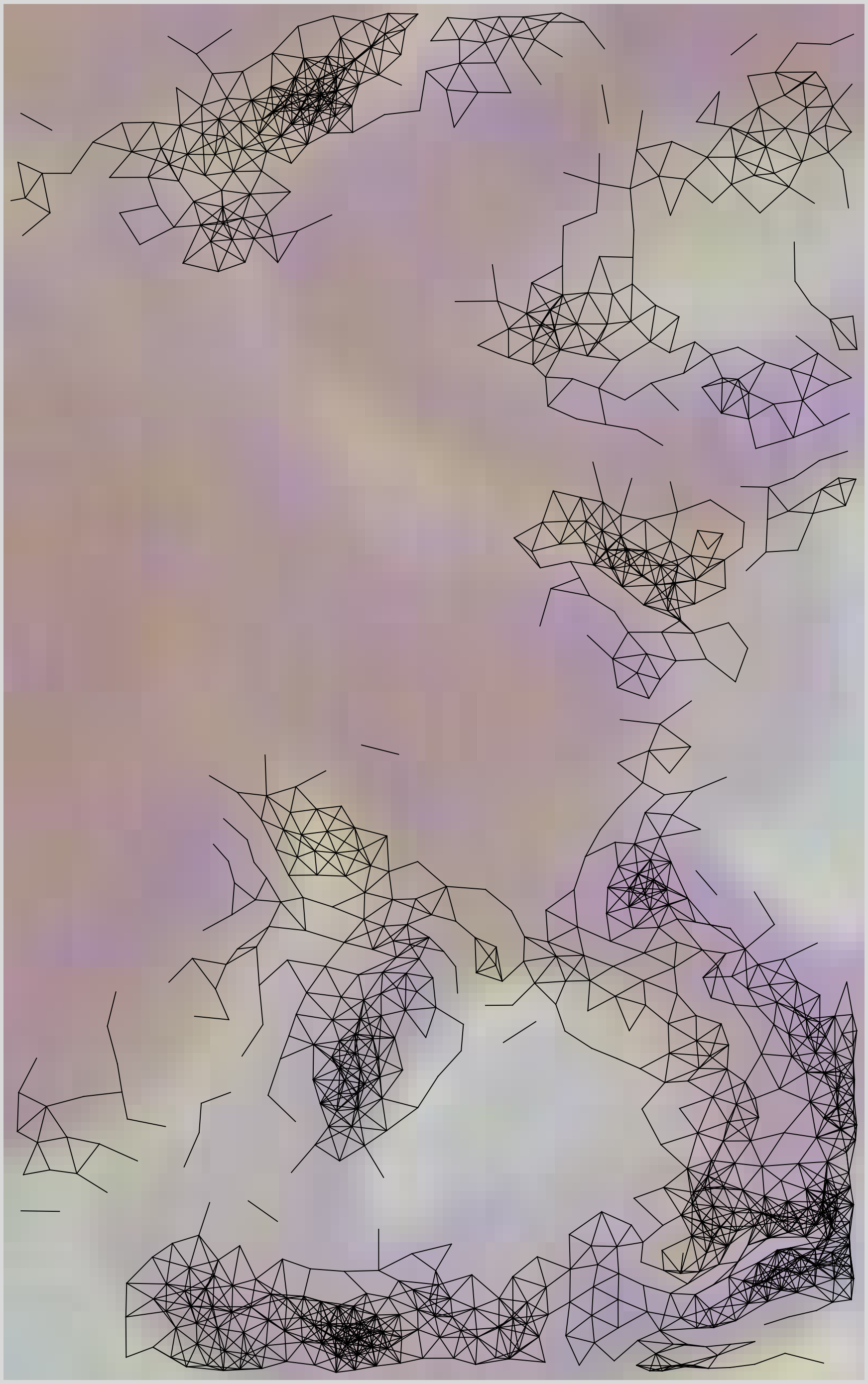
In[30]:=
```
ShowEdges[im_, edgexy_] :=
 Show[{im, Graphics[Map[Line[#] &, edgexy]]}, ImageSize → Large]
```

Display the edges obtained when linking nodes closer than 5 units

In[31]:=

```
ShowEdges[im, edgexy05]
```

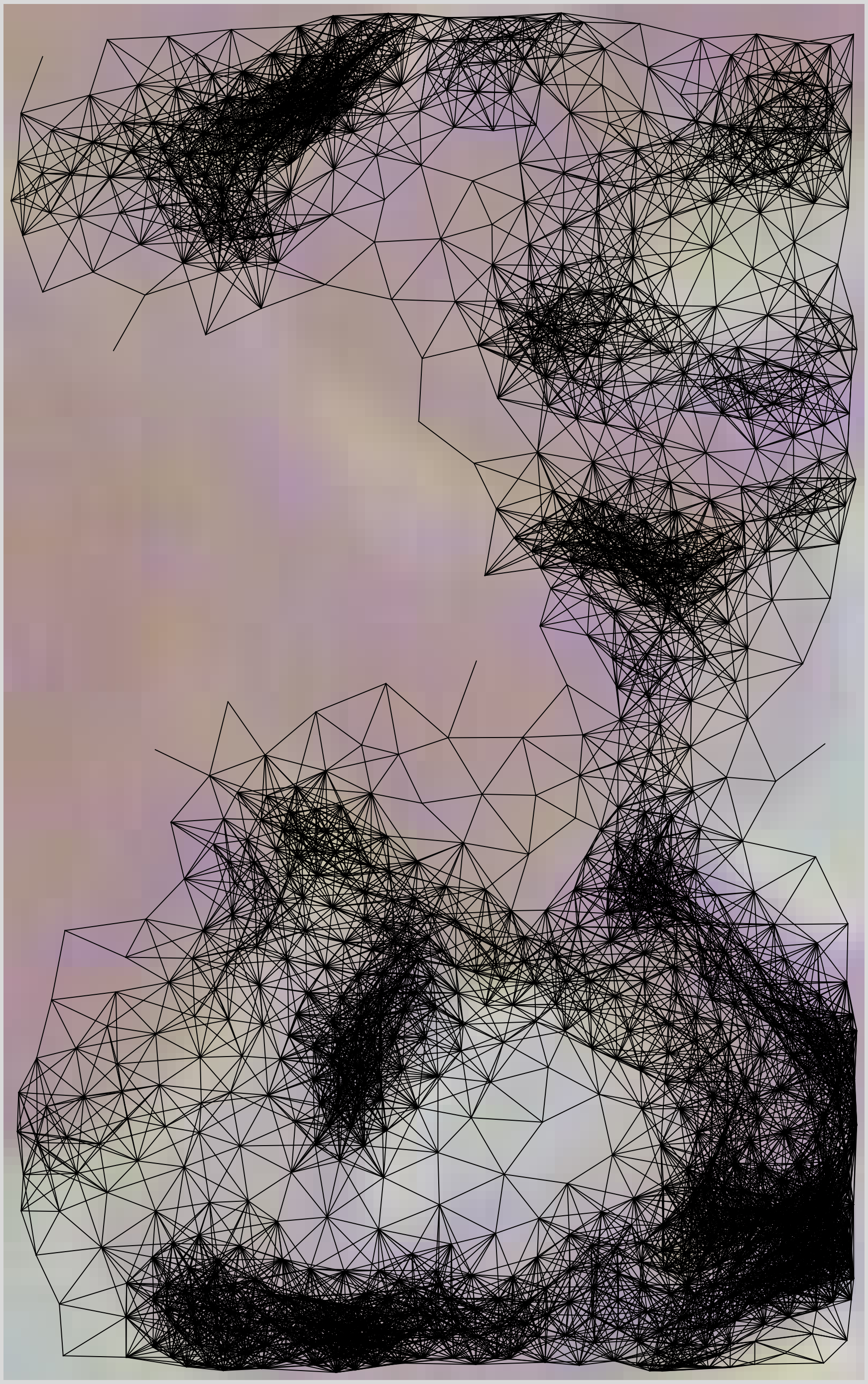
Out[31]=



In[32]:=

```
ShowEdges[im, edgexy10]
```

Out[32]=



Homework questions

Question 1

First, view Matthew Wright’s video on persistent homology (https://www.youtube.com/watch?v=2PSqWBIrn90).

Next, carefully read through steps in the Isolate erythrocytes section. In step In[22] a point cloud is constructed in the data vector. Explain why each point is chosen to have three coordinates instead of the two available in a two-dimensional image.

Question 2

Randomly choose a different image from the data base. Construct the superposition of the image with the edges obtained at cut-off distances d=1,2,5,10,20, using the scaling sc from step 22. How do you interpret regions with densely packed edges by comparison to regions with sparsely packed edges to those with no edges?

Question 3

Modify the scaling sc from step 22 to another value that would preferentially provide links between polygons within red blood cells. Using the new scaling reconstruct five superpositions of edges with the image. Choose appropriate cut-off distances d (they will be different from those in Question 2)

Question 4

The FindEdges function (In[26]) only computes 1-simplexes. Write a FindTriangles function that computes 2-simplexes formed by nodes {i,j,k} that all have pair-wise distances less than the cutoff. Also write a ShowTriangles function that is similar to ShowEdges (In[30]), but superimposes triangles (use Mathematica Triangle function) on the image instead of lines. Exhibit a superimposed image

Question 5

With the rescaling chosen in Q3, construct superposition of triangles with the image at five different cut-off distances of your choice. How would you interpret holes in the simplicial complex?

In[33]:=
```
{edgeij02, edgexy02} = FindEdges[data, 2];
```

In[35]:=
```
{edgeij20, edgexy20} = FindEdges[data, 20];
```

In[36]:=
```
ShowEdges[im, edgexy20]
```

Out[36]=