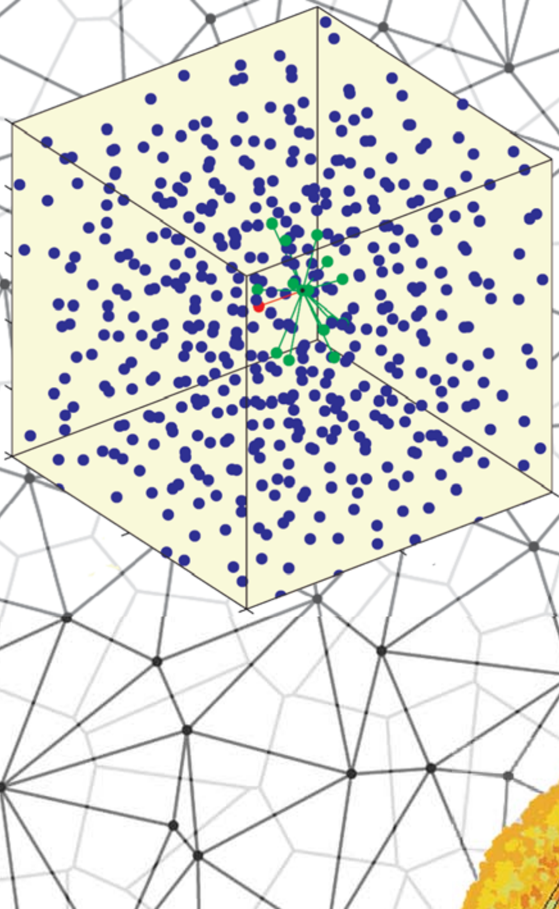
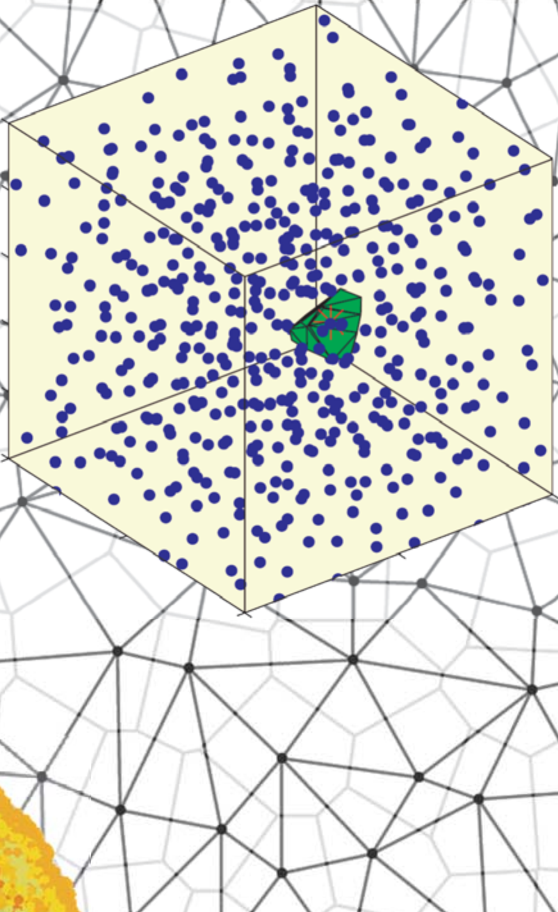
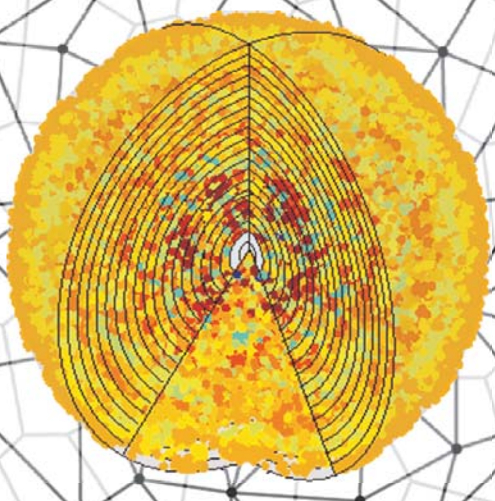
Spatial Analysis 3D

User's Guide







Spatial Analysis 3D:

Statistical and visualization program for three-dimensional spatial point patterns

Version 1.0

Available at:
http://www.nri.ucsb.edu/Labs/breese/SA3D.html

Funded by:
National Institute of Health (MH-069997)

Administered by:
Benjamin E. Reese
University of California Santa Barbara

Written by:
Dan Lofgreen
Channel Islands Consulting
Santa Maria, CA 93455

Phone: (805) 705-6865
FAX: (805) 562-2127

http://www.geocities.com/channelislandsconsulting

With special support from:
Stephen J. Eglen
Department for Applied Mathematics and Theoretical Physics
University of Cambridge, Cambridge UK

Special thanks to Adrian Baddeley for providing C code for estimation of F, G, and K functions in 3D.

Preface

ii What is Spatial Analysis 3D?
ii Why MATLAB?

What is Spatial Analysis 3D?

Spatial Analysis 3D is a user-friendly, graphical user interface (GUI) that allows statistical and visual manipulations of real and simulated three-dimensional spatial point patterns. Examples of the types of analyses performed include those derived from the Delaunay tessellation associated with such spatial point patterns, including nearest neighbor and Voronoi domain analysis, and those associated with the correlation of such point patterns, including autocorrelation analysis and its derived density recovery profile as well as the related K, F, and G-functions. The stimulus for the development of Spatial Analysis 3D has been the study of neuronal positioning within the central nervous system, but many other applications in science, engineering, statistics, and mathematics should benefit from this suite of programs.

Why MATLAB?

MATLAB® is a high-performance language for technical computing. The easy-to-use environment makes it an ideal platform for these types of simulations, computations, and visual presentation of 3D data.

One attractive feature of MATLAB is that the basic element is an array that does not require dimensioning. MATLAB also comes with an extensive set of plotting functions. Where an entire C program would be required to plot a 3-dimensional surface plot, this can be accomplished in one line in MATLAB.

MATLAB has the ability to call your own C or Fortran subroutines as if they were built-in functions. MATLAB callable C and Fortran programs are referred to as MEX-files. MEX-files are dynamically linked subroutines that the MATLAB interpreter can automatically load and execute. This means large pre-existing C and Fortran programs can be called from MATLAB without having to be rewritten as MATLAB M-files. Furthermore, bottleneck computations (usually for-loops) that do not run fast enough in MATLAB can be recoded in C or Fortran for efficiency.

MATLAB comes with a full set of online and printed documentation, including searches where the function of interest is unknown. Because of these features, MATLAB has become the tool of choice for high-productivity research and analysis.

Probably the most important feature of MATLAB is its availability. MATLAB is available for a variety of different platforms including Sun, SGI, Windows 95/NT/2000/XP, DEC Alpha, Mac, Linux, and UNIX.

It is important to note, however, that Spatial Analysis 3D was written for the Windows operating system. Unknown behavior may occur when using other operating systems, and the examples illustrated in Chapter 3 may look different.

Contents

Preface

What is Spatial Analysis 3D? ii
Why MATLAB? ii

Chapter 1 Background

Spatial Analysis 3D 1-2
Delaunay Tessellation and its Derivatives 1-2
    Voronoi Domain Direction Vectors 1-3
Spatial Correlation Analysis and its Derivatives 1-5
K, G and F-functions in 3D 1-8
    The K-function 1-8
    The G-function 1-9
    The F-function 1-9
    Implementation notes for the F-function 1-10
The Boundary Cell Problem 1-10
References 1-12

Chapter 2 Getting Started

Getting Started 2-2

Chapter 3 Running the Software

File Menu 3-2
    Opening and Closing Data Files 3-2
    Exporting Data 3-3
    Printing 3-4
Edit Menu 3-6
    Copying and Sending the Figure Window 3-6
    Editing the Colormap 3-7
Tools Menu 3-9
    Plot Manipulation 3-9
    View All Delaunay Tetrahedra 3-10
    Restore Marker Size 3-11
    View Menu 3-11
    New Random Set 3-13
    Options 3-14

Analysis Menu 3-21
Near Neighbor 3-21
Delaunay Tessellation 3-22
Voronoi Domains 3-24
Autocorrelation 3-25
F-Function 3-28
G-Function 3-29
K-Function 3-30
Plot Menu 3-31
False Color Maps 3-31
Nearest Neighbor Histogram 3-32
Delaunay Segment Length Histogram 3-33
Delaunay Facet Area Histogram 3-34
Delaunay Tetrahedra Volume Histogram 3-36
Voronoi Surface Area Histogram 3-37
Voronoi Volume Histogram 3-38
Voronoi Domain Direction Vectors 3-38
Autocorrelation Histogram 3-42
Autocorrelation Tri-Histogram 3-43
Autocorrelation Exclusion Zone 3-45
Help Menu 3-46
Tip of the Day 3-46
About Spatial Analysis 3-46

Appendix A **File Manifest**

Chapter 1 Background

1-2	**Spatial Analysis 3D**
1-2	**Delaunay Tessellation and its Derivatives**
1-5	**Spatial Correlation Analysis and its Derivatives**
1-8	**K, G and F-functions in 3D**
1-10	**The Boundary Cell Problem**
1-12	**References**

This chapter provides technical information on the simulation methods and algorithms used in the Spatial Analysis 3D program. The material in this chapter is not essential for understanding the use of the program.

Spatial Analysis 3D

Spatial Analysis 3D was born out of a collaborative research effort between Drs. Benjamin Reese, Mary Raven, and Dan Lofgreen at the University of California at Santa Barbara and Dr. Stephen Eglen at the University of Cambridge. It has been supported by a grant from the National Institute of Mental Health through the Neurotechnology Research, Development and Enhancement Program. It grew out of our efforts to quantify the regularity and simulate the patterning found in distributions of nerve cells across the retina, a structure in the central nervous system where uniformity in nerve cell spacing plays a critical role in retinal function.

Retinal nerve cells are distributed as non-random arrays across the retinal surface, and this regularity is fundamental to retinal organization, ensuring a uniform distribution of labor in processing the visual image across global variations in cellular density. Near neighbor analysis has been the classic means to assess the regularity in such retinal arrays, but Voronoi-based analyses have recently come into favor for describing the two-dimensional patterning in retinal mosaics. Autocorrelation analysis and modeling studies have established that the patterning in these mosaics is largely achieved by mechanisms acting during development that prohibit like-type cells from being positioned in close proximity to one another, being independent of the patterning present in other nerve cell types. It became apparent that these tools could be extended to the 3rd dimension in order to ask comparable questions elsewhere in the brain, and hence the birth of Spatial Analysis 3D: by inputting a list of x, y, z coordinates describing the spatial positioning of a population of neurons in three dimensions, one can test whether such cells are randomly distributed within the field, or whether there is any spatial patterning, regularity, minimal spacing, clustering or anisotropy in their distribution.

Delaunay Tessellation and its Derivatives

Given a set of data points in 2D, the Delaunay triangulation is defined as a set of lines (the Delaunay segments) connecting each point to its natural neighbors. A single point will have some variable number of natural neighbors, one of them being the shortest (or nearest) neighbor. In 3D, such a Delaunay tessellation is made up by a set of tetrahedra defined by the population of natural neighbors. Any given point will again have some variable number of neighbors, with each set of three neighbors and the point in question defining a single Delaunay tetrahedron. The population of nearest neighbor distances, like the population of all Delaunay segments or the population of Delaunay triangles in 2D (or tetrahedra in 3D) is a means to describe the spatial relationship between such elements in a population.

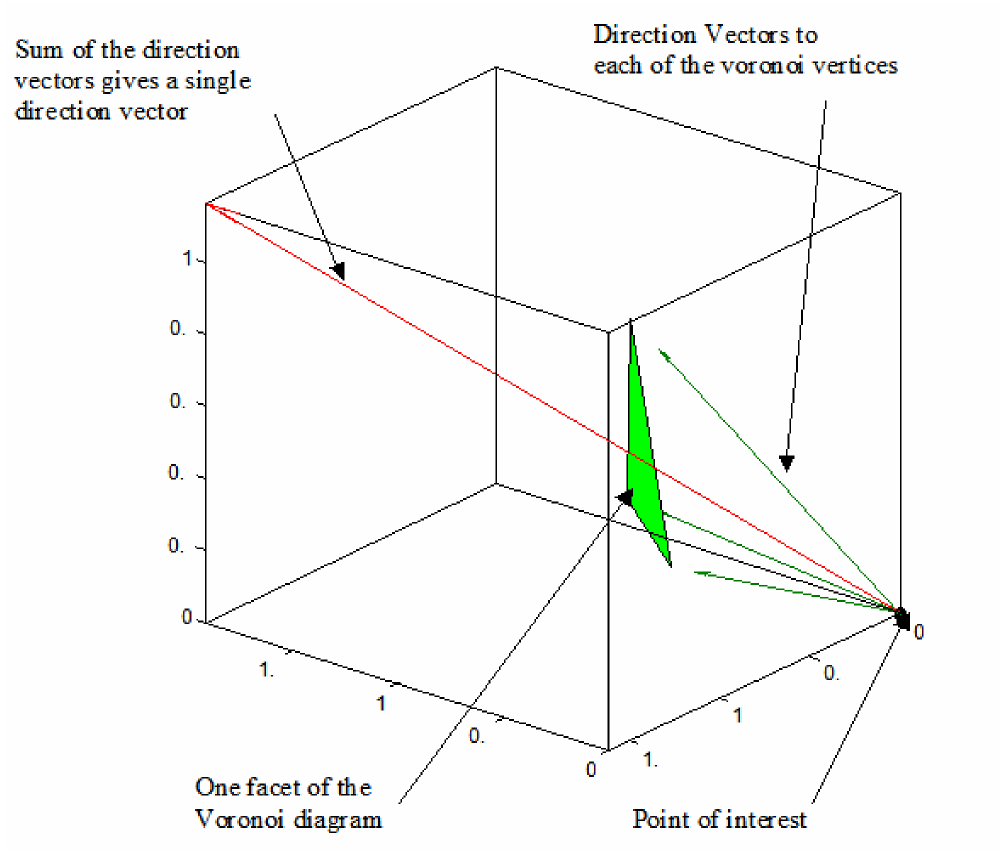
An alternative means of tessellating 2D or 3D space is by dividing it up into the Voronoi domains associated with those points. Voronoi tessellation describes the parcellation of space by proximity to each point – all points in the 2D or 3D field closer to a given point than to any of its neighbors defines the Voronoi domain of that point. In each case, the interconnecting points in the Delaunay tessellation define the Voronoi neighbors, and so the circle circumscribing each Delaunay triangle (or the sphere circumscribing each tetrahedron) defines a vertex of neighboring Voronoi polygons.

The algorithms used to generate the Delaunay tessellations are based on Qhull[1,2]. For more in-depth information regarding Qhull, visit http://www.qhull.org/.

Voronoi Domain Direction Vectors

In the previous section, we discussed the parcellation of space using a Voronoi tessellation. The result of the tessellation is a set of Voronoi domains, closed polyhedrons formed with triangular facets. In some circumstances, the Voronoi domains may display anisotropies that are consistent across the population. We developed a novel vectoring system that can be used to describe such bias in the shape of each Voronoi domain. We call these the Voronoi domain direction vectors.

To generate a direction vector, we consider the contribution of each facet to the whole Voronoi domain. We generate a vector for each facet by first drawing a vector to each vertex of the facet from the cell. The three vectors are then summed together to give a single direction vector for that particular facet. The vector for that facet is then multiplied by the facet area, effectively weighing each facet's direction vector. Direction vectors for all of the facets can then be summed to give a single direction vector for the Voronoi domain itself. The Voronoi direction vector is then normalized if its magnitude is greater than 1. The diagram on the following page outlines how a direction vector for one facet is calculated.


Sum of the direction vectors gives a single direction vector
Direction Vectors to each of the voronoi vertices
One facet of the Voronoi diagram
Point of interest
1
0.
0.
0.
0.
0
1.
1
0.
0
1.
1
0.
0

Consider a Voronoi domain with N facets. For each of those facets, there are three points that define it. The Voronoi direction vector can then be defined as

$$\vec{V} = \sum_{n=1}^{N} A_n \sum_{i=1}^{3} \overrightarrow{v_i^n}$$

where A_n is the area of facet n and $\overrightarrow{v_i^n}$ is the vector to each facet point from the central point.

There are situations with regularly spaced data (though rarely achieved in biological specimens) where the Voronoi domains will be symmetrical. In these cases, the direction vector for one facet may exactly cancel the direction vector of the opposing facet (e.g. <-1, -1, -1> and <1, 1, 1>). For the case where the Voronoi domain is a rectangular prism, the resulting direction vector for the domain will be <0, 0, 0>. To indicate that there is an elongated direction in these types of cases, a second algorithm is applied. When the direction vector is weighted by the facet area, the absolute value of the vector is used. This insures that no vector can cancel another. It also provides an order of magnitude for each independent direction. The resulting vector is then normalized by its largest direction and then the smallest direction is subtracted off. An example of this calculation is given on the following page.

$$\vec{V} = \sum_{n=1}^{N} A_n \sum_{i=1}^{3} \overrightarrow{\left|v_i^n\right|}$$

Note that the magnitude of each vector component is used in this algorithm. The vector above is then normalized by the algorithm described above.

An example of the normalization algorithm is given below.

$$\begin{aligned}
&\vec{V_1} = \langle 4,1,2 \rangle \\
&\vec{V_2} = \vec{V_1} / \max(\vec{V_1}) = \vec{V_1} / 4 = \langle 1,0.25,0.5 \rangle \\
&\vec{V} = \vec{V_2} - \min(\vec{V_2}) = \vec{V_2} - 0.25 = \langle 0.75,0,0.25 \rangle
\end{aligned}$$

The algorithm used above gives a relative value for each direction. In this example, the y-direction has the least influence and the x-direction has the most influence.

Spatial Correlation Analysis and its Derivatives

The spatial plot of the position of each point in a field relative to every other point is the autocorrelation and is another means for describing the patterning associated with a set of points. For a population of points that approximates a lattice, the periodicity in the population will reinforce itself and become more readily apparent in the autocorrelogram. The autocorrelogram also readily discriminates distributions of points that display a tendency for clustering, or for anti-clustering (avoiding being positioned close to one another). One means of displaying the data derived from an autocorrelogram is to plot the density recovery profile (DRP), a histogram of the average density of cells in the correlogram as a function of eccentricity from the origin. For more detailed information regarding the DRP, see Rodieck[3]. For a random distribution of points, the DRP shows an invariant density as a function of distance from the origin. Clustered populations show elevated densities near the origin, while anti-clustered populations show a reduction in density at the origin, climbing to some average (recovered) density as a function of eccentricity. The size of the region within which the probability of finding another point is lower than the average density can be estimated as the "effective radius", defined originally by Rodieck in 2D as the radius of a cylinder of identical volume to that defined by the "empty" region at the origin. The size of the bin-width in the DRP is constrained by both sample size and density, leading some to prefer to compute the K-function instead. Below we derive the statistics used in the 3D autocorrelation analysis.

V	volume of the region containing the reference points
N	number of points in the region
D	density of the points in the region equal to N/V

i	index into each spherical shell 1,2,… (or the corresponding bin in the density recovery profile)
ΔV_i	volume of the spherical shell i
Δr	width of each shell (and bin width of the density recovery profile)
n_i	measured number of points in spherical shell i
λ_i	expected number of points in spherical shell, assuming a random distribution
d_i	density measure for the density recovery profile

The volume of a spherical shell can be shown to be

$$\Delta V_i = 4/3\pi \Delta r^3 \left(3i^2 - 3i + 1\right)$$

The expected number of points in a spherical shell is the product of the expected number to be found in the shell for each reference point times the number of reference points in the sample region N. If we have a random distribution, then the expected number in the shell for each reference point is the density of points D times the volume of the shell ΔV_i.

$$\lambda_i = ND\Delta V_i$$

Let

$$d_i = \frac{n_i}{N\Delta V_i}$$

To calculate effective radius of the dead space we first need to look at the number of points that are "missing" near the origin. We do this by defining a flux for each bin which is the spherical shell volume for each bin times its height relative to the expected density in the sample space:

$$\Delta N_i = \Delta V_i \left(D - d_i\right)$$

Substitution using the equations above

$$\Delta N_i = \frac{\lambda_i - n_i}{N}$$

so that

$$N_e = \frac{1}{N} \sum_{i=1}^{until \cdot n_i > \lambda_i} \left(\lambda_i - n_i\right)$$

Now N_e represents the total number of points in the dead space. If we choose to think of this dead space in terms of a linear measurement, we can compare it to a spherical volume with an effective radius, r_{eff}. By definition we have

$$N_e = 4/3 \cdot \pi \cdot r_{eff}^3 D$$

so that

$$r_{eff} = \left(\frac{3N_e}{4\pi D}\right)^{1/3}$$

If we have too few points, or the bins are too thin and thus capture too few points, the measure of effective radius can be unreliable. For a random (Poisson) distribution, the variance in the number of points in a given area is equal to the mean number λ.

$$\sigma^2 = \lambda$$

with substitution

$$\sigma^2 = ND\Delta V$$
$$\sigma = \sqrt{ND\Delta V}$$

This is the standard deviation about the mean number of points in a bin. To plot this in the density recovery profile, multiply by the scaling factor for each bin:

$$\sigma_i' = \frac{\sqrt{ND\Delta V_i}}{N\Delta V_i} = \frac{1}{\sqrt{V\Delta V_i}}$$

Substituting for ΔV_i

$$\sigma_i = \frac{1}{\sqrt{V \cdot 4/3\pi\Delta r^3\left(3i^2 - 3i + 1\right)}}$$

Now if we let

$$D_c = \frac{1}{\sqrt{V \cdot 4/3\pi\Delta r^3}}$$

then

$$\sigma_i' = \frac{D_c}{\sqrt{3i^2 - 3i + 1}}$$

The value D_c depends only on the sample volume V and the bin width Δr. It has units of spatial density and it is called the critical density. As long as D is large compared to D_c then the estimates for each bin will be good, and the estimate for the effective radius will be reliable. One measure of the robustness of this measure is the ratio of the densities. This ratio is called the reliability factor.

$$k = \frac{D}{D_c}$$

The maximum value of the effective radius is constrained by the observable sample density. It is therefore limited by how tightly the points within the effective radius can be packed. Maximum packing in three dimensions is achieved by hexagonal close packing, where the volume *v* per point would be

$$v = \frac{r_m^3}{\sqrt{2}}$$

Since

$$D = \frac{1}{v}$$

we have

$$r_m^3 = \frac{\sqrt{2}}{D}$$

The packing factor is then defined as the cube of the ratio of the effective radius to the maximum radius, or alternatively, the ratio of the effective volume to the maximum volume

$$p = \frac{V_{eff}}{V_m} = \frac{4/3\pi \cdot r_{eff}^3}{4/3\pi \cdot r_m^3}\frac{D}{D} = \left(\frac{r_{eff}}{r_m}\right)^3$$

Since the effective radius can range from 0 to r_m, the packing factor can range from 0 to 1.

K, G and F-functions in 3D

The spatial statistics community makes wide use of the K, G and F-functions. This section defines those functions for 3D and mentions brief implementation details. The implementation of these routines is provided by Baddeley[4] and is available via the Spatial Analysis 3D software.

The K-function

The K-function is the cumulative version of the density recovery profile (DRP). The advantage of the K-function is that it is not necessary to specify a bin width, and so we are not troubled by setting large bin widths when the number of cell bodies is low. The K-function is defined over a range of distances *t* with the univariate K-function defined in three dimensions as:

$$K(t) = \frac{|B|}{n^2}\sum_{i=1}^{n}\sum_{j \neq i} w(i,j)^{-1}\, I\left(\left\|x_i - x_j\right\| \leq t\right)$$

I(·) is the indicator function; it counts the number of cell pairs that are less than or equal to some distance *t* apart from each other. The term *w(i, j)* is the weighting factor to adjust for border corrections; it measures the fraction of the circumference of the sphere centered at x_i and with radius $||x_i - x_j||$ that is within the sampling window *A*.

Under the null hypothesis of complete spatial randomness (CSR), the theoretical K-function in 3D is $K(t) = 4/3\pi t^3$. The K-function is measured over a range of distances, typically from 0 to *s/4*, where *s = min(h,w,d)* (often *s* can be smaller).

Table 1.1 Notation used in the K,G, and F-functions

term	definition
n	Number of cells in the sample space
x_i	Location of i^{th} individual data point, given as a 3D vector (x,y,z)
B	3D sample volume ("brick" or "bounding box"). All data points are assumed to be sampled within this volume of width *w*, height *h*, and depth *d*
$\|B\|$	Volume of the bounding box $(w \times h \times d)$
K(t)	K-function: average number of cells within distance *t*
F(t)	F-function: empty space function
G(t)	G-function: cumulative nearest neighbor function

The G-function

The *G(t)* function is also a cumulative plot over a range of distances, measuring the fraction of nearest-neighboring distances that are less than or equal to *t*:

$$G(t) = \frac{1}{n}\sum_{i=1}^{n} I(y_i \le t)$$

where y_i is the distance of cell *i* to its nearest-neighbor of the same type. Since *G* is a cumulative probability, the range of *G* is [0,1]. Hence, *G(t)* is normally evaluated over a range from 0 to some value *h* such that *G(h) = 1*.

Under the null hypothesis of complete spatial randomness (CSR), the theoretical *G* function in 3D is $G(t) = 1 - exp(-4/3\lambda\pi t^3)$.

The F-function

The F-function is similar to the G-function, except that rather than measuring the distance from each cell to its nearest-neighboring cell, we measure the distance from each grid point to its nearest neighboring cell. We place (typically) *g* grid points at regular intervals throughout the sample field *A* (see below for implemenation details). This allows us to see whether any parts of the field are devoid of cells (which will not show up in the G-function.) Hence, we define:

$$F(t) = \frac{1}{g}\sum_{i=1}^{g} I(y_i \le t)$$

where y_i is the distance of grid point *i* to the nearest cell.

Under the null hypothesis of complete spatial randomness (CSR), the theoretical F-function in 3D is $F(t) = 1 - exp(-4/3\lambda\pi t^3)$.

Implementation notes for the F-function

The Baddeley implementation of the F-function works by dividing the bounding box into small voxels; each voxel is a cube of side length *v* (this is the VSIDE parameter). If the bounding box is of size $w \times h \times d$, the number of voxels made is roughly $(w/v)\times(h/v)\times(d/v)$ (the actual number is $(w/v + 1)\times(h/v + 1)\times(d/v + 1)$, note the extra 1 in each dimension). For each voxel, the distance to the nearest cell is computed. These distances are then put into a cumulative histogram and normalized by the number of voxels to produce the cumulative probability. Hence, the smaller *v* is, the more voxels are created, and thus the more accurately *F* is estimated. (Conversely as *v* increases, there are fewer voxels and so *F* becomes more discrete.)

However, the more voxels, the more computation required. For example, in a bounding box of size 1000×1000×1000μm^3, if v = 100μm, then $((1000/100) + 1)^3 = 1331$ voxels are created. If, however, we increase *v* by a factor of 2, the number of voxels decreases to 216. It is very important, therefore, to choose the number of voxels carefully. In the Spatial Analysis 3D software, the number of voxels in the largest dimension can be chosen inside the options menu (see Section 2 Running the Software: Options). This value is also known as the number of F-divisions. Typically the number of F-divisions should be between 10 and 100; anything larger will significantly increase the computation time. Thus, a sample space with dimensions 1000μm × 1000μm × 1000μm and a value of 100 for the number of F-divisions will yield 1,030,301 total voxels!

The Boundary Cell Problem

All of the above analyses are plagued by the fact that cells near the boundaries of a field will have incomplete Voronoi domains, uncertain numbers of Delaunay neighbors, and they may have nearest neighbors less than those detectable within the field. Furthermore, the autocorrelation analysis will be biased by the inclusion of these cells. There are a few ways to circumvent the problem of such boundary, or in the present parlance, “infected”, cells (“infected” because their Voronoi, Delaunay, and nearest neighbor data are suspect).

One way is to identify a secondary boundary within the sampled field such that all cells within the secondary boundary have their complete Delaunay and Voronoi tessellations revealed by the presence of neighbors outside of this boundary (but within the total sample). The autocorrelation analysis can proceed similarly, sampling cells within the secondary boundary such that the entire complement of cells surrounding every cell within the secondary boundary are included within the correlogram generated, even if those between the border of the sample and this secondary boundary are never positioned at the origin of the correlogram. These approaches are effective when the density relative to the total size of the sampled field is large; they are more problematic when fields are small and/or densities are low, conditions that are typically exacerbated when working with the third, Z, dimension. The alternative is to attempt a compromise solution, eliminating only those data associated with such infected cells. This is indeed a compromise, since while it may yield accurate estimates of nearest neighbor data for a population of cells, the uncertainty of the tessellation for the infected cells may in some cases alter the actual tessellation of the next tier of cells beyond the infected ones, thereby producing some inaccuracy in the Voronoi and Delaunay analyses. In practice, the difference in the Voronoi domains of such a second tier of boundary cells produced by this uncertainty is not likely to be great, and may be reasonably assumed to affect two population datasets similarly (experimental and control; real and random; etc) if they are matched in density. The present program offers a number of options allowing the user to choose how to deal with these issues in various analyses, recognizing the trade-off between increasing the sample size at the cost of some inaccuracy associated with the number (proportion) of infected cells. One is simply to exclude the data associated with such infected cells. Another option, when conducting correlation-based analyses and the related K-function, is to mathematically correct for the asymmetric contributions made by cells near the boundary, as originally described by Rodieck and by Ripley[5]. Further details are provided in each relevant section.

References

[1] National Science and Technology Research Center for Computation and Visualization of Geometric Structures (The Geometry Center), University of Minnesota, 1993.

[2] C.B. Barber, D.P. Dobkin, and H.T. Huhdanpaa, "The Quickhull Algorithm for Convex Hulls", *ACM Transactions on Mathematical Software*, vol. 22, pp. 469-483, 1996.

[3] R.W. Rodieck, ”The density recovery profile: A method for the analysis of points in the plane applicable to retinal studies”, *Visual Neuroscience*, vol. 6, pp. 95-111, 1991.

[4] A.J. Baddeley, R.A. Moyeed, C.V. Howard, and A. Boyde, “Analysis of a three-dimensional point pattern with replication”, *Applied Statistics*, vol. 42, pp. 641-668, 1993.

[5] B.D. Ripley, “The second-order analysis of stationary point processes”, *Journal of Applied Probability*., vol. 13, pp. 255-266.

Chapter 2 Getting Started

2-2 Getting Started

Getting Started

To start the Spatial Analysis 3D program, MATLAB must be running. If you do not have a full version of MATLAB, you can get information about purchasing it from The MathWorks:

(508)-647-7000	Phone
http://www.mathworks.com	Web
info@mathworks.com	Sales, pricing, and general information

Spatial Analysis 3D was written using MATLAB 7.1 (Version 7.0 with Service Pack 3). The program will not run correctly unless you have MATLAB 7.0 or higher installed. It is highly recommended that the service pack be installed with version 7.0 as this service pack fixes many critical bugs in MATLAB. The source code for the Spatial Analysis 3D program is available at the main website:

http://www.nri.ucsb.edu/Labs/breese/SA3D.html

On a PC or Mac, simply double click on the MATLAB icon. To run MATLAB on a UNIX system, type `matlab` at the operating system prompt, or type `matlab &` to run the program in the background.

Once MATLAB is running you need to "install" the program. When you run the installation program, MATLAB will copy the files into the proper directories. Then, all the mex files will be compiled for your version of MATLAB and operating system. Finally, the path from within MATLAB will be modified so that the program can operate. Additionally, mex files for Windows XP running MATLAB 7.1 are supplied along with the program.

In the instructions that follow, >> denotes the matlab prompt, and $ denotes the unix shell prompt.

Windows Users

1. Download the main file `SpatialAnalysis.zip` and the install file, `InstallSpatialAnalysis.m.` Make sure the two downloaded files are in the same folder on your computer. Alternatively, the `InstallSpatialAnalysis.m` file will download the zip file directly from the main website.

2. Start MATLAB and change the current directory to the folder where the two downloaded files are located. You can change the current directory at the top of the main MATLAB window. At the matlab prompt, type:

```
>> installSpatialAnalysis
```

3. Once the Spatial Analysis 3D program has been installed, simply type in `SpatialAnalysis` at the matlab prompt to start the software in the future.

```
>> SpatialAnalysis
```

UNIX/Mac Users

1. Create a folder where you want to store the program. In this example, the install directory is `/tmp/ucsb3d` (normally however you will permanently install it within your home directory).

```
$ mkdir /tmp/ucsb3d
```

2. Download `SpatialAnalysis.zip`, put into this directory, and unzip it:

```
$ cd /tmp/ucsb3d
$ unzip SpatialAnalysis.zip
```

3. This will create two directories, `SpatialAnalysis3d` and `userfun`. Both these directories will need to be added to the MATLAB path. On UNIX, one way to add the program to your path is to set the MATLABPATH environment variable, e.g. for bash users:

```
$ export
MATLABPATH=/tmp/ucsb3d/SpatialAnalysis3D/:/tmp/ucsb3d/userfun
```

This line could be added to your `~/.bashrc` file so that it is run every time you login.

4. Now start MATLAB (the JVM is required):

```
$ matlab
> SpatialAnalysis -setup
```

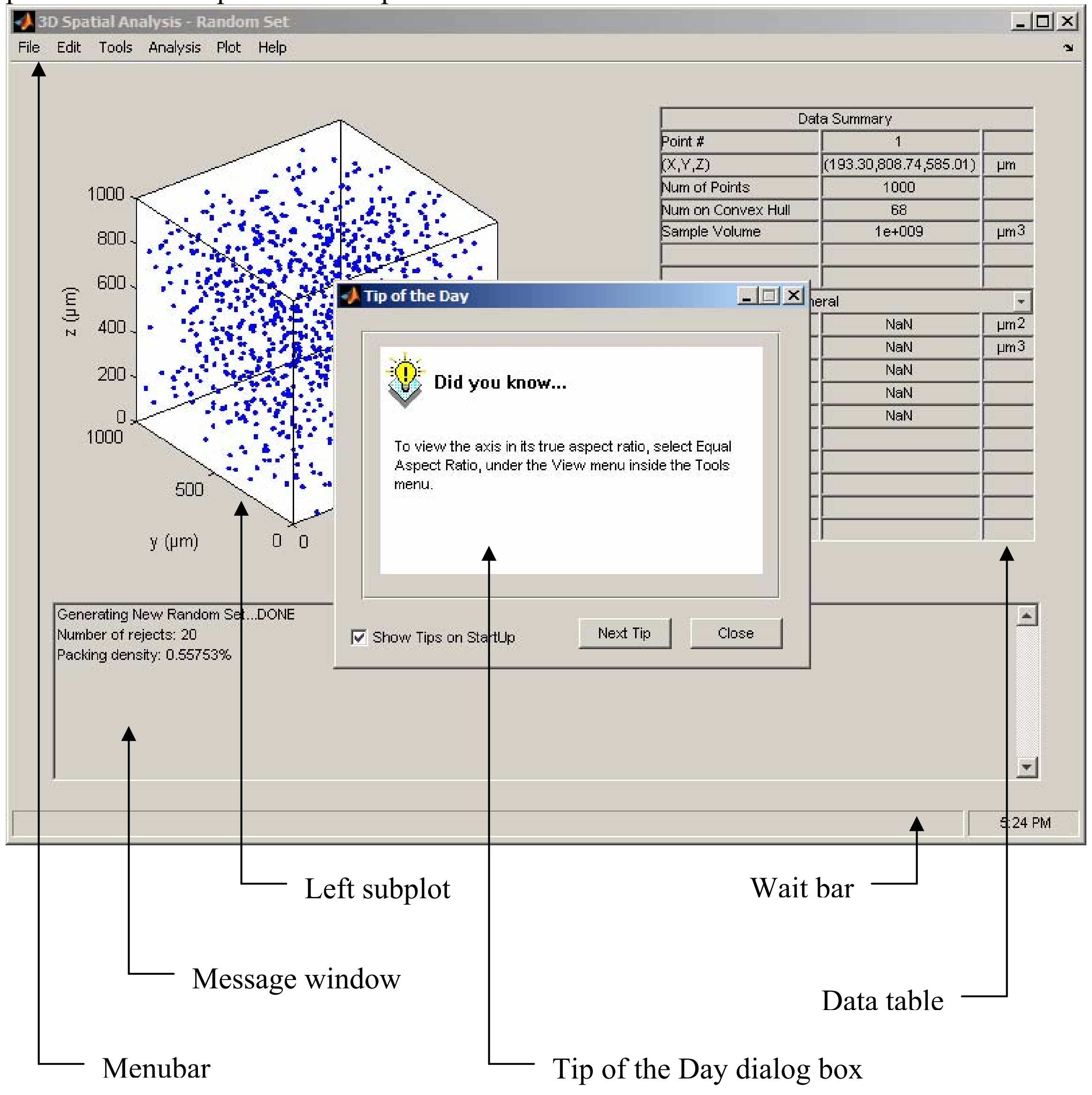
This setup process is needed only once (to compile C functions). Depending on your C compiler, you may get a few warnings.

5. Installation is now complete, and you can start the program within MATLAB simply by typing:

```
> SpatialAnalysis
```

The first time Spatial Analysis 3D starts, the Tip of the Day dialog box will be present. You can scroll through the tips one at a time, view a new one each time you start Spatial Analysis 3D, or close the Tip of the Day for all future sessions. If you do choose to close the Tip of the Day at startup, you can retrieve it from the Help menu or under the options menu.

The following page shows a picture of the Spatial Analysis 3D software. Below the picture is a description of each part of the main window.



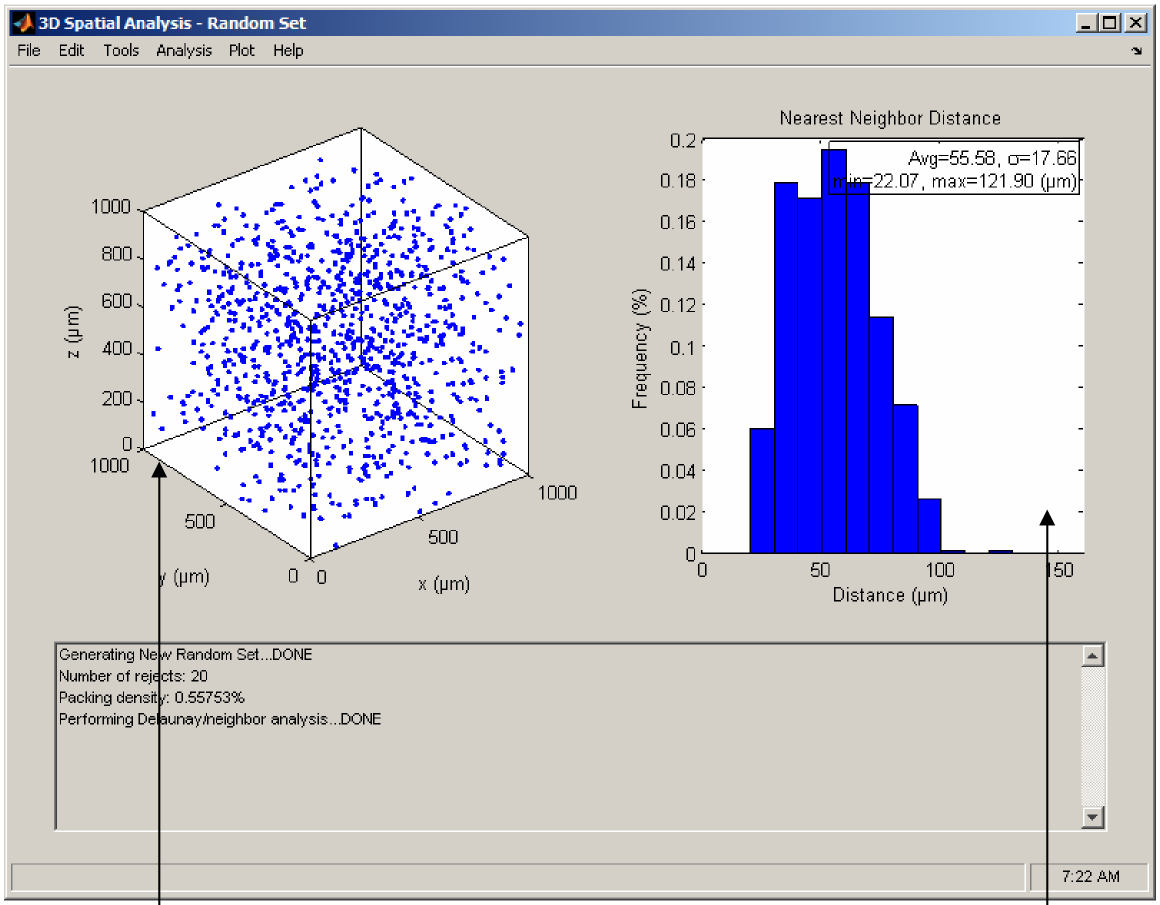
Data Summary		
Point #	1	
(X,Y,Z)	(193.30,808.74,585.01)	µm
Num of Points	1000	
Num on Convex Hull	68	
Sample Volume	1e+009	µm3

```
Generating New Random Set...DONE
Number of rejects: 20
Packing density: 0.55753%
```

Did you know...

To view the axis in its true aspect ratio, select Equal Aspect Ratio, under the View menu inside the Tools menu.

Following certain plotting commands, the data table is hidden from view and a second plot is visible on the right hand side of the main window. An example of this is shown on the following page.


3D Spatial Analysis - Random Set
File Edit Tools Analysis Plot Help
z (µm)
y (µm)
x (µm)
Nearest Neighbor Distance
Avg=55.58, σ=17.66
min=22.07, max=121.90 (µm)
Frequency (%)
Distance (µm)
7:22 AM

```
Generating New Random Set...DONE
Number of rejects: 20
Packing density: 0.55753%
Performing Delaunay/neighbor analysis...DONE
```

Left subplot: Includes all the analysis plots, false color plots and the autocorrelogram

Right subplot: Includes all the histogram plots, including the density recovery plot

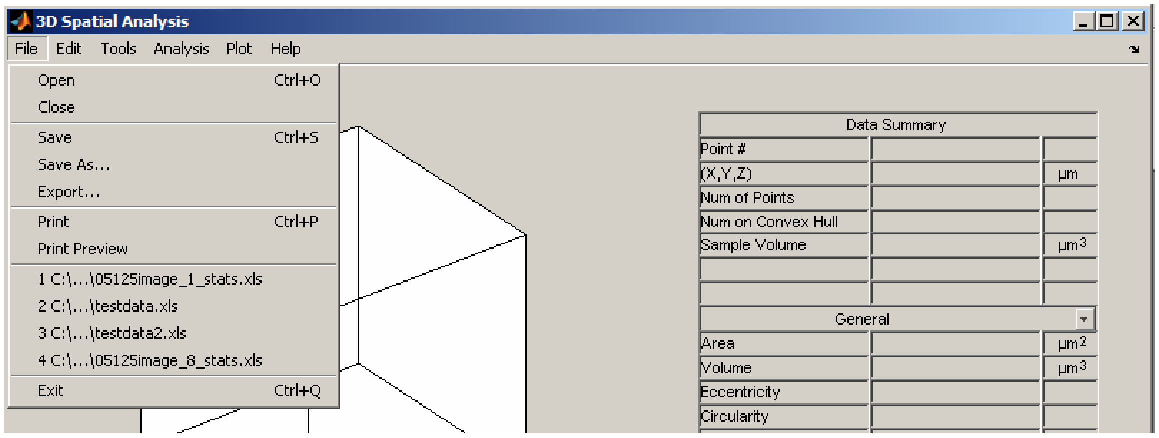
The next chapter describes the operation of the software. Throughout the chapter, the analyses and plots that are illustrated will be taken from the example file, `UserguideData.sa3`, that is supplied with the software. It is recommended that one follow along using the data supplied in the example file.

Chapter 3 Running the Software

3-2	**File Menu**
3-6	**Edit Menu**
3-9	**Tools Menu**
3-21	**Analysis Menu**
3-31	**Plot Menu**
3-46	**Help Menu**

File Menu

Any operation that is file related is handled in the **File Menu**. Under the **File Menu** you can open or close data files, export statistical data, print the graphs, or exit the program.



Opening and Closing Data Files

To open a data file that contains x,y,z data, select the **Open Menu** or use the Ctrl+O keyboard shortcut. You can also choose from the four most recently opened data files beneath the Print Preview Menu item.

The four supported file types for data files are *.xls, Excel worksheet files; *.csv, comma separated values files; *.txt, tab-delimited text files; and *.sa3, Spatial Analysis 3D files. With the first three file types, the file format is generally the same. The first line in the file is a list of column headings. The next N lines are the data values separated by the appropriate delimiter. The *.sa3 files are Matlab binary files that can only be opened by Spatial Analysis 3D. They allow the user to save the current session of the software and open it at a later time.

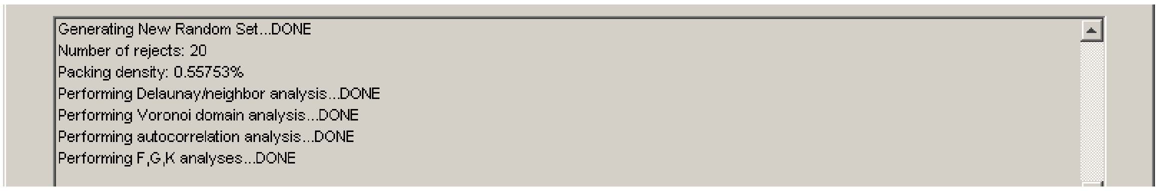
The column headings in the first line must contain the following three values: ‘X’, ‘Y’, and ‘Z’, where the single quotation marks simply represent the fact that X, Y, and Z are characters and are not explicitly used in the file. These headings are capitalized, however. Further headings must not contain any special characters (*&%^$) or spaces. If spaces are found, Spatial Analysis 3D will simply remove them. An example of a *.csv file is shown below:

```
ObjectID,X,Y,Z,T,Volume,Area,Circularity,AverageIntensity,SurfaceArea
294,292.09570,1195.7265,80,1,49.438476,12.359619,0,3342.5,31.72613525
293,298.72460,664.98242,80.33,1,37.078,12.359619,0,3430.664,19.08319
292,495.9414,635.773,78.5,1,49.43847,18.5394,0,3012.25,19.08319
291,282.1513,533.2285,77,1,37.07885,6.17980,0,3299.664,12.35961
...
```

Exporting Data

Spatial Analysis 3D has the ability to export all the statistical data generated in the program to a tab-delimited text file. This file can then be read into any program that reads text files such as Excel.

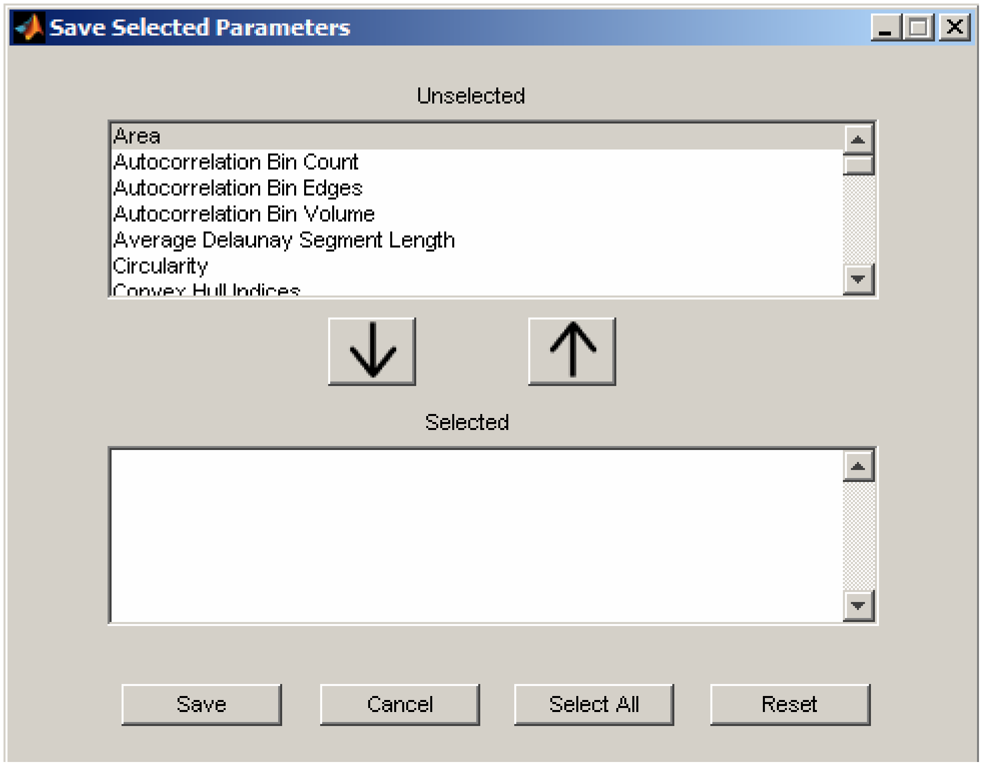
When the **Export Menu** is selected, the program checks to see if the three main analyses (Voronoi, near neighbor/Delaunay, autocorrelation) as well as the F, G, and K analyses have been performed. If they have not, they are performed at this time. Their success is echoed in the message window:



```
Generating New Random Set...DONE
Number of rejects: 20
Packing density: 0.55753%
Performing Delaunay/neighbor analysis...DONE
Performing Voronoi domain analysis...DONE
Performing autocorrelation analysis...DONE
Performing F,G,K analyses...DONE
```

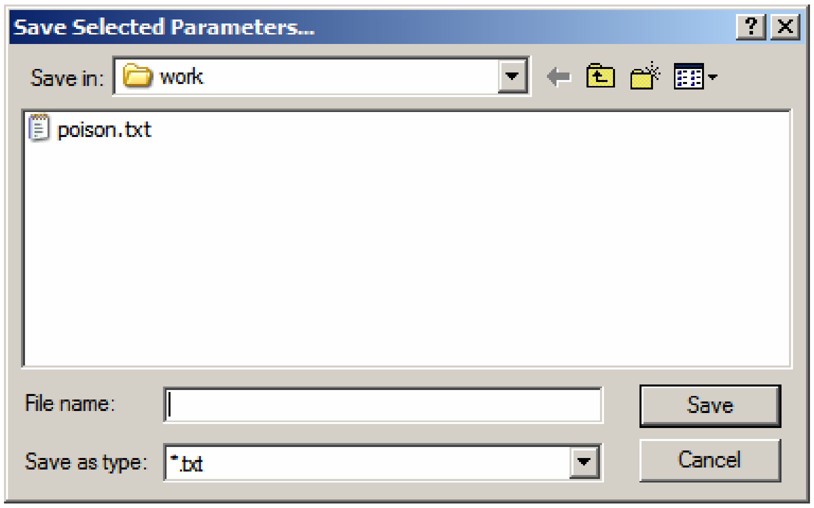
Note that when these analyses are performed under the **Export Menu**, no new plots or graphics are generated in either subplot.

After the analyses have been performed, the program will bring up the **Save Selected Parameters** dialog box.



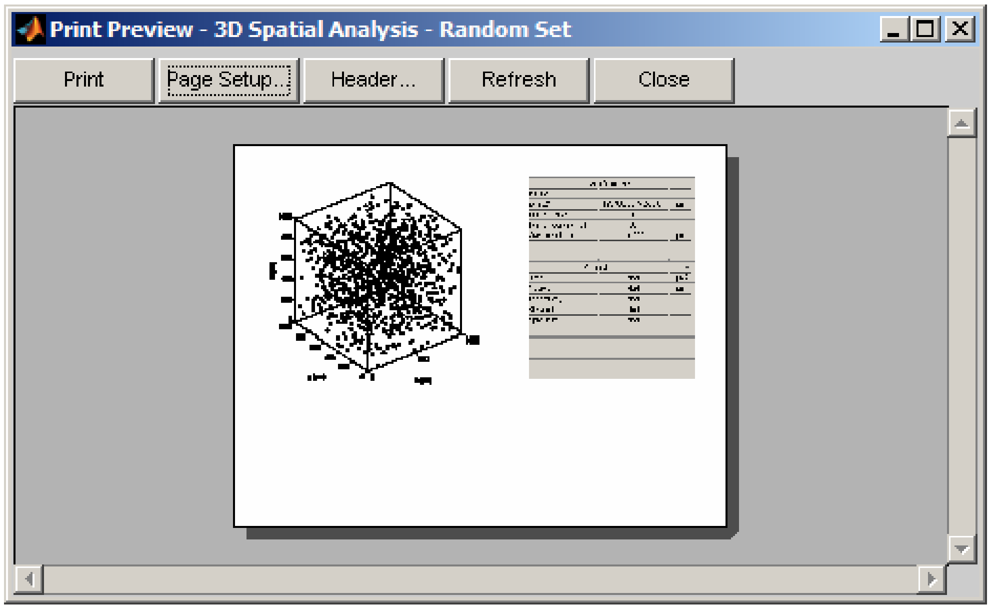
Save Selected Parameters
Unselected
Area
Autocorrelation Bin Count
Autocorrelation Bin Edges
Autocorrelation Bin Volume
Average Delaunay Segment Length
Circularity
Convex Hull Indices
Selected
Save
Cancel
Select All
Reset

You can select parameters to save by highlighting them in the top (Unselected) window and moving them to the lower (Selected) window by using the down arrow button, and vice versa. To select all the parameters, simply click on the **Select All** button. Once you have selected the parameters you wish to save, click on the Save button. This will bring up the familiar **Save As** dialog box. Choose a location and filename for your data, and the program will save the parameters you selected as a tab-delimited text file.

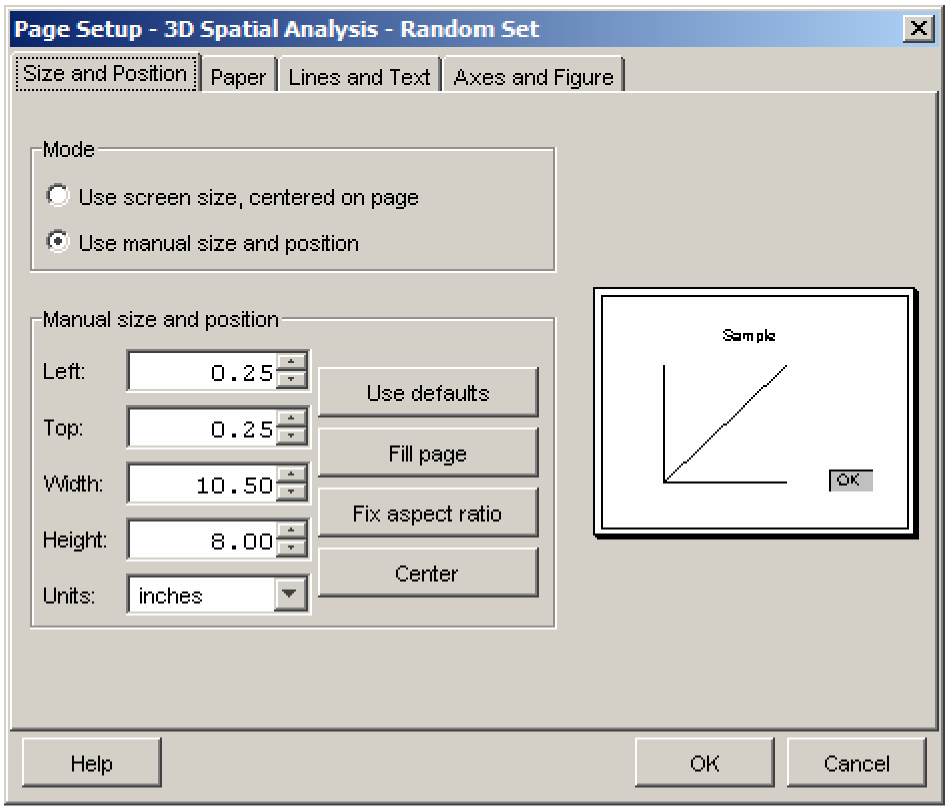


Printing

You can print the main window by selecting the **Print Menu** or use the Ctrl+P keyboard shortcut. You can also preview the print job by selecting the **Print Preview Menu**. Before the main window is printed, the message window, status bar, and the clock are made invisible. They are returned to the visible state after the window has been printed, or the print preview window has been closed.



Within the print preview window, you can choose the page setup that defines how the window will be printed on the page. Simply click on the **Page Setup** button near the top of the window. Here you can select the paper size, paper orientation, the position of the figure on the page, etc. When you are finished with the page setup, click on the **OK** button and you will be taken back to the print preview window.


Page Setup - 3D Spatial Analysis - Random Set

Mode: ( ) Use screen size, centered on page [✓ Use manual size and position]

Manual size and position

Left: 0.25

Top: 0.25

Width: 10.50

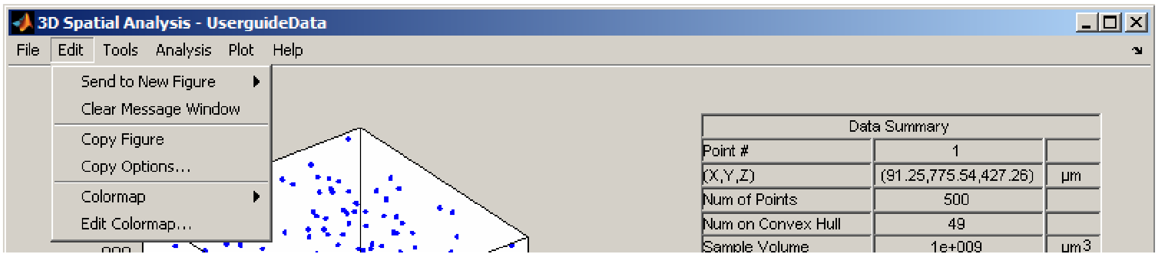
Height: 8.00

Units: inches

While some of these windows will look different on a UNIX or Linux operating system, the basic functionality should remain the same.

Edit Menu

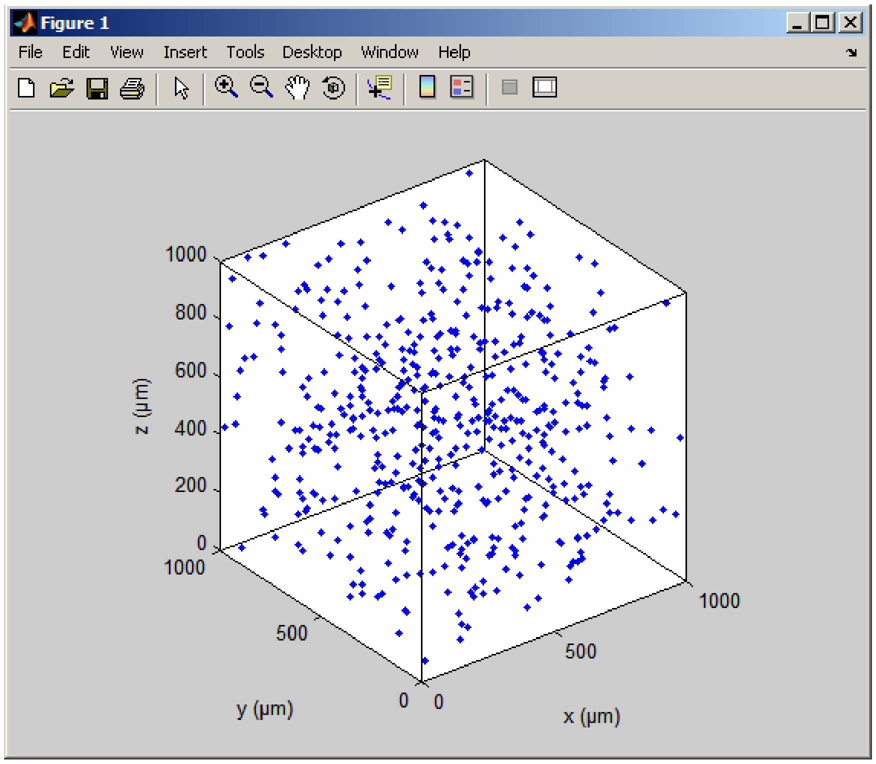
Operations that result in modifying the main window are performed in the **Edit Menu**. These operations include: sending a subplot to a new window, clearing the message window, copying the figure to the clipboard, and modifying the colormap.



Data Summary		
Point #	1	
(X,Y,Z)	(91.25,775.54,427.26)	µm
Num of Points	500	
Num on Convex Hull	49	
Sample Volume	1e+009	µm3

Copying and Sending the Figure Window

You can copy the figure window to the clipboard or send either left or right subplot to a new figure window from the **Edit Menu**. To send the left subplot to a new figure, simply click on the **Send to New Figure** menu and choose **Left Subplot**.



Here the data points plot is copied to a new Matlab figure window. This figure has full Matlab capability such as zoom, rotation, annotation, etc. Use this feature, for example, to prepare figures to put into PowerPoint.

You can also copy the main figure window to the clipboard by using the **Copy Figure** menu. This can be useful if you would like to copy the entire window into PowerPoint for example. The format by which the figure is copied is defined inside the **Copy Options** menu.

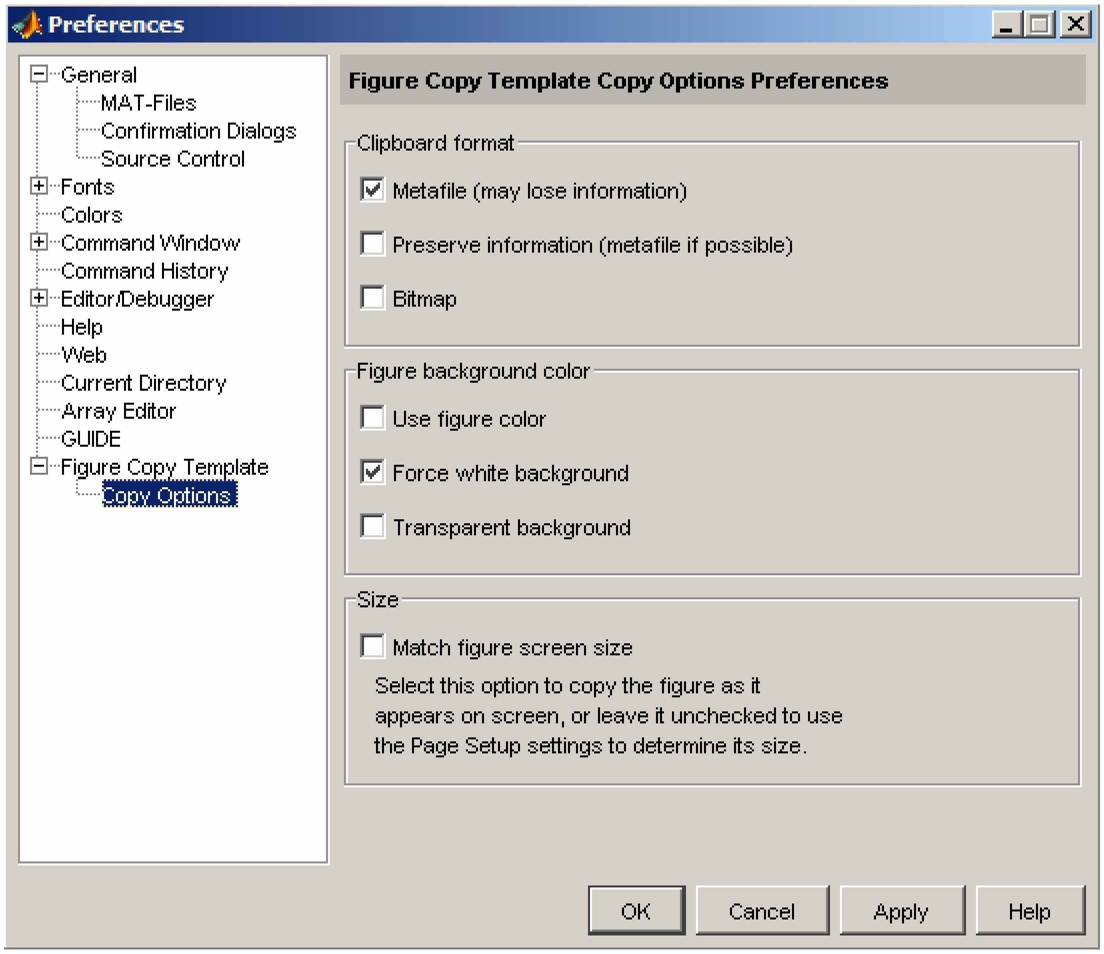


Figure Copy Template Copy Options Preferences

Clipboard format
- [✓] Metafile (may lose information)
- [ ] Preserve information (metafile if possible)
- [ ] Bitmap

Figure background color
- [ ] Use figure color
- [✓] Force white background
- [ ] Transparent background

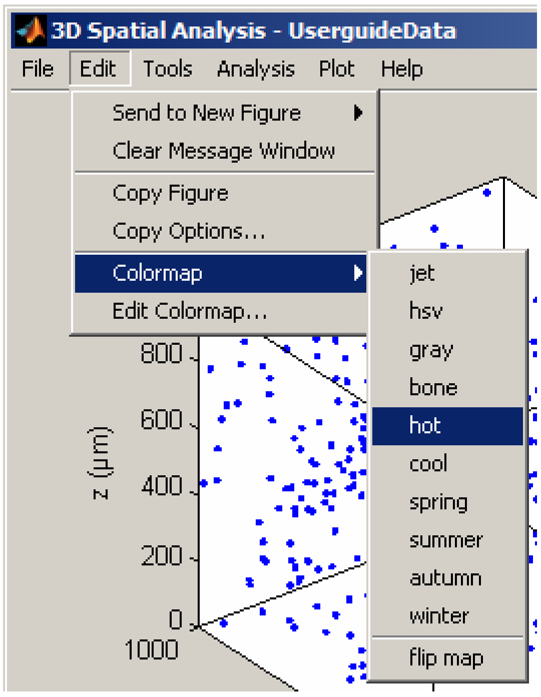
Size
- [ ] Match figure screen size

Select this option to copy the figure as it appears on screen, or leave it unchecked to use the Page Setup settings to determine its size.

For Windows users, copying a figure as a metafile will allow for maximum retention of graphics information and will produce the best-looking graphics. However, with some very complicated graphics, copying as a metafile will cause Matlab to hang up as it is trying to replicate the metafile onto the clipboard. In these cases, copy as a bitmap.

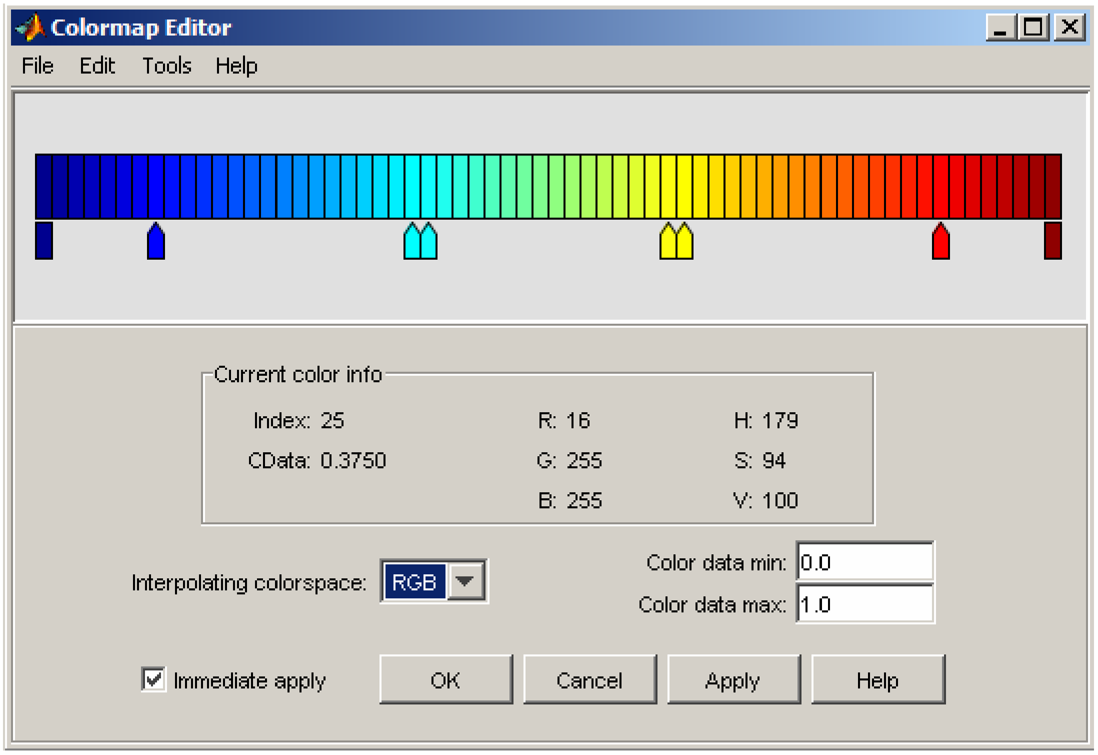
Editing the Colormap

Each MATLAB figure window has a colormap associated with it. A colormap is simply a three-column matrix whose length is equal to the number of colors it defines. Each row of the matrix defines a particular color by specifying three values in the range 0 to 1. These values define the RGB components (i.e., the intensities of the red, green, and blue video components). Matlab then maps the data that is plotted to this colormap; the lowest value to the first color and the highest value to the last color with a linear relationship in between.



Choose between different colormaps by selecting them from the **Colormap Menu**. You can invert the colormap by selecting the **flip map** menu.

You can even modify the current colormap to your own liking by choosing **Edit Colormap…** from the **Edit Menu**. When you do, the Colormap Editor will appear with the current colormap loaded. Change the position, color, or number of markers that define the colormap. When you are finished, apply the new colormap and continue.



Current color info

Index: 25 R: 16 H: 179

CData: 0.3750 G: 255 S: 94

B: 255 V: 100

Interpolating colorspace: RGB

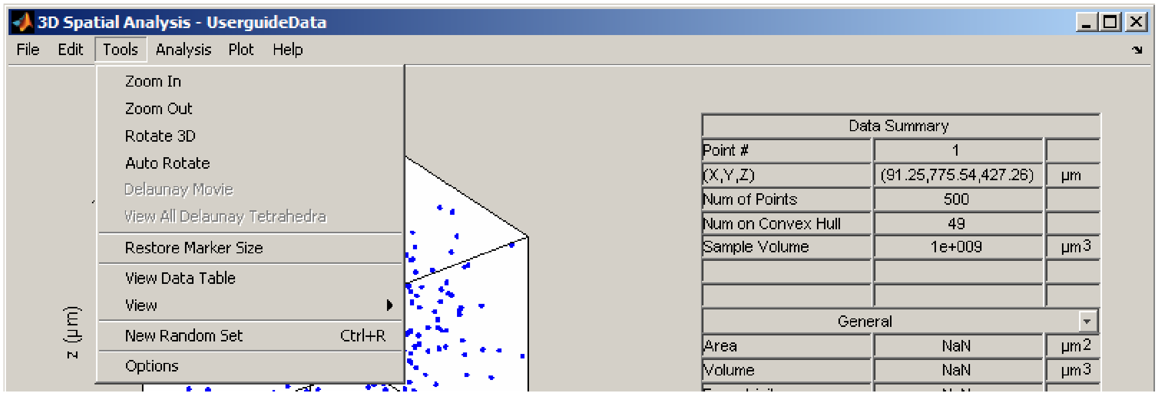
Color data min: 0.0

Color data max: 1.0

[✓] Immediate apply

Tools Menu

Spatial Analysis 3D is equipped with a set of tools that operate on the plots and data table. The **Tools Menu** provides access to these tools and to the options for the software.



Data Summary		
Point #	1	
(X,Y,Z)	(91.25,775.54,427.26)	µm
Num of Points	500	
Num on Convex Hull	49	
Sample Volume	1e+009	µm3
General		
Area	NaN	µm2
Volume	NaN	µm3

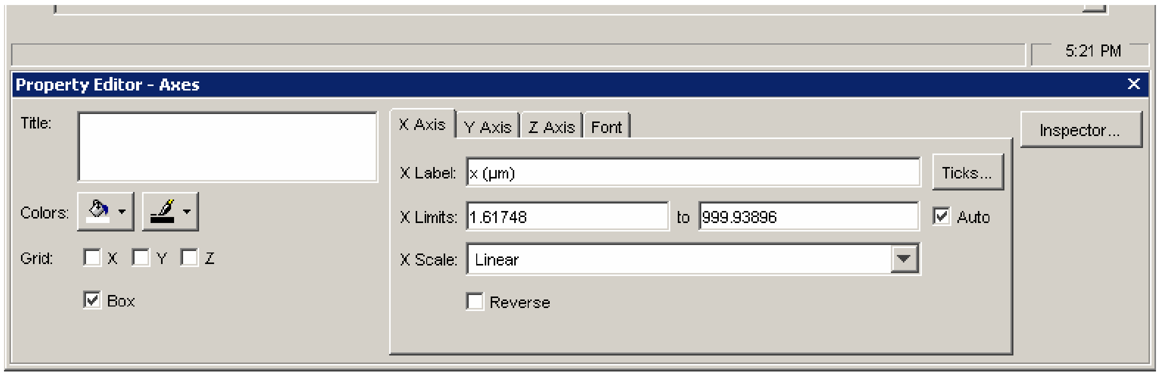
Plot Manipulation

Included in Spatial Analysis 3D are four tools that can be used to manipulate the plot on the left-hand side of the application (left subplot), **Zoom In**, **Zoom Out**, **Rotate 3D**, and **Auto Rotate**. Like the names imply, the zoom commands zoom in and out of the current plot. **Zoom In** will zoom in toward the center of the plot by 2X. In contrast, **Zoom Out** will zoom out from the center of the plot by 2X. When the plot is zoomed in, the subplot will be clipped by any of the following: the data table, message window, or the application itself. If the zoom commands are not having the desired effect, try using the **Send to New Figure** command in the **Edit Menu**, and modifying the plot with a full set of Matlab tools.

There are two ways you can rotate a plot within Spatial Analysis 3D: **Rotate 3D** or **Auto Rotate.** **Rotate 3D** enables you to rotate the plot to any orientation with the mouse. Rotation involves the reorientation of the axes and all the graphics objects contained in the plot. Therefore, none of the data defining the graphics objects is affected by rotation; instead the orientation of the x-, y-, and z-axes change with respect to the viewer. When enabled, **Rotate 3D** provides continuous rotation of the plot and the objects it contains through mouse movement. A numeric readout appears in the lower left corner of the figure during rotation, showing the current azimuth and elevation of the axes. Releasing the mouse button removes the animated box and the readout. To return the plot to its original orientation, double click on the plot with the rotation arrow still enabled.

You can also rotate the plot automatically by using the **Auto Rotate** command. In this way, you can view the plot as it spins around without using the mouse. During the rotation, the plot will start in perspective view (30º elevation, -37.5º azimuth) and rotate through 360º of azimuth with the elevation fixed at 30º. Once the Auto Rotate feature has been started, there is no way to exit the rotation. You simply have to wait for the rotation to stop before you proceed.

In addition to the plot manipulation tools available in the **Tools Menu**, you can also use Matlab's built-in Property Editor to manipulate and edit any of the plots (even the histograms in the right subplot). To enable the Property Editor, simply double click on the desired subplot when no other tools are enabled (i.e. **Rotate 3D**). When the Property Editor is enabled, it will appear at the bottom of the Spatial Analysis 3D figure. The Property Editor window is fixed in its location. You can, however, resize the window and move both windows around simultaneously.


Property Editor - Axes

Title:

Colors:

Grid: ☐ X ☐ Y ☐ Z

☑ Box

X Axis | Y Axis | Z Axis | Font

X Label: x (µm)

X Limits: 1.61748 to 999.93896 ☑ Auto

X Scale: Linear

☐ Reverse

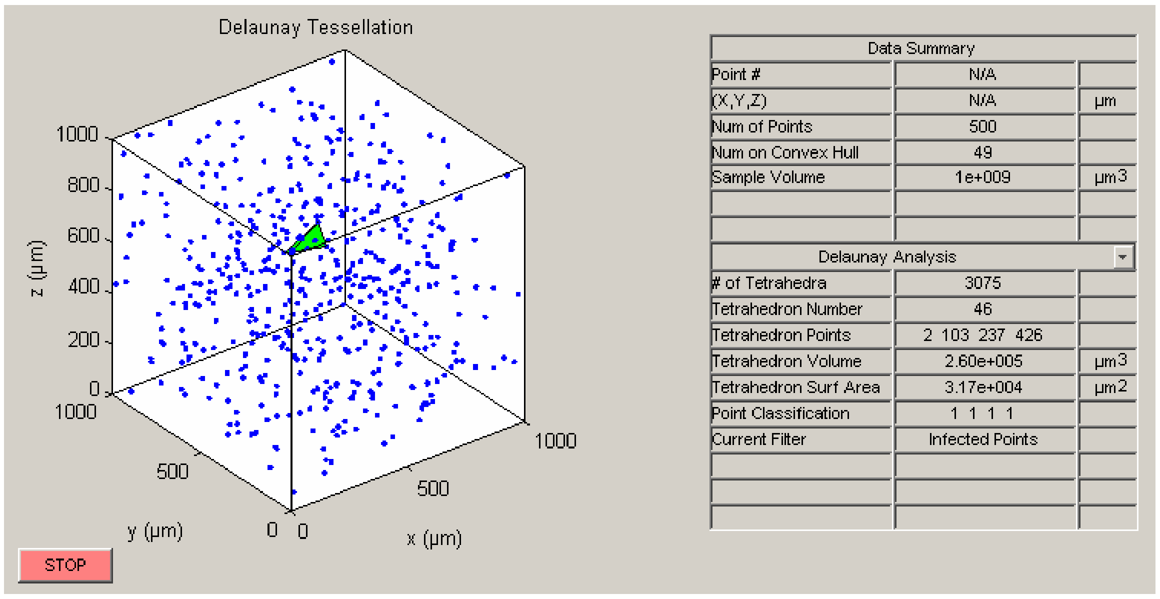
Once the Property Editor has been enabled, you can click on any object in the figure window to view and/or modify that object's properties. Be careful, as changing the properties of certain objects in the figure may have undesirable effects to the Spatial Analysis 3D application.

For more information on the Property Editor, see the Matlab help documentation.

View All Delaunay Tetrahedra

After the Delaunay Tessellation analysis is performed, one can view all the tetrahedra in succession by choosing the **View All Delaunay Tetrahedra** under the **Tools Menu**. This menu item becomes active only after a Delaunay analysis has been performed on the loaded dataset. **View All Delaunay Tetrahedra** will cycle through all tetrahedra that have not been filtered and render them sequentially in the left subplot. As the program moves from tetrahedron to tetrahedron, the individual statistics for each is displayed in the data table. There are often a conspicuously large number of tetrahedra, so minimum time is spent on each one.

Because this operation may take significantly longer than the user anticipated, a red stop button is made visible that the user can use to stop the operation. One may also hit the ESC key to stop the operation as well.



Data Summary		
Point #	N/A	
(X,Y,Z)	N/A	µm
Num of Points	500	
Num on Convex Hull	49	
Sample Volume	1e+009	µm3
Delaunay Analysis		
# of Tetrahedra	3075	
Tetrahedron Number	46	
Tetrahedron Points	2 103 237 426	
Tetrahedron Volume	2.60e+005	µm3
Tetrahedron Surf Area	3.17e+004	µm2
Point Classification	1 1 1 1	
Current Filter	Infected Points	

Restore Marker Size

When plotting the False Color Maps (see **Plot Menu** section), the current point is plotted as a larger sphere than it should be. This is to highlight which point is current. If you would like to view the plot with all points at their default size, select the **Restore Marker Size Menu**. This can be especially useful if you send the plot to a new figure for presentation purposes. The **Restore Marker Size** tool does not affect any plot other than the False Color plots.

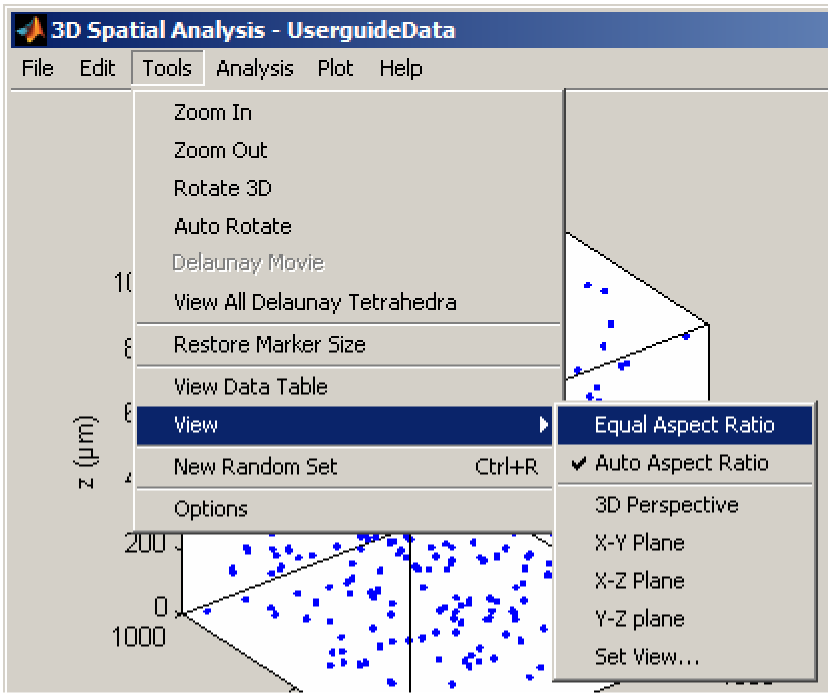
View Data Table

Many of the plotting commands use the right subplot to display the plot. If instead of the right subplot you wish to view the data table, select **View Data Table** from the **Tools Menu.**

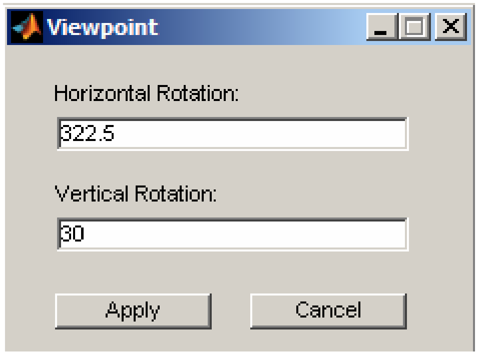
View Menu

The **View Menu** determines the view orientation and aspect ratio of all the plots in the left subplot. By default the perspective is set to 3D perspective and the aspect ratio is set to auto.

The auto **Aspect Ratio** mode that is enabled by default allows Matlab to plot graphics in any shape axis. This means the x-,y-, and z-axes do not have to have the same relative scale. In the case of Spatial Analysis 3D, this distorts the plots and makes it appear to always have a cubic volume. To fix all the axes to the same relative scale, select the **Equal Aspect Ratio Menu**.



The default view for all plots in the left subplot is 3D Perspective (30º elevation, 322.5º (-37.5º) azimuth). You can also select between any of the 2D orientations (e.g. X-Y Plane). If you would like to choose a custom orientation, select the **Set View…** menu.


Viewpoint

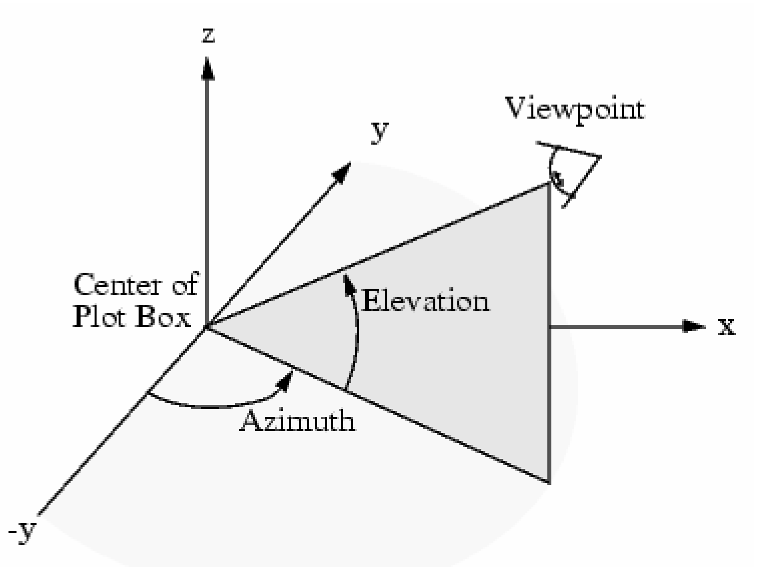
Horizontal Rotation: 322.5

Vertical Rotation: 30

Choosing **Set View…** in the **View Menu** allows the orientation to be freely set.

Azimuth is a polar angle in the x-y plane, with positive angles indicating counterclockwise rotation of the viewpoint. Elevation is the angle above (positive angle) or below (negative angle) the x-y plane.

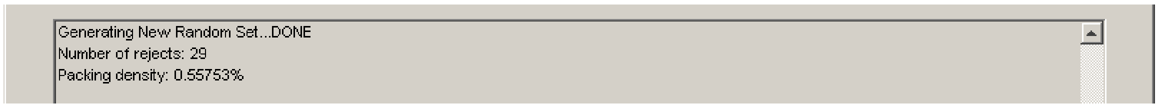
The following diagram illustrates the coordinate system. The arrows indicate positive directions.



New Random Set

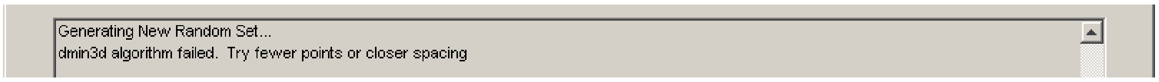
Spatial Analysis 3D has the ability to create random and periodic datasets. Currently, the only periodic dataset available is hexagonal close packed (HCP). To select the parameters governing the creation of these datasets, see the next section, **Options**. When you would like to create a new dataset, however, choose **New Random Set** from the **Tools** menu.

When a new random set is generated, several messages are printed into the message window. If the generation of random data is successful, a message will be printed that tells you the random set has been generated. It will also tell you how many rejected points there were based on the generation criteria, and finally the packing density of the points. Some of the simulated points were rejected because their positioning overlapped with previously generated points, where each point is also simulated to have some physical size.



```
Generating New Random Set...DONE
Number of rejects: 29
Packing density: 0.55753%
```

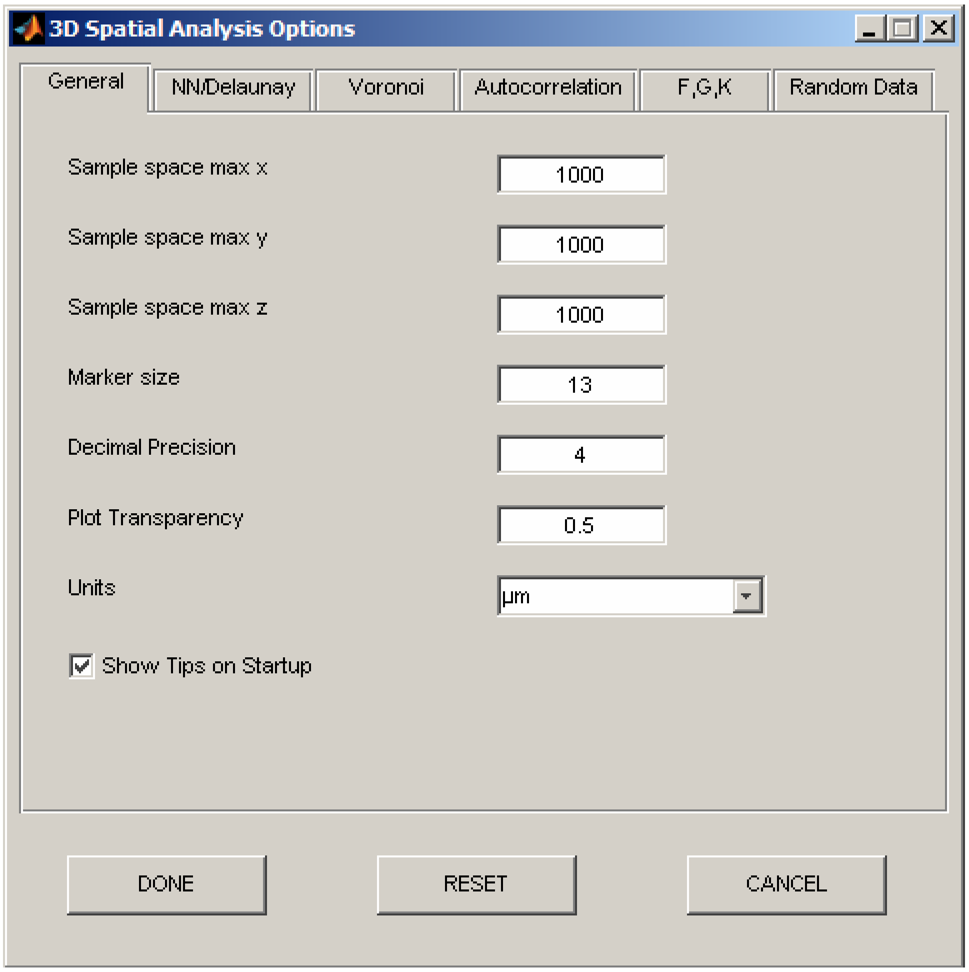
It is possible for a random set generation to fail, for example, if the mean minimal distance is too large for the number of points desired. When this happens, an error message is printed with a suggestion on how to proceed.



```
Generating New Random Set...
dmin3d algorithm failed.  Try fewer points or closer spacing
```

Options

The Spatial Analysis 3D software has many different options that determine the behavior of the analyses. To change these options, simply select the **Options** menu from the Tools menu. After selecting the **Options** menu, you will see the options figure with the **General** tab selected. When you are finished modifying the options, simply click on the **DONE** button, and all the new options will be saved. To exit without saving the changes, click the **CANCEL** button. Clicking the **RESET** button will restore the options back to the default settings, but only under the currently selected tab.

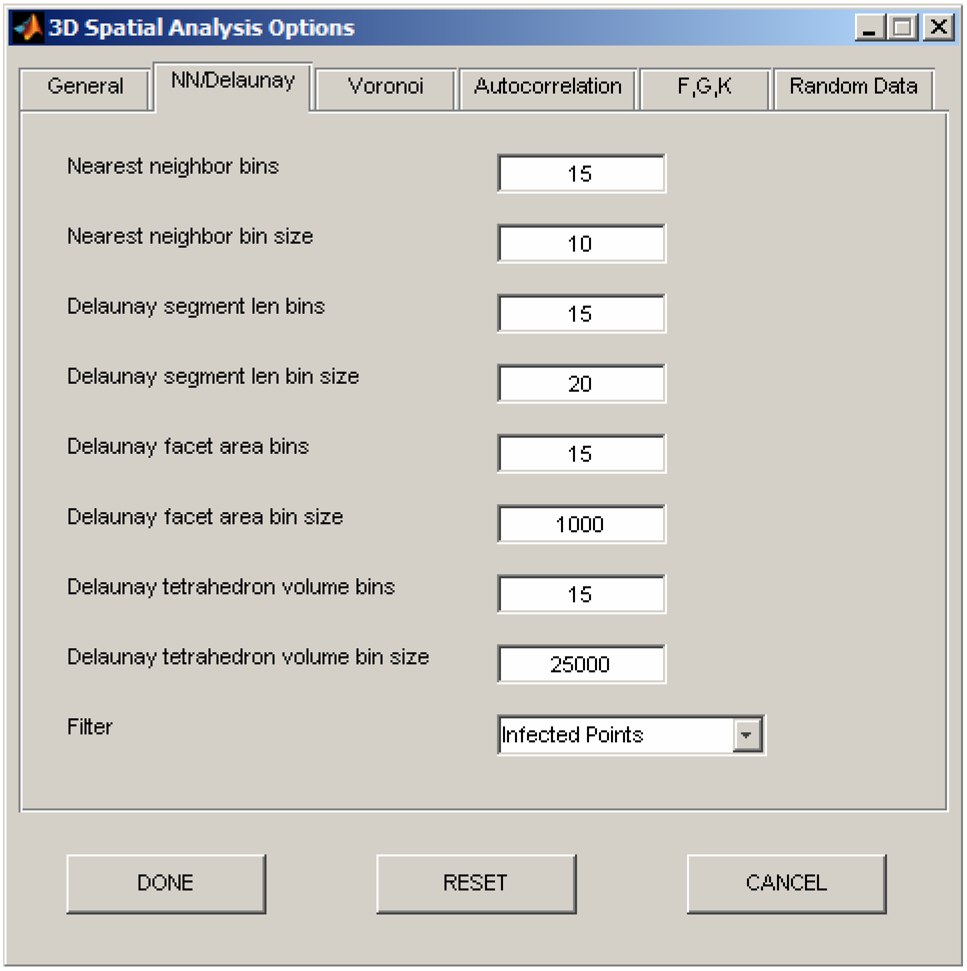


Sample space max x	1000
Sample space max y	1000
Sample space max z	1000
Marker size	13
Decimal Precision	4
Plot Transparency	0.5
Units	µm

☑ Show Tips on Startup

With the General tab selected, you can change options that are general to the overall operation of Spatial Analysis 3D. This is where you define the exact location of the boundaries with respect to the data, and the units used. You can also change the default marker size for the plots, the transparency used when creating transparent patch objects, and the decimal precision used in the data table. The final feature that can be controlled from the General tab is the Tips dialog box that shows when the software starts. If you choose to not show the Tips at startup, but later decide you do want to see the Tips at startup, you can choose that option here.

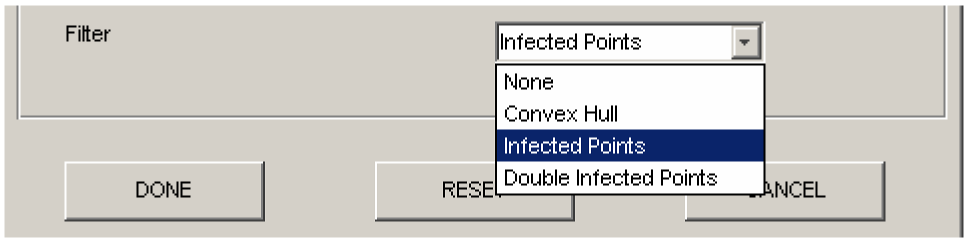
The next tab in the Options figure is the **NN/Delaunay** tab. As the name implies, this is where you change the options related to the near neighbor and Delaunay analysis. It may not be immediately obvious, but the Delaunay tessellation is necessary in determining the near neighbors. Thus, these two analyses are combined under one tab.



Nearest neighbor bins	15
Nearest neighbor bin size	10
Delaunay segment len bins	15
Delaunay segment len bin size	20
Delaunay facet area bins	15
Delaunay facet area bin size	1000
Delaunay tetrahedron volume bins	15
Delaunay tetrahedron volume bin size	25000
Filter	Infected Points

It is often important to compare like datasets using histograms with the same bin size and number of bins. Therefore, under the NN/Delaunay tab, you can choose the bin size and number of bins for all the histogram plots related to near neighbor or Delaunay analysis. We will see later in the **Plot Section** how one can go about determining the appropriate bin size when nothing is known about the dataset.

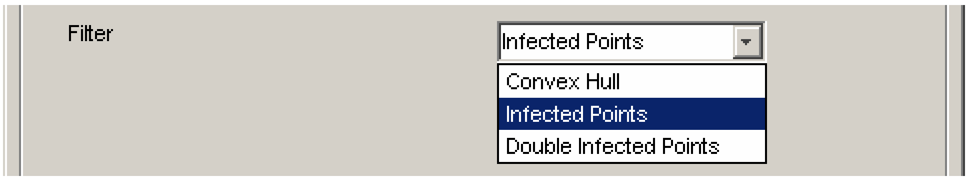
The final option that can be set here is the Filter for the NN/Delaunay analysis. The current choices for this filter are: “None”, “Convex Hull”, “Infected Points”, and “Double Infected Points”.

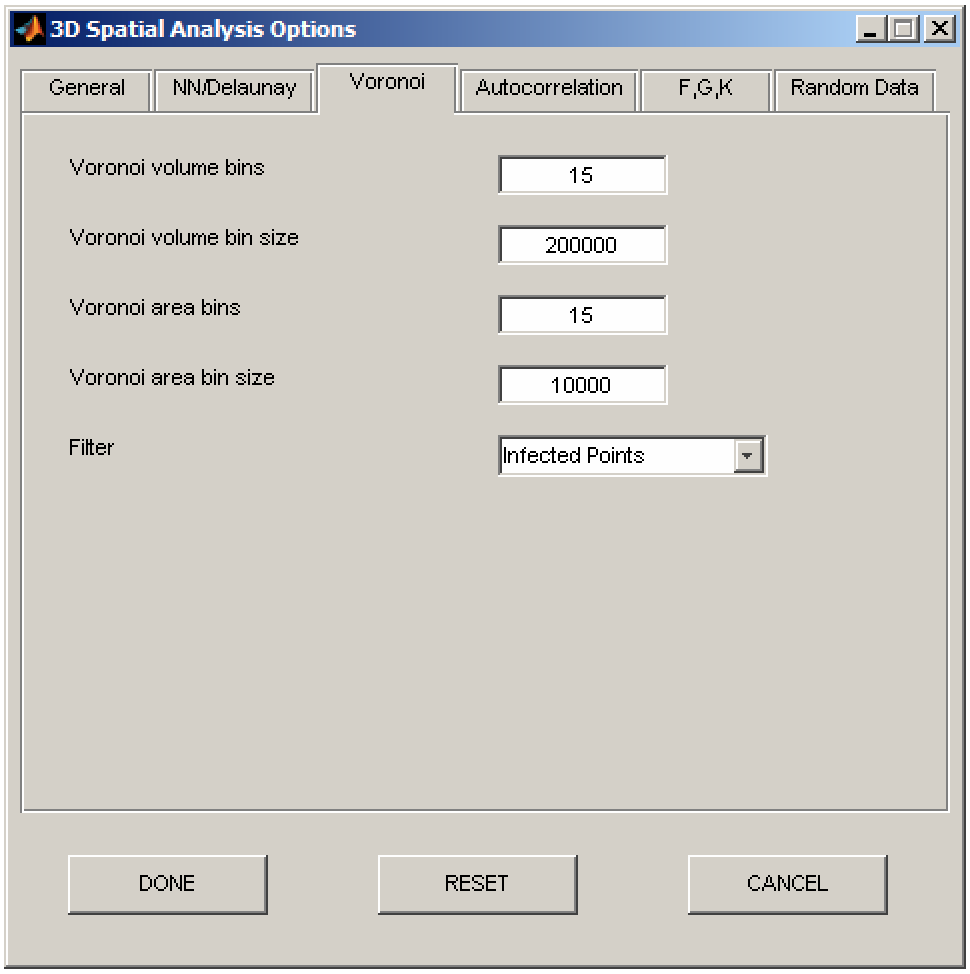


When the filter is set to “None”, all the points in the sample space are used in the near neighbor and Delaunay statistics. Those points that make up the convex hull, defined as the smallest convex region that contains the data set, can be excluded by choosing “Convex Hull” as the filter. The next level of filtering is “Infected Points”. Infected points are defined as those points that are closer to the boundary than their nearest neighbor. This is intuitive in that it is easy to imagine the possibility that a data point exists just outside the boundary and that this point would become the nearest neighbor if it had been included in the dataset. The most restrictive filter to choose from is “Double Infected Points”. In this case, a point is considered double-infected if any of its near neighbors is infected. It is important to remember that as the filter gets more and more restrictive, fewer data points are available for statistical calculations. Thus, there is a trade off between the size of the dataset and the reliability of the statistics.

The next tab in the Options figure is the **Voronoi** tab. Just like the NN/Delaunay options, it is often important to compare like datasets using histograms with the same bin size and number of bins. Therefore, under the Voronoi tab, you can choose the bin size and number of bins for the Voronoi volume and area histograms.

Under the Voronoi options tab one can also choose a filter for the Voronoi analysis. The choices for the filter are: “Convex Hull”, “Infected Points”, and “Double Infected Points”. “Convex Hull” is the lowest level of filter for the Voronoi analysis. By definition, a point on the convex hull will not have an enclosed Voronoi domain, so these points must be filtered. An infected point is defined as any point that has a Voronoi domain that is enclosed, but one or more of the vertices lies outside the sample space boundary. A double infected point is a point where one or more of the Voronoi vertices is a vertex of an infected point or convex hull point. As with the NN/Delaunay filter, as the Voronoi filter gets more restrictive there is a trade off between reliable data and size of the statistical dataset.





Voronoi volume bins 15

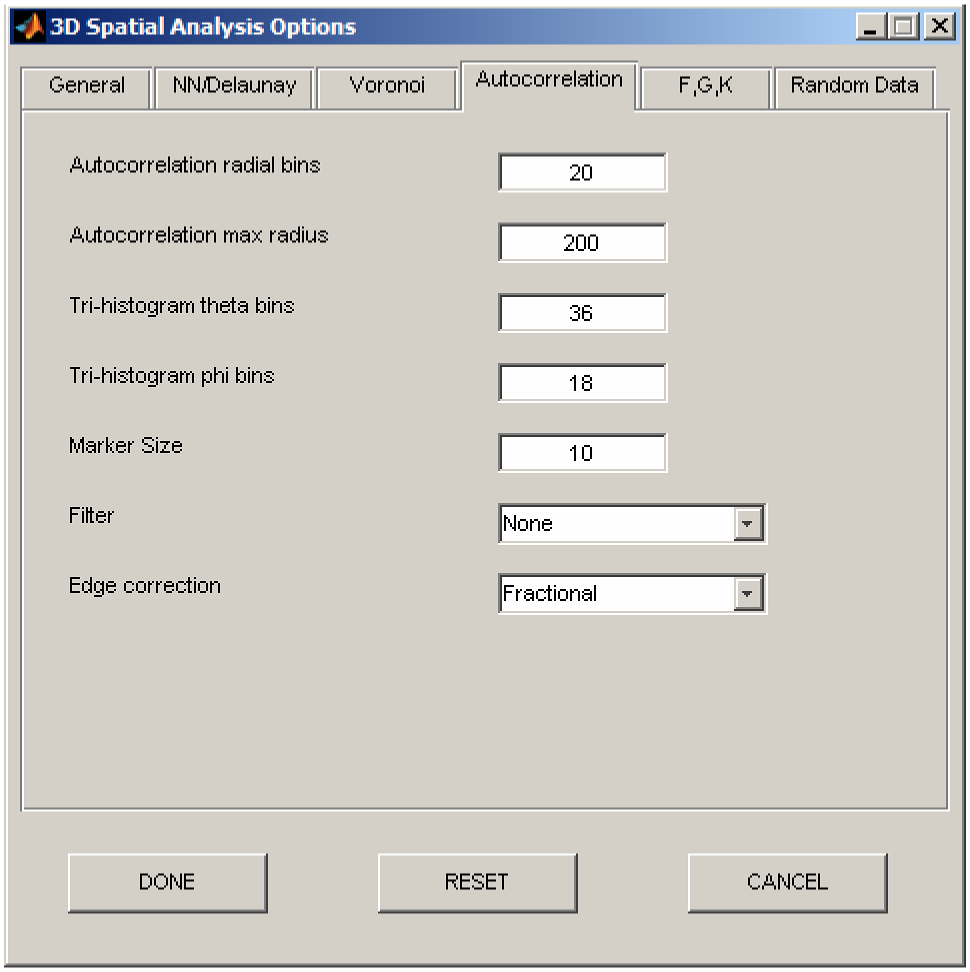
Voronoi volume bin size 200000

Voronoi area bins 15

Voronoi area bin size 10000

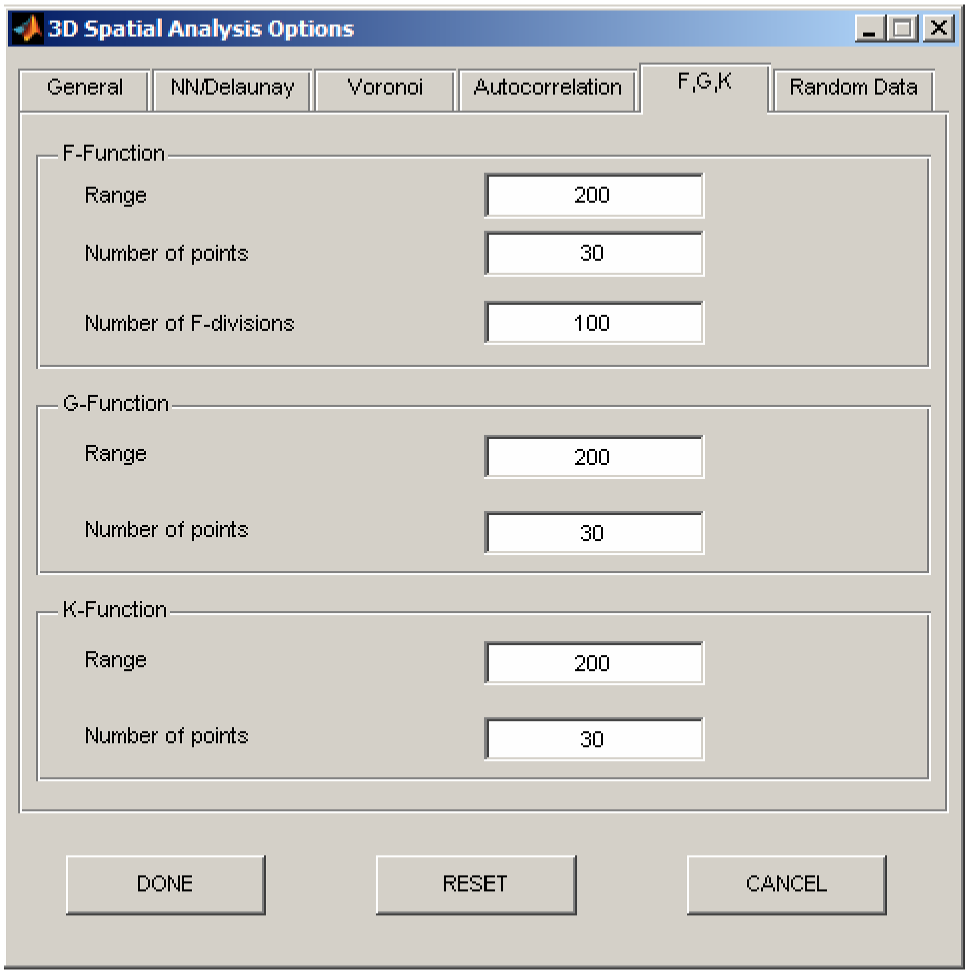
Filter Infected Points

The next tab in the Options figure is the **Autocorrelation** tab. To determine the density recovery profile, and also the tri-histogram plot (see **Plot** section), the number of bins and the maximum radius for the autocorrelation must be known before the autocorrelation is performed. Both of these are set under the **Autocorrelation** tab in the **Options** figure. During the autocorrelation, the program also plots a 3D rendering of the correlation in the left subplot. It does this by plotting each point colored with respect to its density in the density recovery plot, and sized by the option **Marker Size** in the options. The final two options under the Autocorrelation tab are the **Filter** and **Edge Correction** to use during the autocorrelation analysis. The filter determines which points will be used in the autocorrelation analysis. As points get near the boundaries, their near neighbor influence on the autocorrelation is diminished. We can either exclude these points near the boundaries, or add edge correction terms to the calculated densities. The most restrictive of the filters is the "Max Radius", which ignores all points that are closer than the max radius from the boundaries. The only edge correction available is "Fractional" which calculates the fraction of the autocorrelation sphere that lies outside the sample space boundary and scales the density by this fraction.



Autocorrelation radial bins: 20
Autocorrelation max radius: 200
Tri-histogram theta bins: 36
Tri-histogram phi bins: 18
Marker Size: 10
Filter: None
Edge correction: Fractional

The next tab in the Options figure is the **F,K,G** tab. The F, K and G-functions are spatial functions that can evaluate the degree of regularity in a population of cells (see Chapter 1: K, G and F-functions in 3D). The F-function measures the distance from each grid point to its nearest neighboring cell. The K-function is the cumulative version of the density recovery profile. The G-function is a cumulative plot over a range of distances, measuring the fraction of nearest-neighboring distances that are less than or equal to t. For each of these functions, the range over which to calculate them and the number of points in the calculation must be set in the options. The F-function has an additional parameter, “Number of F-divisions”, that must be set. This represents the number of grid points in each direction we will divide our sample space into when performing the calculation.


3D Spatial Analysis Options

General | NN/Delaunay | Voronoi | Autocorrelation | F,G,K | Random Data

F-Function

Range	200
Number of points	30
Number of F-divisions	100

G-Function

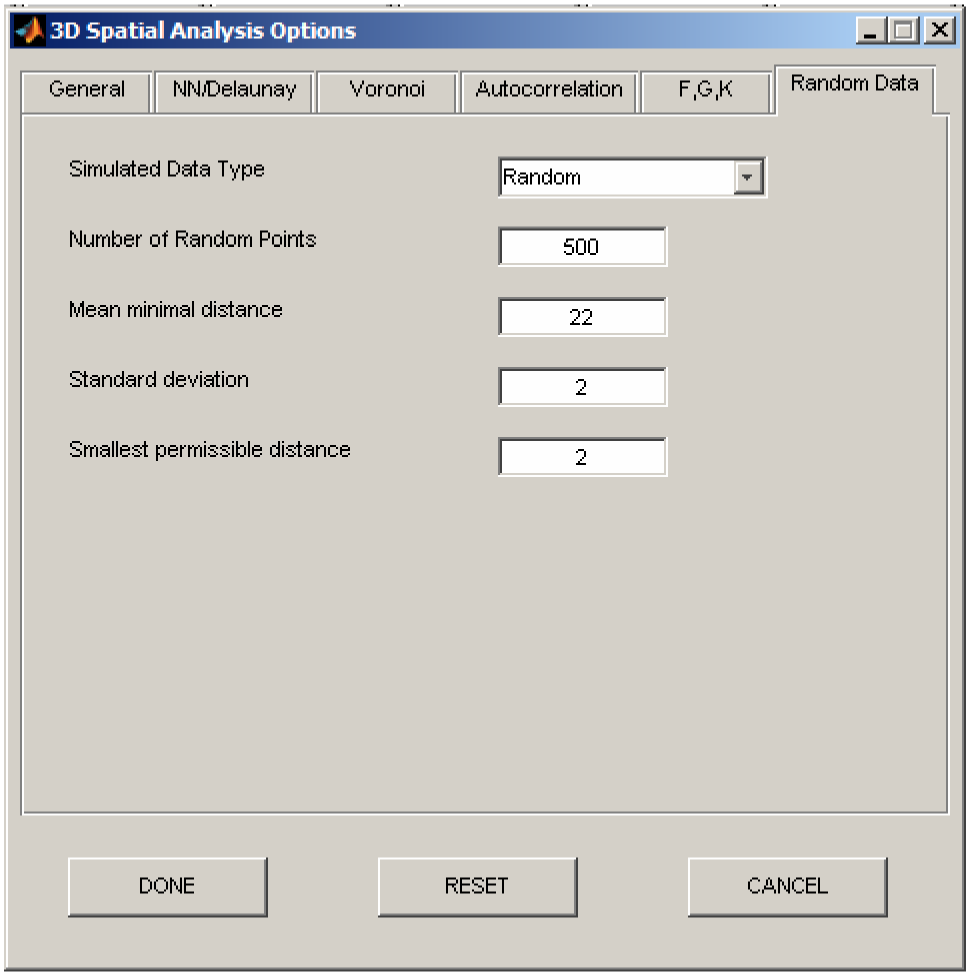
Range	200
Number of points	30

K-Function

Range	200
Number of points	30

The last tab in the **Options** figure is the **Random Data** tab. The **Random Data** tab allows the user to choose the criteria by which a simulated set of data is generated. This simulated data can be used to compare against measured data sets. Currently there are two types of simulated data the user can choose from: **Random** and **HCP** (hexagonal-close-packed).

When **Random** is chosen, the user first chooses how many points will be placed in the sample space. How they are arranged depends on the mean minimal distance established, the standard deviation associated with the mean distance, and the smallest permissible distance, enabling a simulation that respects a physical limitation imposed by, for example, cell size. First, a point is generated inside the sample space in a random location. As each new random point is generated, its shortest distance to all other points is compared against a randomly sampled minimal distance d_{min}, derived from the distribution defined by that mean and standard deviation. Each point is accepted into the simulation if that shortest distance is greater than d_{min}.

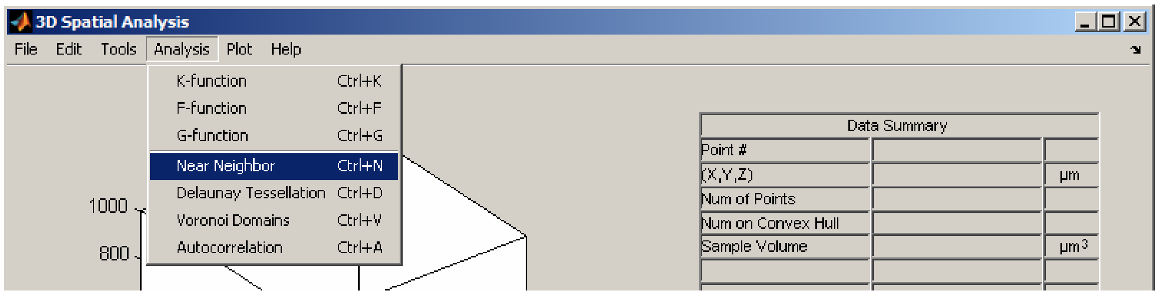


Simulated Data Type: Random
Number of Random Points: 500
Mean minimal distance: 22
Standard deviation: 2
Smallest permissible distance: 2

When **HCP** is selected, a hexagonal lattice will be generated that fills the available volume. The spacing of the lattice is determined by the parameter **Mean minimal distance** in the options menu; from this, the number of data points that are required to fill the volume is automatically calculated. After the lattice has been generated, points are independently moved by adding zero-mean Gaussian noise, with standard deviation determined by the option **Standard deviation**.

Analysis Menu

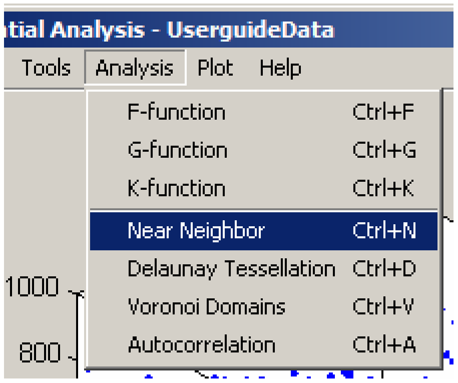
To display any graphics other than the data points, the appropriate data analysis must first be performed. All analysis tools are under the **Analysis Menu.**



You may select the desired analysis by choosing it from the **Analysis Menu** or by using the appropriate keyboard shortcut (e.g. Ctrl+N). Some of the data analysis under this menu are point-based rather than population-based; that is, the data are specific to a particular point only (e.g. Near Neighbor analysis). Other data are population-based, such as Delaunay Tessellation.

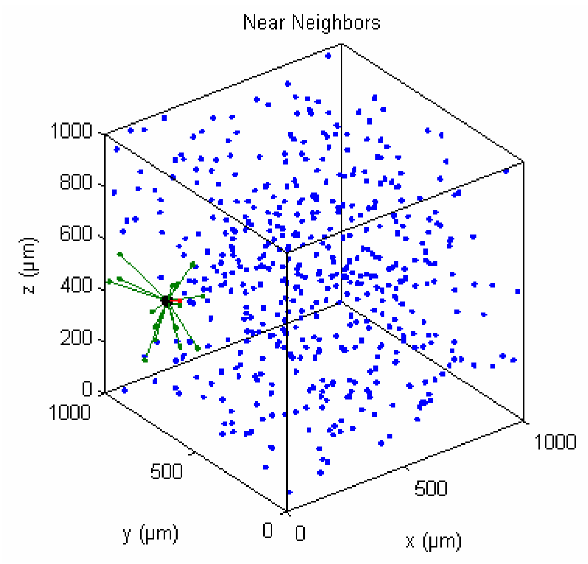
Near Neighbor

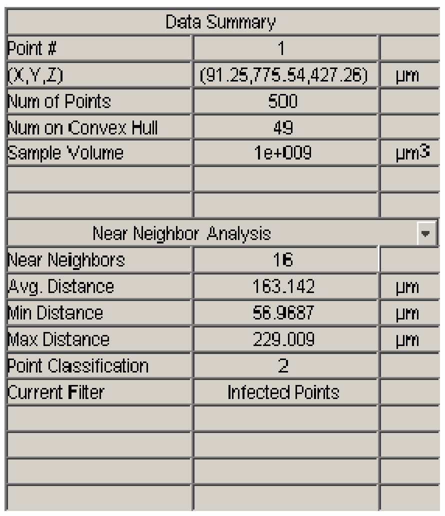
The **Near Neighbor** analysis uses the Delaunay tessellation to find all near neighbors to every data point. All points will have at least one near neighbor (i.e. those near the boundary may have as few as a single near neighbor). One of the near neighbors will be the closest to the point of interest; this point is referred to as the nearest neighbor.



Choose the **Near Neighbor** menu item from the Analysis Menu, or use the Ctrl+N keyboard shortcut.

When the **Near Neighbor** menu is selected, the program performs the Delaunay/Near Neighbor analysis if it has not been performed before. Then, the near neighbor diagram is plotted in the left subplot. In this diagram, the current point is plotted as a black dot while all near neighbors are plotted as green dots with green lines connecting them to the point of interest. The nearest neighbor is plotted in red with a red line connecting it. Finally, the data table is made visible to the right of the subplot. Here, the data summary is shown along with the near neighbor statistics for the current point. Try using the arrow keys to scroll through the points in the plot. The up and down arrows will scroll by one point while the left and right arrows will scroll by ten points at a time.





Data Summary		
Point #	1	
(X,Y,Z)	(91.25,775.54,427.26)	µm
Num of Points	500	
Num on Convex Hull	49	
Sample Volume	1e+009	µm3
Near Neighbor Analysis		
Near Neighbors	16	
Avg. Distance	163.142	µm
Min Distance	56.9687	µm
Max Distance	229.009	µm
Point Classification	2	
Current Filter	Infected Points	

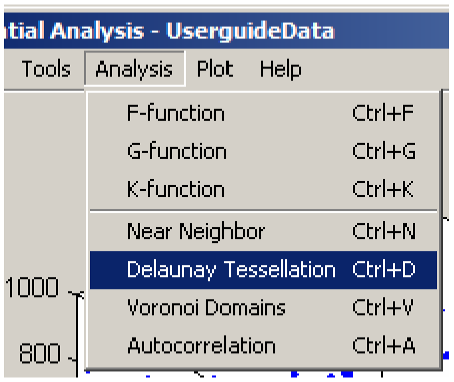
The second to last entry in the data table is the **Point Classification**. The **Point Classification** for the **Near Neighbor** analysis represents the type of data point that is being rendered in the plot. The type of point can be one of the following:

"1"	Normal Point
"2"	Double Infected Point
"3"	Infected Point
"4"	Point on the Convex Hull

A point is flagged if it does not meet the filter characteristics set by the user under the Options menu (see Options section for more details). When a point is flagged, its near neighbor diagram is not rendered in the left subplot and its statistics are not used to generate the histogram plots (see Plot Menu section). Its statistics are available in the data table, however, to the right. For the **Near Neighbor** analysis, the default filter type is **Infected Points**.

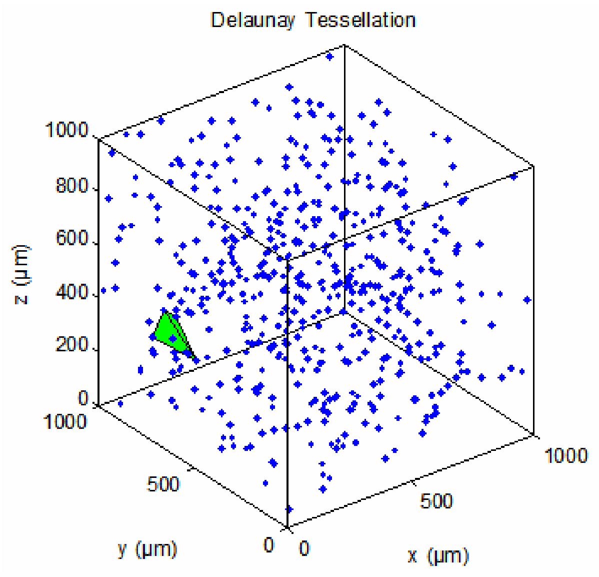
Delaunay Tessellation

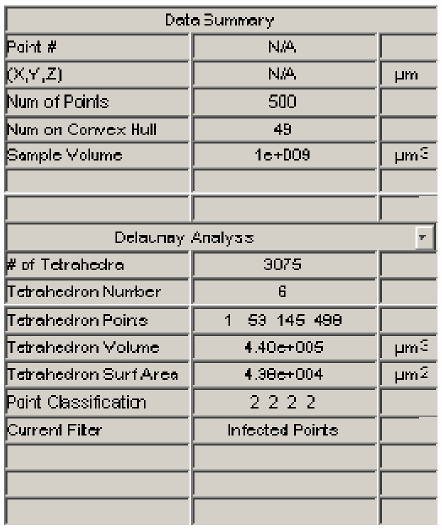
The **Delaunay Tessellation** analysis is based upon an identical analysis to that described above in the **Near Neighbor** analysis, but the derived statistics and graphics plotted are different. In three dimensions, a Delaunay tessellation is a collection of tetrahedra. When the **Delaunay Tessellation** menu is selected, the program renders the Delaunay tessellation by rendering each tetrahedron in the left subplot.



Choose the **Delaunay Tessellation** menu item from the Analysis Menu, or use the Ctrl+D keyboard shortcut.

Like in the **Near Neighbor** analysis, the arrow keys can be used to scroll between the individual tetrahedra. The tetrahedron number that is currently being rendered is shown in the data table as well as the total number of tetrahedra. As before, the up and down arrows scroll through one tetrahedron at a time while the left and right arrows scroll through ten tetrahedra at a time. You can view all the tetrahedra in succession by choosing the **View All Delaunay Tetrahedra** under the **Tools Menu.**



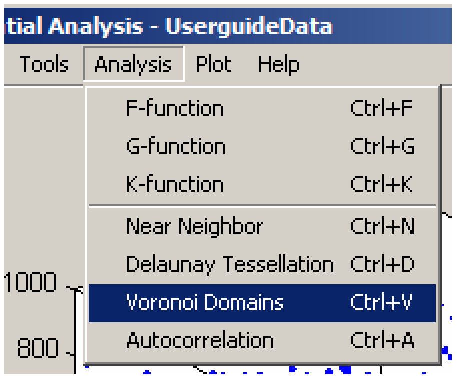


Data Summary		
Point #	N/A	
(X,Y,Z)	N/A	µm
Num of Points	500	
Num on Convex Hull	49	
Sample Volume	1e+009	µm3
Delaunay Analysis		
# of Tetrahedra	3075	
Tetrahedron Number	6	
Tetrahedron Points	1 53 145 498	
Tetrahedron Volume	4.40e+005	µm3
Tetrahedron Surf Area	4.38e+004	µm2
Point Classification	2 2 2 2	
Current Filter	Infected Points	

The Delaunay tetrahedra are sorted by the four point numbers that make up each tetrahedron. These points are always in ascending order and then sorted by the first point number. The tetrahedra are filtered in a similar manner to the near neighbor points above. In each case, there are four points that will have a classification. If any one of these points is flagged, then the tetrahedron itself is flagged and is not rendered. As before, the statistics for tetrahedra that have been flagged are not used in generating histograms, but the data are still available in the data table.

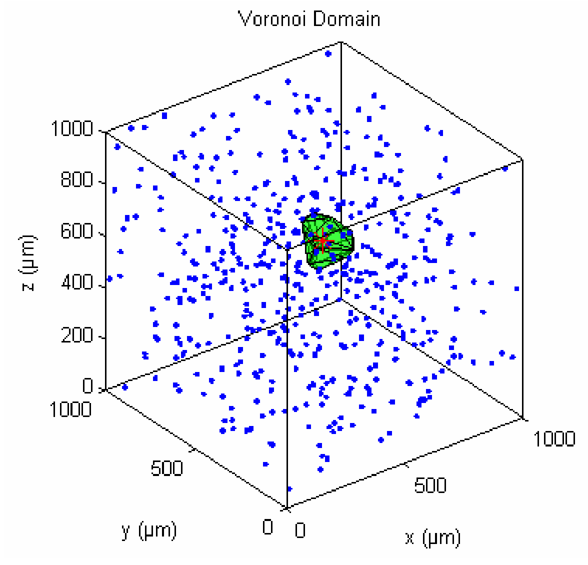
Voronoi Domains

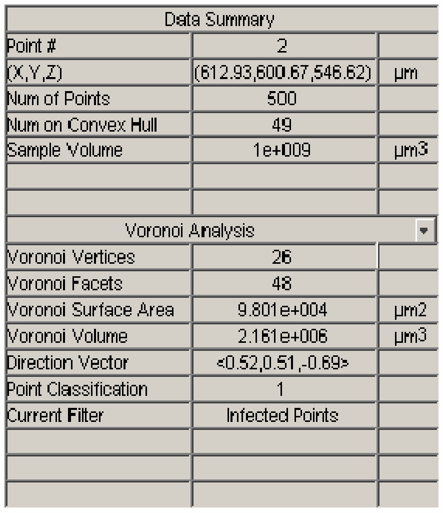
The **Voronoi Domains** menu uses the Voronoi routine in the Qhull package to determine the vertices that make up the Voronoi domains. During the calculation, the program also determines the Voronoi domain volumes and surface areas.



Choose the **Voronoi Domains** menu item from the Analysis Menu, or use the Ctrl+V keyboard shortcut.

When the **Voronoi Domains** menu is selected, the program performs the Voronoi analysis if it had not been performed before. The Voronoi domain associated with a given point is then plotted in the left subplot. In this diagram, the current point is plotted as a red asterisk (*) and the facets as green and semi-transparent. Again, the level of transparency is set in the **Options** menu. Finally, the data table is made visible to the right of the plot. Here, the data summary is shown along with the Voronoi analysis statistics for the current point. Try using the arrow keys to scroll through the points in the plot. The up and down arrows will scroll by one point while the left and right arrows will scroll by ten points at a time.





Data Summary		
Point #	2	
(X,Y,Z)	(612.93,600.67,546.62)	µm
Num of Points	500	
Num on Convex Hull	49	
Sample Volume	1e+009	µm3
Voronoi Analysis		
Voronoi Vertices	26	
Voronoi Facets	48	
Voronoi Surface Area	9.801e+004	µm2
Voronoi Volume	2.161e+006	µm3
Direction Vector	<0.52,0.51,-0.69>	
Point Classification	1	
Current Filter	Infected Points	

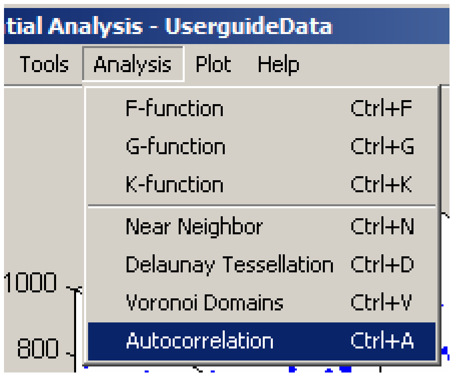
The second to last entry in the data table is the Voronoi Classification. The Voronoi Classification represents the type of data point that is being rendered in the plot. The type of point can be one of the following:

"1"	Normal Point
"2"	Double Infected Point
"3"	Infected Point
"4'	Point on the Convex Hull
"5"	Other

A point is flagged if it does not meet the filter characteristics set by the user under the Options menu. When a point is flagged, its Voronoi domain is not rendered in the left subplot and its statistics are not used to generate the histogram plots (see Plot section). Its statistics are available in the data table, however. For the Voronoi analysis, the default filter type is Infected Points. Point Classification 5, "other", is reserved for points where the Qhull routine failed for some reason. The reason for failure can be viewed in the message window when that point is selected. An example of one such error is, "qhull precision error: initial facet 2 is coplanar with the interior point".

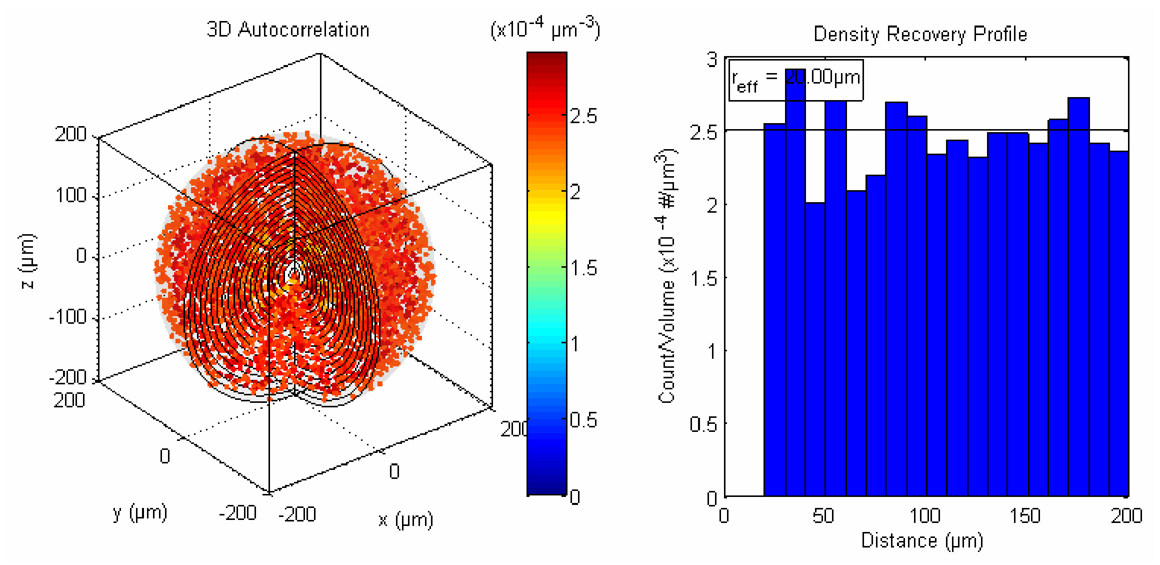
Autocorrelation

The **Autocorrelation** menu will render the autocorrelogram and calculate the associated statistics for the given data set.



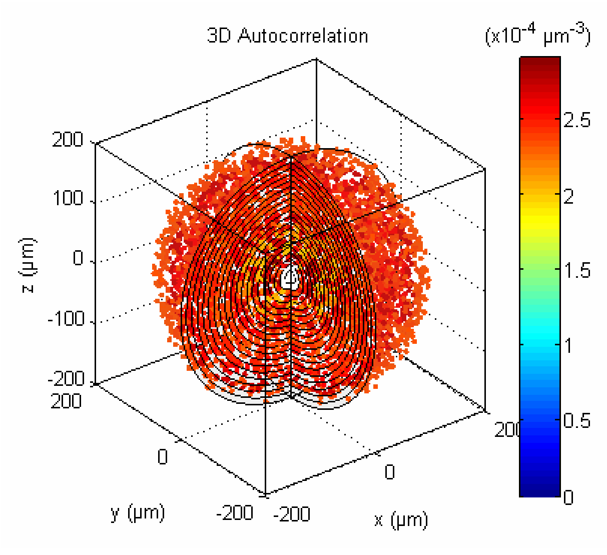
Choose the **Autocorrelation** menu item from the Analysis Menu, or use the Ctrl+A keyboard shortcut.

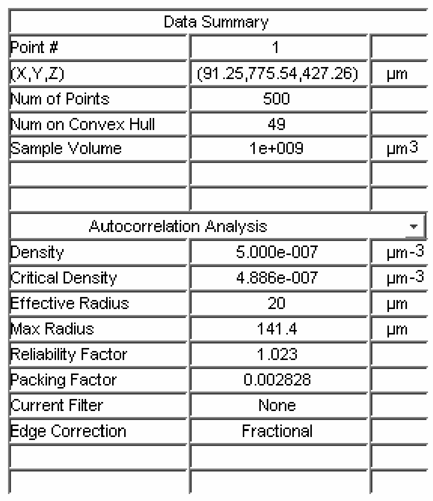
When the **Autocorrelation** menu is selected, the program performs the autocorrelation analysis if it had not been performed before. While the program is performing the autocorrelation analysis, the plot is generated. If the analysis has already been performed, selecting the **Autocorrelation** menu simply makes the plot visible. The autocorrelogram is shown on the left, being a plot of the position of all points in the sample field relative to every other point. It is portrayed as a series of incremental spheres. In the plot to the right, a density histogram for the autocorrelogram is shown. This is also known as the density recovery profile (DRP). The DRP is a graphical portrayal of the change in density as a function of eccentricity from the origin of the autocorrelogram. Both the autocorrelogram and the DRP reveal here a region near the origin in which density is lower than at greater eccentricities, a dead space, or “exclusion zone”, indicating that the probability of finding another cell of like-type is lower than at greater distances from every cell. An estimate of the size of this dead space is the “effective radius” (See Background section for more detail), being the radius of a sphere of equivalent integrated volume to the dead space relative to the average density at greater distances (indicated by the horizontal line in the DRP). The effective radius is indicated by the vertical line, and is indicated in the box. (NOTE: The vertical line in the example below falls directly between two bars and can not therefore been seen in this particular example.)



The number of bins as well as the bin widths are set in the **Options Menu**. To see these bins and how the points are arranged within them, the (-x,-y,-z) and (-x,-y,z) octants have been removed from the plot. Furthermore, the points are colored by their density within the bin.

By default the DRP is shown in the right subplot after the autocorrelation analysis. The only time this is not true is when the autocorrelation statistics are chosen directly from the data table. To view the statistics for the autocorrelation analysis in the data table, simply choose **Tools|View Data Table** from the menu bar. Once the data table is visible, you can view the statistics for the autocorrelation. Provided in the data table are the density, critical density, effective radius for the DRP, max radius, reliability factor, packing factor, the current point filter, and the edge correction algorithm. For more information on these statistics, see **Chapter 1: Spatial Correlation Analysis and its Derivatives.**



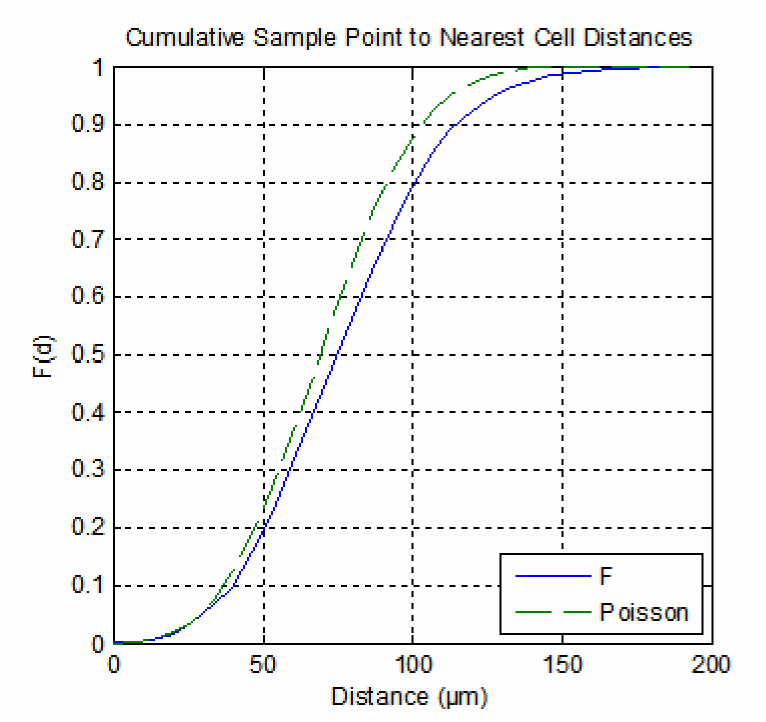


Data Summary		
Point #	1	
(X,Y,Z)	(91.25,775.54,427.26)	µm
Num of Points	500	
Num on Convex Hull	49	
Sample Volume	1e+009	µm3
Autocorrelation Analysis		
Density	5.000e-007	µm-3
Critical Density	4.886e-007	µm-3
Effective Radius	20	µm
Max Radius	141.4	µm
Reliability Factor	1.023	
Packing Factor	0.002828	
Current Filter	None	
Edge Correction	Fractional	

F-Function

The F-function measures the distance from each grid point to its nearest neighboring cell and plots the cumulative probability as a function of distance. The fidelity of the plot is determined by the number of grid points placed in the sample field and the number of points used in the calculation. Modifying these values is done through the **Options** menu (see Options). For more information on the F-function see Chapter 1: The F-function.

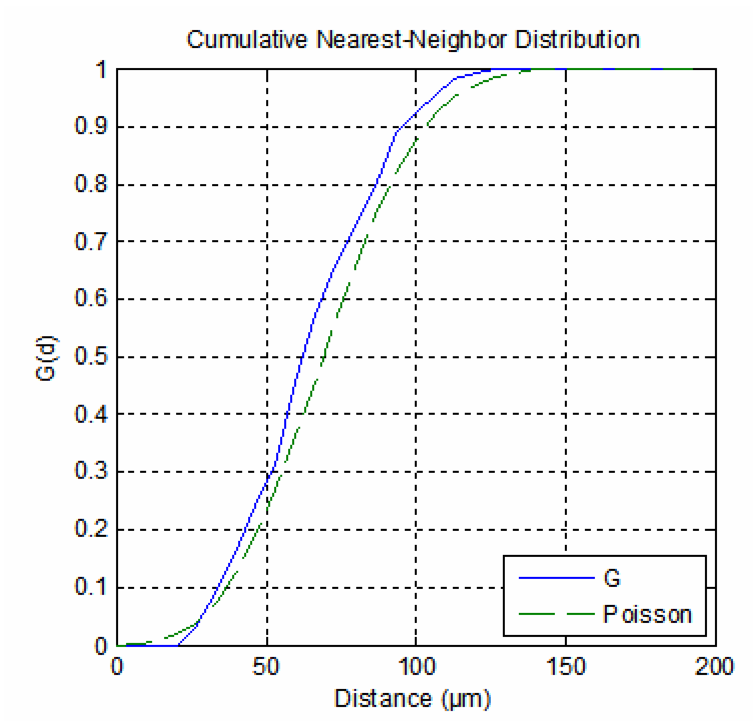
In the figure below, the plot of the F-function for the example dataset is compared to that of a Poisson distribution of cells.



G-Function

The G-function calculates the fraction of nearest-neighboring distances that are less than or equal to a given distance t. The fidelity of the plot is determined the number of points used in the calculation. Modifying number of points and the range is done through the **Options** menu (see Options). For more information on the G-function see Chapter 1: The G-function.

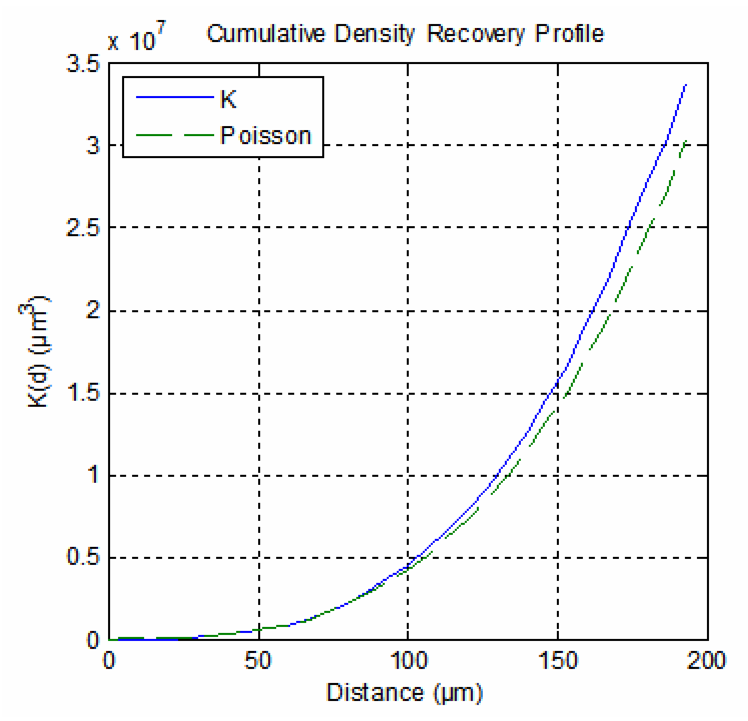
In the figure below, the plot of the G-function for the example dataset is compared to that of a Poisson distribution of cells.



K-Function

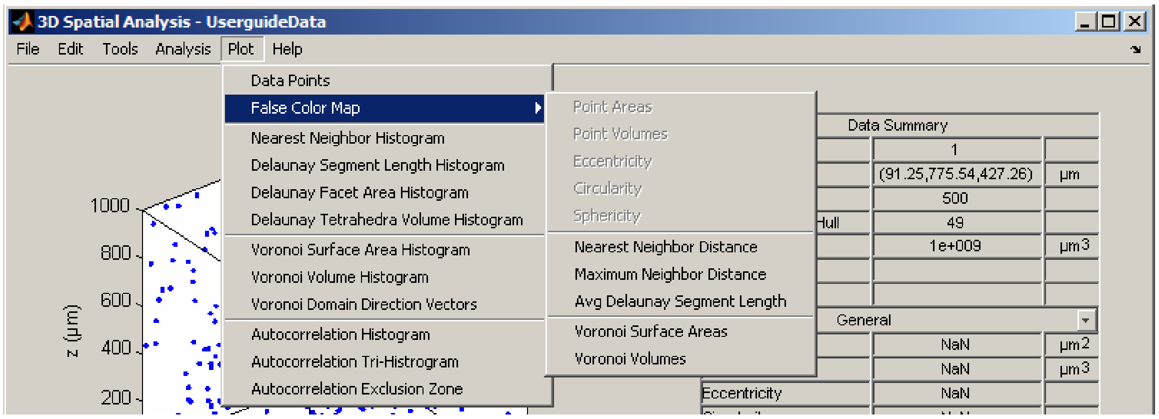
The K-function is the cumulative version of the density recovery profile (see Chapter 3: Autocorrelation). The fidelity of the plot is determined the number of points used in the calculation. Modifying number of points and the range is done through the **Options** menu (see Options). For more information on the K-function see Chapter 1: The K-function.

In the figure below, the plot of the K-function for the example dataset is compared to that of a Poisson distribution of cells.



Plot Menu

The **Plot Menu** is where you will find all the plotting commands for the program. Most of the plotting commands are unavailable until you run the corresponding analysis first (**Near Neighbor/Delaunay, Voronoi, Autocorrelation**). The default plot, being the only one that is always available, is the **Data Points** plot. This is simply the plot of all the data points, measured or simulated, and their spatial location in the left subplot.

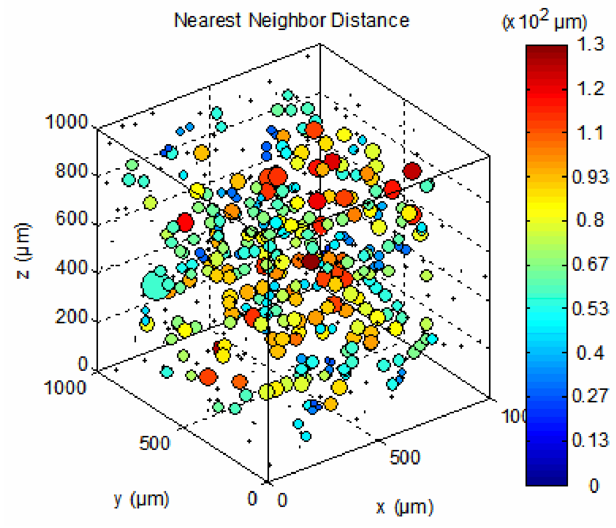

3D Spatial Analysis - UserguideData
File
Edit
Tools
Analysis
Plot
Help
Data Points
False Color Map
Nearest Neighbor Histogram
Delaunay Segment Length Histogram
Delaunay Facet Area Histogram
Delaunay Tetrahedra Volume Histogram
Voronoi Surface Area Histogram
Voronoi Volume Histogram
Voronoi Domain Direction Vectors
Autocorrelation Histogram
Autocorrelation Tri-Histrogram
Autocorrelation Exclusion Zone
Point Areas
Point Volumes
Eccentricity
Circularity
Sphericity
Nearest Neighbor Distance
Maximum Neighbor Distance
Avg Delaunay Segment Length
Voronoi Surface Areas
Voronoi Volumes
Data Summary
General
1000
800
600
400
200
z (µm)

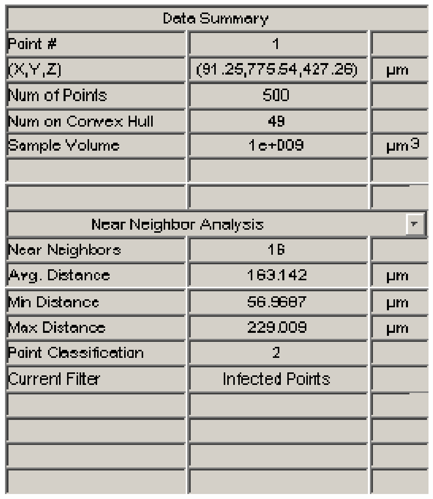
	1	
	(91.25,775.54,427.26)	µm
	500	
Hull	49	
	1e+009	µm3
	NaN	µm2
	NaN	µm3
Eccentricity	NaN	

False Color Maps

Under the **Plot Menu**, there is a submenu item called **False Color Map**. Inside this submenu, there are 5 plot commands available to all datasets, and 5 plot commands available to certain types of measured data. The 5 plot commands available to all data are: **Nearest Neighbor Distance, Maximum Neighbor Distance, Avg. Delaunay Segment Length, Voronoi Surface Areas, and Voronoi Volumes**. None of these commands is available, however, until the corresponding analysis has been performed. In a false color plot, the metric of interest (e.g. Nearest Neighbor Distance) for each point is rendered as a sphere at the location of the point. The size of the sphere as well as its color is in relation to the value of the chosen metric of that point. The colorbar to the right of the axis shows the mapping of the color to the data. For each dataset, the data is normalized to the colorbar. That is, the top most color will be mapped to the largest value in the dataset. Colors are therefore relative to each other, but not across datasets.

When a false color plot is chosen, the corresponding data table is made visible to the right of the colorbar. The first 3 commands will bring up the Near Neighbor Analysis, and the last 2 commands will bring up the Voronoi Analysis. The information in the data table will refer to the current point of interest. This point is rendered in the left subplot as a sphere with the corresponding color, but the size of the sphere is made large so that it can be easily recognized. As you scroll through points using the arrow keys, only the current point of interest is made large, while all others are returned to their appropriate size. Points that have been flagged by the current filter are plotted not as spheres, but as single points on the plot. If you decide you want to send a false color plot to a new window so that you can use it in a presentation (See Edit section), try restoring the current point to its original size by using the **Restore Marker Size** command under the **Tools menu**.



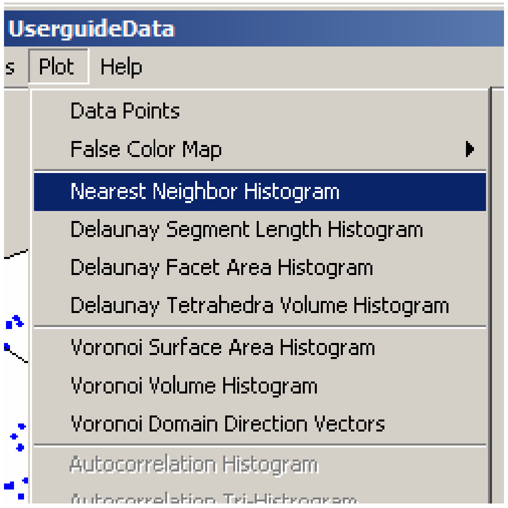


Data Summary		
Point #	1	
(X,Y,Z)	(91.25,775.54,427.26)	µm
Num of Points	500	
Num on Convex Hull	49	
Sample Volume	1e+009	µm3
Near Neighbor Analysis		
Near Neighbors	16	
Avg. Distance	163.142	µm
Min Distance	56.9687	µm
Max Distance	229.009	µm
Point Classification	2	
Current Filter	Infected Points	

The left subplot above contains an example of a false color plot showing the nearest neighbor distance for each data point. Note that the current point is #1 and that this point is shown as a large cyan sphere near the left edge of the plot. The distance to its nearest neighbor is almost 57 µm which according to the colorbar, makes it cyan.

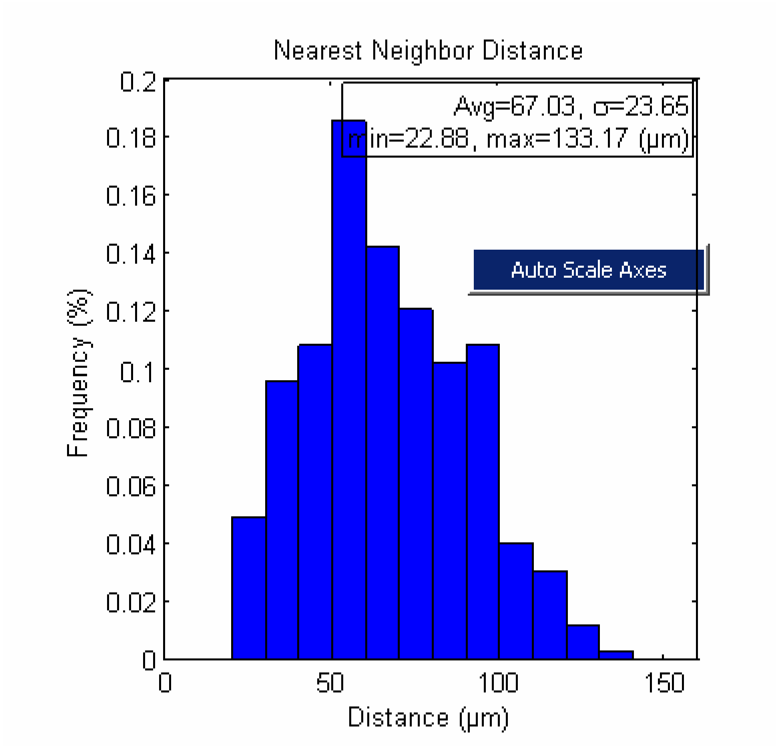
Nearest Neighbor Histogram

A common way to display aggregate statistics in Spatial Analysis 3D is with the use of histograms. The histogram plots take the data from the appropriate analysis and apply the current user-defined filter. The filtered data is then placed into bins defined under the **Options menu.**


UserguideData
s
Plot
Help
Data Points
False Color Map
Nearest Neighbor Histogram
Delaunay Segment Length Histogram
Delaunay Facet Area Histogram
Delaunay Tetrahedra Volume Histogram
Voronoi Surface Area Histogram
Voronoi Volume Histogram
Voronoi Domain Direction Vectors
Autocorrelation Histogram
Autocorrelation Tri-Histogram

To plot the histogram for the nearest neighbor data, choose **Nearest Neighbor Histogram** from the Plot menu.

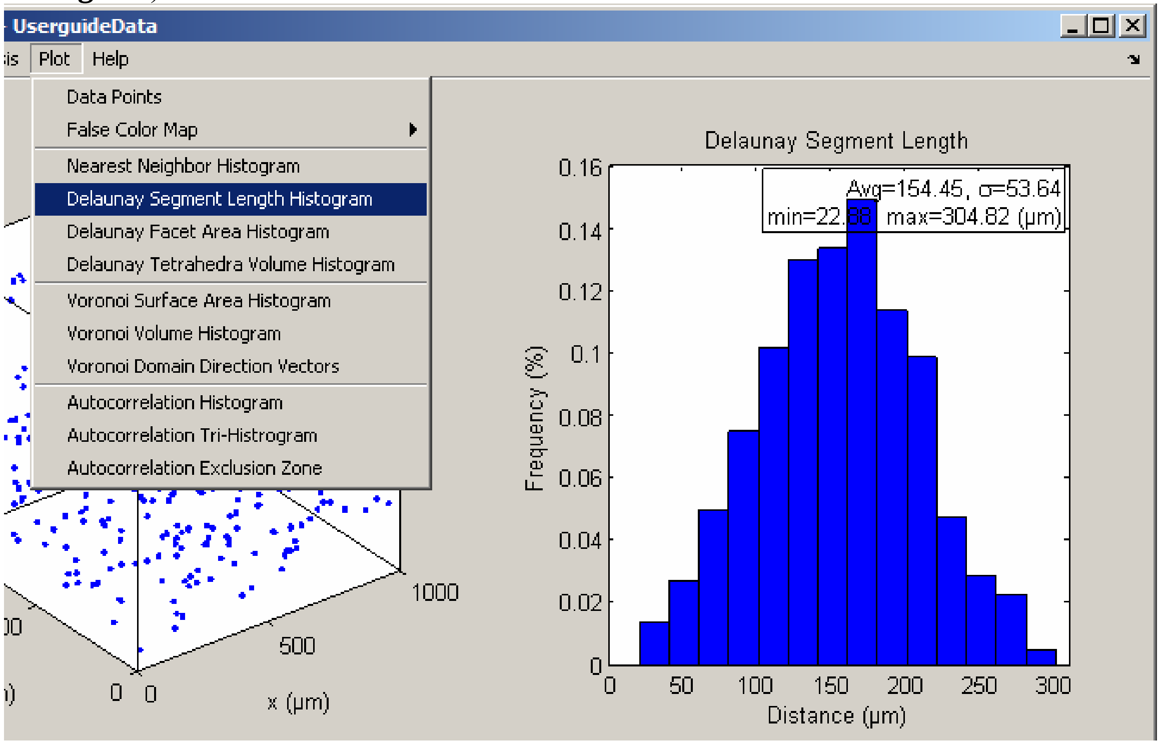
When the nearest neighbor histogram plot is generated, the nearest neighbor data from each point that has not been filtered is used to generate the histogram below. The bin widths, and number of bins used in this example are the default values. The histogram plot is then displayed in the right subplot while the data points are displayed in the left subplot.



There are many circumstances where it will be difficult to determine the bin widths and/or number of bins to use ahead of time. In this case, you can automatically scale the histogram plot by right clicking anywhere on the plot and choosing **Auto Scale Axis**.

Delaunay Segment Length Histogram

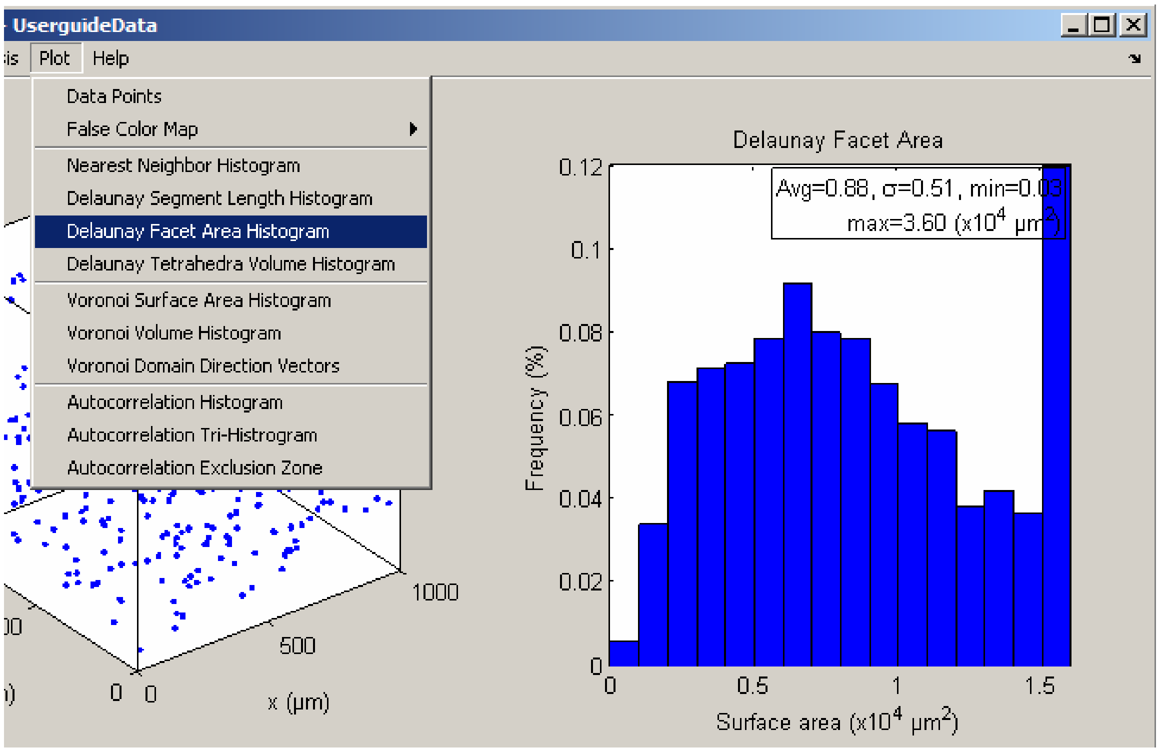
When the Delaunay analysis is performed, the Qhull routine generates a Delaunay tessellation. This tessellation represents a unique set of tetrahedra such that no data points are contained in any circumspheres of the tetrahedra. Each tetrahedron is comprised of 6 segments that connect the 4 data points together, with several tetrahedra sharing a segment. The unique set of these segments (being identical to the unique set of near neighbor distances) that define all the tetrahedra within the sample field are known as Delaunay segments. The lengths of this unique set of Delaunay segments can be summarized into a histogram. Like the nearest neighbor data, the Delaunay segment lengths are filtered by the NN/Delaunay filter defined in the **Options menu**. You can also auto scale the axis by right clicking inside the right subplot (see **Nearest Neighbor Histogram**).


UserguideData
is
Plot
Help
Data Points
False Color Map
Nearest Neighbor Histogram
Delaunay Segment Length Histogram
Delaunay Facet Area Histogram
Delaunay Tetrahedra Volume Histogram
Voronoi Surface Area Histogram
Voronoi Volume Histogram
Voronoi Domain Direction Vectors
Autocorrelation Histogram
Autocorrelation Tri-Histrogram
Autocorrelation Exclusion Zone
1000
500
0
0
x (μm)
Delaunay Segment Length
Avg=154.45, σ=53.64
min=22.88 max=304.82 (μm)
Frequency (%)
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
0
50
100
150
200
250
300
Distance (μm)

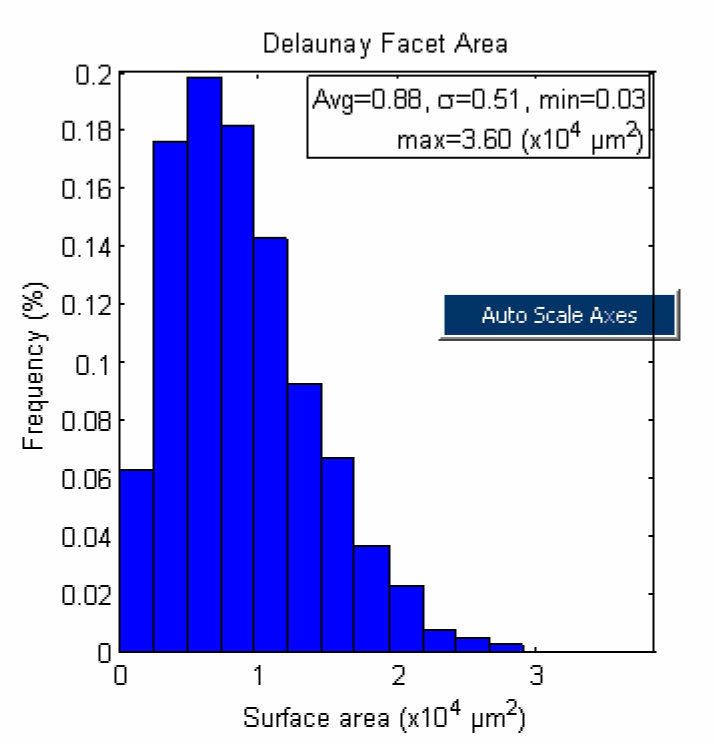
Delaunay Facet Area Histogram

Each Delaunay tetrahedron from the NN/Delaunay analysis is comprised of 4 facets. Most tetrahedra will share each facet with another tetrahedron. Therefore, there exists a unique set of Delaunay facets that can be used to define all the tetrahedra as well, and these too can be summarized in histogram form. Like the nearest neighbor data, the Delaunay facet areas are filtered by the NN/Delaunay filter defined in the Options menu.

On the following page there is a picture of the Delaunay Facet Area Histogram with the default number of bins and the default bin size from the Options menu. As you can see, the data is skewed to the right. That is, a significant portion of the data falls into the last bin, which in this case is anything $>1.5\times10^4$ μm^2. If we look at the statistics contained within the text box inside the histogram axis, we see that the maximum area is actually 3.6×10^4 μm^2. You can either adjust the number of bins or the bin size inside the Options menu to better distribute the data, or you can also auto scale the histogram by right clicking inside the right subplot (see Nearest Neighbor Histogram). The bottom picture on the following page shows how the histogram would look if you use this auto scale options. Note that the statistics in the text box have not changed—this is to be expected since our dataset has not changed, only the plot.


UserguideData
is
Plot
Help
Data Points
False Color Map
Nearest Neighbor Histogram
Delaunay Segment Length Histogram
Delaunay Facet Area Histogram
Delaunay Tetrahedra Volume Histogram
Voronoi Surface Area Histogram
Voronoi Volume Histogram
Voronoi Domain Direction Vectors
Autocorrelation Histogram
Autocorrelation Tri-Histrogram
Autocorrelation Exclusion Zone
Delaunay Facet Area
Avg=0.88, σ=0.51, min=0.03
max=3.60 (x10^4 μm^2)
Frequency (%)
Surface area (x10^4 μm^2)
x (μm)
1000
500
0
0.12
0.1
0.08
0.06
0.04
0.02
0.5
1
1.5

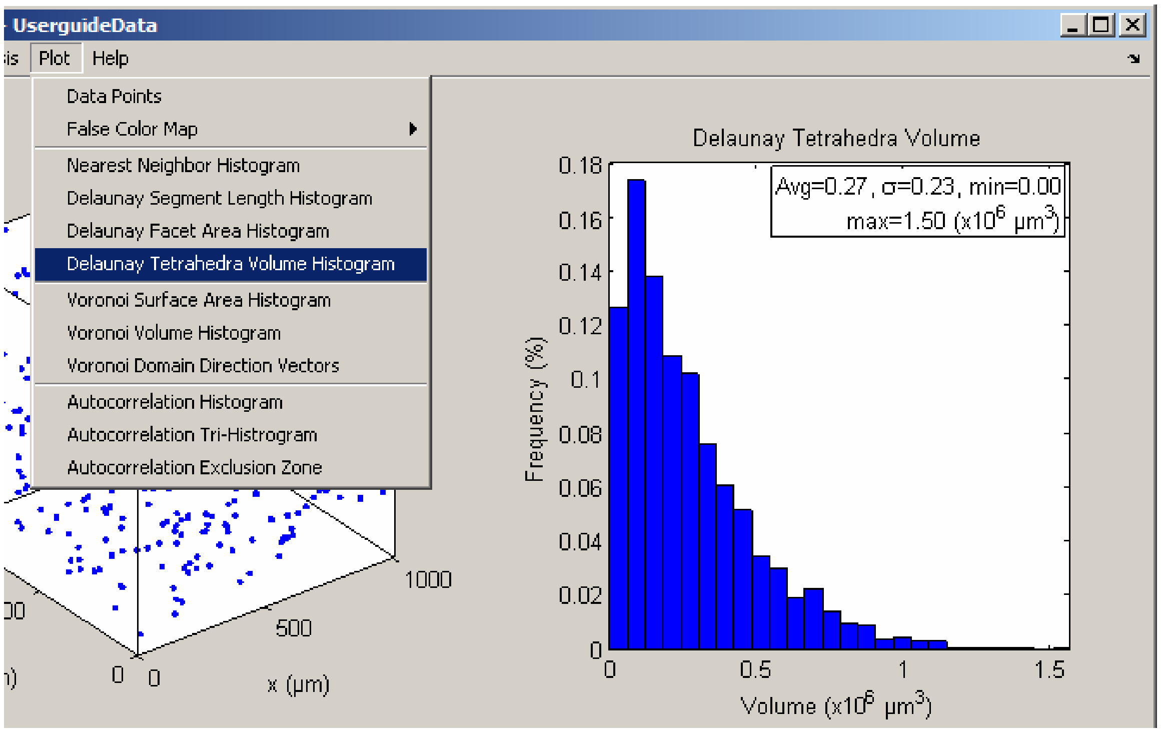
Plot of the Delaunay Facet Areas after using the auto scale feature:



Delaunay Tetrahedra Volume Histogram

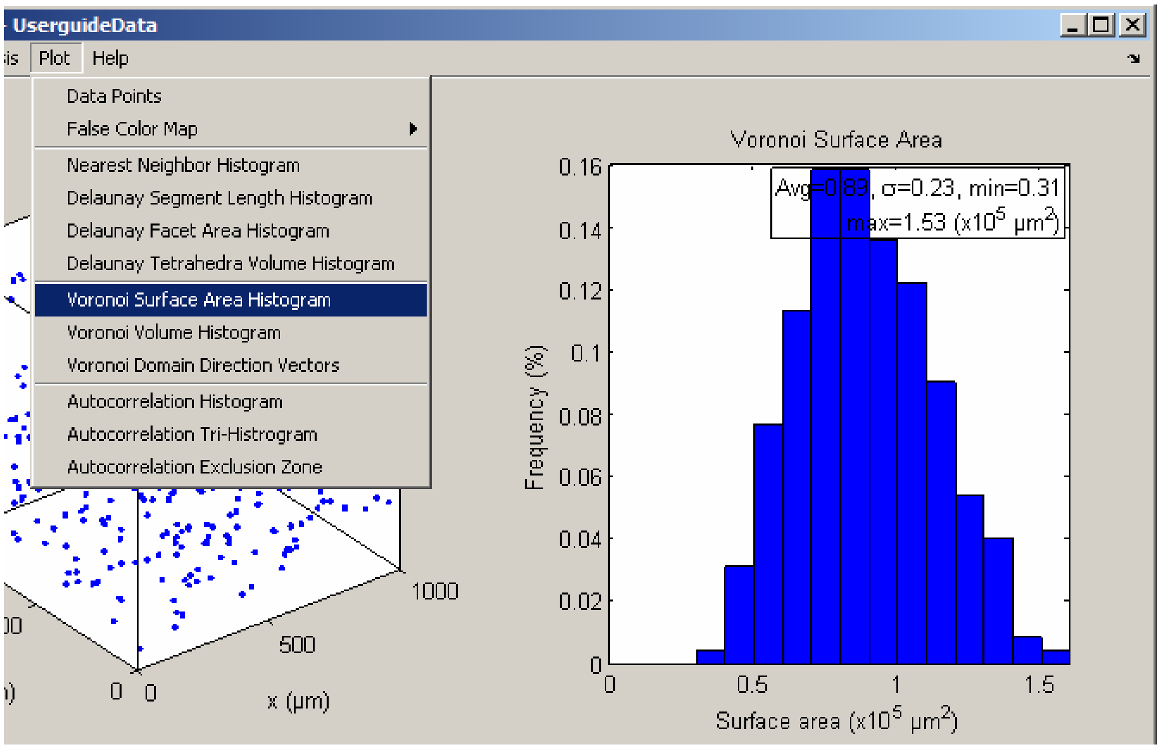
The NN/Delaunay analysis utilizes the Qhull routine to generate a Delaunay tessellation. This tessellation is a unique set of tetrahedra created from the data points such that no points are contained within any circumspheres of the tetrahedra. The Delaunay tetrahedra are 4-sided polyhedrons from which one can calculate the volume. Like the nearest neighbor data, the Delaunay volumes are filtered by the NN/Delaunay filter defined in the Options menu.

Below is a picture of the Delaunay Volume Histogram. In this example, the number of Delaunay tetrahedron volume bins has been set to 25 in the Options menu and the axes were auto scaled using the right mouse button (see Nearest Neighbor Histogram). The result of these operations is a histogram that starts at zero and terminates at the largest value in the dataset, in this case 1.5×10^6 μm^3. The shape of the histogram in this example is quite different than the previous ones, i.e. Nearest Neighbor Histogram, etc. Because the data points in this example were generated to approximate a random distribution, the histogram is skewed with a long tail extending to larger volumes.


UserguideData
is
Plot
Help
Data Points
False Color Map
Nearest Neighbor Histogram
Delaunay Segment Length Histogram
Delaunay Facet Area Histogram
Delaunay Tetrahedra Volume Histogram
Voronoi Surface Area Histogram
Voronoi Volume Histogram
Voronoi Domain Direction Vectors
Autocorrelation Histogram
Autocorrelation Tri-Histrogram
Autocorrelation Exclusion Zone
1000
500
0
0
x (μm)
Delaunay Tetrahedra Volume
Avg=0.27, σ=0.23, min=0.00
max=1.50 (x10^6 μm^3)
Frequency (%)
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
0
0.5
1
1.5
Volume (x10^6 μm^3)

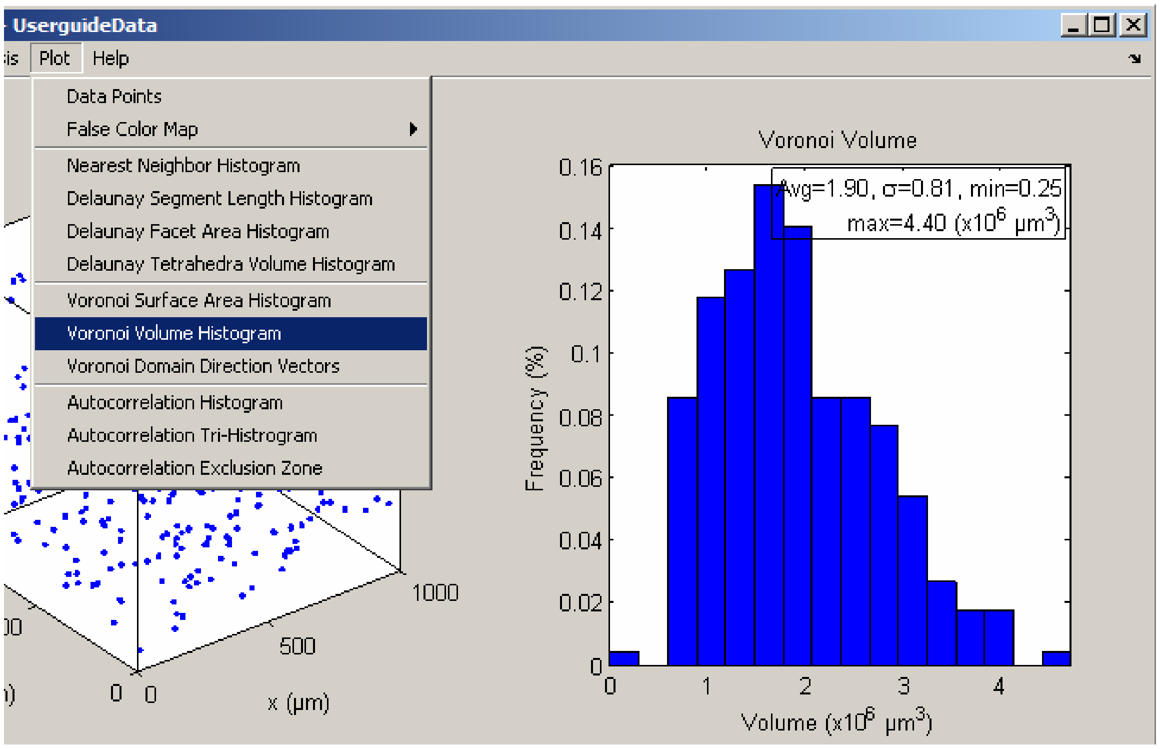
Voronoi Surface Area Histogram

When the Voronoi Domain analysis is performed, Spatial Analysis 3D utilizes the Qhull routine to generate the Voronoi cells associated with the current data set. After the analysis is performed, the resulting data can be summarized and plotted as histograms. In the figure below, the surface area of each Voronoi domain is calculated, the current user-defined filter is applied, and then the filtered data is placed into bins defined under the Options menu. In this example, the number of bins and the bin size used were the default values of 15 and 1x10^4 µm^2 respectively. As with any of the histogram plots, you can auto scale the axis by right clicking inside the subplot and choosing **Auto Scale Axis**. This will not change the value for the bin width inside the Options menu, however.


UserguideData
is
Plot
Help
Data Points
False Color Map
Nearest Neighbor Histogram
Delaunay Segment Length Histogram
Delaunay Facet Area Histogram
Delaunay Tetrahedra Volume Histogram
Voronoi Surface Area Histogram
Voronoi Volume Histogram
Voronoi Domain Direction Vectors
Autocorrelation Histogram
Autocorrelation Tri-Histrogram
Autocorrelation Exclusion Zone
Voronoi Surface Area
Avg=0.89, σ=0.23, min=0.31
max=1.53 (x10^5 µm^2)
Frequency (%)
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
0
0.5
1
1.5
Surface area (x10^5 µm^2)
1000
500
00
0
0
x (µm)

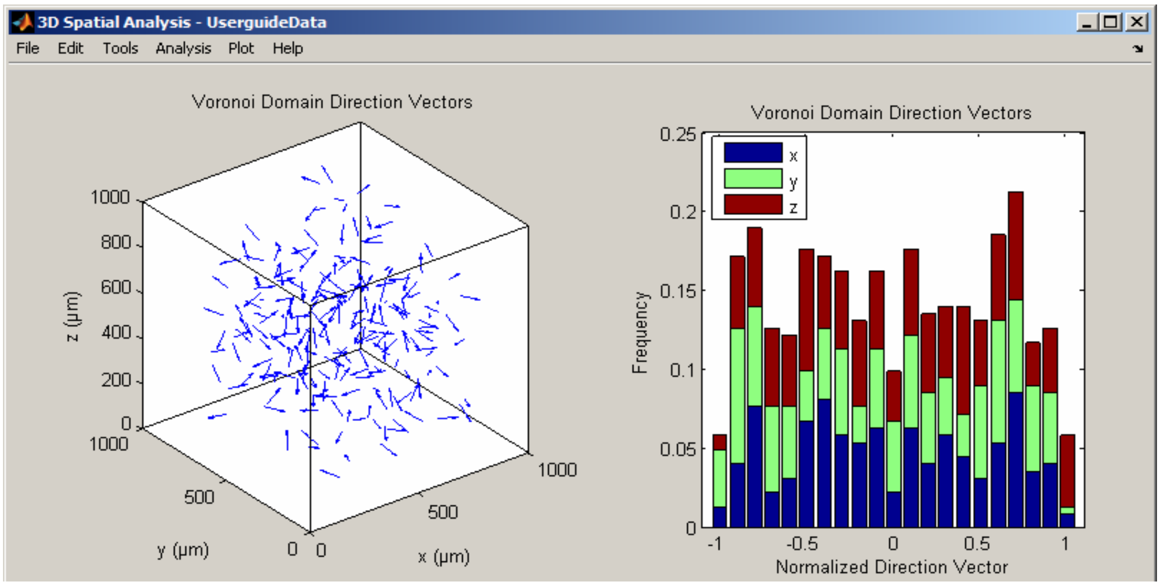
Voronoi Volume Histogram

After the Voronoi Domain analysis has been performed, the data can be summarized and plotted in histogram form. Below is the plot of the filtered Voronoi domain volumes. In this example, the axes were auto scaled by using the right mouse button and choosing "Auto Scale Axis". The default filter "Infected Points" was also applied.

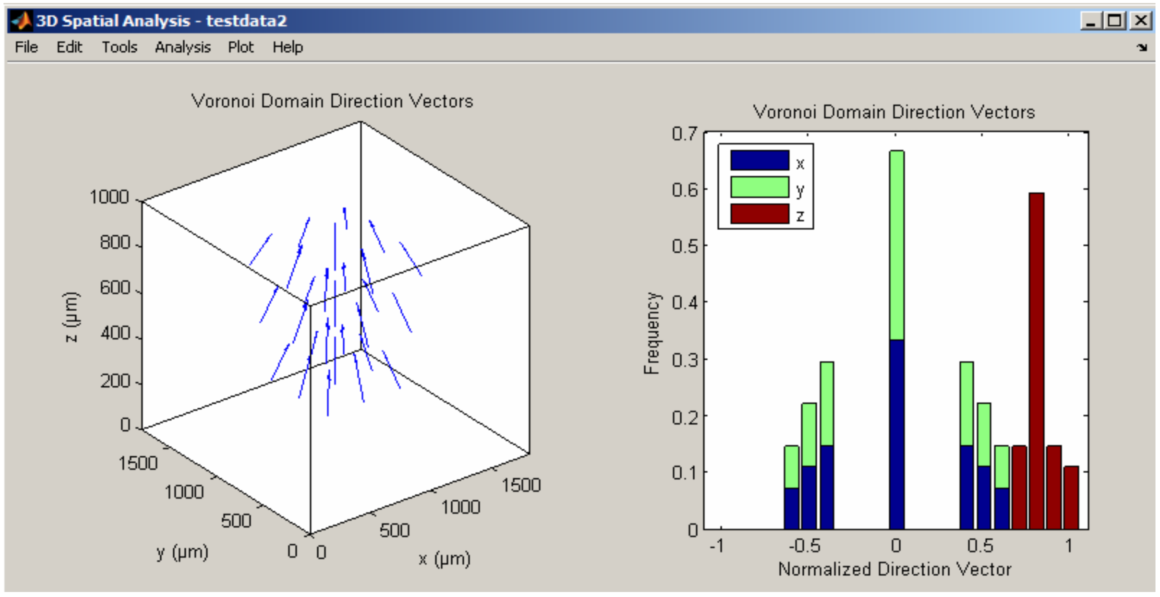

UserguideData
is
Plot
Help
Data Points
False Color Map
Nearest Neighbor Histogram
Delaunay Segment Length Histogram
Delaunay Facet Area Histogram
Delaunay Tetrahedra Volume Histogram
Voronoi Surface Area Histogram
Voronoi Volume Histogram
Voronoi Domain Direction Vectors
Autocorrelation Histogram
Autocorrelation Tri-Histrogram
Autocorrelation Exclusion Zone
Voronoi Volume
Avg=1.90, σ=0.81, min=0.25
max=4.40 (x10^6 μm^3)
Frequency (%)
Volume (x10^6 μm^3)
x (μm)
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
0
1
2
3
4
500
1000

Voronoi Domain Direction Vectors

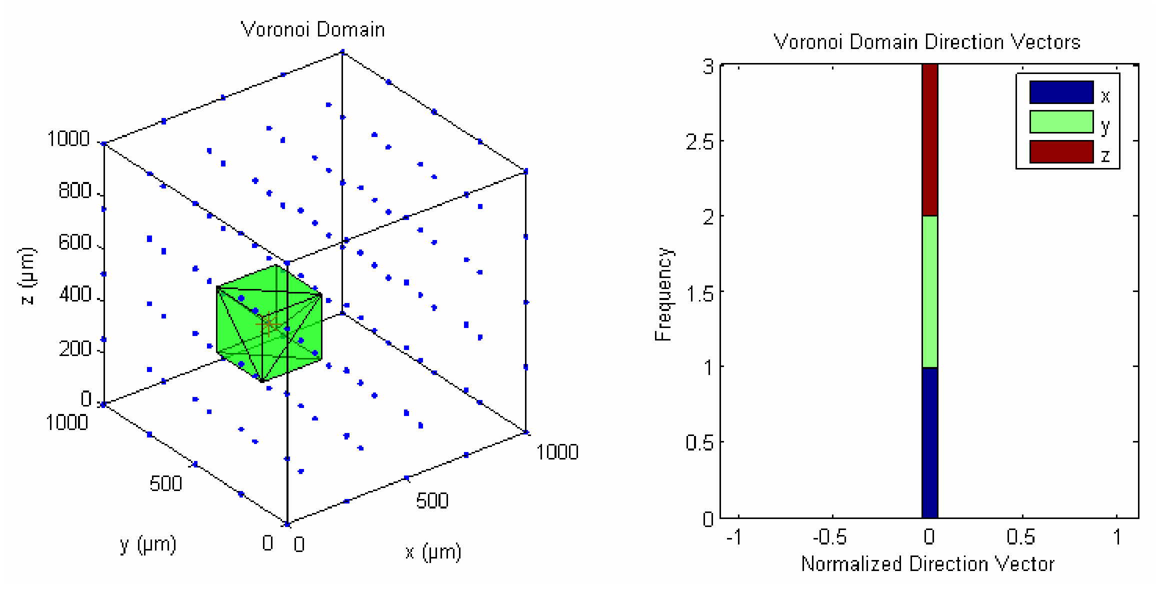
Voronoi domain direction vectors are used to describe the orientation of the Voronoi domains. The definition of the direction vectors along with the algorithms used to generate them are described in **Chapter 1: Voronoi Domain Direction Vectors.** When the **Voronoi Domain Direction Vector menu** is selected, the x, y, and z components of each Voronoi domain direction vector are parsed and a histogram describing these population components is created. In the case where the data points are randomly distributed, the resulting Voronoi domains will have fairly random shapes. This will result in summed direction vectors for each domain that include all different directions. The example on the following page shows a quiver plot of the direction vectors on the left and a histogram of the directions on the right. If we normalize the direction vector to length one or less, we can see that we have a fairly even distribution of all values from -1 to 1 in all directions.


3D Spatial Analysis - UserguideData
File Edit Tools Analysis Plot Help
Voronoi Domain Direction Vectors
z (µm)
y (µm)
x (µm)
Voronoi Domain Direction Vectors
Frequency
Normalized Direction Vector
x
y
z

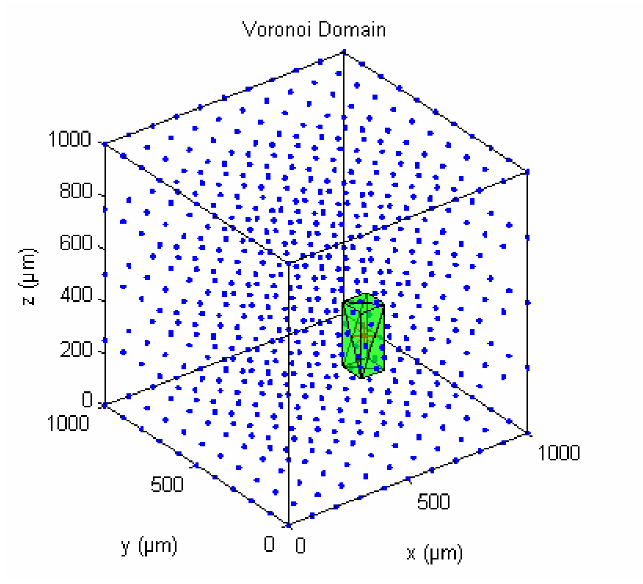
In the example above, we illustrated how a random set of data generates a random set of Voronoi domain direction vectors and how these vectors can be broken down into their x,y, and z components. The x,y, and z components have a uniform probability distribution between –1 and 1 as seen in the histogram plot. If we force the Voronoi domains to vary consistently in their shape, however, the effect will be evident in the direction vectors. Consider a population of points where the spacing between them grows with increasing z-direction. The Voronoi analysis of this population will result in direction vectors like those shown below. Here we see the vectors point in the positive z direction and toward the center of the population. This is also apparent in the histogram where the z-direction is always positive and has the largest values, and the x and y-directions are equal and dominated by the domains in the center with direction vectors of <0,0,1>.

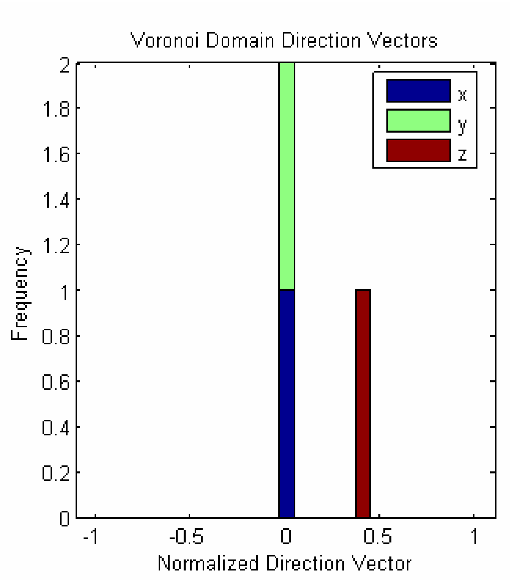

3D Spatial Analysis - testdata2
File Edit Tools Analysis Plot Help
Voronoi Domain Direction Vectors
z (µm)
y (µm)
x (µm)
Voronoi Domain Direction Vectors
Frequency
Normalized Direction Vector
x
y
z

In the background section, we described a situation where two direction vectors of opposing facets of a Voronoi domain may cancel each other out. In fact, if we have a perfectly symmetric Voronoi domain, each facet will have an opposing facet and the resulting direction vector will be <0,0,0>. If the Voronoi domain is three-fold symmetric, that is symmetric in the x,y, and z directions, then the second direction vector algorithm will also yield a <0,0,0> direction vector. In the example below, we have generated a square lattice with equal spacing in the x,y, and z directions. The result of the Voronoi tessellation is a set of square Voronoi domains. In the figure below, an example of one such Voronoi domain is shown along with the histogram of the Voronoi direction vectors. As expected, all the x,y, and z components of the direction vectors are zero.

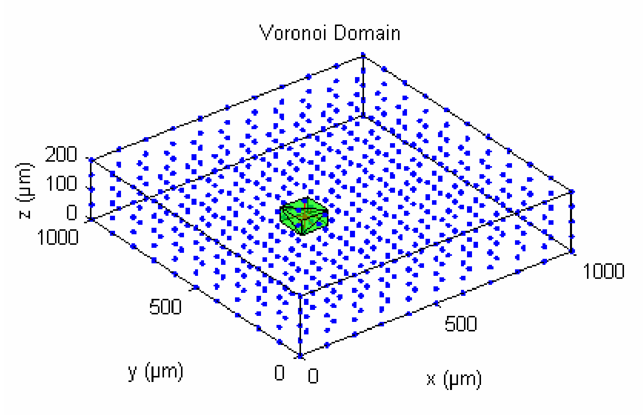


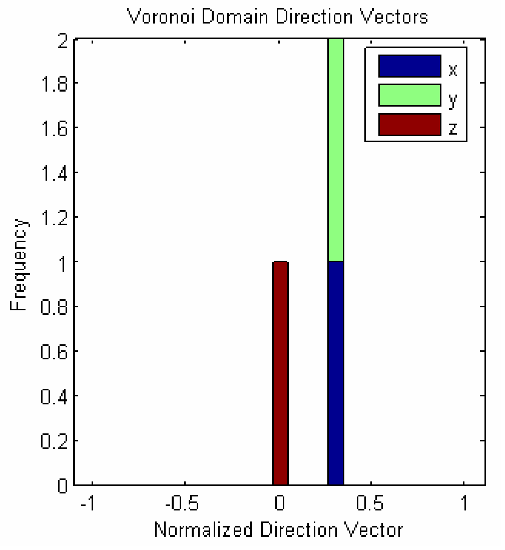
If we take same square lattice but now increase the density of the points in both the x and y directions by 5X and perform the Voronoi tessellation, we get Voronoi domains like those in the following figure. In this figure, the Voronoi domain that is depicted is elongated in the z direction due to the point spacing. If we calculate the initial Voronoi direction vector, each facet still has an opposing facet and the result will be <0,0,0>. We then apply the second algorithm to determine the relative contribution of each direction. The right figure shows the histogram due to that calculation. Here we see that the contribution of the x and y directions is zero and the contribution of the z direction is approximately 0.45. We interpret this by saying the direction vector points only in the z direction and that the Voronoi domain is elongated by approximately 45% in the z direction.





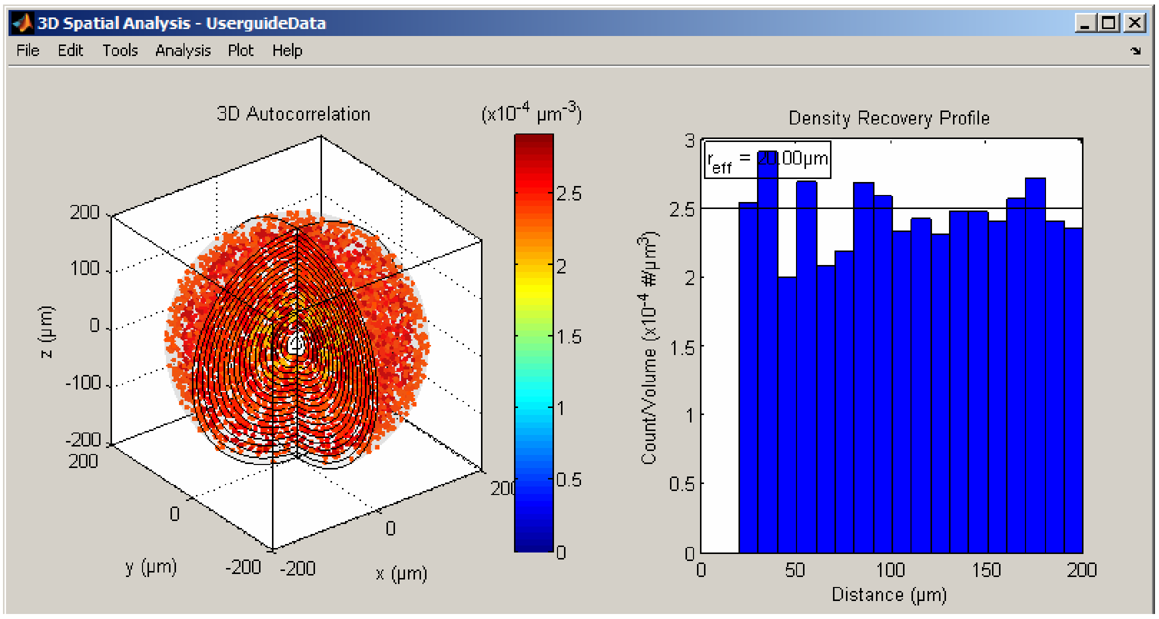
If we now take the original square lattice and increase the density of points in only the z direction by 5X, and perform the Voronoi tessellation, the resulting Voronoi domains will look like those on the left of the figure on the following page. Here we can see that the Voronoi domains are elongated equally in both the x and y directions looking like a short square box. Again, when we perform the initial Voronoi direction vector algorithm the result will be <0,0,0>. When the second algorithm is applied, we arrive at roughly the opposite result of the previous example. Here there is no z component to the direction vector, and the x and y components are equal around 0.33. The Voronoi domain direction vector would therefore point along the diagonal of the domain within the x-y plane. We can also interpret the histogram by saying the Voronoi domains are elongated equally in the x and y directions by approximately 33%. In short, the second algorithm converts the direction vector into an orientation vector for the sake of extracting this information from a population with symmetric Voronoi domains. In practice, the algorithm will rarely be employed in real biological datasets.





Autocorrelation Histogram

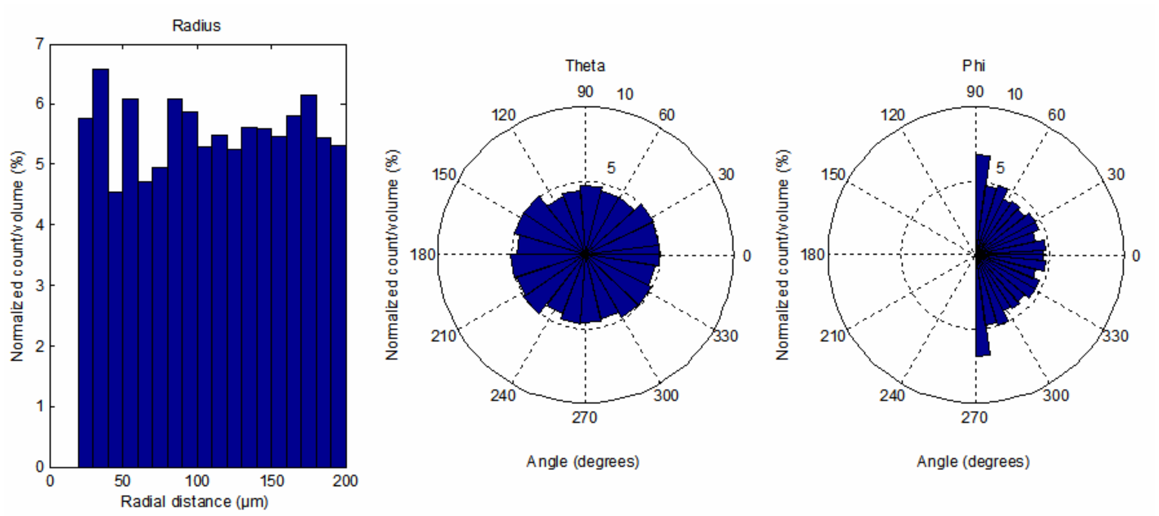
Earlier in this chapter we performed an autocorrelation on the data. This autocorrelation is simply a relative measure of every point's distance from all others. While generating the autocorrelation plot, cumulative statistics are gathered. In the most basic form, we can keep track of all the distances from the point of interest. When shown in histogram form, this diagram becomes the density recovery plot (DRP). To explicitly plot the DRP, choose **Autocorrelation Histogram** from the Plot menu. An example of the DRP is shown on the following page. With a random set of data, the DRP should flatten out toward a constant value as the distance from the point of interest increases. For example, we see that there are distances near zero where no points are found, out to about 20 µm. This region is commonly referred to as the dead space at the origin, or the "exclusion zone", which may be produced by the physical size of the cells themselves, or by some other minimal spacing rule (a d_{min} rule) that precludes their being positioned close to one another. In 3D, this dead space can be thought of as a volume. If we choose to define this volume as a sphere with the equivalent volume, we can calculate an effective radius for the dead space. The effective radius in this example happens to be exactly 20.0 µm, and it is shown in a text box in the upper left hand corner of the DRP.



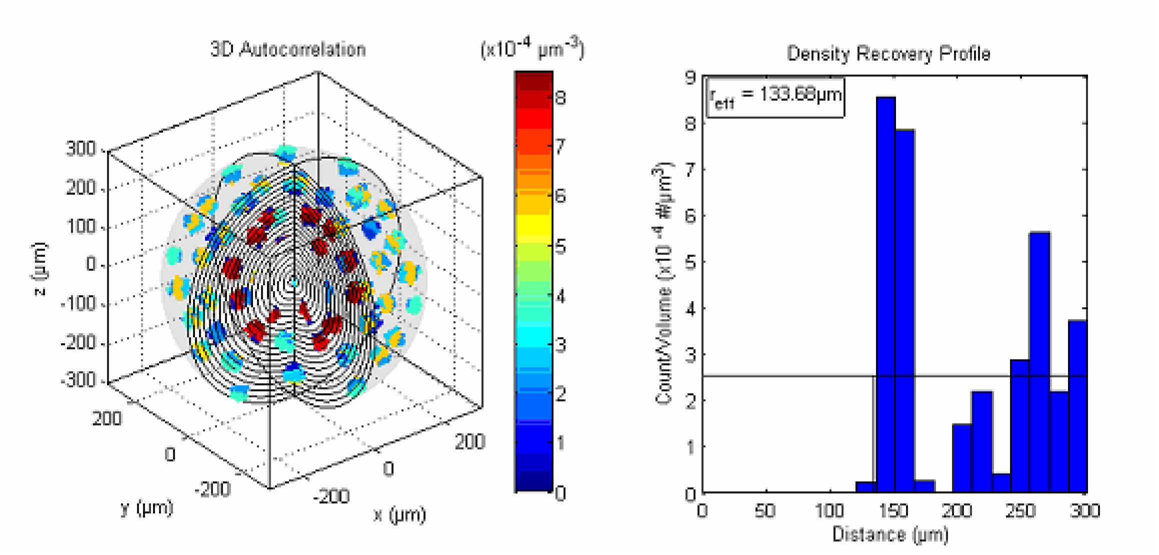
Autocorrelation Tri-Histogram

We discussed the DRP in the previous section, but perhaps a more thorough treatment of the autocorrelation data is the tri-histogram plot. During the autocorrelation calculation, each point is moved to the origin (0,0,0) and all other points are re-plotted with respect to this translation. When looking at the DRP, we are only concerned with the distance these points are from the origin, i.e. radius. There are two additional directions, however, that define the position of the points in such a plot relative to the origin, being azmuthal angle, theta, and elevation angle, phi. Together these three directions, radius, theta, and phi, make up the familiar spherical coordinate system.

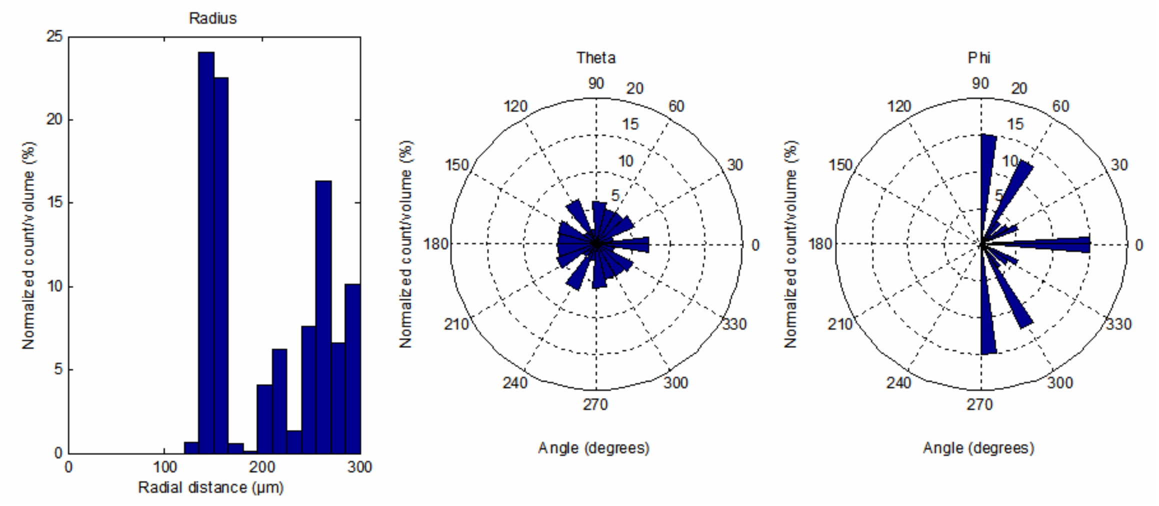
An example of a tri-histogram plot is shown on the following page. This example is the same as that shown in the Autocorrelation Histogram section, except now we are showing both theta and phi as well. Because the dataset used in this example is a random set using the d_{min} rule, the theta and phi data are evenly distributed, i.e. no preferred direction is present in the autocorrelogram. Theta has equal probability from 0 to 2π and phi has equal probability from $-\pi/2$ to $\pi/2$ (phi can only have values in this range). In the case of ordered data, hexagonally close packed for example, points lie not only certain distances away from one another, but also in particular directions. It is possible to use tri-histogram plots to recognize particular patterns embedded in data as a type of "finger printing".



Perhaps a more interesting example that illustrates the efficacy of the tri-histogram plots is an HCP lattice with jitter. In this example, the HCP lattice was generated to have a mean minimal distance of 150 μm and a standard deviation of 5 μm, with the smallest permissible distance of 2 μm. When the autocorrelation analysis is performed on this data, the following autocorrelogram is formed:



From the autocorrelogram one can see that there are local pockets of dense points. While the DRP indicates a number of preferred radial distances from the origin, it does not give us information regarding the preferred angular directions. Looking at the tri-histogram plot, one gets a better sense of the overall distribution of the autocorrelation points.

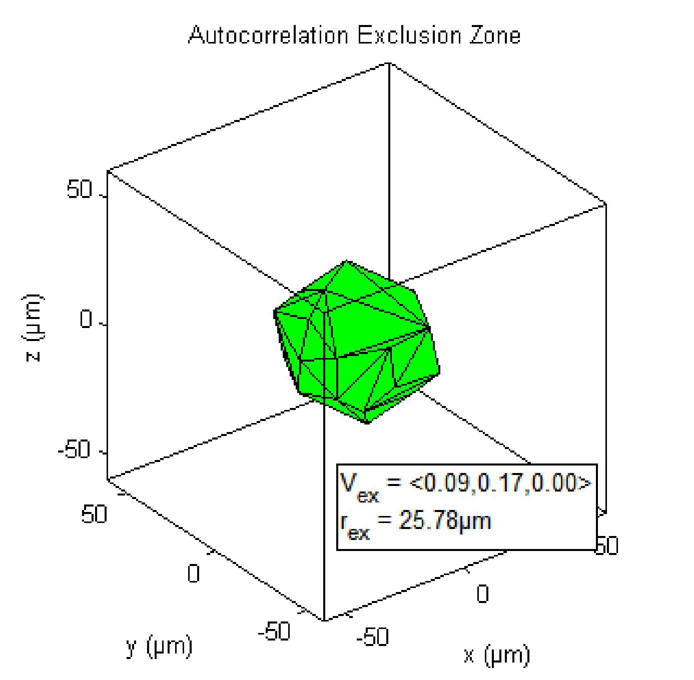


Unlike the tri-histogram plot for the randomly distributed dataset using the d_{min} rule, the histograms for the HCP dataset with jitter show definite preferred direction and distance. In fact, there are certain radial distances and angles that are not allowed under these conditions. With an extensive catalog of tri-histogram plots like those shown above, one might be able to extract a particular ordering or pattern in a dataset from the autocorrelation analysis alone.

Autocorrelation Exclusion Zone

When the autocorrelation diagram is created, there exists a set of near neighbors to the origin (0,0,0). These near neighbors can be found by performing a Delaunay tessellation on the autocorrelation dataset. After the Delaunay tessellation is performed, the near neighbors are taken and a convex hull is created with them. The convex hull represents a convex polyhedron that can be formed using the near neighbor points to the origin. This polyhedron is referred to as the autocorrelation exclusion zone (AEZ) in Spatial Analysis 3D. The AEZ is a 3D representation of the dead zone that exists at the origin of any autocorrelogram. The AEZ is consequently influenced strongly by single points that defy the exclusion zone and reside within the dead zone, and so it may differ significantly from the effective radius portrayed in the autocorrelation or inferred from the DRP.

On the following page is a figure showing the AEZ for the autocorrelation data mentioned above in the example described under Autocorrelation Histogram. From the figure we see that the AEZ looks spherical. Just like the Voronoi domains, we can assign a direction vector, v_{aez}, to the AEZ (see Voronoi Direction Vectors section). This direction vector gives us an idea about the shape of the dead zone, that is, if it is elongated in a particular direction. In this example, our direction vector v_{aez} is <0.09, 0.17, 0.0>, which tells us that the AEZ is in fact quite spherical. We can also take the volume of the AEZ and calculate the radius of the sphere with the equivalent volume; we call this the effective radius of the autocorrelation exclusion zone, r_{aez}. In this example the effective radius, r_{aez}, of the AEZ is 25.78 μm, which happens to approximate well the effective radius derived from the DRP. Again, real data sets that include exceptions to the exclusion zone defined by the majority of the population will yield marked discrepancies between the calculated ER and r_{aez}.

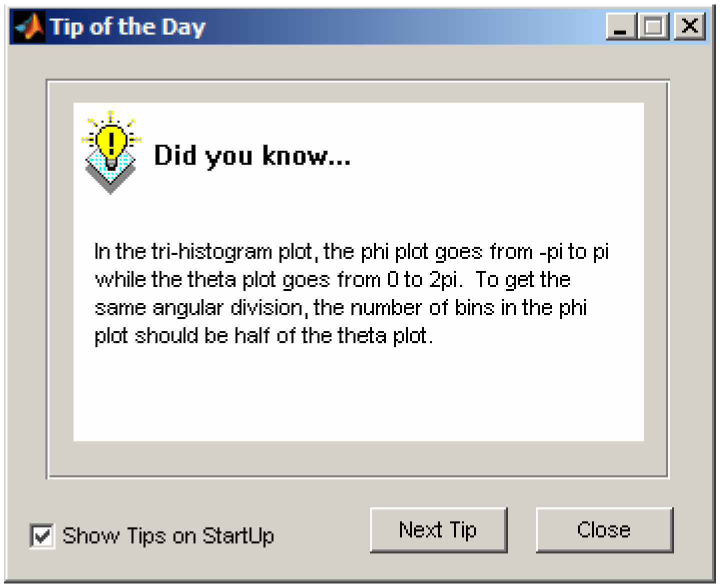


Help Menu

Under the **Help Menu** you will find some helpful information regarding Spatial Analysis 3D. The menu items available are the **Tip of the Day** dialog box and the **About Spatial Analysis** window.

Tip of the Day

As noted in Chapter 2: Getting Started, when Spatial Analysis 3D first starts, the **Tip of the Day** dialog box appears. In this dialog box, you can scroll through all the tips available in the software. If you do not wish to view the **Tip of the Day** dialog box each time Spatial Analysis 3D starts, simply uncheck the box in the bottom left hand corner of the dialog. You can retrieve the dialog again by choosing **Tip of the Day** from the **Help** menu or under the General tab of the **Options** menu. The figure below shows one example of a tip from the Tip of the Day dialog box.

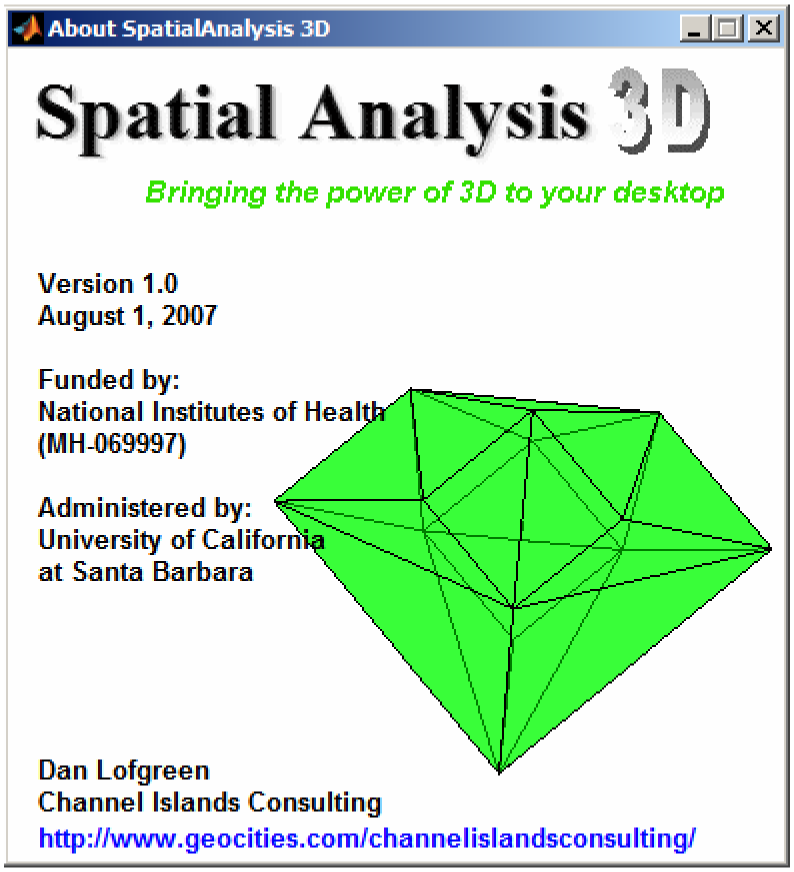


Did you know...

In the tri-histogram plot, the phi plot goes from -pi to pi while the theta plot goes from 0 to 2pi. To get the same angular division, the number of bins in the phi plot should be half of the theta plot.

About Spatial Analysis

The last item under the **Help** menu is the **About Spatial Analysis** menu item. When selected, a figure window will appear that gives some basic information about Spatial Analysis 3D: the version, author, funding agency, and website. The website in the window is a hyperlink and can be used to check for updates. The About Spatial Analysis window is shown as a figure on the following page.



About SpatialAnalysis 3D

Spatial Analysis 3D

Bringing the power of 3D to your desktop

Version 1.0
August 1, 2007

Funded by:
National Institutes of Health
(MH-069997)

Administered by:
University of California
at Santa Barbara

Dan Lofgreen
Channel Islands Consulting
http://www.geocities.com/channelislandsconsulting/

Appendix A File Manifest

Matlab fig files:

```
AboutSpatialAnalysis.fig        SpatialAnalysis.fig
saveselectedparams.fig          TipOfTheDay.fig
SAOptions.fig                   viewdlg.fig
```

Matlab m files:

```
AboutSpatialAnalysis.m          saveselectedparams.m
autocorrelation3.m              SAOptions.m
filternnpoints.m                SpatialAnalysis.m
nearestneighbor3.m              TipOfTheDay.m
polarf.m                        viewdlg.m
rosef.m                         Voronoianal3.m
saveparams.m
```

Matlab Mex files:

```
ACCorrectionFactor3             k3d
dmin3d                          nnanalysis3
f3d                             pyramidarea
g3d                             pyramidvol
```

C files:

```
func.h                          k3d_mex.c
geom.h                          kfg_funs.c
SpatialAnalysis3.h              nnanalysis.c
ACCorrectionFactor3.c           pyramidarea.c
DanVersionsphefrac.c            pyramidvol.c
dmin3d.c                        sint.c
f3.c                            sphefrac.c
g3.c                            sphevol.c
k3.c                            tetraarea.c
triarea.c
```

ACCORRECTIONFACTOR3. M

```
ACCorrectionFactor3  Autocorrelation correction factor

CF = ACCorrectionFactor3([X,Y,Z],[MINX,MINY,MINZ],[MAXX,MAXY,MAXZ],R)
Calculates the correction factor for an autocorrelation calculation.
The function calculates the fraction of a sphere with radius R centered
at [X,Y,Z] that intersects a cube with one corner at [MINX,MINY,MINZ]
and the opposite corner at [MAXX,MAXY,MAXZ].  The input variables are:

   [X,Y,Z]              Nx3 matrix of points
   [MINX,MINY,MINZ]     1x3 vector that defines one corner of the cube
   [MAXX,MAXY,MAXZ]     1x3 vector that defines the opposite corner
   R                    Mx1 vector that defines the radii

The function will return an MxN matrix, CF, that has correction factors
at each of the N points for each of the M radii.

Be aware that erroneous results could occur if a radius R=0 is used.

To compile this mex function in Windows, use the following:

   mex -v ACCorrectionFactor3.c

See also: AUTOCORRELATION3
```

AUTOCORRELATION3. M

```
AUTOCORRELATION3  Calculates the autocorrelation in 3D

[RBINS,COUNT] = AUTOCORRELATION3(P)  Performs an autocorrelation
density recovery profile calculation on points P.  P is an M x 3 matrix
that contains the (x,y,z) coordinates of M points.  The function
returns radii of the bins used in the calculation and the count in each
bin.

[...] = AUTOCORRELATION3(P,OPTIONS)  Performs the calculation with the
default parameters replaced by values in the structure OPTIONS.  The
default parameters are given in {}

options.MaxRadius          Max radius in the autocorrelation {1}
options.NumBins            Number of bins in the autocorrelation {10}
options.MinX               Min x boundary {0}
options.MinY               Min y boundary {0}
options.MinZ               Min z boundary {0}
options.MaxX               Max x boundary {1}
options.MaxY               Max y boundary {1}
options.MaxZ               Max z boundary {1}
options.WaitBar            Handle for the waitbar function {[]}
options.AxesHandle         Handle for the axes to plot the data {[]}
options.MarkerSize         Marker size to use in the plot {6}
options.EdgeCorrection     Edge correction algorithm choices are:
                           'fractional','none' {'fractional'}
options.Filter             Filter to use on the points choices are:
                           'convex hull','max radius','none' {'none'}

[...,BINVOLUME,SAMPLEVOLUME,INDICES] = AUTOCORRELATION3(...)  Returns
additional parameters:

BINVOLUME               Volume of each bin in the calculation
INDICES                 Indices of the points used in the calculation
SAMPLEVOLUME            Volume of the sample space

[...STATS] = AUTOCORRELATION3(...)  Returns a structure that contains
some additional statistics regarding the autocorrelation

STATS.reff              Effective radius of the dead space
STATS.density           Density of the points in the sample space
STATS.Dc                Critical density
STATS.rm                Max radius
STATS.k                 Reliability factor
STATS.p                 Packing factor
STATS.len               Number of points used in the calculation

[...X,Y,Z] = AUTOCORRELATION3(...)  Returns the x,y,z data of ALL the
points used when generating the autocorrelogram.  The number of points
will be (N-1)^2 where N is the number of points that were used in the
calculation.

See also: NEARESTNEIGHBOR3, VORONOIANAL3, ACCORRECTIONFACTOR3
```

DMIN3D. M

```
DMIN3D  Simulate Dmin rule in 3d

[P,R,D] = DMIN3D(NPTS, BOX, DMIN);

Input arguments:

   NPTS   number of points to generate.
   BOX    3-vector storing [wid height depth] of volume to simulate.
   DMIN   4-vector storing [mean sd lower upper] describing the
           parameters of the Normal distribution.  Random values outside
           the range of lower to upper are ignored.  lower is typically
           the soma size diameterr, and if upper is negative, there is
           no upper limit.

Output arguments:

   P      [NPTS,3] matrix storing the simulated points.
   R      NPTS-vector storing the number of rejected  when positioning
           each cell.
   D      successful dmin value used for each point.
   I      extra information about the simulation.  I is a 2-d vector
           I[1] is the total number of rejects (= sum(r))
           I[2] is an estimate of the packing density.

If the simulation could not be made (beyond the packing density),
P is set to -1.  (R and D store the values up to the point where
the simulation was aborted.)

Example:
[p,r,d,i] = dmin3d(200, [300 300 300], [40, 5, 10, -1]);

To compile this mex function in Windows, use the following:

   mex -v dmin3d.c
```

F3D. M

```
F3D  Compute F function in 3d.

RES = F3D(PTS,BOX,RANGE,VSIZE,METHOD) Invokes the mex function kfg_funs
to perform the F-function calculation with the following arguments:

  PTS     array of size [N,3] where N is number of points.
  BOX     vector of length 6 giving [XLO XHI YLO YHI ZLO ZHI]
  RANGE   vector of length 3 [LO HI NVAL] saying the distances
          at which F should be evaluated.
  VSIZE   size of a voxel (See notes).
  METHOD  integer:
          0 (no correction)
          1 (minus sampling).

Return values:
matrix RES with 2 columns and N rows, giving the distance r, and
the value of the F function at that distance r.

The F,G,K functions need compiled only once together.  To compile the
mex function kfg_funs under windows use the following:

  mex -v -g  kfg_funs.c sint.c f3.c g3.c k3.c sphefrac.c sphevol.c

See also: G3D, K3D
```

FILTERNNPOINTS. M

```
FILTERNNPOINTS  Filters the near neighbor points into classes

NNCLASS = FILTERNNPOINTS(P,NNDIST,NNINDICES,MINX,MAXX,MINY,MAXY,
                         MINZ,MAXZ)

Filters each point in P into classes based on the nearest neighbor
distances and the sample space boundary locations.

   P         M x 3 matrix that contains the (x,y,z) coordinates of M
             points.
   NNDIST    M X 1 Array of Nearest neighbor distances
   NNINDEX   M x M matrix of 1's and 0's that indicates if a point is
             a neighbor of another point
   MINX      Minimum boundary in the x-direction
   MAXX      Maximum boundary in the x-direction
   MINY      Minimum boundary in the y-direction
   MAXY      Maximum boundary in the y-direction
   MINZ      Minimum boundary in the z-direction
   MAXZ      Maximum boundary in the z-direction

The function returns an M x 1 vector NNCLASS that specifies the class
of the point

   1     Normal point within the sample window
   2     Point is double infected.  One of its near neighbors is
         infected
   3     Point is infected.  It is closer to any boundary than its
         nearest neighbor.
   4     Point is on the convex hull

See also: NEARESTNEIGHBOR3
```

G3D. M

```
G3D  Compute G function in 3d.

RES = G3D(PTS,BOX,RANGE,METHOD) Invokes the mex function kfg_funs
to perform the G-function calculation with the following arguments:

   PTS      array of size [N,3] where N is number of points.
   BOX      vector of length 6 giving [XLO XHI YLO YHI ZLO ZHI]
   RANGE    vector of length 3 [LO HI NVAL] saying the distances at
            which F should be evaluated.
   METHOD   integer:
            1 (minus sampling).
            3 (Hanisch G3).


Return values:

matrix RES with 2 columns and N rows, giving the distance r, and
the value of the G function at that distance r.

The F,G,K functions need compiled only once together.  To compile the
mex function kfg_funs under windows use the following:

  mex -v -g  kfg_funs.c sint.c f3.c g3.c k3.c sphefrac.c sphevol.c

See also: F3D, K3D
```

K3D. M

```
K3D  Compute K function in 3d.

RES = K3D(PTS,BOX,RANGE,METHOD) Invokes the mex function kfg_funs
to perform the K-function calculation with the following arguments:


   PTS      array of size [N,3] where N is number of points.
   BOX      vector of length 6 giving [XLO XHI YLO YHI ZLO ZHI]
   RANGE    vector of length 3 [LO HI NVAL] saying the distances at
            which F should be evaluated.
   METHOD   integer:
            0 (translation)
            1 (isotropic)


Return values:

matrix RES with 2 columns and NVAL rows, giving the distance r, and
the value of the K function at that distance r.

The F,G,K functions need compiled only once together.  To compile the
mex function kfg_funs under windows use the following:

  mex -v -g  kfg_funs.c sint.c f3.c g3.c k3.c sphefrac.c sphevol.c

See also: F3D, G3D
```

NEARESTNEIGHBOR3. M

```
NEARESTNEIGHBOR3 Calculates the nearest neighbor distances in 3D

NND = NEARESTNEIGHBOR3(P) Calculates the nearest neighbor distances for
each point in P.  P is an M x 3 matrix that contains the (x,y,z)
coordinates of M points.  The function returns the nearest neighbor
distances in an array that is Mx1

[NND,MAXNNDIST,AVGNNDIST,NUMNN,NNINDICES,NNINDEX] = NEARESTNEIGHBOR3(P)
Returns additional parameters: the maximum near neighbor distance,
the average distance of the near neighbors, the number of near
neighbors, the indicies of each near neighbor, and the index of the
nearEST neighbor.

[...,T,DELAUNAYSEGLEN,DELAUNAYAREA,DELAUNAYTOTALAREA,DELAUNAYVOL,
SEGLENINDEX,FACETINDEX,NUMFACETS] =  NEARESTNEIGHBOR3(P)  Returns
additional parameters for the delaunay tessellation.

   T                    Delaunay tessellation matrix T
   DELAUNAYSEGLEN       Delaunay segment lengths
   DELAUNAYAREA         Delaunay facet areas
   DELAUNAYTOTALAREA    Delaunay areas (for the tetrahedra)
   DELAUNAYVOL          Delaunay volumes
   SEGLENINDEX          Indices of the points that make up the segments
   FACETINDEX           Indices of the points that make up the facetes
   NUMFACETS            Number of facets for each point

See also: NNANALYSIS3, AUTOCORRELATION3, DELAUNAY3
```

NNANALYSIS3. M

```
NNANALYSIS3 Near Neighbor spatial analysis in 3D

[SegLen,FacetArea,TotalArea,Vol,SegIndex,FacetIndex,NumFacets,MinNN,
MaxNN,AvgNN,NumNN,NNIndices,NNIndex] = NNANALYSIS3(T,P) Performs a near
neighbor analysis on points P.  P is an M x 3 matrix that contains the
(x,y,z) coordinates of M points.  T is the Delaunay tessellation of
those points, generated using the delaunay3 function.  The function
returns several output arguments associated with the analysis

 SegLen       Length of each connector for each tetrahedron (6 per)
 FacetArea    Area of each facet for each tetrahedron (4 per)
 TotalArea    Total surface area of each tetrahedron
 Vol          Volume of each tetrahedron in the tessellation
 SegIndex     Indices of the connectors for each tetrahedron
 FacetIndex   Indices of the facets for each tetrahedron
 NumFacets    Number of facets each point is associated with
 MinNN        Minimum near neighbor distance
 MaxNN        Maximum near neighbor distance
 AvgNN        Average near neighbor distance
 NumNN        Total number of near neighbors
 NNIndices    Matrix containing the indices of each point's near
              neighbors
 NNIndex      Vector containing the index of the nearEST neighbor point

Dan Lofgreen 12 March 2007

To compile this mex function in Windows, use the following:

   mex -v nnanalysis3.c triarea3.c tetraarea.c

See also: DELAUNAY3, NEARESTNEIGHBOR3
```

PYRAMIDAREA. M

```
PYRAMIDAREA Calculate area of a triangular pyramid

[AREA,VECT] = PYRAMIDAREA(X,K,P)  Calculates the surface area and
direction vector for each triangular pyramid in a delaunay tessellation
or voronoi diagram.  The function accepts an Nx3 matrix, X, of
vertices, where the first column is the x coordinates, the second y
corrdinates and the third the z coordinates.  An Mx3 matrix, K, of
indices into X. These represent the 3 points that make up the base of
the pyramid.  Finally a 1x3 vector, P, that has the [x,y,z] coordinates
of the point of interest, or the apex of the pyramid.

The function returns an Mx1 vector, AREA, that is the area of the base
of each pyramid.  The function also returns an Mx3 vector, VECT, that
is the sum of the 3 vectors from the apex to each point on the base of
the pyramid.  The resultant vectors are not normalized.

To compile this mex function in Windows, use the following:

   mex -v pyramidarea.c triarea3.c

See also CONVHULLN, PYRAMIDVOL
```

PYRAMIDVOL. M

```
PYRAMIDVOL Calculate volume of a triangular pyramid

[VOL,A] = PYRAMIDVOL(X,K,P)  Calculates the surface area and volume
for each triangular pyramid in a delaunay tessellation or voronoi
diagram.  The function accepts an Nx3 matrix, X, of vertices, where the
first column is the x coordinates, the second y corrdinates and the
third the z coordinates.  An Mx3 matrix, K, of indices into X.  These
represent the 3 points that make up the base of the pyramid.  Finally a
1x3 vector, P, that has the [x,y,z] coordinates of the point of
interest, or the apex of the pyramid.

The function returns an Mx1 vector, VOL, that is the volume of each
pyramid defined by K.  The function also returns an Mx1 vector, A, that
is the area of the base of each pyramid.

The volume of a triangular pyramid is 1/3 * Abase * height

To compile this mex function in Windows, use the following:
   mex -v pyramidvol.c triarea3

See also CONVHULLN, PYRAMIDAREA
```

SAOPTIONS. M

```
SAOPTIONS Control code for the Spatial Analysis Options Window

SAOptions by itself will create a new SAOptions window with the default
values loaded into each uicontrol.  SAOptions is always created as a
modal window, so it must be closed in order to continue.

SAOptions(options)  Creates a new SAOptions window and loads the values
specified in options into the appropriate uicontrol.  Options is a
structure with field names equal to the Tag name of the uicontrol.

See also: SPATIALANALYSIS

options.MinX                      Sample space min X
options.MaxX                      Sample space max X
options.MinY                      Sample space min Y
options.MaxY                      Sample space max Y
options.MinZ                      Sample space min Z
options.MaxZ                      Sample space max Z
options.MarkerSize                Marker size for the plots
options.Precision                 Decimal precision for the table
options.Transparency              Facet transparency
options.Units                     Units for the plots and table

options.NumNNBins                 Number of nearest neighbor bins
options.NNBinSize                 Nearest neighbor bin size
options.NumDelaunaySegLenBins     Number of Delaunay seg. length bins
options.DelaunaySegLenBinSize     Delaunay segment length bin size
options.NumDelaunayAreaBins       Number of Delaunay facet area bins
options.DelaunayAreaBinSize       Delaunay facet area bin size
options.NumDelaunayVolBins        Number of Delaunay volume bins
options.DelaunayVolBinSize        Delaunay volume bin size
options.NNFilter                  Delaunay point filter

options.NumVoronoiVolBins         Number of Voronoi volume bins
options.VoronoiVolBinSize         Voronoi volume bin size
options.NumVoronoiAreaBins        Number of Voronoi area bins
options.VoronoiAreaBinSize        Voronoi area bin size
options.VoronoiFilter             Voronoi point filter

options.NumACBins                 Number of autocorrelation(AC) bins
options.MaxRadiusAC               Max radius in the autocorrelation
options.MarkerSizeAC              Markersize for the AC plot
options.EdgeCorrectionAC          Type of edge correction for the AC
options.FilterAC                  Point filter for the AC
options.PLOTAC                    FLAG for plotting the AC

options.fdivisions                Divisions in the F-function (voxel)
options.FfcnRange                 F-function range
options.FfcnNumPoints             Number of points in the F-function
options.GfcnRange                 G-function range
options.GfcnNumPoints             Number of points in the G-function
options.KFcnRange                 K-function range
options.KfcnNumPoints             Number of points in the K-function
```

`options.NumRandPoints`	Number of simulated data points
`options.RandomAvgDist`	Average distance (dmin)
`options.RandomMinDist`	Min acceptable distance for dmin
`options.RandomSTD`	Standard deviation for dmin
`options.SimulatedDataType`	Type of simulated data

SPATIALANALYSIS. M

```
SPATIALANALYSIS 3D Spatial Analysis software

SpatialAnalysis will start the Spatial Analysis 3D software.  The
softare must first be installed to use it.

SpatialAnalysis -setup  Will set up the SpatialAnalysis software.  The
setup feature will initialize some application preferences and compile
the C routines into Matlab mex functions

See also: INSTALLSPATIALANALYSIS, ABOUTSPATIALANALYSIS,
NEARESTNEIGHBOR3, AUTOCORRELATION3, VORONOIANAL3, F3D, G3D, K3D
```

VORONOIANAL3. M

```
VORONOIANAL3 Voronoi diagram analysis in 3D

[AREA,VOLUME] = VORONOIANAL3(P) Performs a Voronoi diagram analysis in
for each point in P.  P is an M x 3 matrix that contains the (x,y,z)
coordinates of M points.  The function returns the surface area and
volume of each Voronoi diagram in AREA and VOLUME respectively.

[AREA,VOLUME,V,C,K,CONVHI] = VORONOIANAL3(P)  Returns additional
parameters:

      V           Voronoi vertices
      C           Voronoi cells C
      K           Cell array of Voronoi indicies for each point
      CONVHI      Convex hull indices

[AREA,VOLUME,V,C,K,CONVHI,VCLASS] = VORONOIANAL3(P,OPTIONS) Returns
additional parameters based on a user-defined options structure

      options.MinX      Min x value in the sample window
      options.MaxX      Max x value in the sample window
      options.MinY      Min y value in the sample window
      options.MaxY      Max y value in the sample window
      options.MinZ      Min z value in the sample window
      options.MaxZ      Max z value in the sample window

VCLASS is an array of length M that specifies the class of Voronoi
diagram

      1     Normal point within the sample window
      2     Point is on the convex hull
      3     The point is infected
      4     convhulln had an error during the analysis

See also: VORONOIN, NEARESTNEIGHBOR3, AUTOCORRELATION3
```

FIG FILES

A FIG-file, with extension .fig, contains a complete description of the GUI layout and the components of the GUI: push buttons, menus, axes, and so on.

`AboutSpatialAnalysis.fig`

Contains the GUI layout for the About Spatial Analysis that can be found in the Help menu of the Spatial Analysis program.

`saveselectedparams.fig`

Contains the GUI layout for the Save Selected Parameters dialog box.

`SAOptions.fig`

Contains the GUI layout for the options window of the Spatial Analysis program.

`SpatialAnalysis.fig`

Contains the GUI layout for the main program, Spatial Analysis.

`TipOfTheDay.fig`

Contains the GUI layout for the Tip of the Day dialog box for the Spatial Analysis program.

`viewdlg.fig`

Contains the GUI layout for the view dialog box. The view dialog allows the user to change the elevation and azimuth angles.

Spatial Analysis 3D:

Statistical and visualization program for three-dimensional spatial point patterns

Version 1.0

Installation CD

Funded by:
National Institutes of Health (MH-069997)

Administered by:
Benjamin E. Reese
University of California Santa Barbara

Written by:
Dan Lofgreen
Channel Islands Consulting
Santa Maria, CA 93455

With special support from:
Stephen J. Eglen
Department for Applied Mathematics and Theoretical Physics
University of Cambridge, Cambridge UK

A

about spatial analysis 3-46, A-17
AEZ *See* autocorrelation exclusion zone
analysis menu 3-21
arrow keys 3-22, 3-23, 3-24, 3-31
auto rotate 3-9
auto scale axis 3-33, 3-34, 3-35, 3-36, 3-37, 3-38
autocorrelation analysis ii, 1-2, 1-5, 1-10, 1-11, 3-17, 3-25, 3-26, 3-43, 3-44
autocorrelation bins 3-18
autocorrelation exclusion zone 3-45
autocorrelation histogram 3-42, 3-43, 3-45
autocorrelation max radius 3-18
autocorrelation tri-histogram 3-17, 3-43
autumn *See* colormap
avg Delaunay segment length 3-31

B

Bitmap 3-7
bone *See* colormap

C

circularity 3-31
clear message window 3-6
close file 3-2
colormap 3-6, 3-7, 3-8
coma separated values (CSV) files 3-2
convex hull 3-15, 3-16, 3-22, 3-25, 3-45
cool *See* colormap
copy figure 3-7
copy options 3-7
critical density 1-7, 3-26
current filter
  autocorrelation analysis 3-17, 3-18, 3-27
  Delaunay analysis 3-23
  near neighbor analysis 3-15, 3-16, 3-22
  Voronoi analysis 3-16, 3-17, 3-24

D

decimal precision *See* options menu General
Delaunay
  facet area bins 3-15
  facet area bin size 3-15
  facet area histogram 3-34
  tessellation ii, 1-2, 1-3, 1-10, 3-10, 3-15, 3-21, 3-22. 3-23, 3-33, 3-36, 33-45
  tetrahedra volume bins 3-15
  tetrahedra volume bin size 3-15
  tetrahedra volume histogram 3-36
  segment len bins 3-15
  segment len bin size 3-15
  segment length histogram 3-33
density 1-5, 1-6, 1-7, 1-8, 3-27
direction vectors 1-3, 1-4, 3-38, 3-39, 3-40, 3-41
double infected points 3-15, 3-16, 3-22, 3-25

E

eccentricity 3-31
edge correction 3-17, 3-26
edit colormap 3-8
edit menu 3-6, 3-8, 3-9
effective radius 1-5, 1-6, 1-7, 1-8, 3-26, 3-42 ,3-45
equal aspect ratio 3-11
Excel worksheets 3-2
exclusion zone *See* autocorrelation exclusion zone
exit 3-2
export 3-2, 3-3

F

F-function ii, 1-9, 1-10, 3-18, 3-19, 3-28
  number of points 3-19
  number of voxels 1-10, 3-19
  range 3-19
false color map 3-11, 3-31 ,3-32
file menu 3-2
file types 3-2

filter *See* options menu
NN/Delaunay 3-15, 3-16, 3-22
Voronoi 3-16, 3-17, 3-25
Autocorrelation 3-17, 3-18
flip map *See* colormap

G
G-function ii, 1-9, 3-18, 3-29
number of points 3-18
range 3-18
gray *See* colormap

H
HCP *See* hexagonal close packed
help menu 3-46
hexagonal close packed 1-8 ,3-13, 3-19, 3-43, 3-44
histograms 1-5, 1-10, 3-10, 3-15, 3-16, 3-17, 3-22, 3-23, 3-25, 3-32, 3-33, 3-34, 3-36, 3-37, 3-38, 3-39, 3-40, 3-41, 3-42, 3-43, 3-44

I
infected points 3-15, 3-16, 3-22, 3-25, 3-38

J
jet *See* colormap

K
K-function ii, 1-5, 1-8, 1-9, 1-11, 3-18, 3-30

L
length *See* segment length

M
marker size *See* options menu
General 3-14
Autocorrelation 3-18
MATLAB ii, 2-2, 2-3, 3-2, 3-6, 3-7, 3-9, 3-10, 3-11, A-1
maximum neighbor distance
mean minimal distance 3-13, 3-19, 3-43
metafile 3-7

N
near neighbor analysis 1-2, 3-21, 3-22, 3-23, 3-31
near neighbor average distance 3-22
nearest neighbor bins 3-15
nearest neighbor bin size 3-15
nearest neighbor distance 1-2, 3-31, 3-32
nearest neighbor histogram 3-32, 3-33
new random set 3-13
number of points *See* F,G,K options
number of random points *See* Random Data options
number of voxels *See* F,G,K options or F-function

O
opening data *See* file menu
options
Autocorrelation 3-17, 3-18
F,G,K 3-18, 3-19
General 3-14
NN/Delaunay 3-15, 3-16
Random Data 3-19, 3-20
Voronoi 3-16, 3-17

P
path 2-2, 2-3, 3-2
plot menu 3-31
plot transparency 3-14
print 3-2, 3-4, 3-5
print preview 3-4, 3-5

Q

Qhull 1-3, 3-24, 3-25, 3-33, 3-36, 3-37

R

random 1-2, 1-5, 1-6, 1-7, 1-9, 1-10, 1-11, 3-19, 3-36 ,3-38, 3-39
random set 3-13, 3-19, 3-42, 3-43, 3-44
reset 3-14
restore marker size 3-11, 3-32
rotate 3D 3-9, 3-10

S

sample space 1-6, 1-9. 1-10, 3-16, 3-17, 3-18, 3-19
sample volume 1-7, 1-9
save 3-2, 3-3, 3-4
save as 3-2
save selected parameters 3-4
scroll 2-2, 3-22, 3-23, 3-24, 3-31, 3-46
send to new figure 3-6, 3-9
segment length 3-31, 3-33
set view 3-12, 3-13
show tips on startup 2-4, 3-14, 3-46
simulated data type 3-19, 3-20
smallest permissible distance 3-19, 3-20
sphericity 3-31
spring *See* colormap
standard deviation 1-7, 3-19, 3-20, 3-43
subplot 2-4, 2-5, 3-6, 3-9 ,3-10, 3-11, 3-17, 3-21, 3-22, 3-24, 3-25, 3-26, 3-31, 3-32, 3-33, 3-34, 3-35, 3-36, 3-37
summer *See* colormap
supported file types (See file types)
surface area 3-24, 3-31, 3-37

T

Tessellation ii, 1-2, 1-3, 1-11, 3-10, 3-15, 3-21, 3-22, 3-23, 3-36, 3-40, 3-41, 3-45
tetrahedron 1-2, 3-10, 3-22, 3-23, 3-33, 3-34, 3-36
tip of the day 2-4, 3-46
tools 1-2, 3-9, 3-10, 3-11, 3-13, 3-14, 3-21, 3-23, 3-26, 3-32
transparency 3-14, 3-24
tri-histogram 3-17, 3-43, 3-44

U

units *See* options menu General

V

vector *See* direction vectors
view all Delaunay tetrahedra
view data table 3-11, 3-26
view
  auto aspect ratio 3-11
  equal aspect ratio 3-11
  3D perspective 3-12, 3-13
  X-Y plane 3-12
  X-Z plane 3-12
  Y-Z plane 3-12
  Set view 3-12, 3-13
volume 3-31
Voronoi
  area bins 3-16, 3-17
  area bin size 3-16, 3-17
  domain direction vectors 1-3, 3-38, 3-39
  surface area 3-31, 3-37
  volume 3-16, 3-31, 3-38
  volume bins 3-16, 3-17
  volume bin size 3-16, 3-17
Voronoi domains 1-3, 1-4, 1-10, 1-11, 3-24, 3-38, 3-39, 3-40, 3-41, 3-45
voxels 1-10, 3-18, 3-19

W

waitbar 2-4, 3-12
winter *See* colormap

X

x data 3-2
X-Y plane *See* view
X-Z plane *See* view

Y
y data 3-2
Y-Z plane *See* view

Z
z data 3-2
zoom in 3-9
zoom out 3-9