Spatial Distribution of Volcanic Features on Mars

By: John Milne

Advisor: Dr. Laurent G. J. Montési

Class: GEOL 394

Date: 04/25/2016

Table of Contents

List of Tables and Figures

1. ABSTRACT:

Nearest neighbor analysis is applied to several volcanic regions of Mars. My hypothesis is the nearest neighbor spacing between volcanoes is characteristic of the thickness of the lithosphere below the volcanoes. Because the lithosphere thickens with the age of the planet, the characteristic spacing among volcanoes between regions of different ages should increase with decreasing age. Baloga et al., (2007) developed a method of applying nearest neighbor analysis to one thousand random Poisson distributions in order to better characterize the behavior of the statistics of areal Poisson distributions. Beggan and Hamilton (2010) introduced software called GIAS to perform this analysis, though it only uses Cartesian distances in its analysis. I developed software to use geodesic distances in the calculations because the spacing between volcanic edifices on Mars is large enough for the curvature of the planet to be in effect. I also develop a dataset of the centroids of the major volcanic edifices of the Tharsis region as well as subsets of the Noachian-aged volcanoes identified by Xiao et al., (2012). Along with the Circum-Hellas Volcanic Province volcanoes defined by Williams et al., (2009), I show that the trends of the mean and the medians of these datasets follow the trend of increasing in size with decreasing age of the region. Unfortunately, the error bars on the means and the quartiles surrounding the medians are of the same order as the means and medians respectively, so these numbers become indistinguishable statistically; thus, the hypothesis is not demonstrated with rigor by this analysis.

2. INTRODUCTION:

In this thesis, I study the spatial distribution of volcanic features on Mars. Specifically, I apply the methods that Richardson et al., (2013) demonstrated for the Syria Planum region of Mars and use those methods on my regions of interest: a set of Noachian-aged volcanoes identified by Xiao et al., (2012), the Circum-Hellas Volcanic Province as defined by Williams et al., (2009), and the Tharsis region. Richardson et al., (2013) used a nearest neighbor analysis to show that the volcanoes in the Syria Planum region were non-randomly spaced.

My hypothesis is that the dominant control on volcano growth on Mars depends on the thickness of the lithosphere below the volcanic field, which will induce a characteristic spacing amongst the volcanic features. The thickness of the lithosphere increases with the planet's age; thus, if my hypothesis is correct, the ages of the regions of interest in this study should exhibit a corresponding change in characteristic spacing associated with their respective ages.

The Martian geologic ages are based on crater count statistics. There are three geologic ages associated with mars: Noachian, Hesperian, and Amazonian. These ages are further subdivided into Early, Middle and Late for each age. The Noachian starts with the creation of Mars and ends at 4.0 Ga. The Hesperian continues from there to 3.0 Ga. The Amazonian spans the rest of Martian geologic history. The ages of the regions of interest in this study are: Noachian for the Noachian volcanoes identified by Xiao et al., (2012), Noachian to Hesperian for the Circum-Hellas Volcanic Province, and Hesperian to Amazonian for the Tharsis region. Therefore, the regions of interest in this study are listed above by order of oldest to youngest.

Nearest neighbor analysis of volcanoes started with the recognition that the Hawaiian island chain has a regular spacing between the islands (Green, (1887)). Once bathymetric data became available, a similar spacing was noted associated with the Louisville seamounts which span most of the Pacific (Lonsdale, (1988)). In these examples, volcanic spacing has been linked to the thickness of the lithosphere (Vogt, 1974) and the flexure associated with a volcanic shield depressing the plate (Watts et

al., (1988)). A general relation between lithospheric thickness and volcano spacing was shown by Hieronymous and Bercovici, (1999):

$$\lambda \propto H^{3/4},$$

where H is the lithospheric thickness and λ is the associated volcano spacing. Figure 1 shows a diagram of the tensile and compressive regions below a volcanic load on a section of the lithosphere. The point of inflection between compression and tension in the plate becomes the path of least resistance for magma to reach the surface and form a new volcano. This inflection point provides a minimum distance away from the existing volcano for a new magma pathway that allows the creation of a new volcano. This characteristic distance becomes λ in the above equation. The characteristic distance is related to the thickness of the lithosphere through the above relation.

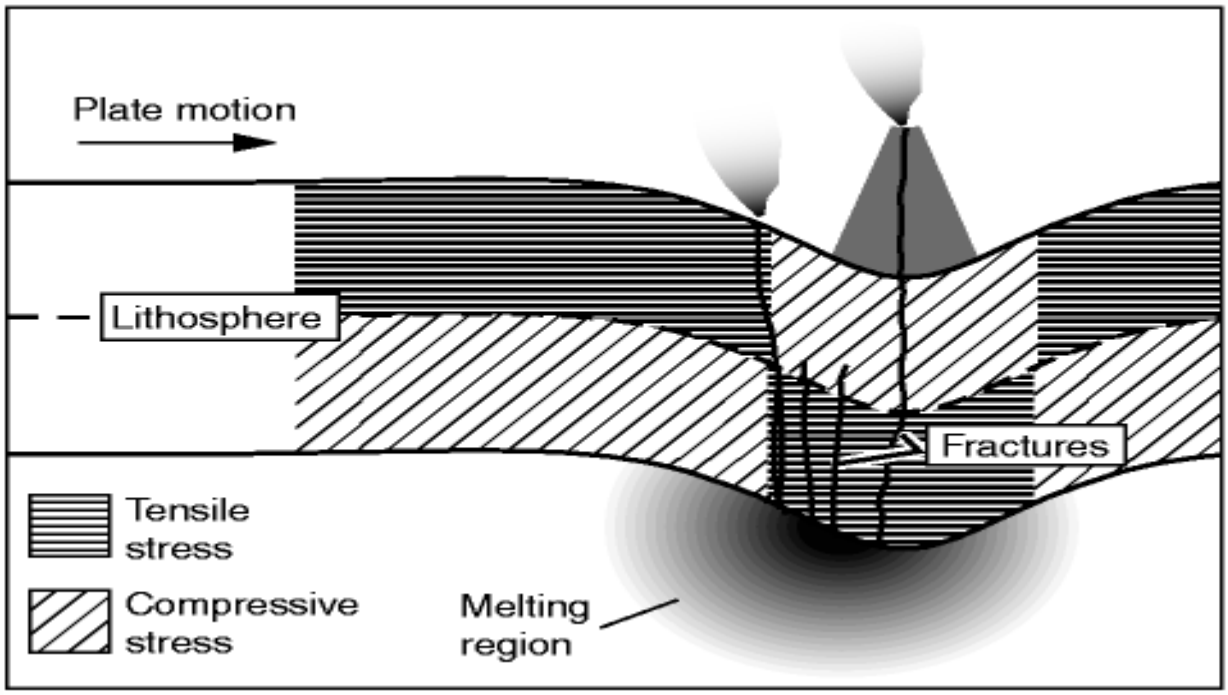

FIGURE 1: THE LOAD OF A VOLCANO ON THE LITHOSPHERE PUTS THE LITHOSPHERE UNDER BOTH COMPRESSION AND EXTENSION. A PATH THROUGH THE INFLECTION POINT IS THE PATH OF LEAST RESISTANCE FOR A NEW MAGMA PATHWAY TO PROPAGATE TO THE SURFACE AND FORM A NEW VOLCANO, AS DEPICTED BY THE LEFTMOST CHANNEL IN THE DIAGRAM GOING THROUGH THE INFLECTION POINT. FROM HIERONYMOUS AND BERCOVICI, (1999).

3. STATISTICS:

Nearest neighbor analysis begins with a dataset of map points. Map points are created from the centroids of the volcanoes in the region. The nearest neighbor to each map point is determined. The statistics for the nearest neighbor determinations are gathered for comparison with a random distribution. Richardson et al., (2013) determined 263 volcanoes exist in the Syria Planum region. Their dataset is shown in Figure 2 against a background of the local regional map of Mars. Figure 3 is the resultant nearest neighbor plot for the dataset without the background map and with links between nearest neighbors drawn. Nearest neighbor analysis involves using the mean and standard deviation of the nearest neighbor data and comparing these statistics with the statistics of a given random distribution to determine if the original data set is random in nature. Datasets found to be non-random

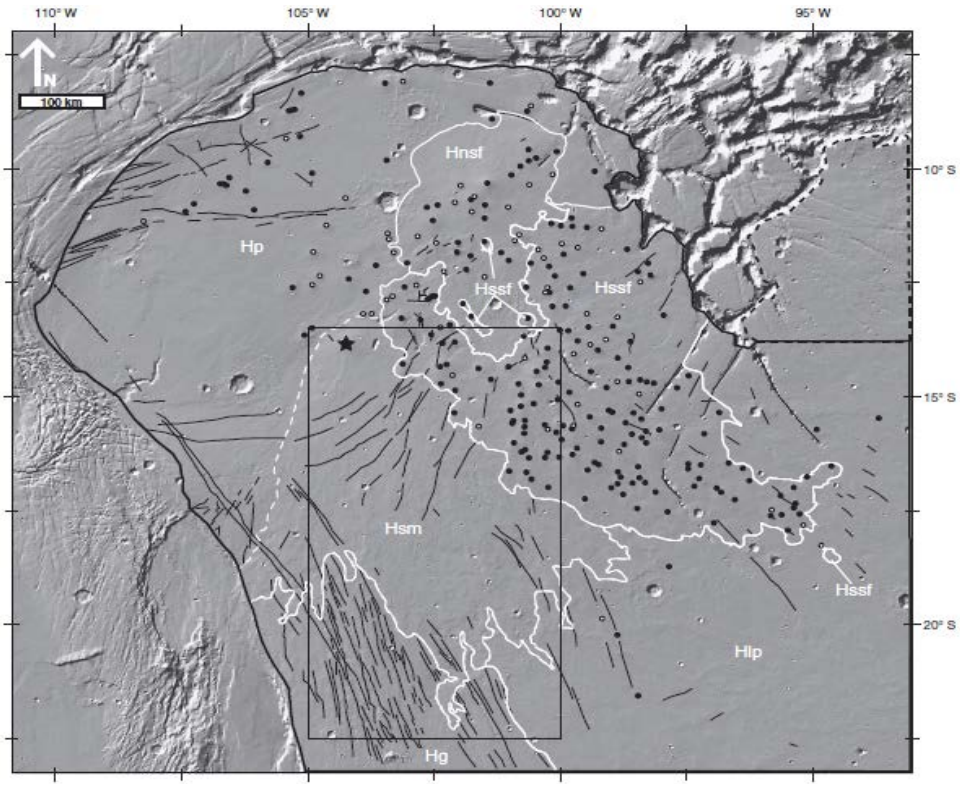

FIGURE 2: MAP OF SYRIA PLANUM WITH DOTS REPRESENTING VOLCANIC FEATURES, LINES REPRESENTING GRABENS IN THE REGION, THE STAR IS SYRIA MONS, WHITE LINES INDICATE DIFFERENT GEOLOGIC UNITS, THE THICK BLACK LINE DELINEATES THE AREAL EXTENT OF SYRIA PLANUM AND THE SQUARE IS AN AREA ANALYZED IN ANOTHER PAPER - FROM RICHARDSON ET. AL, (2013).

produce nearest neighbor statistics which deviate by more than two standard deviations from a random distribution. A non-random set of volcanoes suggests that an underlying structural control exists for that set and the structural control determined where the volcanoes formed.

Baloga et al., (2007) presented the different versions of random distributions which can be used and how to choose between them. They began with the statement of randomness: "For a set of points on a given area, it is assumed that: [1] any point has the same chance of occurring on any subarea as any other point, [2] any subarea of specified size has the same chance of receiving a point as any other subarea of that size, and [3] the placement of each point has not been influenced by that of any other point." From there, they started with the standard Poisson distribution as the random distribution to test against. If one takes a point from an infinite population of points randomly distributed in a plane, then the probability that a circle of radius, r, about that point will contain exactly k points is assumed to follow a Poisson distribution:

$$P(i) = ((\rho_o \pi r^2)^k/(i!))(\exp(-\rho_o \pi r^2)),$$

where P(i) is the probability of i points within radius *r* of a given point and ρ_o is the spatial density of the points on the plane. The probability density distribution is then given by:

$$\rho(r) = dP/dr = 2\pi\rho_o r(\exp(-\rho_o \pi r^2)\ .$$

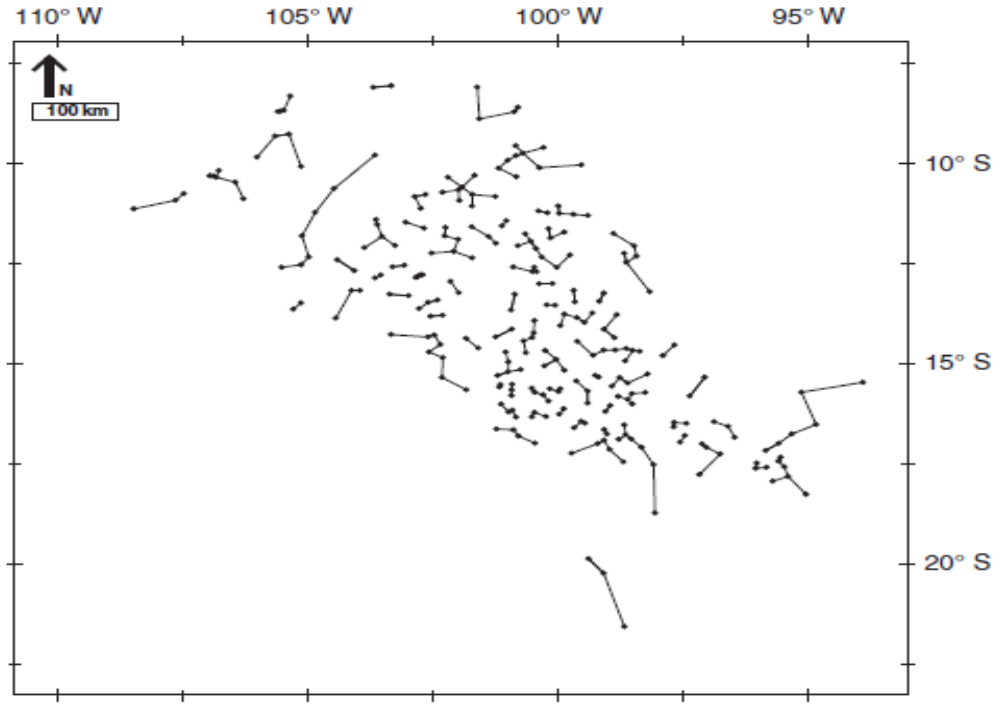

FIGURE 3: RESULT OF NEAREST NEIGHBOR ANALYSIS FROM RICHARDSON ET. AL, (2013). THIS IS THE SAME AREAL EXTENT AS FIGURE 2, MINUS THE BACKGROUND MAP AND LINKS BETWEEN EACH NEAREST NEIGHBOR.

The Poisson distribution has both positive skewness and excess kurtosis. Skewness is defined as a lack of symmetry. A quantity called the standard skewness is commonly used. I use the formula given by Baloga et al., (2007):

$$s = \sqrt{\frac{N}{6}}\left(\left(\Sigma^{N}_{i=1}(A_i - \bar{A})^3\right)/\sigma^3\right)/N),$$

where s is the skewness measure, N is the total number of data points, A_i is an individual data point, $\bar{A}$ is the mean of the dataset and σ is the standard deviation of the dataset. Negative values of skewness indicating a larger tail of the distribution at smaller values of r. The positive skewness of the Poisson distribution makes it asymmetric, with a larger tail for values above the mean.

Kurtosis is the measure of the narrowness of the distribution compared to the normal distribution. The kurtosis of the normal distribution is three, so the common normalization of the kurtosis formula subtracts three from the kurtosis. This is referred to as the excess kurtosis and becomes a measure of narrowness compared to the normal distribution. I use the standard kurtosis formula given by Baloga et al., (2007):

$$k = \sqrt{\frac{N}{24}}\left(\left(\left(\Sigma^{N}_{i=1}(A_i - \bar{A})^4/N\right)/\sigma^4\right) - 3\right),$$

where negative excess kurtosis means the peak of the distribution is flatter than the peak of the normal distribution and vice versa. Thus, the positive kurtosis of the Poisson distribution indicates it is narrower than the normal distribution.

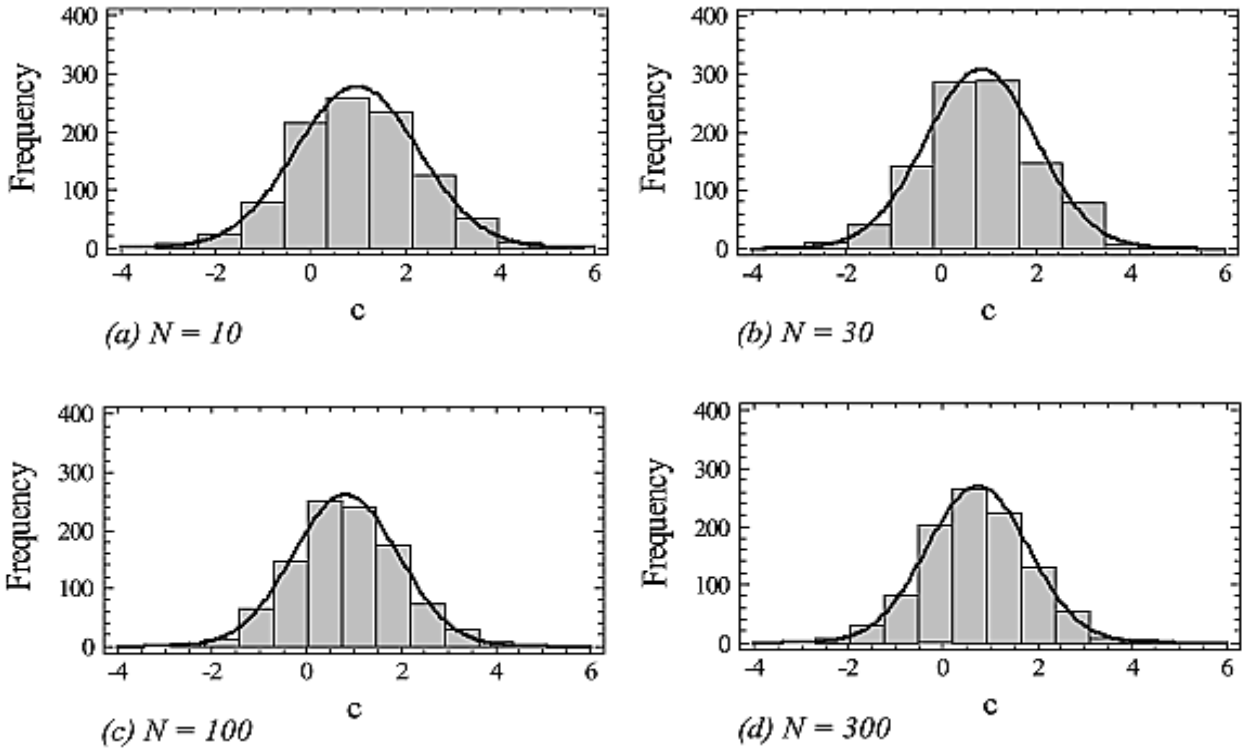

FIGURE 4: THE HISTOGRAMS FOR THE C STATISTIC USING N = 10 (A), 30 (B), 100 (C), AND 300 (D) FOR THE DATASET SIZES – FROM BALOGA ET AL., (2007) SHOWING THE MEANS ALL NEAR 1 INSTEAD OF THE ZERO OF A NORMAL DISTRIBUTION.

When comparing a given dataset against a Poisson distribution, the first statistic Baloga et al., (2007) used as a test is the c-value:

$$c = (r_a - r_e)/\sigma_e,$$

where σ_e is the standard deviation of the Poisson distribution, r_a is the mean of the dataset and r_e is the mean of the Poisson distribution. Originally, analyses assumed the c-value would be normally distributed. Baloga et al., (2007) demonstrated that this is not a good assumption. They created one thousand sets each of distributions for dataset sizes of 10, 30, 100 and 300 and reported the mean value and standard deviation of the resulting distributions of the c-values. The histograms of the c-values from Baloga et al., (2007) are reproduced in Figure 4. These histograms show that the c-values do not have means around zero like the normal distribution. Thus, they determined that the c statistic needs to be used with its actual bounds instead of reporting the standard $\pm 2\sigma$. All figures, therefore, show the bounds for which the c statistic would match the Poisson distribution for a comparable dataset of that size and areal extent due to this change of the c statistic with each dataset. Datasets outside the range of the c statistic are inferred to be non-random in a Poisson sense.

The second statistic Baloga et al., (2007) used as a test is the R-value:

$$R = r_a/r_e.$$

The R-value will tend towards 1 for randomly distributed datasets, where R>1 suggests a maximized packing arrangement while R<1 suggests a more clustered dataset compared to a Poisson distribution. Baloga et al., (2007) used the same one thousand datasets to determine the range of expected values for the R-value. Again, these values are not general and must be determined per dataset and datasets outside the range are considered to be non-random in a Poisson sense.

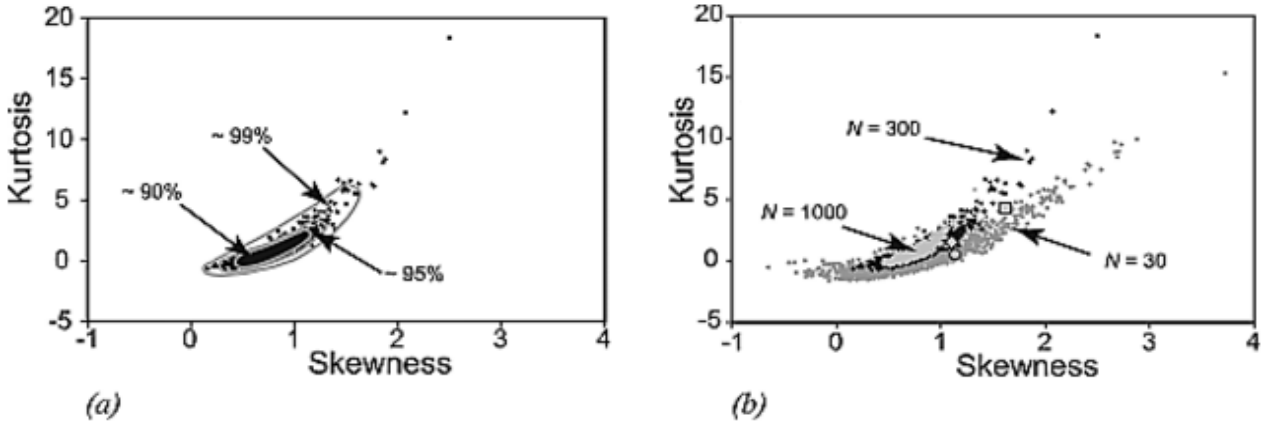

FIGURE 5: (A) PLOT OF THE CORRELATED KURTOSIS VERSUS SKEWNESS STATISTICS, WITH A SAMPLE SIZE OF *N* = 300. THE SIMULATED VALUES FORM A BROAD ARC BUT ARE CONCENTRATED IN AN ELLIPSOID CENTERED ON SKEWNESS ~1 AND KURTOSIS ~0. FROM THE OUTSIDE IN, THE THREE GRAY LINES ENCLOSE 99%, 95%, AND 90%, OF THE 1000 SIMULATIONS, RESPECTIVELY. (B) THE SIMULATED SKEWNESS AND KURTOSIS ELLIPSOIDS FOR SAMPLES SIZES OF *N* = 30 (DARK GRAY DOTS), *N* = 300 (BLACK DOTS), AND *N* = 1000 (LIGHT GRAY DOTS).

Baloga et al., (2007) also use a skewness versus kurtosis plot to compare a dataset's skewness and kurtosis with the range of skewness and kurtosis of the same one thousand distributions (Figure 5). Datasets outside of the range of skewness versus kurtosis are inferred to be non-random.

Beggan and Hamilton, (2010) produced a MATLAB graphical interface called GIAS (Geologic Image Analysis Software) which contains the above set of statistics in a nearest neighbor analysis package. GIAS takes in a set of geographical Cartesian coordinates. I have created new software to do this analysis using geodesic distances because the datasets of interest in this study have a much larger areal extent and the distances between points in these datasets are influenced by the curvature of the planet.

4. METHODS:

The datasets used in this study are lists of longitude and latitude pairs for the centroids of Martian volcanoes of a given Martian volcanic province. The Noachian dataset is taken from Xiao et al., (2012). I use only the Group I volcanoes identified therein to ensure I have actual volcanoes. The Group II and Group III volcanoes identified by Xiao et al., (2012) are so heavily degraded that it is difficult to see that they are indeed actual volcanoes. The Tharsis dataset I derived by identifying the centroids of the named volcanoes in the Tharsis region. The Hellas dataset is taken straight from Williams et al. (2009), who defined the Circum-Hellas Volcanic Province.

The software I wrote takes the longitude/latitude lists of volcano centroids and uses the Haversine formula to find the geodesic distance between each point. The minimum distance associated with each point is determined and indexed in order to find the nearest neighbor for each point. From the list of nearest neighbor distances for a given dataset, the following statistics are calculated: mean, standard deviation, median, 25th and 75th quartiles, skewness and kurtosis.

The software then creates one thousand randomly generated sets of points similar to the Baloga et al., (2007) method for determining the statistics of the Poisson distributions against which the original dataset will be compared. These datasets each have the same number of data points as the original dataset. They are also within five percent of the areal extent of the original dataset, and are further

constrained to be within the longitude and latitude ranges of the original dataset. All of the statistics calculated for the original dataset are also calculated for the one thousand individual random datasets. This allows the mean and standard deviation of the Poisson distributions to be determined for the given dataset size.

The R-value and c-value are subsequently determined to obtain the mean and standard deviation for those values for each of the one thousand distributions as well as the original dataset. The mean and standard deviation of the R-value and the c-value for the one thousand distributions becomes the range against which the original dataset is compared. This range and the R-value and the c-value for the original dataset are plotted against each other to visually determine whether the original dataset's statistics are within the range of the R-value and c-value of the one thousand random distributions.

The skewness and kurtosis are also calculated for each of the one thousand random distributions. A plot of these one thousand values gives the range on a graph for which another random distribution's skewness and kurtosis would plot against; thus, a graphical comparison of the skewness versus kurtosis space also provides a test for randomness.

Ostensibly, these three statistical analyses point in the same direction. They should suggest whether or not the original dataset can be considered random in a Poisson sense. If the three statistics all point to a non-random distribution for the dataset in question, then the inference is that there exists an underlying structural control on the distribution of the volcanoes in that region of Mars.

5. DATA ANALYSIS:

The first dataset analyzed is the Tharsis dataset, consisting of the centroids of the named volcanoes in the Tharsis region. The ages of the named Tharsis volcanoes are all Early Amazonian circa 3.0 Ga. Figure 6 is the plot of those volcanoes along with their names and with their nearest neighbors connected. The geodesic calculator produced the following statistics for the Tharsis dataset (Table 1):

Mean + STD (km)	Median (km)	25^{th} Percentile (km)	75^{th} Percentile (km)	R-value	c-value	skewness	kurtosis
530 ± 340	625	150	820	1.1 ± 1.3	0.2 ± 1.3	-0.05	-1.30

Table 1: Nearest Neighbor statistics for the Tharsis Region of Mars.

Figure 7 is a compilation of the histogram of the nearest neighbors of the Tharsis dataset; a box-and-whisker plot; the R and c statistics plotted against the box encompassing $R \pm 2\sigma$ and $c \pm 2\sigma$ for 1000 randomly-generated Poisson distributions of the same size, latitude/longitude range, and areal extent (± 5%) as the Tharsis dataset; and the skewness versus kurtosis plot.

The R test, the c test and the skewness versus kurtosis plot for the Tharsis dataset all fall within the statistics of the Poisson distributions, so this dataset cannot be distinguished from a random dataset. Thus, the characteristic spacing of 530 ± 340 km does not give any information about the underlying lithospheric thickness.

The next dataset analyzed is the Circum-Hellas Volcanic Province as defined by Williams et al., (2009). The age of these volcanic edifices are approximately 3.8 to 3.6 Ga. This puts the age of the Hellas volcanoes at roughly the boundary between the Noachian and Hesperian ages. Figure 8 is the plot of those volcanoes along with their names and with their nearest neighbors connected.

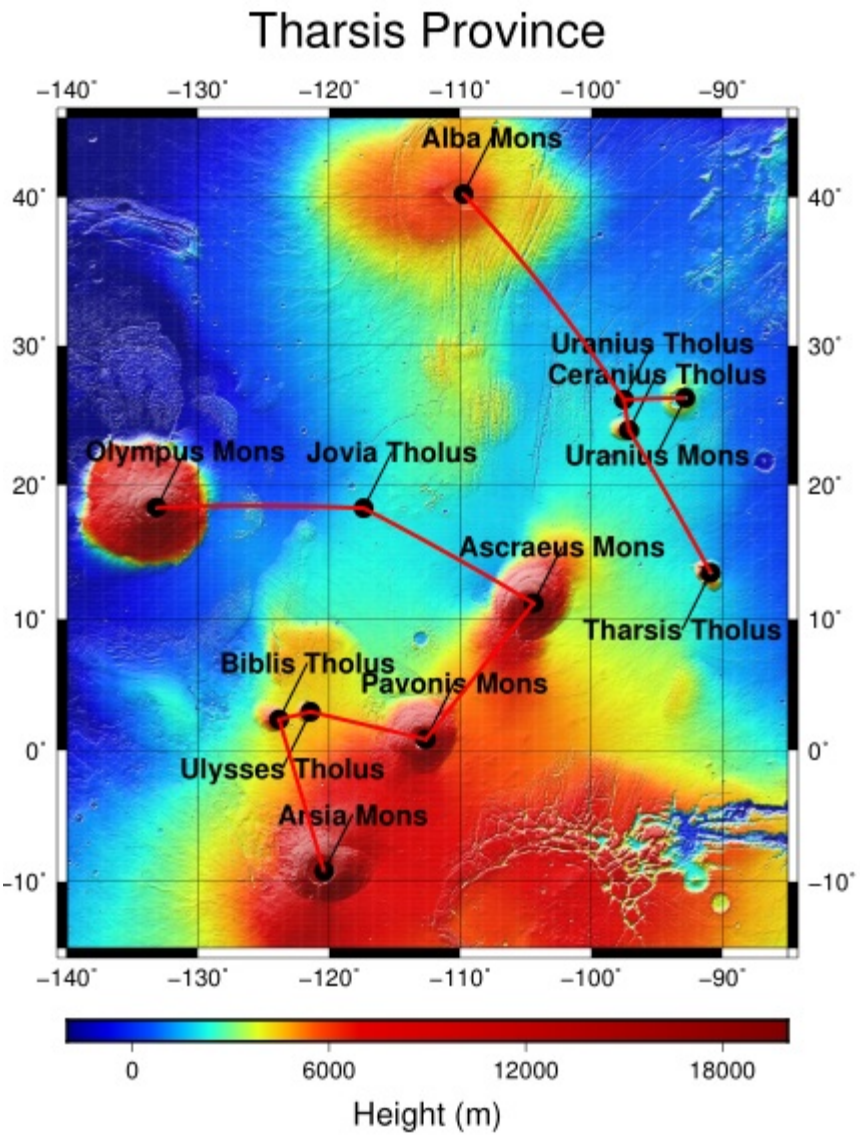

FIGURE 6: THARSIS REGION OF MARS SHOWING THE 12 NAMED VOLCANOES IN THE REGION CONNECTED TO THEIR RESPECTIVE NEAREST NEIGHBORS.

The geodesic calculator produced the following statistics for the Hellas dataset (Table 2):

Mean + STD (km)	Median (km)	25th Percentile (km)	75th Percentile (km)	R-value	c-value	skewness	kurtosis
520 ± 300	390	390	900	1.2 ± 1.3	0.4 ± 1.2	0.45	-0.98

Table 2: Nearest Neighbor statistics for the Circum-Hellas Volcanic Province of Mars.

Figure 9 is a compilation of the histogram of the nearest neighbor distances; a box-and-whisker plot; the R and c statistics plotted against the box encompassing R ± 2σ and c ± 2σ for 1000 randomly-generated Poisson distributions of the same size, latitude/longitude range, and areal extent (± 5%) as the Hellas dataset; and the skewness versus kurtosis plot.

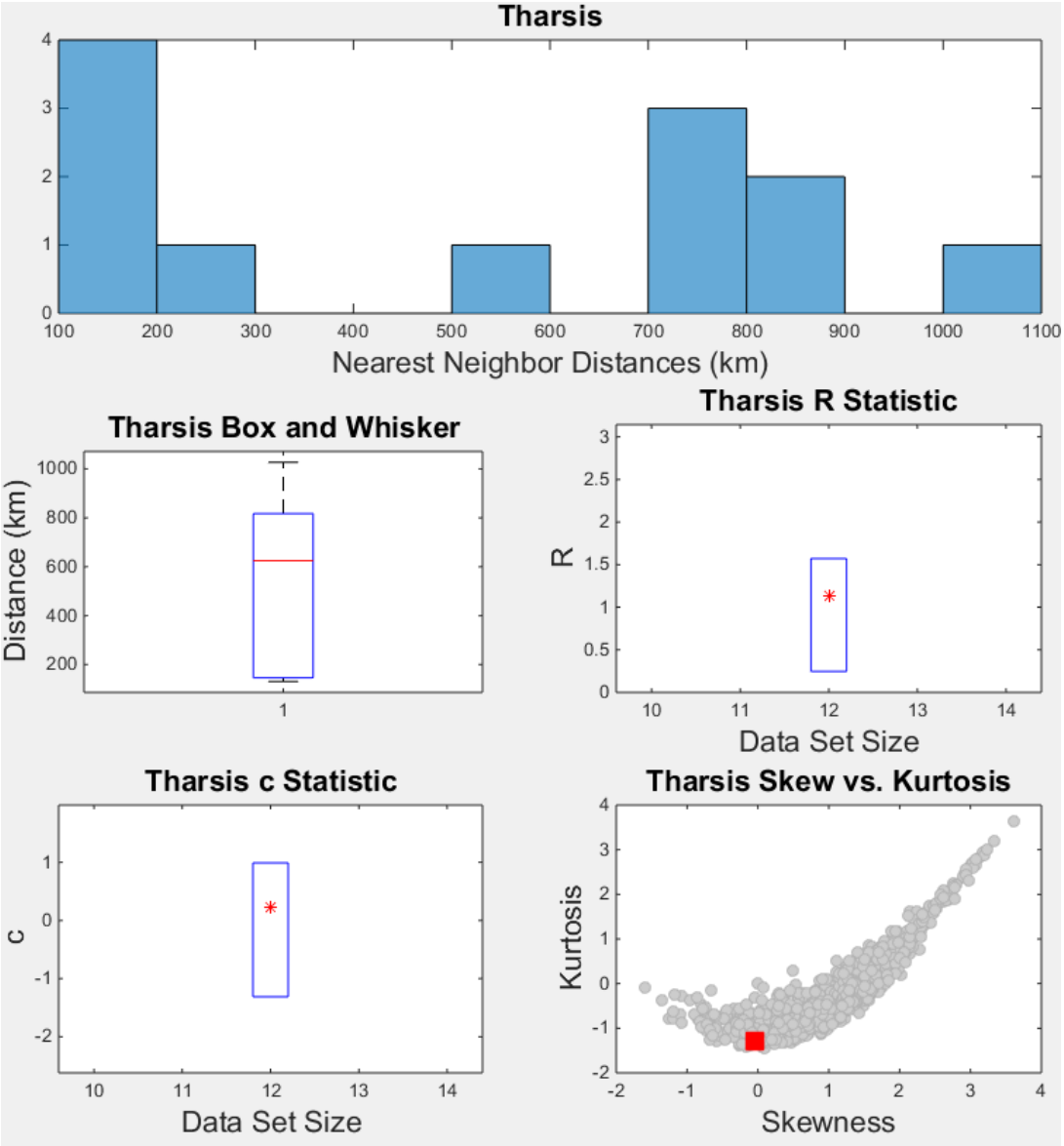

FIGURE 7: THARSIS DATASET COMPILATION OF HISTOGRAM OF NEAREST NEIGHBORS; BOX-AND-WHISKER PLOT SHOWING THE LARGE NUMBER OF OUTLIERS; R AND C VALUES PLOTTED AGAINST THE R ± 2-SIGMA AND C ± 2-SIGMA RANGES OF 1000 POISSON DISTRIBUTIONS; AND THE SKEWNESS VERSUS KURTOSIS PLOT.

The R test, the c test and the skewness versus kurtosis plot for the Hellas dataset all fall within the statistics of the Poisson distributions, so this dataset cannot be distinguished from a random dataset. As such, the characteristic spacing of 520 ± 300 km does not give any information about the underlying lithospheric thickness.

The next dataset analyzed is the set of Noachian-aged volcanoes identified by Xiao et al., (2012). These are some of the oldest volcanoes on Mars and date back as far as 4.1 Ga individually. Figure 10 is plot of these unnamed volcanoes connected to their respective nearest neighbors. The geodesic calculator determined the following statistics for the Noachian dataset (Table 3):

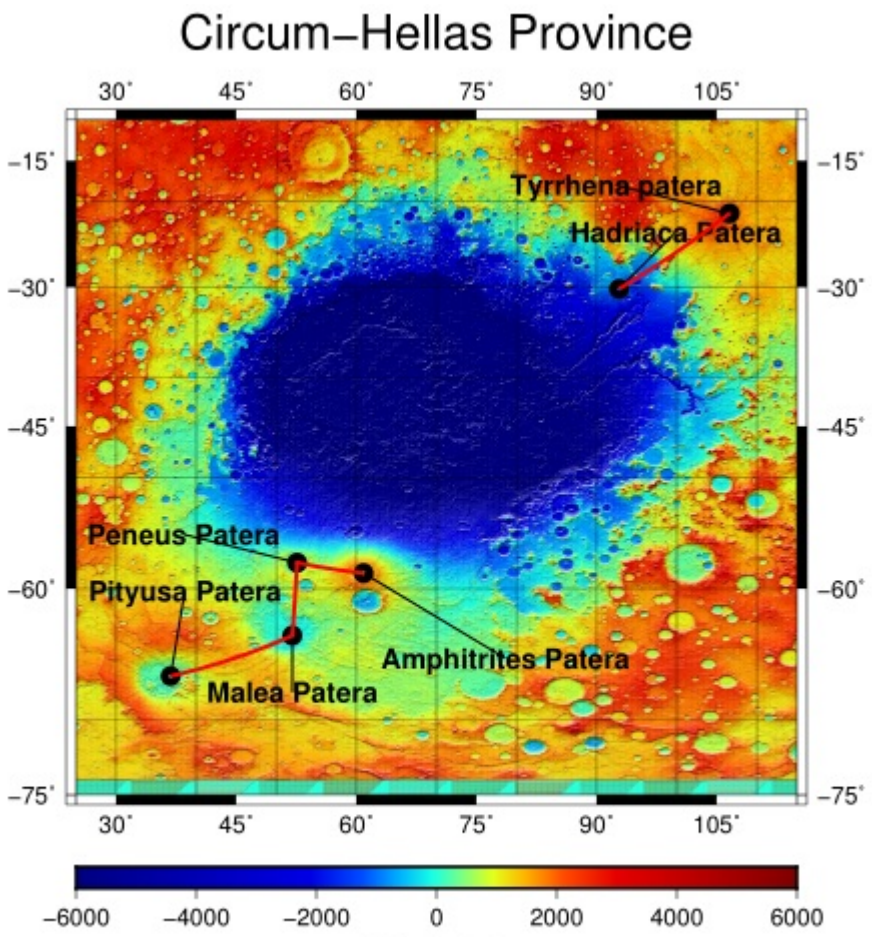

FIGURE 8: CIRCUM-HELLAS VOLCANIC PROVINCE PATERAE CONNECTED TO EACH NEAREST NEIGHBOR.

Mean + STD (km)	Median (km)	25th Percentile (km)	75th Percentile (km)	R-value	c-value	skewness	kurtosis
460 ± 620	220	150	360	0.7 ± 1.6	-0.6 ± 1.6	6.72	8.82

Table 3: Nearest Neighbor statistics for the Noachian volcanoes of Mars.

Figure 11 is the compilation of the histogram of the nearest neighbors of the Noachian dataset; a box-and-whisker plot; the R and c statistics plotted against the box encompassing R ± 2σ and c ± 2σ for 1000 randomly-generated Poisson distributions of the same size, latitude/longitude range, and areal extent (± 5%) as the Noachian dataset; and the skewness versus kurtosis plot.

The R test and c test show the Noachian dataset to be random, but the skewness versus kurtosis plot is not usable in any sense. The 1000 random sets create a diffuse boundary in the area where the Noachian skewness versus kurtosis plots. This represents a gray area where the skewness versus kurtosis plot is unable to be used as a discriminant one way or the other. The characteristic distance for the Noachian dataset is 450 ± 620 km, but, as with the two previous datasets, this characteristic spacing does not give any information about the underlying lithospheric thickness. It should be pointed out that the σ being greater than the mean does not get interpreted as a negative distance for the negative part

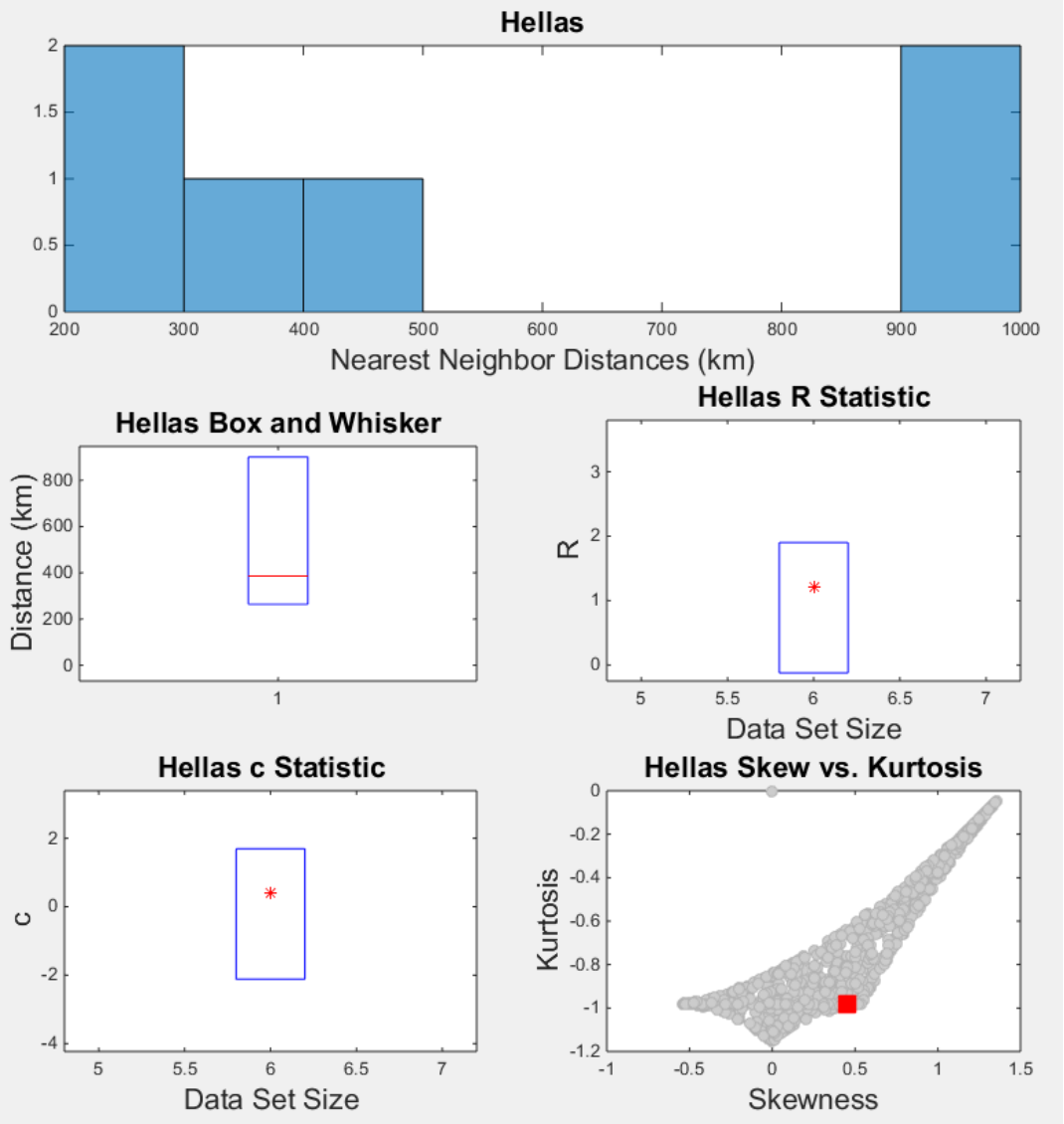

FIGURE 9: COMBINED PLOTS FOR THE HELLAS DATASET OF THE NEAREST NEIGHBOR HISTOGRAM, BOX-AND-WHISKER, R AND C STATISTICS PLOTTED AGAINST THE BOX BOUNDED BY THE R ± 2-SIGMA AND C ± 2-SIGMA STATISTICS RANGES OF 1000 RANDOMLY GENERATED POISSON DISTRIBUTIONS, AND THE SKEWNESS VERSUS KURTOSIS PLOT.

of that statistic. This is just an artifact of the computation of the standard deviation for such a highly skewed dataset and should be understood to mean that the negative half only goes to zero.

The Noachian dataset encompasses most of a hemisphere. Because of this, I also analyzed subsets of the Noachian dataset. I created three subsets, named TightNoachian, SparseNoachian and MainNoachian. These subsets can be picked out of the groupings in Figure 10 as the tight grouping around 0°, the sparse grouping around 150° and the main grouping between -180° and -90°. I used my geodesic calculator on these Noachian subsets and received the following results.

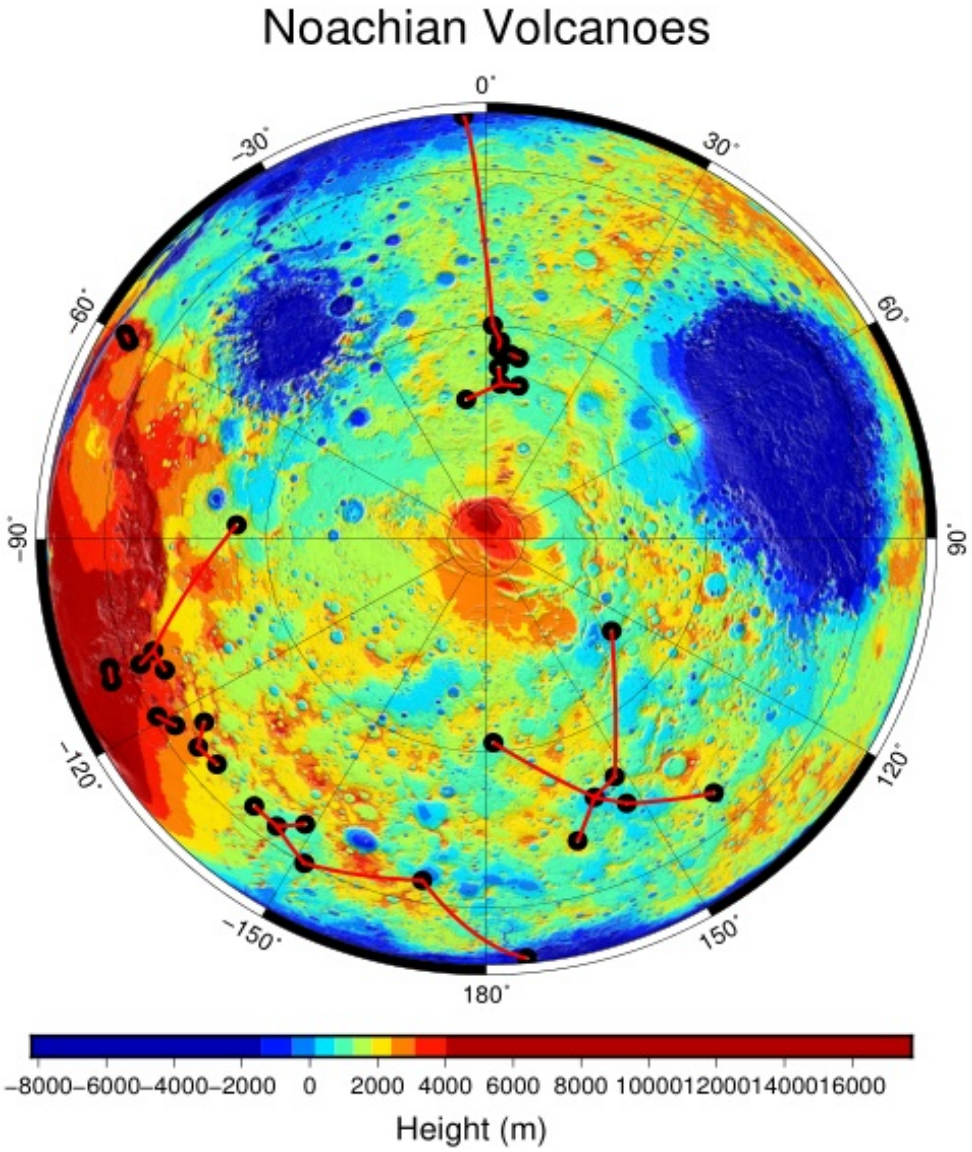

FIGURE 10: NOACHIAN-AGED VOLCANOES IDENTIFIED BY XIAO ET AL. (2012) (GROUP I). LINES CONNECT THE NEAREST NEIGHBORS.

The first subset analyzed is the TightNoachian dataset. It comprises eight volcanoes straddling the 0° longitude line. The geodesic calculator determined the following statistics for the TightNoachian dataset (Table 4):

Mean + STD (km)	Median (km)	25th Percentile (km)	75th Percentile (km)	R-value	c-value	skewness	kurtosis
150 ± 130	71	63	230	1.0 ± 1.6	0.1 ± 1.6	1.13	-0.35

Table 4: Nearest Neighbor statistics for the TightNoachian dataset of volcanoes on Mars.

Figure 12 is a compilation of the histogram of the nearest neighbor distances; a box-and-whisker plot; the R and c statistics plotted against the box encompassing R ± 2σ and c ± 2σ for 1000 randomly-

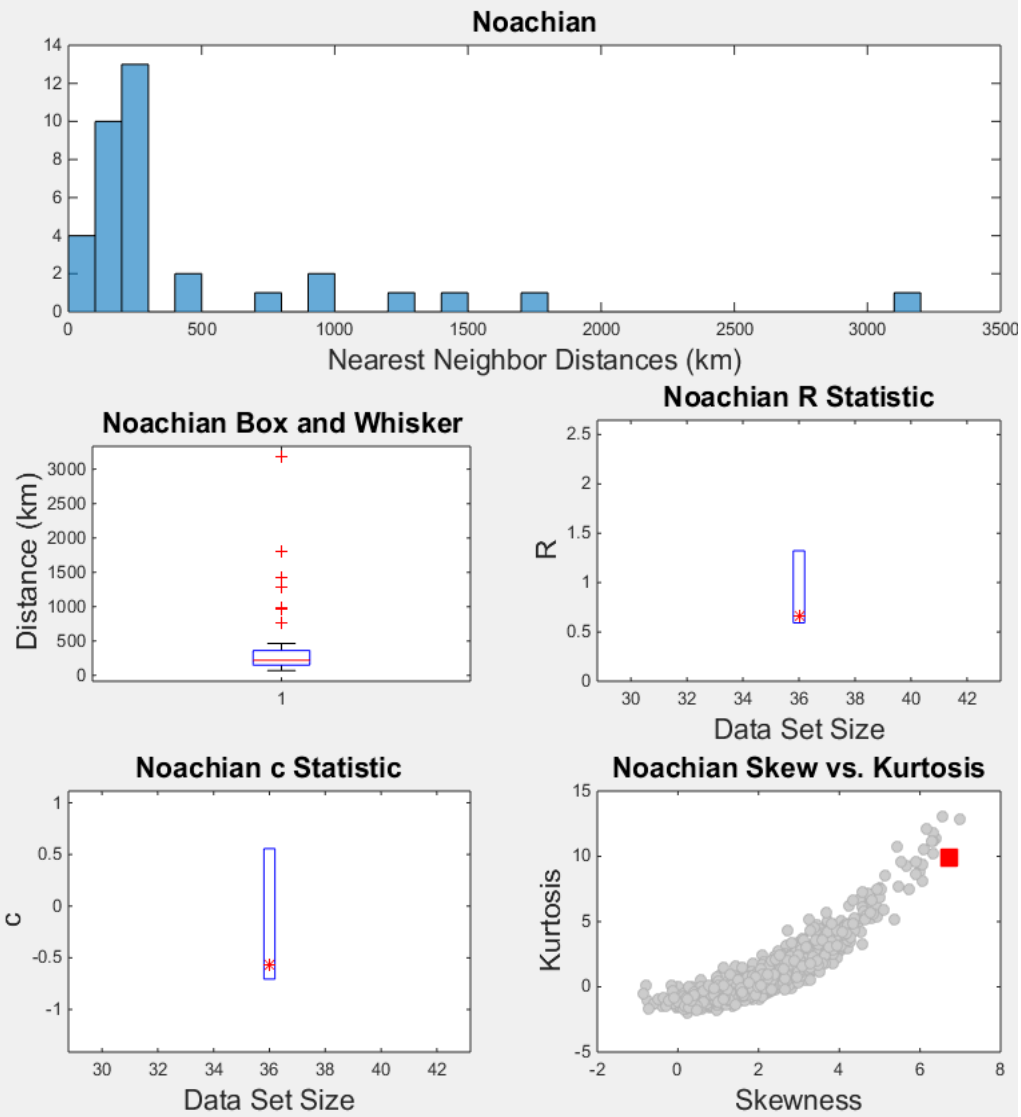

FIGURE 11: NOACHIAN DATASET COMPILATION OF HISTOGRAM OF NEAREST NEIGHBORS; BOX-AND-WHISKER PLOT SHOWING THE LARGE NUMBER OF OUTLIERS; R AND C VALUES PLOTTED AGAINST THE R ± 2-SIGMA AND C ± 2-SIGMA RANGES OF 1000 POISSON DISTRIBUTIONS; AND THE SKEWNESS VERSUS KURTOSIS PLOT.

generated Poisson distributions of the same size, latitude/longitude range, and areal extent (± 5%) as the TightNoachian dataset; and the skewness versus kurtosis plot.

The R test, the c test and the skewness versus kurtosis plot for the TightNoachian dataset all fall within the statistics of the Poisson distributions, so this dataset cannot be distinguished from a random dataset. As such, the characteristic spacing of 150 ± 130 km does not give any information about the underlying lithospheric thickness.

The next subset analyzed is the MainNoachian dataset. It comprises thirty-nine volcanoes centered on the quarter of the southern hemisphere from -180° longitude to -90° longitude. The geodesic calculator determined the following statistics for the MainNoachian dataset (Table 5):

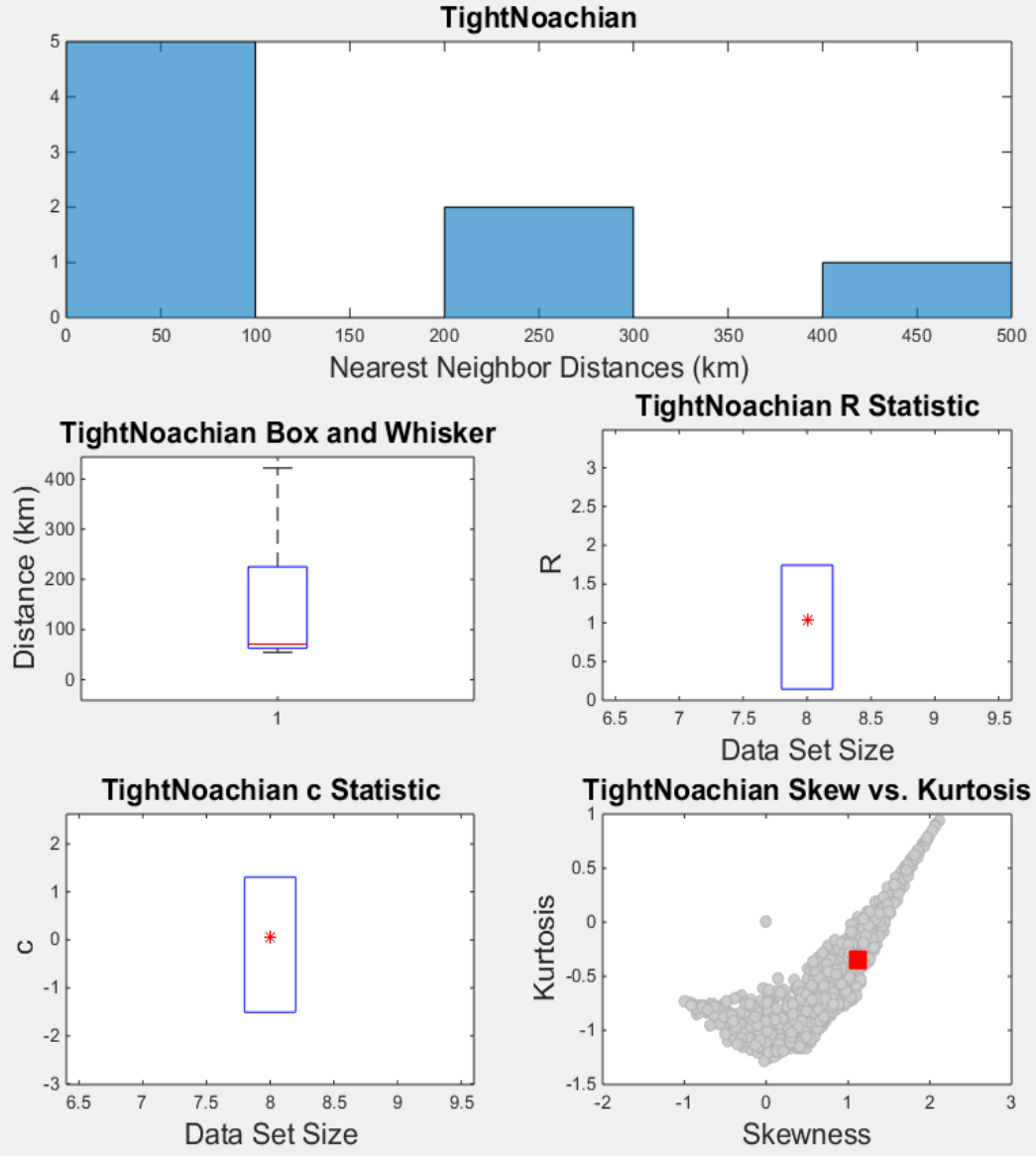

FIGURE 12: NOACHIAN DATASET COMPILATION OF HISTOGRAM OF NEAREST NEIGHBORS; BOX-AND-WHISKER PLOT SHOWING THE LARGE NUMBER OF OUTLIERS; R AND C VALUES PLOTTED AGAINST THE R ± 2-SIGMA AND C ± 2-SIGMA RANGES OF 1000 POISSON DISTRIBUTIONS; AND THE SKEWNESS VERSUS KURTOSIS PLOT.

Mean + STD (km)	Median (km)	25th Percentile (km)	75th Percentile (km)	R-value	c-value	skewness	kurtosis
200 ± 90	170	130	200	1.0 ± 0.9	0.0 ± 0.9	4.47	2.81

Table 5: Nearest Neighbor statistics for the MainNoachian dataset of volcanoes on Mars.

Figure 13 is a compilation of the histogram of the nearest neighbor distances; a box-and-whisker plot; the R and c statistics plotted against the box encompassing R ± 2σ and c ± 2σ for 1000 randomly-generated Poisson distributions of the same size, latitude/longitude range, and areal extent (± 5%) as the MainNoachian dataset; and the skewness versus kurtosis plot.

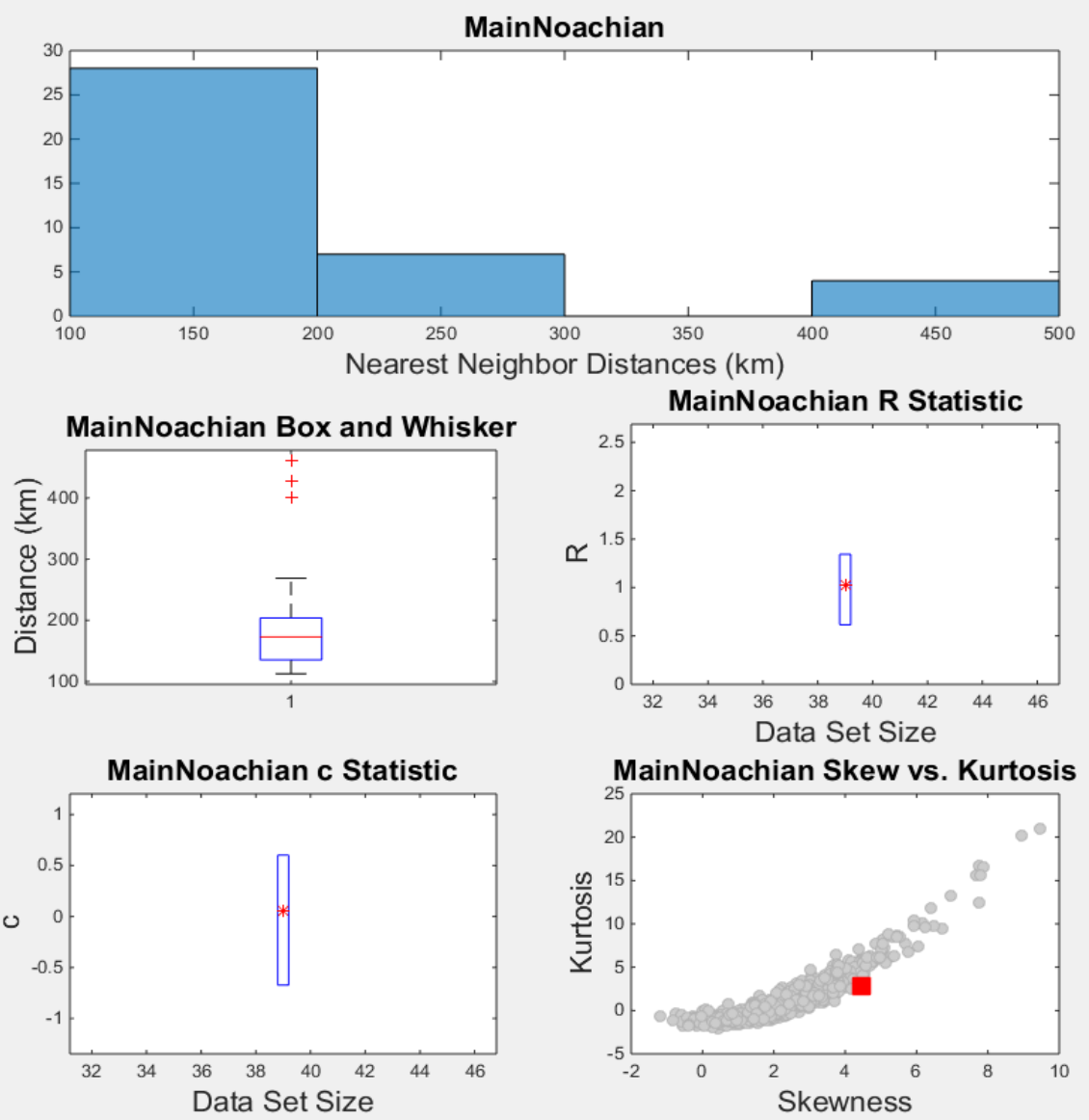

FIGURE 13: MAINNOACHIAN DATASET COMPILATION OF HISTOGRAM OF NEAREST NEIGHBORS; BOX-AND-WHISKER PLOT SHOWING THE LARGE NUMBER OF OUTLIERS; R AND C VALUES PLOTTED AGAINST THE R ± 2-SIGMA AND C ± 2-SIGMA RANGES OF 1000 POISSON DISTRIBUTIONS; AND THE SKEWNESS VERSUS KURTOSIS PLOT.

The R test, the c test and the skewness versus kurtosis plot for the MainNoachian dataset all fall within the statistics of the Poisson distributions, so this dataset cannot be distinguished from a random dataset. As such, the characteristic spacing of 200 ± 90 km does not give any information about the underlying lithospheric thickness.

The last subset analyzed is the SparseNoachian dataset. It comprises nine volcanoes straddling the 150° longitude line. The geodesic calculator determined the following statistics for the SparseNoachian dataset (Table 6):

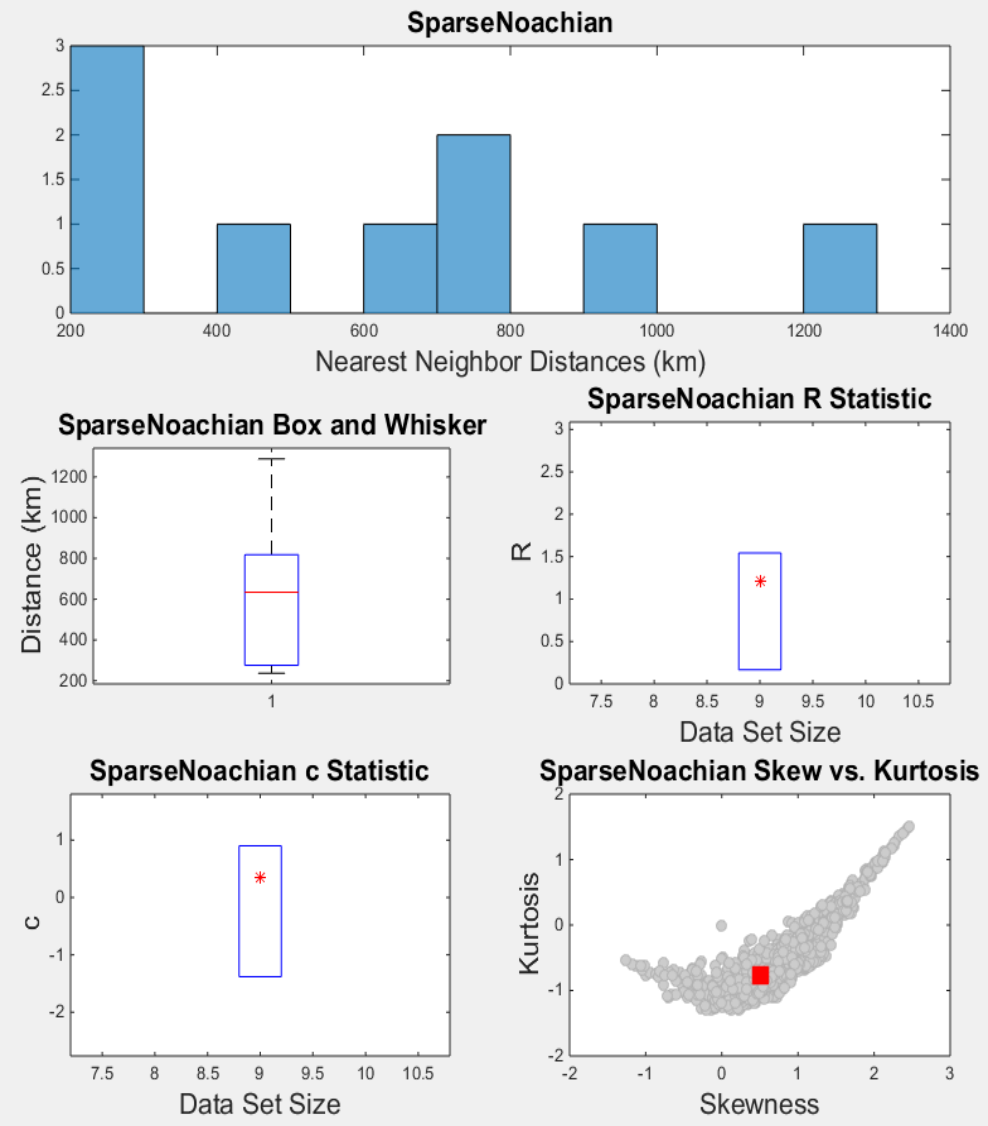

FIGURE 14: MAINNOACHIAN DATASET COMPILATION OF HISTOGRAM OF NEAREST NEIGHBORS; BOX-AND-WHISKER PLOT SHOWING THE LARGE NUMBER OF OUTLIERS; R AND C VALUES PLOTTED AGAINST THE R ± 2-SIGMA AND C ± 2-SIGMA RANGES OF 1000 POISSON DISTRIBUTIONS; AND THE SKEWNESS VERSUS KURTOSIS PLOT.

Mean + STD (km)	Median (km)	25th Percentile (km)	75th Percentile (km)	R-value	c-value	skewness	kurtosis
630 ± 360	630	260	760	1.2 ± 1.1	0.3 ± 1.1	0.51	-0.77

Table 6: Nearest Neighbor statistics for the SparseNoachian dataset of volcanoes on Mars.

Figure 14 is a compilation of the histogram of the nearest neighbor distances; a box-and-whisker plot; the R and c statistics plotted against the box encompassing R ± 2σ and c ± 2σ for 1000 randomly-generated Poisson distributions of the same size, latitude/longitude range, and areal extent (± 5%) as the SparseNoachian dataset; and the skewness versus kurtosis plot.

The R test, the c test and the skewness versus kurtosis plot for the SparseNoachian dataset all fall within the statistics of the Poisson distributions, so this dataset cannot be distinguished from a random dataset. As such, the characteristic spacing of 630 ± 360 km does not give any information about the underlying lithospheric thickness.

Figure 15 is a summary plot of all of the R values and the R test windows of the three main datasets, positioned based on dataset size. As Beggan and Hamilton (2007) showed, the R test window widens as the dataset size decreases. Figure 16 is a summary plot of all of the c values and the c test windows of the three main datasets, again positioned based on dataset size in the same way figure 15 was created. This summary also shows the same trend of increasing window size as dataset size decreases. Figure 17 is the summary plot of all of the R values and the R test windows of the three subset datasets of the Noachian dataset. Figure 18 is the summary plot of all of the c values and the c test windows of the three main datasets. All four plots together show that none of the R and c statistics for the main datasets plot outside of the bounds of random distributions.

Lastly, Figures 19 and 20 are compilations of the box-and-whisker plots for the main datasets and the Noachian subsets respectively, ordered by age of the dataset for the three main datasets and by

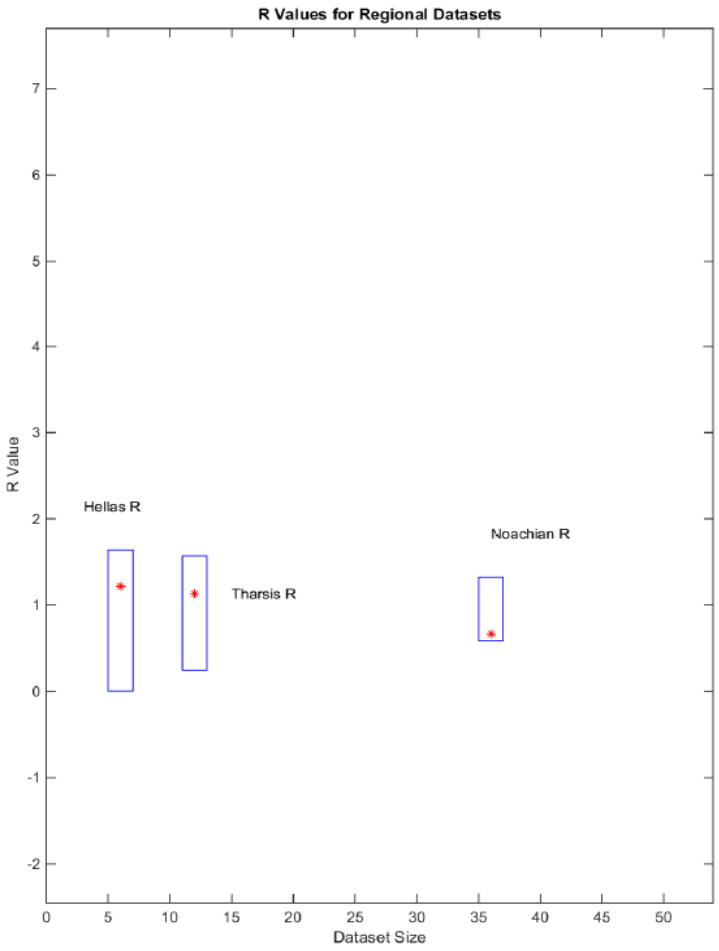

FIGURE 15: COMPILATION PLOT FOR THE MAIN THREE DATASETS' R-VALUES AND THE RESPECTIVE POISSON DISTRIBUTION RANGES. RED STARS INDICATE THE R-VALUE FOR EACH DATASET AND THE BLUE BOXES INDICATE THE RANGE OF THE ASSOCIATED POISSON DISTRIBUTIONS.

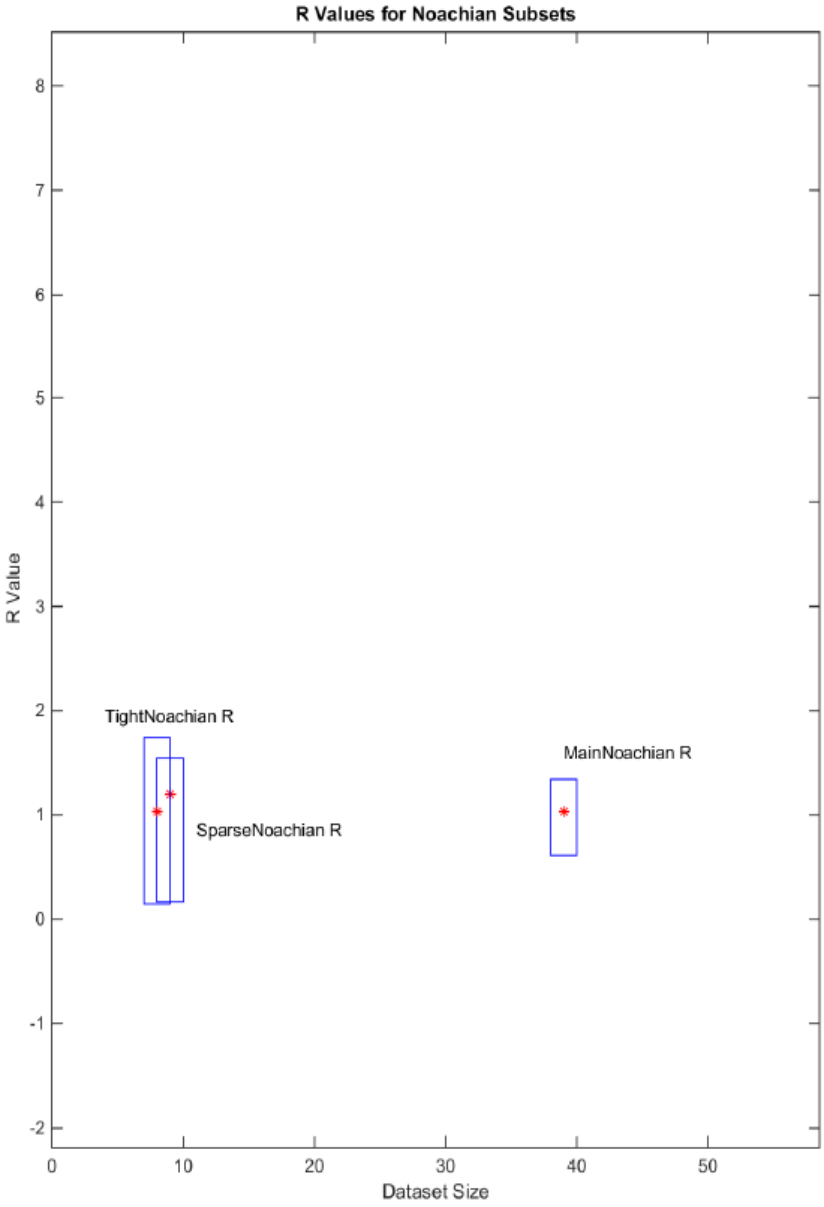

FIGURE 16: COMPILATION PLOT FOR THE NOACHIAN SUBSETS' R-VALUES AND THE RESPECTIVE POISSON DISTRIBUTION RANGES. RED STARS INDICATE THE R-VALUE FOR EACH DATASET AND THE BLUE BOXES INDICATE THE RANGE OF THE ASSOCIATED POISSON DISTRIBUTIONS.

increasing median for the Noachian subsets. This box-and-whisker comparison shows the increasing mean and median with decreasing age for the three main dataset, as would be expected if my hypothesis was correct. The median is generally a better one-number statistic compared to the mean for highly skewed distributions like all of these datasets. Even though the trend observed in the three main datasets follows the expectation of my hypothesis, the error bars on the mean as well as the quartiles that bracket the median are too large to make a definitive case for the hypothesis as all of the number are indistinguishable from each other in a statistical sense.

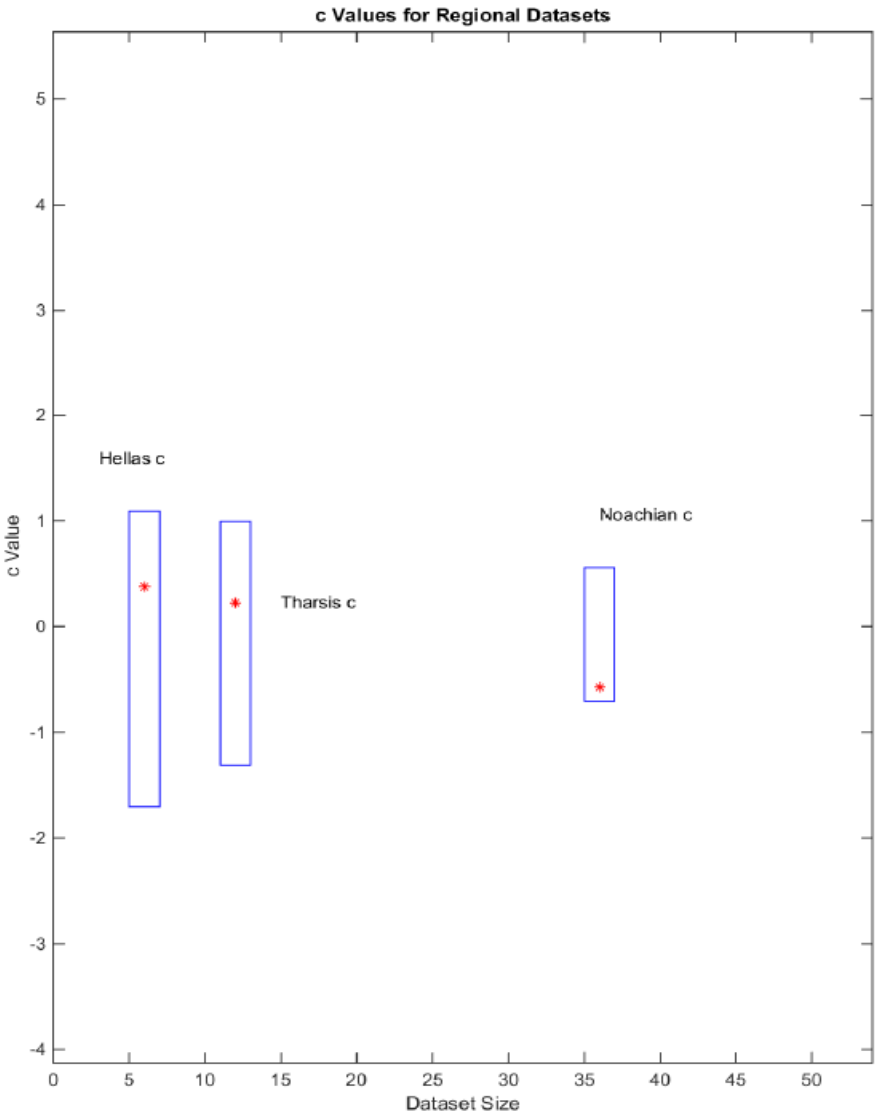

FIGURE 17: COMPILATION PLOT FOR THE MAIN THREE DATASETS' C-VALUES AND THE RESPECTIVE POISSON DISTRIBUTION RANGES. RED STARS INDICATE THE C-VALUE FOR EACH DATASET AND THE BLUE BOXES INDICATE THE RANGE OF THE ASSOCIATED POISSON DISTRIBUTIONS.

6. CONCLUSION:

The R and c tests as well as the skewness versus kurtosis plots for all of the datasets show these datasets are indistinguishable from random datasets. Besides the inability to show that any dataset is non-random, the extremely large error bars and quartile ranges make the characteristic spacings for each dataset indistinguishable from the others. The hypothesis that the size of the characteristic spacing would increase with decreasing age, associated with the increasing thickness of the lithosphere with decreasing age, did not withstand the analysis. There is a tantalizing trend among both the medians and the means of the datasets that increases with decreasing age, as the hypothesis suggests; but again, the large errors only make it tantalizing and not definitive. It would be interesting as a future

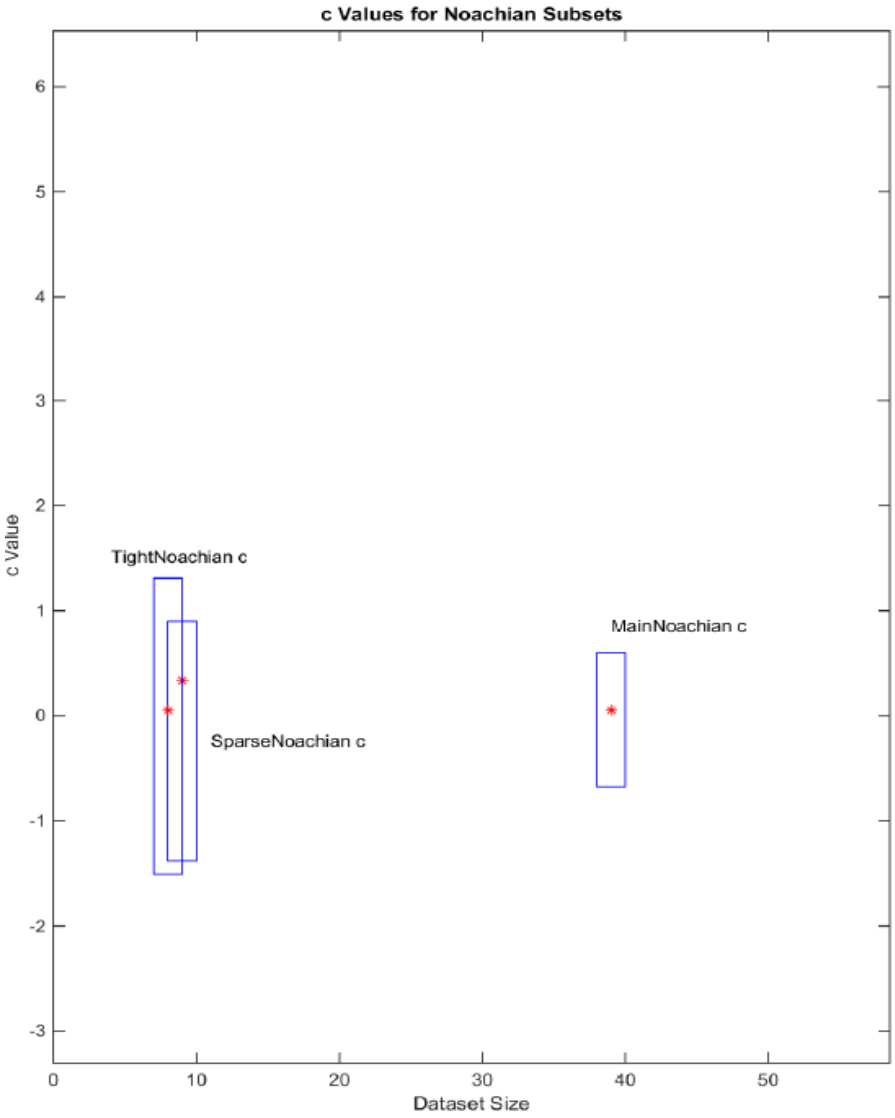

FIGURE 18: COMPILATION PLOT FOR THE NOACHIAN SUBSETS' C-VALUES AND THE RESPECTIVE POISSON DISTRIBUTION RANGES. RED STARS INDICATE THE C-VALUE FOR EACH DATASET AND THE BLUE BOXES INDICATE THE RANGE OF THE ASSOCIATED POISSON DISTRIBUTIONS.

endeavor to try to determine a different set of tests akin to the R and c tests for nearest neighbor analysis using the median instead of the mean given that the median is a better one-number statistic than the mean for highly skewed distributions. Another avenue for future work would be to identify more sets of volcanoes, especially with larger dataset sizes, to provide a better chance of success at reducing the error sizes and actually say something meaningful about the numbers produced by such an analysis.

7. ACKNOWLEDGEMENTS:

I would like to acknowledge the superb support provided by Dr. Laurent Montési. Dr. Montési's weekly meetings were extremely productive throughout the two semesters of this thesis process. On top of the weekly meetings, he created a support group in the form of his graduate students and

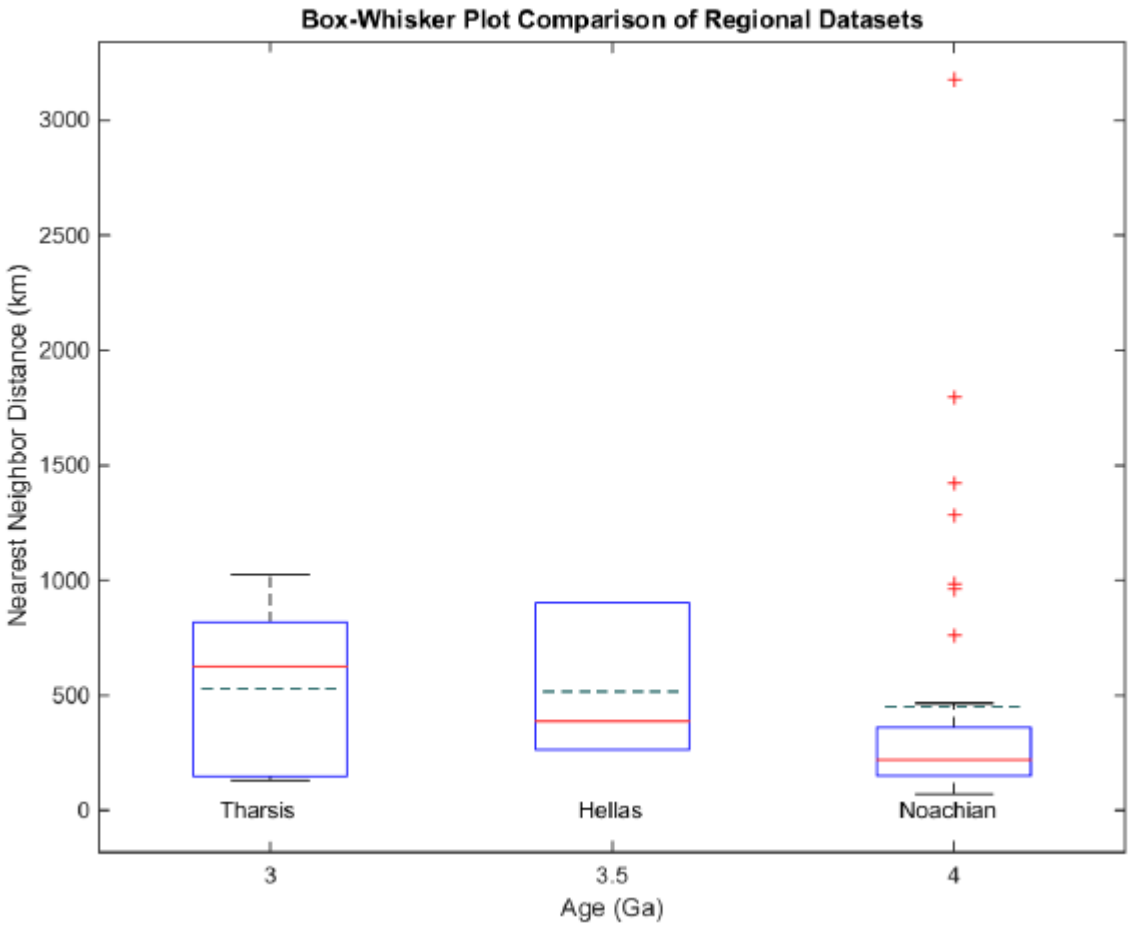

FIGURE 19: COMPILATION BOXPLOT FOR THE MAIN THREE DATASETS. RED LINES ARE THE MEDIANS, GREEN DASHED LINES ARE THE MEANS, BLUE BOXES ARE THE RANGES OF THE POISSON DISTRIBUTIONS AND RED STARS ARE OUTLIER DATA POINTS.

postdocs who provided untold feedback during presentation practice meetings. That support group included the following people: Mark Larson, Kristel Izquierdo, Alexis Martone, Joe Schools, Hailong Bai, and especially Juan Rodriguez-Gonzalez, whose singularly productive feedback during presentation practices was far and above the most helpful of all. The feedback from those meetings directly contributed to the betterment of my presentations generally. Also, my friends Frankie Tansill Davison and George and Millie Tansill provided much free food and moral support during these two semesters. Finally my mother, Ann Milne, and my sister, Jennifer Milne Kozak provided at least as much moral, financial, and mental health support as my friends. I could not have completed this assignment without all of the above!

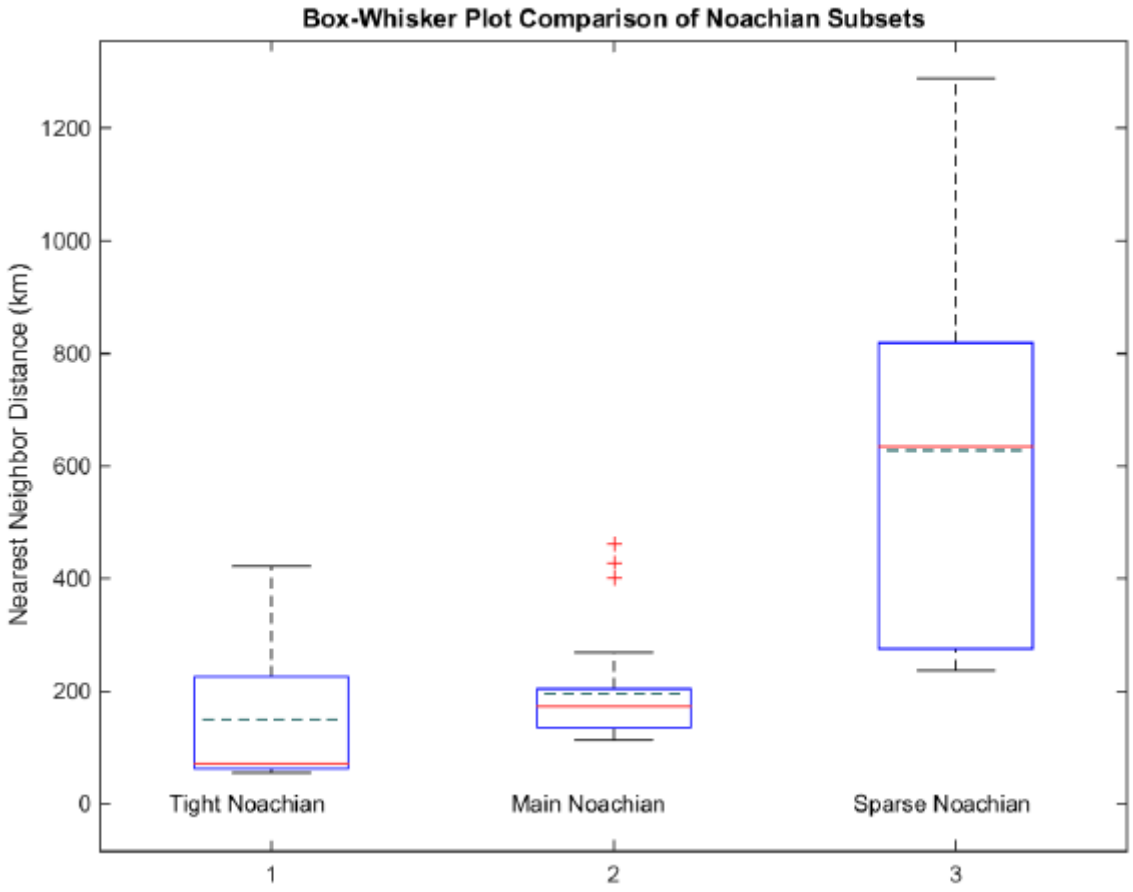

FIGURE 20: COMPILATION BOXPLOT FOR THE NOACHIAN SUBSETS. RED LINES ARE THE MEDIANS, GREEN DASHED LINES ARE THE MEANS, BLUE BOXES ARE THE RANGES OF THE POISSON DISTRIBUTIONS AND RED STARS ARE OUTLIER DATA POINTS.

8. REFERENCES:

Baloga, S.M., Glaze, L.S., Bruno, B.C., (2007). Nearest-neighbor analysis of small features on Mars: applications to tumuli and rootless cones. Journal of Geophysical Research 112, E03002. doi: 10.1029/2005JE002652.

Beggan, C., Hamilton, C.W., (2010). New image processing software for analyzing object size–frequency distributions, geometry, orientation, and spatial distribution. Computers & Geosciences 36, 539–549. DOI: 10.1016/j.cageo.2009.09.003

Cebriá, J.M., Martín-Escorza, C., López-Ruiz, J., Morán-Zenteno, D.J., Martiny, B.M., (2011). Numerical recognition of alignments in monogenetic volcanic areas: examples from the Michoacán–Guanajuato Volcanic Field in Mexico and Calatrava in Spain. Journal of Volcanology and Geothermal Research 201, 73–82. doi:10.1016/j.jvolgeores.2010.07.016.

Greeley, R., Spudis, P.D., (1981). Volcanism on Mars. Reviews of Geophysics and Space Physics 19, 13–41. doi:10.1029/RG019i001p00013.

Green, W. Lothian, Vestiges of the Molten Globe, (1887). p. 277 Part 2

Hieronymous, C.F., Bercovici, D., (1999). Discrete Alternating Hotspot Islands Formed by the Interaction of Magma Transport and Lithospheric Flexure, Nature

Lonsdale, P., (1988). Geography and history of the Louisville Hotspot Chain in the Southwest Pacific, Journal of Geophysical Research-Solid earth and Planets, Volume: 93 Issue: B4 Pages 3078-3104

Richardson, J.A., Bleacher J.E., Glaze L.S., (2013). The volcanic history of Syria Planum, Mars, Journal of Volcanology and Geothermal Research 252, 1-13, doi: 10.1016/j.jvolgeores.2012.11.007

Vogt, P.R., (1974). Volcano Spacing, Fractures, and Thickness of Lithosphere, Earth and Planetary Science Letters Volume: 21 Issue 3, Pages 235-252

Watts, A.B., (1988). Flexure of the Oceanic Lithosphere, Geophysical Journal-Oxford Voume: 92 Issue: 3 Page 519

Williams, D.A., Greeley, R., Fergason, R.L., Kuzmin, R., McCord, T.B., Combe, J.P., Head, J.W., Xiao, L., Manfredi, L., Poulet, F., Pinet, P., Baratoux, D., Plaut, J.J., Raitala, J., Neukum, G., Team, H.C.-I., (2009). The CircumCircum-Hellas Volcanic Province, Mars: overview. Planet Space Sci. 57, 895–916.

Xiao, L., J. Huang, P.R. Christensen, R. Greeley, D.A. Williams, J. Zhao, and Q. He, (2012). Earth and Planetary Science Letters 323-324, 9-18, doi:10.1016/j.epsl.2012.01.027

9. APPENDIX:

Software compilation of code

```
% Complete Nearest Neighbor Analysis routine for Mars datasets
% By: John Milne

clear all;close all;clc;

% Flowchart and dependent functions:

% TotalAnalysis.m
%   GeodesicCalculation.m
%       GeoDistances.m
%           SphereDistance.m
%       NearestNeighbor.m
%   PoissonSets.m
%       PoissonStatistics.m
%   RCTests.m
%   Plots.m
%   StatisticsTable.m

% Load the datasets of interest

load Hellascoords.dat
load Tharsiscoords.dat
load Noachiancoords.dat
load MainNoachian.dat
load SparseNoachian.dat
load TightNoachian.dat

% Run each dataset through the different parts of the analysis.

% Grab the Nearest Neighbor mean, standard deviation, and associated index
% of nearest neighbors as well as the azimuths list for the two-point
% azimuth analysis.

[HellasNNVector,HellasMeanNN,HellasMedianNN,HellasQuartiles,HellasNNStd,Hella
sDistances,HellasIndex,HellasAzimuths] = GeodesicCalculation(Hellascoords);
[TharsisNNVector,TharsisMeanNN,TharsisMedianNN,TharsisQuartiles,TharsisNNStd,
TharsisDistances,TharsisIndex,TharsisAzimuths] =
GeodesicCalculation(Tharsiscoords);
[NoachianNNVector,NoachianMeanNN,NoachianMedianNN,NoachianQuartiles,NoachianN
NStd,NoachianDistances,NoachianIndex,NoachianAzimuths] =
GeodesicCalculation(Noachiancoords);
[MainNoachianNNVector,MainNoachianMeanNN,MainNoachianMedianNN,MainNoachianQua
rtiles,MainNoachianNNStd,MainNoachianDistances,MainNoachianIndex,MainNoachian
Azimuths] = GeodesicCalculation(MainNoachian);
[SparseNoachianNNVector,SparseNoachianMeanNN,SparseNoachianMedianNN,SparseNoa
chianQuartiles,SparseNoachianNNStd,SparseNoachianDistances,SparseNoachianInde
x,SparseNoachianAzimuths] = GeodesicCalculation(SparseNoachian);
[TightNoachianNNVector,TightNoachianMeanNN,TightNoachianMedianNN,TightNoachia
nQuartiles,TightNoachianNNStd,TightNoachianDistances,TightNoachianIndex,Tight
NoachianAzimuths] = GeodesicCalculation(TightNoachian);
```

```
% Create <DataSetSize> Poisson distributions with the same number of points
% as the original dataset, from the same Lat/Long range as the original
% dataset, and within 5% of the areal extent of the original dataset.

DataSetSize = 1000;

[HellasPoissonCoordinates,HellasPoissonMeanNN,HellasPoissonSTDNN,HellasPoisso
nMedian,HellasPoissonQuartiles,HellasPoissonHull,HellasPoissonSkewness,Hellas
PoissonKurtosis] = PoissonSets(Hellascoords,DataSetSize);
[TharsisPoissonCoordinates,TharsisPoissonMeanNN,TharsisPoissonSTDNN,TharsisPo
issonMedian,TharsisPoissonQuartiles,TharsisPoissonHull,TharsisPoissonSkewness
,TharsisPoissonKurtosis] = PoissonSets(Tharsiscoords,DataSetSize);
[NoachianPoissonCoordinates,NoachianPoissonMeanNN,NoachianPoissonSTDNN,Noachi
anPoissonMedian,NoachianPoissonQuartiles,NoachianPoissonHull,NoachianPoissonS
kewness,NoachianPoissonKurtosis] = PoissonSets(Noachiancoords,DataSetSize);
[MainNoachianPoissonCoordinates,MainNoachianPoissonMeanNN,MainNoachianPoisson
STDNN,MainNoachianPoissonMedian,MainNoachianPoissonQuartiles,MainNoachianPois
sonHull,MainNoachianPoissonSkewness,MainNoachianPoissonKurtosis] =
PoissonSets(MainNoachian,DataSetSize);
[SparseNoachianPoissonCoordinates,SparseNoachianPoissonMeanNN,SparseNoachianP
oissonSTDNN,SparseNoachianPoissonMedian,SparseNoachianPoissonQuartiles,Sparse
NoachianPoissonHull,SparseNoachianPoissonSkewness,SparseNoachianPoissonKurtos
is] = PoissonSets(SparseNoachian,DataSetSize);
[TightNoachianPoissonCoordinates,TightNoachianPoissonMeanNN,TightNoachianPois
sonSTDNN,TightNoachianPoissonMedian,TightNoachianPoissonQuartiles,TightNoachi
anPoissonHull,TightNoachianPoissonSkewness,TightNoachianPoissonKurtosis] =
PoissonSets(TightNoachian,DataSetSize);

%  R and c tests for Nearest Neighbor Analysis

[Hellas_R,Hellas_R_std,HellasRTest,Hellas_c,Hellas_c_std,HellasCTest,HellasPo
issonR,HellasPR_std,HellasPoissonc,HellasPc_std,HellasSkew,HellasKurt] =
RCTests(HellasNNVector,HellasPoissonMeanNN,HellasPoissonSTDNN);
[Tharsis_R,Tharsis_R_std,TharsisRTest,Tharsis_c,Tharsis_c_std,TharsisCTest,Th
arsisPoissonR,TharsisPR_std,TharsisPoissonc,TharsisPc_std,TharsisSkew,Tharsis
Kurt] = RCTests(TharsisNNVector,TharsisPoissonMeanNN,TharsisPoissonSTDNN);
[Noachian_R,Noachian_R_std,NoachianRTest,Noachian_c,Noachian_c_std,NoachianCT
est,NoachianPoissonR,NoachianPR_std,NoachianPoissonc,NoachianPc_std,NoachianS
kew,NoachianKurt] =
RCTests(NoachianNNVector,NoachianPoissonMeanNN,NoachianPoissonSTDNN);
[MainNoachian_R,MainNoachian_R_std,MainNoachianRTest,MainNoachian_c,MainNoach
ian_c_std,MainNoachianCTest,MainNoachianPoissonR,MainNoachianPR_std,MainNoach
ianPoissonc,MainNoachianPc_std,MainNoachianSkew,MainNoachianKurt] =
RCTests(MainNoachianNNVector,MainNoachianPoissonMeanNN,MainNoachianPoissonSTD
NN);
[SparseNoachian_R,SparseNoachian_R_std,SparseNoachianRTest,SparseNoachian_c,S
parseNoachian_c_std,SparseNoachianCTest,SparseNoachianPoissonR,SparseNoachian
PR_std,SparseNoachianPoissonc,SparseNoachianPc_std,SparseNoachianSkew,SparseN
oachianKurt] =
RCTests(SparseNoachianNNVector,SparseNoachianPoissonMeanNN,SparseNoachianPois
sonSTDNN);
[TightNoachian_R,TightNoachian_R_std,TightNoachianRTest,TightNoachian_c,Tight
Noachian_c_std,TightNoachianCTest,TightNoachianPoissonR,TightNoachianPR_std,T
ightNoachianPoissonc,TightNoachianPc_std,TightNoachianSkew,TightNoachianKurt]
=
```

```
RCTests(TightNoachianNNVector,TightNoachianPoissonMeanNN,TightNoachianPoisson
STDNN);

% Plot the rest of the results.

Plots(Hellascoords,HellasIndex,HellasNNVector,Hellas_R,HellasPoissonR,HellasP
R_std,Hellas_c,HellasPoissonc,HellasPc_std,HellasPoissonCoordinates,HellasPoi
ssonHull,HellasSkew,HellasKurt,HellasPoissonSkewness,HellasPoissonKurtosis,'H
ellas');
Plots(Tharsiscoords,TharsisIndex,TharsisNNVector,Tharsis_R,TharsisPoissonR,Th
arsisPR_std,Tharsis_c,TharsisPoissonc,TharsisPc_std,TharsisPoissonCoordinates
,TharsisPoissonHull,TharsisSkew,TharsisKurt,TharsisPoissonSkewness,TharsisPoi
ssonKurtosis,'Tharsis');
Plots(Noachiancoords,NoachianIndex,NoachianNNVector,Noachian_R,NoachianPoisso
nR,NoachianPR_std,Noachian_c,NoachianPoissonc,NoachianPc_std,NoachianPoissonC
oordinates,NoachianPoissonHull,NoachianSkew,NoachianKurt,NoachianPoissonSkewn
ess,NoachianPoissonKurtosis,'Noachian');
Plots(MainNoachian,MainNoachianIndex,MainNoachianNNVector,MainNoachian_R,Main
NoachianPoissonR,MainNoachianPR_std,MainNoachian_c,MainNoachianPoissonc,MainN
oachianPc_std,MainNoachianPoissonCoordinates,MainNoachianPoissonHull,MainNoac
hianSkew,MainNoachianKurt,MainNoachianPoissonSkewness,MainNoachianPoissonKurt
osis,'MainNoachian');
Plots(SparseNoachian,SparseNoachianIndex,SparseNoachianNNVector,SparseNoachia
n_R,SparseNoachianPoissonR,SparseNoachianPR_std,SparseNoachian_c,SparseNoachi
anPoissonc,SparseNoachianPc_std,SparseNoachianPoissonCoordinates,SparseNoachi
anPoissonHull,SparseNoachianSkew,SparseNoachianKurt,SparseNoachianPoissonSkew
ness,SparseNoachianPoissonKurtosis,'SparseNoachian');
Plots(TightNoachian,TightNoachianIndex,TightNoachianNNVector,TightNoachian_R,
TightNoachianPoissonR,TightNoachianPR_std,TightNoachian_c,TightNoachianPoisso
nc,TightNoachianPc_std,TightNoachianPoissonCoordinates,TightNoachianPoissonHu
ll,TightNoachianSkew,TightNoachianKurt,TightNoachianPoissonSkewness,TightNoac
hianPoissonKurtosis,'TightNoachian');

% Make a combined plot of the Hellas, Tharsis, and Noachian datasets' R and c
tests and their standard
% deviation ranges

HYRMin = HellasPoissonR - abs(2*HellasPR_std);
HYRMax = HellasPoissonR + abs(2*HellasPR_std);

HYcMin = HellasPoissonc - abs(2*HellasPc_std);
HYcMax = HellasPoissonc + abs(2*HellasPc_std);

TYRMin = TharsisPoissonR - abs(2*TharsisPR_std);
TYRMax = TharsisPoissonR + abs(2*TharsisPR_std);

TYcMin = TharsisPoissonc - abs(2*TharsisPc_std);
TYcMax = TharsisPoissonc + abs(2*TharsisPc_std);

NYRMin = NoachianPoissonR - abs(2*NoachianPR_std);
NYRMax = NoachianPoissonR + abs(2*NoachianPR_std);

NYcMin = NoachianPoissonc - abs(2*NoachianPc_std);
NYcMax = NoachianPoissonc + abs(2*NoachianPc_std);
```

```
XLimMax = length(Noachiancoords) + (0.5*length(Noachiancoords));

TempMaxRHellas = Hellas_R + 2*(abs(Hellas_R_std));
TempMaxRTharsis = Tharsis_R + 2*(abs(Tharsis_R_std));
TempMaxRNoachian = Noachian_R + 2*(abs(Noachian_R_std));
TempMinRHellas = Hellas_R - 2*(abs(Hellas_R_std));
TempMinRTharsis = Tharsis_R - 2*(abs(Tharsis_R_std));
TempMinRNoachian = Noachian_R - 2*(abs(Noachian_R_std));

YLimRMin = 2*max([TempMinRHellas,TempMinRTharsis,TempMinRNoachian]);
YLimRMax = 2*max([TempMaxRHellas,TempMaxRTharsis,TempMaxRNoachian]);

TempMaxcHellas = Hellas_c + 2*(abs(Hellas_c_std));
TempMaxcTharsis = Tharsis_c + 2*(abs(Tharsis_c_std));
TempMaxcNoachian = Noachian_c + 2*(abs(Noachian_c_std));
TempMincHellas = Hellas_c - 2*(abs(Hellas_c_std));
TempMincTharsis = Tharsis_c - 2*(abs(Tharsis_c_std));
TempMincNoachian = Noachian_c - 2*(abs(Noachian_c_std));

YLimcMin = 2*max([TempMincHellas,TempMincTharsis,TempMincNoachian]);
YLimcMax = 2*max([TempMaxcHellas,TempMaxcTharsis,TempMaxcNoachian]);

figure('position',[470 50 400 400]);hold on;box on;orient tall;
title('R Values for Regional Datasets');
xlabel('Dataset Size');
ylabel('R Value');
xlim([0 XLimMax]);
ylim([YLimRMin YLimRMax]);
plot(length(Hellascoords),Hellas_R,'r*');
text((length(Hellascoords) - 3),(HYRMax + 0.5),'Hellas R');
plot([(length(Hellascoords) - 1),(length(Hellascoords) +
1)],[HYRMin,HYRMin],'b-');
plot([(length(Hellascoords) - 1),(length(Hellascoords) +
1)],[HYRMax,HYRMax],'b-');
plot([(length(Hellascoords) - 1),(length(Hellascoords) -
1)],[HYRMin,HYRMax],'b-');
plot([(length(Hellascoords) + 1),(length(Hellascoords) +
1)],[HYRMin,HYRMax],'b-');
plot(length(Tharsiscoords),Tharsis_R,'r*');
text((length(Tharsiscoords) + 3),Tharsis_R,'Tharsis R');
plot([(length(Tharsiscoords) - 1),(length(Tharsiscoords) +
1)],[TYRMin,TYRMin],'b-');
plot([(length(Tharsiscoords) - 1),(length(Tharsiscoords) +
1)],[TYRMax,TYRMax],'b-');
plot([(length(Tharsiscoords) - 1),(length(Tharsiscoords) -
1)],[TYRMin,TYRMax],'b-');
plot([(length(Tharsiscoords) + 1),(length(Tharsiscoords) +
1)],[TYRMin,TYRMax],'b-');
plot(length(Noachiancoords),Noachian_R,'r*');
text(length(Noachiancoords),(NYRMax + 0.5),'Noachian R');
plot([(length(Noachiancoords) - 1),(length(Noachiancoords) +
1)],[NYRMin,NYRMin],'b-');
plot([(length(Noachiancoords) - 1),(length(Noachiancoords) +
1)],[NYRMax,NYRMax],'b-');
plot([(length(Noachiancoords) - 1),(length(Noachiancoords) -
1)],[NYRMin,NYRMax],'b-');
```

```
plot([(length(Noachiancoords) + 1),(length(Noachiancoords) +
1)],[NYRMin,NYRMax],'b-');
hold off;

% An attempt to create a pdf for the figure using strings - fails currently
% The FileName section works since the error shows the filename being
% generated correctly, it errors out with a message that it can't open the
% file at the location specified - and MATLAB should be able to do that.
% The permission at the OS level does not prohibit MATLAB from writing to
% the disk where the MATLAB files currently reside:
% (...\My Documents\JohnMilne\GEOL394\FiguresForThesis)

%f = figure;
%TempFileName = '%sSkewvsKurtosisPlot';
%FileName = sprintf(TempFileName,DataSetName);
%print(f,'-dpdf',FileName);

% Combined plot for R and c values for the Regional datasets

figure('position',[890 50 400 400]);hold on;box on;orient tall;
title('c Values for Regional Datasets');
xlabel('Dataset Size');
ylabel('c Value');
xlim([0 XLimMax]);
ylim([YLimcMin YLimcMax]);
plot(length(Hellascoords),Hellas_c,'r*');
text((length(Hellascoords) - 3),(HYcMax + 0.5),'Hellas c');
plot([(length(Hellascoords) - 1),(length(Hellascoords) +
1)],[HYcMin,HYcMin],'b-');
plot([(length(Hellascoords) - 1),(length(Hellascoords) +
1)],[HYcMax,HYcMax],'b-');
plot([(length(Hellascoords) - 1),(length(Hellascoords) -
1)],[HYcMin,HYcMax],'b-');
plot([(length(Hellascoords) + 1),(length(Hellascoords) +
1)],[HYcMin,HYcMax],'b-');
plot(length(Tharsiscoords),Tharsis_c,'r*');
text((length(Tharsiscoords)+ 3),(Tharsis_c),'Tharsis c');
plot([(length(Tharsiscoords) - 1),(length(Tharsiscoords) +
1)],[TYcMin,TYcMin],'b-');
plot([(length(Tharsiscoords) - 1),(length(Tharsiscoords) +
1)],[TYcMax,TYcMax],'b-');
plot([(length(Tharsiscoords) - 1),(length(Tharsiscoords) -
1)],[TYcMin,TYcMax],'b-');
plot([(length(Tharsiscoords) + 1),(length(Tharsiscoords) +
1)],[TYcMin,TYcMax],'b-');
plot(length(Noachiancoords),Noachian_c,'r*');
text(length(Noachiancoords),(NYcMax + 0.5),'Noachian c');
plot([(length(Noachiancoords) - 1),(length(Noachiancoords) +
1)],[NYcMin,NYcMin],'b-');
plot([(length(Noachiancoords) - 1),(length(Noachiancoords) +
1)],[NYcMax,NYcMax],'b-');
plot([(length(Noachiancoords) - 1),(length(Noachiancoords) -
1)],[NYcMin,NYcMax],'b-');
plot([(length(Noachiancoords) + 1),(length(Noachiancoords) +
1)],[NYcMin,NYcMax],'b-');
hold off;
```

```
% Make a combined plot of the sub-Noachian dataset' R and c tests and their
% standard deviation ranges

MNYRMin = MainNoachianPoissonR - abs(2*MainNoachianPR_std);
MNYRMax = MainNoachianPoissonR + abs(2*MainNoachianPR_std);

MNYcMin = MainNoachianPoissonc - abs(2*MainNoachianPc_std);
MNYcMax = MainNoachianPoissonc + abs(2*MainNoachianPc_std);

SNYRMin = SparseNoachianPoissonR - abs(2*SparseNoachianPR_std);
SNYRMax = SparseNoachianPoissonR + abs(2*SparseNoachianPR_std);

SNYcMin = SparseNoachianPoissonc - abs(2*SparseNoachianPc_std);
SNYcMax = SparseNoachianPoissonc + abs(2*SparseNoachianPc_std);

TNYRMin = TightNoachianPoissonR - abs(2*TightNoachianPR_std);
TNYRMax = TightNoachianPoissonR + abs(2*TightNoachianPR_std);

TNYcMin = TightNoachianPoissonc - abs(2*TightNoachianPc_std);
TNYcMax = TightNoachianPoissonc + abs(2*TightNoachianPc_std);

XLimMax = length(MainNoachian) + (0.5*length(MainNoachian));

TempMaxRMainNoachian = MainNoachian_R + 2*(abs(MainNoachian_R_std));
TempMaxRSparseNoachian = SparseNoachian_R + 2*(abs(SparseNoachian_R_std));
TempMaxRTightNoachian = TightNoachian_R + 2*(abs(TightNoachian_R_std));
TempMinRMainNoachian = MainNoachian_c - 2*(abs(MainNoachian_R_std));
TempMinRSparseNoachian = SparseNoachian_R - 2*(abs(SparseNoachian_R_std));
TempMinRTightNoachian = TightNoachian_R - 2*(abs(TightNoachian_R_std));

YLimRMin =
2*max([TempMinRMainNoachian,TempMinRSparseNoachian,TempMinRTightNoachian]);
YLimRMax =
2*max([TempMaxRMainNoachian,TempMaxRSparseNoachian,TempMaxRTightNoachian]);

TempMaxcMainNoachian = MainNoachian_c + 2*(abs(MainNoachian_c_std));
TempMaxcSparseNoachian = SparseNoachian_c + 2*(abs(SparseNoachian_c_std));
TempMaxcTightNoachian = TightNoachian_c + 2*(abs(TightNoachian_c_std));
TempMincMainNoachian = MainNoachian_c - 2*(abs(MainNoachian_c_std));
TempMincSparseNoachian = SparseNoachian_c - 2*(abs(SparseNoachian_c_std));
TempMincTightNoachian = TightNoachian_c - 2*(abs(TightNoachian_c_std));

YLimcMin =
2*max([TempMincMainNoachian,TempMincSparseNoachian,TempMincTightNoachian]);
YLimcMax =
2*max([TempMaxcMainNoachian,TempMaxcSparseNoachian,TempMaxcTightNoachian]);

figure('position',[470 50 400 400]);hold on;box on;orient tall;
title('R Values for Noachian Subsets');
xlabel('Dataset Size');
ylabel('R Value');
xlim([0 XLimMax]);
ylim([YLimRMin YLimRMax]);
```

```
plot(length(MainNoachian),MainNoachian_R,'r*');
text(length(MainNoachian),(MNYRMax + 0.25),'MainNoachian R');
plot([(length(MainNoachian) - 1),(length(MainNoachian) +
1)],[MNYRMin,MNYRMin],'b-');
plot([(length(MainNoachian) - 1),(length(MainNoachian) +
1)],[MNYRMax,MNYRMax],'b-');
plot([(length(MainNoachian) - 1),(length(MainNoachian) -
1)],[MNYRMin,MNYRMax],'b-');
plot([(length(MainNoachian) + 1),(length(MainNoachian) +
1)],[MNYRMin,MNYRMax],'b-');
plot(length(SparseNoachian),SparseNoachian_R,'r*');
text((length(SparseNoachian) + 2),(((SNYRMax - SNYRMin)/2) +
SNYRMin),'SparseNoachian R');
plot([(length(SparseNoachian) - 1),(length(SparseNoachian) +
1)],[SNYRMin,SNYRMin],'b-');
plot([(length(SparseNoachian) - 1),(length(SparseNoachian) +
1)],[SNYRMax,SNYRMax],'b-');
plot([(length(SparseNoachian) - 1),(length(SparseNoachian) -
1)],[SNYRMin,SNYRMax],'b-');
plot([(length(SparseNoachian) + 1),(length(SparseNoachian) +
1)],[SNYRMin,SNYRMax],'b-');
plot(length(TightNoachian),TightNoachian_R,'r*');
text((length(TightNoachian) - 4),(TNYRMax + 0.2),'TightNoachian R');
plot([(length(TightNoachian) - 1),(length(TightNoachian) +
1)],[TNYRMin,TNYRMin],'b-');
plot([(length(TightNoachian) - 1),(length(TightNoachian) +
1)],[TNYRMax,TNYRMax],'b-');
plot([(length(TightNoachian) - 1),(length(TightNoachian) -
1)],[TNYRMin,TNYRMax],'b-');
plot([(length(TightNoachian) + 1),(length(TightNoachian) +
1)],[TNYRMin,TNYRMax],'b-');
hold off;

figure('position',[890 50 400 400]);hold on;box on;orient tall;
title('c Values for Noachian Subsets');
xlabel('Dataset Size');
ylabel('c Value');
xlim([0 XLimMax]);
ylim([YLimcMin YLimcMax]);
plot(length(MainNoachian),MainNoachian_c,'r*');
text(length(MainNoachian),(MNYcMax + 0.25),'MainNoachian c');
plot([(length(MainNoachian) - 1),(length(MainNoachian) +
1)],[MNYcMin,MNYcMin],'b-');
plot([(length(MainNoachian) - 1),(length(MainNoachian) +
1)],[MNYcMax,MNYcMax],'b-');
plot([(length(MainNoachian) - 1),(length(MainNoachian) -
1)],[MNYcMin,MNYcMax],'b-');
plot([(length(MainNoachian) + 1),(length(MainNoachian) +
1)],[MNYcMin,MNYcMax],'b-');
plot(length(SparseNoachian),SparseNoachian_c,'r*');
text((length(SparseNoachian) + 2),(((SNYcMax - SNYcMin)/2) +
SNYcMin),'SparseNoachian c');
plot([(length(SparseNoachian) - 1),(length(SparseNoachian) +
1)],[SNYcMin,SNYcMin],'b-');
plot([(length(SparseNoachian) - 1),(length(SparseNoachian) +
1)],[SNYcMax,SNYcMax],'b-');
```

```
plot([(length(SparseNoachian) - 1),(length(SparseNoachian) -
1)],[SNYcMin,SNYcMax],'b-');
plot([(length(SparseNoachian) + 1),(length(SparseNoachian) +
1)],[SNYcMin,SNYcMax],'b-');
plot(length(TightNoachian),TightNoachian_c,'r*');
text((length(TightNoachian) - 4),(TNYcMax + 0.2),'TightNoachian c');
plot([(length(TightNoachian) - 1),(length(TightNoachian) +
1)],[TNYcMin,TNYcMin],'b-');
plot([(length(TightNoachian) - 1),(length(TightNoachian) +
1)],[TNYcMax,TNYcMax],'b-');
plot([(length(TightNoachian) - 1),(length(TightNoachian) -
1)],[TNYcMin,TNYcMax],'b-');
plot([(length(TightNoachian) + 1),(length(TightNoachian) +
1)],[TNYcMin,TNYcMax],'b-');
hold off;

% Comparison of the box-and-whisker plots of the big sets.

% Need to make the <region>NNVectors the same length.

HellasPadding = NaN((length(Noachiancoords) - length(Hellascoords)),1);
HellasNNVector = [HellasNNVector;HellasPadding];
TharsisPadding = NaN((length(Noachiancoords) - length(Tharsiscoords)),1);
TharsisNNVector = [TharsisNNVector;TharsisPadding];

Ages = [3,3.5,4];
TempCat = horzcat(TharsisNNVector,HellasNNVector);
BoxPlots = horzcat(TempCat,NoachianNNVector);

figure('position',[1310 50 600 600]);box on;hold on;
boxplot(BoxPlots,Ages);
plot([0.8,1.2],[TharsisMeanNN,TharsisMeanNN],'--','Color',[0 0.3 0.3]);
text(0.85,0,'Tharsis');
plot([1.8,2.2],[HellasMeanNN,HellasMeanNN],'--','Color',[0 0.3 0.3]);
text(1.9,0,'Hellas');
plot([2.8,3.2],[NoachianMeanNN,NoachianMeanNN],'--','Color',[0 0.3 0.3]);
text(2.84,0,'Noachian');
title('Box-Whisker Plot Comparison of Regional Datasets');
xlabel('Age (Ga)');
ylabel('Nearest Neighbor Distance (km)');
hold off;

% Comparison of the box-and-whisker plots of the Noachian subsets

% Again have to make the dataset vectors the same size

SparseNoachianPadding = NaN((length(MainNoachian) -
length(SparseNoachian)),1);
SparseNoachianNNVector = [SparseNoachianNNVector;SparseNoachianPadding];
TightNoachianPadding = NaN((length(MainNoachian) - length(TightNoachian)),1);
TightNoachianNNVector = [TightNoachianNNVector;TightNoachianPadding];

TempCat2 = horzcat(TightNoachianNNVector,MainNoachianNNVector);
BoxPlots2 = horzcat(TempCat2,SparseNoachianNNVector);
```

```
figure('position',[1310 400 600 600]);box on;hold on;
boxplot(BoxPlots2);
plot([0.8,1.2],[TightNoachianMeanNN,TightNoachianMeanNN],'--','Color',[0 0.3
0.3]);
text(0.7,0,'Tight Noachian');
plot([1.8,2.2],[MainNoachianMeanNN,MainNoachianMeanNN],'--','Color',[0 0.3
0.3]);
text(1.7,0,'Main Noachian');
plot([2.8,3.2],[SparseNoachianMeanNN,SparseNoachianMeanNN],'--','Color',[0
0.3 0.3]);
text(2.7,0,'Sparse Noachian');

title('Box-Whisker Plot Comparison of Noachian Subsets');
ylabel('Nearest Neighbor Distance (km)');
hold off;

% Make tables of the interesting statistics

StatisticsTable(HellasMeanNN,HellasNNStd,HellasMedianNN,HellasQuartiles,Hella
sPoissonMedian,HellasPoissonQuartiles,Hellas_R,Hellas_R_std,Hellas_c,Hellas_c
_std,HellasRTest,HellasCTest,HellasSkew,HellasKurt,'Hellas');
StatisticsTable(TharsisMeanNN,TharsisNNStd,TharsisMedianNN,TharsisQuartiles,T
harsisPoissonMedian,TharsisPoissonQuartiles,Tharsis_R,Tharsis_R_std,Tharsis_c
,Tharsis_c_std,TharsisRTest,TharsisCTest,TharsisSkew,TharsisKurt,'Tharsis');
StatisticsTable(NoachianMeanNN,NoachianNNStd,NoachianMedianNN,NoachianQuartil
es,NoachianPoissonMedian,NoachianPoissonQuartiles,Noachian_R,Noachian_R_std,N
oachian_c,Noachian_c_std,NoachianRTest,NoachianCTest,NoachianSkew,NoachianKur
t,'Noachian');
StatisticsTable(MainNoachianMeanNN,MainNoachianNNStd,MainNoachianMedianNN,Mai
nNoachianQuartiles,MainNoachianPoissonMedian,MainNoachianPoissonQuartiles,Mai
nNoachian_R,MainNoachian_R_std,MainNoachian_c,MainNoachian_c_std,MainNoachian
RTest,MainNoachianCTest,MainNoachianSkew,MainNoachianKurt,'MainNoachian');
StatisticsTable(SparseNoachianMeanNN,SparseNoachianNNStd,SparseNoachianMedian
NN,SparseNoachianQuartiles,SparseNoachianPoissonMedian,SparseNoachianPoissonQ
uartiles,SparseNoachian_R,SparseNoachian_R_std,SparseNoachian_c,SparseNoachia
n_c_std,SparseNoachianRTest,SparseNoachianCTest,SparseNoachianSkew,SparseNoac
hianKurt,'SparseNoachian');
StatisticsTable(TightNoachianMeanNN,TightNoachianNNStd,TightNoachianMedianNN,
TightNoachianQuartiles,TightNoachianPoissonMedian,TightNoachianPoissonQuartil
es,TightNoachian_R,TightNoachian_R_std,TightNoachian_c,TightNoachian_c_std,Ti
ghtNoachianRTest,TightNoachianCTest,TightNoachianSkew,TightNoachianKurt,'Tigh
tNoachian');

% Done!  Ta da! :D
```

```
% Nearest neighbor and two-point azimuth calculations using geodesic
% distances for Martian cartesian coordinates (Longitude, Latitude).
% Author: John Milne

function
[NearestNeighborDistances,MeanNN,MedianNN,Quartiles,STDDistances,GeodesicDist
ances,NIndex,GeoAzimuths] = GeodesicCalculation(coords)

% Use the GeoDistances function to get the individual distances and
% azimuths between every point in <coords>.

[GeodesicDistances,GeoAzimuths] = GeoDistances(coords);

% We now have square, upper triangular matrices, GeoDistances and
% GeoAzimuths, that store all of the geodesic distances and azimuths
% respectively between all of the possible combinations of Long/Lat pairs.

% Any given distance measurement is now accessed as a 1x2 index vector.
% So, for example, [3,8] is the set of indices that gives you the
% [row, column] of the calculated distance in the GeoDistances matrix
% between the Lat/Long pair at index 3 and the Lat/Long pair at index 8.

% Now the plan is to determine the nearest neighbor of each Long/Lat pair
% in the dataset.  For a random row in the GeoDistances matrix, the index
% of that row is the row's value, so it is the first number we need.
% The second number we need is the index of the lowest distance (nearest
% neighbor) within the column above the index and the row to the right of
% the index.  Use the NearestNeighbor function to do that:

[NearestNeighborDistances,NIndex] = NearestNeighbor(GeodesicDistances);

% Now that we have the list of nearest neighbor distances, we can do the
% statistics we want on them - specifically get a mean and a standard
% deviation associated with the neareset neighbor distance calculations

MeanNN = mean(NearestNeighborDistances);
STDDistances = std(NearestNeighborDistances);
MedianNN = median(NearestNeighborDistances);

NearestNeighborDistances = sort(NearestNeighborDistances);

Midpoint = floor((length(NearestNeighborDistances))/2);

Quartile25 = median(NearestNeighborDistances(1:Midpoint));
Quartile75 = median(NearestNeighborDistances((Midpoint + 1):end));
Quartiles = [Quartile25,Quartile75];
```

```
function [AllDistances,AllAzimuths] = GeoDistances(CoordinatesMatrix)

% CoordinatesMatrix must be in the form of (Longitude, Latitude), where
% both Longitude and Latitude can themselves be vectors of any, but equal
% lengths.

% GeoDistances calculates the geodesic distance and azimuth (assuming Mars
% is the planet of interest) between all points in CoordinatesMatrix and
% returns indexed, upper triangular matrices of distances and azimuths
% between the individual input points, where the indices of the distance
% measurement (or azimuthal measurement) within the returned matrices are
% the indices of the two points for whom that distance or azimuth was
% measured within the original CoordinatesMatrix.

% Separate the Longitudes and Latitudes into their own vectors

Longitude = CoordinatesMatrix(:,1);
Latitude = CoordinatesMatrix(:,2);

% Need radians vice degrees

CoLatitudeR = (90 - Latitude)*pi/180;
LongitudeR = Longitude*pi/180;

RadiusOfMars = 3396.2;

GeoDistances = NaN(length(CoordinatesMatrix));
GeoAzimuth = GeoDistances;

for i1 = 1:length(CoordinatesMatrix);
    col1 = CoLatitudeR(i1);
    lon1 = LongitudeR(i1);
    for i2 = (i1 + 1):length(CoordinatesMatrix);
        col2 = CoLatitudeR(i2);
        lon2 = LongitudeR(i2);
        % Using the SphereDistance function created for this calculation
        [GeoDistances(i1,i2),GeoAzimuth(i1,i2)] =
(SphereDistance(col1,lon1,col2,lon2));
    end
end

AllDistances = GeoDistances*RadiusOfMars;
AllAzimuths = GeoAzimuth;
```

```
function [Distances,Azimuths] =
SphereDistance(CoLatitude1,Longitude1,CoLatitude2,Longitude2)

% Calculate distance and azimuth from point 1 to point 2.  Distance,
% colatitude and longitude are assumed to be in radians.

Long21 = Longitude2 - Longitude1;

% Have to check if the 360 degree line is being crossed

if ((Longitude1 || Longitude2) > 345) && ((Longitude1) || (Longitude2) < 15)
    if Longitude1 > Longitude2
        Long21 = (360 - Longitude1) + Longitude2;
    else
        Long21 = (360 - Longitude2) + Longitude1;
    end
end

% Haversine formula for geodesic distance calculation

Distances = acos((cos(CoLatitude1).*cos(CoLatitude2)) +
(sin(CoLatitude1).*sin(CoLatitude2).*cos(Long21)));
cosA = ((sin(CoLatitude1).*cos(CoLatitude2)) -
(cos(CoLatitude1).*sin(CoLatitude2).*cos(Long21)))./sin(Distances);
sinA = (sin(CoLatitude2).*sin(Long21))./sin(Distances);

% Also get the azimuths from a one-line calculation

Azimuths = (180/pi)*((pi/2) + atan(sinA/cosA));
```

```
function [NNDistancesVector,IndexVector] = NearestNeighbor(Distances)

% Take the input upper triangular matrix of distance measurements between
% (Lon,Lat) points and determine the nearest neighbors amongst those
% distances assuming each row is the distance from one point to every other
% point in the original list of points, which is itself indexed by row for
% each point.  Return the Nearest Neighbor Distances matrix and the
% associated index that houses the index of where each nearest neighbor
% distance exists within the total list of distances

IndexVector = zeros(length(Distances),1);
NNDistancesVector = IndexVector;
NNDistances = IndexVector;
Index = NNDistances;

for k = 1:length(Distances)
   if k == 1
       TempDistances = Distances(k,k+1:end);
   elseif k == length(Distances)
       TempDistances = Distances(1:k-1,k);
   else
       % Column above first, then row after to complete the new vector
       TempDistances = [Distances(1:k-1,k);transpose(Distances(k,k+1:end))];
   end
   % And we create a new matrix that holds the indexed nearest neighbor
   % distances so we can do statistics on those.
   [NNDistances(k),Index(k)] = min(TempDistances);
   continue
end

NNDistancesVector = NNDistances;
IndexVector = Index;
```

```
function
[PoissonCoordinates,PoissonMeanNN,PoissonSTDNN,PoissonMedian,PoissonQuartiles
,PHull,PoissonSkewness,PoissonKurtosis] = PoissonSets(coords,DataSetSize)

% Now we need to create random spatial distributions that also are also
% bounded by the same set of Long/Lat boundaries as the real dataset.  Each
% new random dataset is then tested against the areal extent of the real
% dataset and only those within 5% of that area are stored in the
% PoissonMetaMatrix as random datasets to be used in further calculations.

Lon = coords(:,1);
Lat = coords(:,2);

[Hull,HullArea] = convhull(coords);

PoissonCounter = 1;
PoissonCoordinatesMatrix = [Lon,Lat];
PoissonMeanNN = zeros(DataSetSize,1);
PoissonMedianNN = zeros(DataSetSize,1);
PoissonQuartilesNN = zeros(DataSetSize,2);
PoissonSTDNN = zeros(DataSetSize,1);
PoissonSkewness = zeros(DataSetSize,1);
PoissonKurtosis = zeros(DataSetSize,1);
PoissonHullBoundary = Hull;

% A function called PoissonStatistics was created to generate a random
% dataset and calculate the required statistics for that dataset.  Then,
% only those random datasets that are within 5% of the areal extent of the
% current dataset get stored in the PoissonMetaMatrix for later use. The
% structure of the PoissonMetaMatrix is a 3D matrix where the x-axis at y=1
% and all z are Longitudes and Latitudes for the input coordinates.  Each
% subsequent x-y plane is then one random set of Long/Lat pairs; thus,
% every z is a new random set of Long/Lat pairs.  The same structure is
% used for storing the means, the standard deviations, the hull and hull
% areas of each random distribution.  The third dimension of
% PoissonMetaMatrix is controlled by the input DataSetSize.

while PoissonCounter < DataSetSize

[PHull,PHullArea,PAverageNNDistance,PoissonSTDNNDistance,PMedian,PQuartiles,P
Skew,PKurtosis,PoissonCoordinates] = PoissonStatistics(coords);
    AreaDiscriminator = (abs(PHullArea -
HullArea))/(max(PHullArea,HullArea));
    if (AreaDiscriminator < 0.05) && (length(Hull) == length(PHull))
        PoissonCounter = PoissonCounter + 1;
        PoissonCoordinatesMatrix =
cat(3,PoissonCoordinatesMatrix,PoissonCoordinates);
        PoissonMeanNN(PoissonCounter) = PAverageNNDistance;
        PoissonSTDNN(PoissonCounter) = PoissonSTDNNDistance;
        PoissonMedianNN(PoissonCounter) = PMedian;
        PoissonQuartilesNN(PoissonCounter,1) = PQuartiles(1,1);
        PoissonQuartilesNN(PoissonCounter,2) = PQuartiles(1,2);
        PoissonSkewness(PoissonCounter) = PSkew;
        PoissonKurtosis(PoissonCounter) = PKurtosis;
        PoissonHullBoundary = cat(3,PoissonHullBoundary,PHull);
        continue
```

```
    end
end
RandomPoisson = (floor((DataSetSize - 1)*rand)) + 1;
PoissonCoordinates = PoissonCoordinatesMatrix(:,:,RandomPoisson);
PHull = PoissonHullBoundary(:,:,RandomPoisson);
PoissonMedian = median(PoissonMedianNN);
PoissonQuartiles =
[mean(PoissonQuartilesNN(:,1)),mean(PoissonQuartilesNN(:,2))];
```

```
function
[PoissonHullBoundary,PoissonHullEnclosedArea,PoissonAverage,PoissonSTD,Poisso
nMedian,PoissonQuartiles,PoissonSkew,PoissonKurtosis,PoissonCoordinates] =
PoissonStatistics(CoordinateMatrixMatch)

% Takes an input of a set of [Lon,Lat] points as a reference and returns
% the coordinates of a random distribution (PoissonCoordinates) of the same
% size as the reference, determines the nearest neighbor distances for each
% point in the random distribution, returns the average, standard
% deviation, standard skewness and standard kurtosis of those nearest
% neighbor distances, calculates a convex hull around the random
% distribution, and returns both the area of the convex hull as well as an
% index matrix used to plot the boundary of the convex hull.

% Specifically designed as a function for the TotalAnalysis.m MATLAB
% program.

Longitude = CoordinateMatrixMatch(:,1);
Latitude = CoordinateMatrixMatch(:,2);
PoissonLat = zeros(length(CoordinateMatrixMatch),1);
PoissonLon = PoissonLat;

% Create the random CoLatitude vector of the same size as the dataset.

counter1 = 1;
CoLat = 90 - Latitude;

while counter1 <= length(CoordinateMatrixMatch)
    TempPoissonLat = 180*rand;
    if ((TempPoissonLat <= max(CoLat))) && (TempPoissonLat >= (min(CoLat)))
        PoissonLat(counter1) = (90 - TempPoissonLat);
        counter1 = counter1 + 1;
    else
        continue
    end
end

% Create the random Longitude vector of the same size as the dataset.

counter2 = 1;

while counter2 <= length(CoordinateMatrixMatch)
    Theta = 360*rand;
    if ((Theta <= max(Longitude)) && (Theta >= min(Longitude)))
        PoissonLon(counter2) = Theta;
        counter2 = counter2 + 1;
    else
        continue
    end
end

% Store the longitudes in latitudes of the new random dataset.

PoissonCoordinates = [PoissonLon,PoissonLat];
```

```
% Use the GeoDistances function to get distances and azimuths to each
% point.

[PoissonDistances,~] = GeoDistances(PoissonCoordinates);

% Use the Nearest Neighbor function to get find the nearest neighbors.

[PoissonNearestNeighbors,~] = NearestNeighbor(PoissonDistances);

% Return the average and standard deviation of the nearest neighbors.

PoissonAverage = mean(PoissonNearestNeighbors);
PoissonSTD = std(PoissonNearestNeighbors);
PoissonMedian = median(PoissonNearestNeighbors);

NNDistances = sort(PoissonNearestNeighbors);

Midpoint = floor((length(NNDistances))/2);

PQuartile25 = median(NNDistances(1:Midpoint));
PQuartile75 = median(NNDistances((Midpoint + 1):end));
PoissonQuartiles = [PQuartile25,PQuartile75];

% Calculate the Skewness for this Poisson distribution

Skew = zeros(length(PoissonNearestNeighbors),1);

for i = 1:length(PoissonNearestNeighbors)
    Skew(i) = ((PoissonNearestNeighbors(i) -
PoissonAverage)^3)/(PoissonSTD^3);
end

PoissonSkew = (sum(Skew)/(length(Skew)))*(sqrt((length(Skew)/6)));

% Calculate the Kurtosis of this Poisson distribution

Kurtosis = zeros(length(PoissonNearestNeighbors),1);

for i = 1:length(PoissonNearestNeighbors)
    Kurtosis(i) = ((PoissonNearestNeighbors(i) -
PoissonAverage)^4)/(PoissonSTD^4);
end

PoissonKurtosis = ((sum(Kurtosis)/(length(Kurtosis))) -
3)*(sqrt((length(Kurtosis)/24)));

% Return the convex hull and its area

[PoissonHullBoundary,PoissonHullEnclosedArea] = convhull(PoissonCoordinates);
```

```
function
[R,R_std,RTest,c,c_std,cTest,MeanPoissonR,STDPoissonR,MeanPoissonc,STDPoisson
c,Skewness,Kurtosis] = RCTests(DataMeanNN,PMeanVector,PSTDVector)

% Nearest neighbor statistics R and c are now calculated.  R = ra/re, where
% ra is the mean of the real dataset and re is the mean of the comparison
% random dataset.  c is (ra - re)/(sigma-e), where sigma-e is the standard
% deviation of the comparison random dataset.

DataSetSize = length(DataMeanNN);

MeanPoisson = mean(PMeanVector);
STDPoisson = mean(PSTDVector);
MeanNN = mean(DataMeanNN);
STDNN = std(DataMeanNN);

R = MeanNN/MeanPoisson;
c = (MeanNN - MeanPoisson)/STDPoisson;

% Skewness and Kurtosis are calculated for the original dataset.

Skew = 0;
Kurt = 0;

for p = 1:DataSetSize
    Skew = Skew + (((DataMeanNN(p) - MeanNN)^3)/(STDNN^3))/DataSetSize;
    Kurt = Kurt + (((DataMeanNN(p) - MeanNN)^4)/(STDNN^4))/DataSetSize;
end

Skewness = Skew*sqrt(DataSetSize/6);
NewKurt = Kurt - 3;
Kurtosis = NewKurt*sqrt(DataSetSize/24);

% Now we need to calculate the standard deviations for R and c, which
% involves the ranges of these values in the random distribution
% population.

PoissonR = zeros(DataSetSize,1);
Poissonc = PoissonR;

for i = 1:DataSetSize
    PoissonR(i) = PMeanVector(i)/MeanPoisson;
    Poissonc(i) = (PMeanVector(i) - MeanPoisson)/STDPoisson;
end

R_std = STDNN/STDPoisson;
MeanPoissonR = mean(PoissonR);
STDPoissonR = std(PoissonR);
MeanPoissonc = mean(Poissonc);
STDPoissonc = std(Poissonc);
c_std = (STDNN - STDPoissonc)/STDPoisson;

% calculating the 2-sigma extremes for the R and c tests
```

```
PlusRRange = MeanPoissonR + abs(2*STDPoissonR);
MinusRRange = MeanPoissonR - abs(2*STDPoissonR);
PluscRange = MeanPoissonc + abs(2*STDPoissonc);
MinuscRange = MeanPoissonc - abs(2*STDPoissonc);

% C-test

cTest = 'true';

if (c >= PluscRange) || (c <= MinuscRange)
    cTest = 'false';
end

% R-test

RTest = 'true';

if (R >= PlusRRange) || (R <= MinusRRange)
    RTest = 'false';
end
```

```
function
Plots(coords,Index,NearestNeighborDistances,R,PoissonR,STDPoissonR,c,Poissonc
,STDPoissonc,Pcoords,PHull,Skewness,Kurtosis,PoisSkew,PoisKurt,DataSetName)

Lon = coords(:,1);
Lat = coords(:,2);
Size = length(coords);

% Nearest Neighbor Plot

figure('position',[10 550 400 400]);hold on;box on;
PlotTitle = '%s Nearest Neighbor Plot';
Title = sprintf(PlotTitle,DataSetName);
for k = 1:Size
    plot([Lon(k),Lon(Index(k))],[Lat(k),Lat(Index(k))],'-k');
    plot(Lon(k),Lat(k),'ok');
end
xlabel('Longitude','FontSize',16);
ylabel('Latitude','FontSize',16);
title(Title,'FontSize',16);
xlim([(min(Lon) - 2) (max(Lon) + 2)]);
ylim([(min(Lat) - 1) (max(Lat) + 1)]);
hold off;

% In order to compare our dataset against a random dataset, we need to get
% an approximate boundary for the dataset for which an area can be
% calculated, which is obtained through the convhull MATLAB command.

[Hull] = convhull(coords);

% Plotting the dataset with its convex hull

PlotTitle = '%s Convex Hull';
Title = sprintf(PlotTitle,DataSetName);
figure('position',[425 550 400 400]);box on;hold on;
plot(Lon(Hull),Lat(Hull),'b-');
plot(Lon,Lat,'r*');
title(Title,'FontSize',16);
xlabel('Longitude','FontSize',16);
ylabel('Latitude','FontSize',16);
hold off;

% Plotting a dataset from the PoissonCoordinatesMatrix

PoissonLongitude = Pcoords(:,1);
PoissonLatitude = Pcoords(:,2);

figure('position',[840 550 400 400]);box on;hold on;
plot(PoissonLongitude(PHull),PoissonLatitude(PHull),'b-');
plot(PoissonLongitude,PoissonLatitude,'r*');
xlabel('Longitude','FontSize',16);
ylabel('Latitude','FontSize',16);
title('Random Poisson Distribution','FontSize',16);
hold off;
```

```
% Plot the histogram over the nearest neighbor plot (we don't really need the
% nearest plot since it's cartesian and we want a plot on a sphere).

figure('position',[10 50 900 900]);box on;
subplot(3,2,[1,2]);
histogram(NearestNeighborDistances,'BinWidth',100);
title(DataSetName,'FontSize',16);
xlabel('Nearest Neighbor Distances (km)','FontSize',16);

% Now we are also doing a box and whisker plot to show the shape of the
% nearest neighbor histogram.

PlotTitle = '%s Box and Whisker';
Title = sprintf(PlotTitle,DataSetName);
subplot(3,2,3);box on;
boxplot(NearestNeighborDistances);hold on;
ylabel('Distance (km)','FontSize',16);
title(Title,'FontSize',16);
hold off;

% Plots of the R and c tests.

% Calculating the 2-sigma extremes for the R and c tests.

PlusRRange = PoissonR + (abs(2*STDPoissonR));
MinusRRange = PoissonR - (abs(2*STDPoissonR));
PluscRange = Poissonc + (abs(2*STDPoissonc));
MinuscRange = Poissonc - (abs(2*STDPoissonc));

XlimMinus = Size - (0.2*Size);
XlimPlus = Size + (0.2*Size);

RYlimMinus = MinusRRange - abs(MinusRRange);
RYlimPlus = 2*abs(PlusRRange);

cYlimMinus = MinuscRange - abs(MinuscRange);
cYlimPlus = 2*abs(PluscRange);

PlotTitle = '%s R Statistic';
Title = sprintf(PlotTitle,DataSetName);
subplot(3,2,4);box on;hold on;
plot([(Size - 0.2),(Size + 0.2)],[PlusRRange,PlusRRange],'b-');
plot([(Size - 0.2),(Size - 0.2)],[MinusRRange,PlusRRange],'b-');
plot([(Size + 0.2),(Size + 0.2)],[MinusRRange,PlusRRange],'b-');
plot([(Size - 0.2),(Size + 0.2)],[MinusRRange,MinusRRange],'b-');
plot(Size,R,'r*');
xlabel('Data Set Size','FontSize',16);
ylabel('R','FontSize',16);
title(Title,'FontSize',16);
xlim([XlimMinus XlimPlus]);ylim([RYlimMinus RYlimPlus]);
hold off;

PlotTitle = '%s c Statistic';
Title = sprintf(PlotTitle,DataSetName);
subplot(3,2,5);box on;hold on;
```

```
plot([(Size - 0.2),(Size + 0.2)],[PluscRange,PluscRange],'b-');
plot([(Size - 0.2),(Size - 0.2)],[MinuscRange,PluscRange],'b-');
plot([(Size + 0.2),(Size + 0.2)],[MinuscRange,PluscRange],'b-');
plot([(Size - 0.2),(Size + 0.2)],[MinuscRange,MinuscRange],'b-');
plot(Size,c,'r*');
xlabel('Data Set Size','FontSize',16);
ylabel('c','FontSize',16);
title(Title,'FontSize',16);
xlim([XlimMinus XlimPlus]);ylim([cYlimMinus cYlimPlus]);
hold off;

% Plot Skewness versus Kurtosis as well as the Skewness and Kurtosis for
% all 1000 random distributions for comparison

PlotTitle = '%s Skew vs. Kurtosis';
Title = sprintf(PlotTitle,DataSetName);
subplot(3,2,6);box on;hold on;
plot(PoisSkew,PoisKurt,'o','MarkerFaceColor','[0.8 0.8
0.8]','MarkerEdgeColor','[0.7 0.7 0.7]');
plot(Skewness,Kurtosis,'Marker','square','MarkerSize',12,'MarkerFaceColor','[
1 0 0]','MarkerEdgeColor','[1 0 0]');
title(Title,'FontSize',16);
xlabel('Skewness','FontSize',16);
ylabel('Kurtosis','FontSize',16);
hold off;
```

```
function
StatisticsTable(Mean_NN,NN_std,Median_NN,Quartiles,PoissonMedian,PQuartiles,R
,R_std,c,c_std,RTest,CTest,Skewness,Kurtosis,DataSetName)

% Function for TotalAnalysis.m - displays the results of TotalAnalysis.m in
% table format

Title = sprintf('    %s Table of Statistics\n',DataSetName);
disp(Title)

Quartile25 = Quartiles(1,1);
Quartile75 = Quartiles(1,2);
PoissonQ25 = PQuartiles(1,1);
PoissonQ75 = PQuartiles(1,2);

Table =
table(Mean_NN,NN_std,Median_NN,Quartile25,Quartile75,PoissonMedian,PoissonQ25
,PoissonQ75,R,R_std,c,c_std,Skewness,Kurtosis);
disp(Table)

RCTests = {CTest,RTest};
VariableNames1 = {'CTest','RTest'};
Table2 = cell2table(RCTests,'VariableNames',VariableNames1);
disp(Table2)
```