

Comparative analysis of prairie dog colony spatial structure

Elizabeth Hasan
University of Colorado, Boulder
Ecology and Evolutionary Biology

April 4th, 2019

Advisor:
Dr. Andrew Martin, Ecology and Evolutionary Biology

Committee Members:
Dr. Barbara Demmig-Adams, Ecology and Evolutionary Biology
Sarah Kelly, MA, Geography

Table of Contents

Abstract	3
Acknowledgements	4
Introduction	5
Hypotheses	7
Methods	10
Results	15
Discussion	35
Conclusions	41
References	43
Code	46

Abstract

Just as food and water are resources for organisms, so is space. The way animals make efficient use of space may be based on resource availability and genetic relatedness among individuals and has the potential to help inform how humans could use space more efficiently. It is unclear if/when animals are using space randomly or according to patterns in spatial organization. My study characterized six prairie dog colonies in Boulder and Gunnison, Colorado, to model spatial dynamics with GPS (global positioning system) data complemented by images of full colonies collected by drones. Spatial burrow data were analyzed for non-random distribution of burrows and clustering, and it was assessed whether there are differences in data collected via GPS or drones. GPS has large spatial error (5-15m), while drones have none, and GPS data take 2-5 times longer to collect. Five out of six colonies had burrows with non-random distributions and exhibited significant levels of clustering distinct from random distributions that display equal levels of clustering and hyper-dispersion (i.e. intentional grouping or dispersion). Clusters were identified with the DBSCAN (density-based spatial clustering of applications with noise) algorithm in R software, and the suitability of this approach for use in future studies on genetic relatedness was evaluated. Network models created based on the spatial burrow data demonstrate how networks can be used to analyze prairie-dog spatial dynamics. In addition, this study served to explore network analysis and develop a protocol for implementation in future studies based on spatial data from individual prairie dogs. Drone surveys are less costly than ground GPS data collection, and both methods yielded similar results. These results inform how prairie dogs use space and provide a foundation to study how spatial organization relates to genetic relatedness and resource availability.

Acknowledgements

I would first like to thank my thesis advisor, Dr. Andrew Martin, for your unwavering support throughout this project. Thank you for spending so much of your invaluable time and energy helping me learn new concepts, write code, and become a better scientist. Your outlook on the world is inspiring and motivates me to never stop exploring the curiosities of the world. I have grown so much throughout my time working with you over the past two years, and I am endlessly grateful for your mentorship.

Thank you to Sean Streich, who made data collection for this project possible. It has been a pleasure to work with you the past two years, and I appreciate the guidance and support you have given me.

I would also like to thank Dr. Barbara Demmig-Adams for helping me edit my paper, and Sarah Kelly for advising me on spatial data analysis.

In addition, thank you to Dr. Eric Vance and Michael Ramsey for your statistical guidance through the Laboratory for Interdisciplinary Statistical Analysis (LISA).

Finally, thank you to my family and friends who have supported me and listened while I talked about my project and practiced presentations.

Introduction

There is a long history of studies focused on understanding the mechanisms underlying the spatial distribution of individuals in a population. Early studies focused on an “ideal free” distribution. In an ideal free distribution, individuals distribute themselves across space such that all individuals obtain the same amount of resources (Fretwel & Lucas 1970; Sutherland 1983). There are two key assumptions inherent in ideal free distributions: individuals move freely across space and density is proportional to the available resources. Fretwell and Lucas (1970) derived a simple model, $\beta = \alpha c^{1/m}$, where β is the fraction of individuals in a population found in each habitat, α is the fraction of resources occurring in each habitat, c is a constant, and m is the strength of interference competition between individuals. The ideal free distribution represents a null hypothesis because the underlying parameters governing the distribution of individuals are very general.

The situation is more complex when individuals do not move freely because of territoriality, or when individuals secure a permanent residence and movement is limited to a home range. In this case, the ideal free distribution may still apply. If individuals behave according to the underlying rules governing an ideal free distribution, and if resources are uniformly distributed, we should expect the area of an individual’s home range to be normally distributed and the differences among individuals to simply reflect stochastic, ecologically-neutral variation. Moreover, if resources are uniformly distributed, the centers of the home ranges of individuals are expected to be evenly distributed across space approximating a hexagonal array (e.g., honey combs).

A variety of factors may influence the distribution of individuals across space. For instance, there may be variation in competitive ability among individuals with effects on home

range size. Similarly, for social organisms home range size may vary as a consequence of relatedness if more related individuals interact cooperatively, and the interactions of less related individuals may be more competitive (Rayor 1988; Verdolin et al. 2014). Additionally, resources may not be uniformly distributed (Travis et al. 1995). Finally, home range size may vary depending on how long a population has occupied a particular defined habitat patch. The packing of home ranges within a defined landscape is likely to increase over time depending on the balance between infilling and growth at the edges of the distribution.

Given the inherent complexity of the distribution of individual home ranges, I decided to explore patterns of habitat occupancy. I focused on trying to estimate several parameters for a social organism -the prairie dog- that occupies defined habitat patches across the landscape. Prairie dogs are fossorial (burrowing) rodents that construct elaborate burrow system; and the locations of burrow systems are marked by burrow entrances (holes) distributed across a bounded geographic space. Prairie dogs form coterries (social groups) that “cooperatively use and defend a common territory” within the overall colony (Rayor 1988). Prairie dogs are essential to the maintenance of grassland biodiversity and ecosystem services, including groundwater recharge, regulation of soil erosion and of soil productive potential, soil carbon storage, and forage availability (Martínez-Estévez et al. 2013).

I focused on the burrow as the unit of measurement that is, in many ways, analogous to a house, which is similar to the method in Messier et al. (1990) of using muskrat dwellings to determine their distribution pattern among habitats. I estimated three parameters, i.e., the home range defined by each burrow entrance, the density of burrows across space, and the spatial distribution of burrows. A similar study by Rayor (1988) used observed space use to study the social interactions among and between coterries. Here, I will expand on the knowledge of prairie

dog space use with a more quantitative approach by using technology and various methods of spatial analysis. I was particularly interested in comparing ground estimates of the location of individual burrows determined via handheld GPS devices and aerial photographs obtained using drones. I was motivated by several hypotheses and objectives:

H₁: Processes controlling spatial organization exist that result in a non-random distribution of burrows.

H₂: Individuals are dependent on each other and organize burrows in clusters.

H₃: Individuals are independent of each other and organized burrows in a hyper-dispersed pattern.

H₄: The distribution of burrows across space in colonies is dependent on factors, such as age of a colony and resource availability, that result in variation in the distribution of burrows.

H₅: Different spatial surveying methods (GPS or drone) will produce the same, or similar, results.

Objective 1: Create network analysis models based on spatial data to demonstrate how network data can be used to model prairie dog colonies.

Objective 2: Identify communities within network models to demonstrate how we can model communities from network data. These communities can be compared with future genetic data testing for relatedness of individuals to assess whether more related individuals engage more often with each other than with less related individuals.

I addressed these questions by first defining the ideal free distribution of competitors between habitat patches- if each individual freely chose a patch to maximize food intake rate, burrows

should be distributed at equilibrium, causing the number of individuals to be proportional to the patch quality and equalizing expected food intake in all occupied patches.

The goal of this study was to use prairie dogs as a model organism to test the methodology used to collect and analyze data on spatial organization, and each research question was tested using multiple spatial statistics. In addition to testing the ideal free distribution using a variety of spatial statistics, this study modeled the use of network analysis to capture the associations of individuals as well as their use of space. A network can be defined as a set of nodes (points) connected by edges (Croft et al. 2004). Social network analysis aims to study the connections and patterns among individuals and groups (Scott 2017). Interactions among animals aid in survival and reproduction to pass genes down through generations (Rosenberg and Arp 2009). Social structure, including hierarchies and disease transmission, affects individuals' fitness and, in turn, evolution (Krause et al. 2007). Social network analysis can be used to study various aspects of animal behavior, and can inform animal conservation (Snijders et al. 2017), welfare and release (Kleinhappel et al. 2016), and shed light on resource management, migration patterns (Bastille-Rousseau et al. 2018), and disease transmission (Perkins et al. 2009) through a better understanding of how individuals in a population are connected. The social networks of prairie dogs have been studied by means of behavioral interactions (Manno et al. 2007; Verdolin et al. 2014), and prairie dogs have been used to model how social structures vary with varying environmental conditions, such as food availability (Travis et al. 1995). In addition, the physical arrangement of burrows within a colony may be another indicator of resource management.

Testing methods for spatial analysis should focus on the method used for data collection. The data collection approach in the present study compares manual identification of burrows with the use of aerial drone imagery, and assesses the utility of each technology for data

collection in ecology. The findings of this study are hoped to provide helpful insight for the cross-disciplinary approach of combining technology, photography, geography, and biology. I tested the advantages and disadvantages of GPS-based versus drone-based technology at a small scale in colonies with hundreds of burrows over a relatively small area. Land management entities traditionally use GPS field surveys, which are costly in terms of time, labor, and financial resources, delaying conservation and management, specifically of endangered species. Use of drones can significantly reduce these costs and increase the efficiency of wildlife surveys (Marvin et al. 2016). In addition, GPS has a large range of locational accuracy, which can greatly impact data on a small scale, e.g., in prairie dog colonies (Marvin et al. 2016). While many studies aim to increase the scale of spatial analysis to monitor animal movement (Chetkiewicz et al 2006; Mueller et al. 2014), the present study performed small-scale spatial analysis with many data points collected via two different methods within a defined location. Furthermore, my study is novel in combining multiple spatial statistics, which had previously only been done in studies of materials science and bioinformatics (Chen et al. 2011; Zhang 2018), to study a biological system.

Methods

Site Selection

Sites were selected by availability and accessibility. Three prairie dog colonies in Gunnison, Colorado, were selected based on size and locational availability of these colonies. These colonies were previously surveyed in Gunnison prairie dog (*Cynomys gunnisoni*) population genetics studies and the locations were known. Three prairie dog colonies in Boulder, Colorado, were selected due to accessibility and surveying permits. These colonies are conveniently located near the University of Colorado and offered the opportunity to extend this

study to a second prairie dog species, the Black-tailed prairie dog (*Cynomys ludovicianus*). Permission from land management entities was received prior to GPS and drone surveying.

Data Collection

All data were collected during September-October of 2018.

GPS Data

Locations of burrows in colonies were collected using Garmin GPSMap 64S GPS units at three colonies in Gunnison (KM, BLM-18, and MR) and three colonies in Boulder (Short, Waldorf, and Superior). All burrows were within colony extents. At each burrow, the GPS location was marked, and flags were placed at each burrow in small colonies to indicate that it had been marked. GPS units are typically accurate to 5-10 meters and 95% accurate to 15 meters.

Drone Imagery

A DJI Phantom 4 pro version 2.0 drone was flown at two colonies in Gunnison (KM and BLM-18) and two colonies in Boulder (Short and Waldorf). Permits were received prior to drone flights. Drone flights were pre-programmed to the colony size that covered the entire colony by traversing a zig-zag pattern. Photographs of each colony were stitched together using Precision Mapper software and downloaded as a georeferenced image. Burrows were visually identified from drone imagery using QGIS version 3.4.2.

Data Analysis

Voronoi Tessellation

Voronoi Tessellation creates polygons, or tiles, around each point that maximize the area each point can take up in space. These models were created using the deldir package in R version 1.1.463. Tessellation models were created for six colonies (KM, BLM-18, MR, Short, Waldorf, and Superior). The area of each tile can be extracted from the tess analysis. Larger tile

areas represent more area per point (i.e. dispersion) and smaller tile areas represent less area per point (i.e. clustering). Histograms of the tile area distributions were plotted for each colony.

Cluster Analysis

Cluster analysis was performed on all six colonies and used drone data to maximize spatial accuracy if available and GPS data if it was not. Coordinates were imported to R version 1.1.463 and were analyzed using the dbscan package. The DBSCAN (density-based spatial clustering of applications with noise) clustering algorithm is a density-based algorithm that groups points in the same cluster if they are within a certain distance (epsilon) of each other and considers points, noise points, if they fall outside of this distance from other points. The epsilon value for each colony was found by plotting 10 k nearest neighbor values for every point, which resulted in an exponential curve. The epsilon value is located at the knee (i.e. steep curve) (Fig. 1). Sensitivity analysis was used to determine that epsilon values slightly above or below the knee of the curve did not affect the cluster output. Cluster plots were created for each colony to visualize clusters identified by the DBSCAN algorithm.

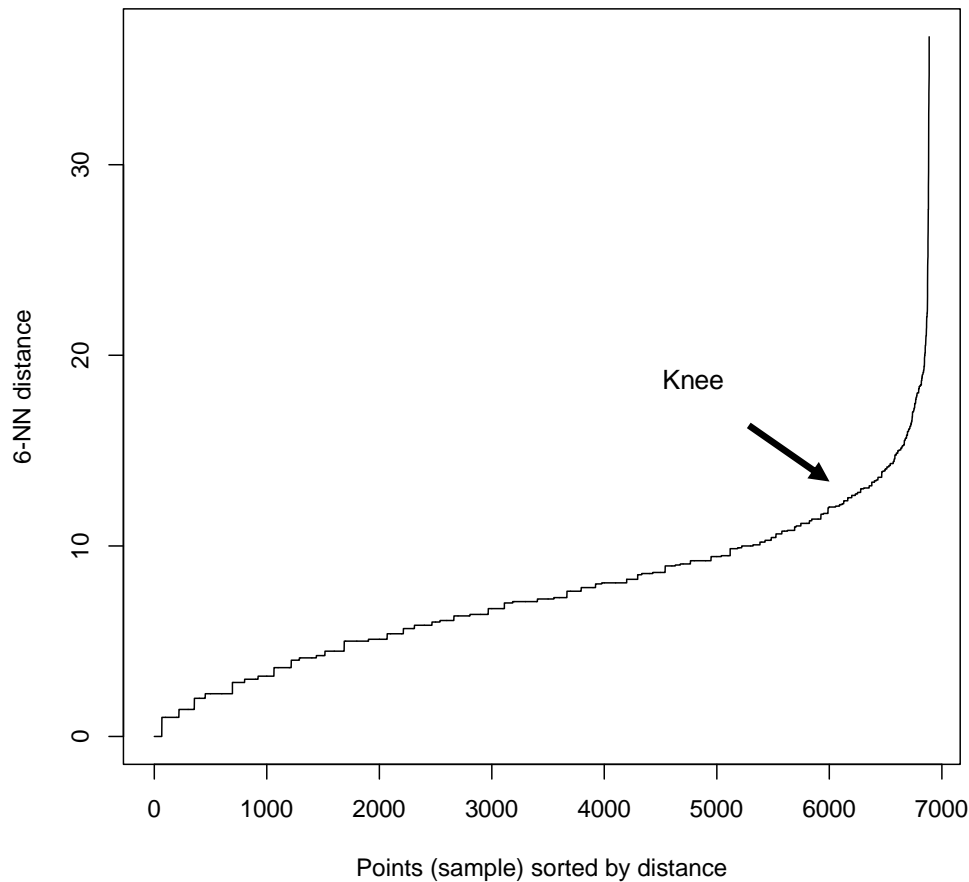


Figure 1: Example of k nearest neighbor graph used to find epsilon. Epsilon is located at the knee of the curve, indicated by an arrow.

Coefficient of Dispersion

Coefficient of dispersion was used to determine whether burrows displayed a random or non-random distribution and displayed a significant effect of clumping or dispersion. The distribution of coefficient of dispersion was calculated for each colony in R version 1.1.463 by randomly selecting 40 burrows in each colony, finding the nearest burrow to each randomly selected point, calculating distances between all points, calculating coefficient of dispersion, then repeating this process 1000 times. The equation for calculating coefficient of dispersion is $CD = \text{variance}/\text{mean}$.

Kernel Density

Kernel density analysis was used to compare the density of burrows between survey methods. Both drone surveys and GPS data collection were performed at Waldorf, Short, KM, and BLM-18. Kernel density identifies the density of points across the colony and creates a raster showing where density is higher and lower than average. The density search radius was set to 15 meters to account for GPS spatial error. Raster outputs for both survey methods for each colony were then overlaid and subtracted. Raster layers contain cells that each have a value; each raster cell contains a density value. Raster subtraction takes the raster values of each cell and subtracts the second input raster from the first input raster to calculate the difference between the layers. The subtraction output is a new raster that contains the values for the difference between input layers. Based on the difference between layers, the similarity of methods in identifying burrows can be determined as well as where in the colony density varies between methods. The output cell size was set to one meter to visualize the density at each burrow as burrows are approximately 1x1 meter.

There are two sources of error in the spatial accuracy of burrow location, which in turn affects the density output. GPS error ranges from 5-15 meters. This causes points to be disassociated with the true burrow location and can shift burrow density. The error is not in a specific direction, so GPS error may cause false clustering or dispersion of points. The second source of error is missing burrows from drone imagery. Most burrows are easily identified from the high-resolution drone imagery. However, Gunnison prairie dogs often burrow under rocks and sage brush. If a burrow is under a rock or sage brush, it may not be visible from above and can be missed in the burrow identification process.

Network Analysis

Network analysis was performed on all six colonies to create network models. This analysis was conducted using spatial data of burrow locations instead of network data from individual prairie dogs and their connections to others. These network models were created to demonstrate the statistics that can be extracted from network analysis and to streamline the data analysis process once network data are obtained.

The `deldir` package in R version 1.1.463 contains a `triang` function that triangulates all points in a colony. Triangulation data were extracted for six colonies and edited to create edge lists. Each focal point was listed with all of its connecting points. These edge lists were then loaded back into R version 1.1.463 for use in network analysis. The `igraph` package in R version 1.1.463 was used to create a model of nodes (burrows) and edges (hypothetical connections by proximity) to demonstrate the network that is formed by connectivity of burrows.

In a true network model, network analysis takes individuals and their connections to each other to create a network of interactions amongst individuals. Degree and Eigen centrality can be used to analyze the how many connections individuals have and how connected individuals are in the overall network.

Communities can be identified within the networks using a clustering method that takes into account the Eigen centrality of individuals. This algorithm identifies individuals that are more connected to individuals within the community than individuals outside the community. Models were created of community outputs using spatial data on burrow location.

The process of learning about network theory and network analysis was extensive. Creating network models of prairie dog colony spatial structure was the original premise of this study for the majority of the study period. Learning this process required literature review on

network theory to learn why and how this modeling technique is used. Learning how to write and run network analysis code in R version 1.1.463 required dedication and patience. The first step was to create Voronoi tessellation models in order to extract edge lists. Edge lists were first created by hand from the tessellation results before it was discovered that triangulation could create edge lists that only required slight modification in Excel. This study has been a great learning experience in new concepts, finding new R packages, being patient while learning and writing code, and understanding sources of error for different analyses.

Results

Voronoi Tessellation

Voronoi tessellation was used to create polygons around every burrow in each prairie dog colony. Voronoi tessellation maximizes the amount of area each point can take up in space. Figure 2 shows tessellation models from six colonies. The distribution of tile areas were plotted for every colony (Fig. 3). All colonies demonstrated an approximately normal distribution with means slightly greater than the mode. Small areas are representative of dense clustering, while large areas are representative of dispersion. Normal distributions of tile areas demonstrate that there is both clustering and dispersion occurring across colonies. Tile areas were used to estimate the variation in home ranges, or ecological footprints, of burrows.

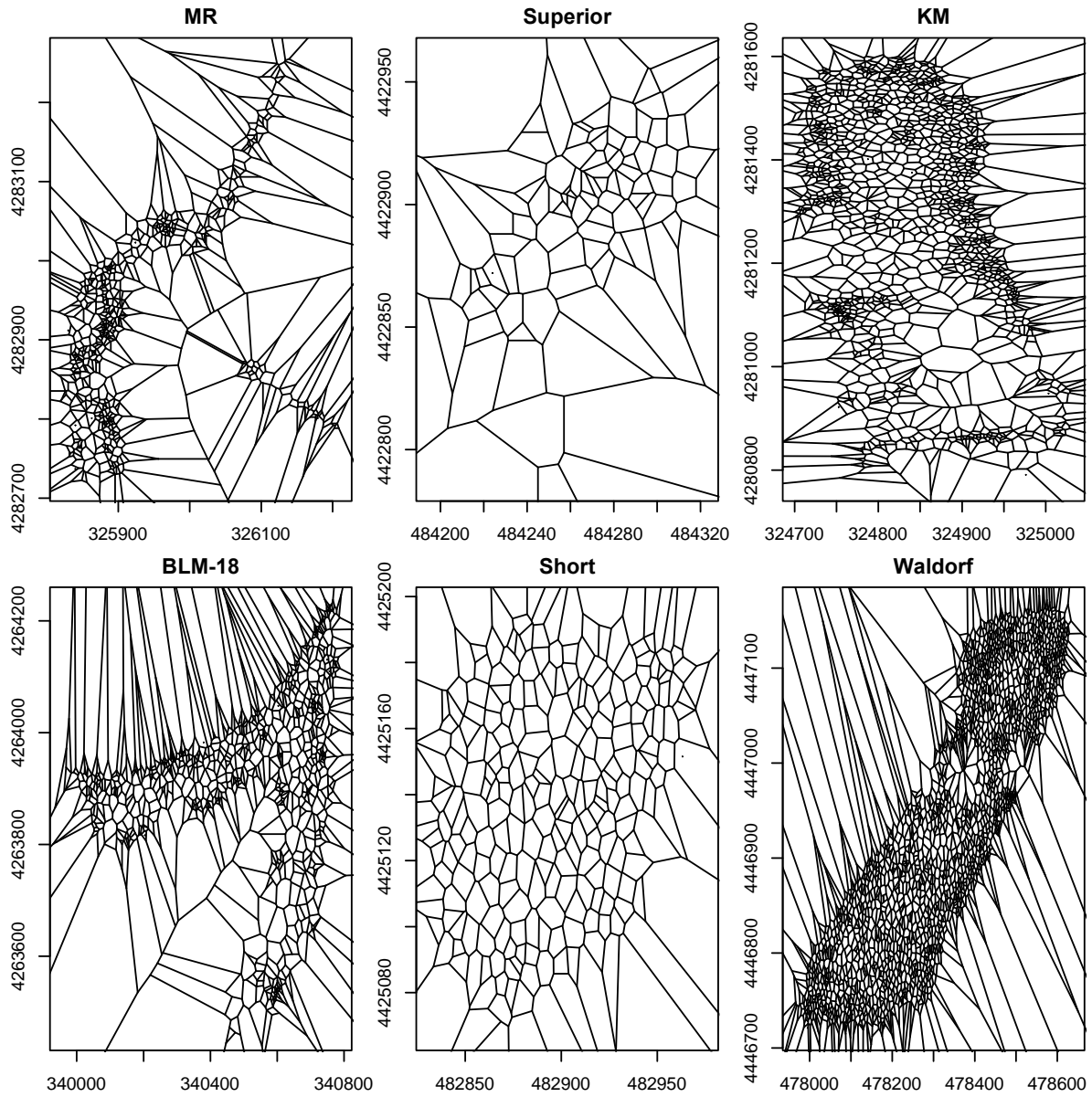


Figure 2: Voronoi Tessellation plots of burrow locations in six prairie dog colonies in Colorado. Tiles (i.e. polygons) around each point (burrow) where any point in the polygon is closer to that center point than to other center points. Coordinates are based on UTM Zone 13. Spatial scale varies between figures.

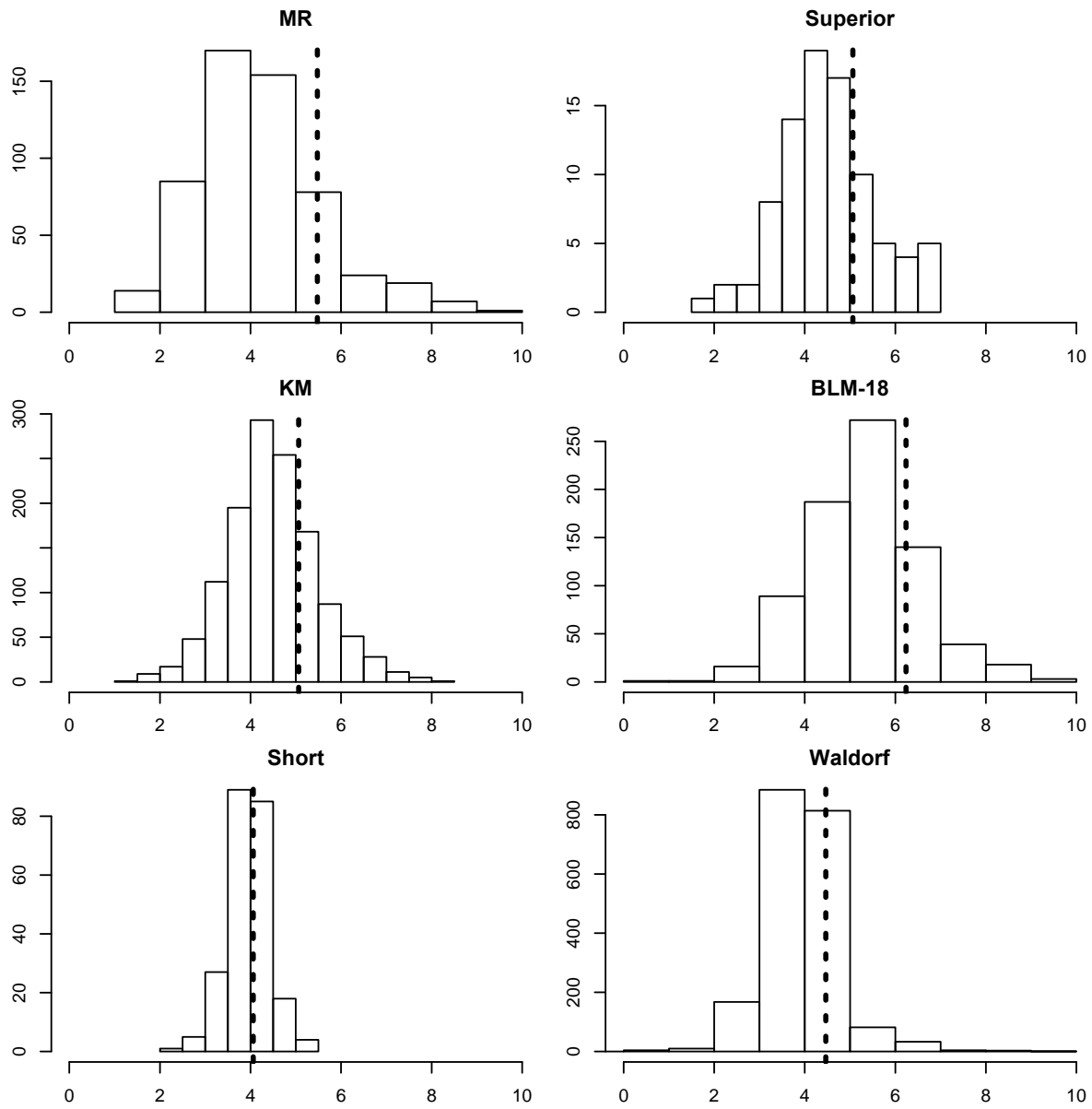


Figure 3: Histograms of tile area distribution generated from Voronoi Tessellation outputs for each colony. Large tile areas demonstrate a greater distance to the surrounding burrows, while small tile areas demonstrate a smaller distance to the surrounding burrows. Dashed lines are the mean tile area for each colony.

Coefficient of Dispersion

The coefficient of dispersion was used to determine if dispersion of burrows in colonies followed a non-random distribution. Five out of the six colonies displayed non-random distribution of points (Fig. 5). The assessment of non-random distribution of points was based on effect size plots of the coefficient of dispersion for each colony (Fig. 5). MR, Superior, KM, BLM-18, and Waldorf all had effect sizes greater than zero and a 95% confidence interval that did not cross zero, thus demonstrating a significant effect. The coefficient of dispersion for Short had a negative effect size and the 95% confidence interval crosses zero, demonstrating that there was no significant effect. Five out of the six colonies displayed significant levels of clustering (Fig. 4). Clustering levels were determined by histograms of the distribution of coefficient of dispersion. One colony, Short, displayed hyper-dispersion, but at an insignificant level indicated by an effect size where the 95% confidence interval crosses zero.

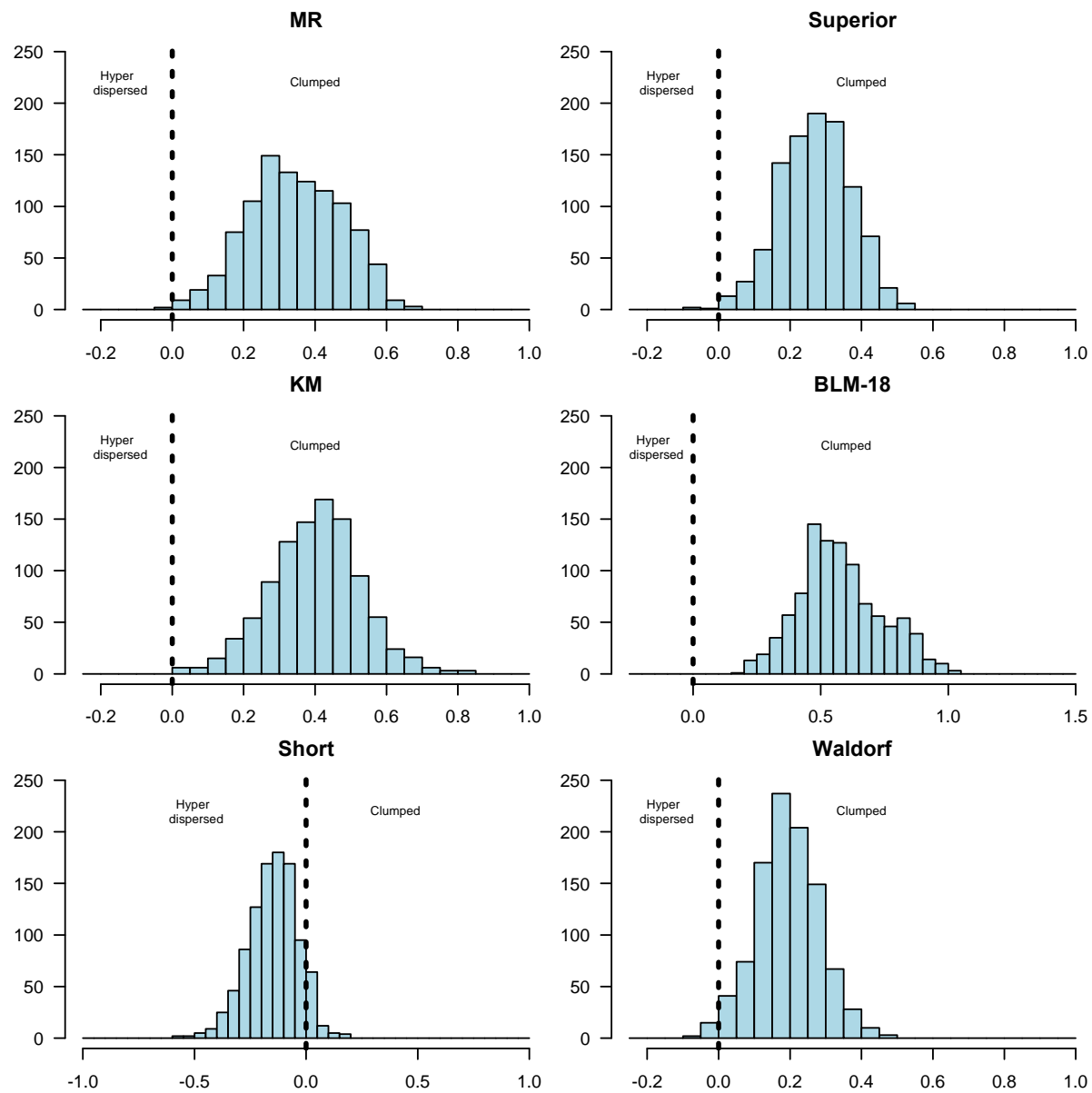


Figure 4: Distribution of distance between points in each colony for 40 randomly chosen points repeated 1000 times. Dashed lines are at 0. Negative values represent hyper-dispersion and positive values represent clumping. Coefficient of dispersion values are log transformed.

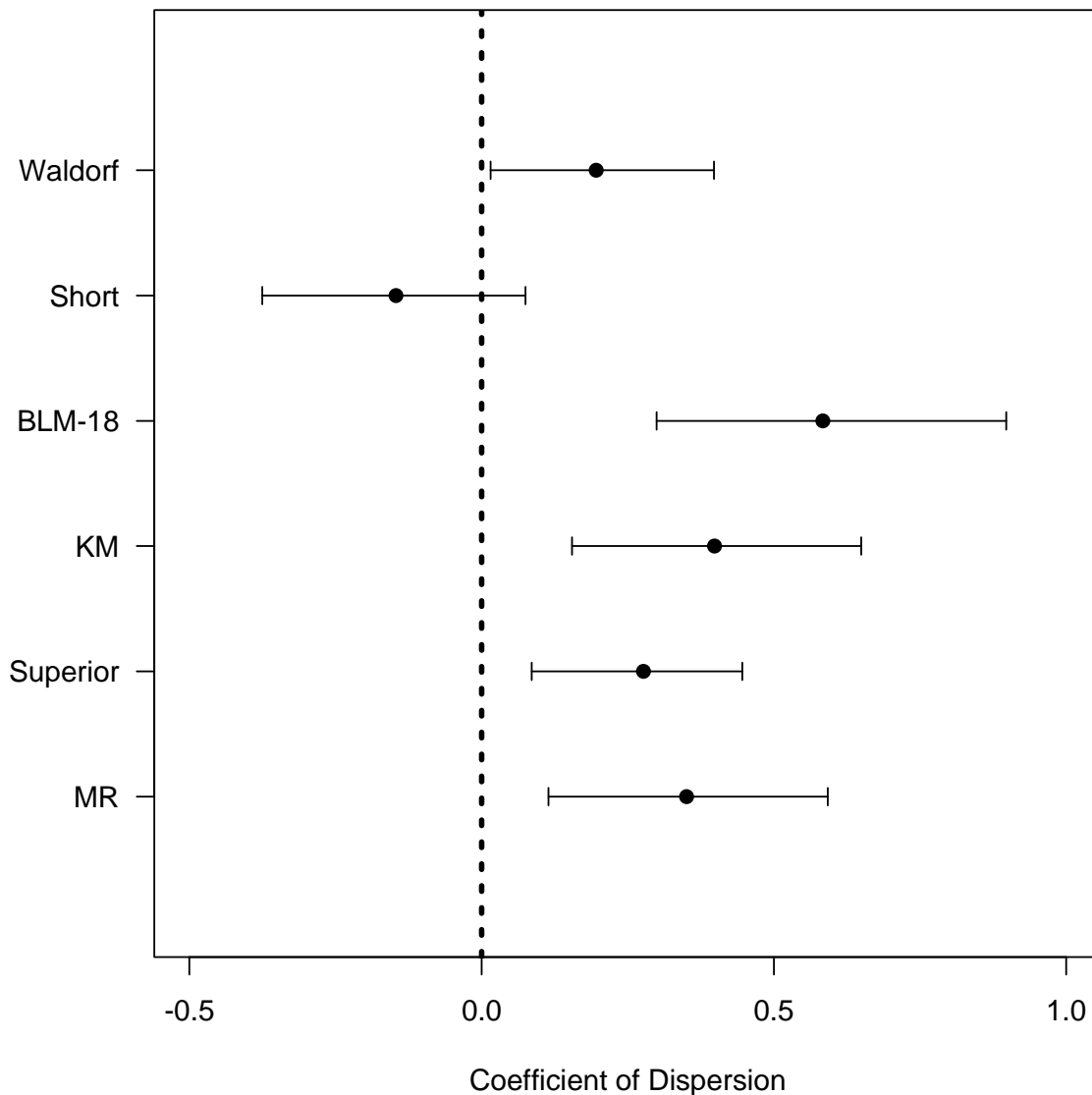


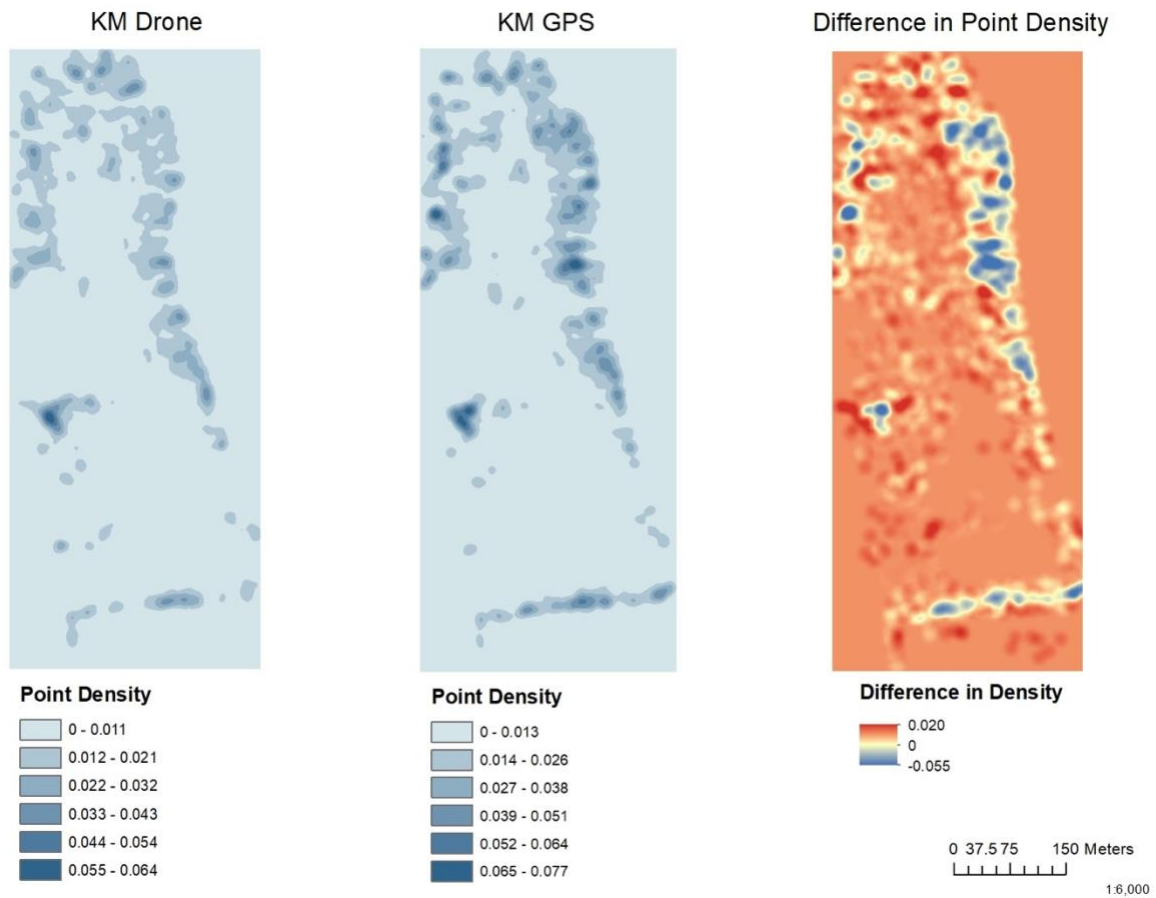
Figure 5: Effect size plot for coefficient of dispersion. Points indicate effect size (mean) of coefficient of dispersion and bars indicate 95% confidence interval. Dashed line is at zero. Coefficient of dispersion values are log transformed.

Kernel Density

Kernel density analysis was used to determine if sampling methods (GPS and drone imagery) deliver equivalent results. Four colonies, where both GPS and drone imagery were collected, were analyzed using kernel density. The individual kernel density raster layers

produced with GPS data and drone imagery for each colony resulted in similar-looking density patterns. However, the final raster layers for difference in density between sampling methods demonstrate that there is significant variation in point density between the sampling methods. (Fig. 6).

6A

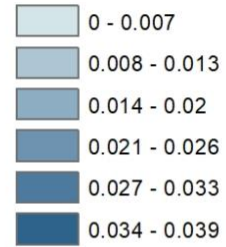


6B

BLM-18 Drone



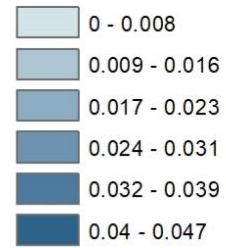
Point Density



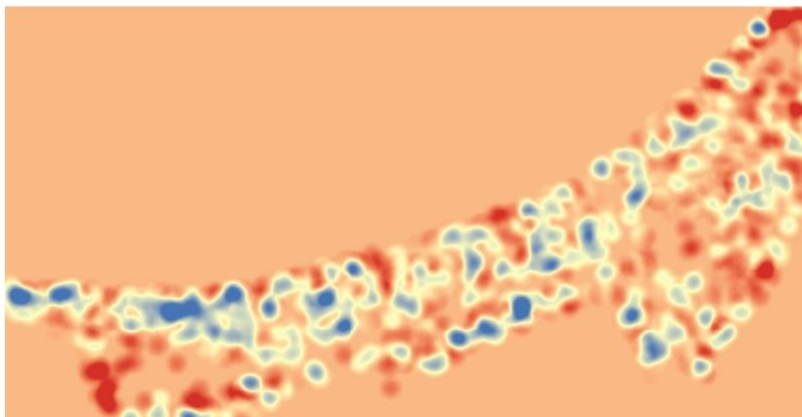
BLM-18 GPS



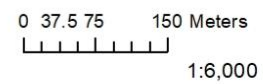
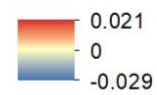
Point Density



Difference in Point Density

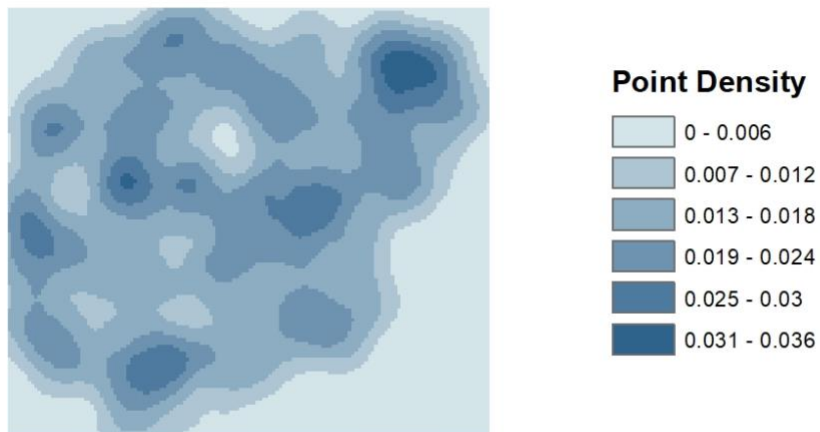


Difference in Density

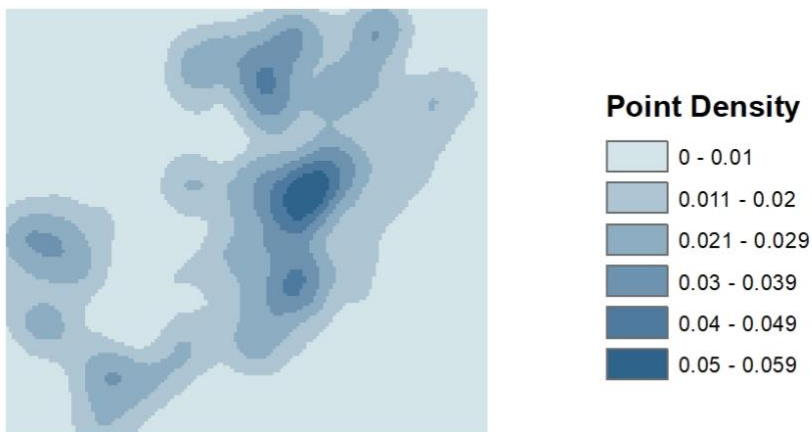


6C

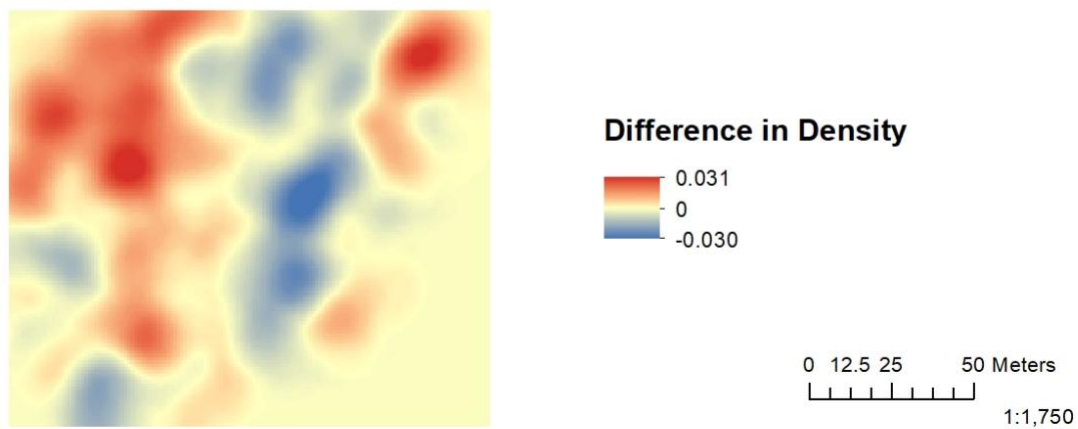
Short Drone



Short GPS



Difference in Point Density



6D

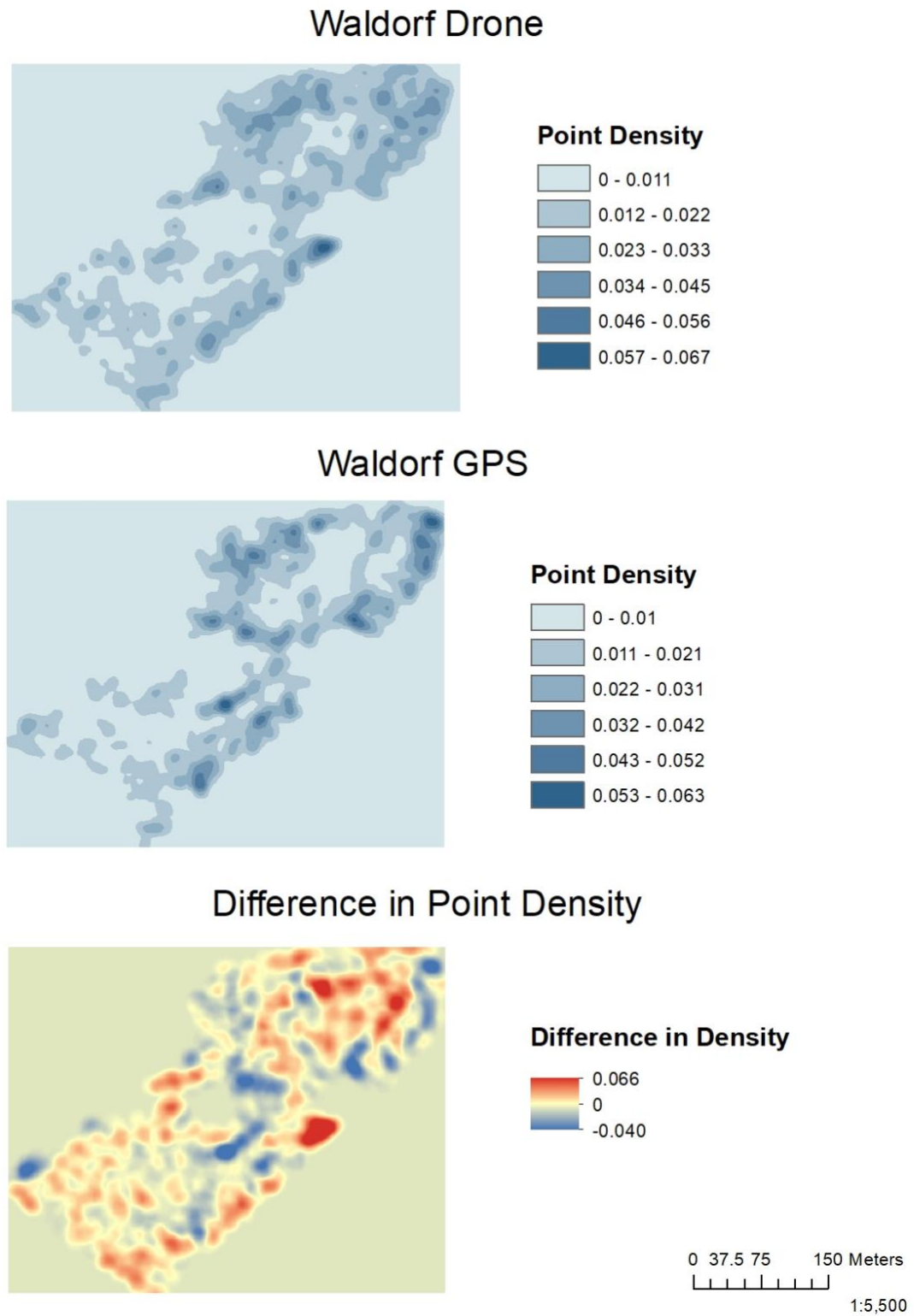


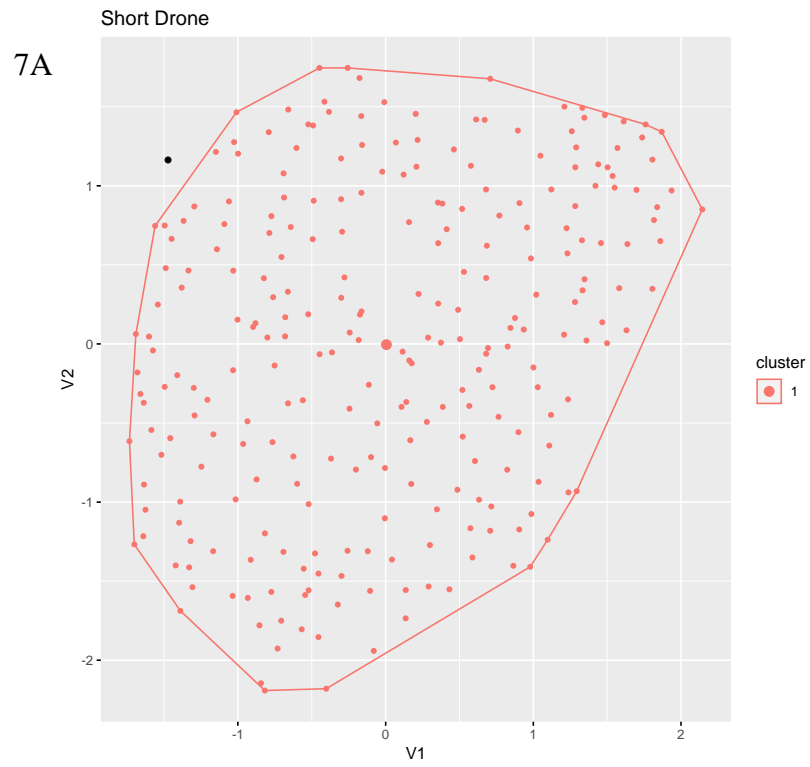
Figure 6: Raster layer outputs from kernel density analysis (blue) and raster subtraction (blue-red). Negative values from raster subtraction are displayed in blue and positive values are

displayed in red. Values of zero (no difference between layers) are displayed in yellow. Raster outputs from GPS data were subtracted from outputs from drone data to create final difference in point density raster layer. Values represent burrow density per square meter.

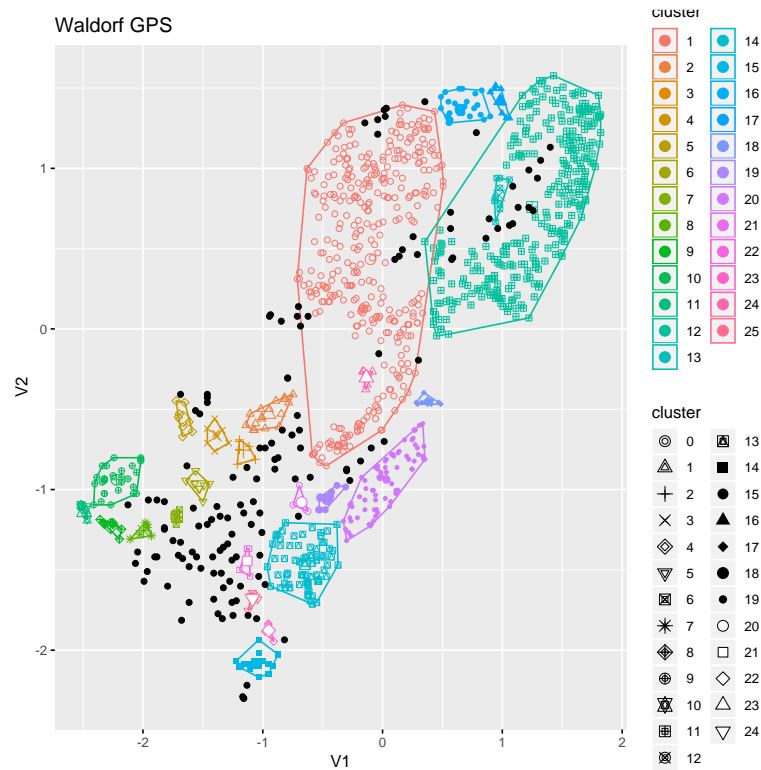
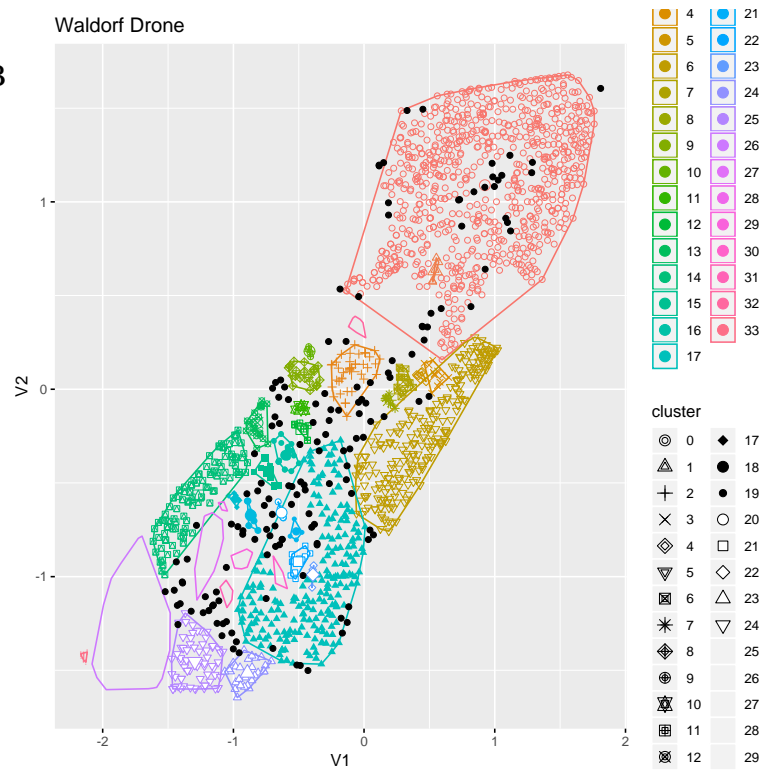
Clusters

Clusters of burrows were identified for six colonies using the DBSCAN method (Fig. 7). Four of these colonies (Short, Waldorf, BLM-18, and KM) were surveyed using GPS data collection and drone imagery. The BLM-18 drone imagery covers a larger extent than the GPS survey; data from drone imagery were constrained to the extent of the GPS data in the cluster figure, but clustering analysis data from the full extent are contained in Table 1. Clusters were generated for both survey methods to compare the clusters identified based on sampling method. Only GPS data were collected at MR and Superior. Number of clusters, number of noise points, and number of burrows for each colony and survey method are listed in Table 1.

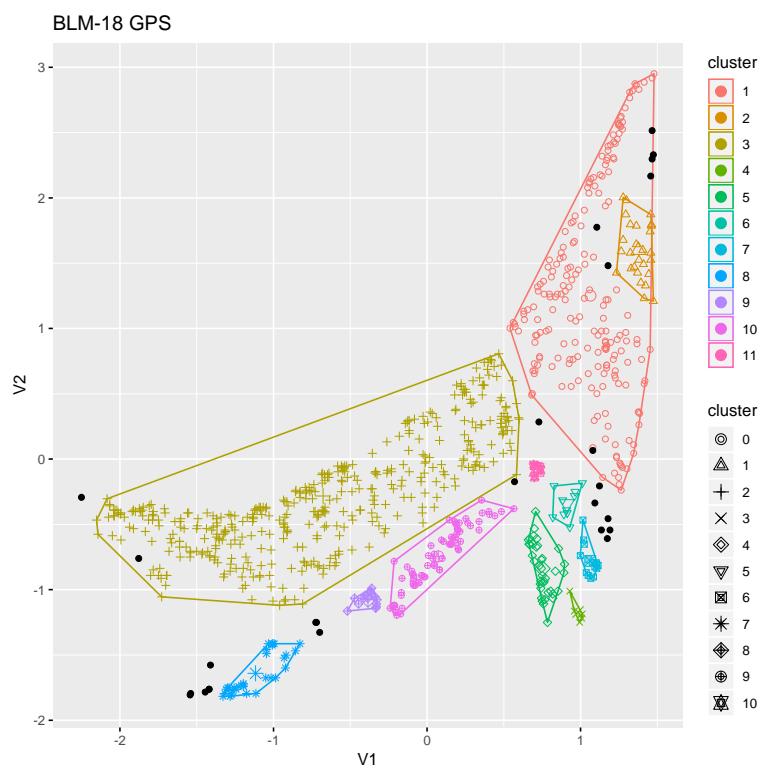
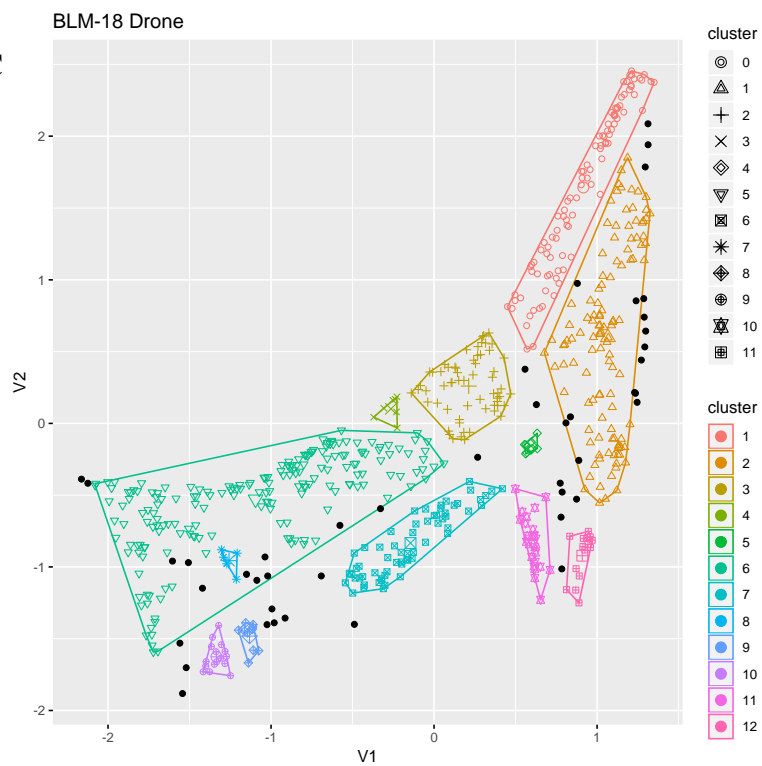
Number of clusters were similar across sampling methods for each colony. Short contained one main cluster identified in the drone imagery, while the GPS data contained two additional clusters. Clusters in Waldorf had a large amount of visual variation between GPS data and drone imagery. GPS data and drone data in the BLM-18 and KM colonies were similar. BLM-18 has two large clusters from GPS data that are each broken up into two clusters in drone imagery. KM has three main clusters that vary slightly between surveying methods.



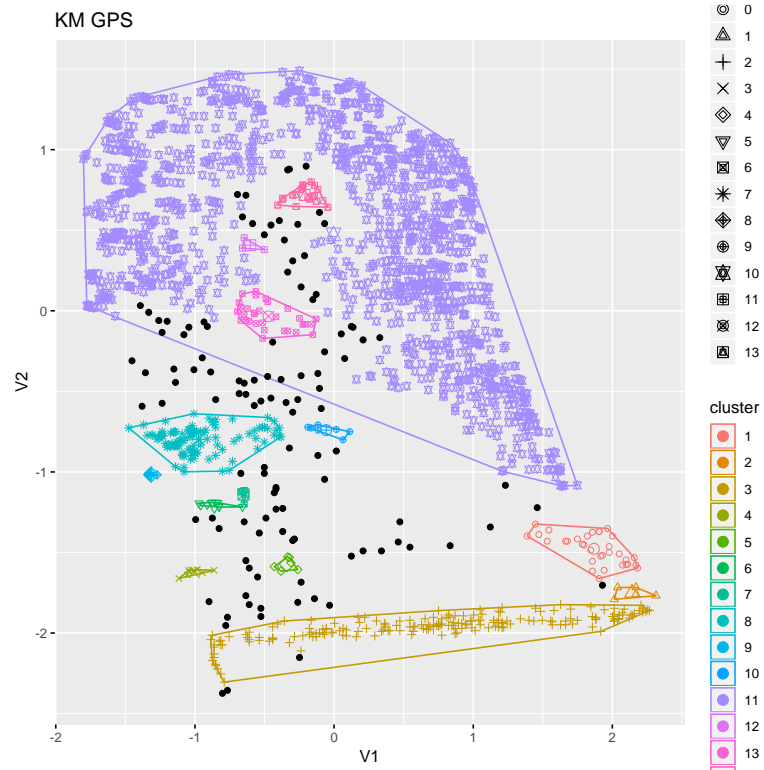
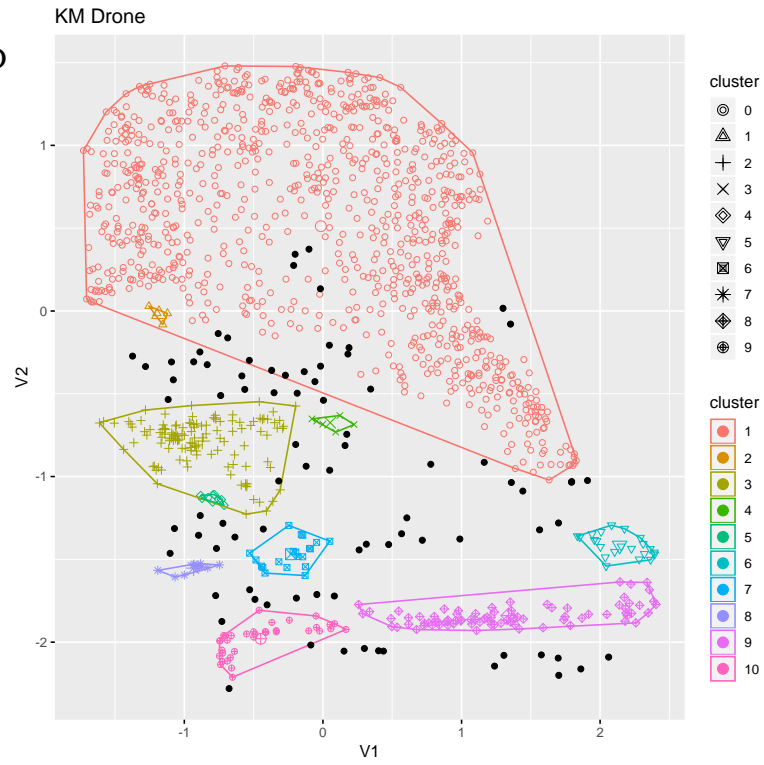
7B



7C



7D



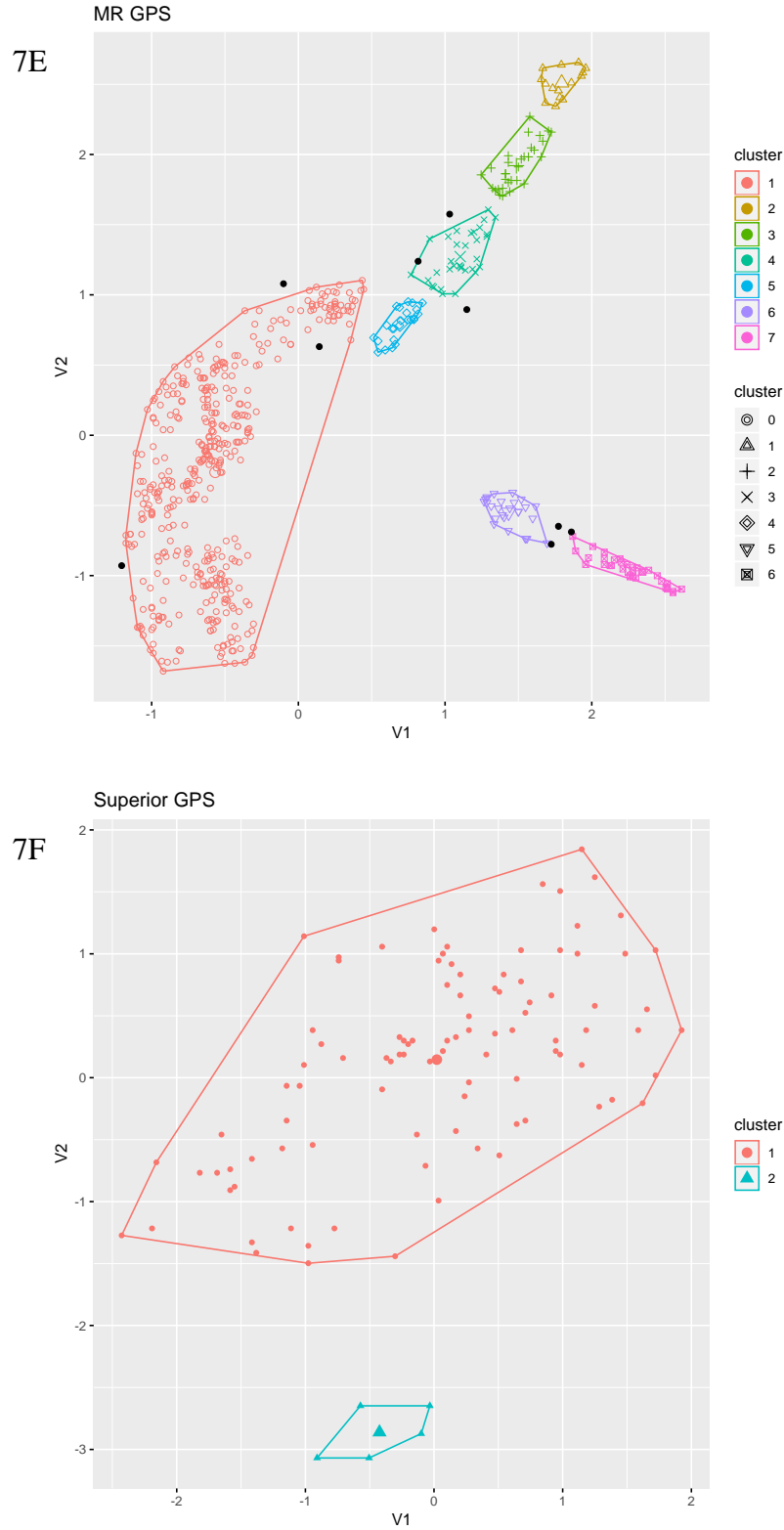


Figure 7: Graphs display all points in each colony and the identified clusters using the DBSCAN algorithm. Clusters are represented by colored polygons. Black points are considered noise points. Both GPS data collection and drone surveys were conducted at Short, Waldorf, BLM-18,

and KM colonies, and clusters were created for each data set. Only GPS data was collected at MR and Short, and clusters were created.

Table 1: Table of number of clusters, noise points, and number of burrows for each colony.

Colony	Number of Clusters	Noise Points	Number of Burrows
Short GPS	3	10	222
Short Drone	1	1	260
Waldorf GPS	25	135	1148
Waldorf Drone	33	149	2076
BLM18 GPS	11	26	876
BLM18 Drone	12	44	656
BLM18 Drone Full Extent	7	27	815
KM GPS	14	113	1872
KM Drone	10	83	1346
MR	7	9	610
Superior	2	0	103

Network Analysis Models

Six colonies throughout Boulder and Gunnison county were analyzed for network statistics. Network graphs were created for each colony to visualize the distribution and connectivity of burrows within each colony (Fig. 8). The colonies varied in size ranging from 101 to 2076 burrows (Table 2). Degree and Eigen centrality were calculated for each of the colonies as well as the degree and Eigen centrality of randomly simulated networks. All values for degree and Eigen centrality of both observed and simulated networks are listed in Table 2.

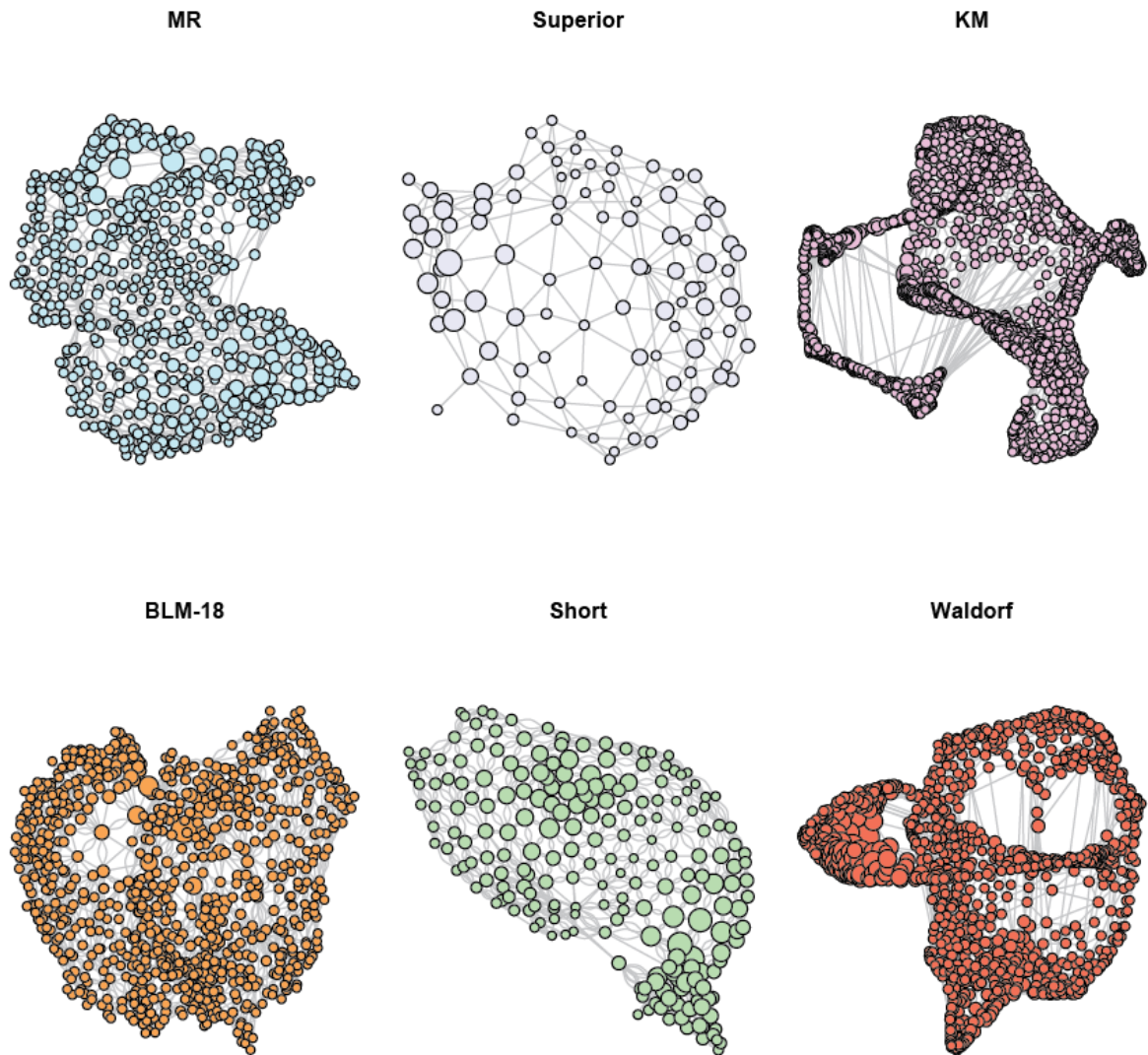


Figure 8: Diagrams of networks consisting of all burrows within prairie dog colonies in Colorado. Each point represents a burrow. The size of each point is scaled using Newman's Algorithm (Newman 2006), which identifies the connectivity of each point as a measure of importance to the overall network. More spatially connected burrows are represented by larger points.

Table 2: Description of statistics calculated for each colony. Statistics for simulated random networks based on each colony's degree statistics and number of burrows are included.

Colony Name	Location	Number of Burrows	Mean Degree	Mean Eigen Centrality	Random Network Degree	Random Network Eigen Centrality
Short	Boulder	222	11.31222	0.3803462	11.67568	0.5289402
Superior	Superior	101	5.378641	0.4438805	5.326733	0.3670229
MR	Gunnison	608	5.626263	0.1246697	5.680921	0.2715417
KM	Gunnison	1867	5.687193	0.03237977	5.81789	0.2110869
Waldorf	Boulder	1139	5.467857	0.07213397	5.474978	0.2953986
BLM-18	Gunnison	872	7.916763	0.04173993	7.947248	0.3378944

Simulated Random Networks

Random networks were simulated using the degree and number of burrows for each colony, and degree and Eigen centrality were calculated (Table 2). A Welch Two Sample t-test revealed that mean Eigen centrality of observed networks was 0.18 and mean Eigen centrality of random networks was 0.36. There was no significant difference between observed and simulated networks (p-value of 0.11). A Welch Two Sample t-test revealed that the mean degree of observed networks was 6.90 and the mean degree of random networks was 6.99. There was no significant difference between observed and simulated networks (p-value of 0.95).

Figure 9 displays observed, random, and small-world networks for the Short colony. The random network uses the degree of the observed network and the number of burrows to model what the network may look like by chance. The small-world network uses the number of burrows to model what the network would look like based on the small world phenomenon. The size of points in all graphs were calculated using Newman's Algorithm, which scales point size based on the relative importance of each point in the overall network (Newman 2006).

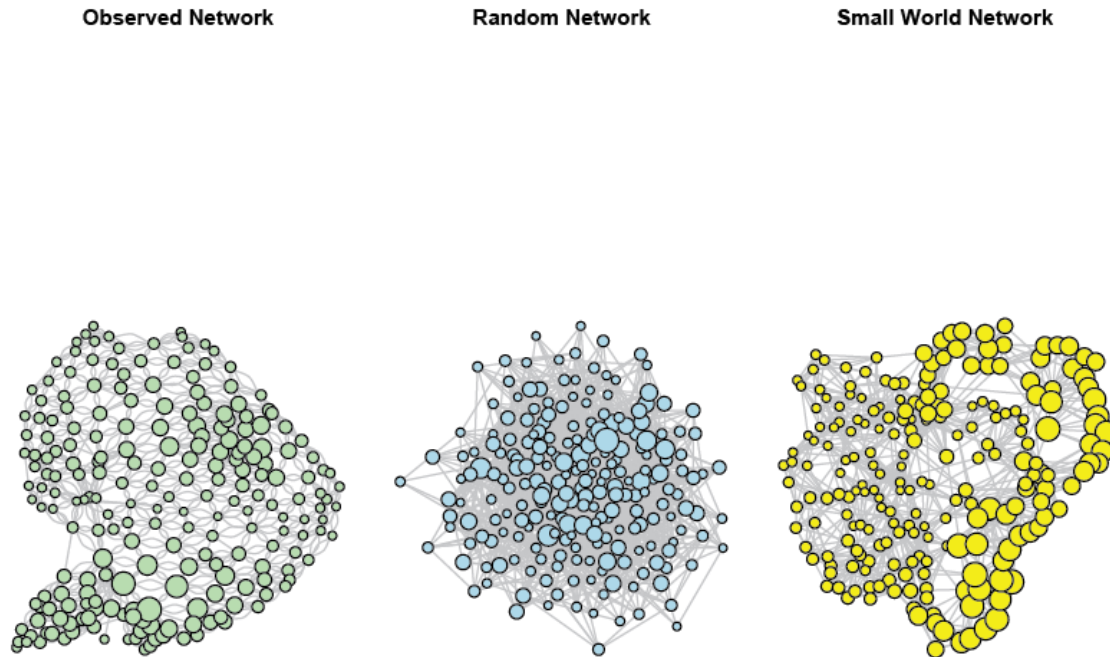


Figure 9: All graphs are from data on the Short colony in Boulder. The observed network is based on the actual distribution of burrows and the random network and small world network are two simulated models. All graphs contain the same number of points. The simulated models represent what may happen by chance and by chance with small world constraints with the same number of burrows and the same degree per burrows (mean observed degree/number of burrows).

Communities

Communities were identified in network analysis models using Newman's Algorithm (Newman 2006) (Fig. 10). On average, 8 communities were detected across the different colonies. These community models simulate how network data can be used to identify communities within prairie dog colonies based on connectivity of individuals. However, this model uses burrows as nodes instead of individual prairie dogs.

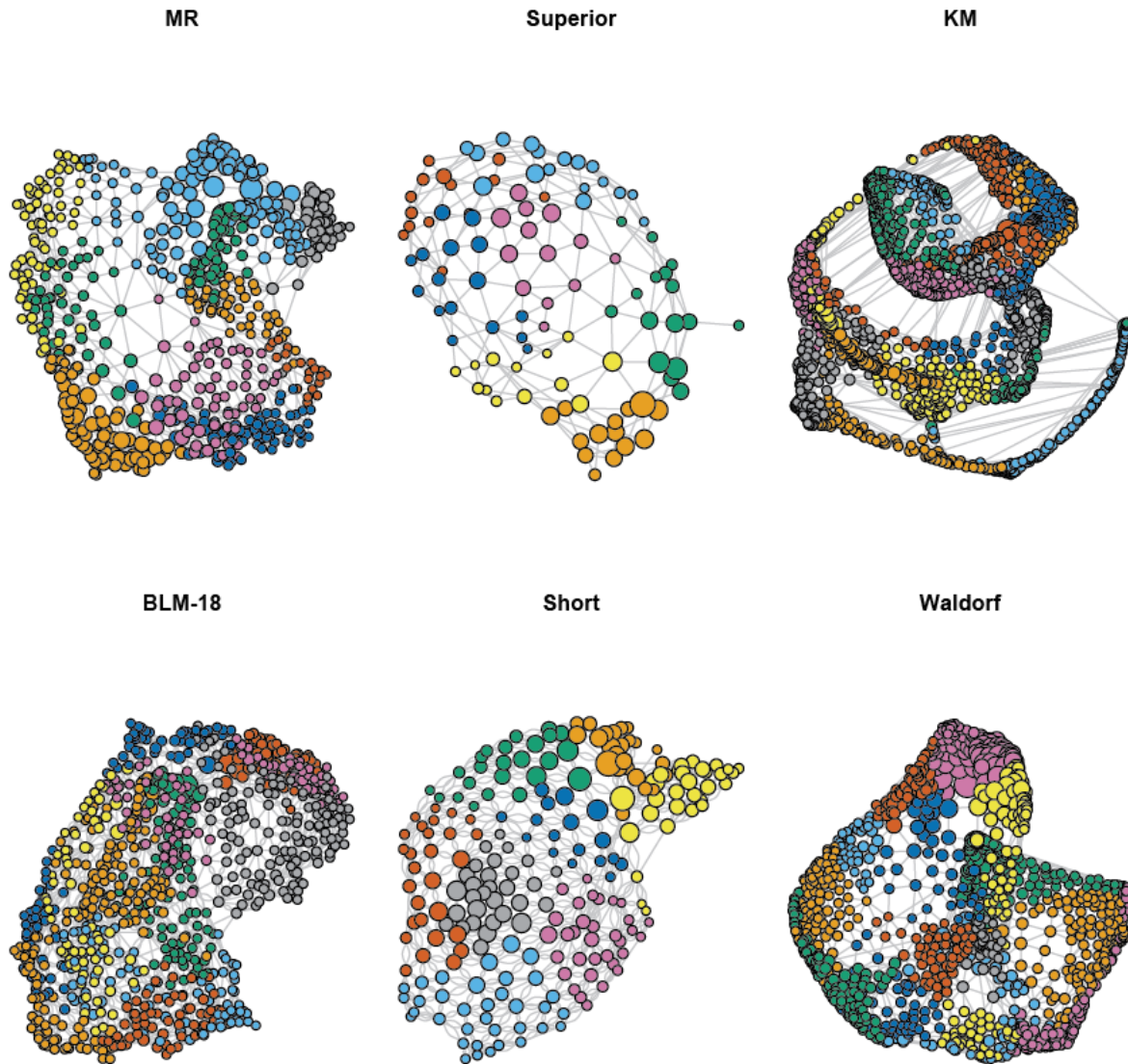


Figure 10: Network graphs of six colonies with points colored by communities. Communities were identified using Newman’s Algorithm (Newman 2006). The size of each point is scaled using Newman’s Algorithm, which identifies the connectivity of each point as a measure of importance to the overall network.

Discussion

This exploratory study aimed to improve the understanding of how animals use space in their environment and set the stage for further analysis based on tagging of individual prairie dogs with GPS units and collection of network data. This study provided insight into how burrows are arranged, how clustering varies throughout each colony, how survey methods affect

data, and how network analysis can be used to study prairie dog networks. Furthermore, the goal of this study was to test methodology for studying how animals use space.

Spatial Organization

The results of the coefficient of dispersion analysis indicate that there may be processes controlling the spatial organization of burrows and support hypothesis one that there is a non-random distribution of burrows in five out of six colonies. The results of the coefficient of dispersion analysis also indicate that there are significant levels of clustering in five out of six colonies, supporting hypothesis two. These results show that prairie dogs intentionally organize burrows into clusters. Future studies on genetic relatedness of individuals would be helpful to assess whether or not organization is driven by family groups within colonies. It is likely that the Short colony, which exhibited neither a non-random distribution nor a significant level of clustering or dispersion, may be influenced by other factors. The Short colony is a small colony in Boulder and may not experience a significant spatial organization pattern due to size, which may be influenced by the age of the colony. It is likely that younger colonies are smaller, and that colonies develop stricter spatial organization as they age and grow. If Short is indeed a young colony, it may not have had enough time to develop a distinct spatial organization pattern. Otherwise, it is possible that Short has established an optimal distribution of space and resources indicated by its Voronoi tessellation plot that resembles a honeycomb. In contrast, four of the five colonies that did exhibit significant spatial organization patterns are larger colonies, which may be older and may have had time to develop strict spatial organization. Prairie dogs are very territorial (Raylor 1988) and as the number of prairie dogs in a colony increases, more territories, or clusters, are likely to develop. The other colony, Superior, that exhibited a significant spatial organization pattern happened to be small, but was a satellite colony of a larger, older colony.

This satellite colony had recently emerged and was surveyed to monitor its development over time. It is possible that this satellite colony exhibited a significant spatial organization pattern despite its small size because it retained the spatial organization of the main colony from which it was directly derived.

Clusters of burrows identified via the DBSCAN algorithm are based on the proximity of burrows to each other and can be used to hypothesize where territories may be located. In the future, clusters of burrows can be compared to the genetic relatedness of individuals that inhabit these burrows to determine whether or not the clusters of burrows are representative of prairie dog family groups.

The results obtained from Voronoi tessellation analysis and tile area distributions from tessellation analysis support hypothesis four that there is variation in the distribution of burrows across space in colonies. Additionally, Voronoi tessellation allowed for identification of home ranges of burrows based on the amount of space each burrow can take up. Burrows with varying sizes have access to varying amounts of resources based on the area of land each burrow has command over. In future studies, tessellation plots can be overlaid onto drone imagery to compare tile areas with the quality and amount of vegetation. We would expect to see that burrows with larger tile areas would have less desirable vegetation than burrows with small tile areas.

Voronoi tessellation suggests that burrow size may vary depending on the number of resources, such that each burrow has the same amount of resources. This would support the ideal free distribution null hypothesis that animals organize such that each individual in a population receives equal resources. However, we know that this does not truly occur in nature due to behavioral dynamics. A similar study by Messier et al. (1990) used muskrat dwellings to test the

ideal free distribution hypothesis and found muskrats to have unequal access to resources due to territoriality. Since prairie dogs are biologically similar to muskrats and also exhibit territoriality, we would expect that certain prairie dog coterries (social groups) would have command over more desirable locations than other coterries. While the results of my study reject the ideal free distribution hypothesis, they agree with Sutherland (1983) that animal distribution is independent of population size based on the fact that five out of six colonies exhibited the same colony structure independent of number of burrows (Table 2).

Comparison of Methods

The finding obtained via kernel density analysis that there are differences in point density between surveying methods refutes hypothesis five that it does not matter which surveying method is used. Difference in density is due to error in both surveying methods. The GPS units used for data collection range in error from 5-15 meters, which can drastically affect how spatial data are displayed (Fig. 11).

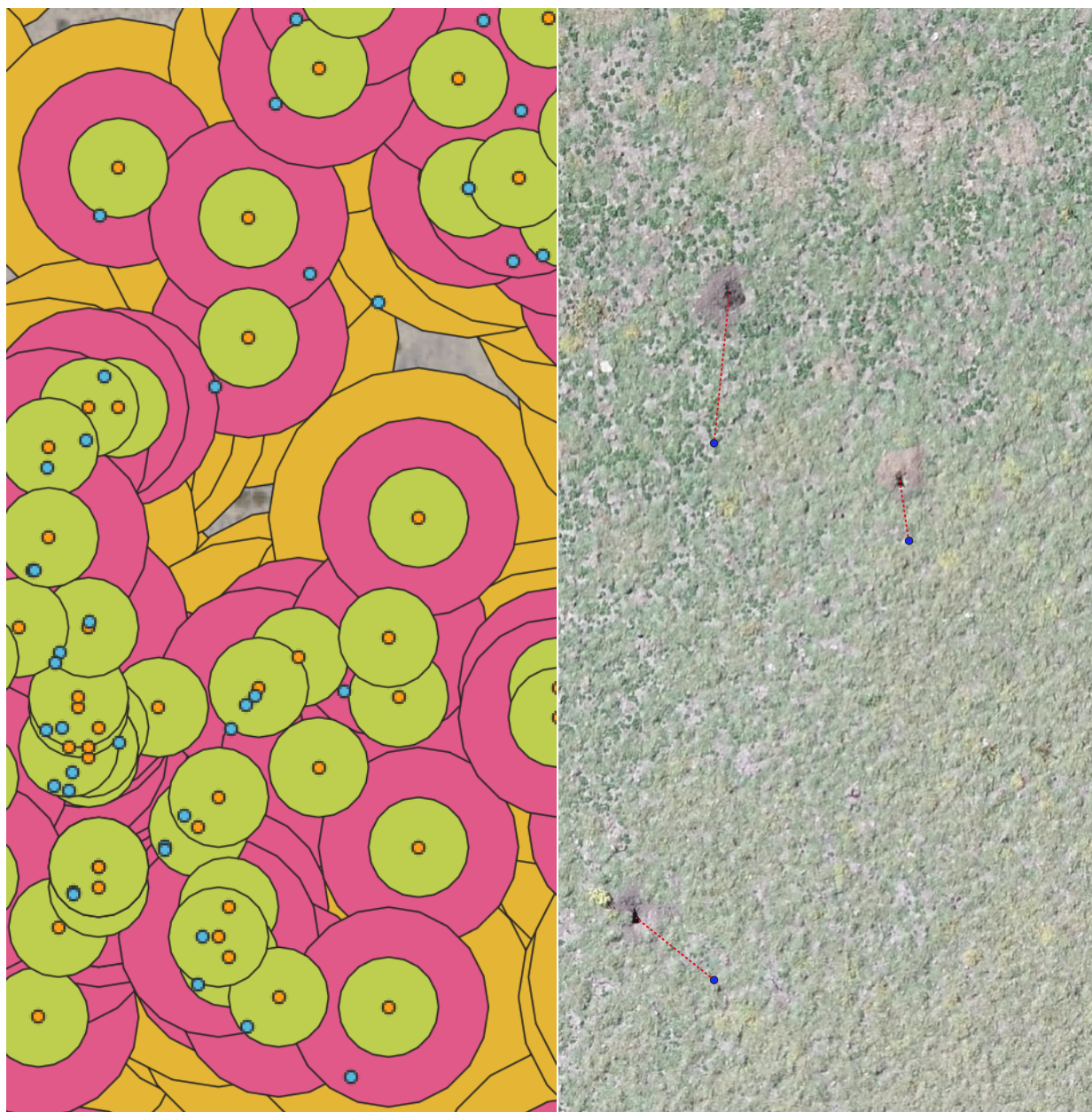


Figure 11: The left panel shows a close-up of a section of the KM colony. Buffers were created at 5, 10, and 15 meters around all GPS points to demonstrate the spatial uncertainty of where that point may actually be located due to GPS error. Blue points are burrows identified from drone imagery. This figure demonstrates how it is not possible to identify which GPS points should be associated with which drone points due to overlap of spatial uncertainty. The right panel is a depiction of GPS error. Blue points are the GPS data points and red dashed lines indicate which burrows the points are supposed to belong to.

Burrows can be missed by surveyors and also in drone images if they are covered by rock or sage brush. Additional, presumably inactive burrows may be marked in drone imagery but not with GPS. In contrast, the spatial accuracy of the drone is much greater than that of the GPS units.

My study aimed to assess the validity of using drones in wildlife biology, and the differences between data from GPS and drones suggest that the surveying method used should be selected depending on the study system. As suggested by Marvin et al. (2016), a combination of data collection technologies can be used to improve accuracy. GPS data have large spatial error, but may provide a more representative count of the number of active burrows. In contrast, drones have very low spatial error, but burrows may be missed or misidentified. An additional consideration is that drone surveys are 2-5 times faster than manual GPS surveys, which can save land management entities time and money.

The finding that the number of burrows identified by GPS versus drone imagery varied for each colony is likely due to sections of colonies being missed during GPS data collection and burrows not being visible in drone imagery due to being located under a rock or sage brush. GPS thus only tells part of the story. We can obtain additional vegetation and environmental data from drones and, by combining technologies, gain a more representative depiction of the world in order to address critical conservation and management issues (Stark et al. 2017).

Network Models

The use of network analysis in biology is growing, as it is a robust tool. Network theory has been used to study migration and landscape connectivity (Prima 2018), protected area design (Lea et al. 2016), social structure (Godfrey et al. 2014; Spiegel et al. 2017), conservation (Snijders et al. 2017), and disease transmission (Perkins et al. 2009). The use of spatial technology and network analysis in the present study is much more efficient and objective than

behavioral observations used in previous studies on prairie dogs (Manno et al. 2007; Verdolin et al. 2014).

Overall, my study provided a protocol for data processing, analysis, and visualization for future studies. The network communities identified here can be used to estimate the number of coteries in a colony. Network models similar to those produced in this study can be used to study space use, resource availability, reproductive dynamics, and disease transmission in prairie dogs and many other organisms. The results of this study can be used to test how the spatial distribution of prairie dogs relates to relatedness. Rayor (1988) used space use to study social organization. However, relatedness was based on use of the same burrows, not genetic testing, and would not have been able to discern mingling of unmarked litters with members of neighboring litters. Further studies may provide evidence of common space use based on genetic relatedness by comparing the clusters and communities defined in my study to genetic family groups. Additionally, Rayor (1988) only used two colonies, while my study used six colonies.

Conclusions

The goal of this study was to evaluate the methods used to study how animals use space. In addition to doing so, the key results from this study show that prairie dog burrows are non-randomly organized in cluster, that burrows have varying ecological footprints, that GPS and drone data collection produce different data, and that networks and clusters can be used to hypothesize the number and locations of coteries in a colony. The results of this study can thus inform future studies on how prairie dog spatial organization relates to genetic relatedness of individuals and environmental resource availability. How animals use space can be study across a wide range of taxa, including humans. While prairie dogs and humans alike are social organisms, we can use animal spatial organization to inspire novel human design of space use.

Especially as percent of developed land cover is increasing, it is essential for humans to make efficient use of space. Studying space use as a part of biological processes that have developed through evolution may help humans determine how to efficiently use space in our environment. “We are moving into a ‘golden age’ of animal tracking science and are beginning to use animals to inform us about crucial changes to the planet and to make predictions of future change, moving from simply studying animals, to using animals to study the planet” (Kays et al. 2015; Marvin et al. 2016).

References

- Bastille-Rousseau G, Douglas-Hamilton I, Blake S, Northrup JM, Wittemyer G. 2018. Applying network theory to animal movements to identify properties of landscape space use. *Ecol Appl.* 28(3):854–864. doi:10.1002/eap.1697.
- Chen B, Suer S, Ercan M, Tada R, Avci R, Kockara S. Constructing Super Rule Tree (SRT) for Protein Motif Clusters Using DBSCAN. :7.
- Chetkiewicz C-LB, St. Clair CC, Boyce MS. 2006. Corridors for Conservation: Integrating Pattern and Process. *Annual Review of Ecology, Evolution, and Systematics.* 37(1):317–342. doi:10.1146/annurev.ecolsys.37.091305.110050.
- Croft DP, Krause J, James R. 2004. Social networks in the guppy (*Poecilia reticulata*). *Proc R Soc B-Biol Sci.* 271:S516–S519. doi:10.1098/rsbl.2004.0206.
- Fretwell SD, Lucas HL. 1970. On territorial behavior and other factors influencing habitat distribution in birds. *Acta Biotheoretica.* 19:21.
- Godfrey SS, Ansari TH, Gardner MG, Farine DR, Bull CM. 2014. A contact-based social network of lizards is defined by low genetic relatedness among strongly connected individuals. *Animal Behaviour.* 97:35–43. doi:10.1016/j.anbehav.2014.08.019.
- Kays R, Crofoot MC, Jetz W, Wikelski M. 2015. Terrestrial animal tracking as an eye on life and planet. *Science.* 348(6240):aaa2478–aaa2478. doi:10.1126/science.aaa2478.
- Kleinhappel TK, John EA, Pike TW, Wilkinson A, Burman OHP. 2016. Animal welfare: a social networks perspective. *Sci Prog.* 99(1):68–82. doi:10.3184/003685016X14495640902331.
- Krause J, Croft DP, James R. 2007. Social network theory in the behavioural sciences: potential applications. *Behav Ecol Sociobiol.* 62(1):15–27. doi:10.1007/s00265-007-0445-8.
- Lea JSE, Humphries NE, von Brandis RG, Clarke CR, Sims DW. 2016. Acoustic telemetry and network analysis reveal the space use of multiple reef predators and enhance marine protected area design. *Proceedings of the Royal Society B: Biological Sciences.* 283(1834):20160717. doi:10.1098/rspb.2016.0717.
- Manno TG. 2008. Social networking in the Columbian ground squirrel, *Spermophilus columbianus*. *Anim Behav.* 75:1221–1228. doi:10.1016/j.anbehav.2007.09.025.
- Martinez-Estevéz L, Balvanera P, Pacheco J, Ceballos G. 2013. Prairie Dog Decline Reduces the Supply of Ecosystem Services and Leads to Desertification of Semiarid Grasslands. *PLOS ONE.* 8(10):e75229. doi:10.1371/journal.pone.0075229.
- Marvin DC, Koh LP, Lynam AJ, Wich S, Davies AB, Krishnamurthy R, Stokes E, Starkey R, Asner GP. 2016. Integrating technologies for scalable ecology and conservation. *Global Ecology and Conservation.* 7:262–275. doi:10.1016/j.gecco.2016.07.002.

- Messier F, Virgl JA, Marinelli L. 1990. Density-dependent habitat selection in muskrats: a test of the ideal free distribution model. *Oecologia*. 84(3):380–385. doi:10.1007/BF00329763.
- Mueller T, Lenz J, Caprano T, Fiedler W, Böhning-Gaese K. 2014. Large frugivorous birds facilitate functional connectivity of fragmented landscapes. Elphick C, editor. *Journal of Applied Ecology*. 51(3):684–692. doi:10.1111/1365-2664.12247.
- Newman MEJ. 2006. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*. 74(3). doi:10.1103/PhysRevE.74.036104.
- Perkins SE, Cagnacci F, Stradiotto A, Arnoldi D, Hudson PJ. 2009. Comparison of social networks derived from ecological data: implications for inferring infectious disease dynamics: Social networks for inferring disease dynamics. *Journal of Animal Ecology*. 78(5):1015–1022. doi:10.1111/j.1365-2656.2009.01557.x.
- Prima M-C, Duchesne T, Fortin A, Rivest L-P, Fortin D. 2018. Combining network theory and reaction-advection-diffusion modelling for predicting animal distribution in dynamic environments. Kriticos D, editor. *Methods in Ecology and Evolution*. 9(5):1221–1231. doi:10.1111/2041-210X.12997.
- Rayor LS. 1988. Social organization and space-use in Gunnison's prairie dog. *Behav Ecol Sociobiol*. 22(1):69–78. doi:10.1007/BF00395699.
- Rosenberg A, Arp R. 2009. *Philosophy of Biology: An Anthology*. John Wiley & Sons.
- Scott J. 2017. *Social Network Analysis*. SAGE.
- Snijders L, Blumstein DT, Stanley CR, Franks DW. 2017. Animal Social Network Theory Can Help Wildlife Conservation. *Trends Ecol Evol*. 32(8):567–577. doi:10.1016/j.tree.2017.05.005.
- Spiegel O, Sih A, Leu ST, Bull CM. 2018. Where should we meet? Mapping social network interactions of sleepy lizards shows sex-dependent social network structure. *Animal Behaviour*. 136:207–215. doi:10.1016/j.anbehav.2017.11.001.
- Stark DJ, Vaughan IP, Evans LJ, Kler H, Goossens B. 2018. Combining drones and satellite tracking as an effective tool for informing policy change in riparian habitats: a proboscis monkey case study. Pettorelli N, Horning N, editors. *Remote Sensing in Ecology and Conservation*. 4(1):44–52. doi:10.1002/rse2.51.
- Sutherland WJ. 1983. Aggregation and the 'Ideal Free' Distribution. *The Journal of Animal Ecology*. 52(3):821. doi:10.2307/4456.
- Travis SE, Slobodchikoff CN, Keim P. 1995. Ecological and Demographic Effects on Intraspecific Variation in the Social System of Prairie Dogs. *Ecology*. 76(6):1794–1803. doi:10.2307/1940711.
- Verdolin JL, Traud AL, Dunn RR. 2014. Key players and hierarchical organization of prairie dog social networks. *Ecol Complex*. 19:140–147. doi:10.1016/j.ecocom.2014.06.003.

Zhang J. 2018. Effects of cell irregularity on the thermal conductivity of carbon honeycombs. *Carbon*. 131:127–136. doi:10.1016/j.carbon.2018.01.097.

Code

```
#####TESSELLATION#####

library(deldir)
quartz()
par(mfrow=c(2,3), mar=c(2,2,2,1), oma=c(1,1,1,1))

#MR
MR <- read.csv("Gunnison_mapping_9-18_MR.csv")
MR_tess <- deldir(MR$X, MR$Y)
plot(MR$X, MR$Y, cex=.0001)
tiles <- plot(MR_tess, wlines="tess", wpoints = "none", number=FALSE, add=TRUE, lty=1)
title("MR")

#Superior
superior <- read.csv("Superior_utms.csv")
superior_tess <- deldir(superior$X, superior$Y)
plot(superior$X, superior$Y, cex=.0001)
tiles <- plot(superior_tess, wlines="tess", wpoints = "none", number=FALSE, add=TRUE, lty=1)
title("Superior")

#KM
KM <- read.csv("KM_burrows_qgis.csv")
KM_tess <- deldir(KM$X, KM$Y)
plot(KM$X, KM$Y, cex=.0001)
tiles <- plot(KM_tess, wlines="tess", wpoints = "none", number=FALSE, add=TRUE, lty=1)
title("KM")

#BLM-18
BLM18 <- read.csv("BLM18_qgis_fullextent.csv")
BLM18_tess <- deldir(BLM18$X, BLM18$Y)
plot(BLM18$X, BLM18$Y, cex=.0001)
tiles <- plot(BLM18_tess, wlines="tess", wpoints = "none", number=FALSE, add=TRUE, lty=1)
title("BLM-18")

#Short
short <- read.csv("short_drone_qgis.csv")
short_tess <- deldir(short$X, short$Y)
plot(short$X, short$Y, cex=0.0001)
plot(short_tess, wlines="tess", wpoints="none", number=FALSE, add=TRUE, lty=1)
title("Short")

#Waldorf
waldorf <- read.csv("Waldorf_qgis_burrows.csv")
waldorf_tess <- deldir(waldorf$X, waldorf$Y)
plot(waldorf$X, waldorf$Y, cex=.0001)
tiles <- plot(waldorf_tess, wlines="tess", wpoints = "none", number=FALSE, add=TRUE, lty=1)
title("Waldorf")

#####

#tile area distribution
quartz()
par(mfrow=c(3,2), mar=c(2,2,2,1), oma=c(1,1,1,1))
```

```

#MR
MR_areas <- tileInfo(MR_tess)$Areas
hist(log(MR_areas), main="MR", xlim=c(0,10))
abline(v= log(mean(MR_areas)), lty=3, lwd=3)

#Superior
superior_areas <- tileInfo(superior_tess)$Areas
hist(log(superior_areas), main="Superior", xlim=c(0,10))
abline(v= log(mean(superior_areas)), lty=3, lwd=3)

#KM
KM_areas <- tileInfo(KM_tess)$Areas
hist(log(KM_areas), main="KM", xlim=c(0,10))
abline(v= log(mean(KM_areas)), lty=3, lwd=3)

#BLM-18
BLM18_areas <- tileInfo(BLM18_tess)$Areas
hist(log(BLM18_areas), main="BLM-18", xlim=c(0,10))
abline(v= log(mean(BLM18_areas)), lty=3, lwd=3)

#Short
short_areas <- tileInfo(short_tess)$Areas
hist(log(short_areas), main="Short", xlim=c(0,10))
abline(v= log(mean(short_areas)), lty=3, lwd=3)

#Waldorf
waldorf_areas <- tileInfo(waldorf_tess)$Areas
hist(log(waldorf_areas), main="Waldorf", xlim=c(0,10))
abline(v= log(mean(waldorf_areas)), lty=3, lwd=3)

#####

#####TRIANGULATION LISTS#####

#MR triangulated points
triangulation_MR <- triMat(MR_tess)
#writes values to a csv
write.csv(triangulation_MR, file = "MR_triangulation.csv")

#in excel paste V3 under V2 along with V1. Add IDs to pasted values. Sort V1 ascending. Remove duplicates in V2.
Rename columns.

#Superior triangulated points
triangulation_superior <- triMat(superior_tess)
#writes values to a csv
write.csv(triangulation_superior, file = "superior_triangulation.csv")

#in excel paste V3 under V2 along with V1. Add IDs to pasted values. Sort V1 ascending. Remove duplicates in V2.
Rename columns.

#KM triangulated points
triangulation_KM <- triMat(KM_tess)
#writes values to a csv
write.csv(triangulation_KM, file = "KM_triangulation.csv")

```

#in excel paste V3 under V2 along with V1. Add IDs to pasted values. Sort V1 ascending. Remove duplicates in V2.
Rename columns.

```
#BLM-18 triangulated points
triangulation_BLM18 <- triMat(BLM18_tess)
#writes values to a csv
write.csv(triangulation_BLM18, file = "BLM18_triangulation.csv")
```

#in excel paste V3 under V2 along with V1. Add IDs to pasted values. Sort V1 ascending. Remove duplicates in V2.
Rename columns.

```
#Waldorf triangulated points
triangulation <- triMat(waldorf_tess )
#writes values to a csv
write.csv(triangulation, file = "waldorf_triangulation.csv")
```

#in excel paste V3 under V2 along with V1. Add IDs to pasted values. Sort V1 ascending. Remove duplicates in V2.
Rename columns.

#####

#####CLUSTERS (DBSCAN)#####

```
#install factoextra to visualize plots
install.packages("factoextra")
#install dbscan
install.packages("dbscan")
```

```
library(dbscan)
library(factoextra)
```

#WALDORF#####

```
#waldorf data to matrix
waldorf <- read.csv("Waldorf_oct-18.csv")
waldorf_waypoints <- cbind(waldorf$X, waldorf$Y)
waldorf_waypoints <- as.matrix(waldorf_waypoints)
length(waldorf_waypoints[,1])
#determine epsilon by plotting k nearest neighbor then looking for the knee in the curve- I used 6 neighbors (really
just changes y axis scale)
kNNdistplot(waldorf_waypoints, 6)
#knee at ~10
```

```
#run dbscan
waldorf_dbscan <- dbscan(waldorf_waypoints, eps = 10, borderPoints = TRUE)
```

```
#do same steps for waldorf digital (drone imagery)
waldorf_digital <- read.csv("Waldorf_qgis_burrows.csv")
waldorf_dig_waypoints <- cbind(waldorf_digital$X, waldorf_digital$Y)
waldorf_dig_waypoints <- as.matrix(waldorf_dig_waypoints)
length(waldorf_dig_waypoints[,1])
kNNdistplot(waldorf_dig_waypoints,1)
waldorf_dig_dbscan <- dbscan(waldorf_dig_waypoints, eps=10, borderPoints = TRUE)
```

```
quartz()
#visualize waldorf dbscan clusters
fviz_cluster(waldorf_dbscan, as.data.frame(waldorf_waypoints), geom = "point", main = "Waldorf GPS")
```



```

quartz()
#visualize waldorf digital clusters
fviz_cluster(waldorf_dig_dbscan, as.data.frame(waldorf_dig_waypoints), geom = "point", main = "Waldorf Drone")

#k means with number of points from dbscan
waldorf_kmeans <- kmeans(waldorf_waypoints, 25)
quartz()
fviz_cluster(waldorf_kmeans, as.data.frame(waldorf_waypoints), geom="point", main="Waldorf GPS Kmeans")
#SHORT#####
short <- read.csv("Short_7_24_18_burrows (1).csv")
short_waypoints <- cbind(short$X, short$Y)
short_waypoints <- as.matrix(short_waypoints)
length(short_waypoints)
#find epsilon
quartz()
kNNdistplot(short_waypoints,6)
#epsilon=10
#run dbscan
short_dbscan <- dbscan(short_waypoints, eps=10, borderPoints = TRUE)
#visualize clusters
fviz_cluster(short_dbscan, as.data.frame(short_waypoints), geom="point", main="Short GPS")

#short drone image points
short_drone <- read.csv("short_drone_qgis.csv")
short_drone_waypoints <- cbind(short_drone$X, short_drone$Y)
short_drone_waypoints <- as.matrix(short_drone_waypoints)
length(short_drone_waypoints[,1])
#find epsilon
quartz()
kNNdistplot(short_drone_waypoints, 6)
#epsilon=11
#run dbscan
short_drone_dbscan <- dbscan(short_drone_waypoints, eps=11, borderPoints = TRUE)
#visualize clusters
fviz_cluster(short_drone_dbscan, as.data.frame(short_drone_waypoints), geom="point", main="Short Drone")

#BLM18#####
BLM18 <- read.csv("Gunnison_mapping_9-18_BLM18.csv")
BLM18_waypoints <- cbind(BLM18$X, BLM18$Y)
BLM18_waypoints <- as.matrix(BLM18_waypoints)
length(BLM18_waypoints[,1])
#find epsilon
quartz()
kNNdistplot(BLM18_waypoints, 6)
#epsilon=20
#run dbscan
BLM18_dbscan <- dbscan(BLM18_waypoints, eps=20, borderPoints = TRUE)
#visualize clusters
fviz_cluster(BLM18_dbscan, as.data.frame(BLM18_waypoints), geom="point", main="BLM-18 GPS")

#BLM-18 drone image points (GPS extent)
BLM18_drone <- read.csv("BLM18_qgis_GPSExtent.csv")
BLM18_drone_waypoints <- cbind(BLM18_drone$X, BLM18_drone$Y)
BLM18_drone_waypoints <- as.matrix(BLM18_drone_waypoints)
length(BLM18_drone_waypoints[,1])
#find epsilon

```

```

quartz()
kNNdistplot(BLM18_drone_waypoints,6)
#epsilon=20
#run dbscan
BLM18_drone_dbscan <- dbscan(BLM18_drone_waypoints, eps=20, borderPoints = TRUE)
#visualize clusters
fviz_cluster(BLM18_drone_dbscan, as.data.frame(BLM18_drone_waypoints), geom="point", main="BLM-18
Drone")

#BLM-18 drone full extent
BLM18_drone_full <- read.csv("BLM18_qgis_fullextent.csv")
BLM18_drone_full_waypoints <- cbind(BLM18_drone_full$X,BLM18_drone_full$Y)
BLM18_drone_full_waypoints <- as.matrix(BLM18_drone_full_waypoints)
length(BLM18_drone_full_waypoints[,1])
#find epsilon
quartz()
kNNdistplot(BLM18_drone_full_waypoints,6)
#epsilon=28
#run dbscan
BLM18_drone_full_dbscan <- dbscan(BLM18_drone_full_waypoints, eps=28, borderPoints = TRUE)
#visualize clusters
fviz_cluster(BLM18_drone_full_dbscan, as.data.frame(BLM18_drone_full_waypoints), geom="point",
main="BLM-18 Drone Full Extent")

#KM#####
KM <- read.csv("Gunnison_mapping_9-18_KM.csv")
KM_waypoints <- cbind(KM$X, KM$Y)
KM_waypoints <- as.matrix(KM_waypoints)
length(KM_waypoints[,1])
#find epsilon
quartz()
kNNdistplot(KM_waypoints,6)
#epsilon=15
#run dbscan
KM_dbscan <- dbscan(KM_waypoints, eps=15, borderPoints = TRUE)
#visualize clusters
fviz_cluster(KM_dbscan, as.data.frame(KM_waypoints), geom="point", main="KM GPS")

#KM drone image points
KM_drone <- read.csv("KM_burrows_qgis.csv")
KM_drone_waypoints <- cbind(KM_drone$X, KM_drone$Y)
KM_drone_waypoints <- as.matrix(KM_drone_waypoints)
length(KM_drone_waypoints[,1])
#find epsilon
quartz()
kNNdistplot(KM_drone_waypoints, 6)
#epsilon=18
KM_drone_dbscan <- dbscan(KM_drone_waypoints, eps=18, borderPoints = TRUE)
#visualize clusters
fviz_cluster(KM_drone_dbscan, as.data.frame(KM_drone_waypoints), geom="point", main="KM Drone")

#MR#####
MR <- read.csv("Gunnison_mapping_9-18_MR.csv")
MR_waypoints <- cbind(MR$X, MR$Y)
MR_waypoints <- as.matrix(MR_waypoints)
length(MR_waypoints[,1])

```

```

#find epsilon
quartz()
kNNdistplot(MR_waypoints,6)
#epsilon=15
#run dbscan
MR_dbscan <- dbscan(MR_waypoints, eps=15, borderPoints = TRUE)
#visualize clusters
fviz_cluster(MR_dbscan, as.data.frame(MR_waypoints), geom="point", main="MR GPS")

#Superior#####
superior <- read.csv("Superior_utms.csv")
superior_waypoints <- cbind(superior$X, superior$Y)
superior_waypoints <- as.matrix(superior_waypoints)
length(superior_waypoints[,1])
#find epsilon
quartz()
kNNdistplot(superior_waypoints,6)
#epsilon=22
#run dbscan
superior_dbscan <- dbscan(superior_waypoints, eps=22, borderPoints = TRUE)
#visualize clusters
fviz_cluster(superior_dbscan, as.data.frame(superior_waypoints), geom="point", main="Superior GPS")

#####

#####COEFFICIENT OF DISPERSION#####

#MR#####
MR_cd <- read.csv("Gunnison_mapping_9-18_MR.csv")

#estimate whether distribution of holes is clustered, random, or hyper-dispersed

#get the number of holes
n_holes <- length(MR_cd$X)

#choose the number of holes to randomly sample for measuring the distance to the nearest hole
n_samples <- 40

#make a vector to store a lot of cd values...we will estimate the mean cd from 1000 replicate samples of 40
randomly selected holes
cd <- rep(NA, 1000)

#run the simulation to generate cd values
for (k in 1:1000){

  #make a vector to store minimum distances
  min_dist <- rep(NA, n_samples)

  for (j in 1:n_samples){

    #choose a random hole
    r_hole <- sample(seq(1,n_holes),1)

    #get the coordinates of random hole
    temp_x <- MR_cd$X[r_hole]
    temp_y <- MR_cd$Y[r_hole]

```

```

#remove the randomly selected hole
x <- MR_cd$X[-r_hole]
y <- MR_cd$Y[-r_hole]

#calculate the distance to all other holes
distances <- rep(NA, n_holes)

for (i in 1:n_holes){
  distances[i] <- sqrt((temp_x - x[i])^2 + (temp_y - y[i])^2)
}

#select the minimum distance
min_dist[j] <- min(distances, na.rm=T)
}

#calculate coefficient of dispersion for each of the 1000 replicates
cd[k] <- var(min_dist)/mean(min_dist)
}

#calculate the statistic
mean_cd <- mean(log10(cd))

#calculate the standard error for mean_cd
con_limit_lower <- quantile(log10(cd),0.025)
con_limit_upper <- quantile(log10(cd),0.975)

#calculate the p value
p <- sum(log10(cd) <= 0)/1000

#make a histogram of the data
freq_cd <- hist(log10(cd), breaks=seq(-0.25, 1, 0.05),col="light blue", las =1, ylim=c(0,250), xlab="Coefficient of
dispersion", main="MR")
abline(v=0, lwd=3, lty=3)
text(0.4, 220, "Clumped", cex = 0.7)
text(-0.15, 220, "Hyper \n dispersed", cex = 0.7)

#draw an effect size graph
plot(NA, NA, xlim=c(-0.2, 1), ylim=c(0,1), xlab="Coefficient of dispersion", yaxt="n", ylab=NA, main="MR",
pch=19)
segments(con_limit_lower, 0.5, con_limit_upper, 0.5)
abline(v=0, lty=3, lwd=2)
points(mean_cd, 0.5)
#####

#Superior#####
superior_cd <- read.csv("Superior_utms.csv")

#estimate whether distribution of holes is clustered, random, or hyper-dispersed

#get the number of holes
n_holes <- length(superior_cd$X)

#choose the number of holes to randomly sample for measuring the distance to the nearest hole
n_samples <- 40

```

```

#make a vector to store a lot of cd values...we will estimate the mean cd from 1000 replicate samples of 40
randomly selected holes
cd <- rep(NA, 1000)

#run the simulation to generate cd values
for (k in 1:1000){

  #make a vector to store minimum distances
  min_dist <- rep(NA, n_samples)

  for (j in 1:n_samples){

    #choose a random hole
    r_hole <- sample(seq(1,n_holes),1)

    #get the coordinates of random hole
    temp_x <- superior_cd$X[r_hole]
    temp_y <- superior_cd$Y[r_hole]

    #remove the randomly selected hole
    x <- superior_cd$X[-r_hole]
    y <- superior_cd$Y[-r_hole]

    #calculate the distance to all other holes
    distances <- rep(NA, n_holes)

    for (i in 1:n_holes){
      distances[i] <- sqrt((temp_x - x[i])^2 + (temp_y - y[i])^2)
    }

    #select the minimum distance
    min_dist[j] <- min(distances, na.rm=T)
  }

  #calculate coefficient of dispersion for each of the 1000 replicates
  cd[k] <- var(min_dist)/mean(min_dist)
}

#calculate the statistic
mean_cd <- mean(log10(cd))

#calculate the standard error for mean_cd
con_limit_lower <- quantile(log10(cd),0.025)
con_limit_upper <- quantile(log10(cd),0.975)

#calculate the p value
p <- sum(log10(cd) <= 0)/1000

#make a histogram of the data
freq_cd <- hist(log10(cd), breaks=seq(-0.25, 1, 0.05),col="light blue", las =1, ylim=c(0,250), xlab="Coefficient of
dispersion", main="Superior")
abline(v=0, lwd=3, lty=3)
text(0.4, 220, "Clumped", cex = 0.7)
text(-0.15, 220, "Hyper \n dispersed", cex = 0.7)

#draw an effect size graph

```

```

plot(NA, NA, xlim=c(-0.2, 1), ylim=c(0,1), xlab="Coefficient of dispersion", yaxt="n", ylab=NA, main="Superior",
pch=19)
segments(con_limit_lower, 0.5, con_limit_upper, 0.5)
abline(v=0, lty=3, lwd=2)
points(mean_cd, 0.5)
#####

#KM#####

KM_cd <- read.csv("KM_burrows_qgis.csv")

#estimate whether distribution of holes is clustered, random, or hyper-dispersed

#get the number of holes
n_holes <- length(KM_cd$X)

#choose the number of holes to randomly sample for measuring the distance to the nearest hole
n_samples <- 40

#make a vector to store a lot of cd values...we will estimate the mean cd from 1000 replicate samples of 40
randomly selected holes
cd <- rep(NA, 1000)

#run the simulation to generate cd values
for (k in 1:1000){

  #make a vector to store minimum distances
  min_dist <- rep(NA, n_samples)

  for (j in 1:n_samples){

    #choose a random hole
    r_hole <- sample(seq(1,n_holes),1)

    #get the coordinates of random hole
    temp_x <- KM_cd$X[r_hole]
    temp_y <- KM_cd$Y[r_hole]

    #remove the randomly selected hole
    x <- KM_cd$X[-r_hole]
    y <- KM_cd$Y[-r_hole]

    #calculate the distance to all other holes
    distances <- rep(NA, n_holes)

    for (i in 1:n_holes){
      distances[i] <- sqrt((temp_x - x[i])^2 + (temp_y - y[i])^2)
    }

    #select the minimum distance
    min_dist[j] <- min(distances, na.rm=T)
  }

  #calculate coefficient of dispersion for each of the 1000 replicates
  cd[k] <- var(min_dist)/mean(min_dist)
}

```

```

#calculate the statistic
mean_cd <- mean(log10(cd))

#calculate the standard error for mean_cd
con_limit_lower <- quantile(log10(cd),0.025)
con_limit_upper <- quantile(log10(cd),0.975)

#calculate the p value
p <- sum(log10(cd) <= 0)/1000

#make a histogram of the data
freq_cd <- hist(log10(cd), breaks=seq(-0.25, 1, 0.05),col="light blue", las =1, ylim=c(0,250), xlab="Coefficient of
dispersion", main="KM")
abline(v=0, lwd=3, lty=3)
text(0.4, 220, "Clumped", cex = 0.7)
text(-0.15, 220, "Hyper \n dispersed", cex = 0.7)

#draw an effect size graph
plot(NA, NA, xlim=c(-0.2, 1), ylim=c(0,1), xlab="Coefficient of dispersion", yaxt="n", ylab=NA, main="KM",
pch=19)
segments(con_limit_lower, 0.5, con_limit_upper, 0.5)
abline(v=0, lty=3, lwd=2)
points(mean_cd, 0.5)
#####

#BLM18#####
BLM18_cd <- read.csv("BLM18_qgis_fullextent.csv")

#estimate whether distribution of holes is clustered, random, or hyper-dispersed

#get the number of holes
n_holes <- length(BLM18_cd$X)

#choose the number of holes to randomly sample for measuring the distance to the nearest hole
n_samples <- 40

#make a vector to store a lot of cd values...we will estimate the mean cd from 1000 replicate samples of 40
randomly selected holes
cd <- rep(NA, 1000)

#run the simulation to generate cd values
for (k in 1:1000){

  #make a vector to store minimum distances
  min_dist <- rep(NA, n_samples)

  for (j in 1:n_samples){

    #choose a random hole
    r_hole <- sample(seq(1,n_holes),1)

    #get the coordinates of random hole
    temp_x <- BLM18_cd$X[r_hole]
    temp_y <- BLM18_cd$Y[r_hole]

    #remove the randomly selected hole

```

```

x <- BLM18_cd$X[-r_hole]
y <- BLM18_cd$Y[-r_hole]

#calculate the distance to all other holes
distances <- rep(NA, n_holes)

for (i in 1:n_holes){
  distances[i] <- sqrt((temp_x - x[i])^2 + (temp_y - y[i])^2)
}

#select the minimum distance
min_dist[j] <- min(distances, na.rm=T)
}

#calculate coefficient of dispersion for each of the 1000 replicates
cd[k] <- var(min_dist)/mean(min_dist)
}

#calculate the statistic
mean_cd <- mean(log10(cd))

#calculate the standard error for mean_cd
con_limit_lower <- quantile(log10(cd),0.025)
con_limit_upper <- quantile(log10(cd),0.975)

#calculate the p value
p <- sum(log10(cd) <= 0)/1000

#make a histogram of the data
freq_cd <- hist(log10(cd), breaks=seq(-0.25, 1.5, 0.05),col="light blue", las =1, ylim=c(0,250), xlab="Coefficient of
dispersion", main="BLM-18")
abline(v=0, lwd=3, lty=3)
text(0.6, 220, "Clumped", cex = 0.7)
text(-0.15, 220, "Hyper \n dispersed", cex = 0.7)

#draw an effect size graph
plot(NA, NA, xlim=c(-0.2, 1), ylim=c(0,1), xlab="Coefficient of dispersion", yaxt="n", ylab=NA, main="BLM-18",
pch=19)
segments(con_limit_lower, 0.5, con_limit_upper, 0.5)
abline(v=0, lty=3, lwd=2)
points(mean_cd, 0.5)
#####

#SHORT#####
#get the data
#pdog <- read.csv("5-30_waypoints.csv")
short_cd <- read.csv("short_drone_qgis.csv")

#estimate whether distribution of holes is clustered, random, or hyper-dispersed

#get the number of holes
n_holes <- length(short_cd$X)

#choose the number of holes to randomly sample for measuring the distance to the nearest hole
n_samples <- 40

```



```

#make a vector to store a lot of cd values...we will estimate the mean cd from 1000 replicate samples of 40
randomly selected holes
cd <- rep(NA, 1000)

#run the simulation to generate cd values
for (k in 1:1000){

  #make a vector to store minimum distances
  min_dist <- rep(NA, n_samples)

  for (j in 1:n_samples){

    #choose a random hole
    r_hole <- sample(seq(1,n_holes),1)

    #get the coordinates of random hole
    temp_x <- short_cd$X[r_hole]
    temp_y <- short_cd$Y[r_hole]

    #remove the randomly selected hole
    x <- short_cd$X[-r_hole]
    y <- short_cd$Y[-r_hole]

    #calculate the distance to all other holes
    distances <- rep(NA, n_holes)

    for (i in 1:n_holes){
      distances[i] <- sqrt((temp_x - x[i])^2 + (temp_y - y[i])^2)
    }

    #select the minimum distance
    min_dist[j] <- min(distances, na.rm=T)
  }

  #calculate coefficient of dispersion for each of the 1000 replicates
  cd[k] <- var(min_dist)/mean(min_dist)
}

#calculate the statistic
mean_cd <- mean(log10(cd))

#calculate the standard error for mean_cd
con_limit_lower <- quantile(log10(cd),0.025)
con_limit_upper <- quantile(log10(cd),0.975)

#calculate the p value
p <- sum(log10(cd) <= 0)/1000

#make a histogram of the data
freq_cd <- hist(log10(cd), breaks=seq(-1, 1, 0.05),col="light blue", las =1, ylim=c(0,250), xlab="Coefficient of
dispersion", main="Short")
abline(v=0, lwd=3, lty=3)
text(0.4, 220, "Clumped", cex = 0.7)
text(-0.5, 220, "Hyper \n dispersed", cex = 0.7)

#draw an effect size graph

```

```

plot(NA, NA, xlim=c(-0.4, 1), ylim=c(0,1), xlab="Coefficient of dispersion", yaxt="n", ylab=NA, main="Short",
pch=19)
segments(con_limit_lower, 0.5, con_limit_upper, 0.5)
abline(v=0, lty=3, lwd=2)
points(mean_cd, 0.5)
#####

#Waldorf#####
waldorf_cd <- read.csv("Waldorf_qgis_burrows.csv")

#estimate whether distribution of holes is clustered, random, or hyper-dispersed

#get the number of holes
n_holes <- length(waldorf_cd$X)

#choose the number of holes to randomly sample for measuring the distance to the nearest hole
n_samples <- 40

#make a vector to store a lot of cd values...we will estimate the mean cd from 1000 replicate samples of 40
randomly selected holes
cd <- rep(NA, 1000)

#run the simulation to generate cd values
for (k in 1:1000){

  #make a vector to store minimum distances
  min_dist <- rep(NA, n_samples)

  for (j in 1:n_samples){

    #choose a random hole
    r_hole <- sample(seq(1,n_holes),1)

    #get the coordinates of random hole
    temp_x <- waldorf_cd$X[r_hole]
    temp_y <- waldorf_cd$Y[r_hole]

    #remove the randomly selected hole
    x <- waldorf_cd$X[-r_hole]
    y <- waldorf_cd$Y[-r_hole]

    #calculate the distance to all other holes
    distances <- rep(NA, n_holes)

    for (i in 1:n_holes){
      distances[i] <- sqrt((temp_x - x[i])^2 + (temp_y - y[i])^2)
    }

    #select the minimum distance
    min_dist[j] <- min(distances, na.rm=T)
  }

  #calculate coefficient of dispersion for each of the 1000 replicates
  cd[k] <- var(min_dist)/mean(min_dist)
}

```

```

#calculate the statistic
mean_cd <- mean(log10(cd))

#calculate the standard error for mean_cd
con_limit_lower <- quantile(log10(cd),0.025)
con_limit_upper <- quantile(log10(cd),0.975)

#calculate the p value
p <- sum(log10(cd) <= 0)/1000

#make a histogram of the data
freq_cd <- hist(log10(cd), breaks=seq(-0.25, 1, 0.05),col="light blue", las =1, ylim=c(0,250), xlab="Coefficient of
dispersion", main="Waldorf")
abline(v=0, lwd=3, lty=3)
text(0.4, 220, "Clumped", cex = 0.7)
text(-0.15, 220, "Hyper \n dispersed", cex = 0.7)

#draw an effect size graph
plot(NA, NA, xlim=c(-0.2, 1), ylim=c(0,1), xlab="Coefficient of dispersion", yaxt="n", ylab=NA, main="Waldorf",
pch=19)
segments(con_limit_lower, 0.5, con_limit_upper, 0.5)
abline(v=0, lty=3, lwd=2)
points(mean_cd, 0.5)
#####

#####NETWORK ANALYSIS#####

install.packages("igraph")
#load igraph library
library(igraph)

#MR
#import edgelist data
MR_edges <- read.csv("MR_triangulation.csv")

#make a matrix from the data
MR_matrix <- as.matrix(MR_edges) # coerces the data set as a matrix

#make data character data
MR_matrix[,1] <- as.character(MR_matrix[,1])
MR_matrix[,2] <- as.character(MR_matrix[,2])

#make an edgelist
m <- graph.edgelist(MR_matrix,directed=F) # turns the edgelist into a 'graph object'

#newman's algorithm for point sizes
community.newman <- function(m) {
  deg <- degree(m)
  ec <- ecount(m)
  B <- get.adjacency(m) - outer(deg, deg, function(x,y) x*y/2/ec)
  diag(B) <- 0
  eigen(B)$vectors[,1]
}

#store newman's community algorithm
bem <- community.newman(m)

```

```

scale <- function(v, a, b) {
  v <- v-min(v) ; v <- v/max(v) ; v <- v * (b-a) ; v+a
}
#scales size of vertices based on degree of centrality
V(m)$size <- scale(abs(bem), 5, 15)

quartz()
#plot the graph
#vertex.size changes the size if each node
plot(m, vertex.label.cex=0.5, edge.arrow.size=0.1, vertex.size =
  V(m)$size, vertex.color = "paleturquoise1", edge.color="gray")

#Superior
#import edgelist data
superior_edges <- read.csv("superior_triangulation.csv")

#make a matrix from the data
superior_matrix <- as.matrix(superior_edges) # coerces the data set as a matrix

#make data character data
superior_matrix[,1] <- as.character(superior_matrix[,1])
superior_matrix[,2] <- as.character(superior_matrix[,2])

#make an edgelist
d <- graph.edgelist(superior_matrix,directed=F) # turns the edgelist into a 'graph object'

#newman's algorithm for point sizes
community.newman <- function(d) {
  deg <- degree(d)
  ec <- ecount(d)
  B <- get.adjacency(d) - outer(deg, deg, function(x,y) x*y/2/ec)
  diag(B) <- 0
  eigen(B)$vectors[,1]
}

#store newman's community algorithm
nem <- community.newman(d)

scale <- function(v, a, b) {
  v <- v-min(v) ; v <- v/max(v) ; v <- v * (b-a) ; v+a
}
#scales size of vertices based on degree of centrality
V(d)$size <- scale(abs(nem), 5, 15)

quartz()
#plot the graph
#vertex.size changes the size if each node
plot(d, vertex.label.cex=0.5, edge.arrow.size=0.1, vertex.size =
  V(d)$size, vertex.color = "lavender", edge.color="gray")

#KM
#import edgelist data
KM_edges <- read.csv("KM_triangulation.csv")

#make a matrix from the data

```

```

KM_matrix <- as.matrix(KM_edges) # coerces the data set as a matrix

#make data character data
KM_matrix[,1] <- as.character(KM_matrix[,1])
KM_matrix[,2] <- as.character(KM_matrix[,2])

#make an edgelist
k <- graph.edgelist(KM_matrix,directed=F) # turns the edgelist into a 'graph object'

#newman's algorithm for point sizes
community.newman <- function(k) {
  deg <- degree(k)
  ec <- ecount(k)
  B <- get.adjacency(k) - outer(deg, deg, function(x,y) x*y/2/ec)
  diag(B) <- 0
  eigen(B)$vectors[,1]
}

#store newman's community algorithm
dem <- community.newman(k)

scale <- function(v, a, b) {
  v <- v-min(v) ; v <- v/max(v) ; v <- v * (b-a) ; v+a
}
#scales size of vertices based on degree of centrality
V(k)$size <- scale(abs(dem), 5, 15)

quartz()
#plot the graph
#vertex.size changes the size if each node
plot(k, vertex.label.cex=0.5, edge.arrow.size=0.1, vertex.size =
  V(k)$size, vertex.color = "plum1", edge.color="gray")

#BLM-18
#import edgelist data
BLM18_edges <- read.csv("BLM18_triangulation.csv")

#make a matrix from the data
BLM18_matrix <- as.matrix(BLM18_edges) # coerces the data set as a matrix

#make data character data
BLM18_matrix[,1] <- as.character(BLM18_matrix[,1])
BLM18_matrix[,2] <- as.character(BLM18_matrix[,2])

#make an edgelist
b <- graph.edgelist(BLM18_matrix,directed=F) # turns the edgelist into a 'graph object'

#newman's algorithm for point sizes
community.newman <- function(b) {
  deg <- degree(b)
  ec <- ecount(b)
  B <- get.adjacency(b) - outer(deg, deg, function(x,y) x*y/2/ec)
  diag(B) <- 0
  eigen(B)$vectors[,1]
}

```

```

#store newman's community algorithm
fem <- community.newman(b)

scale <- function(v, a, b) {
  v <- v-min(v) ; v <- v/max(v) ; v <- v * (b-a) ; v+a
}
#scales size of vertices based on degree of centrality
V(b)$size <- scale(abs(fem), 5, 15)

quartz()
#plot the graph
#vertex.size changes the size if each node
plot(b, vertex.label.cex=0.5, edge.arrow.size=0.1, vertex.size =
  V(b)$size, vertex.color = "tan1", edge.color="gray")

#Short
el <- read.csv("Short_edge_list - Sheet1.csv")

#make a matrix from the data
meep <- as.matrix(el) # coerces the data set as a matrix

#make data character data
meep[,1] <- as.character(meep[,1])
meep[,2] <- as.character(meep[,2])

#make an edgelist
g <- graph.edgelist(meep,directed=F) # turns the edgelist into a 'graph object'

#trying to run a function that should identify communities in the network
community.newman <- function(g) {
  deg <- degree(g)
  ec <- ecount(g)
  B <- get.adjacency(g) - outer(deg, deg, function(x,y) x*y/2/ec)
  diag(B) <- 0
  eigen(B)$vectors[,1]
}

#store newman's community algorithm
mem <- community.newman(g)

#color code communities... only two communities in the example so I don't know how it would work for this
without knowing how many communities to look for
V(g)$color <- ifelse(mem < 0, "grey", "green")

scale <- function(v, a, b) {
  v <- v-min(v) ; v <- v/max(v) ; v <- v * (b-a) ; v+a
}
#scales size of vertices based on degree of centrality
V(g)$size <- scale(abs(mem), 5, 15)
E(g)$color <- "grey"
E(g)[ V(g)[color=="grey"] %--% V(g)[color=="green"] ]$color <- "red"
plot(g, layout=layout.kamada.kawai, vertex.color="a:color",
  vertex.size="a:size")

quartz()
#plot the graph

```

```

#vertex.size changes the size if each node
plot(g, vertex.label.cex=0.5, edge.arrow.size=0.1, vertex.size =
      V(g)$size, vertex.color = "darkseagreen2")

#Short random and small world networks

#small world graph
quartz()
small_world <- sample_smallworld(1,222,5, .05)
plot(small_world, vertex.label.cex=0.25, edge.arrow.size=0.1, vertex.size=5, edge.color="black",
      vertex.color="yellow")

#plot histograms of degree for small world and short
quartz()
par(mfrow=c(2,1))
hist(deg_short, breaks=seq(0,30))

#calculate variance of random network 1000 times
sd_degree_small <- rep(NA, 1000)

for (i in 1:1000) {
  mean_small <- sample_smallworld(1,222, 5,0.05)
  deg_small <- degree(mean_small)
  sd_degree_small[i] <- sd(deg_small)
}

sd_observed <- sd(deg_short)
hist(sd_degree_small, breaks = seq(0,30))
abline(v=sd_observed)

#run Newman's algorithm for both random networks
community.newman <- function(small_world) {
  deg <- degree(small_world)
  ec <- ecount(small_world)
  B <- get.adjacency(small_world) - outer(deg, deg, function(x,y) x*y/2/ec)
  diag(B) <- 0
  eigen(B)$vectors[,1]
}

#store newman's community algorithm
sem <- community.newman(small_world)

community.newman <- function(rand_prob) {
  deg <- degree(rand_prob)
  ec <- ecount(rand_prob)
  B <- get.adjacency(rand_prob) - outer(deg, deg, function(x,y) x*y/2/ec)
  diag(B) <- 0
  eigen(B)$vectors[,1]
}

#store newman's community algorithm
rem <- community.newman(rand_prob)
#color code communities... only two communities in the example so I don't know how it would work for this
without knowing how many communities to look for
V(g)$color <- ifelse(mem < 0, "grey", "green")

```

```

scale <- function(v, a, b) {
  v <- v-min(v) ; v <- v/max(v) ; v <- v * (b-a) ; v+a
}
#scales size of vertices based on degree of centrality... I think?
V(g)$size <- scale(abs(mem), 5, 15)
V(small_world)$size <- scale(abs(sem), 5, 15)
V(rand_prob)$size <- scale(abs(rem),5,15)
#plot observed network, random network, and small world network on same page
quartz()
par(mar=c(0,0.2,0.2,0),mfrow=c(1,3))
#observed
plot(g, vertex.label.cex=0.001, edge.arrow.size=0.1, vertex.size =
  V(g)$size, vertex.color = "darkseagreen2", edge.color="grey")
title("Observed Network", line=-5)
#random
plot(rand_prob, vertex.label.cex=0.001, edge.arrow.size=0.1, vertex.size=V(rand_prob)$size, edge.color="grey",
  vertex.color="light blue")
title("Random Network", line=-5)
#small world
plot(small_world, vertex.label.cex=0.001, edge.arrow.size=0.1, vertex.size=V(small_world)$size,
  edge.color="grey", vertex.color="yellow")
title("Small World Network", line=-5, cex=2)

#Waldorf
#import edgelist data
waldorf_edges <- read.csv("waldorf_triangulation.csv")

#make a matrix from the data
waldorf_matrix <- as.matrix(waldorf_edges) # coerces the data set as a matrix

#make data character data
waldorf_matrix[,1] <- as.character(waldorf_matrix[,1])
waldorf_matrix[,2] <- as.character(waldorf_matrix[,2])

#make an edgelist
f <- graph.edgelist(waldorf_matrix,directed=F) # turns the edgelist into a 'graph object'

#newman's algorithm for point sizes
community.newman <- function(f) {
  deg <- degree(f)
  ec <- ecount(f)
  B <- get.adjacency(f) - outer(deg, deg, function(x,y) x*y/2/ec)
  diag(B) <- 0
  eigen(B)$vectors[,1]
}

#store newman's community algorithm
vem <- community.newman(f)

scale <- function(v, a, b) {
  v <- v-min(v) ; v <- v/max(v) ; v <- v * (b-a) ; v+a
}
#scales size of vertices based on degree of centrality
V(f)$size <- scale(abs(vem), 5, 15)

quartz()

```



```

#plot the graph
#vertex.size changes the size if each node
plot(f, vertex.label.cex=0.5, edge.arrow.size=0.1, vertex.size =
      V(f)$size, vertex.color = "coral1", edge.color="gray")

#####

#####COMMUNITIES#####

cluster_MR <- cluster_leading_eigen(m)
quartz()
plot(m, vertex.label.cex=0.001, edge.arrow.size=0.1, vertex.size =
      V(m)$size, vertex.color = cluster_MR$membership, edge.color="gray")

cluster_superior <- cluster_leading_eigen(d)
quartz()
plot(d, vertex.label.cex=0.5, edge.arrow.size=0.1, vertex.size =
      V(d)$size, vertex.color = cluster_superior$membership, edge.color="gray")

cluster_KM <- cluster_leading_eigen(k)
quartz()
plot(k, vertex.label.cex=0.001, edge.arrow.size=0.1, vertex.size =
      V(k)$size, vertex.color = cluster_KM$membership, edge.color="gray", main="KM Communities Using
cluster_leading_eigen()")
cluster_BLM18 <- cluster_leading_eigen(b)
quartz()
plot(b, vertex.label.cex=0.001, edge.arrow.size=0.1, vertex.size =
      V(b)$size, vertex.color = cluster_BLM18$membership, edge.color="gray")
cluster_short <- cluster_leading_eigen(g)
quartz()
plot(g, vertex.label.cex=0.001, edge.arrow.size=0.1, vertex.size =
      V(g)$size, vertex.color = cluster_short$membership, edge.color="gray")
cluster_waldorf <- cluster_leading_eigen(f)
quartz()
plot(f, vertex.label.cex=0.001, edge.arrow.size=0.1, vertex.size =
      V(f)$size, vertex.color = cluster_waldorf$membership, edge.color="gray", main="Manual Burrow ID")

```