Voronoi Clustering

John Burkardt (ARC/ICAM)
Virginia Tech

.....

Math/CS 4414:
"Voronoi Clustering"
http://people.sc.fsu.edu/~jburkardt/presentations/
clustering_voronoi.pdf

.....

ARC: Advanced Research Computing
ICAM: Interdisciplinary Center for Applied Mathematics

28 September 2009

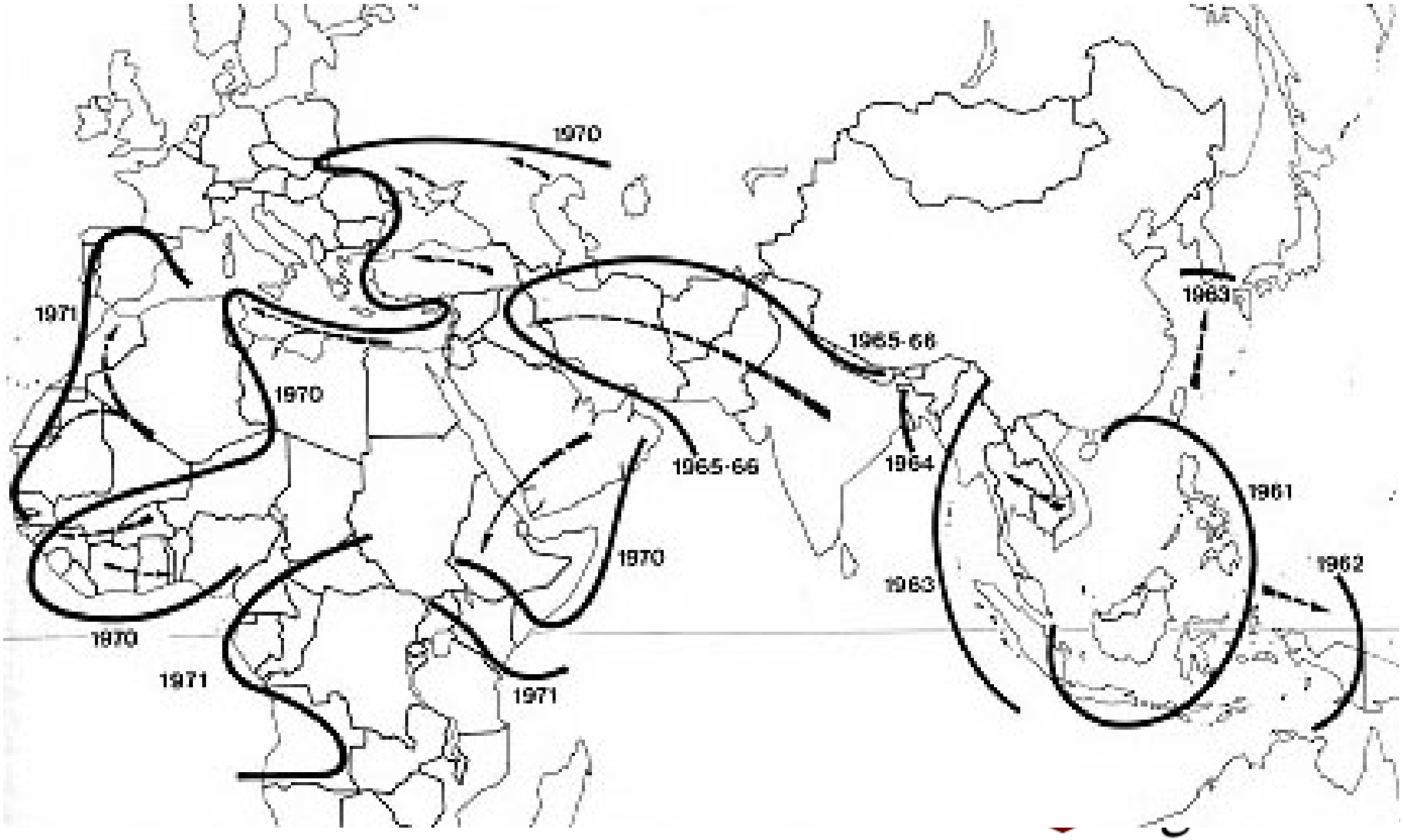
VirginiaTech

Voronoi Clustering

1. **Death in Golden Square**
2. The Voronoi Diagram
3. Voronoi Computation
4. Centered Voronoi Diagrams
5. Conclusion

VirginiaTech

Golden Square: The Spread of Cholera



Golden Square: The Spread of Cholera

In the early nineteenth century, European doctors working in India reported a strange new disease called **cholera**.

Cholera was agonizing and fatal. It would strike a village, first sickening a few people, then more and more. Most of the victims would die within three days.

Suddenly, the disease would be gone from the village - but it hadn't disappeared - it would suddenly break out in a few nearby villages, and repeat its destruction.

Then there came reports of cholera in Persia, and Turkey. The disease was moving, and it was heading straight for England.

VirginiaTech

Golden Square: The Death Map

Golden Square: The Death Map

The picture illustrates the theory that cholera was transmitted by **miasm**: an invisible evil-smelling cloud of disease.

Miasm explained why cholera victims often lived in poor areas full of tanneries, butcher shops, and general dirty conditions.

It also explained why cholera did not simply spread out across the entire population, but seemed to stay in one location for a time, devastate a neighborhood, then move on.

It also suggested that little could be done to prevent the disease.

Only 30 years later would the actual cause of cholera be identified.

VirginiaTech

Golden Square: The Cholera Bacteria

VirginiaTech

Golden Square: Dr John Snow

Dr John Snow argued that the miasm theory was unscientific.

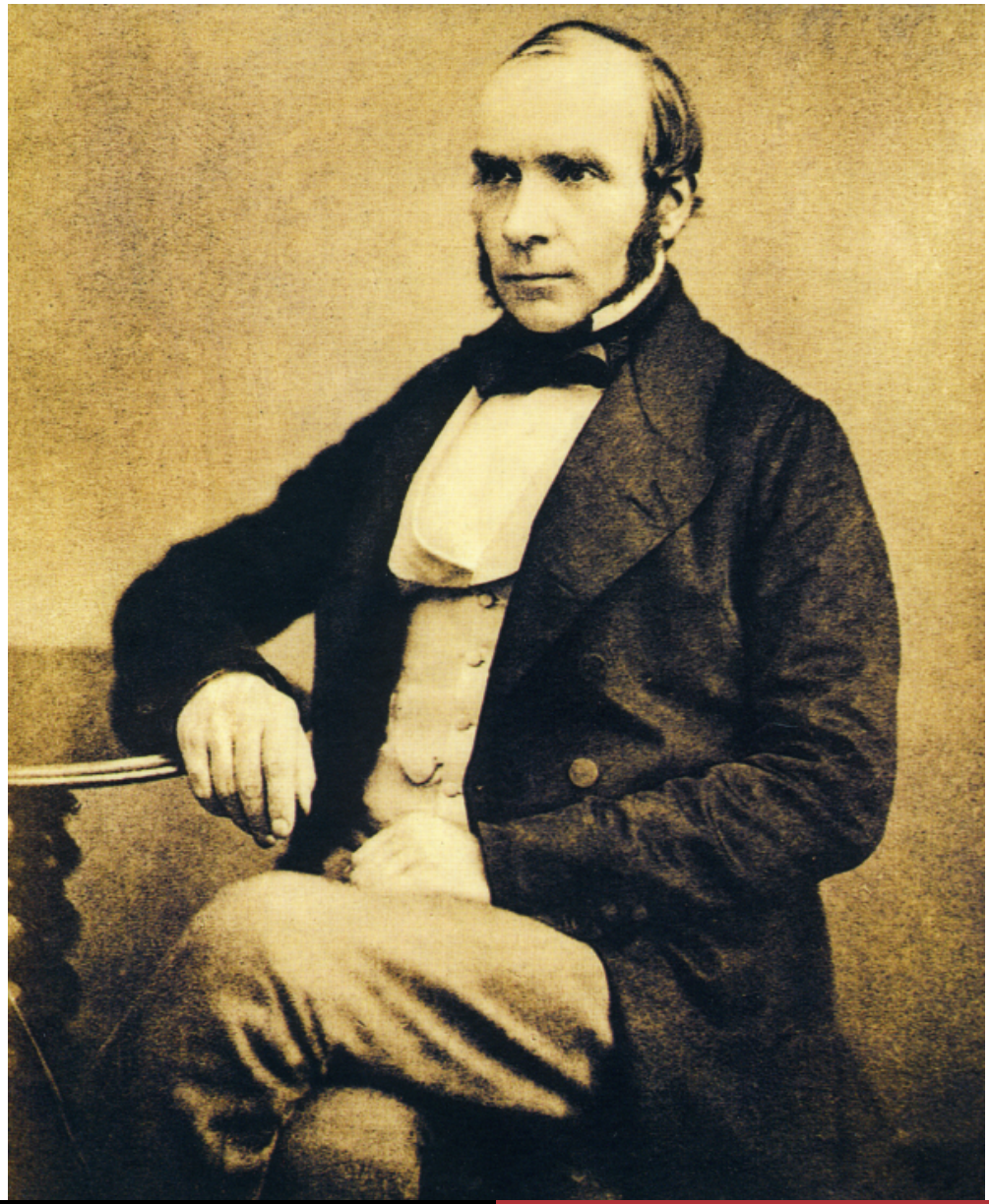
Even though he could not show what was causing the disease, he felt strongly that it was transmitted **by water**.

Without being able to see bacteria, this is somewhat like trying to do physics without any proof of atoms!

Dr Snow was trying to form an improved model of disease, that explained the facts he knew, and might also tell him how to prevent disease.

VirginiaTech

Golden Square: Dr John Snow



Golden Square: Evidence That Miasms Were a Bad Model

The miasm theory didn't satisfy Dr Snow because it didn't try to explain the patterns he had noticed.

In one small outbreak of cholera, only people living near the Thames river got sick. To Dr Snow, this suggested something in the river water. To the miasm defenders, there must have been a disease cloud floating along the low-lying river areas.

A second outbreak occurred away from the river, in an area with piped-in water. People getting water from one company were healthy, while others got sick. The intake pipes for that one company were located upstream from London. Again, the miasm defenders were not impressed by the small amount of data.

VirginiaTech

Golden Square: Cholera Outbreak of 1854

rginiaTech

Golden Square: How Water Was Polluted

For most people in London, there was **no running water**. For drinking, cooking, cleaning and bathing, they went to one of the town pumps which drew water directly from the ground.

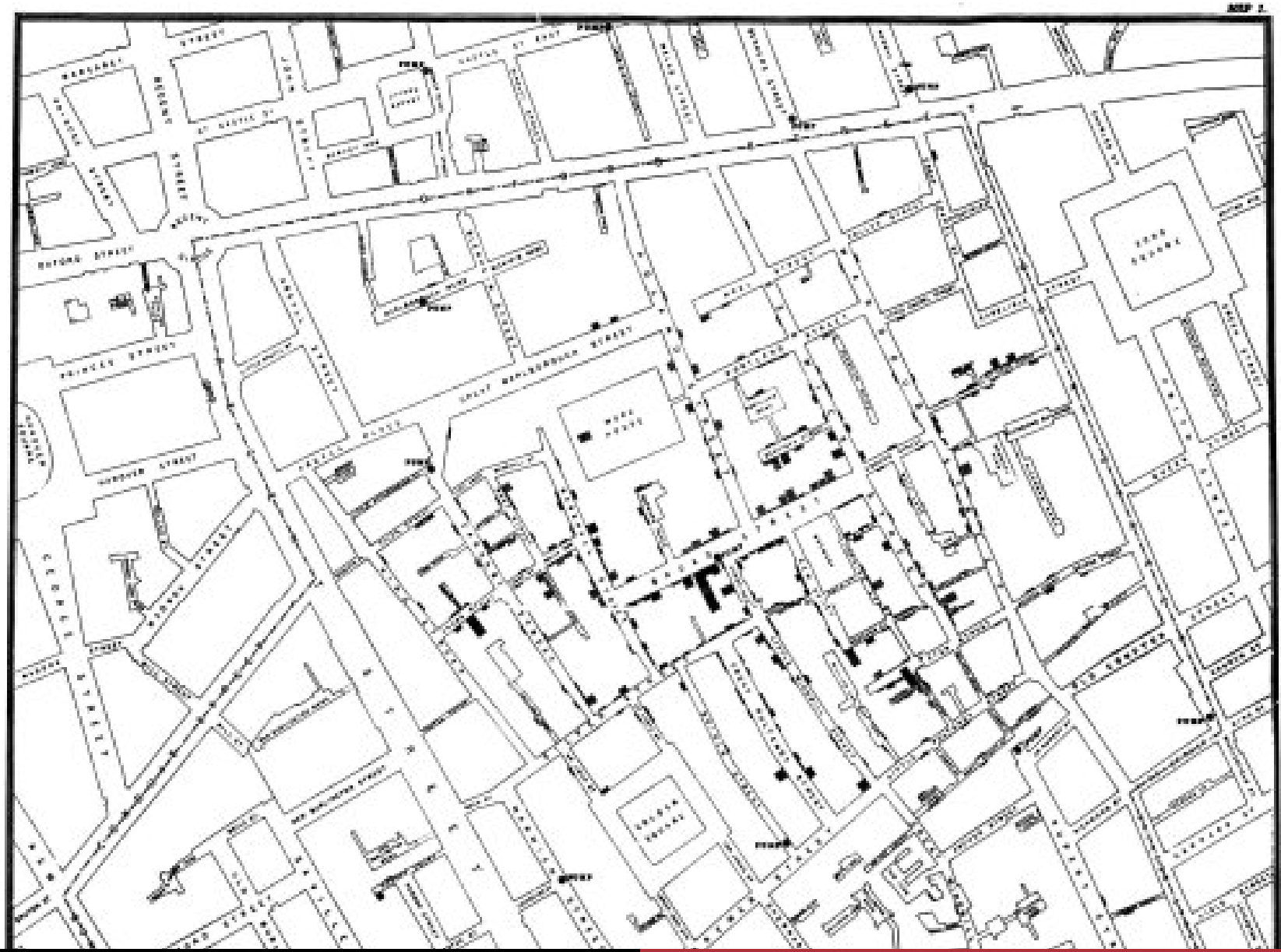
Most people had **no sewage system**. People used chamberpots and buckets, which were emptied into cisterns and carted off by an informal network of *night-soil carters*.

People were used to getting sick from the water now and then.

But when cholera bacteria arrived in London, and seeped from a cistern into the water, people died by the hundreds.

VirginiaTech

Golden Square: A Map



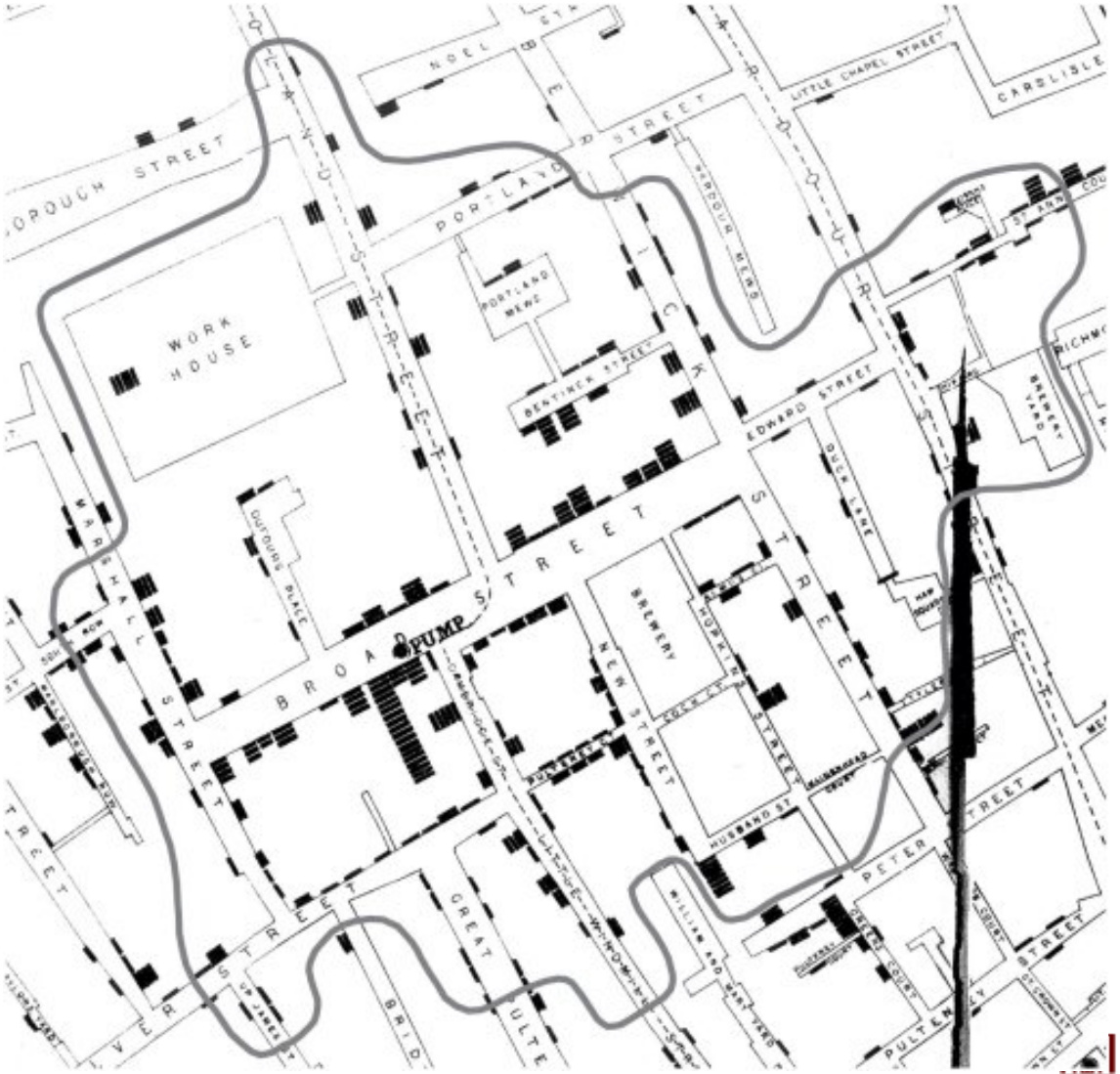
Golden Square: John Snow's Investigation

Dr John Snow suspected the water pump on Broad Street, but he needed evidence (no one knew about germs):

- He made a map of the district.
- He marked every house where cholera victims had lived;
- He paced the distance to the nearest pumps;
- The houses closest to the Broad Street pump were circled by a black line.

VirginiaTech

Golden Square: The Pump and its Neighborhood



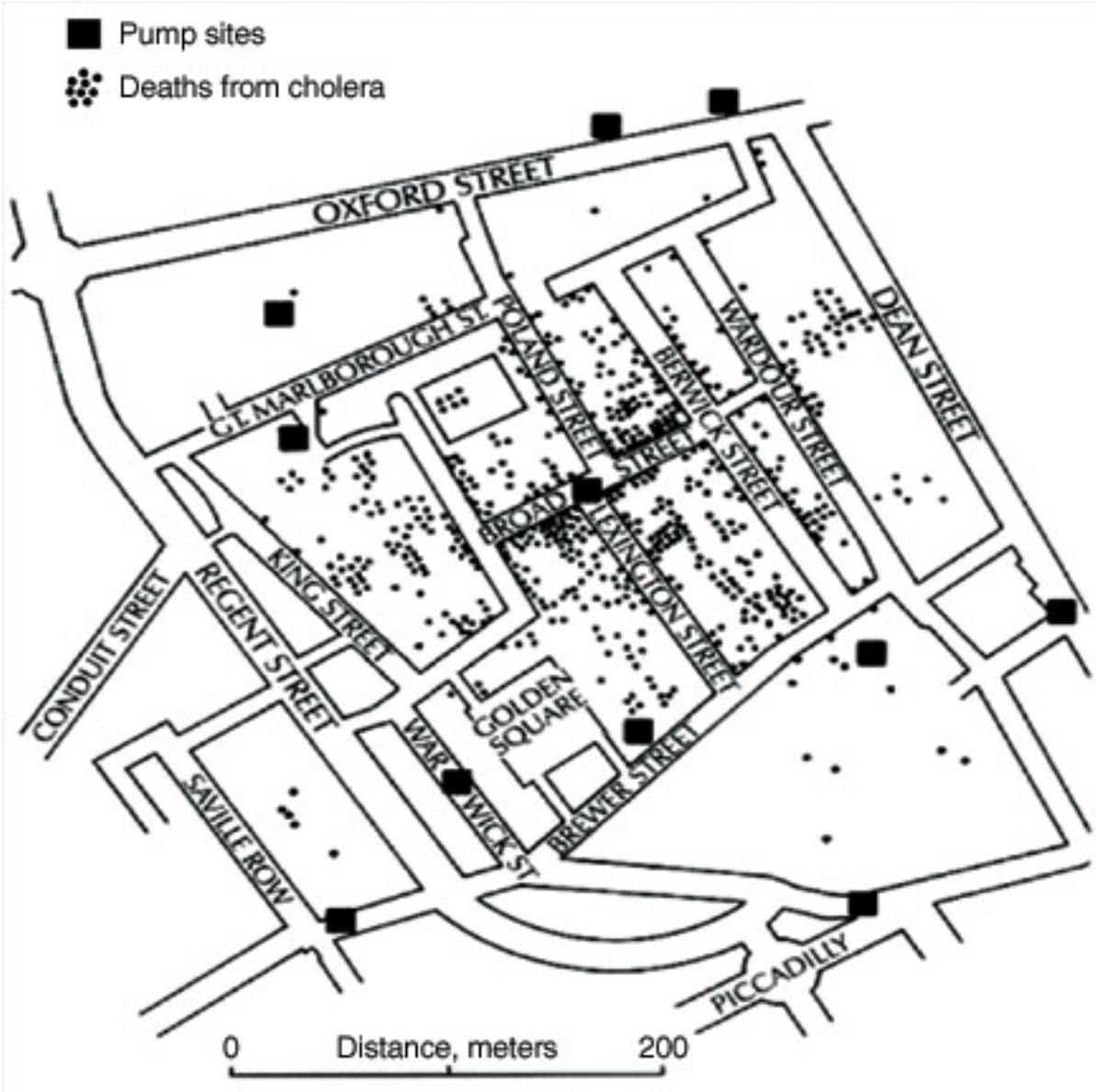
Golden Square: John Snow's Investigation

He personally interviewed people to explain exceptions to the pattern he found:

- healthy workers at a beer factory didn't drink any water!;
- some sick children lived outside the district, but came to school there;
- a woman who lived miles away had her son deliver "fresh" water from the Broad Street pump because she liked its taste.

VirginiaTech

Golden Square: See the Pumps as "Suspects"



VirginiaTech

Golden Square: Conclusion

Snow's map strongly suggested that people who died were precisely those for whom the Golden Square pump was the closest.

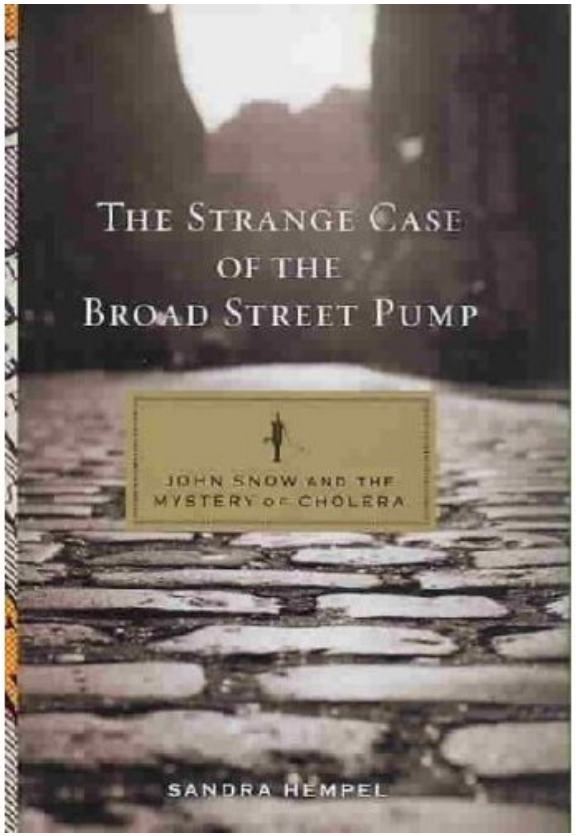
The miasmic cloud theory required the miasm, purely by chance, to settle only over people who used the Golden Square pump.

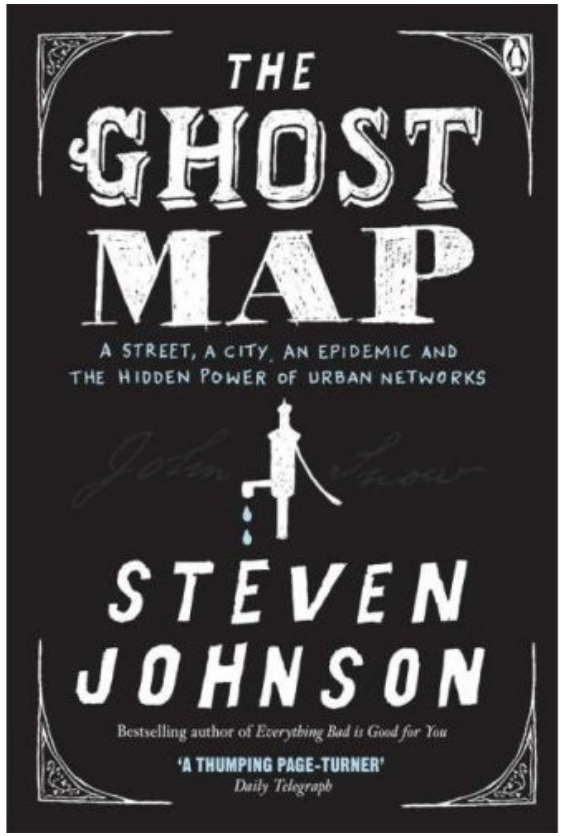
Dr Snow's map destroyed the miasm theory, because it could show **where** deaths occurred in a way that suggested **why**.

Dr Snow was doing epidemiology, but also mathematics.
His map is an interesting example of a **Voronoi diagram**.

VirginiaTech

Golden Square: Two Books





"The Strange Case of the Broad Street Pump" by Sandra Hempel.
"The Ghost Map" by Steven Johnson

VirginiaTech

Voronoi Clustering

1. Death in Golden Square
2. **THE VORONOI DIAGRAM**
3. Voronoi Computation
4. Centered Voronoi Diagrams
5. Conclusion

VirginiaTech

Voronoi Diagram: Definition

Suppose that in some **D**-dimensional region $\mathcal{R}$

- we have a set $\mathcal{P}$ of "data points";
- we have a set $\mathcal{C}$ of "centers";
- we have a distance function $d(p_i, c_j)$.

A **Voronoi diagram** $\mathcal{V}(\mathcal{P}, \mathcal{C}, d(*, *))$ assigns each data point $\mathbf{p_i}$ to the center $\mathbf{c_i}$ with smallest distance $d(p_i, c_j)$.

This induces a partition of the data points into clusters.

VirginiaTech

Voronoi Diagram: MATLAB Commands

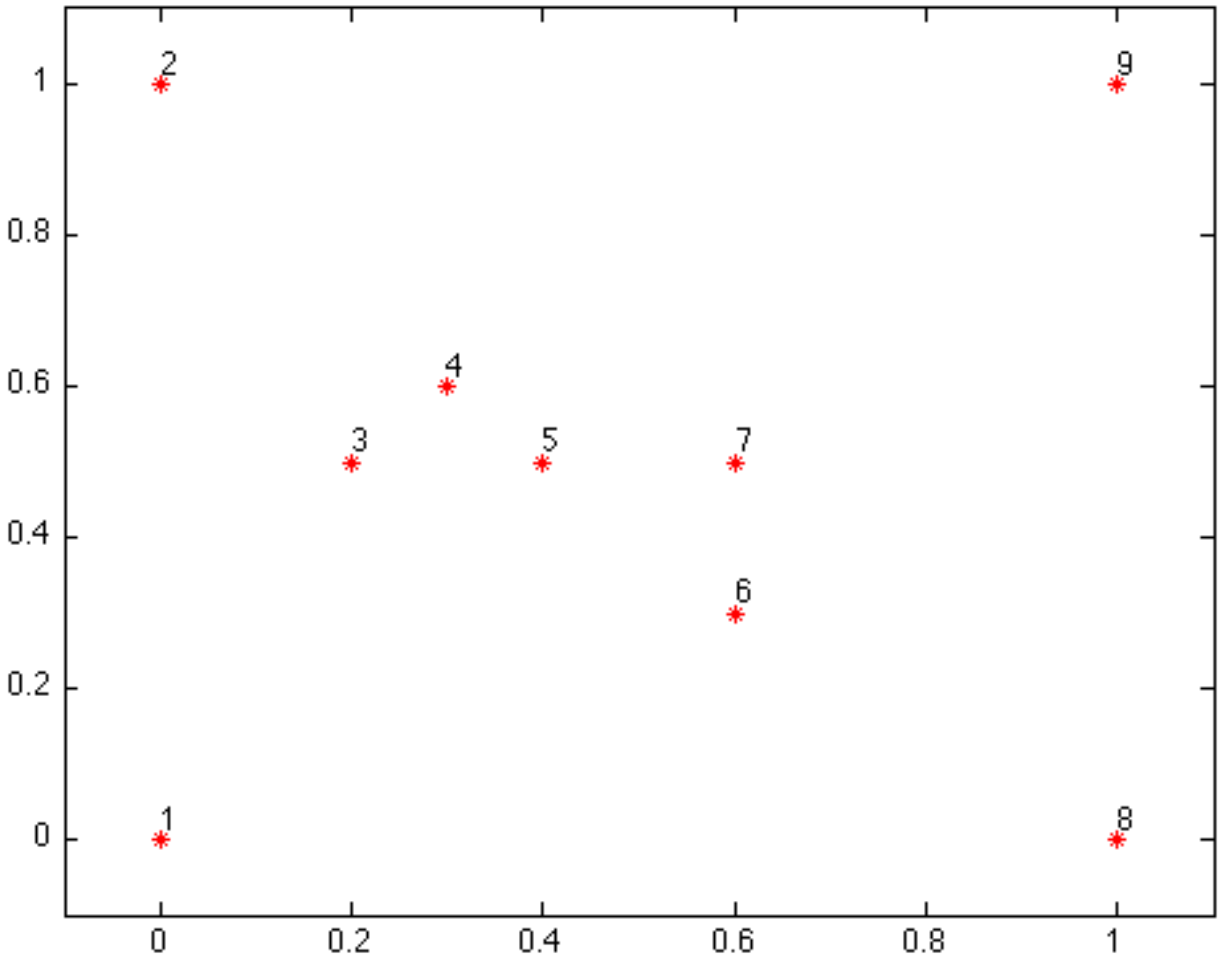
MATLAB can read a file of data and plot the Voronoi diagram.

The **size** command is useful so we can check that we are indexing the array of points correctly.

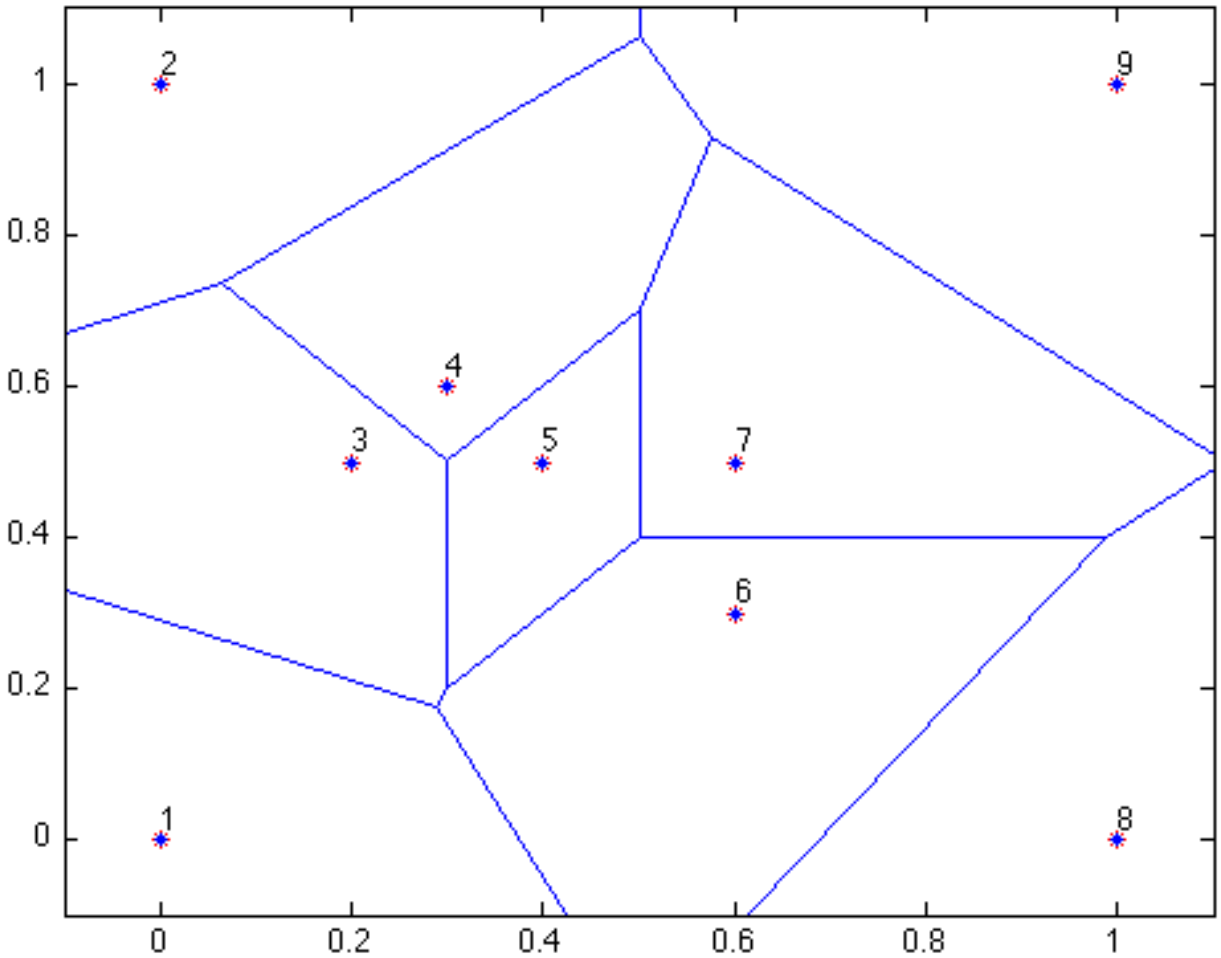
```
p9 = load ( 'p9.txt' );
size ( p9 );

plot ( p9(:,1), p9(:,2), 'r*' )   <-- See the points
text ( p9(:,1), p9(:,2)+0.02, '1|2|3|4|5|6|7|8|9' )
axis ( [ -0.1, 1.1, -0.1, 1.1] )  <-- Improve the view

voronoi ( p9(:,1), p9(:,2) )
```

VirginiaTech

Voronoi Diagram: 9 points



Voronoi Diagram: 9 point Voronoi diagram



Voronoi Diagram: Properties of the Voronoi Regions

For the continuous case, it can be more natural to refer to *regions* rather than *clusters*; in both cases, we mean the subsets of the data points that are closest to a particular center.

Assuming we are using the ℓ_2 distance function,

- All regions are convex (no indentations);
- All regions are polygonal;
- Each region is the intersection of half planes;
- The infinite regions have centers on the convex hull.

VirginiaTech

Voronoi Diagram: Properties of the Voronoi Edges

The Voronoi **edges** are the line segments of the boundary;

If you think of the Voronoi regions as defined by expanding a circle from the center, an edge occurs when two neighboring circles smash into each other.

- Each edge bisects the line connecting two centers;
- each edge contains points equidistant from 2 centers;
- Two centers have a common boundary if and only if they are connected in the Delaunay triangulation.

VirginiaTech

Voronoi Diagram: Properties of the Voronoi Vertices

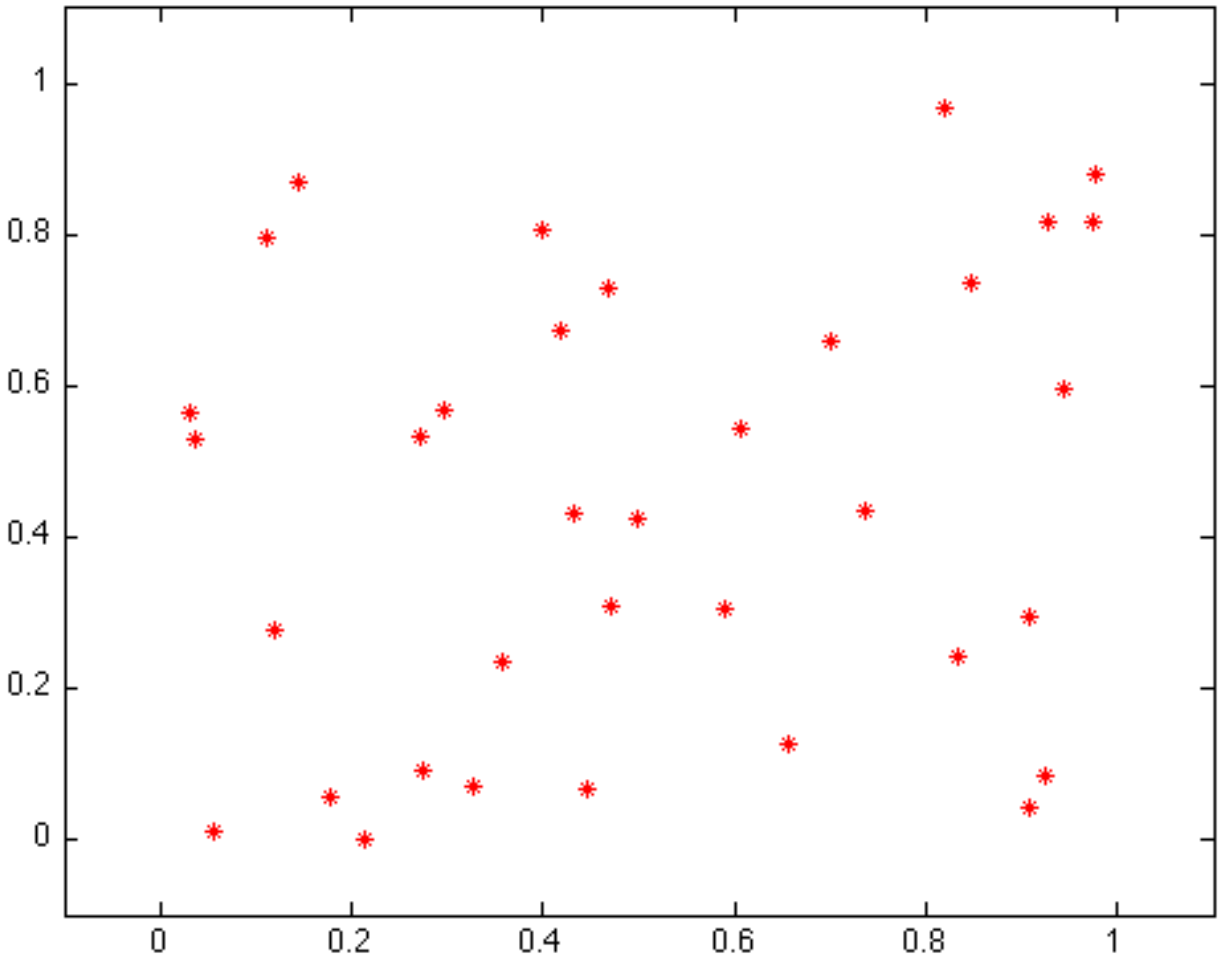
The Voronoi **vertices** are end points of the edges.

If you think of the Voronoi edges as defined by two neighboring circles smashing into each other, a vertex occurs when a third neighbor circle joins the smashing.

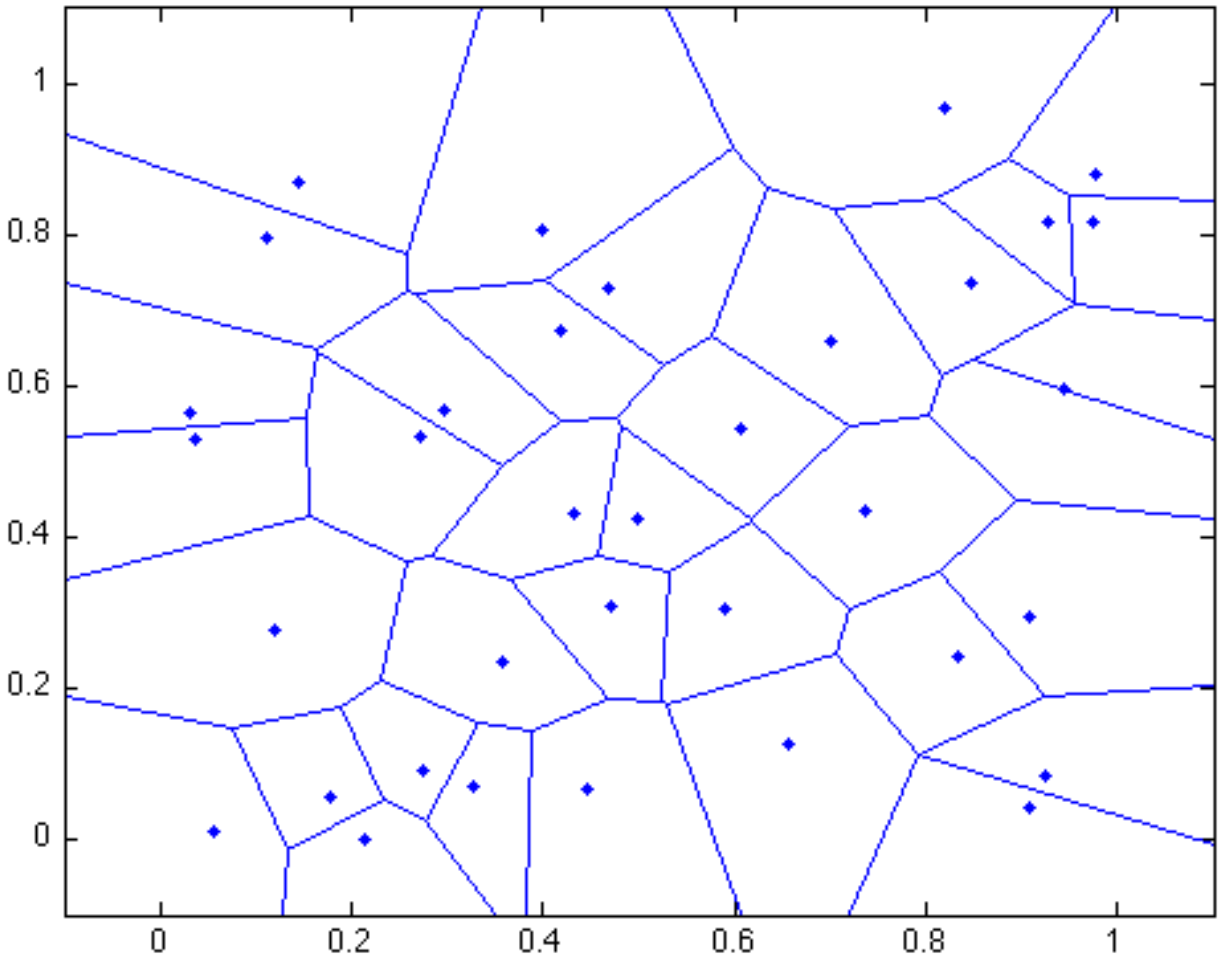
- a vertex is the endpoint of 3 edges;
- a vertex is equidistant from 3 centers;
- a vertex is the center of a circle through 3 centers;
- a vertex circle is empty (contains no other centers).

VirginiaTech

Voronoi Diagram: 36 points



Voronoi Diagram: 36 points Voronoi diagram



Voronoi Diagram: Related Problems

Most algorithms and software for the Voronoi diagram only work in the (infinite) plane with the Euclidean distance.

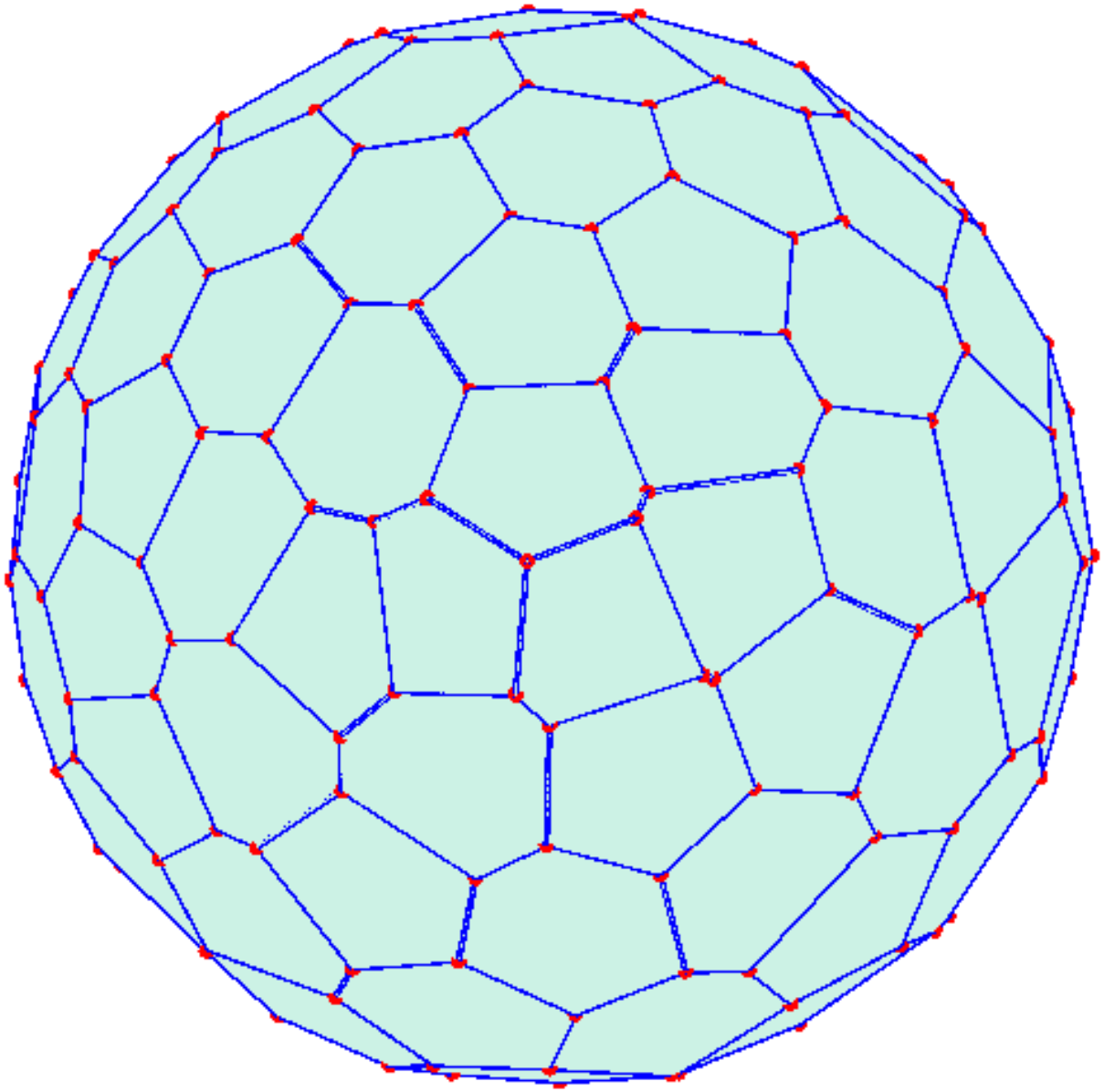
It's easy to want to extend these ideas to related problems:

- Restrict to some finite shape (perhaps with holes);
- "Wrap-around" regions (like a torus, or Asteroid);
- Define Voronoi diagrams on a surface;
- Problems in higher dimensions.
- Include a "density" function;

The sphere is one important region for which exact and approximate Voronoi algorithms have been developed.

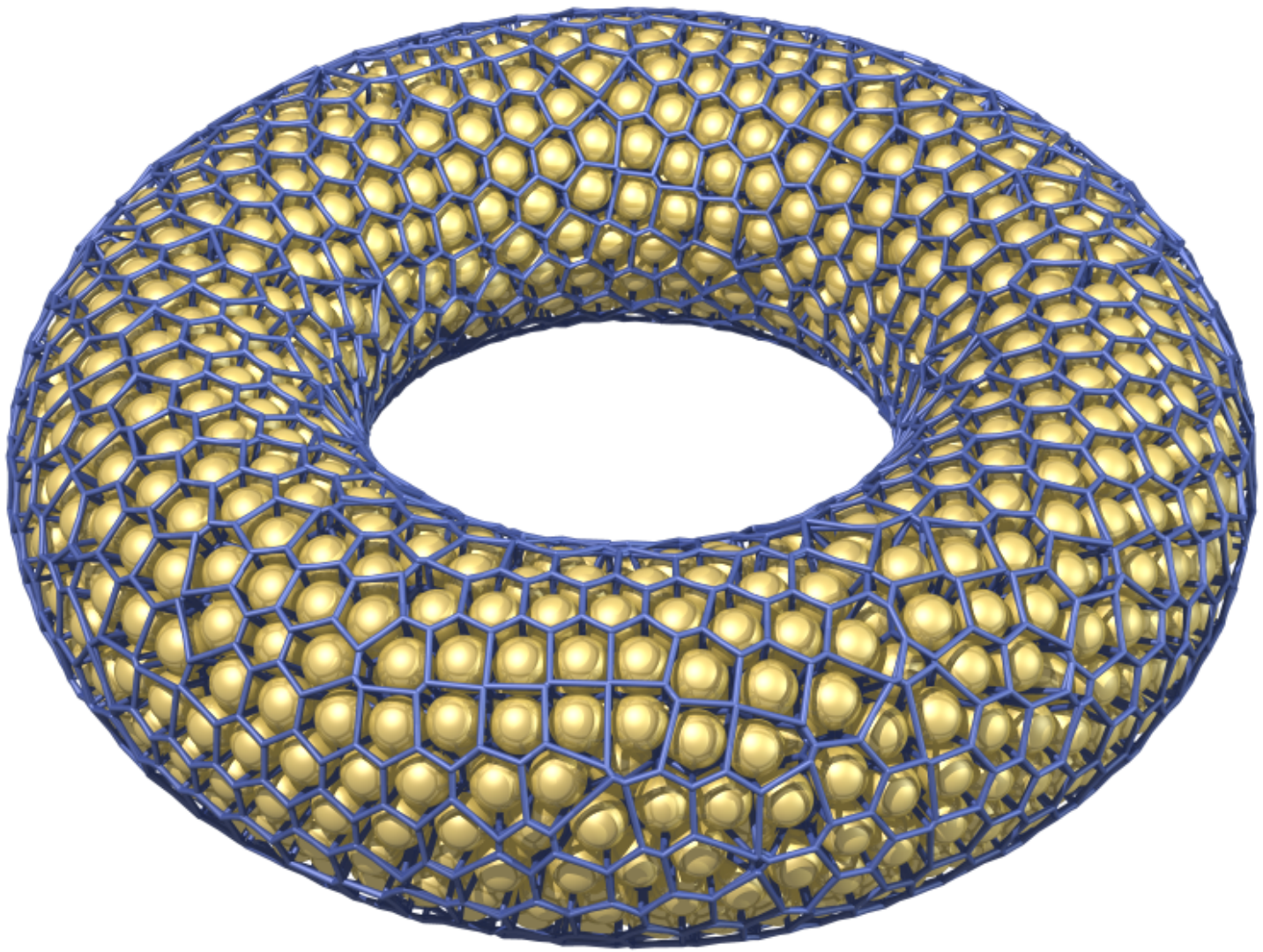
VirginiaTech

Voronoi Diagram: On a Sphere, by the Stripack Program



VirginiaTech

Voronoi Diagram: On a Torus, by the Voro++ Program



iaTech

Voronoi Clustering

1. Death in Golden Square
2. The Voronoi Diagram
3. **VORONOI COMPUTATION**
4. Centered Voronoi Diagrams
5. Conclusion

VirginiaTech

Voronoi Computation

In some ways, a Voronoi diagram is just a **picture**; in other ways, it is a **data structure** which stores geometric information about boundaries, edges, nearest neighbors, and so on.

How would you compute and store this information in a computer?

It's almost a bunch of polygons, but of different shapes and sizes and orders - and some polygons are "infinite".

If you need to compute the exact locations of vertices and edges, then this is a hard computation, and you need someone else's software.

VirginiaTech

Voronoi Computation: A Geometric Calculation

MATLAB's **voronoi** command returns the vertices:

[vx,vy] = voronoi (px, py)

The first Voronoi edge: (vx(1,1),vy(1,1)) to (vx(2,1),vy(2,1));

The 17-th Voronoi edge: (vx(1,17),vy(1,17)) to (vx(2,17),vy(2,17));

The semi-infinite Voronoi edges are "suggested" by finite line segments.

A suitable **plot** command can draw the edges.

VirginiaTech

Voronoi: MATLAB Commands

Here we use two **plot** commands to show the centers and the edges. It's what the **voronoi** command would show us ordinarily.

```
p9 = load ( 'p9.txt' );

plot ( p9(:,1), p9(:,2), 'r*' )   <-- See the points

[ vx, vy ] = voronoi ( p9(:,1), p9(:,2) );
hold on
plot ( vx(1:2,:), vy(1:2,:) )     <-- See the edges

axis ( [ -0.1, 1.1, -0.1, 1.1] )  <-- Improve the view
```

VirginiaTech

Voronoi Computation: Algorithms

Prewritten software is not needed if you only want to see an approximate picture of a Voronoi diagram, and you don't need the details,

And if you are working on an unusual geometry, higher dimensions, strange distance functions, or have other conditions, there usually isn't any prewritten software available.

This is a situation in which a *sampling method* can be useful.

VirginiaTech

Voronoi Computation: Voronoi Plot by Sampling

To make an **M** by **N** image **A(1:M,1:N)** of a Voronoi diagram in the unit square:

- choose **K** distinct colors **RGB(1:K)**, one for each center **C**;
- use the map (I,J) $\rightarrow$ (X,Y) = ((J-1)/(N-1),(M-I)/(M-1));
- for every pixel (I,J), compute the (X,Y) coordinates;
- find **C***, the center which is closest to (X,Y);
- set **A(I,J)** = **RGB(C*)**.

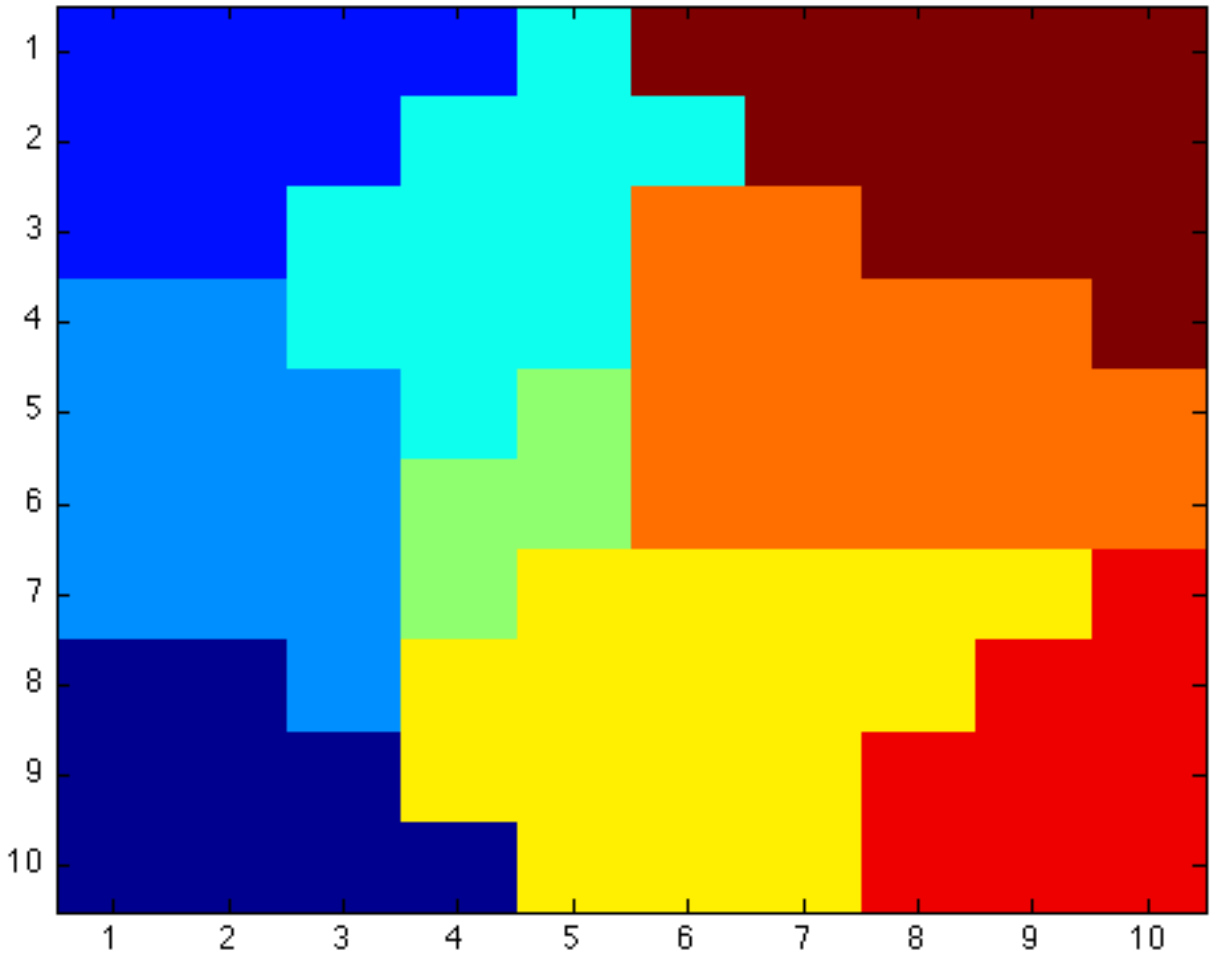
VirginiaTech

Voronoi Computation: Pixel Plot

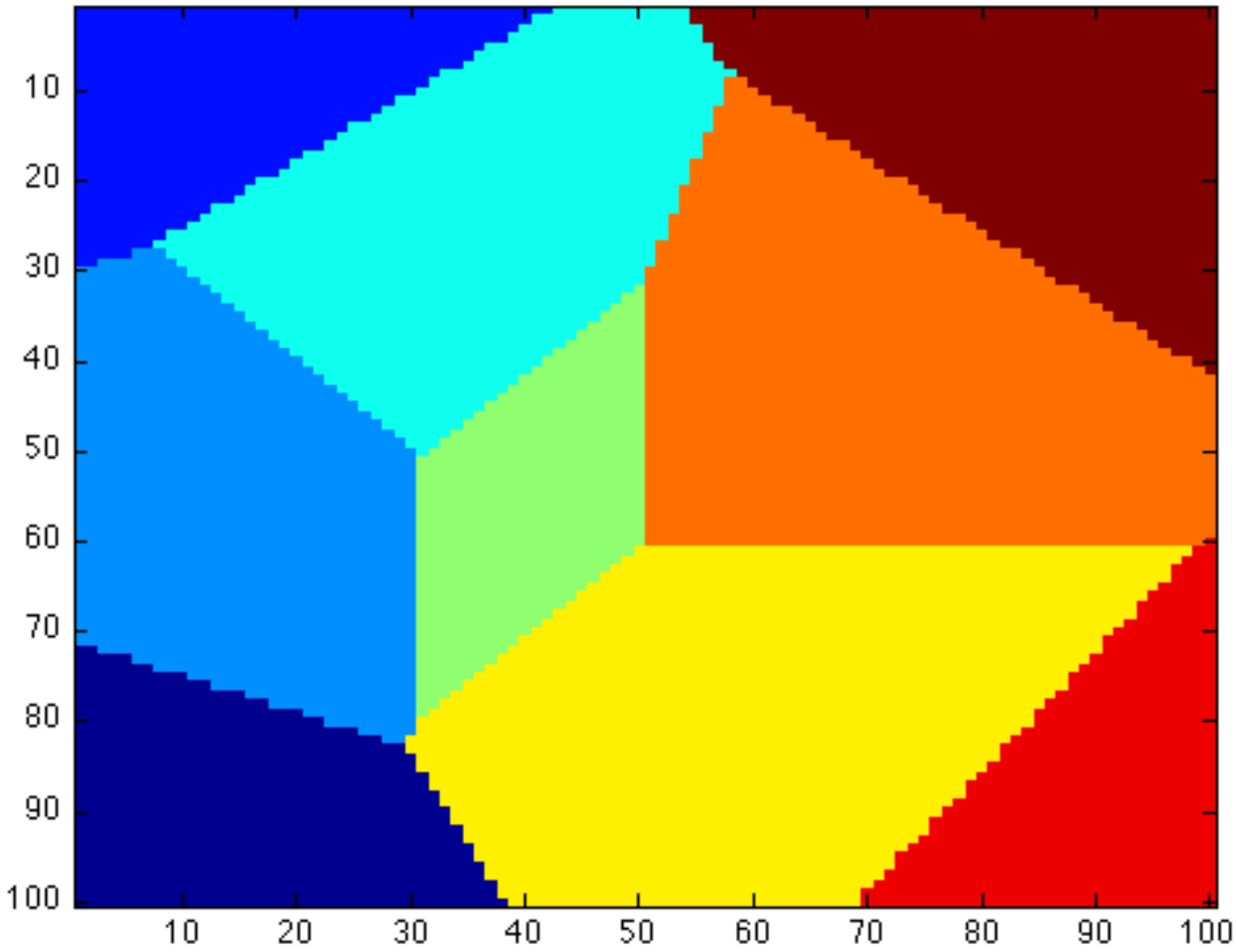
```
function a = pixel_plot ( m, n )

  p = load ( 'p9.txt' );
  k = size ( p, 1 );
  pc(1:k) = uint8 ( 1 : k );
  a = zeros ( m, n );

  for i = 1 : m
    y = ( m - i ) / ( m - 1 );
    for j = 1 : n
      x = ( j - 1 ) / ( n - 1 );
      dmin = Inf;
      closest = 0;
      for kk = 1 : k
        d = norm ( p(kk,1:2) - [ x, y ] );
        if ( d < dmin )
          dmin = d;
          closest = kk;
        end
      end
      a(i,j) = pc(closest);
    end
  end

  imagesc ( a );

  return
end
```

VirginiaTech

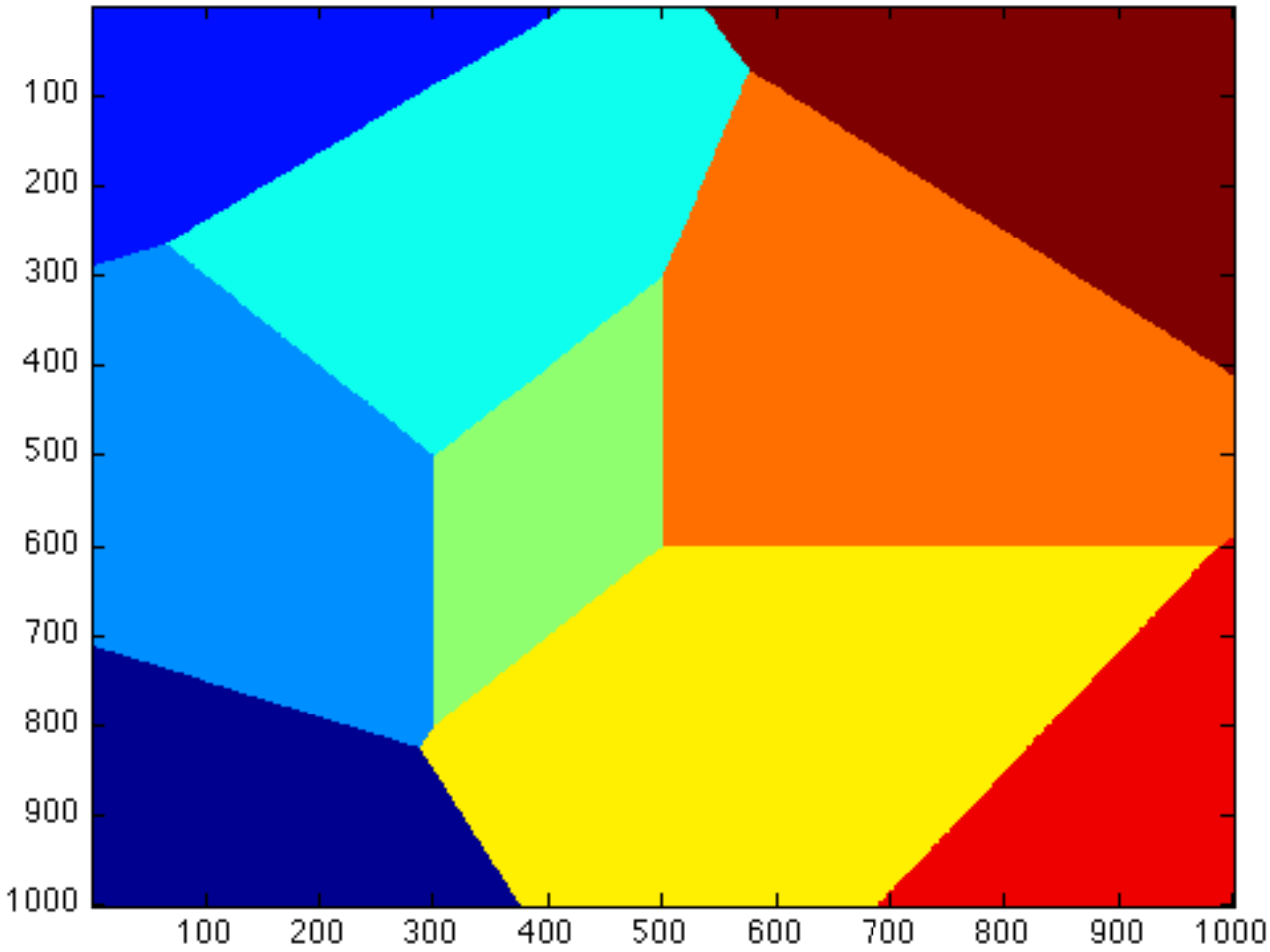
Voronoi Computation: pixel_plot(10,10)



Voronoi Computation: pixel_plot(100,100)



Voronoi Computation: pixel_plot(1000,1000)



VirginiaTech

Voronoi Computation: Advantages of Sampling Method

Now that we have a code that works (well enough, anyway), we can easily do any of the following:

- estimate the area of the Voronoi regions;
- double the distance value whenever center 5 is involved;
- make a map in which each point chooses the **furthest** center;
- replace Euclidean distance by L1 or L∞ distance;

VirginiaTech

Voronoi Computation: Sampling Method for Spheres

To use the sampling method on a sphere, some new issues arise:

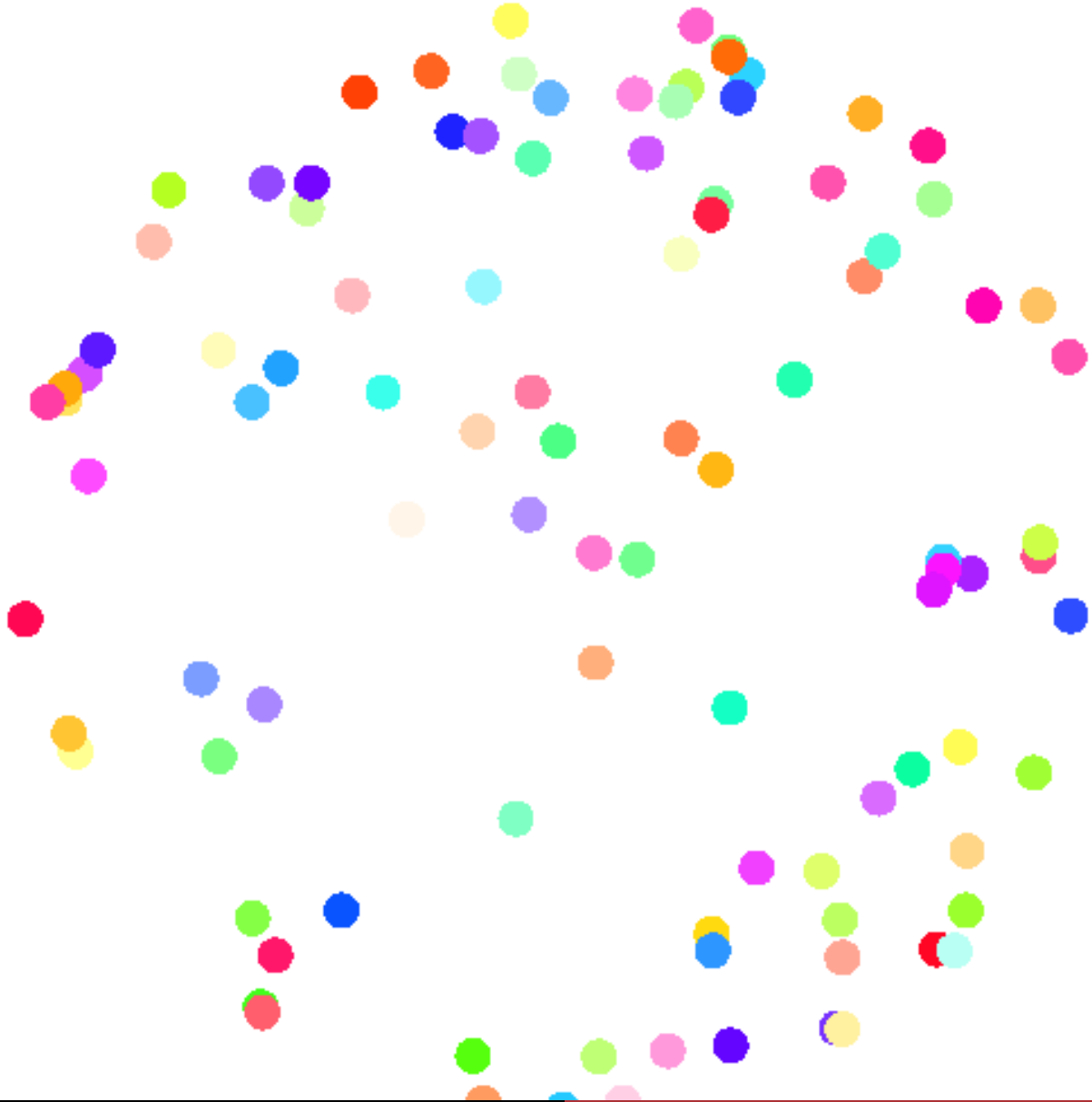
- how can we choose points uniformly at random;
- the pixel array is not the surface of the sphere, but a projection of some of it;
- figure out the formula for distance along the surface (it's essentially an angle)

We will now run an interactive program that suggests how a Voronoi diagram on a sphere can be drawn by sampling.

sphere_voronoi_display_open_gl 100

VirginiaTech

Voronoi Computation: Sampling Method for Spheres



irginiaTech

Voronoi Computation: Sampling Method for Spheres

'irginiaTech

Voronoi Computation: ASSIGNMENT

Download the program **pixel_plot.m** and the data file **p9.txt**.

Modify it to estimate the area of each of the Voronoi regions.

Start by creating a vector: **area = zeros(k,1);**

If a pixel is assigned to cluster **closest**:
area(closest) = area(closest) + 1;.

Your areas are in terms of pixels. Convert this to an area in "square inches" or whatever units we are using for the unit square.

Turn in 11 numbers: the pixel sizes you used **M** and **N**, and the 9 areas. This assignment is due by Friday, 2 October.

VirginiaTech

Voronoi Clustering

1. Death in Golden Square
2. The Voronoi Diagram
3. Voronoi Computation
4. **CENTERED VORONOI DIAGRAMS**
5. Conclusion

VirginiaTech

Centered Voronoi Diagrams

The Voronoi diagram seems to exhibit the ordering imposed by a set of center points that attract their nearest points.

However, since the placement of the centers is arbitrary, the overall picture can vary a lot. The shape and size of the regions in particular seem somewhat random.

Is there a way that we can impose more order?

Yes! ... if we are willing to allow the centers to move.

The Voronoi diagram is really a "snapshot" of the situation at the beginning of one step of the continuous K-Means algorithm, when we have updated the clusters.

VirginiaTech

Centered Voronoi Diagrams: Center the Generators

As I mentioned in the previous talk, we can take a given set of centers with a "disorderly" Voronoi diagram, and repeatedly replace the centers by centroids.

This will gradually produce a more orderly pattern in which the clusters are roughly the same size and shape, and the centers are truly at the center.

The key to this approach is to be able to compute or estimate the centroid of a set of points that are defined implicitly by the nearness relation.

VirginiaTech

Centered Voronoi Diagrams: Algorithm

This algorithm needs the centroid $(\bar{x}_i, \bar{y}_i)$ of each cluster C_i.

If C_i is a polygon, we use geometry to find the centroid.

If C_i is more complicated, we must use calculus:

$$\bar{x}_i = \frac{\int_{C_i} x \, dx \, dy}{\int_{C_i} dx \, dy}, \quad \bar{y}_i = \frac{\int_{C_i} y \, dx \, dy}{\int_{C_i} dx \, dy},$$

But exact calculations are limited to simple, small, regular problems!

VirginiaTech

Centered Voronoi Diagrams: Algorithm

We can estimate the centroid using sampling.

Generate a large number of sample points p in the entire region.

The centroid of cluster C_i is **approximately** the average of the sample points p that belong to C_i!

$$\bar{x}_i = \frac{\sum_{p \in C_i} x_p}{\sum_{p \in C_i} 1}, \quad \bar{y}_i = \frac{\sum_{p \in C_i} y_p}{\sum_{p \in C_i} 1}$$

This allows us to work with complicated shapes, spheres, higher dimensions and so on!

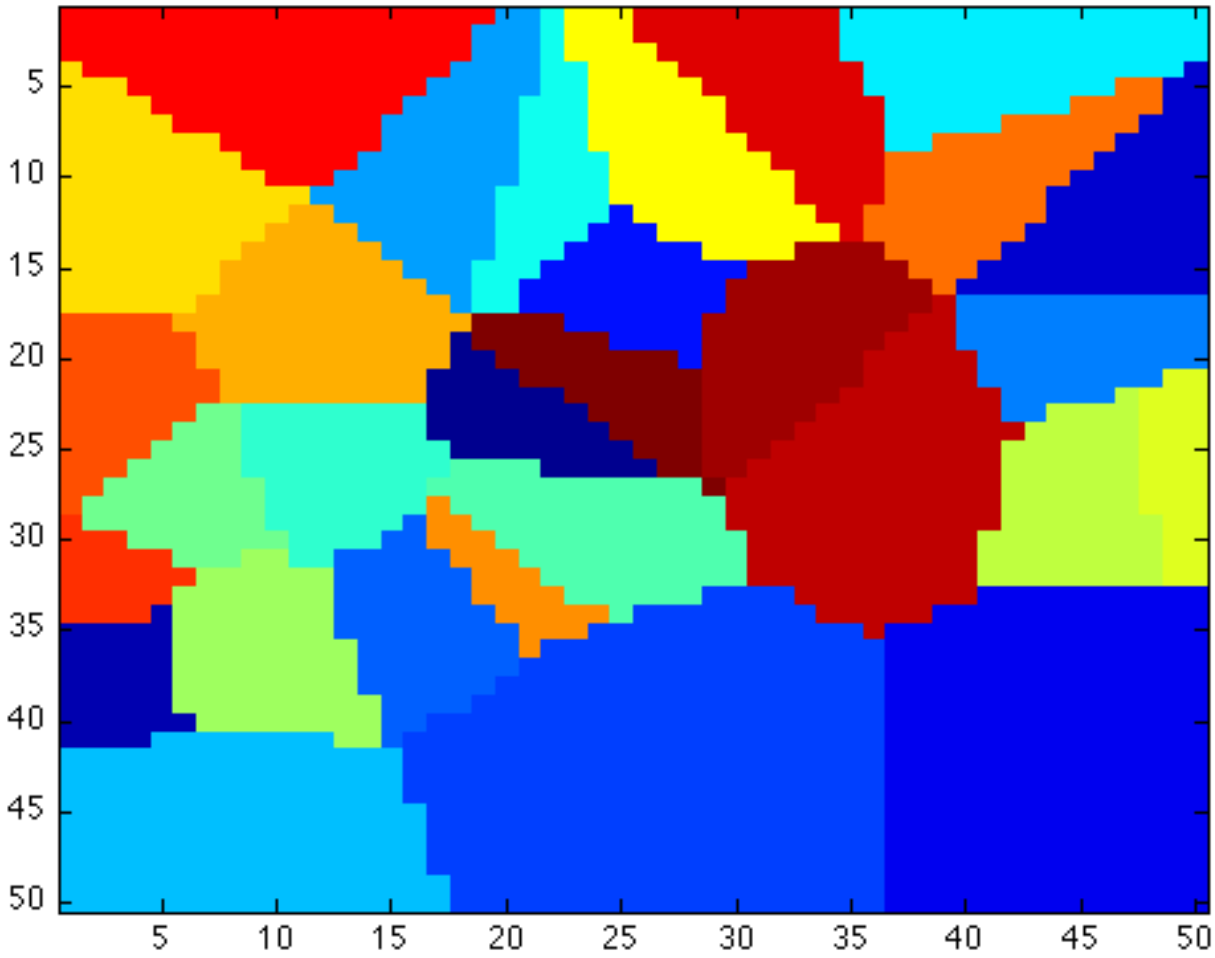
VirginiaTech

Centered Voronoi Diagrams: CVD.M

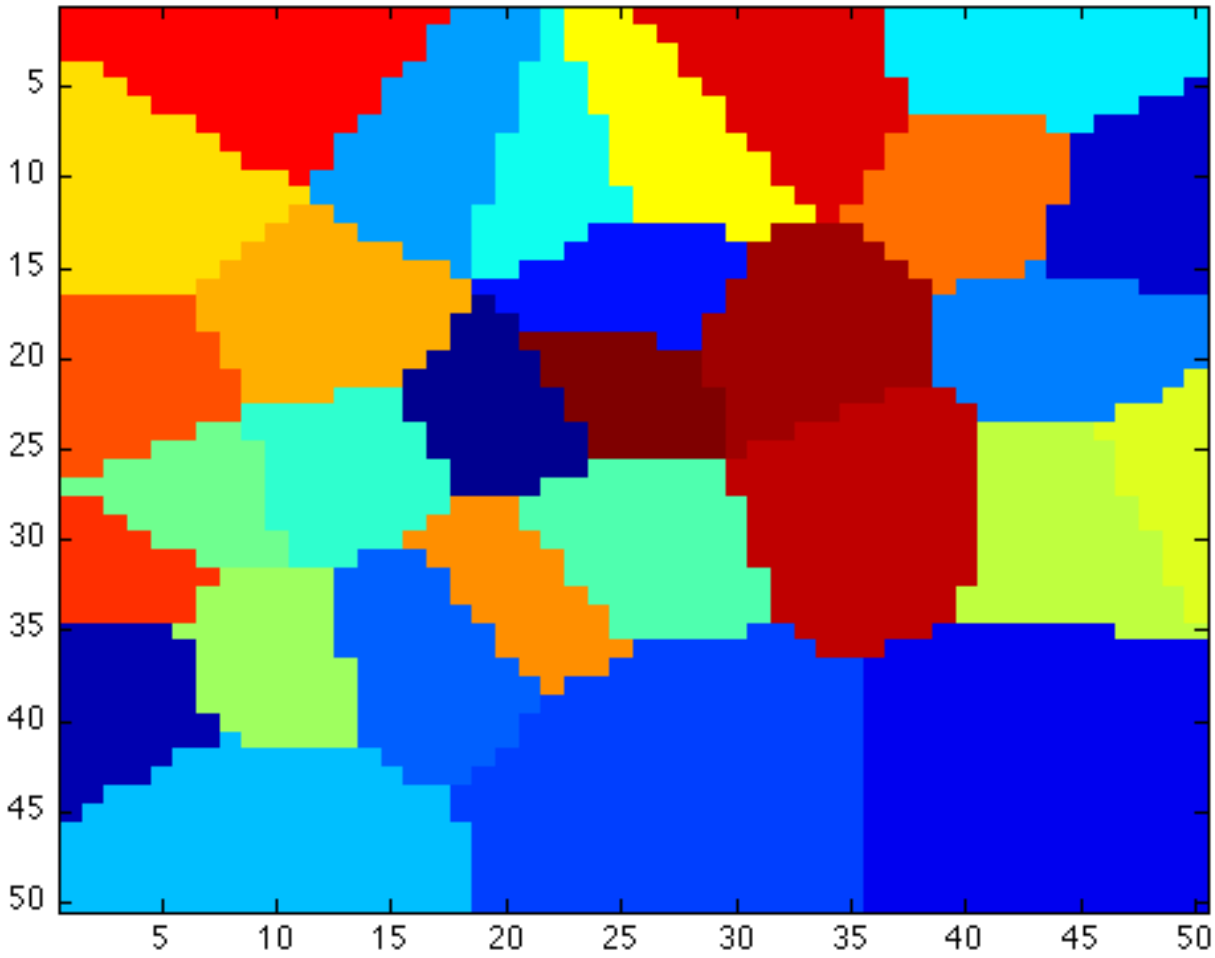
```
function cvd ( m, n, k )
  p = rand ( k, 2 );
  pc(1:k) = uint8 ( 1 : k );
  a = zeros ( m, n );
  for it = 1 : 20
    p_new = zeros(k,2);
    p_count = zeros(k);
    for i = 1 : m
      y = ( m - i ) / ( m - 1 );
      for j = 1 : n
        x = ( j - 1 ) / ( n - 1 );
        dmin = Inf;
        kkmin = 0;
        for kk = 1 : k
          d = norm ( p(kk,1:2) - [ x, y ] );
          if ( d < dmin )
            dmin = d;
            kkmin = kk;
          end
        end
        p_new(kkmin,1) = p_new(kkmin,1) + x;
        p_new(kkmin,2) = p_new(kkmin,2) + y;
        p_count(kkmin) = p_count(kkmin) + 1;
        a(i,j) = pc(kkmin);
      end
    end
    for kk = 1 : k
      p(kk,1:2) = p_new(kk,1:2) / p_count(kk);
    end
    imagesc ( a );
    pause
  end
```

VirginiaTech

Centered Voronoi Diagrams: Step 01

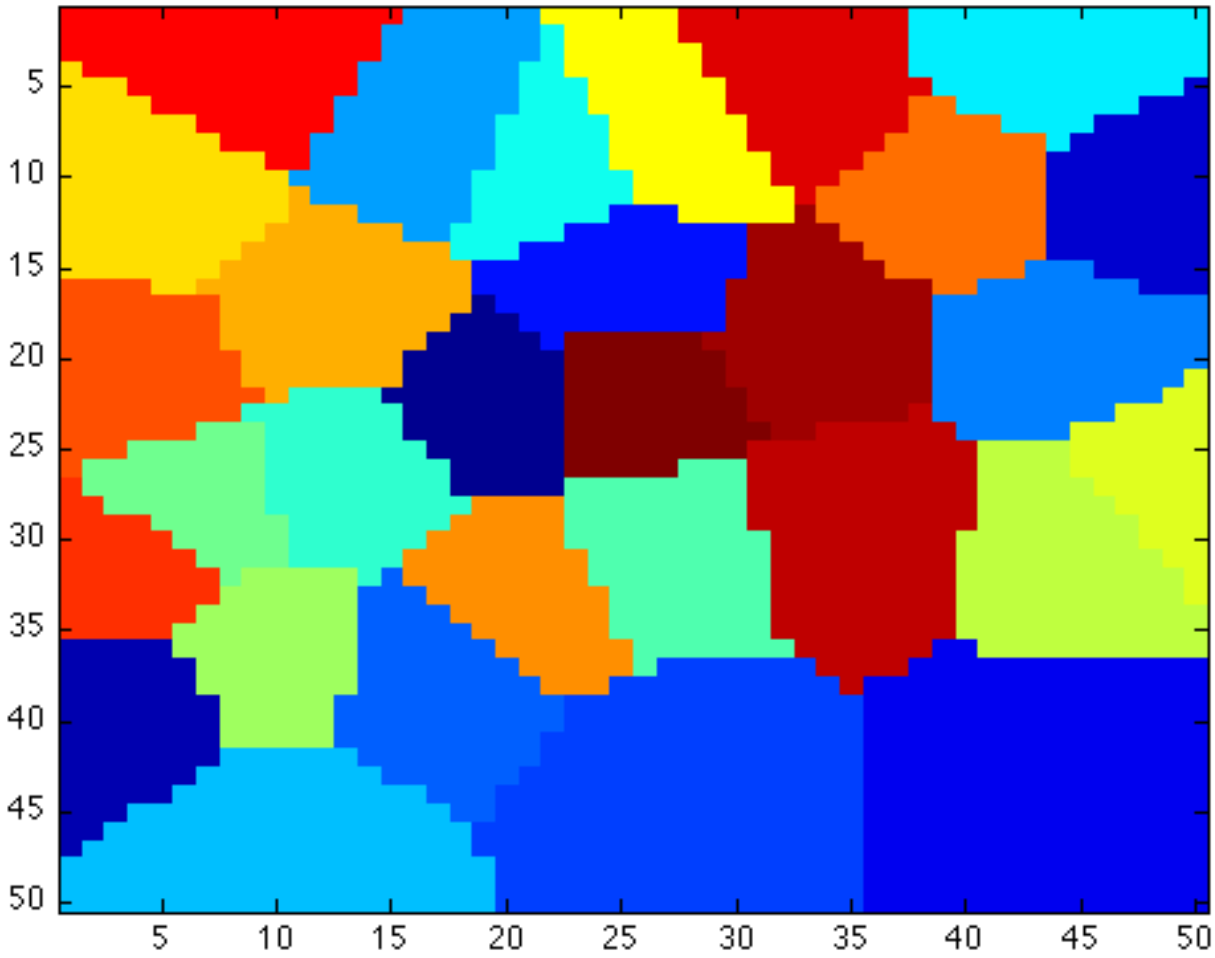


Centered Voronoi Diagrams: Step 02

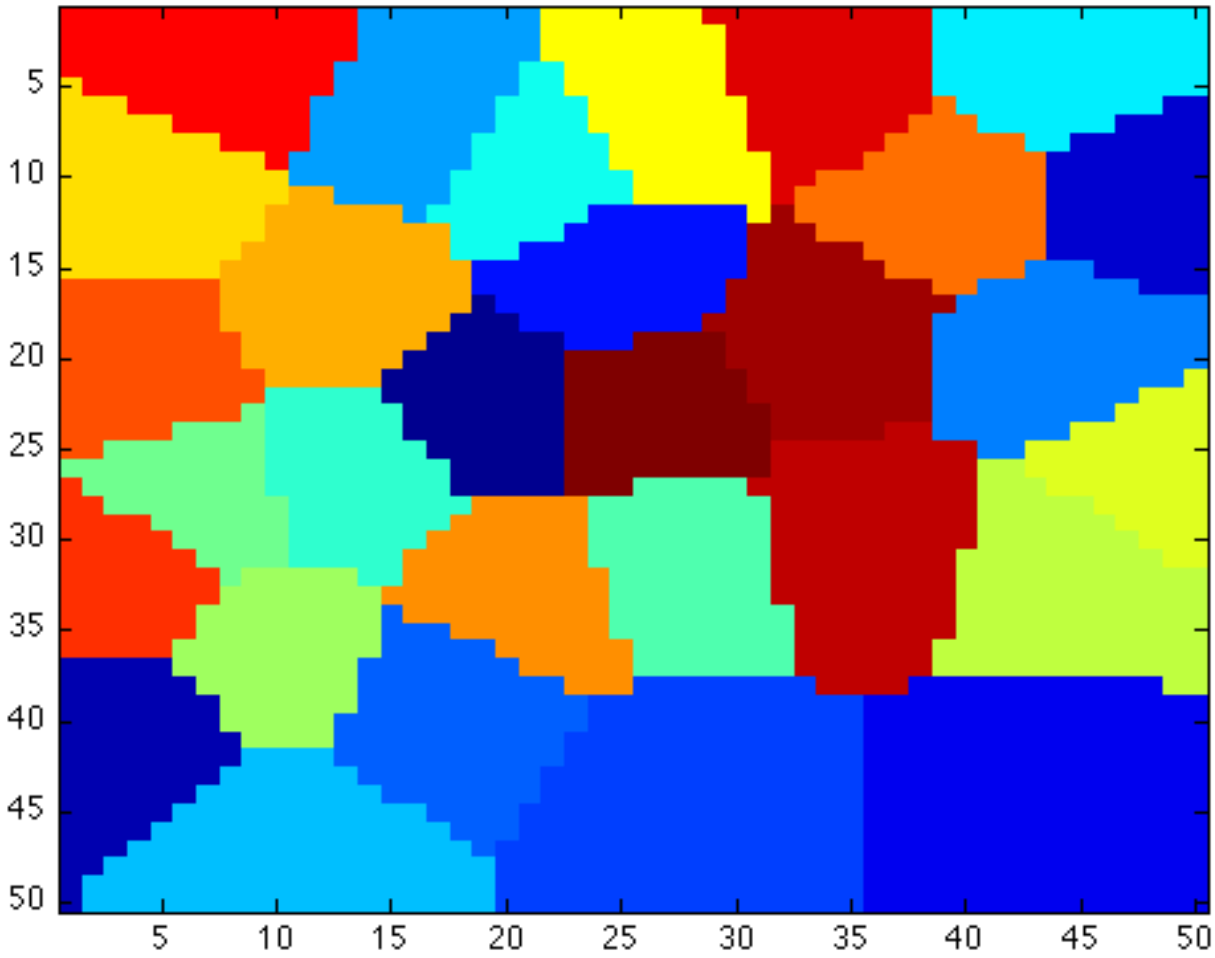


VirginiaTech

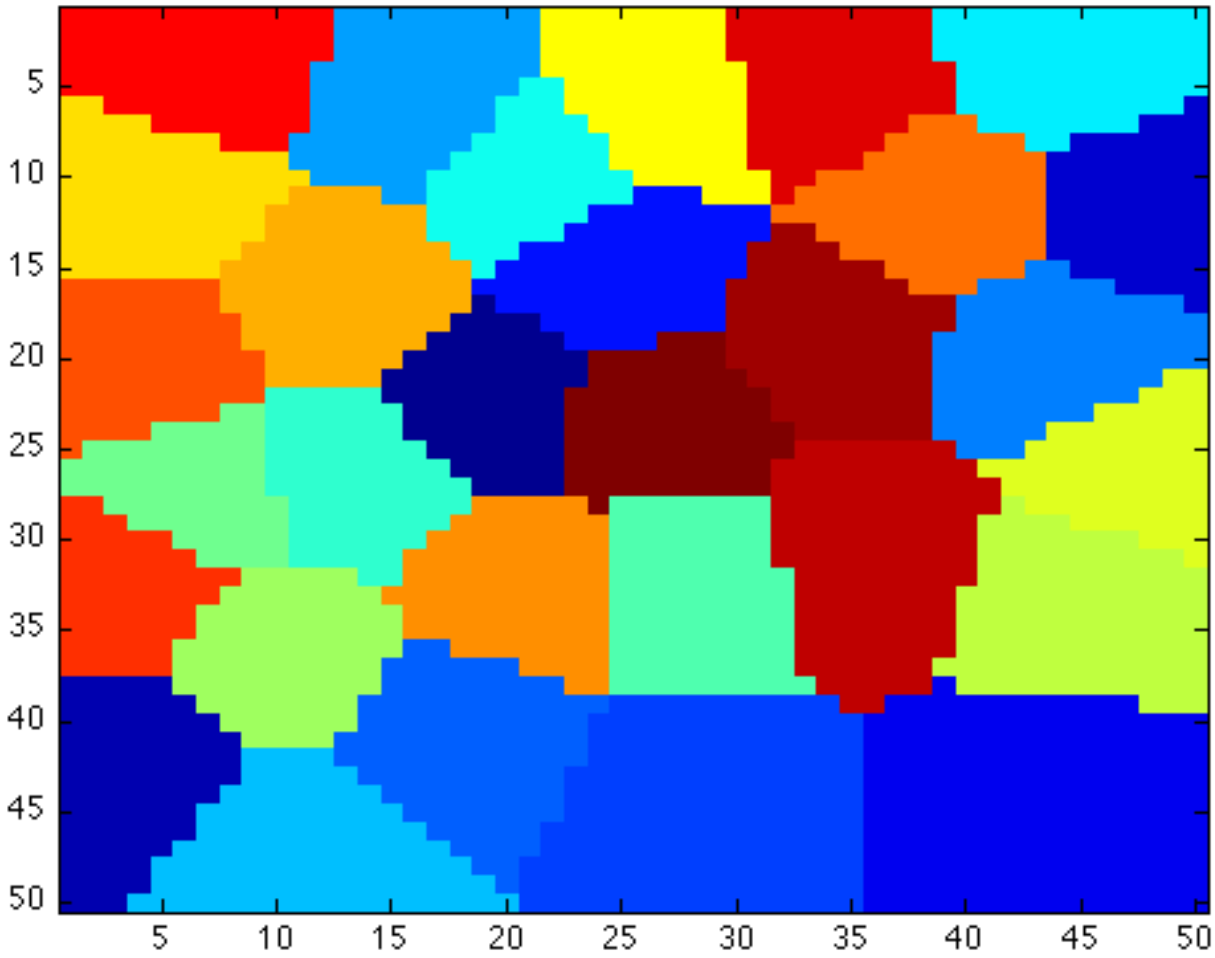
Centered Voronoi Diagrams: Step 03



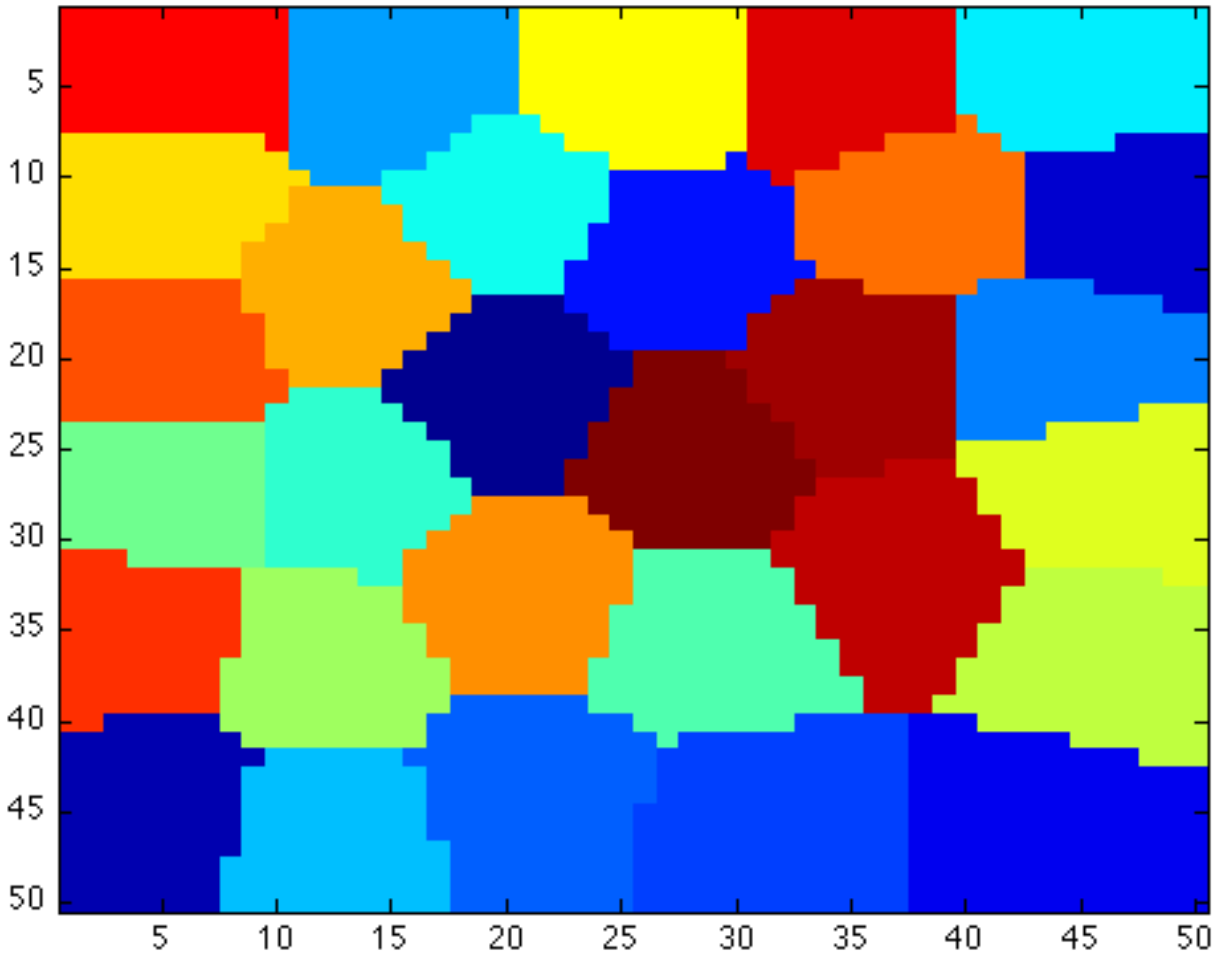
Centered Voronoi Diagrams: Step 04



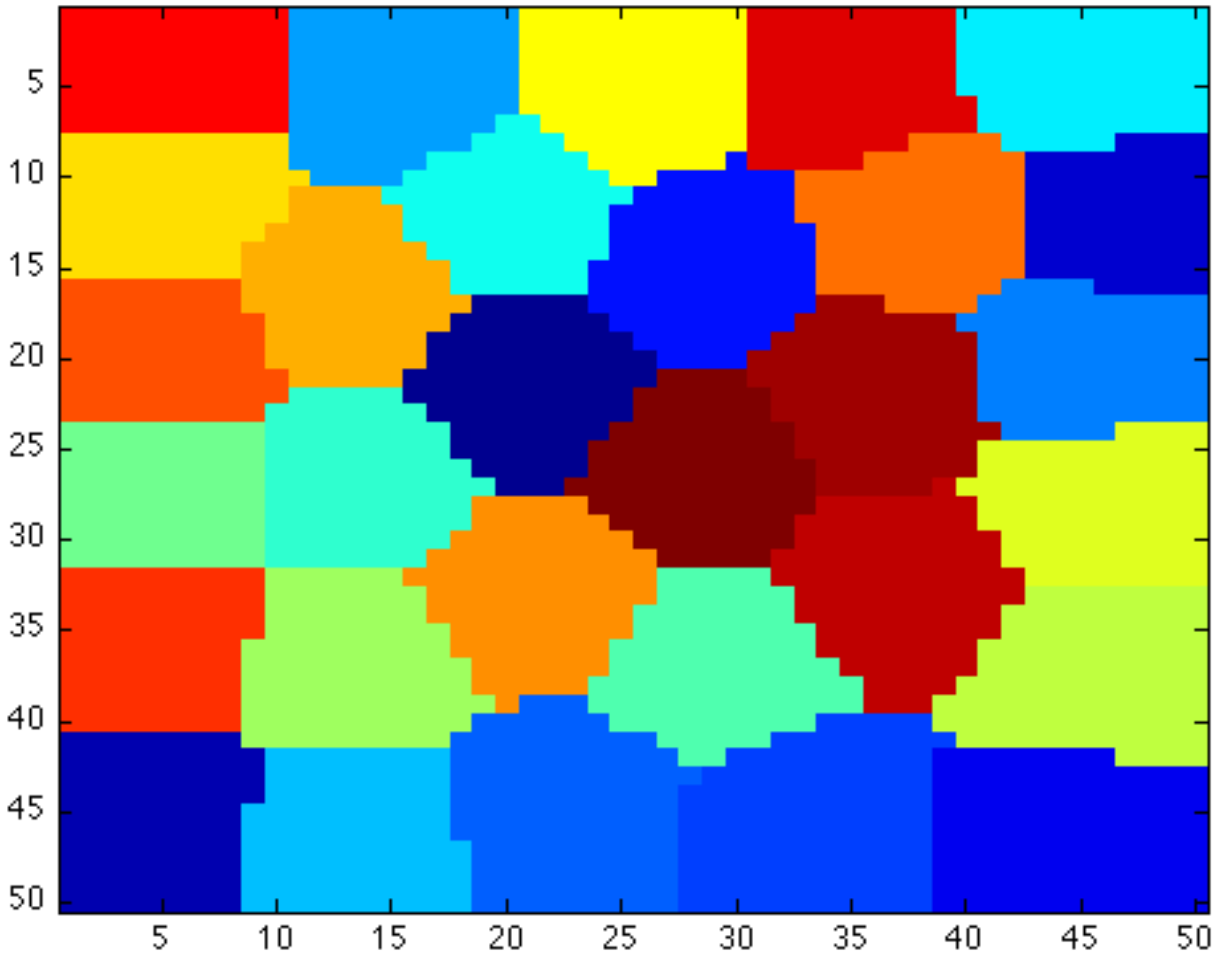
Centered Voronoi Diagrams: Step 05



Centered Voronoi Diagrams: Step 10

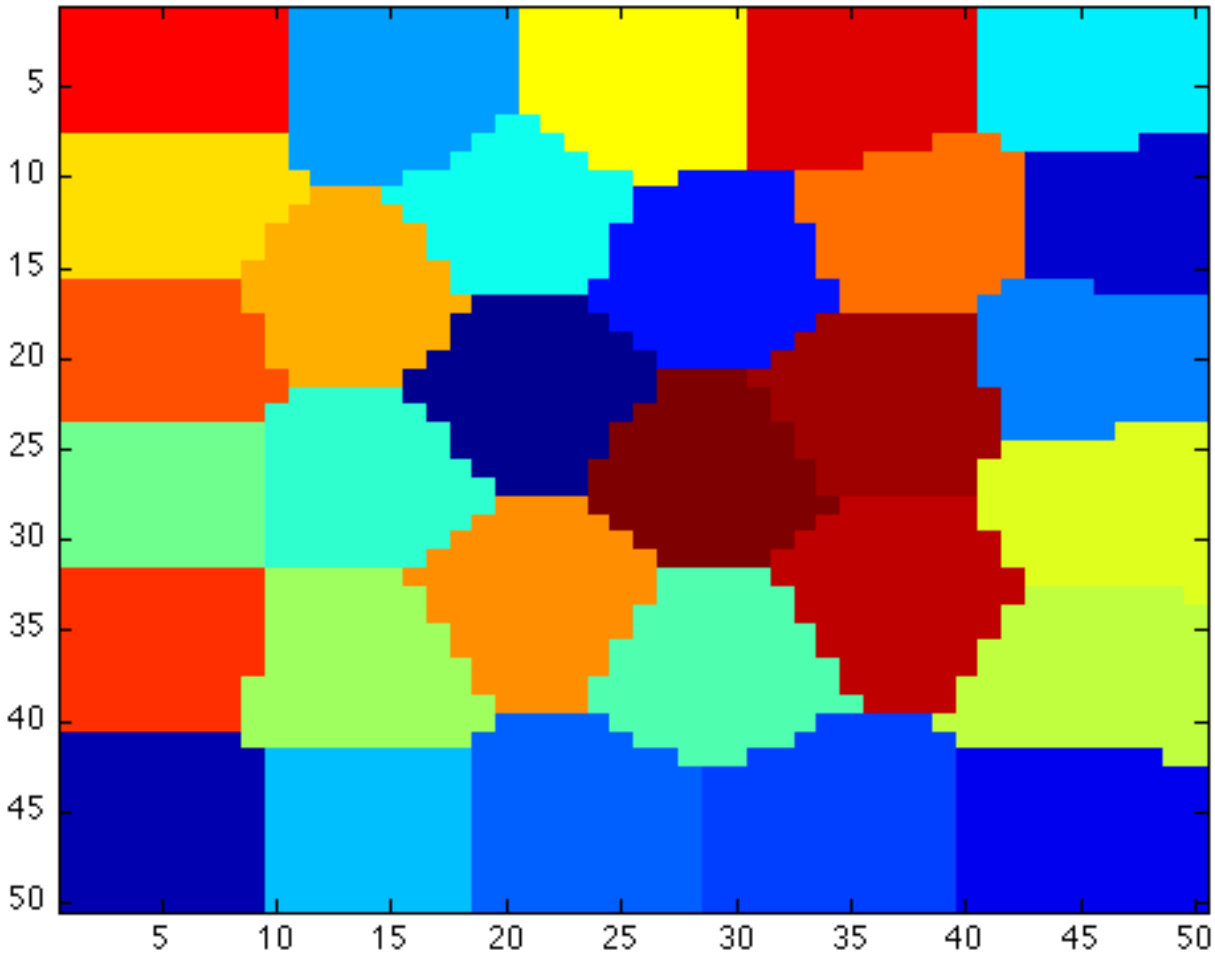


Centered Voronoi Diagrams: Step 15



VirginiaTech

Centered Voronoi Diagrams: Step 20



Centered Voronoi Diagrams: Density Functions

By using **density functions**, we can vary the size of the regions.

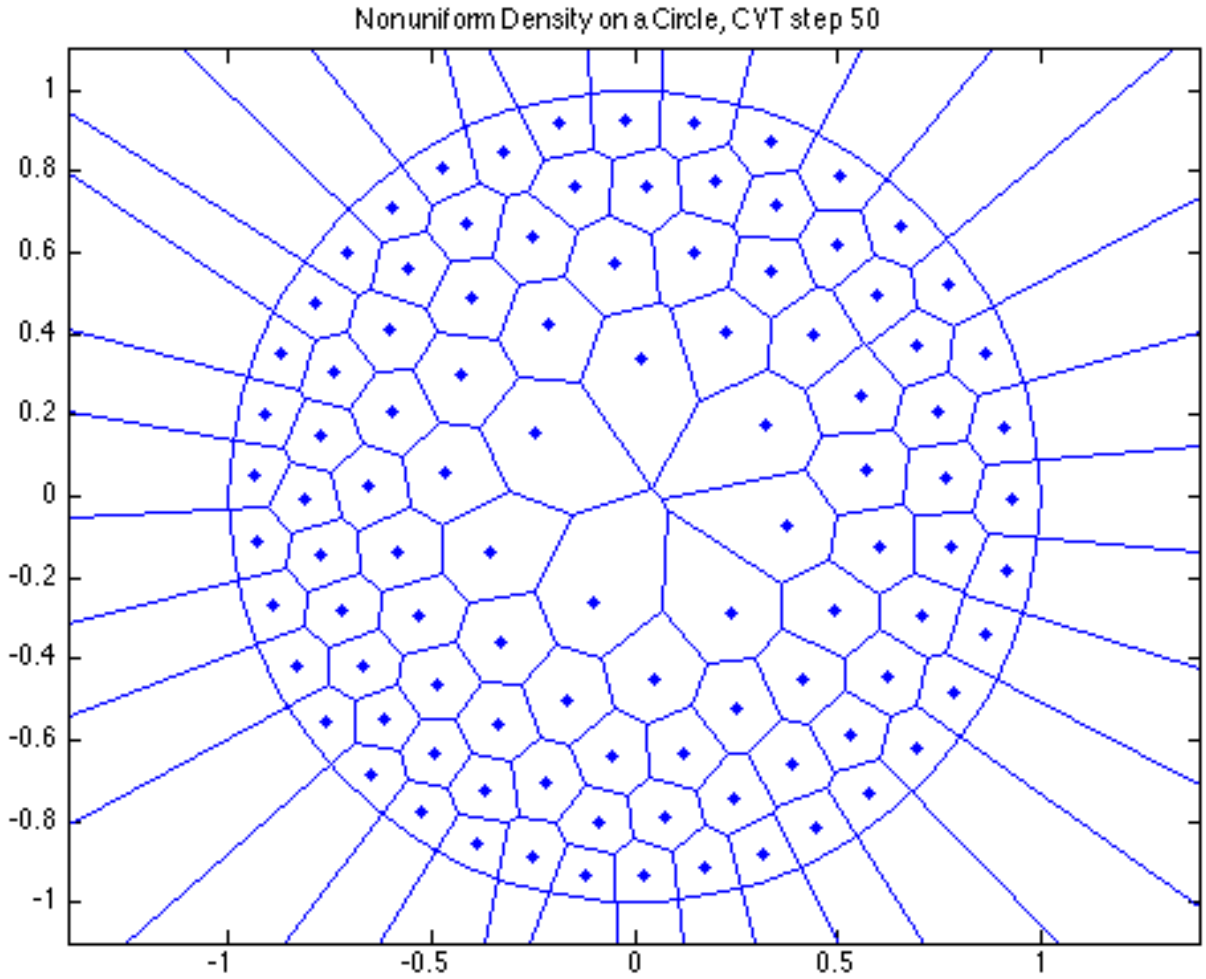
Two regions will not have the same area, but rather the same "weight", defined as the integral of the density.

This is like dividing a country into provinces of equal population.

Using this idea, for instance, you can generate a mesh for a computational region, and force the mesh to be fine near the boundaries, and very fine near corners or transition zones.

VirginiaTech

Centered Voronoi Diagrams: Nonuniform Density



VirginiaTech

THE DEATH MAP

1. Death in Golden Square
2. The Voronoi Diagram
3. Voronoi Computation
4. Centered Voronoi Diagrams
5. **CONCLUSION**

VirginiaTech

Conclusion: Voronoi Diagrams

The Voronoi diagram is a mathematical tool for analyzing situations in which geometric space is to broken up into subregions according to the placement of special center points.

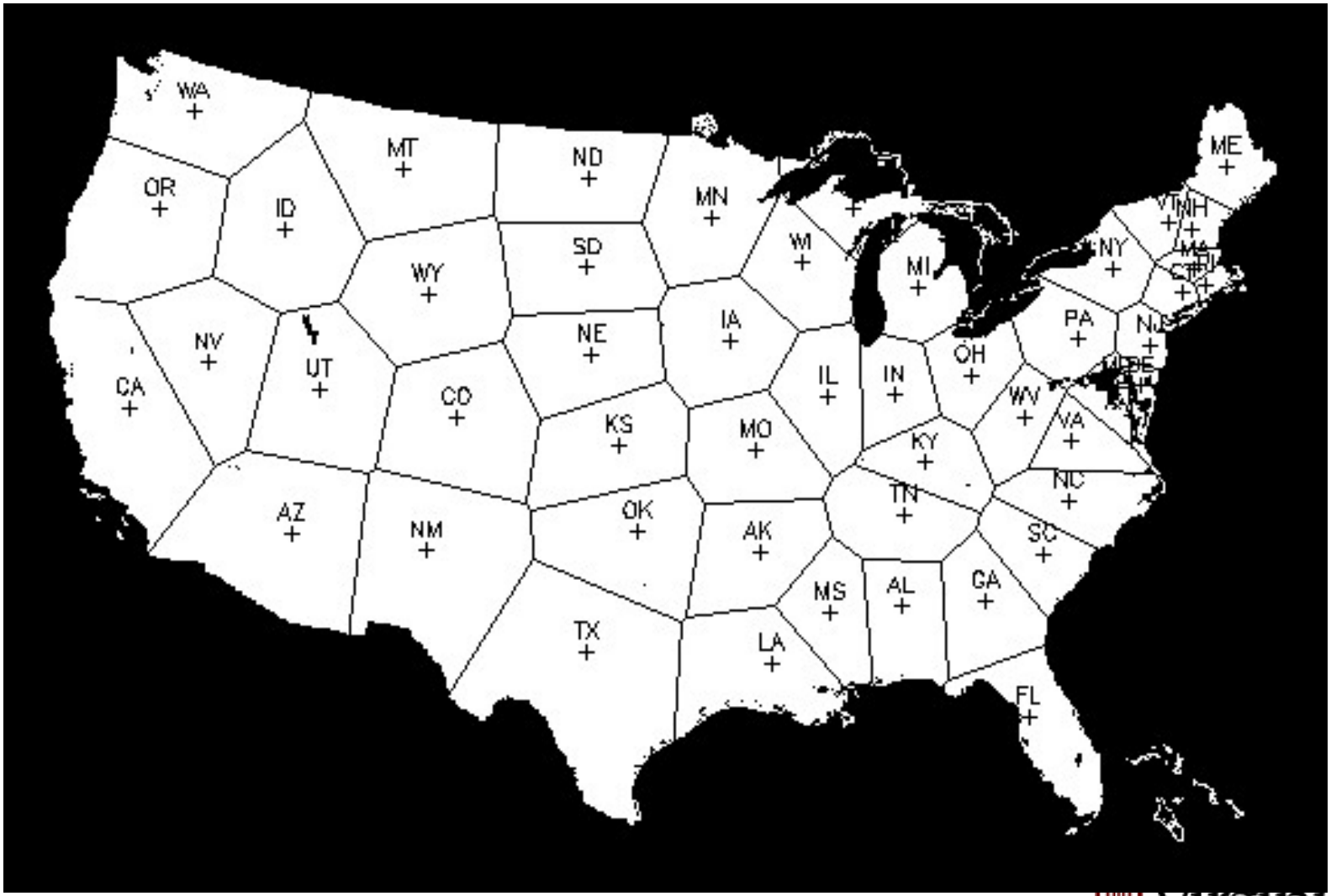
Recognizing that the pattern of disease in Golden Square corresponded to the clustering induced by the pumps was John Snow's key insight.

In biological systems, such as tissue made up of cells with nuclei, it is the case that the cells compete for space, and that the nuclei can move towards the center of the cell.

In such *adaptive* situations, where the centers are not permanently fixed in one position but can move, we'd expect that the clusters would have roughly the same shape and size, like a centered Voronoi Diagram.

VirginiaTech

Conclusion: The Voronoi States of America



Conclusion: Computational Geometry

The Voronoi diagram is one topic from the field of **computational geometry**.

- distances, areas, volumes, angles of complicated shapes;
- nearest neighbors;
- shortest paths, and best arrangements of points;
- convex hulls, triangulations, Voronoi diagrams;
- meshes, decomposition of shapes into triangles or tetrahedrons;
- motion of an object through a field of obstacles.
- visibility.

VirginiaTech

Conclusion: The John Snow Memorial

VirginiaTech