
Jillian J. Rutherford
University of New Mexico

Follow this and additional works at: https://digitalrepository.unm.edu/geog_etds

Recommended Citation

This Thesis is brought to you for free and open access by the Electronic Theses and Dissertations at UNM Digital Repository. It has been accepted for inclusion in Geography ETDs by an authorized administrator of UNM Digital Repository. For more information, please contact disc@unm.edu.
Jillian Rutherford

Candidate

Geography and Environmental Studies

Department

This thesis is approved, and it is acceptable in quality and form for publication:

Approved by the Thesis Committee:

Dr. Christopher D. Lippitt, Chairperson

Dr. Melissa Emery Thompson

Dr. Xi Gong
IN PURSUIT OF COMPARABILITY: EVALUATING INFLUENCE AND INTERACTION AMONG COMPONENTS OF THE HOME RANGE ESTIMATION PROCESS IN CHIMPANZEE DATASETS

by

JILLIAN J. RUTHERFORD
B.Sc., Animal Biology, Thompson Rivers University, 2012
A.Sc., Exotic Animal Training and Management, Moorpark College, 2016
B.Sc., Geography, University of New Mexico, 2019

THESIS
Submitted in Partial Fulfillment of the Requirements of the Degree of

Master of Science
Geography

The University of New Mexico
Albuquerque, New Mexico, USA

July, 2021
ACKNOWLEDGEMENTS

I am indebted to a number of individuals for their parts in shaping this research and facilitating my ability to pursue it. I thank Chris Lippitt for his mentorship, research guidance and his ability to so seamlessly unite scholarly drive with humility, humanity, and warmth. I also thank him for taking me on as a research assistant, a position that gave me the space and opportunity to grow my skills and the financial stability to focus on my research. I thank and credit Melissa Emery Thompson for the pivotal role she played in opening the door to pursuing research in this field, for placing in me the trust and freedom to build my skills through trying and failing along the way, and for the encouragement that has given me confidence in my ability to contribute meaningfully. I thank Xi Gong for an impeccable introduction to the field of geography, for teaching me most of what I know about GIScience, and for making it possible (and enjoyable!) to learn and write the code that this and future research hinges upon. For the many hours of stimulating discussions, shared research journeys, and collective workshopping, I thank the faculty, postdoctoral researchers, and fellow graduate students of the KCP lab group at UNM. Being a part of this group has made my research experience undeniably richer.

I thank the directors of the Kibale Chimpanzee Project, Richard Wrangham, Martin Muller, Melissa Emery Thompson, and Zarin Machanda, and KCP’s field director Emily Otali, for allowing me to access to the treasure trove of spatial data in the database and to contribute to this project. I am also indebted to the KCP field team, including current field assistants Bashil Musabe, Seezi Atwijuze, Steven Alio, James Kyomuhendo, Fred Baguma, and Wilberforce Tweheyo and assistant field manager Maggie Kobusingye, for their work in collecting millions of GPS points and transcribing/transforming enormous quantities of data daily. This work is only made possible by the knowledge, skills, and labor of these individuals. Finally, pursuing research in a land to which you have no claim comes with great privilege, and I would like to acknowledge the sacrifices and contributions of the inhabitants of the communities surrounding Kibale National Park.
IN PURSUIT OF COMPARABILITY: EVALUATING INFLUENCE AND INTERACTION AMONG COMPONENTS OF THE HOME RANGE ESTIMATION PROCESS IN CHIMPANZEE DATASETS

by

JILLIAN J. RUTHERFORD

B.Sc., Animal Biology, Thompson Rivers University, 2012
A.Sc., Exotic Animal Training and Management, Moorpark College, 2016
B.Sc., Geography, University of New Mexico, 2019
M.Sc., Geography, University of New Mexico, 2021

ABSTRACT

The home range (HR) is a fundamental manifestation of an animal’s spatial behavior, but its study has been limited by unclear, unreliable, and unstandardized methodological approaches for measurement. The lack of a comprehensive evaluation of the influence of components of the HR estimation process has hampered consensus over best practices for collecting and analysing movement data. This leads to wide variation in practice, thwarting direct comparability between HR estimates and curtailing the cohesive power of cross-study findings necessary for testing hypotheses and developing appropriate conservation and management strategies. In this thesis I explore the influence of and interaction between common methods of estimating HR and important characteristics of movement datasets, using an extensive collection of GPS data from a long-term wild chimpanzee (Pan troglodytes schweinfurthii) community. I use subsampling techniques to generate 732 movement datasets that range broadly in terms of sample size, autocorrelation, sampling duration, sampling interval, and number of tracking days. I then apply 8 methods of estimating HR to each dataset and evaluate the influence of method and data characteristics on the resulting HR size using linear mixed effects modeling. I found that the area-corrected Autocorrelated Kernel Density Estimate (AKDEc) routinely outperforms other methods of estimation and produces HR outputs least likely to contain the influence of non-spatial dataset variation, leading to the most directly comparable results. However, the AKDEc does not perform reliably on small datasets lacking sufficient autocorrelation detail, as results from extreme subsampling. My findings confirm that autocorrelation is an inherent and negotiable component of modern tracking datasets, and that attempting to eliminate it is a detrimental practice. Additionally, I found that sampling duration has the largest magnitude
effect on variation in HR estimates among characteristics, but that the effect of sampling interval is the most unpredictable. Finally, I found that direct comparisons of HR estimates are potentially feasible when performed under similar data collection strategies using the Minimum Convex Polygon or Grid Square Sum estimators but unfeasible when derived from density-based estimators. These findings point to the importance of structuring data collection to maximize the information content of a dataset, reporting critical details about data collection metrics, dataset characteristics, and HR estimator application, and applying an HR estimator that is appropriate to the data and resistant to the influence of non-spatial characteristics.
Table of Contents

List of Figures ........................................................................................................................................................................ ix
List of Tables ................................................................................................................................................................................ xi
Chapter 1 - Introduction ............................................................................................................................................................... 1
  1.1 Identifying the Problem .......................................................................................................................................................... 1
  1.2 Research Aims ......................................................................................................................................................................... 3
Chapter 2 – The Theory and Application of Home Range Estimation .......................................................................................... 5
  2.1 Overview .................................................................................................................................................................................... 5
  2.2 Home Range Theory ................................................................................................................................................................. 5
  2.3 Collecting Animal Tracking Data ............................................................................................................................................. 9
  2.4 Properties of Animal Tracking Datasets ................................................................................................................................ 12
    2.4.1 Sampling Duration and Schedule ..................................................................................................................................... 12
    2.4.2 Sample Size .......................................................................................................................................................................... 14
    2.4.3 Autocorrelation ..................................................................................................................................................................... 16
    2.4.4 Range Residency ................................................................................................................................................................. 17
  2.5 Methods of Estimating Home Range ...................................................................................................................................... 19
    2.5.1 Minimum Convex Polygon ................................................................................................................................................ 21
    2.5.2 Grid Square Sum ................................................................................................................................................................. 22
    2.5.3 Kernel Density Estimate ..................................................................................................................................................... 24
    2.5.4 Autocorrelated Kernel Density Estimate ............................................................................................................................. 28
    2.5.5 Software Considerations ................................................................................................................................................... 30
  2.6 Conclusions ................................................................................................................................................................................ 31
Chapter 3 – Evaluating Influence and Interaction Among Components of the Home Range Estimation Process in Chimpanzees ................................................................................................................. 32
  3.1 Introduction ................................................................................................................................................................................... 32
  3.2 Background ................................................................................................................................................................................... 35
    3.2.1 History of Home Range Research in Chimpanzees .................................................................................................................. 35
    3.2.2 Influences of Social Behaviour on Chimpanzee Home Range ............................................................................................... 37
    3.2.3 Methodological Considerations for Studying Home Range in Chimpanzees ...................................................................... 40
    3.2.4 Value of Home Range Answers for Future Questions ....................................................................................................... 41
    3.2.5 On Assessing “Truth” in Home Range Analysis .................................................................................................................. 42
  3.3 Materials and Methods ................................................................................................................................................................. 43
3.3.1 Study Area and Subjects ................................................................. 43
3.3.2 Data Collection and Management .................................................. 45
3.3.3 Data Selection and Permutation ..................................................... 47
3.3.4 Dataset Processing and Assessment ................................................ 50
3.3.5 Application of Home Range Estimators .......................................... 54
3.3.6 Statistical Analyses ................................................................. 57
3.4 Results ......................................................................................... 59
3.4.1 Dataset Generation and Home Range Estimation .............................. 59
3.4.2 Linear Mixed Model Output ......................................................... 62
3.4.3 Addressing Hypotheses and Predictions ....................................... 64
3.5 Discussion and Conclusions .............................................................. 80
3.5.1 Datasets and the Influence of Data Characteristics .......................... 80
3.5.2 Evaluating Estimators ................................................................. 84
3.5.3 Feasibility of Meaningful Comparisons/Translations ...................... 89
3.5.4 Recommended Guidelines .......................................................... 90
3.5.5 Assumptions, Limitations, and Future Directions ............................ 92
Appendixes ....................................................................................... 94
Appendix A: Supplementary Details ...................................................... 94
Appendix B: Scripts ............................................................................ 96
  B.1 Build PostgreSQL Relational Database (SQL) ...................................... 96
  B.2 Query PostgreSQL for Original Dataset (SQL) ................................ 98
  B.3 Query PostgreSQL for Datasets with 6-month Sampling Duration (SQL) 98
  B.4 Query PostgreSQL for Datasets with 3-month Sampling Duration (SQL) 98
  B.5 Query PostgreSQL for Datasets with 1-month Sampling Duration (SQL) 99
  B.6 Generate Permuted Datasets with 15-minute Sampling Interval (Python) 99
  B.7 Generate Permuted Datasets with 1-hour Sampling Interval (Python) .... 107
  B.8 Generate Permuted Datasets with 4-hour Sampling Interval (Python) .... 113
  B.9 Generate Permuted Datasets with 24-hour Sampling Interval (Python) ... 121
  B.10 Calculate Schoener’s Ratio (R) .................................................... 128
  B.11 Assess Range Residency (R) ....................................................... 129
  B.12 Estimate HR via Minimum Convex Polygon (R) .............................. 133
  B.13 Estimate HR via Grid Square Sum (100 m) (R) .............................. 134
B.14 Estimate HR via Grid Square Sum (200 m) (R) ............................................. 136
B.15 Estimate HR via Grid Square Sum (500 m) (R) ............................................. 138
B.16 Estimate HR via Kernel Density Estimate (href) (R) ........................................ 140
B.17 Estimate HR via Kernel Density Estimate (hadhoc) (R) .................................... 141
B.18 Estimate HR via Kernel Density Estimate (hlscv) (R) ....................................... 144
B.19 Estimate HR via Autocorrelated Kernel Density Estimate (including Select
Movement Model and Calculate Effective Sample Size) (R) ................................. 145

Literature Cited .............................................................................................................. 146
List of Figures

Figure 1. The study site, surrounding the habitat of the Kanyawara chimpanzee community (light green) situated within Kibale National Park, Uganda. Also shown is the 270 km trail system (light grey lines), national park border (thick white line), and Makerere University Biological Field Station (yellow). ................................................................. 44

Figure 2. An entity relationship diagram of the PostgreSQL database. Primary keys for each table are in bold, and table relationships are symbolized with dashed blue lines. ................. 46

Figure 3. A flowchart of the steps of data selection and permutation (purple boxes), and data processing and assessment (blue boxes). The associated smaller, lighter boxes hanging from the lower right-hand corner represent data that is extracted from the corresponding step. .... 51

Figure 4. A flowchart of steps of identifying (teal boxes) and applying (yellow boxes) HR estimators. The blue box at the beginning of the flow chart represents last step of data processing and assessment ................................................................. 54

Figure 5. Distribution of dataset sample size values. Note that the y-axis is presented on a log10 scale ................................................................. 60

Figure 6. Distribution of autocorrelation values (as measured by Schoener’s ratio). The vertical, dark purple line occurs where Schoener’s ratio equals 2, indicating the absence of autocorrelation. Values left of the line are increasingly positive autocorrelation, while values right of the line are increasingly negative autocorrelation ................................................................. 60

Figure 7. Range residency as measured through variogram analysis, mean squared distance from centre of activity (MSD) and linearity index (LI) for chimps (a) DL, (b) ES, (c) TT, (d) WL. For MSD and LI plots, the solid red line indicates the value of the original dataset and the dashed lines represent the bounds of the 95% confidence interval ................................................................. 61

Figure 8. Distribution of HR output sizes predicted by the best fitting model (black dots, jittered), across estimators. Estimated marginal means and associated confidence intervals are presented in white ................................. 65

Figure 9. Influence of sample size on predicted HR output size, by estimator. Regression lines were fitted for each estimator using the HR output size values predicted by the best fitting model ................................................................. 67

Figure 10. Influence of sample size on predicted HR output size, by estimator, fitted for only those datasets with a sample size of less than 10,000 points. ................................................................. 67

Figure 11. Influence of autocorrelation on predicted HR output size, by estimator. Note that a Schoener’s Ratio of 2 is equivalent to zero autocorrelation. As Schoener’s Ratio decreases below 2, positive autocorrelation increases, and as Schoener’s Ratio increases above 2, negative autocorrelation increases. ................................................................. 68
Figure 12. Influence of sampling duration on predicted HR output size. Estimated marginal means were extracted from each level of the interaction between sampling duration and estimator in the best fitting model. ................................................................. 69

Figure 13. Influence of sampling interval on predicted HR output size. Estimated marginal means were extracted from each level of the interaction between sampling duration and estimator in the best fitting model. ................................................................. 70

Figure 14. Distribution of estimated marginal means derived from each combination of sampling duration and sampling interval, across estimators. ................................................................. 71

Figure 15. Interaction-style plots of HR output (estimated marginal) means, faceted by sampling interval................................................................. 72

Figure 16. Interaction-style plots of HR output (estimated marginal) means, faceted by sampling duration................................................................. 73

Figure 17. Influence of tracking days on predicted HR output size, by estimator. Regression lines were fitted for each estimator using the HR output size values predicted by the best fitting model................................................................. 74

Figure 18. Pearson correlation matrix depicting the relationship strength between estimator pairs of HR outputs calculated from the same datasets. On the top are the absolute values of the correlation coefficient and the significance of the correlation represented by stars (*: p<0.05, **: p<0.01, ***: p<0.001) and also the size of text. On the bottom are bivariate scatterplots with a fitted line in red................................................................. 77

Figure 19. The two strongest correlations exist within a) the MCP100 – MCP95 estimator pair and b) the GSS500 – MCP95 estimator pair. The two weakest correlations exist within c) the KDE95href – GSS500 pair and d) the KDE95href – GSS200 pair (d). Best fit lines are displayed in solid red and the line y = x is displayed in dashed black. ................................................................. 78
List of Tables

Table 1. Twenty different dataset structures were assessed in this study, including an original dataset structure and 19 permutation structures identified from the literature. Ten permuted datasets were generated from each permutation structure, leading to a total of 191 datasets (1 original dataset and 190 permuted datasets) for each chimpanzee. ........................................ 49

Table 2. Eight HR estimators were selected for evaluation, including two from the Minimum Convex Polygon family, two from the Grid Square Sum family, three from the Kernel Density Estimate family, and one from the (area-corrected) Autocorrelated Kernel Density Estimate family. Details and shorthand for each estimator are also provided. ......................... 55

Table 3. Terms included in the linear mixed effects modelling approach to analysis, including term details................................................................. 58

Table 4. Number of datasets, and range in sample size and autocorrelation values for each subject. ........................................................................................................ 59

Table 5. Linear mixed model for HR area. The values reported here are taken directly from the model output and have not been back-transformed. Significance values for each predictor were determined through a Type III Analysis of Variance with Satterthwaite’s method. Estimates and standard error for the levels of each interaction are provided in Appendix A. 63

Table 6. A measure of the degree of variation in HR output size generated by each estimator. ........................................................................................................ 64

Table 7. Contrast analysis of the estimated marginal means of HR output size for all estimator pairs......................................................................................... 65

Table 8. Change in the (estimated marginal) mean HR output size associated with increases in sampling duration, across estimators. Only significant values (p < 0.05) are presented.... 69

Table 9. Change in the (estimated marginal) mean HR output size associated with increases in sampling interval, across estimators. Only significant values (p < 0.05) are presented. .... 70

Table 10. A summary of the response of each estimator to changes in each data characteristic. Units of measurement are as follows; sample size: ΔHR size (km²) per increase in 1000 relocations, autocorrelation: ΔHR size (km²) per decrease of 1 in Schoener’s Ratio, tracking days: ΔHR size (km²) per increase of 25% of tracking days possible, sampling duration/sampling interval/interaction: standard deviation of estimated marginal means. Dark green indicates values that are least influenced, followed by values in light green. Dark orange indicates values that are most strongly influenced, followed by values in light orange. ........................................ 75
Table 11. Correlation of HR outputs between estimator pairs. Light blue highlighting indicates pairs within the same broad estimator family (density/no-density) and dark blue highlighting indicates pairs in the same immediate estimator family.

Table 12. Kanyawara chimpanzee subjects and sample sizes for the year 2017. Rows in grey relate to the four individuals examined in this study.

Table 13. Select dataset characteristics from nineteen studies of chimpanzee HR analysis. Values listed pertain to the individual datasets analysed for HR within the study. Where a range of values is provided, multiple HR assessments were made. Where there was insufficient information to assess a characteristic, it is denoted as “-”.

Chapter 1 - Introduction

1.1 Identifying the Problem

“Regardless of field of interest, a prerequisite to studying space use and movement patterns is that they be reliably measured.” (Bekoff and Mech 1984)

The movement of an animal across its landscape is a spatial manifestation of the decisions that must be made in response to physiological, social, and ecological drivers. Taken together, these movement decisions form patterns of ranging, and the accurate characterization of these patterns can lead us to important insights about the salient factors that drive animal behaviour, including the influence of increasing anthropogenic activities and obstructions worldwide. Our knowledge about the relationships between drivers of movement, anthropogenic pressures, and ranging patterns is currently limited, in part, by unclear, unreliable, and unstandardized methodological approaches to measuring some of the most fundamental manifestations of spatial behaviour.

The full characterization of an animal’s ranging patterns typically includes an estimation of the size, shape, and location of their *home range* (HR). A large number of methodological options for assessing HRs have accrued naturally over the decades (Laver and Kelly 2008) alongside developments in theory and computing power. Advances in technology have provided many new ways to track and collect movement data, accompanied by variation in the structure and magnitude of resulting datasets. Furthermore, methods of estimating HR from these datasets vary significantly in their inherent assumptions, mathematical complexity, statistical power, and suitability for the wide variety of research questions and datasets that exist within HR research today. The conspicuous lack of consensus over which of the available methodological avenues should be pursued in any given situation may be due to poor translation from simulation studies to empirical settings, lack of access to the software or skills necessary to deploy more computationally complex techniques, and/or a reflection of the natural temporal delay between innovation and application. As a result, researchers of spatial ecology across disciplines continue to employ a wide variety of tracking and HR estimation approaches (Laver and Kelly 2008; Powell 2000), but rarely appear to give thorough consideration to the trade-offs that are incurred or
the relationships that exist amongst the many options. The inconsistency in approach thwarts
direct comparability across space and time and curtails the cohesive power of cross-study
findings, even within the same research site.

Conservation and management decisions are generally developed and applied not at
the single-site level, but at the population level, drawing information from multiple sites and
time periods in order to make large-scale recommendations (Heinicke, Mundry, Boesch,
2010). It follows that the strength of these synoptic assessments directly relates to the
accuracy and comparability of the input components. If methodological variation causes
incomparability between estimates of HR across sites and studies, a synthesis based on these
data cannot accurately represent spatial behaviour at the population level, and its ability to
inform effective conservation strategies is greatly attenuated at best and dangerously
misleading at worst.

A number of important advances have been made towards understanding the
compromises, sensitivities, and magnitude of influence among different components of the
HR estimation process, including characteristics of dataset structure and methods of
estimation (Börger et al. 2006; Harris et al. 1990; Noonan et al. 2019; Powell 2000; Seaman
et al. 1999a; Swihart and Slade 1985, 1997). While such findings indicate that many of these
factors have significant consequences, there are no contemporary and comprehensive
comparisons of the sensitivity of HR estimation methods to input data characteristics and the
relative degree of importance among factors considered. Moreover, most of these studies
were undertaken either in theoretical and simulated contexts or within select ecological
systems which may not be wholly translatable to other scenarios or species. Simulated data
must always be produced from machines that can only model details the movement patterns
that humans provide, which are currently poorly understood and may never fully capture the
range of extrinsic and intrinsic drivers of movement. Additionally, these datasets will always
lack the degree of stochasticity and ecological interference experienced by real, wild
individuals, and therefore can only approximate but not equal wild datasets. Wildlife
managers and researchers therefore still lack a comprehensive analysis of how their strategies
of data collection and method selection affect the practical estimation of HR within the real
movement datasets they capture, and how these choices may influence the usability of their results in the future. To provide a clear path forwards, we must make efforts to extend our investigations in this area across species and beyond computer simulations to data collected from free-ranging animals in a number of habitats.

In this study, I investigate the influence and interaction among components of the HR estimation process using data from a long-term chimpanzee (*Pan troglodytes schweinfurthii*) field site in East Africa, which benefits the geographic research community by situating an inquiry of the mechanisms underlying a specific form of spatial analysis within a *real-world* data scenario in order to broaden our understanding beyond the purely theoretical and strengthen confidence in the applicability of our recommendations. It also benefits the chimpanzee research community with a rigorous examination of how these mechanisms operate in their subjects and datasets, supplying information necessary for the expansion of the critical but underdeveloped study of chimpanzee spatial ecology. Furthermore, it supports our ability to maximize the utility of the global wealth of chimpanzee spatial data and improve our collective ability to conserve this endangered species. Finally, the group structure of the eastern chimpanzee (*Pan troglodytes schweinfurthii*) has commonalities with a variety of other mammalian social structures and the results of this analysis may help inform HR research in other primate species or social mammals. The contribution of this study is therefore not only additional knowledge for the fields of geography and primatology, but also a fortification of the interdisciplinary bridge that enables the flow of information between subject areas in the pursuit of fruitful discovery.

1.2 Research Aims

The ultimate goal of this body of work is to provide a foundation for understanding the influence of and interaction between components of the HR estimation process in real-world datasets. Reaching this goal will help us to maximize the global value of our results. It will generate prospective benefits by providing practitioners with the material necessary to make informed decisions when structuring the collection and analysis of movement data in the future, and it will generate retrospective benefits by guiding the appropriate interpretation
of the historical HR data available to us. To accomplish this goal, this research is designed in accordance with four central aims:

1. To illuminate and compare the behaviour of the most common methods of estimating HR across datasets with varying structures, identifying those that produce the most similar or dissimilar outcomes with respect to HR mean size and variation. Achieving this aim will lead both to guidance towards the selection of an optimal method in future studies and guidance for retrospective studies towards identifying appropriate or inappropriate result comparisons.

2. To identify the relative magnitude of influence of dataset characteristics upon HR estimates, illustrating what interrelationships may exist between these factors and how their influences operate across methods. Achieving this aim will support recommendations for best practices in the process of structuring the collection of movement data, and in selecting appropriate methods of estimation for a given dataset.

3. To guide the development of a heuristic means of converting the values of HR estimates generated by different approaches, allowing for adjusted interpretations of the results generated by previous and future studies for comparative purposes.

4. To increase the confidence with which researchers of spatial ecology can mutually agree upon the most appropriate methods of structuring data collection and analyzing HR, leading to better standardization across space, time, species and applications.

This study achieves the above aims by applying a selection of historically popular and recently developed HR estimation methods to a series of datasets generated by modifying the data characteristics of original wild chimpanzee data collected at a historic chimpanzee research site to reflect common configurations found in the chimpanzee literature. The size and nature of the consequences of HR estimation method and dataset characteristics on the size of resulting HR estimates are then analysed using linear mixed effects models and conclusions are drawn about the interrelationships between various components of the HR estimation process, and the feasibility of meaningfully translating between the results of various HR analyses.
Chapter 2 – The Theory and Application of Home Range Estimation

2.1 Overview

The task of estimating an animal’s HR appears relatively straightforward at first glance: a method of quantifying movement is applied to a given dataset, generating a spatial output that has a distinct size and clear boundaries. However, the growing body of literature devoted to the subject, spanning nearly a century, is a testament to the contrary. This chapter offers an overview of important factors that relate to conceptualizing and measuring HR and provides a foundation that validates the impetus and importance of the research carried out in Chapter 3. It begins with a brief survey of the important theoretical contributions to our understanding of the phenomenon of HR, including discussion on models of movement and questions about the validity of the HR concept across species and groups. Because the estimation of HR hinges upon a collection of animal movement data, the next section recounts the evolution of practices and technology associated with collecting animal tracking data followed by a discussion of the properties of animal tracking datasets, including how they may be intentionally or unintentionally modified by data collection practices, and what their impact is on HR outputs. This information, along with an analysis of the history, advantages, and critiques of each method of estimating HR, form the crux of the research at hand. Finally, all but the oldest forms of estimating HR rely upon modern powers of computation, often including specialized software packages across a number of platforms. Because these packages are not created equally, this chapter finishes with a brief treatment of software considerations.

2.2 Home Range Theory

The first allusions to the concept of HR were presented by Seton (1909), who suggested that “[n]o wild animal roams at random over the country; each has a home-region, even if it has not an actual home.” Early works focusing on the separate but related phenomena of “territory” made important contributions to the nascent field (Blair 1940; Carpenter 1934; Noble 1939), but a formal description of the idea did not arrive until Burt’s 1943 article, where he defined HR as “that area traversed by the individual in of food gathering, mating, and caring for young”. While it remains the most frequently cited, this
definition conveys neither the biological mechanisms by which HR operates nor any meaningful avenues for analysis or quantification, leading to confusion in both measurement and management. Some of this confusion stems from a lack of precise language, wherein the distinct characteristics of total range size, core area, presence/absence across regions, and relative time spent among regions have been muddled together or used interchangeably (Fieberg and Börger 2012). Each concept is valid for answering certain questions, but none of these measurements alone can provide a complete description of the mechanisms or drivers that contribute to the phenomenon of HR as originally defined by Burt. While an updated and universally accepted replacement definition remains elusive, the field has seen some (uneven) bursts of progress towards developing theories and applications.

Theoretical contributions to understanding HR lag far behind applications for measuring it. To explain this imbalance, Powell and Mitchell (2012) suggest that an enthusiasm for the technologies that enable large-scale location data collection has eclipsed interest in a theory explaining how biological processes interface to produce this phenomenon. While limited in number, several works are notable for their contributions to building a theoretical model describing HR. These fall into three general categories: the random walk model, the optimal foraging theory model, and the statistical model. A suite of studies have developed the “correlated random walk” structure into a mechanistic model of movements that collectively form a HR. In this approach, the movements of an animal can be mathematically modelled by a sequence of paths with different speeds, turning frequencies, and orientations, characterized with probability distributions that are influenced by certain biologically meaningful environmental stimuli (e.g. foreign scent marks) and some degree of stochasticity (Benhamou 2006; Edwards et al. 2007; Moorcroft, Moorcroft, and Lewis 2006). This approach has utility, but is mostly restricted thus far to territorial carnivores and central-place foragers (Börger, Dalziel, and Fryxell 2008). Akin to optimal foraging theory, Mitchell and Powell (2004) presented a mechanistic model linking the distribution of relevant resources across a habitat to an animal’s spatial behaviour within it. In this model, a HR results from the balance between an animal’s efforts to maximize resource acquisition (such as in the selection of high-yield patches), and the limiting costs of acquiring resources (such as the amount of energy expended through movement and antagonist interactions with conspecifics). This avenue is promising, and likely to be significantly advanced by the
incorporation of research into the spatial cognition of animals. Fieberg and Börger (2012) support this with their statement, “Home ranges are emergent, dynamic patterns of space use and memory, determined by complex interactions between individuals and their environment”, suggesting that a complete understanding of animal space use in an optimal foraging theory context will involve investigating how individual animals build and utilize mental representations of their surroundings. The statistical modelling category is closely connected with some of the most prominent HR estimation techniques, discussed later. In this approach, the spatial distribution of location data is used to produce a probability density function, which describes the relative likelihood of encountering a subject throughout its observed region (Börger, Dalziel, and Fryxell 2008). The shape and size of this function’s output are then used to appraise potential biological processes underlying HR formation. While these three distinct models provide their own value, a unified approach connecting the ideas together would greatly advance our understanding of how HRs function. Noticeably absent from any of the more developed theories is a consideration of the influence of competition, which will likely be a fruitful avenue for future exploration.

Perhaps because the theoretical underpinnings of HR are comparatively underdeveloped, Powell and Mitchell’s “technological cart before the conceptual horse” situation described above has led to an abundance of HR assessments that are mathematically sophisticated but poorly theoretically situated. The danger in failing to root an investigation of HR within a base of theory is that it becomes unclear which biological processes contributing to the manifestation of HR are actually under investigation, and therefore difficult to identify which of the many available research designs and methodologies are most suitable for adequately measuring them. Some of the first logical choices in designing an appropriate HR study, which should be heavily informed by both HR theory and ecological theory, are the selection of sampling duration, number and type of subjects, and method of estimation.

Any HR estimate should be contextualized with a temporal unit that corresponds to a biologically meaningful sampling period (Fieberg and Börger 2012), without which, the explanatory potential of the estimate is severely curtailed. Similarly, the number and demographic identity of study subjects should be selected with attention given to the unique
social, biological, and life-history characteristics of the population of interest. In many species the movement behaviours of individuals differ drastically between demographic categories (e.g. male vs. female, juvenile vs. adult, reproductive vs. non-reproductive status), such that one individual’s movement cannot be considered representative of another’s. Species also differ in the degree of spatial cohesion among groupings. Social species such as herding ungulates, swarming insects and schooling fish may display patterns of high cohesion in which members overwhelmingly move as a single unit with a uniformly shared HR. In these cases, it may be appropriate to extrapolate the movement behaviour of one individual to the whole group. However, other species exhibit fission-fusion behaviour; a form of dynamic grouping in which composition changes frequently as groups split and merge (Kummer 1971). In these cases where individuals do not demonstrate cohesive movement, it is inappropriate to extrapolate group HR from the movement behaviour of a few individuals (Fieberg and Börger 2012; Powell and Mitchell 2012).

Information about sampling duration and number/type of subjects will help inform decisions about which of the many available methods is most capable of answering the biological question at hand. Methods that describe only the area and boundaries of a HR hinge heavily on peripheral contours, and as such are informed and disproportionately influenced by the outermost data points, in which we have the least confidence (Seaman et al. 1999a). They have value in describing the extent of observed ranging, but are of limited use in understanding how animals (for whom these abstract boundaries are relatively unimportant or fuzzy at best) create, perceive, and respond to their habitats (Powell & Mitchell, 2012). Methods that generate a utilization distribution advance us further towards more complex questions about the relationship between HR and resource distribution or spatial behaviour. However, these too are incomplete measures because they oversimplify the relationship between the biological value of a region and the time spent within it by ignoring the significance of critical but seldomly-used resources while overinflating the significance of patches that are relatively valueless but in close proximity to high-use resources. These two examples highlight the importance of critically examining the nature of the HR related question being asked (be it about territoriality, habitat selection, physical activity etc.) and matching the question to an appropriate method of estimation, justified by clear and strong support from bodies of theory.
HR estimation should be considered as only the first step in the journey of exploring how animals use space and should not be confused with a complete and full depiction of this complex phenomenon. Without the addition of ancillary information such as the spatial distribution of nonspatial behaviours and environmental factors, HR metrics tell us little about why animals use space the way they do. However, it is an important first step and we must ensure that HR is adequately measured or risk beginning the journey on the wrong foot. While the focus of this thesis remains on teasing apart the influence of factors in the approach used to collect animal tracking data and calculate the area and boundaries of a HR estimate, the author acknowledges and strongly recommends that a solid foundation in HR theory and thorough assessment of the particular HR questions being asked are critically necessary before the application of these methods can be expected to yield biologically significant and insightful results. Use of the term “home range” in this study refers to the phenomenon broadly, rather than a specific definition, as the collection of HR estimators that are analysed here produce outputs that align with a number of HR conceptualizations.

2.3 Collecting Animal Tracking Data

Studies that aim to measure HR first require the collection of subject location data. The vocabulary used to refer to this type of data may include terms such as “positional estimates”, “fixes”, “relocations”, “observations” and more, but the activities of procuring such data can be collectively identified as “tracking”, which Cooke (2004) describes as “the most basic type of telemetry that involves determining where an animal is located spatially”. In some circumstances, the process of tracking is accompanied by measurements of physiological, behavioural, or energetic status, collectively termed “biotelemetry”. The techniques for collecting tracking data are varied and have evolved greatly since the inception of spatial ecology research. The earliest and most basic method, sometimes still in use today, is simple visual surveillance of subject movements in conjunction with some means of generating hand-plotted locations on a map. This method is often aided by various strategies of subject marking such as ear notching, toe clipping, tattooing and branding (Nietfeld, Barrett, and Silvy 1994), or in some circumstances with attached physical markers such as colored bands, numbered tags, or light emitters to enhance visibility (Stonehouse
More invasive tracking methods such as capture-and-release grid trapping have been historically common with small and shy animals who are difficult to observe, such as rodents (Dice and Clark 1952; Trevor-deutsch and Hackett 1980). These approaches are both limited by the ability of the human observer to physically locate a subject, which may prove challenging across large and/or heterogenous landscapes, with especially small subjects, or in animals that have cryptic movement behaviour such as nocturnal and fossorial species (Fuller and Fuller 2012; Kenward 2001). Though sometimes still employed, visual tracking and trapping have largely been replaced by more technologically aided forms.

A substantial development was introduced to the field of wildlife tracking by Cochran and Lorde (1963) in the form of very high frequency (VHF) radio telemetry, a process by which location information is obtained from an animal through the detection of radio waves (Fuller and Fuller 2012). In this system, a transmitter affixed to or implanted in a study subject radiates a signal that is picked up by a researcher holding a receiver in the direction of the subject. The signal may be translated into location data either by leading researchers directly to the subject or by enabling the researchers to triangulate the approximate location of the study subject at a distance through the intersection of three or more signal angles (Berg 2015; Fuller and Fuller 2012). VHF telemetry facilitates systematic sampling beyond what is possible with opportunistic visual observation and trapping studies, paving the way for greater flexibility in research design (Kenward 2001). More recently, GPS technology has been introduced to the tracking process. In this system, an animal-borne device receives a signal from three or more satellites within the Global Positioning System constellation and calculates the relative distance to each. The device uses a trilateration process to pinpoint its location on Earth, and stores this “fix” alongside other attributes such as date and time. Users can program GPS devices to collect fixes at regular intervals, down to the sub-second, which are either stored “on-board” the animal-borne device until retrieval at the end of the study period, or remotely relayed to receiving stations at regular intervals or in near real-time (Tomkiewicz et al. 2010). Those that store data on-board may be accompanied by programmable release mechanisms (Sager-Fradkin et al. 2008) and/or VHF tracking capabilities to aid in unit recovery, eliminating the need for secondary capture of subjects (Tomkiewicz et al. 2010). Remote transfer of data is especially useful in studies that require
immediate location estimates as input to *in-situ* research activities, such as in nest or den location, or where device retrieval would be impractical, such as in marine species.

GPS is advantageous over previous methods for a number of reasons. In comparison to visual and VHF methods, GPS systems excel at the rate and volume of data generated (Coelho et al. 2007). This is partly because they can be programmed and automated to perform a specific task at regular intervals, and partly because their remote nature facilitates the collection of location data in circumstances that would preclude human presence, such as inclement weather or inaccessible locales (Tomkiewicz et al. 2010). Of particular used to avian studies, bird-borne GPS units are capable of generating location both on the ground and in the air (Tomkiewicz et al. 2010). Additionally, GPS systems are now capable of location accuracy levels down to 5-10m, which enables research into questions of biology and movement that require greater levels of exactitude (Liu et al. 2018). As commercial markets develop smaller, more efficient, and higher-powered GPS systems, the applications for GPS in wildlife tracking will continue to expand and evolve.

Despite improvements, GPS tracking remains subject to some disadvantages, the most significant of which is satellite “visibility”. Because of its reliance upon satellite communication, GPS tracking may not be suitable for study sites with high mountains or dense canopy as these factors will block satellite signals and greatly increase location error (D’Eon et al. 2002; Dominy and Duncan 2001; Frair et al. 2010). In heterogenous habitats, variable environmental circumstances will cause location accuracy to differ between fixes, which may complicate analysis. High start-up costs can also make GPS units prohibitively expensive (Tomkiewicz et al. 2010), and there are the logistical challenges of capturing and collaring subjects who may be elusive or easily fatally distressed (D. L. Murray and Fuller 2000). It has been widely recommended that animal-borne units should not exceed five percent of the subject’s total mass (Casper 2009; D. L. Murray and Fuller 2000), and therefore the usability of currently available GPS devices is limited by the size of study subjects. These constraints, coupled with increasing scope and intensity of data needs, have driven demand for the development of devices with both reduced size and longer lifespan. Fortunately, commercial markets have risen to the challenge and improved models are released regularly.
Finally, in an era of digital data storage, there arise ethical issues and dangers relating to the collection of animal tracking data. In the first place, some species are federally protected from the activities of trapping and collaring, which prevents or impedes the application of GPS technology in these circumstances. Where data can be obtained, ethical considerations regarding access to information also exist. In the hands of unintended users, illegitimately acquired GPS tracking data may facilitate the disturbance, capture, harm, or slaughter of tracked individuals and their companions, leading to catastrophic results for the study population, potential survival of the species, and any ongoing research projects (Cooke et al. 2017). It is therefore necessary for researchers to make concerted efforts towards developing secure data management practices when collecting animal tracking data in this rapidly advancing field.

2.4 Properties of Animal Tracking Datasets

Animal tracking datasets can be characterized by properties that reflect the structure of data collection such as sampling duration, sampling schedule, size, and autocorrelation (a derivative product of the other attributes), and also by those that reflect the nature of underlying movement patterns such as range residency. A close examination of these properties is an important aspect of determining whether the collected data is sufficient to answer the posed research questions, and which methods will be most appropriate for any subsequent HR estimations. An exploration of these properties in the intended dataset is therefore a fundamental precursor to any HR research project.

2.4.1 Sampling Duration and Schedule

As discussed previously, the research design for any study of HR should consider how the biological and environmental realities of the study population relate to the ranging question(s) to be answered. Some questions will require far longer periods of study or a higher frequency of sampling in certain species or habitats than in others, and therefore sampling duration and sampling schedule should be adjusted accordingly. Sampling duration refers to the time elapsed between the first and last observations in a dataset, and has a large impact on both the measurements that directly result from the methods of estimating HR and
on the breadth of conclusions that can be drawn from the results. To encompass a sample size adequate for HR estimation, sampling duration should be informed by estimated HR crossing time, which is the average time it takes an individual moving at average species pace to traverse the linear extent of its approximate HR (Noonan et al. 2019). Those with shorter HR crossing times will visit the various regions of their HR at faster rate and require less total observation time to provide researchers with sufficient data for the confident estimation of their HR. Those who have longer HR crossing times will provide sufficient movement data at a slower rate, and consequentially require longer duration of sampling for the same task. Because HR crossing time may vary between individuals, sampling duration should encompass the temporal extent necessary for all individuals (Noonan et al. 2019). Failing to achieve the minimum required sampling duration will result in estimator-specific biases and incomplete HR outputs (Fleming and Calabrese 2016). Conversely, when sampling duration is too large there is a risk that a subject’s true HR has shifted over the course of sampling due to environmental response or migratory events (Powell 2000). This circumstance should be ruled out before proceeding or accounted for in discussions and interpretation. In order to produce biologically meaningful results, sampling duration should therefore be carefully defined and arranged to reflect the specific environmental variables, ecological patterns or life stages appropriate to the study species and great caution must be taken in extrapolating HR estimates beyond the period from which the data were collected (Fieberg and Börger 2012; Powell 2000). Notable works have suggested that ranging studies should prioritize sampling duration over other factors when attempting to allocate resources towards maximum sample size (Noonan et al. 2019).

Sampling schedule, another dataset property, refers to the frequency and daily patterning of successive observations, and subdivides into the parameters of sampling interval and number of tracking days. Sampling interval is the time elapsed between collected relocations, while number of tracking days is the count of unique dates during the sampling period. “High intensity” sampling schedules have relatively smaller and more regular sampling intervals, conducted over a high number of tracking days. These are common in studies that make use of the programming advantages and remote nature of GPS technologies, producing massive tracking datasets over short periods of time. However, not all studies are capable of or well suited to this high level of technological involvement. These
may instead employ “low intensity” sampling schedules with larger and more irregular sampling intervals, characterized by opportunistic observations that may occur over fewer tracking days, creating smaller datasets over longer periods. Each of these approaches has different ramifications for later analysis and while regular, high-frequency sampling schedules that use GPS units to produce large datasets are currently a popular approach to the design of animal tracking research, they also overwhelmingly result in datasets with some form of irregularity due to practical and logistical constraints of real-world data collection (Fleming et al. 2018; Swihart and Slade 1997). Irregular sampling frequency may also be intentionally introduced into the study design. For instance, all studies must negotiate a compromise between sampling frequency and battery life, in some circumstances leading to the implementation of “duty cycling” (Tomkiewicz et al. 2010), a programming option wherein the GPS device collects samples more frequently when the subject is suspected to be active and less frequently when the subject is likely to be asleep. The presence of irregular sampling frequency in HR studies will introduce variable levels of autocorrelation (discussed later) throughout the dataset (Swihart and Slade 1997) and has been demonstrated to cause significant bias in HR results, the direction of which is inconsistent amongst methods of estimation (Fleming et al. 2018). Fortunately, algorithmic advances now possess the capability of implementing optimal weighting schemes for HR estimation methods based on probability density functions, which effectively counter the effects of over- and under-sampling (Fleming and Calabrese 2020). The characteristic of sampling interval additionally influences the degree of autocorrelation in a dataset, as expanded upon below, and so is often modified by researchers, post-collection, through a subsampling procedure applied to the dataset. Once again, the sampling design choices should relate to the biological realities of the study subjects, and directly inform the choice of estimation method.

2.4.2 Sample Size

The size of a tracking dataset can exert a large influence on the results of a HR estimation (Bekoff and Mech 1984; Harris et al. 1990; Seaman et al. 1999b). Size is dictated by sampling duration and schedule, which are both influenced by the method of generating observations. Prior to the mid-2000s, dataset size was limited by the rate at which humans
could spot, trap, infer and record the location of a study subject, which was in turn limited by human effort, mobility, and visual acuity, generally resulting in datasets with upper limits of 200 relocations (Laver and Kelly 2008). With the advent of remote GPS systems, many human components have been removed from the active tracking process, leading to data increases of hundreds or thousands of relocations per individual, per day (Urbano et al. 2010). This is possible not only because GPS systems can track animals around the clock, in obscure locations, and in every weather condition, but also because sampling frequency can be adjusted to an almost instantaneous level. Over time, GPS-based animal tracking datasets are capable of easily accumulating millions of relocations.

A number of efforts have been made to identify the minimum sample sizes required for various HR estimation methods through empirical and computer simulations. These studies have resulted in inconclusive optimal values (n=100-200, Bekoff and Mech 1984; n=60-80, Arthur and Schwartz 1999; n=30-50, Seaman et al. 1999b; n=100-300, Girard et al. 2002; n=100, Börger et al. 2006; n=900, Boyle et al. 2009) but often conclude that “more is better” (Bekoff and Mech 1984; Seaman et al. 1999b). Recent studies have proposed that the quest to identify a minimum sample size is actually a proxy for quantifying information content (Fleming and Calabrese 2017), complicated by the fact that these two attributes do not directly covary in animal tracking datasets due to inherent autocorrelation (described below). To accurately portray the true information content of a dataset it has been suggested that we must look beyond “total” sample size and instead calculate “effective” sample size (\(\bar{N}_{area}\)). Effective sample size is a measure of the number of statistically independent relocations in a dataset (Calabrese, Fleming, and Gurarie 2016; Fleming and Calabrese 2017), and can be calculated by dividing the sampling duration by the timescale over which autocorrelation approaches zero. The latter phenomenon has been elsewhere described and referred to as “time to (statistical/biological) independence” (Lair 1987; Swihart and Slade 1997). A dataset with a very large total size but high autocorrelation may have a small effective sample size and contain less substantive information than a dataset with a small total size but low autocorrelation. Straightforward means of enacting a calculation of effective sample size are still elusive (but see Noonan et al. 2019), but properly assessing the size of a dataset should nevertheless include more than a simple glance at the number of relocations, and deserves serious consideration when designing a HR research project.
2.4.3 Autocorrelation

The first law of geography states that “everything is related to everything else, but near things are more related than distant things.” (Tobler 1970). This forms the basis of the principle of spatial autocorrelation, a nearly ubiquitous property of geographic data that measures the degree to which regions located closer in space have more similar or less similar attribute values than would be expected from a random distribution. This data characteristic is often the result of underlying ecological or sociological processes that work in a spatially non-random way to produce attribute values that change gradually and continuously across space, rather than patchily and randomly. While spatial autocorrelation is a critical factor in many ecological studies, temporal autocorrelation is the more relevant variant in datasets used for HR estimation. Temporal autocorrelation, in this context, refers to the degree to which relocations sampled closely in time will have more similar location coordinates than those sampled further apart in time, operating as a function of non-random animal movement patterns. The presence and degree of temporal autocorrelation (hereafter referred to simply as “autocorrelation”) in a tracking dataset is critically important in the study of animal HRs and plays a large role in ongoing methodological development.

In early animal tracking studies, autocorrelation was not well documented because limitations on human tracking abilities meant that datasets contained sparse relocations widely separated in time with little dependence upon one another. This encouraged some studies to suggest that autocorrelation does not exert a significant influence on the outcome of most HR estimations and therefore does not warrant specific attention (Blundell, Maier, and Debevec 2001; Fieberg 2007; Solla, Bonduriansky, and Brooks 1999). Following from this, the explicit exploratory analysis of autocorrelation in tracking datasets used to estimate HR has been rare amongst contemporary studies (Dray, Royer-Carenzi, and Calenge 2010). However, because technological advances allow for a much higher volume and temporal resolution of data to be collected, modern datasets are often much more strongly autocorrelated than those investigated in early studies, and the role of autocorrelation in the output has proved far greater (Noonan et al. 2019). The degree of temporal autocorrelation present in a dataset can be calculated using Schoener’s ratio (Schoener 1981) or Swihart and Slade’s index (Swihart and Slade 1985), and understanding its structure can lead to many important biological insights (Dray, Royer-Carenzi, and Calenge 2010). In acknowledging
that this phenomenon does exist and exerts an influence in most modern datasets, it is imperative to address its presence or risk producing skewed results. Many popular methods of estimating HR assume that data are independent and identically distributed, and it is inappropriate to use these methods with highly autocorrelated datasets because they generate significant bias in results (Noonan et al. 2019). Some suggest that we should instead seek to minimize or eliminate inherent autocorrelation through techniques such as subsampling or increasing sampling interval during collection (Swihart and Slade 1997; Worton 1987). These practices have been criticized for eliminating biologically meaningful information (Solla, Bonduriansky, and Brooks 1999), unintentionally increasing the effects of autocorrelation (Fleming et al. 2015), and forcing an obligatory trade-off between sampling interval and sample size in studies with fixed duration (Hansteen, Andreassen, and Ims 1997). Others suggest that a standardized sampling procedure be implemented in order to mitigate the bias in HR estimations caused by variation in sampling regime (Börger et al. 2006), but when subjects exhibit substantially different movement patterns (such as variation in home-range crossing time) this approach will cause variation in the autocorrelation structure of each dataset, leading once again to inconsistent bias in results (Fleming and Calabrese 2017). It is therefore often not feasible to completely eliminate autocorrelation in a tracking dataset, and we should instead seek to take advantage of the biological information it contains and pursue methods that account for and incorporate this characteristic into their results. Fortunately, such methods have recently become available and have provided a promising avenue for doing just this (Fleming et al. 2015; Fleming and Calabrese 2017).

2.4.4 Range Residency

Prior to the estimation of a subject’s HR, it is imperative to confirm a somewhat obvious but often overlooked precursor (Laver and Kelly 2008): does the data indicate that a stable HR actually exists? In other words, does the subject demonstrate any form of range residency (known also as site fidelity), or are their movement patterns coherent with nomadic wandering? If movement is observed to be restricted to a smaller area than would be produced by random wandering, range residency can be said to exist over a given time period (Spencer, Cameron, and Swihart 1990). However, this property may not be true or detectable
in all cases. Some species travel across such large areas and return so infrequently to any preferred or central location that HR stability is absent and HR estimation is not appropriate (Calabrese, Fleming, and Gurarie 2016; Morato et al. 2016). In other situations, a subject may indeed have range resident movement behaviour, but the duration or intensity of location sampling is insufficient to capture it (Fleming and Calabrese 2017; Morato et al. 2016). Additionally, due to seasonality, migration patterns, changing environmental conditions or dispersal events, subjects may show temporal variation in their movement patterns, leading to poor range residency over the study period (Isbell, Cheney, and Seyfarth 1990; Singh et al. 2012; Watts 1998; L. A. Wauters, Lens, and Dhondt 1995). In cases where range residency cannot be ascertained, HR estimation is liable to lead to the faulty measurement of random movements instead of ecologically meaningful patterns, and any conclusions about underlying phenomena drawn from these studies would be erroneous and inappropriate (Noonan et al. 2019; Spencer, Cameron, and Swihart 1990).

There exist several mechanisms by which to carry out this vital component of preliminary dataset exploration. A popular method using random walk comparisons was devised in the late 1980’s by Munger (1984) and modified by Danielson and Swihart (1987). In this method, all distances separating subsequent relocations in a dataset are recorded and assigned a random angle between 0° and 360°. Starting at (0, 0) on an X,Y plane, a movement length corresponding to the first recorded distance and associated random angle is generated, creating a new location at its terminus. This becomes the starting points for the next movement length, and so on, until a random movement path is generated. The procedure is then repeated to produce a collection of random trajectories. Mean squared distance from the centre of activity (MSD) (Calhoun and Casby 1958) and a linearity index (LI) (Bell and Kramer 1979) can be calculated for both the randomly generated and the actual trajectories. If the original dataset trajectory displays more constrained MSD and LI values than the collection of randomly generated trajectories, at an appropriate significance level, range residency can be said to exist. More recently, some studies have been using variogram analysis to visually interpret patterns of movement consistent with range residency (Morato et al. 2016; Noonan et al. 2019). In the context of animal tracking, an empirical variogram is a plot that describes the correlation between spatial separation and temporal separation for all point pairs in a dataset. This is accomplished by plotting the semi-variance function (which
calculates variability in the distance separating all point pairs as a function of the time lag that separates them) against time-lag (the temporal separation between two GPS fixes) (Fleming et al. 2014). The semi-variance function reveals the average squared distance an animal travels over a given time lag and can therefore reveal range-residency. If the semi-variance function reaches an asymptote, it indicates movement that is limited and characteristic of non-random, HR behaviour. If no asymptote is reached, the data are likely to be unsuitable for HR analysis (Calabrese, Fleming, and Gurarie 2016; Fleming et al. 2014). The time lag at which an asymptote first appears represents approximately the time needed for the subject to make several HR crossings. The empirical variogram can be produced in a statistical computing program and various components of the structure of the resulting function reveal valuable information about the spatial structure of the dataset. Applying one or both of these means of assessing range residency is an important step in the process of analyzing HR and can save a great deal of time by identifying those datasets in which HR is unlikely to truly exist.

2.5 Methods of Estimating Home Range

At the outset of any discussion on HR methodology, it should be made clear that there is no single “correct” method of estimating HR (Fieberg and Börger 2012), although there certainly are methods more or less appropriate for a given dataset. In the words of Bekoff and Merch (1984), “the "correct" method of analysis is closely related to the questions being asked and to the method used to generate the data; there does not appear to be a right or a wrong way to analyse space use or movement patterns.” Most studies aim to investigate one of two kinds of space-use questions: 1) Where has the animal actually been during the study period? or 2) What are the seasonal/annual/lifetime space requirements for this individual given their movement behaviour? This seemingly subtle difference in word choice should lead to large and fundamental differences in the methods used to answer each question. Fleming et al. (2015) and Noonan et al. (2019) both explain that the first question necessitates an estimation of occurrence, where we rely on observed data to interpolate the animal relocations that were not observed during the study period. The second question requires an estimation of range, in which we can extrapolate the future and past space needs.
of the animal from observed data. The outcomes of an examination of occurrence distribution cannot inform our understanding of future or past space use but do provide important evidence of resource use and environmental behaviours when applied in conjunction with ancillary environmental data. The outcomes of an examination of range distribution are more useful in understanding how the space requirements of an animal relate to its life history and conservation or management needs, and this will be the focus of this thesis.

What are the characteristics of an appropriate method of estimating HR (also known as an HR estimator)? It must account for the spatiotemporal characteristics of the data to be analysed, such that any inherent assumptions are fully satisfied. For instance, if an estimator assumes that data are independent and identically distributed, it is not an appropriate choice for a dataset with a highly autocorrelated spatial structure. Most HR estimators are designed for datasets with regular sampling intervals because they do not explicitly account for over- or under-sampling across the study area. The ideal estimator should make allowances for variable location error and missing information between relocations. It should also lend itself to the objective and data-driven selection of parameters, rather than requiring the kind of subjective input that seduces researchers into selecting parameters that “fit nicely” any pre-conceived ideas about HR character. Finally, it should fit the definition of range being used for the study and be capable of answering the detailed space use question that is posed by the research team (e.g. if the investigators seek to understand the relative importance of resources across a habitat, a simple delineation of external HR boundaries will not suffice).

The variety of methods currently available span the range of ability in all of these categories, and includes both geometric and statistical variants, where geometric methods consist of nonparametric assessments of space use while statistical methods rely upon an underlying probabilistic model (Fleming et al. 2015). This variation leads to degrees of appropriateness with respect to estimator choice, and it is important to select an appropriate methodology based upon research question and dataset properties, rather than tradition or convention. What follows below is a brief introduction to the most widely used geometric and statistical HR estimation methods, as well as a recently developed statistical method that addresses the problem of autocorrelation.
2.5.1 Minimum Convex Polygon

One of the oldest and simplest geometric methods for generating a HR estimate is the minimum convex polygon (MCP), also known as the “convex hull”, in which the outermost relocations of a dataset are connected by straight lines, creating an enclosed polygonal area (Hayne 1949; Mohr 1947). MCP users justify this method’s selection based upon the claims of “strict comparability” between studies made by Harris et al. (1990). It remains one of the most widely used tools for HR analysis (Laver and Kelly 2008), with recommenders including professional groups such as the IUCN Red List (IUCN 2012). This method’s attractiveness lies in its simplicity, statistically unencumbered computation, and supposed ease of interpretation. However, an extensive body of work extending almost back to MCP’s conception has heavily criticized this method as inappropriate for most questions pertaining to HR estimation.

The descriptive power of the MCP method is restricted to focus on the unstable boundary properties of a HR, eliminating the vast majority of movement data collected within these bounds from the analysis process (Powell 2000) and ignoring important details of internal structure and differing intensity of space use across the HR. It performs poorly and unreliably with small sample sizes (Anderson 1982; Bekoff and Mech 1984; Börger et al. 2006; Jennrich and Turner 1969; Seaman et al. 1999b) and is highly sensitive to outliers, extreme observations, and any error contained therein (Burgman and Fox 2003). Because HRs are more likely to be amoeboid in shape than a convenient convex polygonal structure (Burt 1943), this method has a tendency to incorporate large areas that are not part of the subject’s HR but fall within the boundaries of outlying relocations (Hayne 1949). Additionally, this method likely underestimates true HR size because relocation data are often opportunistic animal sightings rather than exhaustive censuses of animal movement across the study period, and the outermost relocations are therefore unlikely to have captured the most extreme geographic forays of the subject (Hayne 1949). An approach intended to mitigate some of these problems comes in the form of a “point-peeled” MCP estimate, in which only the innermost specified percentage (commonly 95%) of relocation data points are supplied to the MCP estimator. The intent of eliminating the outermost 5% of locations from analysis is to increase the possibility that extreme movement events such as exploratory forays or occasional sallies into extra-HR regions are eliminated from consideration,
reducing reliance on unstable movement behaviour. However, this approach neglects the possibility that all of the movement observed in the dataset may be entirely within the bounds of the space needed/used by the subject and outermost points only seem extreme due to the chance distribution of other data points collected. Moreover, determining this stable or unstable character of the most extreme relocations in a given dataset is nearly impossible without consulting the subject about their motivation, and discarding a blanket percentage of data without biologically motivated reasoning may result in a severe underestimation of the animal’s true HR. Due to these limitations, attempts to measure HR using the MCP method are liable to result in coarse, biased, or erroneous results in most cases. Burgman and Fox (2003, p.27) concisely summarize the collective assessment of those who have investigated this method thoroughly: “There is little to recommend convex hulls for range estimation.”

2.5.2 Grid Square Sum

The grid square sum (GSS) method is derived from the “quadrat count”, a broadly applied point pattern analysis. While not specific to HR estimation, it was a common approach in early HR studies. The GSS method operates on the principle that varying intensity of space use can be visualized by tallying the number of relocations within regular regional subdivisions of the study area. This is achieved by first generating a regular grid of user-defined quadrat/cell size and applying it to the study area. In early studies this often meant physically imprinting the grid upon the ground by cutting trails or laying markers (Adams and Davis 1967), which can be unfortunately prone to error in steep terrain or densely vegetated regions (Sterling, Nguyen, and Fashing 2000). In more recent studies this grid is instead superimposed upon a cartographic representation of the study area, often in conjunction with a geographic information system (Bertolani 2010; Boesch and Lehmann 2004). Each relocation collected is then plotted across this grid system and the number occurring within each quadrat is tallied and transformed into a density value. The boundary of a HR can be estimated by delineating those cells deemed “relevant” based on several factors. Relevancy has been variously interpreted to mean only those cells in which relocations actually occur (Rongstad and Tester 1969), those cells in which relocations occur and those the subject is presumed to have crossed through in order to reach these cells
(Chapman and Wrangham 1993), or those cells in which relocations occur and all of the space enclosed by these cells (Zoellick and Smith 1992). The HR area can then be calculated by multiplying the number of relevant cells by cell size. Selecting an appropriate cell size is “an arbitrary decision for which no biologically based, objective procedures are known” (White and Garrott 1990, p.168), although there have been suggestions that cell size should relate to the scale of subject movement (Adams and Davis 1967; Powell 2000). In practice, the cell sizes used in avian and mammalian studies have commonly ranged from 10 m² (Sterling, Nguyen, and Fashing 2000) to 500 m² (Boesch and Lehmann 2004; Grueter et al. 2008; M L Wilson, Hauser, and Wrangham 2001).

The GSS method improves upon the MCP method for questions concerning an animal’s relative intensity of space use across their range (Adams and Davis 1967; Horner and Powell 1990). Other advantages include a format that is ready for both computer generation and analysis (Rongstad and Tester 1969), its ability to avoid unintentionally including large un-used areas (Zoellick and Smith 1992), and its consistently more conservative results (Grueter et al. 2008). However, critics of this method stress the significant source of bias that the selection of grid cell size introduces (Anderson 1982; Grueter et al. 2008; Kool and Croft 1992; Lloyd 1967). Large cells tend to cause over-estimation of HR (Boesch and Lehmann 2004) and may obscure clustering patterns. Small cells can lead to errors of accuracy in plotting relocations but have higher levels of precision (Sterling, Nguyen, and Fashing 2000). Regardless of size, these cells do not allow for measures of partial use; if an animal is observed to be present in only a small portion of the cell, the entire cell area is added to the HR estimate (Kool and Croft 1992). This phenomenon is exacerbated in studies with large cell sizes (Rongstad and Tester 1969). Additionally, method variations that exclude non-contiguous “used” cells from the HR output risk ignoring real but spatially detached HR regions (Rongstad and Tester 1969), while those variations that include all areas enclosed by “used” cells cause the same shortfalls as the minimum convex polygon in this regard. Finally, the results of the GSS method respond significantly to sampling intensity, and therefore are increasingly biased with more sporadic observation practices (Grueter et al. 2008). The GSS method has largely fallen out of favour in recent years, giving way to methods that provide the same capability for assessing space use intensity, but at a higher resolution.
2.5.3 Kernel Density Estimate

Advances upon the basic geometric methods of estimating HR arrived in the late 1980's in the form of nonparametric statistical tools. The kernel density estimate (KDE) has been hailed as the most useful of these (Powell 2000; Seaman and Powell 1996) and was formally introduced to the HR community by Worton (1989), although its usage did not become widespread until after the work of Seaman (1993). It is among the most popular current methods of estimating HR, and its use is supported by recommendations from empirical studies (Fleming et al. 2015; Kie et al. 2010; Laver and Kelly 2008; Seaman and Powell 1996; Walter et al. 2011) and spatial ecology texts (Powell 2000). The KDE method, as described by Silverman (1986) and Worton (1989), operates by placing a bivariate probability density function (a “kernel”) over each relocation in the study area. The kernel can be visualized as a hill with a user-defined shape and size. In regions where relocations are dense, these hills “stack up” and create tall peaks, while regions with relatively sparse relocations will have fewer overlapping hills and a resulting lower profile. Once all kernels have been placed, a fine grid is computationally superimposed upon the study area and the sum of all kernel volumes at each grid intersection is calculated. The resulting continuous surface of values represents the utilization distribution, which is defined as a “two dimensional relative frequency distribution for the points of location of an animal over a period of time” (Van Winkle 1975). In other words, it is an estimate of the probability an animal will be found in a given part of its HR during the study period. Here, high values relate to areas with a greater probability of intense space use and low values indicate low likelihood of use. Isopleths of utilization intensity are generated by connecting grid intersections with equal probability values.

A HR can be quantified from this utilization distribution as “the smallest subregion which accounts for a specified proportion” of total utilization (Seaman and Powell 1996). The 95% isopleth is commonly specified as the delimiter of HR in this method (Worton, 1989). It is important to note that in the 95% KDE method all data points are considered, and a 95% boundary contour is generated from the resulting utilization distribution, while in the 95% MCP method only 95% of the data points are used to create the HR boundary. Confusion in this area has occasionally led to the incorrect application of point-peeling to datasets intended for KDE estimation. Some recommend that a more conservative approach
to boundary specification should be taken, exchanging the standard 95% isopleth for a value between 50% and 80% (Börger et al. 2006). However, this will further complicate comparative efforts and others advocate for using the 95% value for consistency (Laver and Kelly 2008). There exist a number of software options for the implementation of KDE including R (R Core Team 2020), ArcGIS® (Environmental Systems Research Institute, Inc., Redlands, CA, USA), Biotas (Ecological Software Solutions LLC, Hegymagas, Hungary), Home Range Tools (Rodgers et al. 2015) and Ranges (Anatrack Ltd. Wareham, UK).

The advantages of this method over those previously mentioned are primarily associated with its probabilistic nature, which produces a more reasonable two-dimensional estimate of the HR boundary than simply delineating the outermost relocations. The utilization distribution also adds a third dimension to an estimate of HR: intensity of use (Kie et al. 2010; Worton 1989). Unlike the grid square sum method, which also attempts to capture intensity of use, kernel methods are not susceptible to bias through grid size and placement (Seaman and Powell 1996; Silverman 1986). As a nonparametric tool, kernel methods can be suitably applied to datasets that do not fit an underlying bivariate normal distribution, as is common in most animal HRs (Seaman and Powell 1996; Worton 1989). This method also improves upon the biological relevance of HR estimates, reasonably assuming that an animal’s familiarity and use of their surroundings decrease gradually away from areas in which they are observed, rather than sharply ceasing outside the most extreme relocations (Powell 2000). Disadvantages of the general kernel density estimate fall into three large categories: bandwidth selection, autocorrelation, and sample size.

One of the most critical parameters involved in implementing the KDE method is the “bandwidth”, which determines the width of the kernel hill (Silverman 1986). Theoretical and empirical studies have demonstrated that bandwidth selection has a strong effect on model outputs (Gitzen and Millspaugh 2003; Hemson et al. 2005; Kie et al. 2010; Seaman et al. 1999b; Seaman and Powell 1996), but literature provides few guidelines to those in pursuit of the “correct” choice. Silverman (1986) explains that “[N]o objective method exists at present to tie bandwidth to biology or to location error, except that bandwidth should be greater than location error”. Some interesting notions relating bandwidth to the perceptive abilities of study subjects have been suggested by Powell and Mitchell (2012), but have yet
to be implemented in applied studies. Bandwidths that are too small result in outputs that expose positional errors and obscure broad patterns in favour of ultra-fine feature definition, while those that are too large cause over-smoothing and conceal meaningful details in ranging patterns (Powell 2000). Small bandwidths typically lead to smaller HR estimates, while large bandwidths result in larger estimates (Hemson et al. 2005). The influence of bandwidth selection occurs disproportionately in the outermost regions of the resulting probability density estimate (Seaman et al. 1999b), and this area is responsible for the resulting shape and size of the HR.

Several bandwidth selection strategies have been made available, including the reference function, least squares cross-validation and an ad hoc rule-based approach. The “reference bandwidth” (h<sub>ref</sub>) can be analytically determined for a bivariate normal distribution by calculating the square root of the mean variance of x and y coordinates and dividing the result by the sixth root of the number of points (Worton 1995). Most HR software programs can easily and quickly calculate and implement this value. However, animal ranging data is very rarely distributed in a bivariate normal pattern and therefore the use of this bandwidth generally produces non-optimal results (Horner and Powell 1990; Seaman and Powell 1996). The “least squares cross validation bandwidth” (h<sub>LSCV</sub>) is determined by identifying the bandwidth value that minimizes the mean integrated square error (MISE) between the fitted surfaces of various kernel density estimates and the surface of the true density function of the dataset. The fitted surfaces are generated by sequentially removing one point at a time from the dataset and re-calculating the kernel density estimate, while the true surface is calculated by using all data points. Generally, a linear search algorithm that tests hundreds of multiples of h<sub>ref</sub> is then applied, resulting in a bandwidth value associated with the smallest MISE. However, there are a number of variations in how to implement the calculation for finding the h<sub>LSCV</sub>, which can cause subtle but real differences in outcome that potentially affect the comparability of resulting HR estimates (Gitzen and Millspaugh 2003). A greater problem is the tendency for this method of selecting bandwidth to “fail to converge” entirely, especially when datasets are large or even moderately sized, or have a number of identical or very closely spaced points (Hemson et al. 2005). These traits are both common in GPS-generated datasets, making the LSCV method of determining bandwidth largely unusable in modern studies. Exceptions to this are the studies that have
limited and sparse relocations from animals demonstrating relatively uniform space use. Even when $h_{LSCV}$ can be determined, practical implementations of this bandwidth often reveal HR estimates that appear far from optimal based on biological knowledge of the species in question (Powell 2000). The “ad hoc bandwidth” ($h_{ad hoc}$) approach incorporates more biological interpretation and subjectivity on the part of the researcher in order to ultimately identify the bandwidth that corresponds to a contiguous HR estimate. This is achieved by manually and systematically multiplying the $h_{ref}$ bandwidth by increments of 0.1 or 0.05 and assessing the shape and contiguity of the resulting HR until a desirable output is achieved. In most cases, this desirable output occurs when the outermost isopleth is continuous, and the smallest bandwidth to achieve this is selected as $h_{ad hoc}$. This method, too, has limitations based in subjectivity and is not universally superior to the other options (Schuler et al. 2014). Further to these bandwidth selection strategies, it is possible to specify whether bandwidth is “fixed” and unchanging across the study area, or “adaptive” and responds to the local density of points (Worton 1989), although the effect of this choice appears far less consequential to results (Kie 2013; Worton 1995) than the considerations that follow.

The debate over whether autocorrelation poses a significant problem for the analysis of animal tracking data has generated some confusion about the appropriateness of kernel approaches to estimating HR. The KDE method explicitly assumes that data are independent and identically distributed (Worton 1989) but animal tracking datasets, as described previously, very rarely meet these criteria. Studies using KDE methods frequently justify their choice by citing a trio of articles which concluded that autocorrelation was an irrelevant influence upon the KDE output (Blundell, Maier, and Debevec 2001; Fieberg 2007; Solla, Bonduriansky, and Brooks 1999). However, several recent papers have taken advantage of new statistical tools for visualizing, simulating, and manipulating the degree of autocorrelation in generated datasets, leading to robust demonstrations that the KDE method severely underestimates HR when assumptions of data independence are violated (Fleming et al. 2014; Noonan et al. 2019). In their rigorous and comprehensive evaluation of the effects of autocorrelation in HR estimation Noonen et al. (2019) concluded that traditional kernel methods are “only statistically optimal when tracking data are sampled so coarsely that the data appear uncorrelated in time, or for a far longer period than the time scale over which any autocorrelation persists.”
In addition to bandwidth selection and the influence of autocorrelation, the size of a dataset has been demonstrated to significantly influence the output of the KDE method (Seaman et al., 1999). Some studies show that small datasets tend to cause overestimation (Seaman & Powell, 1996; Seaman et al., 1999), while others show the exact opposite effect (Hansteen, Andreassen, and Ims 1997). These conflicting results may be caused by a focus on total sample size rather than effective sample size, as explained previously. When effective sample size is large enough, kernel methods do indeed appear to surmount the challenges that small sample size can cause (Noonan et al. 2019). However, this necessitates the collection of samples that are orders of magnitude larger than typically generated in animal tracking studies, which is often logistically unrealistic due to limitations in GPS collar battery life. These were the problems faced by practitioners desiring a statistically derived HR output until the most recent method was introduced, which promises to mitigate or eliminate many of the concerns laid out above.

2.5.4 Autocorrelated Kernel Density Estimate

A modification on the traditional kernel density estimator was introduced by Fleming et al. (2015), who named it the “Autocorrelated Kernel Density Estimator” (AKDE). The AKDE is unique among the estimators discussed so far in that it is capable of specifically accounting for and treating data that is spatially autocorrelated, making it uniquely suited to modern animal tracking datasets (Fleming and Calabrese 2017). Rather than battling to overcome the burden of autocorrelation by implementing a subsampling procedure in order to reduce dataset autocorrelation for estimators that assume independence (Börger et al. 2006; Girard et al. 2002; Hemson et al. 2005), the AKDE takes advantage of autocorrelation in two ways. First, it utilizes the information that the autocorrelation structure represents (a relationship between past and future data points) to inform the selection of an appropriate bandwidth. Practically speaking, this is carried out with a version of the Gaussian reference function that is specifically tailored to account for autocorrelation in the dataset. Second, it uses the autocorrelation structure to fit a continuous time movement model that most closely approximates underlying patterns of movement inherent in the dataset. The most common candidate movement models are the independent and identically distributed (IID) process, the
Ornstein-Uhlenbeck (OU) process, and the Ornstein-Uhlenbeck-Foraging (OUF) process. The IID model assumes uncorrelated positions and velocities, while the OU model assumes correlated positions but uncorrelated velocities, and the OUF model assumes correlation in both positions and velocities (Noonan et al. 2019).

While the original AKDE was an improvement on the traditional KDE (by removing negative bias due to autocorrelation in the data) it was not without flaw. The incorporation of a Gaussian reference function to determine optimal bandwidth led to over-smoothing in situations with a small effective sample size, subsequently causing significant positive overestimation of the HR estimate’s area. An updated version that corrected for this area bias was soon introduced by Fleming and Calabrese (2017), who renamed it the “area-corrected Autocorrelated Kernel Density Estimator” (AKDEc). When autocorrelation is absent in the dataset, the AKDEc method will produce the same results as the traditional KDE method (Fleming et al. 2015), and can therefore be applied equally to all datasets containing varying spatial structures with no adverse effects to comparability. When autocorrelation is present in the dataset, the AKDEc possesses a variety of desirable characteristics. It has been demonstrated with simulation to more accurately reflect future animal space use patterns from autocorrelated datasets than conventional estimators and consistently outperforms them with very small effective sample sizes (Noonan et al. 2019). It can generate accurate confidence intervals, by translating uncertainty from model selection into uncertainty about the resulting HR estimate, which is an important aspect of reporting statistical results. Finally, because it addresses the autocorrelation which typically causes incomparability between datasets of different size, duration, and sampling regime, it permits a much broader range of comparative studies (Fleming and Calabrese 2017).

While it boasts many valuable advantages, the AKDEc is not without its own weaknesses. A drawback to any kernel method is the tendency for estimates to leak over impassable anthropomorphic barriers or other hard boundaries such as water/land transitions for strictly aquatic or terrestrial species (Worton 1995). The magnitude of this leaking error is generally proportional to bandwidth, which may be substantial. Empirical research suggests that the positive bias introduced by the AKDEc in studies with hard boundaries nevertheless remains an improvement upon the negative bias introduced by methods that do not account
Another factor causing concern is that the output of the AKDEc may appear foreign or untrustworthy to those used to working with the aforementioned HR estimators. Contrary to the expectations developed by many ecologists that the contours of a HR estimate should snugly fit the outermost relocations, AKDEc results may at first seem excessively large. These outputs are typically greater in size than even the outputs of the MCP method (Fleming et al. 2015), which tend to generate the largest results in comparative studies. It may take some adjustment for those accustomed to snug contours, but the AKDEc promises to better account for the long-term space use requirements of a subject (Noonan et al. 2019). Although it is relatively recent and has yet to be widely employed, current research suggests that the use of a tool such as the AKDEc within the field of HR estimation will improve our ability to make statistically rigorous predictions about future space use and incorporate this information into a HR estimate that more fully represents the nonstationary, autocorrelated, continuous movement process that underlie animal tracking datasets.

2.5.5 Software Considerations

One final note on implementing a selected method of estimating HR: not all software packages are created equal. Underlying their seemingly straightforward graphical user interface options for selecting parameters such as method, bandwidth, and isopleth, each software package relies upon a set of processes, equations, and default behaviours that may not be exactly the same. Lawson and Rodgers (1997) directly compared the output of common HR estimators from five commonly used (at the time) HR software packages and found large differences in the output despite near-identical user directions for each. These findings have been echoed in a handful of other studies (Gitzen and Millspaugh 2003; Hemson et al. 2005; Larkin and Halkin 1994). If inter- and intra-study comparisons are to be achieved, researchers must communicate not only the methods they chose, but also the software packages and relevant instructions, code, and settings specified within.
2.6 Conclusions

The information conveyed in this chapter supports the rejection of any notions that estimating HR is a straightforward and streamlined process through highlighting the plethora of variation available, the decisions to be made, and the impact of these choices and associated biases in each of the process. From the very conceptualization of what HR represents, to strategies for collecting animal movement data, through selecting a method of measuring HR and ultimately employing a given computational approach to doing so, the process of studying this phenomenon should be navigated thoughtfully. It is critical to thoroughly consider the impacts of each decision made, and carefully evaluate the outcome in the right context. There are some clear instances where the literature points to a given method or choice as the most acceptable and appropriate way forward (or at least points out those that should be avoided), but in many instances we are left with unanswered questions and an incomplete understanding of how the various components of this process work together. This leaves those outside the developmental core of this field to rely upon what is most popular rather that what is most geographically sound and biologically justifiable.

It is the aim of the following research to fill in some of these gaps by systematically analyzing the interplay of animal tracking dataset characteristics and methods of estimating HR using real data and all of the biological pressures, human errors, and associated messiness they contain. Simulation risks overlooking important features introduced by the realities of working within functioning ecological systems, complete with inconsistencies, entropy, and the unpredictability of field research. The outcome of this study is intended to help facilitate a better working understanding of the system in which the topics covered in this chapter exist, in pursuit of supporting scholars, managers, and decision makers within the fields of spatial ecology, primatology and beyond to successfully find answers to the question of HR.
Chapter 3 – Evaluating Influence and Interaction Among Components of the Home Range Estimation Process in Chimpanzees

3.1 Introduction

The wide variety of methodologies observed in contemporary HR analyses are reflected in the chimpanzee HR literature through significant variation in how chimpanzee location data are collected and processed and the methods of estimation that are subsequently applied (Amsler 2009; Basabose 2005; Bertolani 2010; Bortolamiol et al. 2014, 2016; Chapman and Wrangham 1993; Green, Boruff, and Grueter 2019; Hasegawa 1990; Herbinger, Boesch, and Rothe 2001; Lehmann and Boesch 2003, 2005; Mitani and Amsler 2003; Moore et al. 2018; C. M. Murray, Mane, and Pusey 2007; Newton-Fisher 2003; Vieira, Kerry, and Hockings 2019; Williams et al. 2004; Michael L. Wilson, Hauser, and Wrangham 2007). These processes have generated chimpanzee movement datasets of vastly different sizes, durations, resolutions, and sampling structures, and led to highly variable HR outputs. Such methodological variation obstructs direct comparability of results, which is critical for the production of knowledge surrounding chimpanzee spatial ecology, leading to conclusions about ranging that are largely restricted in relevance to the site and study from which they were produced (Amsler 2009; Bertolani 2010). Under these conditions, the chimpanzee research community cannot take full advantage of the combined substantial investment in long-term tracking data collected across equatorial Africa over the past five decades.

Some attempts have been made, using primate data (snub-nosed monkeys, Grueter et al. 2008; chacma baboons, Pebsworth, Morgan, and Huffman 2012) or chimpanzee data specifically (Vieira, Kerry, and Hockings 2019), to assess and compare the outcomes of some of the most commonly applied HR estimators. Other studies have indirectly reported on the influence of several methods and aspects of data structure in chimpanzee datasets by applying a handful of different estimators or subsampling approaches within the same analysis (Amsler 2009; Bertolani 2010). As in the HR literature as a whole, the conclusions drawn by these studies fail to reach a consensus about which method or sampling approach is most appropriate, and an explicit investigation of the factors influencing HR outcomes in chimpanzees (or other primates) remains absent.
In this thesis I address the problem of HR comparability by examining the influence of methodological choices or constraints and dataset characteristics using real data collected at a historic chimpanzee research site. I apply a selection of recently developed and historically popular HR estimation methods to a series of datasets generated by modifying the data characteristics of an original dataset to reflect common configurations found in the chimpanzee literature. Such an approach illuminates the kind and degree of consequences imposed by methods and data characteristics upon comparability, and lends to the consideration of the feasibility of meaningfully translating between the results of various HR analyses. The overarching goal of this research is to provide a richer understanding and comparison of how the main elements of the HR estimation process function alone and in concert to affect the outcome, in the ultimate pursuit of identifying how this knowledge can be used to maximize the comparability of HR estimates in both future and retrospective analyses. This research goal will be met through asking three central questions: What is the impact of estimator choice on HR outputs? How do non-spatial dataset characteristics affect HR estimation? What kind of relationships exist between the estimates generated by different methods and data structures?

I test the following hypotheses:

1) HR estimators informed most by components of the relationship between relocations in a dataset (e.g. density, autocorrelation) produce more consistent outputs across variations in dataset structure than estimators informed most by the locations of individual, peripheral relocations.

2) Because sampling interval is a large contributor to the information structure of a movement dataset, it should exert a stronger influence than other data characteristics on HR output size, and a stronger influence within HR estimators that consider fewer aspects of information structure.

3) Because of their shared mechanisms of estimation, estimators from the same family should generate a stronger relationship between their respective HR outputs than those from different families.
I predict that across data structures, MCP and GSS estimators will yield larger variation in the size of HR outputs than KDE and AKDEc estimators (Prediction 1a), and that controlling for variation in data characteristics, KDE and AKDEc estimators will tend to produce HR outputs more similar in size to one another than to or between outputs from MCP and GSS methods (Prediction 1b). I expect that the AKDEc estimator will produce HR outputs that are least influenced by sample size, autocorrelation, sampling duration, sampling interval, and number of tracking days, when controlling for other data characteristics, followed by KDE estimators, and then MCP and GSS estimators (Prediction 1c).

Aside from the influence of estimator choice, and considering all of the dataset permutations and their varying data structures, I predict that sampling interval will explain more variance in HR estimate size than sample size, autocorrelation, sampling duration, and number of tracking days (Prediction 2a). I expect increases in sampling interval to be associated with the largest decreases in HR output size in MCP and GSS estimators, followed by KDE estimators, followed by the AKDEc estimator (Prediction 2b). Furthermore, when sampling interval is small, I expect that the difference in estimator capacity to account for the associated increased information structure of datasets will be exacerbated, yielding larger variation between the size of HR outputs across estimators (Prediction 2c). Finally, I predict that the MCP estimator pair (MCP100 and MCP95), GSS estimator pair (GSS200 and GSS500), and KDE/AKDEc estimator triad (KDE95href, KDE95hadhoc, AKDEc), when applied to the same set of datasets, will demonstrate stronger pairwise linear relationships in the size of HR outputs than pairs involving estimators from different estimator families (e.g. MCP100 and GSS500) (Prediction 3a).
3.2 Background

3.2.1 History of Home Range Research in Chimpanzees

Research on the properties of chimpanzee HR began more than four decades ago and has covered a number of field sites in a variety of habitats spread across species’ range. Some chimpanzee communities have been the subject of sustained focus and multiple HR assessments, while others have only recently been analysed for the first time. Both eastern (Pan troglodytes schweinfurthii) and western (Pan troglodytes verus) subspecies are represented, although eastern studies dominate the literature. Despite increasing access to personal computing power and the growing size of chimpanzee tracking datasets, the rate at which chimpanzee HR studies are published has not steadily increased over time. Activity in this area was most abundant in the early 2000’s, during which time general HR estimates and descriptions were produced across a number of established sites. However, the previously discussed method inadequacy and HR estimate incomparability have acted as barriers to further HR investigation and hypothesis testing, and there have been comparatively few HR-related findings published since. Nevertheless, the conclusions drawn from existing literature shed light onto important characteristics of chimpanzee space use and provide a platform from which further analyses can expand.

Prior to the year 2000, variations of the MCP and GSS methods were applied at what would become the long-term, well-established field sites of Gombe (Tanzania), Mahale (Tanzania), and Kanyawara (Uganda), conducted by Wrangham (1975), Hasegawa (1990), and Chapman and Wrangham (1993) respectively. In addition to quantifying HR estimates for each community, these studies collectively uncovered some notable relationships between ranging patterns and other aspects of chimpanzee ecology including discoveries that chimpanzee range size varies with food availability, female ranging patterns change in response to the oestrous cycle, individuals have separate but overlapping core areas within a larger community HR, males frequent HR boundaries more often than females, and females appear to demonstrate more circumscribed space use than males. These last three ranging characteristics suggested that chimpanzees live in a socio-spatial arrangement in which females exist within small core areas dispersed within and bounded by a larger shared male range, a model that has been both supported and debated since. The early 2000’s saw HR analysed in more chimpanzee communities including Taï (Côte d’Ivoire), Sonso (Uganda),
Ngogo (Uganda) and Kahuzi (Democratic Republic of the Congo), as well as several re-examinations at previous sites. It was during this period that the KDE method was first applied, in association with the first explicit treatment of dataset autocorrelation (Newton-Fisher 2003). The size range of HR estimates widened with increasing variety in methodology and community characteristics. Conclusions drawn from this era include observations that western chimpanzees may range over much larger areas and demonstrate more movement similarity between the sexes than their eastern counterparts (Herbinger, Boesch, and Rothe 2001; Lehmann and Boesch 2005), community ranges are aggressively and cooperatively defended by males (Williams et al. 2004), the number of adult males can better predict HR size than fruit availability in western communities (Lehmann and Boesch 2003), clustering occurs in the individual HRs of subgroups of males which supports the existence of social bonds of a higher order than the dyad (Mitani and Amsler 2003), and clustered patterns of space use within the HR may be caused by seasonal dependence upon fragmented forest patches (Basabose 2005).

In the late 2000s, re-analyses of the Kanyawara, Gombe, and Ngogo HRs took place, most of which included the KDE method this time around. Findings from this period revealed that dominant females have smaller and more stable core areas within the group HR than those of lower rank (C. M. Murray, Mane, and Pusey 2007), and that vocalization and grouping behaviour is modified near the HR periphery (Michael L. Wilson, Hauser, and Wrangham 2007). A pair of dissertations published at the end of the decade provided the most extensive inspections of the influence of methodology on estimating chimpanzee HR to date. Amsler (2009) compared the MCP, GSS, and KDE methods of estimating HR at the Ngogo site in her dissertation, finding that the MCP and GSS methods provided similarly sized outputs, while the KDE output was much smaller. She also performed a variety of dataset subsampling procedures, and compared the HR results she found at Ngogo to those at different chimpanzee sites, using the same methods. Bertolani (2010) completed a similarly styled dissertation at Kanyawara, finding again that the various methods caused substantial difference in HR output, which is exacerbated in communities that have strongly non-convex ranging patterns. From 2011 onwards, chimpanzee HR analyses have all applied the KDE estimator, though many still include the MCP approach for “comparability” purposes. Chimpanzee sites that had been previously unstudied in terms of HR, including Nyungwe
(Rwanda) and Madina (Guinea-Bissau), were analysed for basic ranging properties during this time period (Moore et al. 2018; Vieira, Kerry, and Hockings 2019), while other studies at formerly analysed sites investigated how ranging behaviour interacts with salient variables of chimpanzee life. Research into the influence of environmental and anthropogenic factors upon chimpanzee ranging patterns concluded that the chimpanzees will willingly travel through anthropogenic elements and transitional environments that pass through their ranges (Bortolamiol et al. 2013, 2016), that extrinsic drivers of movement such as topographic variation and vegetation characteristics lead to preferred routes of travel throughout the range (Green, Boruff, and Grueter 2019), and that seasonal fluctuations in fruit availability cause changes in the intensity of use across a HR (Green et al. 2020).

While studies within the existing body of literature on chimpanzee HR have uncovered some foundational ranging characteristics and ecological relationships at their respective sites, the majority predate the availability of the AKDEc method, which has promised to overcome many of the shortfalls related to the MCP, GSS, and KDE methods. There remain a number of avenues worthy of pursuit in future research of chimpanzee HR, both in re-assessing ranges using the most acclaimed tools available, and in forging a path towards making meaningful comparisons between site and studies.

3.2.2 Influences of Social Behaviour on Chimpanzee Home Range

Chimpanzee communities form a rich social environment within which individuals must modify both their collective and individual movement behaviours in response to an ever shifting social context (Goodall 1986). Two of the greatest social influences on the study of chimpanzee HR are the fission-fusion community structure and strong territoriality at inter-community boundaries. The fission-fusion social structure of chimpanzee populations distinguishes their ranging patterns from those of solitary or highly spatially cohesive species. Seldom does an entire chimpanzee community range together in a single, unified group for long. Instead, subgroups (parties) of chimpanzees split and merge throughout the day along lines of affiliative and coalitionary relationships, reproductive pursuits (Tutin 1979), and resource availability (White and Wrangham 1988) in order to optimize group size. The fission-fusion events within chimpanzee communities are characterized by their dynamic
nature, in which the members of a single community will encounter each other on a flexible but regular basis. Occasionally, a temporary ecological condition under which high-value food items are densely concentrated, such as the flush of ripe fruit on a large fruiting tree, will drive the whole community to converge to eat and nest in the same area for several days. Rarely, permanent fission events occur in which a community splits into two or more stable groups (Feldblum et al. 2018), but these events are not typical and will therefore not be further discussed here in relation to their impact on the study of HR. The influence of fission-fusion social structure poses some unique constraints on the study of HR in chimpanzees. Unlike for herding species, a chimpanzee’s individual ranging behaviour cannot be reasonably extrapolated from the behaviour of another individual (with the exception of infants and their mothers), nor from the group as a whole. It is therefore critical that a subject’s identity be known and recorded in association with the collection of ranging data and analysed first in isolation from others before any conclusions can be drawn. However, neither can the movements of an individual be considered entirely independent from those of other community members as in solitary and isolated species. Within their highly dynamic social lives, chimpanzees are constantly responding to the changing social context around them, including the pressures on movement behaviour from reproductive objectives, socialization needs, the influence of dominance status, and the necessities of defending territory boundaries against hostile neighbours. Despite the importance of assessing range on an individual level, chimpanzee communities clearly share common habitat, and the measurement of this collective area is critical for both academic studies and the deployment of effective conservation and management strategies. Previous studies have investigated the degree to which the ranges of individuals may overlap to form community range (Lehmann and Boesch 2005; Williams et al. 2002; Wrangham 1979) but the results found at different sites have not been consistent and, to the author’s knowledge, these conclusions have not been substantiated with the much larger GPS datasets now available.

Territoriality is another important concept that shapes chimpanzee ranging behaviour and is important, though somewhat tangential, in a discussion of HR. From early on within the HR literature, scholars were careful to distinguish between the concepts of HR and territory (Burt 1943). Though both relate to inhabited space, territory is defined as an actively defended region (Noble 1939), while HR simply encompasses the area traversed by an
individual in its every-day activities. In some species these two phenomena are coincident, but for many they are not. For those species in which territory and HR do not directly coincide, territory is simply the protected part of the HR. In these cases, HRs of individuals may overlap, but their territories will not. The boundaries of a territory can only be confirmed through direct evidence of active defense, which makes the study of this phenomenon more challenging than that of HR and the resultant literature much more sparse. Chimpanzees exhibit powerful forms of territorial behaviour, in which parties regularly patrol the boundaries between their own territory and those of nearby communities. These excursions may end in violent and sometimes deadly conflicts with other groups when contact is made. Patrols and active conflict are most often engaged in by males (Watts and Mitani 2001) and, as such, male chimpanzees demonstrate a higher degree of territoriality than female chimpanzees. The likelihood of males engaging in territorial behaviour is dependent on factors such as whether or not he is alone or with others, and a numerical assessment of the other party’s relative size (Goodall 1986; Manson et al. 1991; M L Wilson, Hauser, and Wrangham 2001). Single males are highly unlikely to defend a territory against a neighbouring group of chimpanzees if alone, because the physical risks of failure are so great. Though females may occasionally join boundary patrols, and exhibit behaviour in which they attempt to indirectly drive others away from their preferred ranges through means of intimidation via social dominance strategies (C. M. Murray, Mane, and Pusey 2007), without active range defense this cannot be said to constitute the form of territoriality that has been described in the literature. Additionally, young females typically emigrate to a new community upon maturation. There is some evidence to suggest that while females (especially those with infants) seem to be most vulnerable to attack and also most likely to avoid boundary areas, even those females who are quasi-established within a given community may be able to successfully move between communities and across territorial lines (Williams et al. 2002). Therefore, the pressures of territoriality on HR are not uniform across individuals, and work differently depending on the social context and demographic status of those involved. Territorial boundaries between communities are contested areas, affected by competition for resources and reproductive opportunities, and can shift according to the success or failure of inter-community interactions (Mitani, Watts, and Amsler 2010; Michael L. Wilson et al. 2012). The increasing territory of one community can therefore
impose constraints on the ranging behaviour of individuals in a nearby community, and individual HRs may shrink away from an encroaching territorial boundary as it becomes increasingly less safe to venture near this border. For the reasons outlined above, it is important to consider the influence of territory and territoriality on HR across individuals.

3.2.3 Methodological Considerations for Studying Home Range in Chimpanzees

Studying HR in chimpanzees requires a unique approach, both in the technologies used to collect ranging data and in the methods used to analyse it. The advent of VHF and GPS technology, which has been central to the explosion of tracking data generated in other species, has had its utility attenuated in the study of chimpanzee ranging because of the many barriers that make it unfeasible to directly affix GPS collars to the chimpanzees. First, the strength and dexterity of this species equips individuals with the ability to damage or remove expensive collars with reasonable ease (Humle et al. 2011). Second, the capture and restraint procedures necessary to both attach and retrieve these collars can carry serious risks to the health (Kaur et al. 2008; Negrey et al. 2019) and wellbeing (Dore et al. 2020) of the target individual. Third, because chimpanzees rarely travel alone, non-target individuals are likely to witness and interfere in the process, creating dangerous conditions for both the researchers and the anaesthetized individual. Fourth, the ability for researchers to study primates at close proximity is reliant upon a habituation process in which the primates grow accustomed to unthreatening human presence. Serious negative interactions with humans can potentially jeopardize an entire long-term research project (Lonsdorf et al. 2014) and are therefore avoided whenever possible. Most modern studies of chimpanzee ranging therefore collect data using the GPS locations of tracking team members as a proxy for the location of the chimpanzee themselves, which is deemed suitable as a substitute because the distance between trackers and chimpanzees does not typically exceed the positional error associated with GPS units under the dense canopy of most chimpanzee habitat. Complicating this consideration is the fact that most chimpanzee research occurs at long-term field sites that have well-established conventions for data collection (GPS or otherwise), which may not be optimal for producing the kind of movement datasets that lend themselves to ready HR analysis. Tradition and protocol, unchallenged, may pose restrictions on the design of a
sampling schedule optimized for new data collection and fuel temptations to use inappropriate but existing available data. It is important to strive for best practices when designing a prospective ranging study, which may require introducing novel forms or schedules of data collection. Similarly, it is crucial to curate the most appropriate dataset from any existing database for the chosen methods of estimation, rather than accepting all data without critical reflection on its quality and suitability.

Two salient considerations in a chimpanzee HR analysis, beyond those concerning data collection, relate to the selection of a HR estimation method. The issue of autocorrelation, addressed earlier, is highly relevant to modern studies of chimpanzees, where widespread GPS data collection produces datasets that are often highly autocorrelated. Additionally, the expansion and development of human populations and the introduction of national and local managed areas has created a landscape in which many chimpanzee communities (including the one in this study) live near or directly adjacent to anthropogenic barriers such as fences, agriculture, or human settlements. It is important to consider how these barriers may exert unexpected influences on otherwise unobstructed movement patterns of the chimpanzees. This should lead us to evaluate the ability of any proposed HR estimator to account for such hard boundaries or, at the very least, should stimulate careful interpretation of HR estimates from those that cannot.

3.2.4 Value of Home Range Answers for Future Questions

There is significant value in the production of comparable and reliable HR estimates for informing unanswered questions about chimpanzee spatial ecology that will contribute to a better understanding of chimpanzee behaviour and a better capacity for protecting the species. For instance, the relationship between HR size and territory size is likely strong for this territorial species, and the size and shift of HR may reflect the changing capacity for males in a community to defend resources and other members. By extension, this capacity is predicted to have significance for the reproductive success of individuals in the community, and thus HR estimates may serve as evidence of behavioural measures or as benchmarks for comparing these measures across communities and habitats.
Because HRs are a signature of the many movement decisions individuals make in response to their environments, the size and shift of HR can also imply changes in the quality or suitability of habitat experienced by the chimpanzees. For instance, population density calculations are often used as proxies of habitat quality across animal ecology (Van Horne 1983). Evaluating habitat measures across space can help answer questions about the differential success of individuals or communities, and evaluating these measures across time provides a means by which to assess the influence of a changing climate, forest progression, or fluctuating community demographics amongst other temporally dependent phenomena. Comparable HRs are also a fundamental component of answering broad questions about the influence of habitat, social structure, or anthropogenic activity at the species-level, because without these it is impossible to discern whether associated movement behaviours are characteristic of the species as a whole or are unique to a study group.

Finally, comparable HRs form the foundation of many conservation programs targeted at vulnerable species including the chimpanzee, which is identified by the IUCN Red List as “Endangered” and “Critically Endangered” across its range due to habitat loss, habitat fragmentation and increasing human-wildlife conflicts across their range (Plumptre 2010). Pursuing the production of comparable and reliable HR estimates is important not only for the production of new academic knowledge and reliable management strategies, but will have direct impacts on our ability to mitigate and reverse rapid and catastrophic population declines.

3.2.5 On Assessing “Truth” in Home Range Analysis

An ideal study of HR would identify and produce estimates that are as close as possible to the “true” HR of the study subjects. However, it is exceedingly difficult (outside of computer simulation studies) to qualify estimates generated from a given method as true or not because we still lack an understanding of how movement data collected on the variety of schedules that exist relate to a subject’s actual and perceptual HR. For the reasons outlined in Chapter 2, it is not enough to compare a HR output to the distribution of data points and simply evaluate those with good overlap as “true”. This missing benchmark prevents us from meaningfully pursuing most questions of accuracy in studies using real-world empirical data.
Fortunately, many questions involving HR are focused less on the absolute size of a given range and more on variations in range size across individuals, environmental conditions, and time. For these questions, it is most important that the primary source of HR variability stems from differences in movement patterns rather than differences in methodology. As put by Börger et al. (2006, p.1395), “the fact that estimated HR size is not the ‘true’ value is of less importance than any artefact in the variance structure of the data, i.e. the ‘true’ variance differences within and between individuals”.

The investigation of accuracy and the quest for increasing precision in HR estimates is not within the goals or purview of this study. Instead, this study is designed to address the behaviour of different HR estimators under varying data structure conditions, and the influence and interaction between different components of the estimation process on HR outcomes. Following the assertion made by Wauters et al. (2007), “[a] reliable HR estimator should produce similar HR estimates over different subsamples taken from the same source distribution of fixes…”. In other words, an estimator’s reliability is evidenced by its tendency to produce similar HR outputs from data representing the same underlying movement pattern, despite variation in data structure. In combination with this knowledge, identifying the strength and nature of influences exerted by different components of data structure across HR estimators will allow us to comment on which methodologies are most and least likely to produce reliable and comparable estimates.

3.3 Materials and Methods

3.3.1 Study Area and Subjects

The subjects of this study belong to the Kanyawara chimpanzee community (n=54-56 c. 2017), which resides in the north-west region of Kibale National Park in western Uganda (0°31’- 0°37’N and 30°19’- 30°25’E) (Figure 1). The terrain in this region consists of moderately steep hills and valleys situated around 1500 m above sea level, with vegetation comprised of primary deciduous forest interspersed with patches of grassland, swamp, and secondary forest (Chapman and Wrangham 1993). The study area is navigable by 270 km of dirt roads and semi-permanent trails, monitored and managed by the Makarere University Biological Field Station and the Ugandan Wildlife Authority. The Kanyawara chimpanzee
community is bounded to the west by the park border, beyond which lies an established agricultural community inhabited by citizens of Uganda’s Kabarole District. To the north, east, and south, they are surrounded by protected forest containing several other chimpanzee communities (both habituated and unhabituated). The Kanyawara chimpanzees have been studied since 1983 (Isabirye-Basuta 1988), and are individually identifiable and habituated to researcher presence. A considerable long-term database of genealogy, endocrinology, movement and behaviour is generated and maintained by Ugandan and US-based members of the Kibale Chimpanzee Project, established by Dr. Richard Wrangham in 1987.

Figure 1. The study site, surrounding the habitat of the Kanyawara chimpanzee community (light green) situated within Kibale National Park, Uganda. Also shown is the 270 km trail system (light grey lines), national park border (thick white line), and Makerere University Biological Field Station (yellow).
3.3.2 Data Collection and Management

3.3.2.1 Collecting the Chimpanzee Relocation Dataset

The Kibale Chimpanzee Project (KCP) has collected and stored GPS relocation data from the Kanyawara chimpanzee community since 2009, by tracking the chimpanzees on a daily and year-round basis. Whenever possible, chimpanzees are observed and followed from the time they leave their nests in the morning until they make new nests in the evening. The tracking team locates chimpanzees each morning by visiting nesting sites from the previous evening, by following calls and signs of movement, or by waiting near fruiting trees. The team then identifies a focal individual from the first party they encounter and attempts to remain with that individual during any fission or fusion events later on. During tracking, a member of the team will activate a Garmin GPSMap 64s handheld device to autogenerate GPS fixes while other members work cooperatively to record detailed information on chimpanzee IDs, party composition, health and reproductive status, and environmental and behavioural interactions.

The KCP GPS database contains some uneveness and gaps of information, caused by variation in the frequency with which GPS units have been deployed over the years, and variation in the frequency with which GPS fixes are collected over the course of a day. These irregularities stem from a number of causes that are largely unavoidable under the conditions of tropical field work including device malfunction, data loss and corruption, poor satellite reception and subject inaccessibility. Nevertheless, the total size of this data collection is immense compared to other field sites, exceeding 800,000 GPS fixes. Because a single fix is normally associated with the presence of more than one (and up to 56) individual chimpanzees, we can multiply each fix by the number of associated individuals to produce the total number of unique chimpanzee relocations. The number of unique chimpanzee relocations in the KCP GPS database exceeds 8,000,000. The data used in this study draw from this database, in addition to identification details and chimpanzee biographical information stored and maintained in other KCP databases.
3.3.2.2 Storing and Managing the Chimpanzee Relocation Dataset

To effectively work with the quantity and diversity of data in this study, it was first necessary to implement a data management system capable of accepting complex queries and handling spatial information. For this purpose, a relational database management system using the open-source PostgreSQL software was most appropriate because this software is ACID (atomicity, consistency, isolation, durability) compliant and uses the SQL language to navigate powerful tasks and large, complicated workloads. Most importantly, it inherently accepts non-spatial data and can also accept spatial data through the extension PostGIS. Figure 2 illustrates the structure of this database.

![Figure 2](image.png)

Figure 2. An entity relationship diagram of the PostgreSQL database. Primary keys for each table are in bold, and table relationships are symbolized with dashed blue lines.

3.3.2.3 Ethical Considerations

All methods used to collect the data in this study were performed in accordance with standard protocols approved by the Institutional Animal Care and Use Committees at Harvard University and the University of New Mexico (no. 19-200862-MC) and comply with the best practices for field primatology recommended by the American Society for Primatologists. Appropriate permits for study were obtained from the Uganda National Council for Science and Technology, the Uganda Wildlife Authority, and Makerere University Biological Field Station.
3.3.3 Data Selection and Permutation

3.3.3.1 Selecting Data from the Long-Term Database

This study uses data extracted from the KCP database from the year 2017. The temporal extent of data selection was set at twelve months to capture a full seasonal cycle of environmental conditions while reducing the risk of encompassing true HR shifts over a longer period. The year 2017 was a suitable choice because it is the largest and most evenly recorded annual collection currently available in the database, containing a total of 200,580 GPS fixes collected across 257 tracking days, ranging from 3 to 1,711 relocations per day (mean = 780 relocations/day). The total number of unique relocations for this year is 1,520,180. Because the extended juvenile period characteristic of chimpanzee life history conflates the movement of mothers and their dependents until late in development, only adult chimpanzees were considered in this study. The Kibale Chimpanzee Project considers males to reach adulthood at age 15, and females to reach adulthood upon their first maximally tumescent oestrous swelling (age 10-13) (Emery Thompson et al. 2020). Based upon these criteria, 27 adult individuals were encountered and tracked during the year 2017. The number of relocations per adult individual ranges from 45,469 – 87,815 (mean = 68,562) for males and 12,740 – 70,181 (mean = 53,760) for females. Because it is unfeasible within the time limits of this study to analyse ranging for all 27 adult chimpanzees, four of the most highly tracked individuals were selected and their relocations were extracted into four individual datasets: ES (male, 87,815 relocations), TT (male, 73,179 relocations), DL (female, 79,129 relocations), and WL (female, 70,181 relocations). In addition to being highly tracked, these individuals were selected because they represent a broad range of social and demographic positions within the group, which likely translate to a wide array of movement behaviours. At the time these data were collected ES was the highest-ranking male, TT was a low-ranking male, DL was a recent immigrant and young nulliparous female, and WL was an established parous female. Each of the four original individual datasets was subjected to all of the data processing, assessment, and permutation tasks described below. For more details about the number of relocations and tracking days associated with each adult individual in the KCP database, see Appendix A.
3.3.3.2 Permuting the Original Individual Datasets

Because a central goal of this study is to explore the influence of variation in the non-spatial characteristics of movement datasets rather than the influence of differences in underlying movement patterns, the approach taken here was to generate a series of new and “permuted” datasets that mimic various data collection strategies from each original individual dataset. The original and permuted datasets belonging to each individual will therefore all reflect the same underlying movement pattern of that individual yet vary substantially in their data characteristics. The task of creating permuted datasets was accomplished through a series of selective queries against the original individual datasets designed to alter sampling duration and sampling interval (see below for details). When sampling duration and sampling interval are modified, other dataset characteristics including sample size, degree of autocorrelation, and number of tracking days will vary inherently as a result, and the relationship between any of these characteristics within a dataset (alone or in concert) and the associated HR outputs can be assessed.

Previously published analyses of chimpanzee ranging provide a guide to identifying the permutations that best reflect the range of common datasets in the chimpanzee HR literature to date. To this end, I reviewed 19 studies of chimpanzee HR conducted between 1975 and 2019, and for each study assessed the sample size (number of relocations), sampling duration, sampling interval, and number of tracking days of the datasets used to generate HR estimates. Six of the studies provided sufficient detail for the assessment of all the above characteristics, while 13 failed to furnish adequate detail for one or more characteristics. Significant variation existed across each characteristic where reported; the total number of relocations ranged from 18 to 30,543, the sampling duration ranged from 1 month to 15 years, the sampling interval ranged from 15 minutes to 110 hours, and the number of tracking days ranged from 3 to 2,888. For more details on the data characteristics extracted from each study, see Appendix A. Using this review as a guide, 19 approaches to permuting sampling duration and sampling interval within each of the four original individual datasets were selected and applied (Table 1).
Table 1. Twenty different dataset structures were assessed in this study, including an original dataset structure and 19 permutation structures identified from the literature. Ten permuted datasets were generated from each permutation structure, leading to a total of 191 datasets (1 original dataset and 190 permuted datasets) for each chimpanzee.

<table>
<thead>
<tr>
<th>Permutation ID</th>
<th>Sampling Duration</th>
<th>Sampling Interval</th>
<th>No. Datasets</th>
</tr>
</thead>
<tbody>
<tr>
<td>original</td>
<td>1 year</td>
<td>1 second</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1 year</td>
<td>15 minutes</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>1 year</td>
<td>1 hour</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>1 year</td>
<td>4 hours</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>1 year</td>
<td>24 hours</td>
<td>10</td>
</tr>
<tr>
<td>5</td>
<td>6 months</td>
<td>1 second</td>
<td>10</td>
</tr>
<tr>
<td>6</td>
<td>6 months</td>
<td>15 minutes</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>6 months</td>
<td>1 hour</td>
<td>10</td>
</tr>
<tr>
<td>8</td>
<td>6 months</td>
<td>4 hours</td>
<td>10</td>
</tr>
<tr>
<td>9</td>
<td>6 months</td>
<td>24 hours</td>
<td>10</td>
</tr>
<tr>
<td>10</td>
<td>3 months</td>
<td>1 second</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>3 months</td>
<td>15 minutes</td>
<td>10</td>
</tr>
<tr>
<td>12</td>
<td>3 months</td>
<td>1 hour</td>
<td>10</td>
</tr>
<tr>
<td>13</td>
<td>3 months</td>
<td>4 hours</td>
<td>10</td>
</tr>
<tr>
<td>14</td>
<td>3 months</td>
<td>24 hours</td>
<td>10</td>
</tr>
<tr>
<td>15</td>
<td>1 month</td>
<td>1 second</td>
<td>10</td>
</tr>
<tr>
<td>16</td>
<td>1 month</td>
<td>15 minutes</td>
<td>10</td>
</tr>
<tr>
<td>17</td>
<td>1 month</td>
<td>1 hour</td>
<td>10</td>
</tr>
<tr>
<td>18</td>
<td>1 month</td>
<td>4 hours</td>
<td>10</td>
</tr>
<tr>
<td>19</td>
<td>1 month</td>
<td>24 hours</td>
<td>10</td>
</tr>
</tbody>
</table>

Total no. datasets per individual: 191

What follows is a description of the process by which the datasets were generated, and this process was repeated for each chimpanzee individual. I first accessed the original dataset, which has a one-year sampling duration and a one-second minimum sampling interval, through a simple query of the PostgreSQL database for all relocations belonging to the target individual during the year 2017. To generate permutation structures one through four I supplied the original dataset to a Python script that extracts only those relocations separated by the specified sampling interval and produced ten different datasets for each structure by randomly varying the start time of the sampling interval. To generate permutation structures five through nine I first supplied a query to the PostgreSQL database to extract all relocations within a six-month period during the year 2017, randomly varying the start date of this period ten times to produce ten different datasets with a six-month
sampling duration and one-second minimum sampling interval. I then submitted these ten datasets to the Python script above (with some modifications to eliminate the varying sampling interval start time) to generate datasets with their respective sampling durations and sampling intervals. To generate permutation structures 10 through 19 I followed the method outlined for permutation structures five through nine but specified a three-month or one-month duration instead of a six-month duration in the initial PostgreSQL queries. The final collection of unique datasets per individual (n=191) therefore includes one original dataset and ten repeat datasets for each of the nineteen permutation structures. Because this study examines four individual chimpanzees, the total number of unique datasets possible was 764. Extremely small datasets have been noted elsewhere as tending to be unreliable with a number of different HR estimation methods, and so I applied a decision rule to eliminate from further assessment all datasets with less than ten relocations.

3.3.4 Dataset Processing and Assessment

After applying the permutation process outlined above, the original and permuted datasets required exposure to a formatting scheme to be usable with the various software packages applied in later steps, followed by a series of preliminary analyses to examine and describe several important dataset characteristics (outlined below) for the final goal of ascertaining the appropriateness of pursuing questions about HR in each. These tasks were made possible using the Movebank online movement data archive service (Kranstauber et al. 2011), and the rhr (Signer and Balkenhol 2015) and ctmm packages (Fleming and Calabrese 2020) in the R statistical computing software (v 3.6.1) (R Core Team 2020). The methodology for the tasks of data processing and assessment can be visualized by the flowchart in Figure 3.
Figure 3. A flowchart of the steps of data selection and permutation (purple boxes), and data processing and assessment (blue boxes). The associated smaller, lighter boxes hanging from the lower right-hand corner represent data that is extracted from the corresponding step.

3.3.4.1 Formatting Datasets in Movebank

Most of the HR estimation methods identified as relevant for this study require the use of R packages that have specific formatting requirements for input data. These packages are designed to accept data generated by Movebank, which is an online database and web application hosted by the Max Planck Institute of Animal Behavior. Movebank was created in an effort to provide a common template for storing and sharing movement data, enabling researchers to upload movement data in a wide array of input formats, structure and standardize this data for maximal efficiency, securely archive, manage, analyse and selectively share the information, and download selected data from the site in one of several standardized formats. I uploaded all 732 original and permuted datasets generated in the section above to the Movebank platform, formatted them using a standardized design, and downloaded them as Movebank-modified CSV files before importing them directly into the program R.
3.3.4.2 Assessing Autocorrelation

A common means of assessing the degree of temporal autocorrelation in a movement dataset is the calculation of Schoener’s ratio (Hansteen, Andreassen, and Ims 1997; Schoener 1981; Swihart and Slade 1985, 1997). Schoener’s ratio, $t^2/r^2$, is a bivariate scalar measure of temporal autocorrelation in a temporally ordered set of relocations, given in the equation:

$$\begin{align*}
t^2 &= \frac{1}{m} \sum_{i=1}^{m} (x_{i+1} - x_i)^2 + \frac{1}{m} \sum_{i=1}^{m} (y_{i+1} - y_i)^2 \\
r^2 &= \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2 + \frac{1}{n-1} \sum_{i=1}^{n} (y_i - \bar{y})^2
\end{align*}$$

Eq. 1

where $r^2$ is the squared mean distance between successive relocations, $r^2$ is the squared mean distance between each relocation and the dataset centroid, $n$ is the number of relocations in the dataset, $m$ is the number of pairs of successive relocations, $i$ is the order of successive relocations, $x$ and $y$ are the coordinates of each relocation, and $\bar{x}$ and $\bar{y}$ are the coordinates of the centroid. The quotient of $t^2$ and $r^2$ is compared to the theoretical value of 2, which occurs under the hypothesis of independent and identically distributed data (Swihart and Slade 1985). Positive autocorrelation is present when $t^2/r^2 < 2$, whereas negative autocorrelation is present when $t^2/r^2 > 2$. I applied Schoener’s ratio to each dataset using a purpose-built R script.

3.3.4.3 Assessing Range Residency

Because the permuted datasets generated previously reflect the underlying movement pattern of the original dataset from which they were derived, establishing range residency was only necessary in each of the four original datasets, and provides sufficient evidence of an individual’s site fidelity in order to warrant the application of HR estimators across datasets. To ensure that the condition of range residency was fully met before proceeding, I took two separate approaches to evaluate this characteristic. The first approach stems from the guidelines provided by Spencer et al. (1990) and promoted by Laver and Kelly (2008) that combine random walk modelling with metrics of dispersion. To accomplish this, I wrote an R script that first generates a series of 100 random trajectories from each original
individual datasets using the procedure described in Chapter 2, Section 2.4.4. After calculating the mean squared distance from the centre of activity (MSD) and linearity index (LI) values for each of the random trajectories and original individual dataset, the script produces a graph to enable an assessment of whether the MSD and LI values of the original dataset were significantly different than the range of those values produced from the randomly generated trajectories. This workflow was made possible using the rhr package. The second approach derives from work of Calabrese et al. (2016). To accomplish this, I wrote and ran an R script that uses the functions of the ctmm package to plot the estimated location variance against time-lag and generate a variogram, finally producing a graph to enable the inspection of the resulting variogram curve for the presence of a clear asymptote.

3.3.4.4 Selecting Movement Models

To apply the AKDEc HR estimator it is imperative to fit and select a movement model to each dataset. Continuous-time stochastic movement models are appropriate for a wide variety of ranging datasets because they can reflect movement processes that have no autocorrelation, those with solely positional (Markovian) autocorrelation, and those with multi-scale positional and velocity autocorrelation. The movement models considered here are independent and identically distributed (IID), Ornstein-Uhlenbeck (OU), and Ornstein-Uhlenbeck with Foraging (OUF) (Fleming et al. 2014). The IID model represents a completely random distribution and will be considered to serve as a null condition. OU movement is a form of the classical diffusion expressed in Brownian motion, but is modified to limit the extent of movement to a finite area with a central location to which the subject has a tendency to continually return. OUF is a more complex extension of OU movement that includes a parameter describing random periods of foraging with typical duration, which captures autocorrelation in velocity. Using the ctmm package, I wrote and ran an R script that calculated the fit of each of these movement models to the empirical variogram produced from each original and permuted dataset in prior steps, and selected the best model as the one with the lowest associated Akaike Information Criteria (AIC) value.
3.3.5 Application of Home Range Estimators

After selecting, permuting, formatting, processing, and assessing each of the datasets, the steps of identifying and applying several methods of estimating HR are able to proceed. The methodology I followed for this section can be visualized by the flowchart in Figure 4.

Figure 4. A flowchart of steps of identifying (teal boxes) and applying (yellow boxes) HR estimators. The blue box at the beginning of the flow chart represents last step of data processing and assessment.

3.3.5.1 Choosing Home Range Estimators to Assess

The exhaustive review of HR studies undertaken by Laver and Kelly (2008) provides a broad survey of the most common methods of estimating HR within the field of ecology at the time of publishing. A number of studies confirm the application of many of these estimators in chimpanzee research (Amsler 2009; Bertolani 2010; Chapman and Wrangham 1993; Green, Boruff, and Grueter 2019; Herbinger, Boesch, and Rothe 2001; Kouakou, Boesch, and Kuehl 2011; Newton-Fisher 2003; Riedel, Franz, and Boesch 2011; Vieira, Kerry, and Hockings 2019). I selected eight estimators to be evaluated in this study based on their ubiquity, popularity, and/or significance in relation to research surrounding chimpanzee movement and HRs (Table 2).
Table 2. Eight HR estimators were selected for evaluation, including two from the Minimum Convex Polygon family, two from the Grid Square Sum family, three from the Kernel Density Estimate family, and one from the (area-corrected) Autocorrelated Kernel Density Estimate family. Details and shorthand for each estimator are also provided.

<table>
<thead>
<tr>
<th>Estimator Family</th>
<th>Estimator Details</th>
<th>Estimator Shorthand</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Convex Polygon</td>
<td>100% relocations considered</td>
<td>MCP100</td>
</tr>
<tr>
<td>Minimum Convex Polygon</td>
<td>Inner 95% relocations considered</td>
<td>MCP95</td>
</tr>
<tr>
<td>Grid Square Sum</td>
<td>200 m x 200 m grid cells, occupied only</td>
<td>GSS200</td>
</tr>
<tr>
<td>Grid Square Sum</td>
<td>500 m x 500 m grid cells, occupied only</td>
<td>GSS500</td>
</tr>
<tr>
<td>Kernel Density Estimate</td>
<td>95% contour, fixed smoothing, bandwidth determined by reference method</td>
<td>KDE95href</td>
</tr>
<tr>
<td>Kernel Density Estimate</td>
<td>95% contour, fixed smoothing, bandwidth determined by ad hoc method</td>
<td>KDE95hadhoc</td>
</tr>
<tr>
<td>Kernel Density Estimate</td>
<td>95% contour, fixed smoothing, bandwidth determined by LSCV method</td>
<td>KDE95hlscv</td>
</tr>
<tr>
<td>Autocorrelated Kernel Density Estimate (area-corrected)</td>
<td>unweighted</td>
<td>AKDEc</td>
</tr>
</tbody>
</table>

3.3.5.2 Applying Home Range Estimators

The package adehabitat in R v 3.6.1 contains an mcp() function which allows users to carry out a HR estimation using the MCP approach. I applied both MCP estimators considered here (MCP100 and MCP95) using a purpose-built script which first imports each original and permuted dataset and provides users with a parameter in which to specify the percentage of relocations to consider, and by extension, the percentage of outliers to exclude. Outliers here are classified according to their distance from the centroid of the HR, which itself is computed from the arithmetic mean of all coordinates. For each MCP estimator, I therefore specified the percent parameter as 100 or 95, accordingly. I then ran the script to apply both estimators to all datasets, producing polygons for each HR and saving an automatically calculated HR area output from each polygon to a CSV file.
Carrying out a HR estimation using the GSS approach does not require the use of any specialized packages within the R v 3.6.1 environment. Rather, it is possible using R base libraries. I wrote and ran a simple script to apply both of the GSS estimators considered here (GSS200 and GSS500) to all original and permuted datasets. The script first generates a grid covering the study area, with a parameter to specify appropriate grid cell dimensions (200 m x 200 m, or 500 m x 500 m accordingly). A spatial join is then performed between this grid layer each dataset, imported as a relocations layer, identifying which grid cells are intersected by relocations. HR size is then calculated by multiplying the number of “occupied” grid cells by the cell dimension.

The adehabitatHR (Calenge 2006) package in R v 3.6.1 offers users the kernelUD() function as a means of applying the three KDE estimators considered here (KDE95href, KDE95hadhoc, and KDE95hlscv). To produce HR estimates from each dataset using the KDE95href estimator I wrote and ran a script that creates a utilization distribution from a dataset by supplying the “href” argument to the bandwidth parameter of the kernelUD() function. It then generates a HR outline by extracting the 95% volume contour using the getverticeshr() function, converts the outline to a polygon, and calculates HR area from this polygon. To apply the KDE95hadhoc estimator, I wrote and ran a script that generates a series of utilization distributions from each dataset by supplying the kernelUD() function with sequentially adjusted reference bandwidths, reduced or increased by increments of 0.10. I then visually identified the “optimal bandwidth” as the smallest bandwidth that produces a contiguous HR polygon lacking any lacunae, following the protocol outlined by Kie (2013). This user-input bandwidth is then supplied to the kernelUD() function to produce a utilization distribution and generates a HR outline by extracting the 95% volume contour. The outline is converted to a polygon and HR area is extracted. To apply the KDE95hlscv estimator, I wrote and ran a script that follows the same process as the KDE95href estimator but instead supplies the “LSCV” argument to the bandwidth parameter of the kernelUD() function.

The area-corrected Autocorrelated Kernel Density Estimate (AKDEc) approach to estimating HR is made possible through the ctmm package in the R v 3.6.1 environment. I wrote and ran a script that imports each dataset in CSV format, converts it to a telemetry
object that serves as input to the variogram() function, fits a correlated movement model to
the empirical variogram produced from the dataset, and supplies the chosen model
information to the akde() function in order to generate a utilization distribution. It then
creates a polygon from this utilization distribution and extracts HR area from said polygon.

3.3.5.3 Evaluating Estimator Output

Following preliminary analyses of the area values derived from HR estimates across
methods, I implemented a decision rule to eliminate HR estimates that occurred at the
extreme tail ends of the distribution of values because these would likely be rejected in any
modern study as illogical outliers and artifacts of poor estimation methodology. This decision
rule, in which the top and bottom one percentile of estimates were discarded, effectively
eliminated from further analyses those HR estimates that were clearly beyond the bounds of
reasonable outputs for this community, such as those at the sub-square kilometre level and
those exceeding 50 km².

3.3.6 Statistical Analyses

To assess which of the factors manipulated in this study significantly impacts
variation in HR outputs, and to understand the magnitude and direction of those impacts, I
incorporated linear mixed effects modeling as part of the analysis structure using the lmer
function in the lme4 package (Bates et al. 2015) in R v 3.6.1. All models contained HR size
(log-transformed) as the response variable and included a random effect (“query ID”) that
was included to control for the influence of the approach used to query the database, which
may have introduced subject and seasonal non-independence between certain subsets of
permuted datasets. I first constructed a base model including only the random effect, and then
constructed a series of 15 subsequent models adding one additional term (effect or
interaction, see Table 3) to each. Method, sex, sampling duration, and sampling interval were
considered as categorical predictors in the model, while sample size, autocorrelation, and
tracking days (calculated as a percent of tracking days possible given the sampling duration)
were considered as continuous predictors. All continuous predictors were standardized using
the scale() function to counteract problems with model convergence stemming from
explanatory variables with widely differing magnitudes. I included interactions between method and each of sex, sample size, autocorrelation, sampling duration, sampling interval, and tracking days, as well as interactions between sampling interval, sampling duration, and tracking days in and of themselves to look for potential synergies, interdependencies, or counteractive effects between any of the factors that can be manipulated during the research design process. I compared the models against one another to determine whether the addition of each term significantly improved the model fit using an Analysis of Variance (ANOVA), and selected the model with the best fit based on AIC. I visually assessed diagnostic plots from the best fitting model to evaluate assumptions of linearity, homoscedasticity, and normality of residuals, and I calculated correlation coefficients and variance inflation factors to confirm the absence of problematic levels of collinearity. Significant terms in the best fitting model were identifiable through log-likelihood ratio tests.

Scripts used for all tasks applied in the Python, SQL, and R environments in this thesis are available in Appendix B.

Table 3. Terms included in the linear mixed effects modelling approach to analysis, including term details.

<table>
<thead>
<tr>
<th>Fixed Effects</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>method</td>
<td>factor; seven levels</td>
</tr>
<tr>
<td>sex</td>
<td>factor; two levels</td>
</tr>
<tr>
<td>sample size</td>
<td>covariate</td>
</tr>
<tr>
<td>autocorrelation</td>
<td>covariate; measured with Schoener’s Ratio</td>
</tr>
<tr>
<td>sampling duration</td>
<td>factor; four levels</td>
</tr>
<tr>
<td>sampling interval</td>
<td>factor; five levels</td>
</tr>
<tr>
<td>tracking days</td>
<td>covariate; measured as % of possible days given SD</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Random Effects</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>query ID</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Interactions</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>method*sex</td>
<td></td>
</tr>
<tr>
<td>method*sample size</td>
<td></td>
</tr>
<tr>
<td>method*autocorrelation</td>
<td></td>
</tr>
<tr>
<td>method*sampling duration</td>
<td></td>
</tr>
<tr>
<td>method*sampling interval</td>
<td></td>
</tr>
<tr>
<td>method*tracking days</td>
<td></td>
</tr>
<tr>
<td>sampling interval*sampling duration</td>
<td></td>
</tr>
<tr>
<td>sampling interval*tracking days</td>
<td></td>
</tr>
<tr>
<td>sampling duration*tracking days</td>
<td></td>
</tr>
</tbody>
</table>
3.4 Results

3.4.1 Dataset Generation and Home Range Estimation

The dataset generation process outlined in section 3.3.3.2 resulted in a total of 732 datasets, which is less than the projected total of 764 because 32 of these met the elimination criteria for having fewer than ten relocations. The remaining datasets range widely in size and degree of autocorrelation (Table 4), although both distributions are heavily skewed. Most datasets are comparatively small with more than 50% having fewer than 200 relocations (Figure 5), and most datasets are highly positively autocorrelated (as measured through Schoener’s ratio, SR) with 1.6% showing negative autocorrelation (SR > 2), 22.3% showing moderate positive autocorrelation (1 ≤ SR < 2), 40.7% showing high positive autocorrelation (0.1 ≤ SR < 1) and 35.4% showing severe positive autocorrelation (SR < 0.1) (Figure 6).

Table 4. Number of datasets, and range in sample size and autocorrelation values for each subject.

<table>
<thead>
<tr>
<th>Subject ID</th>
<th>DL</th>
<th>ES</th>
<th>TT</th>
<th>WL</th>
</tr>
</thead>
<tbody>
<tr>
<td>no. of datasets</td>
<td>183</td>
<td>186</td>
<td>180</td>
<td>183</td>
</tr>
<tr>
<td>sample size</td>
<td>10 – 79,129</td>
<td>11 – 87,815</td>
<td>10 – 73,179</td>
<td>11 – 70,181</td>
</tr>
<tr>
<td>autocorrelation (Schoener’s ratio)</td>
<td>0.0011 – 2.3252</td>
<td>0.0009 – 2.5671</td>
<td>0.0001 – 2.3723</td>
<td>0.0011 – 2.2857</td>
</tr>
</tbody>
</table>

The condition of range residency was sufficiently met in each original individual dataset through both random walk modelling and variogram analysis approaches. All four original datasets returned MSD and LI values outside the confidence interval of the range produced from the associated randomly generated trajectories, except for the LI value from TT’s dataset which was just narrowly inside the confidence interval (Figure 7). All four datasets produced a variogram with a clearly identifiable asymptotic curve (Figure 7). In combination, these results strongly support that the movement patterns of these individuals during the study period were more constrained than random wandering or migratory behaviour, and consistent with range residency. The majority of datasets (61.8%) were fit with an OUF movement model by the ctmm package, while the remainder were fit with an OU model (31.1%) or an IID model (7.9%).
Figure 5. Distribution of dataset sample size values. Note that the y-axis is presented on a log_{10} scale.

Figure 6. Distribution of autocorrelation values (as measured by Schoener's ratio). The vertical, dark purple line occurs where Schoener's ratio equals 2, indicating the absence of autocorrelation. Values left of the line are increasingly positive autocorrelation, while values right of the line are increasingly negative autocorrelation.
Figure 7. Range residency as measured through variogram analysis, mean squared distance from centre of activity (MSD) and linearity index (LI) for chimps (a) DL, (b) ES, (c) TT, (d) WL. For MSD and LI plots, the solid red line indicates the value of the original dataset and the dashed lines represent the bounds of the 95% confidence interval.
Seven of the eight estimators selected in section 3.3.5.1 were successfully applied to all 732 datasets, resulting in 5124 HR estimates. The exception was the KDE95hlscv estimator, in which the LSCV algorithm routinely failed to converge during the run of the R script, and obstructed the generation of a HR estimate from the majority of datasets. Further attempts to apply this estimator using used the Home Range Tools for ArcGIS (HRT) software (Rodgers et al. 2015) also failed. This problem, common to GPS datasets with large (or even moderate) number of relocations that are closely spaced, was insurmountable and therefore led to the removal of this estimator from further analysis. Of the 5124 HR estimates successfully generated, 102 fell within the top and bottom one percentile for HR area and were eliminated following the implementation of the decision rule outlined in section 3.3.5.6, leaving 5022 HR estimates and 657 datasets for which a HR output from all seven estimators is available.

3.4.2 Linear Mixed Model Output

Model fit was significantly improved (see Table 5) by all of the fixed effects and interactions outlined in section 3.3.6 except for the interaction between sampling duration and tracking days ($\chi^2 = 2.64$, d.f. = 3, $p = 0.4488$). Within the best fitting model, all terms were highly significant ($p < 0.00005$) except for sample size alone ($p = 0.10640$), though sample size was a significant effect through its interaction with method. The corresponding estimates and standard error for each coefficient, provided in Table 5, have not been back-transformed to account for the log-transformation of the response variable within the model. Visual inspection of diagnostic plots from the best fitting model confirmed assumptions of linearity, homoscedasticity, and normality of residuals. All correlation coefficients between continuous predictors were less than 0.35, and all variance inflation factors were less than 8, indicating no major concerns with collinearity.
Table 5. Linear mixed model for HR area. The values reported here are taken directly from the model output and have not been back-transformed. Significance values for each predictor were determined through a Type III Analysis of Variance with Satterthwaite’s method. Estimates and standard error for the levels of each interaction are provided in Appendix A.

<table>
<thead>
<tr>
<th>Model predictors</th>
<th>Estimate</th>
<th>Var.</th>
<th>SE</th>
<th>SD</th>
<th>$\chi^2$</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Fixed Predictors</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(intercept)*</td>
<td>1.878</td>
<td>0.066</td>
<td></td>
<td></td>
<td>2912.01</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>estimator</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GSS200</td>
<td>-1.011</td>
<td>0.039</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GSS500</td>
<td>-0.283</td>
<td>0.037</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KDE95hadhoc</td>
<td>-0.399</td>
<td>0.038</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KDE95href</td>
<td>-0.436</td>
<td>0.038</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCP100</td>
<td>-0.561</td>
<td>0.038</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCP95</td>
<td>-0.715</td>
<td>0.038</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sex</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>male</td>
<td>0.291</td>
<td>0.060</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sample size</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt; 0.001</td>
<td>0.016</td>
<td></td>
<td></td>
<td></td>
<td>344.32</td>
<td>0.10640</td>
</tr>
<tr>
<td>autocorrelation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-0.137</td>
<td>0.018</td>
<td></td>
<td></td>
<td></td>
<td>1469.28</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>sampling duration</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 months</td>
<td>0.308</td>
<td>0.078</td>
<td></td>
<td></td>
<td>111.39</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>6 months</td>
<td>0.554</td>
<td>0.086</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 year</td>
<td>0.544</td>
<td>0.188</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sampling interval</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15 mins</td>
<td>-0.066</td>
<td>0.031</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 hour</td>
<td>-0.097</td>
<td>0.035</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 hours</td>
<td>-0.041</td>
<td>0.056</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24 hours</td>
<td>-0.027</td>
<td>0.060</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tracking days</td>
<td>-0.110</td>
<td>0.018</td>
<td></td>
<td></td>
<td>11.21</td>
<td>0.00001</td>
</tr>
<tr>
<td><strong>Random Effects</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>query ID</td>
<td></td>
<td>0.107</td>
<td>0.327</td>
<td></td>
<td></td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td><strong>Interaction Effects</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>method*sex</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>46.40</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>method*sample size</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>774.46</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>method*autocorrelation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1690.36</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>method*sampling duration</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1020.42</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>method*sampling interval</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>852.27</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>method*tracking days</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1948.56</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>sampling interval*sampling duration</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>72.54</td>
<td>&lt;0.00001</td>
</tr>
<tr>
<td>sampling interval*tracking days</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>78.24</td>
<td>&lt;0.00001</td>
</tr>
</tbody>
</table>

* The categorical reference categories used to compute the estimate and standard error for the intercept are as follows; method: AKDEc, sex: female, sampling duration: 1 month, sampling interval: 1 second.
3.4.3 Addressing Hypotheses and Predictions

3.4.3.1 Hypothesis 1

HR estimators informed most by components of the relationship between relocations in a dataset (e.g. density, autocorrelation) produce more consistent outputs across variations in dataset structure than estimators informed most by the locations of individual, peripheral relocations.

As a first step towards investigating the influence and interplay of HR estimator choice on HR outcomes, I performed a one-way analysis of variance (ANOVA) on the best fitting model, which confirmed the statistical significance of method of estimation on HR output size ($F = 3922.83, p < 0.00001$). To assess and compare variation in the size of expected HR outputs across estimators, I calculated a coefficient of variation for each estimator from the HR output size values predicted by the best fitting model ($n = 657$) (Table 6) and found that, in support of Prediction 1a, MCP and GSS estimators consistently yielded larger variation than the AKDEc and KDE estimators. I next explored the tendencies of certain estimators to generate more or less similar HR outputs through conducting post hoc analyses on the estimated marginal means of each estimator (AKDEc: 10.61 km², GSS200: 5.41, GSS500: 10.39, KDE95hadhoc: 9.58, KDE95href: 10.38, MCP100: 10.79, MCP95: 7.20) (Figure 8), extracted with the package emmeans. Pairwise comparisons derived through contrast analysis demonstrated that 15 of the 21 estimator pairs produced significantly different marginal means ($p < 0.05$, Tukey’s adjustment), while the AKDEc, GSS500, KDE95href, and MCP100 estimators produced means that were insignificantly different from one another (Table 7). While the pairs within the KDE and AKDEc estimator triad all ranked in the bottom third of the range of absolute size differences, they did not produce HR outputs universally more similar in size to one another than to, or between, outputs from MCP and GSS estimator pairs, leading to the rejection of Prediction 1b.

Table 6. A measure of the degree of variation in HR output size generated by each estimator.

<table>
<thead>
<tr>
<th>Estimator</th>
<th>Coefficient of Variation</th>
</tr>
</thead>
<tbody>
<tr>
<td>AKDEc</td>
<td>29.3 %</td>
</tr>
<tr>
<td>KDE95href</td>
<td>31.2 %</td>
</tr>
<tr>
<td>KDE95hadhoc</td>
<td>32.6 %</td>
</tr>
<tr>
<td>MCP95</td>
<td>40.6 %</td>
</tr>
<tr>
<td>GSS500</td>
<td>46.0 %</td>
</tr>
<tr>
<td>MCP100</td>
<td>47.4 %</td>
</tr>
<tr>
<td>GSS200</td>
<td>69.2 %</td>
</tr>
</tbody>
</table>
Table 7. Contrast analysis of the estimated marginal means of HR output size for all estimator pairs.

<table>
<thead>
<tr>
<th>Estimator Pair</th>
<th>Absolute Difference in Size of Estimated Marginal Means</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCP100 - GSS200</td>
<td>5.38</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>AKDEc - GSS200</td>
<td>5.19</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>GSS500 - GSS200</td>
<td>4.98</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>KDE95href - GSS200</td>
<td>4.96</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>KDE95hadhoc - GSS200</td>
<td>4.16</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>MCP100 - MCP95</td>
<td>3.59</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>AKDEc - MCP95</td>
<td>3.41</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>GSS500 - MCP95</td>
<td>3.19</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>KDE95href - MCP95</td>
<td>3.18</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>KDE95hadhoc - MCP95</td>
<td>2.38</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>GSS200 - MCP95</td>
<td>1.79</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>KDE95hadhoc - MCP100</td>
<td>1.22</td>
<td>0.00551</td>
</tr>
<tr>
<td>AKDEc - KDE95hadhoc</td>
<td>1.03</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>KDE95hadhoc - GSS500</td>
<td>0.82</td>
<td>0.01269</td>
</tr>
<tr>
<td>KDE95hadhoc - KDE95href</td>
<td>0.80</td>
<td>&lt;0.0001</td>
</tr>
<tr>
<td>KDE95href - MCP100</td>
<td>0.41</td>
<td>0.82551</td>
</tr>
<tr>
<td>GSS500 - MCP100</td>
<td>0.40</td>
<td>0.18963</td>
</tr>
<tr>
<td>AKDEc - KDE95href</td>
<td>0.23</td>
<td>0.24545</td>
</tr>
<tr>
<td>AKDEc - GSS500</td>
<td>0.21</td>
<td>0.98603</td>
</tr>
<tr>
<td>AKDEc - MCP100</td>
<td>0.19</td>
<td>0.99537</td>
</tr>
<tr>
<td>KDE95href - GSS500</td>
<td>0.01</td>
<td>1.00000</td>
</tr>
</tbody>
</table>

Figure 8. Distribution of HR output sizes predicted by the best fitting model (black dots, jittered), across estimators. Estimated marginal means and associated confidence intervals are presented in white.
I next assessed the degree of influence on HR output size, among estimators, caused by each of sample size, autocorrelation, sampling duration, sampling interval and number of tracking days. First, all of the interactions in the linear mixed effects model between estimator and each of the aforementioned characteristics were significant (p < 0.00001), providing evidence that the estimators examined here differ in their responses to variation across all of these dimensions of data structure. For sample size, both the degree and direction of influence varied among estimators (Figure 9). As sample size increased, the model-predicted HR outputs produced by the GSS estimators demonstrated the strongest positive response, followed by the MCP estimators, the AKDEc estimator, and finally the KDE estimators (Table 10). Curiously, the KDE95href estimator appeared to be both least strongly influenced by sample size and also oppositely affected; as sample size increased the predicted KDE95href outputs decreased in size. It is important to note that while the regression line produced from the KDE95href outputs on Figure 9 does not appear to closely correspond to the overall trend of data points predicted by the model for that estimator, especially at higher sample sizes, a closer examination revealed that the deviant KDE95href response appears driven by the behavior of the estimator at sample sizes having less than 10,000 points, in which the downward linear trend is consistent with the distribution of predicted data points (Figure 10).
Figure 9. Influence of sample size on predicted HR output size, by estimator. Regression lines were fitted for each estimator using the HR output size values predicted by the best fitting model.

Figure 10. Influence of sample size on predicted HR output size, by estimator, fitted for only those datasets with a sample size of less than 10,000 points.
The influence of autocorrelation on HR output size across estimators showed a nearly identical pattern to that of sample size in both relative degree and direction (Figure 11). As the degree of positive autocorrelation increased (measured here by Schoener’s Ratio decreasing, below of the value of 2) the predicted HR outputs of all estimators except KDE95href also increased and did so with largely the same relative strengths as seen with sample size above. Similarly, the predicted outputs of the KDE95href estimator trended in the opposite direction and decreased as positive autocorrelation increased. The only exception to the shared patterns between the influences of sample size and autocorrelation was that the GSS200 estimator was most strongly influenced by sample size but descended in rank below the GSS500 and MCP100 estimators in its response to the influence of autocorrelation (Table 10).

![Figure 11. Influence of autocorrelation on predicted HR output size, by estimator. Note that a Schoener’s Ratio of 2 is equivalent to zero autocorrelation. As Schoener’s Ratio decreases below 2, positive autocorrelation increases, and as Schoener’s Ratio increases above 2, negative autocorrelation increases.](image-url)
Increases in sampling duration were nearly always associated with increases in the estimated marginal mean of HR output size (Figure 12), though significance varied. All estimators showed significant increases in output size in the shift from 1- to 3-month sampling duration, and all except the AKDEc and KDE95href increased significantly again in the 3- to 6-month shift. None of the increases in mean HR output size were significant in the 6-month to 1-year shift (Table 8). To determine overall variation in HR output size across sampling durations for each estimator, I calculated the standard deviation from the estimated marginal means of each estimator over all levels of sampling duration (Table 10) finding that sampling duration (when all other characteristics are controlled for) exerts the strongest influence on the MCP100 estimator and the weakest influence on the KDE95href estimator.

Table 8. Change in the (estimated marginal) mean HR output size associated with increases in sampling duration, across estimators. Only significant values ($p < 0.05$) are presented.

<table>
<thead>
<tr>
<th>Estimator</th>
<th>Change in mean HR output (km$^2$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1m $\rightarrow$ 3m</td>
</tr>
<tr>
<td>AKDEc</td>
<td>+ 2.54</td>
</tr>
<tr>
<td>GSS200</td>
<td>+ 2.14</td>
</tr>
<tr>
<td>GSS500</td>
<td>+ 3.97</td>
</tr>
<tr>
<td>KDE95shadhoc</td>
<td>+ 3.33</td>
</tr>
<tr>
<td>KDE95href</td>
<td>+ 2.80</td>
</tr>
<tr>
<td>MCP100</td>
<td>+ 4.55</td>
</tr>
<tr>
<td>MCP95</td>
<td>+ 3.15</td>
</tr>
</tbody>
</table>

Figure 12. Influence of sampling duration on predicted HR output size. Estimated marginal means were extracted from each level of the interaction between sampling duration and estimator in the best fitting model.
Increases in sampling interval were generally associated with decreases in HR output size but, as with sampling duration, this effect varied across estimators in both degree and direction. Significant negative trends in (estimated marginal) mean HR output size were apparent across all increases in sampling duration for the GSS200 and GSS500 estimators (Figure 13). Similar trends were observed for the MCP100 and MCP95 estimators, but only had significance at larger intervals. The direction of the KDE95hadhoc estimator response was inconsistent, and the KDE95href estimator again demonstrated behaviour opposite to the others. The AKDEc estimator produced mean HR outputs that had no significant differences across all sampling intervals (Table 9). As with sampling duration, I calculated the standard deviation from the estimated marginal means of each estimator to determine overall variation in HR output size (Table 10) and found that sampling interval alone exerts the strongest influence on the GSS500 estimator and weakest influence on the AKDEc.

Table 9. Change in the (estimated marginal) mean HR output size associated with increases in sampling interval, across estimators. Only significant values (p < 0.05) are presented.

<table>
<thead>
<tr>
<th>Estimator</th>
<th>Change in mean HR output (km²)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1s → 15m</td>
</tr>
<tr>
<td>AKDEc</td>
<td>-</td>
</tr>
<tr>
<td>GSS200</td>
<td>- 4.61</td>
</tr>
<tr>
<td>GSS500</td>
<td>- 3.77</td>
</tr>
<tr>
<td>KDE95hadhoc</td>
<td>-</td>
</tr>
<tr>
<td>KDE95href</td>
<td>+ 2.90</td>
</tr>
<tr>
<td>MCP100</td>
<td>-</td>
</tr>
<tr>
<td>MCP95</td>
<td>-</td>
</tr>
</tbody>
</table>

Figure 13. Influence of sampling interval on predicted HR output size. Estimated marginal means were extracted from each level of the interaction between sampling duration and estimator in the best fitting model.
Because there was a significant interaction between sampling duration and sampling interval in the model, I examined how each estimator responded to simultaneous variation in both factors. I extracted the estimated marginal means derived from each combination of levels, for each estimator (n=20) (Figure 14) and calculated the standard deviation, finding that, overall, variation across both sampling duration and sampling interval has the strongest impact on the GSS500 estimator and the weakest on the AKDEc estimator (Table 10).

Next, I visually examined interaction-style plots of these estimated marginal means to identify trends across both data dimensions (Figure 15, Figure 16). The behaviour of the AKDEc and KDE95hadhoc estimators appears highly conserved across changes in sampling interval at each sampling duration, and across changes in sampling duration at each sampling interval, though larger sampling durations shift the mean HR outputs upwards in both of these cases. For the GSS and MCP estimators, decreasing sampling interval and increasing sampling duration both magnify the slope of the response curve across changes in the corresponding characteristic. There is also an upwards shift in mean HR output size at both larger sampling durations and smaller sampling intervals. As elsewhere, the KDE95href estimator is unique in that both increasing sampling interval across changes in sampling duration, and increasing sampling duration across changes in sampling interval, elicit increases in mean HR output size until the transition to the largest sampling interval or sampling duration, at which time the direction of the response is reversed. Both larger sampling intervals and larger sampling durations appear to exacerbate this behaviour.
Figure 15. Interaction-style plots of HR output (estimated marginal) means, faceted by sampling interval.
Figure 16. Interaction-style plots of HR output (estimated marginal) means, faceted by sampling duration.
In terms of tracking days (measured here as a percentage of the number of tracking days possible for the dataset given the sampling duration), the degree and direction of influence once again both differed widely among estimators (Figure 17). As tracking days increased, the model-predicted HR outputs produced by the GSS estimators also increased while those produced by all other estimators decreased. While the KDE and AKDEc estimators were collectively the most strongly influenced and the GSS and MCP estimators were collectively the least strongly influenced, there were no patterns indicating coherent behaviour by estimators within the same immediate family (Table 10).

Figure 17. Influence of tracking days on predicted HR output size, by estimator. Regression lines were fitted for each estimator using the HR output size values predicted by the best fitting model.
Table 10. A summary of the response of each estimator to changes in each data characteristic. Units of measurement are as follows; sample size: $\Delta HR$ size (km$^2$) per increase in 1000 relocations; autocorrelation: $\Delta HR$ size (km$^2$) per decrease of 1 in Schoener’s Ratio; tracking days: $\Delta HR$ size (km$^2$) per increase of 25% of tracking days possible; sampling duration/sampling interval/interaction: standard deviation of estimated marginal means. Dark green indicates values that are least influenced, followed by values in light green. Dark orange indicates values that are most strongly influenced, followed by values in light orange.

<table>
<thead>
<tr>
<th>Estimator</th>
<th>sample size</th>
<th>autocorrelation</th>
<th>tracking days</th>
<th>sampling duration</th>
<th>sampling interval</th>
<th>interaction (sampling duration * sampling interval)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AKDEc</td>
<td>+ 0.06</td>
<td>- 2.27</td>
<td>- 1.94</td>
<td>2.57</td>
<td>0.64</td>
<td>2.38</td>
</tr>
<tr>
<td>GSS200</td>
<td>+ 0.13</td>
<td>- 4.06</td>
<td>+ 0.54</td>
<td>4.05</td>
<td>2.76</td>
<td>5.32</td>
</tr>
<tr>
<td>GSS500</td>
<td>+ 0.13</td>
<td>- 4.88</td>
<td>+ 0.46</td>
<td>6.23</td>
<td>3.26</td>
<td>6.77</td>
</tr>
<tr>
<td>KDE95hadhoc</td>
<td>+ 0.05</td>
<td>- 1.37</td>
<td>- 1.53</td>
<td>3.64</td>
<td>0.69</td>
<td>3.27</td>
</tr>
<tr>
<td>KDE95href</td>
<td>- 0.04</td>
<td>+ 0.60</td>
<td>- 2.54</td>
<td>2.55</td>
<td>1.86</td>
<td>2.91</td>
</tr>
<tr>
<td>MCP100</td>
<td>+ 0.12</td>
<td>- 4.30</td>
<td>- 0.93</td>
<td>6.50</td>
<td>2.26</td>
<td>6.31</td>
</tr>
<tr>
<td>MCP95</td>
<td>+ 0.09</td>
<td>- 3.25</td>
<td>- 0.38</td>
<td>3.52</td>
<td>1.34</td>
<td>3.43</td>
</tr>
</tbody>
</table>

Support for Prediction 1c was mixed across data characteristics. For sample size and autocorrelation, the AKDEc estimator produced HR outputs that were more strongly influenced than those of the KDE estimators, contrary to what was predicted. However, the MCP and GSS estimators both behaved as expected and were more strongly influenced by each data characteristic than the density-informed estimators. The curious behaviour of the KDE95href estimator, opposite in direction to the trends followed by all other estimators for these characteristics, introduces a degree of distrust in this estimator, regardless of its seeming resistance to variation in the characteristics of sample size and autocorrelation. The results in regards to tracking days were completely opposite to the behaviour expected by Prediction 1c. In these cases, the MCP and GSS estimators demonstrated the least influence, followed by the AKDEc and KDE estimators collectively. In terms of sampling duration, Prediction 1c is mostly supported, with the caveat that the KDE95href shows slightly less influence than the AKDEc estimator. Finally, Prediction 1c is fully supported for both sampling interval and for the interaction between sampling duration and sampling interval.
3.4.3.2 Hypothesis 2

Because sampling interval is a large contributor to the information structure of a movement dataset, it should exert a stronger influence than other data characteristics on HR output size, and a stronger influence within HR estimators that consider fewer aspects of information structure.

To assess whether sampling interval explains the most variance in HR output size of all the data characteristics examined here, I first derived the conditional coefficient of determination from the full model ($R^2 = 0.953$), and then removed each characteristic (and the interactions in which it was involved) from the model one at a time with replacement and derived $R^2$ again in each case (model minus sample size: $R^2 = 0.951$, model minus autocorrelation: $R^2 = 0.952$, model minus sampling duration: $R^2 = 0.915$, model minus sampling interval: $R^2 = 0.931$, model minus tracking days: $R^2 = 0.929$). I subtracted each of these values from the associated value from the full model to determine the amount of unique variance in HR output size that each characteristic explains above and beyond the other variables in the model, and identified that sampling duration was clearly the superior, leading to the rejection of Prediction 2a.

In support of Prediction 2b, the largest decreases in HR output size in relation to increasing sampling interval were indeed associated with the MCP and GSS estimators followed by smaller overall decreases in the outputs of the KDE estimators, and finally no significant decreases in the AKDEc estimator outputs (Table 9, Figure 13). These results suggest that estimators that consider fewer aspects of a dataset’s information structure are most vulnerable to variation in that information structure. Finally, to compare the degree of deviation among estimator outputs at different sampling intervals, I calculated the standard deviation of the estimated marginal means from all estimators for each sampling interval ($n=7$) and found that, contrary to Prediction 2c, the largest variation in mean size of HR outputs across estimators occurred at the largest sampling interval (1s: SD = 2.19, 15m: SD = 2.03, 1h: SD = 2.53, 4h: SD = 3.26, 24h: SD = 3.42).
3.4.3.3 Hypothesis 3

Because of their shared mechanisms of estimation, estimators from the same family should generate a stronger relationship between their respective HR outputs than those from different families.

I calculated a Pearson correlation coefficient for each pair of estimators to measure the strength of the relationship of HR outputs derived from the same datasets (n=657). All estimator pairs demonstrated positive correlations and all correlations were significant, but the strength of these correlations ranged widely. The strongest correlations were associated with the MCP100 – MCP95 pair and the GSS500 – MCP95 pair, and the weakest correlations were seen in the KDE95href – GSS200 pair and the KDE95href – GSS500 pair (Figure 18). The pattern of correlation outcomes demonstrated mixed support for Prediction 3a, which suggested that the relationship between HR outputs would be strongest between estimator pairs from the same estimator family (Table 11). The MCP estimator pair and GSS estimator pair both ranked in the top four strongest correlations, but the pairs within the KDE/AKDEc triad fell squarely in the middle of the range of correlation values. If we broaden the definition of “family” to categorize those estimators that do and do not factor relocation density into their calculation, there is greater evidence of strong within-family relationships between estimators.

![Figure 18. Pearson correlation matrix depicting the relationship strength between estimator pairs of HR outputs calculated from the same datasets. On the top are the absolute values of the correlation coefficient and the significance of the correlation represented by stars (*: p<0.05, **: p<0.01, ***: p<0.001) and also the size of text. On the bottom are bivariate scatterplots with a fitted line in red.](image-url)
Figure 19. The two strongest correlations exist within a) the MCP100 – MCP95 estimator pair and b) the GSS500 – MCP95 estimator pair. The two weakest correlations exist within c) the KDE95href – GSS500 pair and d) the KDE95href – GSS200 pair. Best fit lines are displayed in solid red and the line $y = x$ is displayed in dashed black.
Table 11. Correlation of HR outputs between estimator pairs. Light blue highlighting indicates pairs within the same broad estimator family (density/no-density) and dark blue highlighting indicates pairs in the same immediate estimator family.

<table>
<thead>
<tr>
<th>Estimator Pair</th>
<th>Correlation Coefficient</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCP100 - MCP95</td>
<td>0.97</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>GSS500 - MCP95</td>
<td>0.95</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>GSS500 - MCP100</td>
<td>0.95</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>GSS200 - GSS500</td>
<td>0.95</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>GSS200 - MCP95</td>
<td>0.88</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>GSS200 - MCP100</td>
<td>0.85</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>AKDEc - KDE95hadhoc</td>
<td>0.83</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>KDE95hadhoc - MCP95</td>
<td>0.82</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>KDE95hadhoc - MCP100</td>
<td>0.82</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>AKDEc - MCP95</td>
<td>0.79</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>KDE95hadhoc - KDE95href</td>
<td>0.78</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>AKDEc - KDE95href</td>
<td>0.77</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>KDE95hadhoc - GSS500</td>
<td>0.73</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>AKDEc - MCP100</td>
<td>0.73</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>AKDEc - GSS500</td>
<td>0.62</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>KDE95hadhoc - GSS200</td>
<td>0.56</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>KDE95href - MCP100</td>
<td>0.54</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>KDE95href - MCP95</td>
<td>0.50</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>AKDEc - GSS200</td>
<td>0.48</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>KDE95href - GSS500</td>
<td>0.35</td>
<td>&lt; 0.00001</td>
</tr>
<tr>
<td>KDE95href - GSS200</td>
<td>0.10</td>
<td>0.01407</td>
</tr>
</tbody>
</table>
3.5 Discussion and Conclusions

In this thesis I investigated the influence of variation in dataset characteristics and estimator choice on HR size and comparability in wild chimpanzees, reproducing and examining a wide range of data collection and HR estimation methodologies including those most commonly applied in the past and those most likely to appear in near-future studies. The results of this research provide insights into the kinds of datasets that are valuable, the impact of variation in different dimensions of data structure, the behaviour of HR estimators under varying data conditions, and the feasibility of meaningfully translating between the results of HR analyses. These lead naturally to several important recommendations and guidelines for those wishing to maximize the reliability and comparative utility of future projects that include HR estimation.

3.5.1 Datasets and the Influence of Data Characteristics

3.5.1.1 Sample Size and Autocorrelation

The process through which datasets were generated in this study was successful in reproducing the range of characteristics evident in the body of literature on this subject, but many of these datasets turned out to be unsatisfactory or unreliable despite being accurate reflections of those published elsewhere. Even following the application of the decision rule to eliminate extremely small datasets and following the permutation process intended to generate various autocorrelation structures, some of the datasets still appeared insufficient for appropriate HR estimation with certain estimators in two ways. First, datasets with less than 200 points were mostly insufficient for the AKDEc estimator, which relies on the information structure of the relocations to interpret the underlying movement pattern and select an appropriate movement model with which to extrapolate HR. In these cases, this estimator erroneously selected the IID (independent and identically distributed) model, which we know is incorrect both from a basic understanding of animal movement and from the fact that the OU or OUF models were selected for 91.1% of datasets. This led the AKDEc to perform poorly on these datasets, producing unreasonable HR estimates, and indicates that despite its superior quality in many other regards (see discussion below), the AKDEc
estimator may not be an appropriate choice for HR estimation when the datasets available are especially sparse (though the alternatives may also be poor choices for different reasons).

Second, while the permutations of sampling interval were implemented partly to produce variation in the degree of autocorrelation across datasets, the vast majority still contained some (and often a high) degree of positive autocorrelation. For instance, at the 4-hour sampling interval, which has been elsewhere used in chimpanzee literature in an attempt to eliminate autocorrelation entirely (Amsler 2009; Bertolani 2010), only 1.4% of the datasets fully reached or exceeded the point of zero positive autocorrelation (as measured by a Schoener’s Ratio of 2). Only 12.5% of datasets came within even the lowest quarter of the range of positive autocorrelation at this sampling interval. Moreover, even those datasets with the very largest sampling interval (24-hour) were overwhelmingly positively autocorrelated. The results of this study strongly support those conclusions referred to in the discussion on autocorrelation above, that attempting to eliminate autocorrelation in modern datasets is a wasteful practice (and likely unfeasible in many scenarios) because it requires the abandonment of such vast swaths of data as to render the final product nearly void of the important and biologically meaningful information. Instead, this research suggests that we must accept autocorrelation as an inherent and unnegotiable component of modern tracking datasets, liable only to increase in the future as GPS data collection technologies develop. The sound path forward will be to search for, use, and develop those methods that can at least account for and at best, make use of, the autocorrelation structure of our datasets, while simultaneously discarding those for which autocorrelated data is explicitly contraindicated.

3.5.1.2 Sampling Duration, Sampling Interval, and Interactions

Overall, this study demonstrated that variation in sampling duration across datasets explained associated variation in HR output size more than any other data characteristic when examined in isolation. At a closer glance, however, this result appears largely driven by the behaviour of a certain few estimators that are especially vulnerable to the effects of sampling duration. While the increases in mean HR output size associated with increases in sampling duration beyond 6 months were not statistically significant for any estimator, visual assessment of Figures 12, 15 and 16 imply that the estimators that are highly informed by the
most distal relocations in a dataset (GSS200, GSS500, MCP100) are more strongly influenced by comparatively longer sampling durations. Extreme distal relocations have a high propensity for reflecting an individual’s “exploratory forays or occasional sallies” into extra-HR regions, and it may be that the above estimators are especially sensitive to longer sampling durations precisely because this added tracking time increases the chances of capturing these forays, reflecting them as inflated HR estimates. In contrast, the estimators that are more informed by the relative distribution of the majority of relocations (AKDEc and KDE estimators) appear more robustly resistant to changes in sampling duration across the board, and produce a near-plateau effect after 6 months. Because of the lack of significance in the increases in HR output size associated with the transition in sampling duration from 6-months to 1-year, a 6-month sampling duration may be sufficient to capture information about this population’s movement patterns necessary to generate representative HR estimates. This duration makes intuitive sense, given that the eastern chimpanzee’s equatorial rainforest habitat operates on a twice-annual (approx. 6 month) wet-dry seasonal cycle rather than a 12-month, four-season cycle, leading to greater temporal consistency in both environmental drivers of movement and movement patterns themselves. Variation in sampling duration would therefore appear to be a larger problem for some estimators (e.g. GSS500, MCP100) than others (e.g. KDE95href, AKDEc), and greater care should be taken to standardize the duration of a dataset when it is intended for use with the non-density-informed estimators than with the density-informed estimators.

While we know that sampling interval is a large contributor to the information content of a dataset because of its direct influence on both the number of data points collected and the relationship between them (degree of autocorrelation), the variation in HR output size explained by this characteristic alone was less than that explained by sampling duration. Despite its smaller effect size, however, the influence of sampling interval was much more unpredictable across estimators, with wider variation in magnitude, direction, and overall trend of responses. It could be that this deviation in response reflects estimator capacity to properly attend to variation in the quality and quantity of information across datasets. The largest influences of sampling interval appear at the extreme ends of the spectrum, in the shifts from 15-minutes to 1-second or from 4-hours to 24-hours. The latter shift most greatly exacerbates the difference in estimator response, causing the most variation
in mean HR outputs across estimators. While contrary to the prediction made in this study that a very small sampling interval would cause the greatest variation in HR outputs, this outcome reveals that too little information (as caused by large sampling intervals) causes more variance across estimators than does too much.

Overall, increasing sampling duration while sampling interval is held constant resulted in larger mean HR output size, and this phenomenon was evident to a slightly lesser degree when decreasing sampling interval while holding sampling duration constant. However, beyond examinations of each characteristic in isolation, the results of this study demonstrated that there is an important interaction between sampling duration and sampling interval on HR output size. Variation across one characteristic can heighten or buffer the influence of the other characteristic, and both the magnitude of this effect and the overall variation caused by changes across both levels differs among estimators. The variation in HR outputs from the GSS200, GSS500, and MCP100 estimators appeared particularly intensified by variation across both dimensions. In general, smaller sampling intervals exacerbated the influence of variation in sampling duration, and larger sampling durations exacerbated the influence of variation in sampling interval. Research involving data collect methodologies with one or both of large sampling duration and small sampling interval should therefore apply extra consideration towards estimator choice, and may ultimately produce results that are less comparable to HR estimates generated from data collection strategies with other parameters.

3.5.1.3 Tracking Days

The number of tracking days in a dataset, measured as a percentage of the number possible given the sampling duration of the dataset, played a curious role in HR output size. For most of the other data characteristics, the same response trend was followed by all estimators, or, if one responded uniquely, it was consistently the KDE95href estimator. Here, however, we see both of the GSS estimators performing oppositely to the rest. Similarly curious is the influence of tracking days, which appeared strongest on the estimators traditionally most resistant to variation in data characteristics, and weakest on those that are otherwise most susceptible. Explanations for this divergent behaviour may be somehow
rooted in the way that tracking days were measured here, but these results do not immediately lend themselves to meaningful interpretation or recommendations about how tracking days should best be structured to minimize the undue influence of this characteristic on HR estimates.

3.5.2 Evaluating Estimators

Right off the bat, the dramatic “failure to converge” of the KDE95hlscv estimator on nearly all datasets to which it was applied supports the conclusions made elsewhere in the HR literature that this estimator is unsuitable for use with most modern tracking datasets. Moreover, what limited utility remains will continue to fade rapidly as the field of HR analysis enters an era of big data and GPS technology, producing ever larger and more finely resolved datasets that will be increasingly difficult for this estimator to manage. The failure of this estimator in the present study precludes conclusions about its vulnerability or resilience to variation in data structure, and about its comparability to the other estimators examined here. Because of this outcome, I find there is nothing to recommend use of the kernel density estimate using a bandwidth selected by Least-Squares Cross-Validation for modern HR analysis.

Of the estimators that were successfully assessed in this study, the GSS and MCP variants collectively performed the most poorly on nearly all metrics, which is to say that they were most heavily influenced by non-spatial characteristics of data structure. These estimators are therefore more liable to reflect artifacts of a data collection strategy in their HR outputs, obfuscating or eclipsing the true underlying movement pattern. When applied to the collection of datasets generated by this study, the GSS200 estimator produced HR outputs with the greatest overall variation in size, and the associated mean HR size was least comparable to the means of all other estimators. Additionally, the GSS200 estimator was the most strongly influenced by both sample size and sampling interval, indicating that this approach is poorly suited for comparisons between HR outputs from datasets that vary even moderately in these respects. While more comparable in mean HR output size to the other estimators, the GSS500 estimator also produced a high amount of overall variation in the size of HR outputs, and performed either worst or second worst under the influence of all data
characteristics (except for the strange effect of tracking days described above). Thus, there is little to recommend a grid-square-sum approach (using any cell size) for the assessment of HR, especially under research conditions where the comparison of HR outputs from dissimilar datasets is anticipated.

The MCP100 estimator ranked second-worst only to the GSS200 estimator in the amount of overall variation in the size of HR outputs it produced. The mean HR output size from this estimator was within the range of those of the AKDEc and KDE estimators, both when the overall mean of outputs was examined and when the range of estimated marginal means from each combination of the levels of sampling duration and sampling interval was examined. However, based on a visual assessment of the distribution of data points in Figure 8, there is not in fact a higher density of MCP100 estimates near the calculated mean and it appears instead that the MCP100 estimator only generated this outcome by “regression to the mean”, indicating that the comparability of this estimator may be weaker than our results suggest. The MCP100 estimator was the most highly influenced by sampling duration, and also highly influenced by autocorrelation and the interaction between sampling interval and sampling duration. Unlike the comparable mean HR output size produced by the MCP100 estimator, the MCP95 estimator regularly produced significantly smaller HR outputs than all other estimators except GSS200. However, a positive feature of the MCP95 estimator is that it proved less vulnerable than those estimators already discussed to the effects of all data characteristics examined (though it was not least vulnerable amongst all estimators examined in this study to any of the characteristics except tracking days). Additionally, when exposed to the wide variation in data structure across the datasets examined here, the overall variation in the size of HR outputs produced by the MCP95 approach was less than that of any estimator discussed thus far. It would seem therefore that, of the non-density-based estimators, the MCP95 approach is most liable to reflect underlying movement pattern above and beyond artifacts of data structure, though it may be subject to significant underestimations in size, which could lead to numerous problems for applications such as conservation or management actions based on allocating sufficient protected space to populations under threat. Ultimately, most modern HR analyses will have access to density-based estimators and, as we will see, the MCP95 estimator is generally inferior.
The KDE95href estimator performed strongly in producing the second-lowest amount of variation in HR output size, with a mean HR output that compared well to three other estimators (AKDEc, GSS500, and MCP100). Beyond these advantages, however, the behavior of this estimator was unexpected, unintuitive, and highly divergent from the trends followed by the rest of the group. For instance, while KDE95href is technically least susceptible to the influence of variation in both sample size and autocorrelation, the influence that did exist was in the opposite direction from what is both expected and demonstrated by the other estimators (a positive association between increasing sample size/autocorrelation and HR output size). Why should larger datasets lead to smaller HR outputs with this estimator? It is likely a facet of the way in which the “href” bandwidth is calculated; the square root of the mean variance of x and y coordinates is divided by the sixth root of the number of points. If we consider a hypothetical situation in which a given dataset fully captures the extent of an individual’s ranging, but we increase the duration of sampling and collect more relocations from the individual which follow the same movement patterns (no exploratory forays), the numerator (mean variance of x and y coordinates) will not increase but the denominator (sixth root of the number of points) will, leading to a smaller calculated value for the bandwidth of the larger dataset. A smaller bandwidth applied to the same distribution will result in a smaller overall HR size (Hemson et al. 2005). This is an important problem for the KDE95href estimator and has the capacity to distort results in unanticipated ways, particularly for GPS datasets. Similarly, autocorrelation and sampling interval both had directionally opposite effects on this estimator in comparison to the rest of the estimators, indicating that KDE95href may be uniquely poorly suited to interpreting underlying movement patterns from the overlaying components of a dataset’s information structure. Even compared to it’s closest estimator relatives (KDE95hadhoc and AKDEc), we see a nearly mirror-image response to the characteristics of sample size, autocorrelation, and sampling interval. Curiously, KDE95href shares the same trends as other estimators (and responds nearly identically to AKDEc) in regard to the influence of sampling duration, producing no significant difference in HR output size when sampling duration is increased beyond three months. Despite its apparent immunity to the influence of sampling duration, the strange and incongruous behavior of the KDE95href estimator across a number of
dimensions of data structure should introduce reservations about its capacity to produce reliable and comparable HR estimates.

While it produced slightly more overall variation in HR output size than the AKDEc and KDE95href estimators, and produced a mean HR output not quite within the range of the very similar group (AKDEc, KDE95href, MCP100, GSS500), the KDE95hadhoc estimator followed the same trends as all other estimators except KDE95href in regard to the influence of each data characteristic and also demonstrated either a low or moderately low degree of influence by each. If we eliminate KDE95href from consideration, the KDE95hadhoc estimator ranks most resistant to the influence of sample size and autocorrelation, though would still be outperformed by AKDEc in terms of resistance to variation in sampling duration, sampling interval, and the interaction between the two. Indeed, the KDE95hadhoc estimator doesn’t produce the same plateau in estimated mean HR output size in the transition from a 6-month to a 1-year sampling duration as the ADKEc and KDE95href estimators, suggesting it may be more sensitive to the exploratory forays that sampling duration may capture. However, the response of this estimator across the various levels of sampling interval and sampling duration appear remarkably consistent and predictable, especially across changes in sampling interval. These results suggest that the KDE95hadhoc estimator would outperform any of the non-density estimators, and in many cases the KDE95href estimator, in the face of variation in data collection strategies. This advantage likely results from the introduction of human subjectivity in the bandwidth selection process, in which the bandwidth is adjusted such that the resulting HR output does not contain any lacunae and has a continuous outermost isopleth, buffering some of the more extreme effects of poorly suited bandwidth calculations. Whether or not this approach is valid remains untestable, as “…no objective method exists at present to tie bandwidth to biology…” (Powell 2000, p.87). We cannot ignore, however, that use of the KDE95hadhoc estimator introduces a level of human bias towards what a HR should look like and tends to produce both slightly smaller outputs than most estimators and lead to slightly more overall variation across datasets than estimators such as the AKDEc.

Of all the estimators in this study, the AKDEc estimator produced the least variable HR outputs from the range of varying data conditions across the collection of permuted
datasets. Its mean HR output was also the most comparable to that of other estimators: AKDEc was involved in three of the top four most comparable estimator pairs (but see below for a discussion on comparability). This estimator was also least influenced by variation in sample interval as it was the only one to produce no significant change in HR size across any of the changes in sampling interval levels, and also least influenced by the interaction between sampling duration and sampling interval as evidenced through the remarkable consistency across level changes shown in the interaction plots of Figures 15 and 16. While the AKDEc did not produce significant changes in mean HR size associated with increases of sampling duration beyond three months, and did result in one of the lowest amounts of variation in response to this characteristic, it was still outperformed for this metric by the KDE95href by a small margin (AKDEc: SD = 2.57, KDE95href: SD = 2.55). Surprisingly, it was well outperformed by both KDE estimators in terms of their resistance to the influence of variation in both sample size and autocorrelation. This is curious given that the AKDEc estimator was specifically designed to directly account for autocorrelation. I suspect that the cause of this behavior was driven by the 7.9% of datasets that were inappropriately assigned an IID movement model by the AKDEc estimator due to small size, a hypothesis that future extensions of this research may be able to confirm.

What can be made of the information derived from this study on the behavior of estimators across conditions of varying data characteristics? While we cannot accept Prediction 1c as a fully accurate characterization of the relative influence of variation across all data characteristics on the estimators examined here, it has been demonstrated that the density-based estimators as a whole do outperform the non-density estimators in almost all characteristics and that, of the density estimators, the AKDEc estimator appears the most reliably resilient to variation across the greatest number of characteristics. The results also suggest that the AKDEc estimator was the best able to handle the changes in information structure and content introduced by variation in sampling interval, followed by the KDE estimators and then the GSS and MCP estimators as predicted. While the results here fail to provide an undisputed champion that outperforms all other estimators under all conditions, the AKDEc estimator is clearly a superior performer for most situations and is associated with the fewest theoretical and empirically demonstrated concerns discussed in Chapter 2.
3.5.3 Feasibility of Meaningful Comparisons/Translations

A central aim of this thesis was to investigate the feasibility of comparing between the HR outputs produced by different estimators and under different data conditions. A component of evaluating this feasibility is assessing the degree of consistency in the similarity of HR outputs from different estimators when applied to the same datasets. To this end, a high correlation coefficient between the outputs of a given pair would indicate that a reliable translation between estimators might be made. Of the estimator pairs examined here, several produced especially high correlation coefficients (> 0.85). These exclusively occurred among the non-density-based estimators, though not exclusively between pairs from the same estimator family. Correlation coefficients involving density-based estimators were all lower, ranging from 0.1 – 0.83. However, any mathematical conversion of HR size even among highly correlated estimator pairs cannot be performed on a 1-to-1 basis and instead must also account for the slope of a given regression. In nearly all estimator pairs examined here, the regression slope sits far from 1. For instance, the MCP100-MCP95 estimator pair has the highest overall correlation coefficient (0.97), but in looking closer at a plot of this relationship (Figure 18a), we can see that the slope of the regression line is closer to 0.6, indicating that the difference between the expected outputs of each estimator will increase as HR size increases. Based on their estimated marginal means, these estimators produce outputs that differ on average by 3.59 km². In our case this would amount to an over- or underestimation of more than 30%, given that the most likely HR size of this chimpanzee population (inferred from the distribution of all outputs produced in this study) lies around 10 km². Direct comparisons between HR results from the MCP100 and MCP95 would therefore become increasingly less meaningful as HR estimate size increases.

Before making any translations across estimators it is also critical to note the relative positions of an estimator pair’s regression slope and the line of $y = x$, particularly if these lines cross, as this will dictate the direction of the mathematical translation at various HR sizes. For instance, the regression slope for the GSS500 – KDE95href estimator pair is weak and crosses the line of $y = x$ near 10km². At HR sizes smaller than this, the GSS500 estimator produces increasingly larger outputs than the KDE95href estimator, but this behavior is reversed at the larger end of the spectrum. These are important nuances that complicate the task of comparing HR outputs across studies involving different populations,
underlying movement patterns, and approaches to estimating HR. There may be some support for applying a mathematical translation in order to compare estimates derived from different non-density-based estimators, but these possibilities disappear when density-based estimators are involved. In these cases, a more suitable approach would be access the original data and apply the same estimator (preferably resistant to data variation) to both datasets.

3.5.4 Recommended Guidelines

The results of this study lend themselves to several important recommendations for the estimation of HR in chimpanzees, both as stand-alone products and for the purposes of comparability with future or past results. These include guidelines for structuring the collection of tracking data, for selecting an appropriate HR estimator based on data structure, and for maximizing comparability. In terms of structuring data collection, it is insufficient to simply aim for a minimum sample size. The quest for optimal sample sizes has, in the past, led to widely different conclusions likely in part because sample size is not an accurate characterization of the information inherent in a dataset and it is this information, in addition to the way it is structured, that more greatly influences HR outcomes. This idea is supported in this study by the relatively meager effect of sample size alone on HR size variability. A more important objective in the structuring of data collection is to maximize the quality and quantity of information in a dataset such that movement patterns are most accurately characterized, and the means by which to accomplish this will relate to the biology and ecology of the species in question. For instance, the results here would suggest that a sufficient sampling duration for eastern chimpanzees is six months, and that extremely large sampling intervals capture too little movement information for many estimators. Given this, it would therefore be better to collect data from eastern chimpanzees with a smaller sampling interval over the course of six months than with a large sampling interval over the course of two years. Tracking days, sample size, and autocorrelation appear less important measures of overall dataset quality, and researchers should focus on making careful selections informed by species biology with regard to sampling duration and sampling interval of their data collection strategies, especially as they relate to the strengths and weaknesses of the estimator(s) to be applied.
In terms of the most appropriate estimator for a given dataset, the AKDEc estimator will, in most cases, outperform the others and result in HR outputs that are least likely to contain the undue influence of dataset variation beyond that of the underlying movement pattern, leading to the most directly comparable results with other AKDEc-produced HRs over time and space including those with different data collection strategies. If the AKDEc estimator is not a possibility because of limitations of computing power or insufficient sample size, a substitute should be chosen with respect to the characteristics of the intended dataset(s) and relative estimator sensitivity to these. For instance, if two datasets with highly different sampling intervals are to be compared, the GSS200 estimator would be least desirable because it is the most highly influenced by variation this characteristic. Similarly, the MCP100 estimator would likely perform most poorly on two datasets with highly differing sampling durations. It will also be important to note and attempt to account for associated estimator-specific shortcomings: in this case GSS200 is likely to seriously underestimate HR size in comparison to the other estimators, and MCP100 likely performs poorly with datasets containing significant exploratory forays. Despite widespread popularity, the results of this thesis do not support the use of the KDE95href due to its unreliable behavior, and would recommend the KDE95hadhoc variant instead if a kernel-density estimator other AKDEc must be used.

One of the most important guidelines supported by this study, which has been advocated for in the past and should be strongly adhered to in the future, is the critical importance of reporting details pertaining to estimator application and data collection. A study that reports that HR was estimated using a “kernel density estimate” approach but fails to mention whether the bandwidth selection was performed using the “href”, “hadhoc” “hlscv” or another calculation precludes any form of meaningful result interpretation with regard to the potential effect of the estimator variant on the HR outcome. Similarly, a study that reports using an MCP estimator without detailing what (if any) percentage of points were peeled, or reports using a GSS estimator without defining the grid cell size commit the same offence. This appeal for detail extends also to the datasets examined, for which the size, degree of autocorrelation, sampling duration, and sampling interval should be reported at minimum.
Finally, the comparison of HR estimates produced by different studies using different data collection strategies, datasets, and estimators simply cannot be carried out without giving consideration to the ways in which the unique dataset characteristics and estimators applied to the datasets in question will influence the relationship. Comparisons of HR outputs derived with the same estimator and from datasets under the same data collection strategies are more liable to be representative, but should take into account the likely influence of varying data conditions specific to that estimator. Comparisons across estimators are likely to be less fungible, but potentially still meaningful if made within the group of non-density-based estimators. The KDE95href estimator in particular appears highly incomparable to all other estimators examined here, including its fellow density-based estimators AKDEc and KDE95hadhoc. Any attempts towards comparison of HR outputs should be accompanied by a confidence interval that reflects the likelihood of accuracy outlined here, and must involve some form of mathematical conversion to account for a pair’s unique regression slope.

3.5.5 Assumptions, Limitations, and Future Directions

This thesis makes some inherent assumptions and is beset by certain limitations that may circumscribe the breadth of its conclusions. First, it assumes that the approach used to collect the chimpanzee data analyzed here was rigorous, appropriate, and did not introduce undue bias to the original datasets. This is reasonable, given the unavoidable limitations in tracking chimpanzees and the fact that no current alternatives could feasibly eliminate all forms of bias, though this may change as tracking technology improves. Second, it assumes that the four chimpanzees selected are representative of the diversity of ranging strategies present in this community. This assumption is reasonable given that the individuals span sex, dominance status, and reproductive status, which are three large contributors to behavioural variation in chimpanzees (Goodall 1986). However, care must still be taken when extrapolating these results to other chimpanzee individuals and future work would benefit from applying these same methods to more chimpanzee individuals, especially those that are poorly sampled in tracking datasets. Third, it assumes that the data permutations performed on the original datasets are sufficient to represent dataset variations stemming from different collection methods, but future research should broaden these permutations beyond those
levels of sampling duration and sampling interval specified here, to understand what differences, if any, are present at more extreme values such as multi-year sampling durations or multi-day sampling intervals.

The limitations that impose constraints on this study are, first, that the HR estimators explored here are restricted to those judged by the author to be the most commonly employed within the chimpanzee literature. This does not include less popular range estimators such as the Local Convex Hull, nor any occurrence estimators such as Brownian Bridge, continuous-time correlated random walk, or time-series Kriging. Therefore, additional work in this avenue will be necessary to make any conclusions or recommendations about the influence of those estimators on HR comparability. Second, this study was site-specific, assessing only the chimpanzee movement of a particular community from a particular forest, and we cannot know that the conclusions drawn here will hold true for other field sites or in other habitats and biomes. Parallel work at other sites would therefore be valuable confounding or corroborating evidence. Finally, this study only examined one metric of HR comparability: size. However, shape and location (measured through estimate overlap) are also important measures of HR and a similar study investigating these two metrics would be necessary to fully characterise the influence of data characteristics and estimators on chimpanzee HR comparability.
Appendixes

Appendix A: Supplementary Details

Table 12. Kanyawara chimpanzee subjects and sample sizes for the year 2017. Rows in grey relate to the four individuals examined in this study.

<table>
<thead>
<tr>
<th>Chimp ID</th>
<th>Sex</th>
<th># Tracking Days</th>
<th>Sample Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>AT</td>
<td>M</td>
<td>156</td>
<td>87211</td>
</tr>
<tr>
<td>BB</td>
<td>M</td>
<td>104</td>
<td>47754</td>
</tr>
<tr>
<td>ES</td>
<td>M</td>
<td>166</td>
<td>87815</td>
</tr>
<tr>
<td>LK</td>
<td>M</td>
<td>154</td>
<td>77892</td>
</tr>
<tr>
<td>MX</td>
<td>M</td>
<td>119</td>
<td>48741</td>
</tr>
<tr>
<td>OG</td>
<td>M</td>
<td>167</td>
<td>82509</td>
</tr>
<tr>
<td>PB</td>
<td>M</td>
<td>140</td>
<td>68356</td>
</tr>
<tr>
<td>TJ</td>
<td>M</td>
<td>124</td>
<td>66692</td>
</tr>
<tr>
<td>TT</td>
<td>M</td>
<td>145</td>
<td>73179</td>
</tr>
<tr>
<td>YB</td>
<td>M</td>
<td>99</td>
<td>45469</td>
</tr>
<tr>
<td>AL</td>
<td>F</td>
<td>118</td>
<td>54585</td>
</tr>
<tr>
<td>DL</td>
<td>F</td>
<td>160</td>
<td>79129</td>
</tr>
<tr>
<td>GG</td>
<td>F</td>
<td>44</td>
<td>12740</td>
</tr>
<tr>
<td>JU</td>
<td>F</td>
<td>112</td>
<td>50565</td>
</tr>
<tr>
<td>LN</td>
<td>F</td>
<td>71</td>
<td>27127</td>
</tr>
<tr>
<td>ML</td>
<td>F</td>
<td>139</td>
<td>53558</td>
</tr>
<tr>
<td>NP</td>
<td>F</td>
<td>107</td>
<td>45876</td>
</tr>
<tr>
<td>OM</td>
<td>F</td>
<td>115</td>
<td>59670</td>
</tr>
<tr>
<td>OT</td>
<td>F</td>
<td>129</td>
<td>67920</td>
</tr>
<tr>
<td>OU</td>
<td>F</td>
<td>55</td>
<td>27544</td>
</tr>
<tr>
<td>PO</td>
<td>F</td>
<td>119</td>
<td>56021</td>
</tr>
<tr>
<td>QT</td>
<td>F</td>
<td>153</td>
<td>66719</td>
</tr>
<tr>
<td>RD</td>
<td>F</td>
<td>4</td>
<td>622</td>
</tr>
<tr>
<td>TG</td>
<td>F</td>
<td>138</td>
<td>68225</td>
</tr>
<tr>
<td>UM</td>
<td>F</td>
<td>104</td>
<td>45396</td>
</tr>
<tr>
<td>WA</td>
<td>F</td>
<td>116</td>
<td>48684</td>
</tr>
<tr>
<td>WL</td>
<td>F</td>
<td>146</td>
<td>70181</td>
</tr>
</tbody>
</table>
Table 13. Select dataset characteristics from nineteen studies of chimpanzee HR analysis. Values listed pertain to the individual datasets analysed for HR within the study. Where a range of values is provided, multiple HR assessments were made. Where there was insufficient information to assess a characteristic, it is denoted as “-”.

<table>
<thead>
<tr>
<th>Publication</th>
<th>Number of Relocations</th>
<th>Sampling Duration&lt;sup&gt;a&lt;/sup&gt;</th>
<th>Sampling Interval&lt;sup&gt;b&lt;/sup&gt;</th>
<th>Tracking Days</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wrangham, 1975</td>
<td>40 - 1400</td>
<td>1 year</td>
<td>30 min</td>
<td>-</td>
</tr>
<tr>
<td>Hasegawa, 1990</td>
<td>1843</td>
<td>1 year</td>
<td>30 min</td>
<td>-</td>
</tr>
<tr>
<td>Chapman &amp; Wrangham, 1993</td>
<td>44 - 409</td>
<td>3 years</td>
<td>24 hr</td>
<td>44 - 409</td>
</tr>
<tr>
<td>Herbinger, Boesch &amp; Rothe, 2001</td>
<td>3212 - 4470</td>
<td>10 months</td>
<td>30 min</td>
<td>3 - 26</td>
</tr>
<tr>
<td>Lehmann &amp; Boesch, 2003</td>
<td>-</td>
<td>10 years</td>
<td>-</td>
<td>2628</td>
</tr>
<tr>
<td>Mitani &amp; Amsler, 2003</td>
<td>-</td>
<td>4 years</td>
<td>24 hr</td>
<td>511</td>
</tr>
<tr>
<td>Newton-Fisher, 2003</td>
<td>111 - 836</td>
<td>15 months</td>
<td>4 hr</td>
<td>-</td>
</tr>
<tr>
<td>Williams et al., 2004</td>
<td>6245 - 14002</td>
<td>1 year</td>
<td>15 min</td>
<td>-</td>
</tr>
<tr>
<td>Basabose, 2005</td>
<td>-</td>
<td>5 years</td>
<td>-</td>
<td>729</td>
</tr>
<tr>
<td>Lehmann &amp; Boesch, 2005</td>
<td>-</td>
<td>1 year</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Murray, Mane &amp; Pusey 2007</td>
<td>18 – 32</td>
<td>2 years</td>
<td>4-6 hr</td>
<td>-</td>
</tr>
<tr>
<td>Wilson, Hauser &amp; Wrangham, 2007</td>
<td>-</td>
<td>3 years</td>
<td>15 min</td>
<td>-</td>
</tr>
<tr>
<td>Amsler, 2009</td>
<td>251 - 4459</td>
<td>33 months</td>
<td>30 min - 24 hr</td>
<td>251</td>
</tr>
<tr>
<td>Bertolani, 2010</td>
<td>60 - 30,543</td>
<td>20 months</td>
<td>30 min - 110 hr</td>
<td>60 - 393</td>
</tr>
<tr>
<td>Bortolamiol et al., 2014</td>
<td>2051</td>
<td>4 years</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Bortolamiol et al., 2016</td>
<td>79 - 285</td>
<td>4 years</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Moore et al., 2018</td>
<td>? - 6271</td>
<td>1 month - 15 yr</td>
<td>6 hr</td>
<td>? - 2,888</td>
</tr>
<tr>
<td>Green, Boruff &amp; Grueter, 2019</td>
<td>551</td>
<td>14 months</td>
<td>3 hrs</td>
<td>106</td>
</tr>
<tr>
<td>Vieira, Kerry &amp; Hockings, 2019</td>
<td>833</td>
<td>10 months</td>
<td>-</td>
<td>77</td>
</tr>
</tbody>
</table>

<sup>a</sup> Sampling duration defined as the time elapsed between the first and last observations in a dataset, inclusive of any interruptions in data collection.

<sup>b</sup> During daytime (non-nesting) hours only.
Appendix B: Scripts

B.1 Build PostgreSQL Relational Database (SQL)

```
CREATE DATABASE chimpanzee;
\c chimpanzee

CREATE TABLE chimp_individuals (  
    chimp_id VARCHAR(4) NOT NULL PRIMARY KEY,  
    chimp_name VARCHAR(50) NOT NULL,  
    chimp_sex CHAR(1) NOT NULL,  
    chimp_mother_id VARCHAR(4),  
    chimp_father_id VARCHAR(4),  
    chimp_date_of_birth DATE,  
    chimp_date_last_seen DATE,  
    female_date_of_adulthood DATE,  
    CONSTRAINT sex_constraint CHECK (chimp_sex = 'F' OR chimp_sex = 'M' OR chimp_sex = 'U'),  
    CONSTRAINT death_constraint CHECK (chimp_date_last_seen >= chimp_date_of_birth )
    );

\copy chimp_individuals FROM  
'\:\:\users\jillianrutherford\documents\masters\databasedocuments\chimp_individuals.csv' DELIMITER ',' CSV HEADER;

CREATE TABLE KPC_scans (  
    scan_increment_key VARCHAR(50) NOT NULL PRIMARY KEY,  
    date DATE NOT NULL,  
    scan_time TIME(0) NOT NULL,  
    observer_id VARCHAR(4) NOT NULL,  
    kpc_report_id VARCHAR(30) NOT NULL,  
    track_id VARCHAR(10),  
    track_increment_key VARCHAR(50) UNIQUE
    );

\copy KPC_scans FROM  
'\:\:\users\jillianrutherford\documents\masters\databasedocuments\KPC_scans.csv' DELIMITER ',' NULL AS 'NULL' CSV HEADER;

CREATE TABLE chimps_in_scan (  
    scan_increment_key VARCHAR(50) NOT NULL REFERENCES kpc_scans(scan_increment_key)  
    ON DELETE CASCADE ON UPDATE CASCADE,  
    chimp_id VARCHAR(4) NOT NULL REFERENCES chimp_individuals(chimp_id)  
    ON DELETE CASCADE ON UPDATE CASCADE);  

\copy chimps_in_scan FROM  
'\:\:\users\jillianrutherford\documents\masters\databasedocuments\chimps_in_scan2009.csv' DELIMITER ',' CSV HEADER;
\copy chimps_in_scan FROM  
'\:\:\users\jillianrutherford\documents\masters\databasedocuments\chimps_in_scan2010.csv' DELIMITER ',' CSV;
\copy chimps_in_scan FROM  
'\:\:\users\jillianrutherford\documents\masters\databasedocuments\chimps_in_scan2011.csv' DELIMITER ',' CSV;
\copy chimps_in_scan FROM  
'\:\:\users\jillianrutherford\documents\masters\databasedocuments\chimps_in_scan2012.csv' DELIMITER ',' CSV;
\copy chimps_in_scan FROM  
'\:\:\users\jillianrutherford\documents\masters\databasedocuments\chimps_in_scan2013.csv' DELIMITER ',' CSV;
\copy chimps_in_scan FROM  
'\:\:\users\jillianrutherford\documents\masters\databasedocuments\chimps_in_scan2016.csv' DELIMITER ',' CSV;
```
\copy chimps_in_scan FROM 'C:\Users\jillianrutherford\Documents\Masters\DatabaseDocuments\chimps_in_scan2017.csv' DELIMITER ',' CSV;

CREATE TABLE relocations (
    object_id INT NOT NULL PRIMARY KEY,
    elevation DOUBLE PRECISION NOT NULL,
    date_time TIMESTAMP NOT NULL,
    track_increment_key VARCHAR(50),
    latitude DOUBLE PRECISION,
    longitude DOUBLE PRECISION,
    geometry GEOMETRY,
    CONSTRAINT enforce_dimensions CHECK (st_ndims(geometry) = 2),
    CONSTRAINT enforce_geomtype CHECK (geometrytype(geometry) = 'POINT'::text OR geometry IS NULL) );

\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\COPY relocations2016.csv FROM 'C:\Users\jillianrutherford\Documents\Masters\DatabaseDocuments\relocations2009.csv' DELIMITER ',' CSV HEADER;
\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\COPY relocations2012.csv FROM 'C:\Users\jillianrutherford\Documents\Masters\DatabaseDocuments\relocations2010.csv' DELIMITER ',' CSV HEADER;
\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\COPY relocations2014.csv FROM 'C:\Users\jillianrutherford\Documents\Masters\DatabaseDocuments\relocations2011.csv' DELIMITER ',' CSV HEADER;
\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\COPY relocations2016.csv FROM 'C:\Users\jillianrutherford\Documents\Masters\DatabaseDocuments\relocations2013.csv' DELIMITER ',' CSV HEADER;
\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\COPY relocations2016.csv FROM 'C:\Users\jillianrutherford\Documents\Masters\DatabaseDocuments\relocations2014.csv' DELIMITER ',' CSV HEADER;
\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\COPY relocations2016.csv FROM 'C:\Users\jillianrutherford\Documents\Masters\DatabaseDocuments\relocations2016.csv' DELIMITER ',' CSV HEADER;
\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\copy relocations\COPY relocations2016.csv FROM 'C:\Users\jillianrutherford\Documents\Masters\DatabaseDocuments\relocations2017.csv' DELIMITER ',' CSV HEADER;

UPDATE relocations
SET geometry = ST_GeomFromText('POINT(' || longitude || ' ' || latitude || ')', 32636);

DELETE FROM relocations WHERE NOT EXISTS
(SELECT FROM kpc_scans WHERE kpc_scans.track_increment_key = relocations.track_increment_key);

ALTER TABLE relocations
ADD CONSTRAINT relocations_fkey FOREIGN KEY (track_increment_key) REFERENCES kpc_scans (track_increment_key) ON DELETE CASCADE ON UPDATE CASCADE;
B.2 Query PostgreSQL for Original Dataset (SQL)

\copy(
  SELECT relocations.*,
    chimp_individuals.chimp_id
  FROM relocations natural
  JOIN kpc_scans natural
  JOIN chimp_individuals
  WHERE chimp_individuals.chimp_id = 'WL'
  AND kpc_scans.date::text LIKE '2017%'
) TO 'C:\Users\jillianrutherford\Documents\Masters\RelocationExtracts\RX_2017WL.csv' delimiter ',' csv header;

B.3 Query PostgreSQL for Datasets with 6-month Sampling Duration (SQL)

\copy (WITH temp AS
  (SELECT relocations.*,
    chimp_individuals.chimp_id
  FROM relocations natural
  JOIN kpc_scans natural
  JOIN chimp_individuals
  WHERE chimp_individuals.chimp_id = 'TT'
  AND relocations.date_time >= (SELECT timestamp '2017-01-01 00:00:00' +
    random() * (timestamp '2017-07-01 00:00:00' - timestamp '2017-01-01 00:00:00'))
  AND relocations.date_time < date '2018-01-01')
  SELECT *
  FROM temp
  WHERE date_time <= (SELECT date_time FROM temp LIMIT 1)::date + integer '180')
TO 'C:\Users\jillianrutherford\Documents\Masters\Thesis Queried Datasets\TT\RX_2017TT_SD_6m_01.csv' delimiter ',' csv header;

B.4 Query PostgreSQL for Datasets with 3-month Sampling Duration (SQL)

\copy (WITH temp AS
  (SELECT relocations.*,
    chimp_individuals.chimp_id
  FROM relocations natural
  JOIN kpc_scans natural
  JOIN chimp_individuals
  WHERE chimp_individuals.chimp_id = 'TT'
  AND relocations.date_time >= (SELECT timestamp '2017-01-01 00:00:00' +
    random() * (timestamp '2017-10-01 00:00:00' - timestamp '2017-01-01 00:00:00'))
  AND relocations.date_time < date '2018-01-01')
  SELECT *
  FROM temp
  WHERE date_time <= (SELECT date_time FROM temp LIMIT 1)::date + integer '90')
TO 'C:\Users\jillianrutherford\Documents\Masters\Thesis Queried Datasets\TT\RX_2017TT_SD_3m_01.csv' delimiter ',' csv header;
B.5 Query PostgreSQL for Datasets with 1-month Sampling Duration (SQL)

```
\copy (WITH temp AS
  (SELECT relocations.*,
    chimp_individuals.chimp_id
  FROM relocations natural
  JOIN kpc_scans natural
  JOIN chimp_individuals
  WHERE chimp_individuals.chimp_id = 'TT'
  AND relocations.date_time >= (SELECT timestamp '2017-01-01 00:00:00' +
    random() * (timestamp '2017-12-01 00:00:00' - timestamp '2017-01-01 00:00:00'))
  AND relocations.date_time < date '2018-01-01')
  SELECT *
  FROM temp
  WHERE date_time <= (SELECT date_time FROM temp LIMIT 1)::date + integer '30')
  TO 'C:\Users\jillianrutherford\Documents\Masters\Thesis Queried
Datasets\TT\RX_2017TT_SD_1m_01.csv' delimiter ',' csv header;
```

B.6 Generate Permutated Datasets with 15-minute Sampling Interval (Python)

```python
import csv, sys, operator, datetime

#Specify source file root and output file root
filein = r'C:\Users\jillianrutherford\Documents\Masters\RelocationExtracts\OriginalDBQueryedDatasets\DL\RX_2017DL_SD1m_SI1s_'
fileout = r'C:\Users\jillianrutherford\Documents\Masters\RelocationExtracts\OriginalDBQueryedDatasets\DL\RX_2017DL_SD1m_SI15m_'

#Round1
with open(filein + '01.csv') as csvfile:
    #Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    #Loop through the rows of the source file and append those matching the
specified time marks
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "00":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "15":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "30":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "45":
            entry = line
            listoflists.append(entry)
    #Create a new list object (to weed out duplicates from the previous one)
    newlistoflists = []
newlistoflists.append(["object_id", 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])

#Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
for index, elem in enumerate(listoflists):
    checkid = elem[2]
    checkid = checkid[:16]
    laastelem = (listoflists[index-1])
    laastelemtime = laastelem[2]
    laastelemtime = laastelemtime[:16]
    if checkid != laastelemtime:
        newlistoflists.append(elem)

outname = fileout + '01.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerow(newlistoflists)

#Round 2
with open(filein + '02.csv') as csvfile:
    #Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(2))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    #Loop through the rows of the source file and append those matching the specified time marks
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "01":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "16":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "31":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "46":
            entry = line
            listoflists.append(entry)
    #Create a new list object (to weed out duplicates from the previous one)
    newlistoflists = []
    newlistoflists.append(["object_id", 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])
    #Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
    for index, elem in enumerate(listoflists):
        checkid = elem[2]
        checkid = checkid[:16]
        laastelem = (listoflists[index-1])
        laastelemtime = laastelem[2]
        laastelemtime = laastelemtime[:16]
        if checkid != laastelemtime:
            newlistoflists.append(elem)

outname = fileout + '02.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)

#Round 3
with open(filein + '03.csv') as csvfile:
    #Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    #Loop through the rows of the source file and append those matching the
    specified time marks
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "03":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "18":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "33":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "48":
            entry = line
            listoflists.append(entry)
    #Create a new list object (to weed out duplicates from the previous one)
    newlistoflists = []
    newlistoflists.append(['object_id','elevation','date_time','track_increment_key','latitude','longitude','geometry'])
    #Loop through the rows of the original list object to extract the datetime and
    compare it to the previous list entry. If unique, append to the new list. If
    duplicate, leave out.
    for index, elem in enumerate(listoflists):
        checkid = elem[2]
        checkid = checkid[:16]
        lastelem = (listoflists[index-1])
        lastelemtime = lastelem[2]
        lastelemtime = lastelemtime[:16]
        if checkid != lastelemtime:
            newlistoflists.append(elem)

outname = fileout + '03.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)

#Round 4
with open(filein + '04.csv') as csvfile:
    #Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    #Loop through the rows of the source file and append those matching the
    specified time marks
    for line in rowlist:
        rowtime = str(line[2])
if rowtime[-5:-3] in "04":
    entry = line
    listoflists.append(entry)
elif rowtime[-5:-3] in "19":
    entry = line
    listoflists.append(entry)
elif rowtime[-5:-3] in "34":
    entry = line
    listoflists.append(entry)
elif rowtime[-5:-3] in "49":
    entry = line
    listoflists.append(entry)
#Create a new list object (to weed out duplicates from the previous one)
newlistoflists = []

newlistoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])

#Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
for index, elem in enumerate(listoflists):
    checkid = elem[2]
    checkid = checkid[:16]
    lastelem = (listoflists[index - 1])
    lastelemtime = lastelem[2]
    lastelemtime = lastelemtime[:16]
    if checkid != lastelemtime:
        newlistoflists.append(elem)

outname = fileout + '04.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerow(newlistoflists)

#Round 5
with open(filein + '05.csv') as csvfile:
    data = csv.reader(csvfile, delimiter = ',')
    rowlist = sorted(data, key = operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    #Loop through the rows of the source file and append those matching the specified time marks
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "05":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "20":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "35":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "50":
            entry = line
            listoflists.append(entry)
        #Create a new list object (to weed out duplicates from the previous one)
        newlistoflists = []
newlistoflists.append(["object_id", 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])

# Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
for index, elem in enumerate(listoflists):
    checkid = elem[2]
    checkid = checkid[:16]
    lastelem = (listoflists[index-1])
    lastelemtime = lastelem[2]
    lastelemtime = lastelemtime[:16]
    if checkid != lastelemtime:
        newlistoflists.append(elem)

outname = fileout + '05.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)

# Round 6
with open(filein + '06.csv') as csvfile:
    # Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=’,’)
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    # Create a new list object
    listoflists = []
    # Loop through the rows of the source file and append those matching the specified time marks
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "06":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "21":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "36":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "51":
            entry = line
            listoflists.append(entry)
        # Create a new list object (to weed out duplicates from the previous one)
        newlistoflists = []
        newlistoflists.append(["object_id", 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])
        # Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
        for index, elem in enumerate(listoflists):
            checkid = elem[2]
            checkid = checkid[:16]
            lastelem = (listoflists[index-1])
            lastelemtime = lastelem[2]
            lastelemtime = lastelemtime[:16]
            if checkid != lastelemtime:
                newlistoflists.append(elem)

outname = fileout + '06.csv'
```python
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)

#Round 7
with open(filein + '07.csv') as csvfile:
    #Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    #Loop through the rows of the source file and append those matching the
    specified time marks
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "07":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "22":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "37":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "52":
            entry = line
            listoflists.append(entry)

    #Create a new list object (to weed out duplicates from the previous one)
    newlistoflists = []

    #Loop through the rows of the original list object to extract the datetime and
    compare it to the previous list entry. If unique, append to the new list. If
    duplicate, leave out.
    for index, elem in enumerate(listoflists):
        checkid = elem[2]
        checkid = checkid[:16]
        lastelem = listoflists[index-1]
        lastelemtime = lastelem[2]
        lastelemtime = lastelemtime[:16]
        if checkid != lastelemtime:
            newlistoflists.append(elem)

outname = fileout + '07.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)

#Round 8
with open(filein + '08.csv') as csvfile:
    #Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    #Loop through the rows of the source file and append those matching the
    specified time marks
    for line in rowlist:
        rowtime = str(line[2])
```

if rowtime[-5:-3] in "08":
    entry = line
    listoflists.append(entry)
else if rowtime[-5:-3] in "23":
    entry = line
    listoflists.append(entry)
elif rowtime[-5:-3] in "38":
    entry = line
    listoflists.append(entry)
elif rowtime[-5:-3] in "53":
    entry = line
    listoflists.append(entry)

#Create a new list object (to weed out duplicates from the previous one)
newlistoflists = []
newlistoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])

#Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
for index, elem in enumerate(listoflists):
    checkid = elem[2]
    checkid = checkid[0:16]
    lastelem = (listoflists[index-1])
    lastelemtime = lastelem[2]
    lastelemtime = lastelemtime[0:16]
    if checkid != lastelemtime:
        newlistoflists.append(elem)

outname = fileout + '08.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerow(newlistoflists)

#Round 9
with open(filein + '09.csv') as csvfile:
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    newlistoflists = []
    #Loop through the rows of the source file and append those matching the specified time marks
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "09":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "24":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "39":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "54":
            entry = line
            listoflists.append(entry)
        #Create a new list object (to weed out duplicates from the previous one)
        newlistoflists = []
        newlistoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])
# Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.

```python
for index, elem in enumerate(listoflists):
    checkid = elem[2]
    checkid = checkid[:16]
    lastelem = listoflists[index-1]
    lastelemtime = lastelem[2]
    lastelemtime = lastelemtime[:16]
    if checkid != lastelemtime:
        newlistoflists.append(elem)
```

```python
outname = fileout + '09.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)
```

# Round 10
```python
with open(filein + '10.csv') as csvfile:
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    listoflists = []
    # Loop through the rows of the original list object and append those matching the specified time marks
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "10":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "25":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "40":
            entry = line
            listoflists.append(entry)
        elif rowtime[-5:-3] in "55":
            entry = line
            listoflists.append(entry)
    newlistoflists = []
    # Create a new list object (to weed out duplicates from the previous one)
    newlistoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key',
                             'latitude', 'longitude', 'geometry'])
    # Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
    for index, elem in enumerate(listoflists):
        checkid = elem[2]
        checkid = checkid[:16]
        lastelem = listoflists[index-1]
        lastelemtime = lastelem[2]
        lastelemtime = lastelemtime[:16]
        if checkid != lastelemtime:
            newlistoflists.append(elem)
```

```python
outname = fileout + '10.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)
```
B.7 Generate Permuted Datasets with 1-hour Sampling Interval (Python)

```python
import csv, sys, operator, datetime

#Specify source file root and output file root
filein = r'C:\Users\jillianrutherford\Documents\Masters\RelocationExtracts\OriginalDBQuery
datasets\WL\RX_2017WL_SD1m_SI1s_'
fileout = r'C:\Users\jillianrutherford\Documents\Masters\RelocationExtracts\OriginalDBQuery
datasets\WL\RX_2017WL_SD1m_SI1h_'

#Round 1
with open(filein + '01.csv') as csvfile:
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    listoflists = []
    #Loop through the rows of the source file and append those matching the specified time mark
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "00":
            entry = line
            listoflists.append(entry)
    #Create a new list object (to weed out duplicates from the previous one)
    newlistoflists = []
    newlistoflists.append(['object_id','elevation','date_time','track_increment_key','latitude','longitude','geometry'])
    #Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
    for index, elem in enumerate(listoflists):
        checkid = elem[2]
        checkid = checkid[:16]
        lastelem = (listoflists[index-1])
        lastelemtime = lastelem[2]
        lastelemtime = lastelemtime[:16]
        if checkid != lastelemtime:
            newlistoflists.append(elem)
    outname = fileout + '01.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)

#Round 2
with open(filein + '02.csv') as csvfile:
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    listoflists = []
    #Loop through the rows of the source file and append those matching the specified time mark
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "05":
            entry = line
            listoflists.append(entry)
    #Create a new list object (to weed out duplicates from the previous one)
    newlistoflists = []
    newlistoflists.append(['object_id','elevation','date_time','track_increment_key','latitude','longitude','geometry'])
    #Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
    for index, elem in enumerate(listoflists):
        checkid = elem[2]
        checkid = checkid[:16]
        lastelem = (listoflists[index-1])
        lastelemtime = lastelem[2]
        lastelemtime = lastelemtime[:16]
        if checkid != lastelemtime:
            newlistoflists.append(elem)
    outname = fileout + '02.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)
```
entry = line
listoflists.append(entry)
#Create a new list object (to weed out duplicates from the previous one)
newlistoflists = []

newlistoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])

#Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
for index, elem in enumerate(listoflists):
    checkid = elem[2]
    checkid = checkid[:16]
    lastelem = listoflists[index-1]
    lastelemtime = lastelem[2]
    lastelemtime = lastelemtime[:16]
    if checkid != lastelemtime:
        newlistoflists.append(elem)

outname = fileout + '02.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)

#Round 3
with open(filein + '03.csv') as csvfile:
    #read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    #Loop through the rows of the source file and append those matching the specified time mark
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "10":
            entry = line
            listoflists.append(entry)
    #Create a new list object (to weed out duplicates from the previous one)
    newlistoflists = []

newlistoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])

#Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
for index, elem in enumerate(listoflists):
    checkid = elem[2]
    checkid = checkid[:16]
    lastelem = listoflists[index-1]
    lastelemtime = lastelem[2]
    lastelemtime = lastelemtime[:16]
    if checkid != lastelemtime:
        newlistoflists.append(elem)

outname = fileout + '03.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)

#Round 4
with open(filein + '04.csv') as csvfile:
    #read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    #Loop through the rows of the source file and append those matching the specified time mark
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "15":
            entry = line
            listoflists.append(entry)
    #Create a new list object (to weed out duplicates from the previous one)
    newlistoflists = []
    newlistoflists.append(['object_id','elevation','date_time','track_increment_key','latitude','longitude','geometry'])
    #Loop
    for index, elem in enumerate(listoflists):
        checkid = elem[2]
        checkid = checkid[:16]
        lastelem = (listoflists[index-1])
        lastelemtime = lastelem[2]
        lastelemtime = lastelemtime[:16]
        if checkid != lastelemtime:
            newlistoflists.append(elem)
    outname = fileout + '04.csv'
    with open(outname, "wb") as newfile:
        writer = csv.writer(newfile)
        writer.writerows(newlistoflists)

#Round 5
with open(filein + '05.csv') as csvfile:
    #read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    #Loop through the rows of the source file and append those matching the specified time mark
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "20":
            entry = line
            listoflists.append(entry)
    #Create a new list object (to weed out duplicates from the previous one)
    newlistoflists = []
    newlistoflists.append(['object_id','elevation','date_time','track_increment_key','latitude','longitude','geometry'])
    #Loop
    for index, elem in enumerate(listoflists):
        checkid = elem[2]
        checkid = checkid[:16]
        lastelem = (listoflists[index-1])
        lastelemtime = lastelem[2]
        lastelemtime = lastelemtime[:16]
        if checkid != lastelemtime:
newlistoflists.append(elem)

outname = fileout + '05.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)

# Round 6
with open(filein + '06.csv') as csvfile:
    # read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    # Create a new list object
    listoflists = []
    # Loop through the rows of the source file and append those matching the specified time mark
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "25":
            entry = line
            listoflists.append(entry)
    # Create a new list object (to weed out duplicates from the previous one)
    newlistoflists = []
    newlistoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])
    # Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
    for index, elem in enumerate(listoflists):
        checkid = elem[2]
        checkid = checkid[:16]
        lastelem = (listoflists[index-1])
        lastelemtime = lastelem[2]
        lastelemtime = lastelemtime[:16]
        if checkid != lastelemtime:
            newlistoflists.append(elem)

outname = fileout + '06.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)

# Round 7
with open(filein + '07.csv') as csvfile:
    # read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    # Create a new list object
    listoflists = []
    # Loop through the rows of the source file and append those matching the specified time mark
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "30":
            entry = line
            listoflists.append(entry)
    # Create a new list object (to weed out duplicates from the previous one)
    newlistoflists = []
newlistoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])

# Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
for index, elem in enumerate(listoflists):
    checkid = elem[2]
    checkid = checkid[:16]
    lastelem = (listoflists[index-1])
    lastelemtime = laastelem[2]
    laastelemtime = laastelemtime[:16]
    if checkid != laastelemtime:
        newlistoflists.append(elem)

outname = fileout + '07.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)

# Round 8
with open(filein + '08.csv') as csvfile:
    # Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    # Create a new list object
    listoflists = []
    # Loop through the rows of the source file and append those matching the specified time mark
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "35":
            entry = line
            listoflists.append(entry)
    # Create a new list object (to weed out duplicates from the previous one)
    newlistoflists = []
    newlistoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])

    # Loop through.
    for index, elem in enumerate(listoflists):
        checkid = elem[2]
        checkid = checkid[:16]
        laastelem = (listoflists[index-1])
        laastelemtime = laastelem[2]
        laastelemtime = laastelemtime[:16]
        if checkid != laastelemtime:
            newlistoflists.append(elem)

outname = fileout + '08.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)

# Round 9
with open(filein + '09.csv') as csvfile:
    # Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    # Create a new list object
listoflists = []
#Loop through
for line in rowlist:
    rowtime = str(line[2])
    if rowtime[-5:-3] in "40":
        entry = line
        listoflists.append(entry)
#Create a new list object (to weed out duplicates from the previous one)
newlistoflists = []
newlistoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])
#Loop through.
for index, elem in enumerate(listoflists):
    checkid = elem[2]
    checkid = checkid[:16]
    lastelem = (listoflists[index - 1])
    lastelemtime = lastelem[2]
    lastelemtime = lastelemtime[:16]
    if checkid != lastelemtime:
        newlistoflists.append(elem)
outname = fileout + '09.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)

#Round 10
with open(filein + '10.csv') as csvfile:
    #read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    #Loop through the rows of the source file and append those matching the specified time mark
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "50":
            entry = line
            listoflists.append(entry)
    #Create a new list object (to weed out duplicates from the previous one)
    newlistoflists = []
newlistoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])
#Loop through.
for index, elem in enumerate(listoflists):
    checkid = elem[2]
    checkid = checkid[:16]
    lastelem = (listoflists[index - 1])
    lastelemtime = lastelem[2]
    lastelemtime = lastelemtime[:16]
    if checkid != lastelemtime:
        newlistoflists.append(elem)
outname = fileout + '10.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)
import csv, sys, operator, datetime

#Specify source file root and output file root
filein = r'C:\Users\jillianrutherford\Documents\Masters\RelocationExtracts\OriginalDBQuery_datasets\WL\RX_2017WL_SD1m_SI1s_'
fileout = r'C:\Users\jillianrutherford\Documents\Masters\RelocationExtracts\OriginalDBQuery_datasets\WL\RX_2017WL_SD1m_SI4h_'

#Round1
with open(filein + '01.csv') as csvfile:
    #Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    #Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
    for index, elem in enumerate(listoflists):
        checkid = elem[2]
        checkid = checkid[:16]
        lastelem = (listoflists[index-1])
        lastelemtime = lastelem[2]
        if checkid != lastelemtime:
            newlistoflists.append(elem)

#Round2
with open(outname + '01.csv') as csvfile:
    #Read in the source file and create a list of rows (minus the header)
data = csv.reader(csvfile, delimiter=',')
rowlist = sorted(data, key=operator.itemgetter(0))
rowlist.pop()
# Create a new list object
listoflists = []
# Loop through the rows of the source file and append those matching the specified time marks
for line in rowlist:
    rowtime = str(line[2])
    if rowtime[-5:-3] in "00":
        if rowtime[-8:-6] in "07":
            entry = line
            listoflists.append(entry)
        elif rowtime[-8:-6] in "11":
            entry = line
            listoflists.append(entry)
        elif rowtime[-8:-6] in "15":
            entry = line
            listoflists.append(entry)
        elif rowtime[-8:-6] in "19":
            entry = line
            listoflists.append(entry)
    elif rowtime[-5:-3] in "10":
        if rowtime[-8:-6] in "06":
            entry = line
            listoflists.append(entry)
        elif rowtime[-8:-6] in "10":
            entry = line
            listoflists.append(entry)
# Create a new list object (to weed out duplicates from the previous one)
newlistoflists = []
newlistoflists.append(["object_id","elevation","date_time","track_increment_key","latitude","longitude","geometry"])
# Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
for index, elem in enumerate(listoflists):
    checkid = elem[2]
    checkid = checkid[:16]
    lastelem = (listoflists[index-1])
    lastelemtime = lastelem[2]
    lastelemtime = lastelemtime[:16]
    if checkid != lastelemtime:
        newlistoflists.append(elem)
outname = fileout + '02.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)

# Round 3
with open(filein + '03.csv') as csvfile:
    # Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    # Create a new list object
    listoflists = []
    # Loop through the rows of the source file and append those matching the specified time marks
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "10":
            if rowtime[-8:-6] in "06":
                entry = line
                listoflists.append(entry)
            elif rowtime[-8:-6] in "10":
                entry = line
                listoflists.append(entry)
entry = line
listoflists.append(entry)
else if rowtime[-8:-6] in "14":
    entry = line
listoflists.append(entry)
else if rowtime[-8:-6] in "18":
    entry = line
listoflists.append(entry)
# Create a new list object (to weed out duplicates from the previous one)
newlistoflists = []

newlistoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])
# Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
    for index, elem in enumerate(listoflists):
        checkid = elem[2]
        checkid = checkid[:16]
        lastelem = (listoflists[index-1])
        lastelemtime = lastelem[2]
        lastelemtime = lastelemtime[:16]
        if checkid != lastelemtime:
            newlistoflists.append(elem)

outname = fileout + '.03.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)

# Round4
with open(filein + '.04.csv') as csvfile:
    # Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    # Create a new list object
    listoflists = []
    # Loop through the rows of the source file and append those matching the specified time marks
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "20":
            if rowtime[-8:-6] in "06":
                entry = line
                listoflists.append(entry)
            else if rowtime[-8:-6] in "10":
                entry = line
                listoflists.append(entry)
            elif rowtime[-8:-6] in "14":
                entry = line
                listoflists.append(entry)
            elif rowtime[-8:-6] in "18":
                entry = line
                listoflists.append(entry)
        # Create a new list object (to weed out duplicates from the previous one)
        newlistoflists = []

        newlistoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])
# Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.

```python
for index, elem in enumerate(listoflists):
    checkid = elem[2]
    checkid = checkid[:16]
    lastelem = (listoflists[index-1])
    lastelemtime = lastelem[2]
    lastelemtime = lastelemtime[:16]
    if checkid != lastelemtime:
        newlistoflists.append(elem)
```

outname = fileout + '04.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)

# Round5
with open(filein + '05.csv') as csvfile:
    # Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    # Create a new list object
    listoflists = []
    # Loop through the rows of the source file and append those matching the specified time marks
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "30":
            if rowtime[-8:-6] in "06":
                entry = line
                listoflists.append(entry)
            elif rowtime[-8:-6] in "10":
                entry = line
                listoflists.append(entry)
            elif rowtime[-8:-6] in "14":
                entry = line
                listoflists.append(entry)
            elif rowtime[-8:-6] in "18":
                entry = line
                listoflists.append(entry)
        # Create a new list object (to weed out duplicates from the previous one)
        newlistoflists = []
        newlistoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])
        # Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
        for index, elem in enumerate(listoflists):
            checkid = elem[2]
            checkid = checkid[:16]
            lastelem = (listoflists[index-1])
            lastelemtime = lastelem[2]
            lastelemtime = lastelemtime[:16]
            if checkid != lastelemtime:
                newlistoflists.append(elem)
```

outname = fileout + '05.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
with open(filein + '06.csv') as csvfile:
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    #Loop through the rows of the source file and append those matching the specified time marks
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "40":
            entry = line
            listoflists.append(entry)
        elif rowtime[-8:-6] in "06":
            entry = line
            listoflists.append(entry)
        elif rowtime[-8:-6] in "10":
            entry = line
            listoflists.append(entry)
        elif rowtime[-8:-6] in "14":
            entry = line
            listoflists.append(entry)
        elif rowtime[-8:-6] in "18":
            entry = line
            listoflists.append(entry)
    #Create a new list object (to weed out duplicates from the previous one)
    newlistoflist = []
    #Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
    for index, elem in enumerate(listoflists):
        checkid = elem[2]
        checkid = checkid[:16]
        lastelem = (listoflists[index-1])
        lastelemtime = lastelem[2]
        lastelemtime = lastelemtime[:16]
        if checkid != lastelemtime:
            newlistoflist.append(elem)

outname = fileout + '06.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflist)

#Round7
with open(filein + '07.csv') as csvfile:
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    #Loop through the rows of the source file and append those matching the specified time marks
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "50":
            entry = line
            listoflists.append(entry)
```python
if rowtime[-8:-6] in "06":
    entry = line
    listoflists.append(entry)
elif rowtime[-8:-6] in "10":
    entry = line
    listoflists.append(entry)
elif rowtime[-8:-6] in "14":
    entry = line
    listoflists.append(entry)
elif rowtime[-8:-6] in "18":
    entry = line
    listoflists.append(entry)
#Create a new list object (to weed out duplicates from the previous one)
newlistoflists = []
newlistoflists.append(["object_id","elevation","date_time","track_increment_key","latitude","longitude","geometry")
#Loop through the rows of the original list object to extract the datetime
#and compare it to the previous list entry. If unique, append to the new list. If
duplicate, leave out.
for index, elem in enumerate(listoflists):
    checkid = elem[2]
    checkid = checkid[:16]
    lastelem = (listoflists[index-1])
    lastelemtime = lastelem[2]
    lastelemtime = lastelemtime[:16]
    if checkid != lastelemtime:
        newlistoflists.append(elem)
outname = fileout + '07.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)
#Round8
with open(filein + '08.csv') as csvfile:
    #Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter='
')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    #Loop through the rows of the source file and append those matching the
    specified time marks
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "05":
            if rowtime[-8:-6] in "06":
                entry = line
                listoflists.append(entry)
            elif rowtime[-8:-6] in "10":
                entry = line
                listoflists.append(entry)
            elif rowtime[-8:-6] in "14":
                entry = line
                listoflists.append(entry)
            elif rowtime[-8:-6] in "18":
                entry = line
                listoflists.append(entry)
        #Create a new list object (to weed out duplicates from the previous one)
        newlistoflists = []
```
newlistoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])

# Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.

for index, elem in enumerate(listoflists):
    checkid = elem[2]
    checkid = checkid[:16]
    lastelem = (listoflists[index-1])
    lastelemtime = lastelem[2]
    lastelemtime = lastelemtime[:16]
    if checkid != lastelemtime:
        newlistoflists.append(elem)

outname = fileout + '08.csv'
with open(outname, 'wb') as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)

# Round9
with open(filein + '09.csv') as csvfile:
    # Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    # Create a new list object
    listoflists = []
    # Loop through the rows of the source file and append those matching the specified time marks
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in '00':
            if rowtime[-8:-6] in '05':
                entry = line
                listoflists.append(entry)
            elif rowtime[-8:-6] in '19':
                entry = line
                listoflists.append(entry)
            elif rowtime[-8:-6] in '13':
                entry = line
                listoflists.append(entry)
            elif rowtime[-8:-6] in '17':
                entry = line
                listoflists.append(entry)
        # Create a new list object (to weed out duplicates from the previous one)
        newlistoflists = []
        newlistoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])
        # Loop through the rows of the original list object to extract the datetime and compare it to the previous list entry. If unique, append to the new list. If duplicate, leave out.
        for index, elem in enumerate(listoflists):
            checkid = elem[2]
            checkid = checkid[:16]
            lastelem = (listoflists[index-1])
            lastelemtime = lastelem[2]
            lastelemtime = lastelemtime[:16]
            if checkid != lastelemtime:
                newlistoflists.append(elem)
outname = fileout + '09.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)

#Round10
with open(filein + '10.csv') as csvfile:
    #Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    #Loop through the rows of the source file and append those matching the
    specified time marks
    for line in rowlist:
        rowtime = str(line[2])
        if rowtime[-5:-3] in "50":
            if rowtime[-8:-6] in "05":
                entry = line
                listoflists.append(entry)
            elif rowtime[-8:-6] in "19":
                entry = line
                listoflists.append(entry)
            elif rowtime[-8:-6] in "13":
                entry = line
                listoflists.append(entry)
            elif rowtime[-8:-6] in "17":
                entry = line
                listoflists.append(entry)
        #Create a new list object (to weed out duplicates from the previous one)
        newlistoflists = []
        newlistoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key',
                               'latitude', 'longitude', 'geometry'])
        #Loop through the rows of the original list object to extract the datetime
        #and compare it to the previous list entry. If unique, append to the new list. If
duplicate, leave out.
        for index, elem in enumerate(listoflists):
            checkid = elem[2]
            checkid = checkid[:16]
            lastelem = (listoflists[index-1])
            lastelemtime = lastelem[2]
            lastelemtime = lastelemtime[:16]
            if checkid != lastelemtime:
                newlistoflists.append(elem)

outname = fileout + '10.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(newlistoflists)
B.9 Generate Permuted Datasets with 24-hour Sampling Interval (Python)

```python
import csv, sys, operator, datetime

# Specify source file root and output file root
filein = r'C:\Users\jillianrutherford\Documents\Masters\RelocationExtracts\OriginalDBQueries\datasets\WL\RX_2017WL_SD1m_SI1s_'
fileout = r'C:\Users\jillianrutherford\Documents\Masters\RelocationExtracts\OriginalDBQueries\datasets\WL\RX_2017WL_SD1m_SI24h_'

# Round 1
with open(filein + '01.csv') as csvfile:
    # Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    # Create a new list object
    listoflists = []
    listoflists.append(['object_id','elevation','date_time','track_increment_key','latitude','longitude','geometry'])
    # Loop through the rows of the source file and add the first record from each unique date
    for index, elem in enumerate(rowlist):
        checkid = str(elem[2])
        checkid = checkid[:10]
        lastelem = (rowlist[index-1])
        lastelemtime = lastelem[2]
        lastelemtime = lastelemtime[:10]
        if checkid != lastelemtime:
            listoflists.append(elem)

outname = fileout + '01.csv'
with open(outname, 'wb') as newfile:
    writer = csv.writer(newfile)
    writer.writerows(listoflists)

# Round 2
with open(filein + '02.csv') as csvfile:
    # Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    # Create a new list object
    listoflists = []
    listoflists.append(['object_id','elevation','date_time','track_increment_key','latitude','longitude','geometry'])
    # Loop through the rows of the source file and add the second record from each unique date
    for index, elem in enumerate(rowlist):
        checkid = str(elem[2])
        checkid = checkid[:10]
        lastelem = (rowlist[index-1])
        lastelemtime = lastelem[2]
        lastelemtime = lastelemtime[:10]
        lastelem2 = (rowlist[index-2])
        lastelemtime2 = lastelem2[2]
        lastelemtime2 = lastelemtime2[:10]
        if checkid == lastelemtime and checkid != lastelemtime2:
```
listoflists.append(elem)

outname = fileout + '02.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(listoflists)

#Round 3
with open(filein + '03.csv') as csvfile:
    #Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    listoflists.append(['object_id','elevation','date_time','track_increment_key','latitude','longitude','geometry'])
    #Loop through the rows of the source file and add the third record from each unique date
    for index, elem in enumerate(rowlist):
        checkid = str(elem[2])
        checkid = checkid[:10]
        lastelem = (rowlist[index-1])
        lastelemtime = lastelem[2]
        lastelemtime = lastelemtime[:10]
        lastelem2 = (rowlist[index-2])
        lastelemtime2 = lastelem2[2]
        lastelemtime2 = lastelemtime2[:10]
        lastelem3 = (rowlist[index-3])
        lastelemtime3 = lastelem3[2]
        lastelemtime3 = lastelemtime3[:10]
        if checkid == lastelemtime2 and checkid != lastelemtime3:
            listoflists.append(elem)

outname = fileout + '03.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(listoflists)

#Round 4
with open(filein + '04.csv') as csvfile:
    #Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    listoflists.append(['object_id','elevation','date_time','track_increment_key','latitude','longitude','geometry'])
    #Loop through the rows of the source file and add the fourth record from each unique date
    for index, elem in enumerate(rowlist):
        checkid = str(elem[2])
        checkid = checkid[:10]
        lastelem = (rowlist[index-1])
        lastelemtime = lastelem[2]
        lastelemtime = lastelemtime[:10]
        lastelem2 = (rowlist[index-2])
        lastelemtime2 = lastelem2[2]
        lastelemtime2 = lastelemtime2[:10]
lastelem3 = (rowlist[index-3])
lastelemtime3 = lastelem3[2]
lastelemtime3 = lastelemtime3[:10]
lastelem4 = (rowlist[index-4])
lastelemtime4 = lastelem4[2]
lastelemtime4 = lastelemtime4[:10]
if checkid == lastelemtime3 and checkid != lastelemtime4:
    listoflists.append(elem)

outname = fileout + '04.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(listoflists)

#Round 5
with open(filein + '05.csv') as csvfile:
    #Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []

    listoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])
    #Loop through the rows of the source file and add the fifth record from each unique date
    for index, elem in enumerate(rowlist):
        checkid = str(elem[2])
        checkid = checkid[:10]
        lastelem = (rowlist[index-1])
        lastelemtime = lastelem[2]
        lastelemtime = lastelemtime[:10]
        lastelem2 = (rowlist[index-2])
        lastelemtime2 = lastelem2[2]
        lastelemtime2 = lastelemtime2[:10]
        lastelem3 = (rowlist[index-3])
        lastelemtime3 = lastelem3[2]
        lastelemtime3 = lastelemtime3[:10]
        lastelem4 = (rowlist[index-4])
        lastelemtime4 = lastelem4[2]
        lastelemtime4 = lastelemtime4[:10]
        lastelem5 = (rowlist[index-5])
        lastelemtime5 = lastelem5[2]
        lastelemtime5 = lastelemtime5[:10]
        if checkid == lastelemtime4 and checkid != lastelemtime5:
            listoflists.append(elem)

outname = fileout + '05.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(listoflists)

#Round 6
with open(filein + '06.csv') as csvfile:
    #Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []

    listoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])
    #Loop through the rows of the source file and add the fifth record from each unique date
    for index, elem in enumerate(rowlist):
        checkid = str(elem[2])
        checkid = checkid[:10]
        lastelem = (rowlist[index-1])
        lastelemtime = lastelem[2]
        lastelemtime = lastelemtime[:10]
        lastelem2 = (rowlist[index-2])
        lastelemtime2 = lastelem2[2]
        lastelemtime2 = lastelemtime2[:10]
        lastelem3 = (rowlist[index-3])
        lastelemtime3 = lastelem3[2]
        lastelemtime3 = lastelemtime3[:10]
        lastelem4 = (rowlist[index-4])
        lastelemtime4 = lastelem4[2]
        lastelemtime4 = lastelemtime4[:10]
        lastelem5 = (rowlist[index-5])
        lastelemtime5 = lastelem5[2]
        lastelemtime5 = lastelemtime5[:10]
        if checkid == lastelemtime4 and checkid != lastelemtime5:
            listoflists.append(elem)
listoflists.append(['object_id','elevation','date_time','track_increment_key','latitude','longitude','geometry'])

# Loop through the rows of the source file and add the sixth record from each unique date
for index, elem in enumerate(rowlist):
    checkid = str(elem[2])
    checkid = checkid[:10]
    lastelem = (rowlist[index-1])
    lastelemtime = lastelem[2]
    lastelemtime = lastelemtime[:10]
    lastelem2 = (rowlist[index-2])
    lastelemtime2 = lastelem2[2]
    lastelemtime2 = lastelemtime2[:10]
    lastelem3 = (rowlist[index-3])
    lastelemtime3 = lastelem3[2]
    lastelemtime3 = lastelemtime3[:10]
    lastelem4 = (rowlist[index-4])
    lastelemtime4 = lastelem4[2]
    lastelemtime4 = lastelemtime4[:10]
    lastelem5 = (rowlist[index-5])
    lastelemtime5 = lastelem5[2]
    lastelemtime5 = lastelemtime5[:10]
    lastelem6 = (rowlist[index-6])
    lastelemtime6 = lastelem6[2]
    lastelemtime6 = lastelemtime6[:10]
    if checkid == lastelemtime5 and checkid != lastelemtime6:
        listoflists.append(elem)

outname = fileout + '06.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(listoflists)

# Round 7
with open(filein + '07.csv') as csvfile:
    # Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    # Create a new list object
    listoflists = []
    listoflists.append(['object_id','elevation','date_time','track_increment_key','latitude','longitude','geometry'])
    # Loop through the rows of the source file and add the seventh record from each unique date
    for index, elem in enumerate(rowlist):
        checkid = str(elem[2])
        checkid = checkid[:10]
        lastelem = (rowlist[index-1])
        lastelemtime = lastelem[2]
        lastelemtime = lastelemtime[:10]
        lastelem2 = (rowlist[index-2])
        lastelemtime2 = lastelem2[2]
        lastelemtime2 = lastelemtime2[:10]
        lastelem3 = (rowlist[index-3])
        lastelemtime3 = lastelem3[2]
        lastelemtime3 = lastelemtime3[:10]
        lastelem4 = (rowlist[index-4])
        lastelemtime4 = lastelem4[2]
        lastelemtime4 = lastelemtime4[:10]
lastelem5 = (rowlist[index-5])
lastelemtime5 = lastelem5[2]
lastelemtime5 = lastelemtime5[:10]
lastelem6 = (rowlist[index-6])
lastelemtime6 = lastelem6[2]
lastelemtime6 = lastelemtime6[:10]
lastelem7 = (rowlist[index-7])
lastelemtime7 = lastelem7[2]
lastelemtime7 = lastelemtime7[:10]
if checkid == lastelemtime6 and checkid != lastelemtime7:
   listoflists.append(elem)

outname = fileout + '07.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(listoflists)

#Round 8
with open(filein + '08.csv') as csvfile:
    #Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    listoflists.append(['object_id','elevation','date_time','track_increment_key','latitude','longitude','geometry'])
    #Loop through the rows of the source file and add the eigth record from each unique date
    for index, elem in enumerate(rowlist):
        checkid = str(elem[2])
        checkid = checkid[:10]
        lastelem = (rowlist[index-1])
lastelemtime = lastelem[2]
lastelemtime = lastelemtime[:10]
lastelem2 = (rowlist[index-2])
lastelemtime2 = lastelem2[2]
lastelemtime2 = lastelemtime2[:10]
lastelem3 = (rowlist[index-3])
lastelemtime3 = lastelem3[2]
lastelemtime3 = lastelemtime3[:10]
lastelem4 = (rowlist[index-4])
lastelemtime4 = lastelem4[2]
lastelemtime4 = lastelemtime4[:10]
lastelem5 = (rowlist[index-5])
lastelemtime5 = lastelem5[2]
lastelemtime5 = lastelemtime5[:10]
lastelem6 = (rowlist[index-6])
lastelemtime6 = lastelem6[2]
lastelemtime6 = lastelemtime6[:10]
lastelem7 = (rowlist[index-7])
lastelemtime7 = lastelem7[2]
lastelemtime7 = lastelemtime7[:10]
lastelem8 = (rowlist[index-8])
lastelemtime8 = lastelem8[2]
lastelemtime8 = lastelemtime8[:10]
if checkid == lastelemtime7 and checkid != lastelemtime8:
   listoflists.append(elem)

outname = fileout + '08.csv'
with open(outname, "wb") as newfile:
writer = csv.writer(newfile)
writer.writerows(listoflists)

#Round 9
with open(filein + '09.csv') as csvfile:
    #Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
    listoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])
    #Loop through the rows of the source file and add the ninth record from each unique date
    for index, elem in enumerate(rowlist):
        checkid = str(elem[2])
        checkid = checkid[:10]
        lastelem = (rowlist[index-1])
        lastelemtime = lastelem[2]
        lastelemtime = lastelemtime[:10]
        lastelem2 = (rowlist[index-2])
        lastelemtime2 = lastelem2[2]
        lastelemtime2 = lastelemtime2[:10]
        lastelem3 = (rowlist[index-3])
        lastelemtime3 = lastelem3[2]
        lastelemtime3 = lastelemtime3[:10]
        lastelem4 = (rowlist[index-4])
        lastelemtime4 = lastelem4[2]
        lastelemtime4 = lastelemtime4[:10]
        lastelem5 = (rowlist[index-5])
        lastelemtime5 = lastelem5[2]
        lastelemtime5 = lastelemtime5[:10]
        lastelem6 = (rowlist[index-6])
        lastelemtime6 = lastelem6[2]
        lastelemtime6 = lastelemtime6[:10]
        lastelem7 = (rowlist[index-7])
        lastelemtime7 = lastelem7[2]
        lastelem7 = lastelem7[:10]
        lastelem8 = (rowlist[index-8])
        lastelemtime8 = lastelem8[2]
        lastelemtime8 = lastelemtime8[:10]
        lastelem9 = (rowlist[index-9])
        lastelemtime9 = lastelem9[2]
        lastelem9 = lastelem9[:10]
        if checkid == lastelemtime8 and checkid != lastelemtime9:
            listoflists.append(elem)

outname = fileout + '09.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(listoflists)

#Round 10
with open(filein + '10.csv') as csvfile:
    #Read in the source file and create a list of rows (minus the header)
    data = csv.reader(csvfile, delimiter=',')
    rowlist = sorted(data, key=operator.itemgetter(0))
    rowlist.pop()
    #Create a new list object
    listoflists = []
listoflists.append(['object_id', 'elevation', 'date_time', 'track_increment_key', 'latitude', 'longitude', 'geometry'])

# Loop through the rows of the source file and add the tenth record from each unique date
for index, elem in enumerate(rowlist):
    checkid = str(elem[2])
    checkid = checkid[:10]
    lastelem = rowlist[index-1]
    lastelemtime = lastelem[2]
    lastelemtime = lastelemtime[:10]
    lastelem2 = rowlist[index-2]
    lastelemtime2 = lastelemtime2[:10]
    lastelem3 = rowlist[index-3]
    lastelemtime3 = lastelemtime3[:10]
    lastelem4 = rowlist[index-4]
    lastelemtime4 = lastelemtime4[:10]
    lastelem5 = rowlist[index-5]
    lastelemtime5 = lastelemtime5[:10]
    lastelem6 = rowlist[index-6]
    lastelemtime6 = lastelemtime6[:10]
    lastelem7 = rowlist[index-7]
    lastelemtime7 = lastelemtime7[:10]
    lastelem8 = rowlist[index-8]
    lastelemtime8 = lastelemtime8[:10]
    lastelem9 = rowlist[index-9]
    lastelemtime9 = lastelemtime9[:10]
    lastelem10 = rowlist[index-10]
    lastelemtime10 = lastelemtime10[:10]
    if checkid == lastelemtime9 and checkid != lastelemtime10:
        listoflists.append(elem)

outname = fileout + '10.csv'
with open(outname, "wb") as newfile:
    writer = csv.writer(newfile)
    writer.writerows(listoflists)
B.10 Calculate Schoener’s Ratio (R)
#Calculate Autocorrelation via self-generated Schoener's Ratio
library(geosphere)
library(rgdal)

#Get a list of all the files in the directory to be looped over
files <-
list.files(path="C:/Users/jillianrutherford/Documents/Masters/MovebankFiles/WL", pattern="*.csv", full.names=TRUE, recursive=FALSE)

#Create a CSV files to add Schoener's Ratio values to
file.create("shoeners.csv")

#Build function "schoener"
schoener <- function(x, y, sd) {
  #Calculate t2 (mean squared distance between successive observations)
t2 <- mean(sd)

  #Calculate r2 (mean squared distance from centre of activity) ***requires UTM***
  mx <- mean(x)
  my <- mean(y)
r2 <- mean((x-mx)^2 + (y-my)^2)

  ratio <- t2/r2
  return(ratio)
}

#Loop through files in directory
for(i in files) {
  #Get filename
  print(i)
  filename <- substring(i,63)
  filename <- substring(filename,1, nchar(filename)-4)

  #Import and format data
  data <- read.csv(i, header=TRUE)
  relocations <- data.frame(data$location.long, data$location.lat)

  #Create a list of step lengths (for calculating t2 later)
  relocations$steplength <- c(0, distCosine(head(relocations,-1),
tail(relocations,-1)))
  relocations <- relocations[complete.cases(relocations),] #remove NaNs
  relocations$squared_dist <- c((relocations$steplength)^2)
  squared_dist <- as.vector(relocations[,4])

  #Transform original data into UTM (for calculating r2 later)
  relolonglat <- SpatialPoints(cbind(relocations$data.location.long,
  relocations$data.location.lat), proj4string=CRS("+init=epsg:32636")) #define original projection
  reloUTM <- spTransform(relolonglat, CRS("+init=epsg:32636"))
  relocations.p <- data.frame(reloUTM$coords.x1, reloUTM$coords.x2)

  x <- relocations.p[,1]
  y <- relocations.p[,2]
  sd <- squared_dist

  #Calculate Schoener's Ratio and add to csv file
  SR <- schoener(x,y,sd)
  write.table(cbind(filename, SR), file="shoeners.csv", append=TRUE,
  col.names=FALSE, sep="", )
}
B.11 Assess Range Residency (R)

#Specify input file and filenames for final plots
details <- "2017WL"
filename <- paste("C:/Users/jillianrutherford/Documents/Masters/MovebankFiles/WL/MF_", details, ".csv", sep="")
msdfile <- paste("MSD_", details,".jpg", sep="")
lifile <- paste("LI_", details,".jpg", sep="")
variofilm <- paste("VAR_", details,".jpg", sep="")

#PART 1: RANGE RESIDENCY VIA RANDOM WALK
###############################################
#Load libraries
library(geosphere)
library(rgdal)
library(raster)
library(progress)

#Import data
data <- read.csv(filename, header=TRUE)
relocations <- data.frame(data$location.long, data$location.lat)

#Step 1: Generate random walk trajectories
#Create a list of step lengths
relocations$steplength <- c(0, distCosine(head(relocations, -1), tail(relocations, -1)))
relocations <- relocations[complete.cases(relocations),] #remove NaNs
steplist <- as.vector(relocations[,3])

#Build function "movement" that creates new steps
#"xy" = starting point, "step" = step length, "heading" = turning angle in radians
movement <- function(xy, step, heading) {
  #First, define pi
  pi = 3.141593

  #Split the starting point into x_init and y_init
  x_init <- xy[1,1]
y_init <- xy[1,2]

  #Translate negative pi values into positive values (headings range from 0 to 2pi)
  if (heading < 0) {
    heading <- abs(heading) + pi
  }

  #Solve for change in x and y given step length and heading (using trigonometry)
y_change <- sin(heading)*step
y_new <- y_init + y_change

  x_change <- cos(heading)*step
  x_new <- x_init + x_change

  #Create a small data frame to save new coordinates of a single manufactured step
move.temp <- as.data.frame(matrix(0,1,4))
move.temp[1,1] <- x_new
move.temp[1,2] <- y_new
move.temp[1,3] <- step
move.temp[1,4] <- heading

  return(move.temp)
#Build function "multi.move" that builds a trajectory N times

```r
pb <- progress_bar$new(format = " downloading [:bar] :percent eta: :eta", total = 100, clear=FALSE, width = 102)
multi.move <- function(N) {
  all.paths <- list()
  pb$tick(0)
  Sys.sleep(3)

  #Repeat the enclosed N times
  for (j in 1:N) {
    #Implement a progress bar
    pb$tick()
    Sys.sleep(1 / 100)

    #Create dataframe to store each trajectory
    steps.df <- data.frame(matrix(0, nrow(relocations), 4))
    colnames(steps.df) <- c("x", "y", "step.length", "turn.angle")

    #Build trajectory
    for (i in 2:nrow(relocations)) {
      step <- sample(steplist, 1, replace = FALSE)
      heading <- runif(n=1, min=-pi, max=pi)
      next.pt <- movement(steps.df[(i-1), 1:2], step, heading)
      steps.df[i,] <- next.pt
    }

    #Add each trajectory to a growing list
    all.paths[[j]] <- steps.df
  }

  return(all.paths)
}
```

#Supply the trajectory-building function with number of reps
multi.paths <- multi.move(100)

#Visualize the results - NOT NECESSARY FOR ALL RUNS BUT NICE OPTION (EACH BELOW REPRESENTS ONE REPEAT)
#plot(multi.paths[[1]]$x, multi.paths[[1]]$y, pch=19, type='b', col='red', ylim=(-20000, 20000), xlim=(-20000,20000))
#points(multi.paths[[2]]$x, multi.paths[[2]]$y, pch=19, type='b', col='blue')
#points(multi.paths[[3]]$x, multi.paths[[3]]$y, pch=19, type='b', col='green')

#Step 2: Calculate MSD and LI
#Transform original data into UTM
relolonglat <- SpatialPoints(cbind(relocations$data.location.long, relocations$data.location.lat), proj4string=CRS("+proj=longlat"))
#define original projection
reloUTM <- spTransform(relolonglat, CRS("+init=epsg:32636"))
orig_traj <- data.frame(reloUTM$coords.x1, reloUTM$coords.x2)

#MSD
#Build MSD function
MSD <- function(x, y) {
  mx <- mean(x)
  my <- mean(y)
  msd <- mean((x-mx)^2 + (y-my)^2)
  return(msd)
}
# Calculate MSD for original trajectory
input_orig <- orig_traj
xcoords_orig <- input_orig[,1]
ycoords_orig <- input_orig[,2]
msd_orig <- MSD(xcoords_orig, ycoords_orig)

# Calculate MSD for each random walk trajectory
MSD_list <- list()
for (i in 1:length(multi.paths)) {
  input_rw <- multi.paths[[i]]
  xcoords_rw <- input_rw[,1]
ycoords_rw <- input_rw[,2]
  msd_rw <- MSD(xcoords_rw, ycoords_rw)
  MSD_list[[i]] <- msd_rw
}

# Plot all MSDs
MSDvalues <- unlist(MSD_list)
maxMSD <- max(MSDvalues)
quartermaxMSD <- maxMSD/4
hist(MSDvalues, breaks=30, main="Range Residency - MSD", xlab="Mean Squared Distance from Centre of Activity", xlim = c(0,maxMSD+quartermaxMSD))
abline(v=msd_orig, col="red", lwd=2)

# Calculate and plot CI for MSD
alpha <- 0.05
msdCI <- quantile(MSDvalues, probs=c(alpha/2, 1-alpha/2))
abline(v=msdCI[1], col="red", lwd=1, lty=2)
abline(v=msdCI[2], col="red", lwd=1, lty=2)

# LI
# Build LI function
LI <- function(x,y) {
  straight.distance <- pointDistance(x, y, type='Euclidean', lonlat = FALSE)
total.distance <- sum(steplist)
  linearity_index <- straight.distance/total.distance
  return(linearity_index)
}

# Calculate LI for original trajectory
x1_orig <- head(orig_traj$reloUTM.coords.x1,1)
y1_orig <- head(orig_traj$reloUTM.coords.x2,1)
x2_orig <- tail(orig_traj$reloUTM.coords.x1,1)
y2_orig <- tail(orig_traj$reloUTM.coords.x2,1)
p1_orig <- c(x1_orig, y1_orig)
p2_orig <- c(x2_orig, y2_orig)
li_orig <- LI(p1_orig, p2_orig)

# Calculate MSD for each random walk trajectory
LI_list <- list()
for (i in 1:length(multi.paths)) {
  input_rw <- multi.paths[[i]]
  x1_rw <- head(input_rw$x,1)
y1_rw <- head(input_rw$y,1)
x2_rw <- tail(input_rw$x,1)
y2_rw <- tail(input_rw$y,1)
p1_rw <- c(x1_rw, y1_rw)
p2_rw <- c(x2_rw, y2_rw)
  li_rw <- LI(p1_rw, p2_rw)
  LI_list[[i]] <- li_rw
}
#Plot all LIs
LIvalues <- unlist(LI_list)
maxLI <- max(LIvalues)
quartemaxLI <- maxLI/4
hist(LIvalues, breaks=30, main="Range Residency - LI", xlab = "Linearity Index", xlim = c(0,maxLI+quartemaxLI))
abline(v=li_orig, col="red", lwd=2)

#Calculate and plot CI for LI
alpha <- 0.05
liCI <- quantile(LIvalues, probs=c(alpha/2, 1-alpha/2))
abline(v=liCI[1], col="red", lwd=1, lty=2)
abline(v=liCI[2], col="red", lwd=1, lty=2)

#Export MSD and LI plots to folder
jpeg(file = msdfile)
hist(MSDvalues, breaks=30, main=paste("Range Residency - MSD - ", details, sep=""), xlab = "Mean Squared Distance from Centre of Activity", xlim = c(0,maxMSD+quartemaxMSD))
abline(v=msd_orig, col="red", lwd=2)
alpha <- 0.05
msdCI <- quantile(MSDvalues, probs=c(alpha/2, 1-alpha/2))
abline(v=msdCI[1], col="red", lwd=1, lty=2)
abline(v=msdCI[2], col="red", lwd=1, lty=2)
dev.off()

jpeg(file = lifile)
hist(LIvalues, breaks=30, main=paste("Range Residency - LI - ", details, sep=""), xlab = "Linearity Index", xlim = c(0,maxLI+quartemaxLI))
abline(v=li_orig, col="red", lwd=2)
alpha <- 0.05
liCI <- quantile(LIvalues, probs=c(alpha/2, 1-alpha/2))
abline(v=liCI[1], col="red", lwd=1, lty=2)
abline(v=liCI[2], col="red", lwd=1, lty=2)
dev.off()

#PART 2: RANGE RESIDENCY VIA VARIOGRAM
#################################################################
#Import data as telemetry object
tel_obj <- as.telemetry(filename, projection="+proj=utm +zone=36 +ellps=WGS84 +datum=WGS84 +units=m +no_defs")

#Visualize XY scatterplot
plot(tel_obj, units=FALSE, col="blue")
title("XY Scatterplot of Relocations")

#Create variogram
variogram <- variogram(tel_obj, fast=TRUE, dt=c(1*60, 6*3600))

#Visualize variogram
plot(variogram, fraction=0.65)
title(paste("Range Resident Test - Varioigram - ", details, sep=""))

#Export variogram plots to folder
jpeg(file = variogramfile)
plot(variogram, fraction=0.65)
title(paste("Range Resident Test - Varioigram - ", details, sep=""))
dev.off()
B.12 Estimate HR via Minimum Convex Polygon (R)

#Install and load the necessary packages
library("adehabitatHR")
library("rgdal")
library("sp")

setwd("C:/Users/jillianrutherford/Documents/Masters/ThesisResults/MCP")

#Get a list of all the files in the directory to be looped over
files <- list.files(path="C:/Users/jillianrutherford/Documents/Masters/MovebankFiles/DL", pattern="*.csv", full.names=TRUE, recursive=FALSE)

#Create a CSV files to add MCP95 and MCP100 area values to respectively
#file.create("MCP95area_TT.csv")
#file.create("MCP100area_TT.csv")

#Apply functions below to all files
for(i in files) {
  print(i)
  filename <- substring(i,63)
  filename <- substring(filename,1, nchar(filename)-4)
  #Import the CSV
  tracks <- read.csv(i, header=TRUE)
  #Convert the CSV to a SpatialPointsDataFrame, and convert latlong coordinates to UTM
  xy <- data.frame(x=tracks$location.long, y=tracks$location.lat)
  coordinates(xy) <- c("x", "y")
  proj4string(xy) <- CRS("+proj=longlat +datum=WGS84")
  tracks_projected <- spTransform(xy, CRS(paste("+proj=utm +zone=36", ellps=WGS84", sep='"')))

  #Apply MCP95 estimator and save as a shapefile
  output95 <- mcp(tracks_projected, percent=95, unin="m", unout = "km2")
  writeOGR(output95, ".", paste(filename,"_MCP95", sep=""), driver = "ESRI Shapefile")

  #Apply MCP100 estimator and save as a shapefile
  output100 <- mcp(tracks_projected, percent=100, unin="m", unout = "km2")
  writeOGR(output100, ".", paste(filename,"_MCP100", sep=""), driver = "ESRI Shapefile")

  #Calculate the area of both MCP95 and MCP100, and append each to designated CSV file
  mcparea <- mcp.area(tracks_projected, percent= seq(95,100, by=5), unin="m", unout = "km2", plotit=FALSE)
  mcp95area <- mcparea$a[[1]]
  write.table(cbind(filename, "MCP95", mcp95area), file="MCP95area_DL.csv", append=TRUE, col.names=FALSE, sep='",')
  mcp100area <- mcparea$a[[2]]
  write.table(cbind(filename, "MCP100", mcp100area), file="MCP100area_DL.csv", append=TRUE, col.names=FALSE, sep='",')
}
# Install and load packages
library(sp)
library(rgdal)
library(raster)
library(adehabitatMA)

# Set working directory
setwd("C:/Users/jillianrutherford/Documents/Masters/ThesisResults/GSS")

# Get a list of all the files in the directory to be looped over
files <- list.files(path="C:/Users/jillianrutherford/Documents/Masters/MovebankFiles/WL", pattern="*.csv", full.names=TRUE, recursive=FALSE)

# To create the grid
# Import CSV of all 2017 locations for all chimps to get extent
tracks <- read.csv("C:/Users/jillianrutherford/Documents/Masters/RelocationExtracts/RX_2017_TOTAL.csv", header=TRUE)

# Convert the tracks CSV to a SpatialPointsDataFrame, and convert latlong coordinates to UTM
coords <- data.frame(x=tracks$longitude, y=tracks$latitude)
crs <- "+proj=utm +zone=36 +ellps=WGS84 +datum=WGS84"
tracks.spdf <- SpatialPointsDataFrame(coords=coords, data=tracks, proj4string=CRS(crs))
bbox <- bbox(tracks.spdf)
print(bbox)

# First create points for each grid cell centre, using the extent provided by the bbox function above (self-tweaked for 2017 specifically)
x <- seq(from=203000.00, to=209500.00, by=100) # the "by" parameter is the spacing between each point, equivalent to cell size
y <- seq(from=56000.00, to=65800.00, by=100)

# Second, use the coordinates of each cell centre (created above) to make a gridded object
xy <- expand.grid(x=x, y=y) # this just looks like a bunch of points across the field
grid.pts <- SpatialPointsDataFrame(coords=xy, data=xy, proj4string = CRS(crs))
gridded(grid.pts) <- TRUE # this looks closer but is not quite right still

# Third, make the grid into a SpatialPolygonsDataFrame
grid <- as(grid.pts, "SpatialPolygons")
grid.spdf <- SpatialPolygonsDataFrame(grid, data=data.frame(id=row.names(names(grid)), row.names=row.names(grid)))
plot(grid) # this now looks like a grid across the field

# Give each grid cell a label
names.grid <- sapply(grid.spdf@polygons, function(x) slot(x,"ID"))
text(coordinates(grid.spdf), labels=sapply(slot(grid.spdf, "polygons"), function(i) slot(i,"ID")), cex=0.3)

# Show the relocations used to make the grid, over the grid, to ensure it covers everything
points(tracks.spdf, col="red")
# TO CALCULATE GSS AREA
# Apply functions below to all files
for(i in files)
{
    print(i)
    filename <- substring(i, 63)
    filename <- substring(filename, 1, nchar(filename) - 4)

    # Import the CSV
    relos <- read.csv(i, header=TRUE)

    # Convert the CSV to a SpatialPointsDataFrame, and convert latlong coordinates to UTM
    xy2 <- data.frame(x=relos$location.long, y=relos$location.lat)
    coordinates(xy2) <- c("x", "y")
    proj4string(xy2) <- CRS("+proj=longlat +datum=WGS84")
    relos_projected <- spTransform(xy2, CRS(paste("+proj=utm +zone=36", ellps="WGS84", sep='')))
    relos_projected <- as.data.frame(relos_projected)
    relos.spdf <- SpatialPointsDataFrame(coords=relos_projected, data=relos_projected, proj4string=CRS(crs))

    # Check that the relos line up with the grid
    plot(grid.spdf)
    plot(relos.spdf, add=TRUE)

    # Identify which relocations fall within which grid cell by adding a grid ID field to the tracks.spdf
    o = over(relos.spdf, grid.spdf)
    relos.spdf = cbind(relos.spdf@data, o)
    head(relos.spdf)

    # Calculate number of grid cells used (number of unique values in recently added grid ID column)
    unique <- unique(relos.spdf$id)
    numcells <- length(unique)

    # Calculate size of home range and add to created csv file
    areakm <- numcells*(100*100)/1000000
    write.table(cbind(filename, "GSS100", areakm), file="GSS100area_WL.csv", append=TRUE, col.names=FALSE, sep="",

    # Extract subset of polygons from grid.spdf, and save as shapefile
    unique <- sapply(unique, as.character)
    occupiedcells <- grid.spdf[unique,]
    plot(occupiedcells)
    writeOGR(occupiedcells, ".", paste(filename,"_GSS100", sep=""), driver = "ESRI Shapefile")
}
}
B.14 Estimate HR via Grid Square Sum (200 m) (R)

#Install and load packages
library(sp)
library(rgdal)
library(raster)
library(adehabitatMA)

#Set working directory
setwd("C:/Users/jillianrutherford/Documents/Masters/ThesisResults/GSSShapefiles")

#Get a list of all the files in the directory to be looped over
files <- list.files(path="C:/Users/jillianrutherford/Documents/Masters/MovebankFiles/WL", pattern="*.csv", full.names=TRUE, recursive=FALSE)

#TO CREATE THE GRID
#Import CSV of all 2017 locations for all chimps to get extent
tracks <- read.csv("C:/Users/jillianrutherford/Documents/Masters/RelocationExtracts/RX_2017_TOTAL.csv", header=TRUE)

#Convert the tracks CSV to a SpatialPointsDataFrame, and convert latlong coordinates to UTM
coords <- data.frame(x=tracks$longitude, y=tracks$latitude)
crs <- "+proj=utm +zone=36 +ellps=WGS84 +datum=WGS84"
tracks.spdf <- SpatialPointsDataFrame(coords=coords, data=tracks, proj4string=CRS(crs))
bbox <- bbox(tracks.spdf)
print(bbox)

#First create points for each grid cell centre, using the extent provided by the bbox function above (self-tweaked for 2017 specifically)
x <- seq(from=203000.00, to=209500.00, by=200) #the "by" parameter is the spacing between each point, equivalent to cell size
y <- seq(from=56000.00, to=65800.00, by=200)

#Second, use the coordinates of each cell centre (created above) to make a gridded object
xy <- expand.grid(x=x, y=y) #this just looks like a bunch of points across the field
grid(pts) <- SpatialPointsDataFrame(coords=xy, data=xy, proj4string = CRS(crs))
gridded(grid(pts)) <- TRUE #this looks closer but is not quite right still

#Third, make the grid into a SpatialPolygonsDataFrame
grid <- as(grid(pts), "SpatialPolygons")
grid.spdf <- SpatialPolygonsDataFrame(grid, data=data.frame(id=row.names(grid), row.names=row.names(grid)))
plot(grid) #this now looks like a grid across the field

#Give each grid cell a label
names(grid) <- sapply(grid.spdf@polygons, function(x) slot(x, "ID"))
text(coordinates(grid.spdf), labels=sapply(slot(grid.spdf, "polygons"), function(i) slot(i, "ID")), cex=0.3)

#Show the relocations used to make the grid, over the grid, to ensure it covers everything
points(tracks.spdf, col="red")
#TO CALCULATE GSS AREA
#Apply functions below to all files

for(i in files)
{
  print(i)
  filename <- substring(i, 63)
  filename <- substring(filename, 1, nchar(filename) - 4)

  #Import the CSV
  relos <- read.csv(i, header=TRUE)

  #Convert the CSV to a SpatialPointsDataFrame, and convert latlong coordinates to UTM
  xy2 <- data.frame(x=relos$location.long, y=relos$location.lat)
  coordinates(xy2) <- c("x", "y")
  proj4string(xy2) <- CRS("+proj=longlat +datum=WGS84")
  relos_projected <- spTransform(xy2, CRS(paste("+proj=utm +zone=36", "
  ellps=WGS84", sep=''))) # +ellps=WGS84"))
  relos_projected <- as.data.frame(relos_projected)
  relos.spdf <- SpatialPointsDataFrame(coords=relos_projected, data=relos_projected, proj4string=CRS(crs))

  #Check that the relos line up with the grid
  plot(grid.spdf)
  plot(relos.spdf, add=TRUE)

  #Identify which relocations fall within which grid cell by adding a grid ID field to the tracks.spdf
  o = over(relos.spdf, grid.spdf)
  relos.spdf = cbind(relos.spdf@data, o)
  head(relos.spdf)

  #Calculate number of grid cells used (number of unique values in recently added grid ID column)
  unique <- unique(relos.spdf$id)
  numcells <- length(unique)

  #Calculate size of home range and add to created csv file
  areakm <- numcells*200*200)/1000000
  write.table(cbind(filename, "GSS200", areakm), file="GSS200area_WL.csv", append=TRUE, col.names=FALSE, sep="","

  #Extract subset of polygons from grid.spdf, and save as shapefile
  unique <- sapply(unique, as.character)
  occupiedcells <- grid.spdf[unique,]
  plot(occupiedcells)
  writeOGR(occupiedcells, ".", paste(filename,"_GSS200", sep=""), driver = "ESRI Shapefile")
}

B.15 Estimate HR via Grid Square Sum (500 m) (R)

# Install and load packages
library(sp)
library(rgdal)
library(raster)
library(adehabitatMA)

# Set working directory
setwd("C:/Users/jillianrutherford/Documents/Masters/ThesisResults/GSSShapefiles")

# Get a list of all the files in the directory to be looped over
files <- list.files(path = "C:/Users/jillianrutherford/Documents/Masters/MovebankFiles/WL", pattern = "*.csv", full.names=TRUE, recursive=FALSE)

# TO CREATE THE GRID
# Import CSV of all 2017 locations for all chimps to get extent
tracks <- read.csv("C:/Users/jillianrutherford/Documents/Masters/RelocationExtracts/RX_2017_TOTAL.csv", header=TRUE)

# Convert the tracks CSV to a SpatialPointsDataFrame, and convert latlong coordinates to UTM
cords <- data.frame(x=tracks$longitude, y=tracks$latitude)
crs <- "+proj=utm +zone=36 +ellps=WGS84 +datum=WGS84"
tracks.spdf <- SpatialPointsDataFrame(coords=cords, data=tracks, proj4string=CRS(crs))
bbox <- bbox(tracks.spdf)
print(bbox)

# First create points for each grid cell centre, using the extent provided by the bbox function above (self-tweaked for 2017 specifically)
x <- seq(from=203000.00, to=209500.00, by=500) # the "by" parameter is the spacing between each point, equivalent to cell size
y <- seq(from=56000.00, to=65800.00, by=500)

# Second, use the coordinates of each cell centre (created above) to make a gridded object
xy <- expand.grid(x=x, y=y) # this just looks like a bunch of points across the field
grid.pts <- SpatialPointsDataFrame(coords=xy, data=xy, proj4string = CRS(crs))
gridded(grid.pts) <- TRUE # this looks closer but is not quite right still

# Third, make the grid into a SpatialPolygonsDataFrame
grid <- as(grid.pts, "SpatialPolygons")
grid.spdf <- SpatialPolygonsDataFrame(grid, data=data.frame(id=row.names(grid), row.names=row.names(grid)))
plot(grid) # this now looks like a grid across the field

# Give each grid cell a label
names.grid <- sapply(grid.spdf@polygons, function(x) slot(x,"ID"))
text(coordinates(grid.spdf), labels=sapply(slot(grid.spdf, "polygons"), function(i) slot(i,"ID")), cex=0.3)

# Show the relocations used to make the grid, over the grid, to ensure it covers everything
points(tracks.spdf, col="red")

# TO CALCULATE GSS AREA
# Apply functions below to all files
for(i in files)
{
  print(i)
  filename <- substring(i,63)
  filename <- substring(filename,1, nchar(filename)-4)

  #Import the CSV
  relos <- read.csv(i, header=TRUE)

  #Convert the CSV to a SpatialPointsDataFrame, and convert latlong coordinates to UTM
  xy2 <- data.frame(x=relos$location.long, y=relos$location.lat)
  coordinates(xy2) <- c("x", "y")
  proj4string(xy2) <- CRS("+proj=longlat +datum=WGS84")
  relos_projected <- spTransform(xy2, CRS(paste("+proj=utm +zone=36",
  ellps=WGS84",sep='')))
  relos_projected <- as.data.frame(relos_projected)
  relos.spdf <- SpatialPointsDataFrame(coords=relos_projected,
  data=relos_projected, proj4string=CRS(crs))
  #Check that the relos line up with the grid
  plot(grid.spdf)
  plot(relos.spdf, add=TRUE)

  #Identify which relocations fall within which grid cell by adding a grid ID field to the tracks.spdf
  o = over(relos.spdf, grid.spdf)
  relos.spdf = cbind(relos.spdf@data, o)
  head(relos.spdf)

  #Calculate number of grid cells used (number of unique values in recently added grid ID column)
  unique <- unique(relos.spdf$id)
  numcells <- length(unique)

  #Calculate size of home range and add to created csv file
  areakm <- numcells*(500*500)/1000000
  write.table(cbind(filename, "GSS500", areakm), file="GSS500area_WL.csv", append=TRUE, col.names=FALSE, sep="",)

  #Extract subset of polygons from grid.spdf, and save as shapefile
  unique <- sapply(unique, as.character)
  occupiedcells <- grid.spdf[unique,]
  plot(occupiedcells)
  writeOGR(occupiedcells, ".", paste(filename,"_GSS500", sep=""), driver = "ESRI Shapefile")
}
B.16 Estimate HR via Kernel Density Estimate (href) (R)

library(sp)
library(rgdal)
library(raster)
library(adefhabitatHR)
library(maptools)

setwd("C:/Users/jillianrutherford/Documents/Masters/ThesisResults/KDE/KDEShapefiles")

#Get a list of all the files in the directory to be looped over
files <- list.files(path="C:/Users/jillianrutherford/Documents/Masters/MovebankFiles/WL", pattern="*.csv", full.names=TRUE, recursive=FALSE)

#Identify correct crs for later
crs <- "+proj=utm +zone=36 +ellps=WGS84 +datum=WGS84"

#TO CALCULATE KDE95href AREA
for(i in files) {
  print(i)
  filename <- substring(i,63)
  filename <- substring(filename,1, nchar(filename)-4)

  #Import the CSV and convert to a SpatialPointsDataFrame
  relos <- read.csv(i, header=TRUE)
  xy2 <- data.frame(x=relos$location.long, y=relos$location.lat)
  coordinates(xy2) <- c("x", "y")
  proj4string(xy2) <- CRS("+proj=longlat +datum=WGS84")
  relos_projected <- spTransform(xy2, CRS(paste("+proj=utm +zone=36",
    ellps=WGS84", sep=""))
  #relos_projected <- as.data.frame(relos_projected)

  #Create a grid to supply to the kernel function for display
  x <- seq(from=202000.00, to=211000.00, by=100)  #can change this to be smaller
  for a smoother looking output, eg 100 -> 10
  y <- seq(from=56000.00, to=67800.00, by=100)
  grid <- expand.grid(x=x, y=y)
  coordinates(grid) <- ~x+y
  gridded(grid) <- TRUE
  plot(grid)

  #Perform KDE with bandwidth determined by href
  kernel.href <- kernelUD(relos_projected, h="href", grid=grid)
  #plot the output
  kernel.href[[1]]@h  #identify what href calculated as bandwidth
  ver <- getverticeshr(kernel.href, percent=95, unin="m", unout="km2")
  plot(ver, add=TRUE)

  #Calculate size of home range and add to created csv file
  areakm <- ver$area
  write.table(cbind(filename, "KDE95href", areakm), file="KDE95hrefarea_WL.csv", append=TRUE, col.names=FALSE, sep="",
    row.names=FALSE)

  #Export KDE href outline to shapefile
  writeOGR(ver, ".", paste(filename,"_KDE95href", sep=""), driver = "ESRI Shapefile")
}
B.17 Estimate HR via Kernel Density Estimate (hadhoc) (R)

#Load packages
library(sp)
library(rgdal)
library(raster)
library(adehabitatHR)
library(maptools)

setwd("C:/Users/jillianrutherford/Documents/Masters/ThesisResults/KDE/KDEShapefiles")

#Create a grid to supply to the kernel function for display
x <- seq(from=202000.00, to=211000.00, by=100) #can change this to be smaller for a smoother looking output, eg 100 -> 10
y <- seq(from=56000.00, to=67800.00, by=100)
grid <- expand.grid(x=x, y=y)
coordinates(grid) <- ~x+y
grided(grid) <- TRUE
plot(grid)

###START HERE###

#Import data and convert to a SpatialPointsDataFrame
file <- "MF_2017WL_SD6m_SI24h_10.csv"
filename <- substring(file,1, nchar(file)-4)
relos <- read.csv(paste("C:/Users/jillianrutherford/Documents/Masters/MovebankFiles/WL/", file, sep=""), header=TRUE, stringsAsFactors=FALSE)
relos.spdf <- relos[, c("individual.local.identifier", "location.long", "location.lat")]
coordinates(relos.spdf) <- c("location.long", "location.lat")
proj4string(relos.spdf) <- CRS("+proj=longlat +datum=WGS84")
relos.spdf <- spTransform(relos.spdf, CRS(paste("+proj=utm +zone=36", ellps=WGS84", sep='"))

#Identify the value of original href
kernel.href <- kernelUD(relos.spdf, h="href", grid=grid)
image(kernel.href) #plot the output
href <- kernel.href[[1]]@h #identify what href calculated as bandwidth
orig <- href[[1]]

#Create modified hrefs
href_200 <- (2.0*orig)
href_190 <- (1.9*orig)
href_180 <- (1.8*orig)
href_170 <- (1.7*orig)
href_160 <- (1.6*orig)
href_150 <- (1.5*orig)
href_140 <- (1.4*orig)
href_130 <- (1.3*orig)
href_120 <- (1.2*orig)
href_110 <- (1.1*orig)
href_100 <- orig
href_90 <- (0.9*orig)
href_80 <- (0.8*orig)
href_70 <- (0.7*orig)
href_60 <- (0.6*orig)
href_50 <- (0.5*orig)

#Produce KDE for reduced hrefs and plot
#200% href
kernel.href200 <- kernelUD(relos.spdf, h[href=200, grid=grid])
kernel.href200[[1]]@h  #identify what href calculated as bandwidth
ver200 <- getverticeshr(kernel.href200, percent=95, unin="m", unout="km2")
plot(ver200)
title(main="href*2.0")

#190% href
kernel.href190 <- kernelUD(relos.spdf, h[href=190, grid=grid])
kernel.href190[[1]]@h  #identify what href calculated as bandwidth
ver190 <- getverticeshr(kernel.href190, percent=95, unin="m", unout="km2")
plot(ver190)
title(main="href*1.9")

#180% href
kernel.href180 <- kernelUD(relos.spdf, h[href=180, grid=grid])
kernel.href180[[1]]@h  #identify what href calculated as bandwidth
ver180 <- getverticeshr(kernel.href180, percent=95, unin="m", unout="km2")
plot(ver180)
title(main="href*1.8")

#170% href
kernel.href170 <- kernelUD(relos.spdf, h[href=170, grid=grid])
kernel.href170[[1]]@h  #identify what href calculated as bandwidth
ver170 <- getverticeshr(kernel.href170, percent=95, unin="m", unout="km2")
plot(ver170)
title(main="href*1.7")

#160% href
kernel.href160 <- kernelUD(relos.spdf, h[href=160, grid=grid])
kernel.href160[[1]]@h  #identify what href calculated as bandwidth
ver160 <- getverticeshr(kernel.href160, percent=95, unin="m", unout="km2")
plot(ver160)
title(main="href*1.6")

#150% href
kernel.href150 <- kernelUD(relos.spdf, h[href=150, grid=grid])
kernel.href150[[1]]@h  #identify what href calculated as bandwidth
ver150 <- getverticeshr(kernel.href150, percent=95, unin="m", unout="km2")
plot(ver150)
title(main="href*1.5")

#140% href
kernel.href140 <- kernelUD(relos.spdf, h[href=140, grid=grid])
kernel.href140[[1]]@h  #identify what href calculated as bandwidth
ver140 <- getverticeshr(kernel.href140, percent=95, unin="m", unout="km2")
plot(ver140)
title(main="href*1.4")

#130% href
kernel.href130 <- kernelUD(relos.spdf, h[href=130, grid=grid])
kernel.href130[[1]]@h  #identify what href calculated as bandwidth
ver130 <- getverticeshr(kernel.href130, percent=95, unin="m", unout="km2")
plot(ver130)
title(main="href*1.3")

#120% href
kernel.href120 <- kernelUD(relos.spdf, h[href=120, grid=grid])
kernel.href120[[1]]@h  #identify what href calculated as bandwidth
ver120 <- getverticeshr(kernel.href120, percent=95, unin="m", unout="km2")
plot(ver120)
title(main="href*1.2")
#110% href
kernel.href110 <- kernelUD(relos.spdf, h[href_110, grid=grid])
kernel.href110[[1]]@h  #identify what href calculated as bandwidth
ver110 <- getverticeshr(kernel.href110, percent=95, unin="m", unout="km2")
plot(ver110)
title(main="href*1.1")

#100% href
kernel.href100 <- kernelUD(relos.spdf, h[href_100, grid=grid])
kernel.href100[[1]]@h  #identify what href calculated as bandwidth
ver100 <- getverticeshr(kernel.href100, percent=95, unin="m", unout="km2")
plot(ver100)
title(main="href*1.0")

#90% href
kernel.href90 <- kernelUD(relos.spdf, h[href_90, grid=grid])
kernel.href90[[1]]@h  #identify what href calculated as bandwidth
ver90 <- getverticeshr(kernel.href90, percent=95, unin="m", unout="km2")
plot(ver90)
title(main="href*0.9")

#80% href
kernel.href80 <- kernelUD(relos.spdf, h[href_80, grid=grid])
kernel.href80[[1]]@h  #identify what href calculated as bandwidth
ver80 <- getverticeshr(kernel.href80, percent=95, unin="m", unout="km2")
plot(ver80)
title(main="href*0.8")

#70% href
kernel.href70 <- kernelUD(relos.spdf, h[href_70, grid=grid])
kernel.href70[[1]]@h  #identify what href calculated as bandwidth
ver70 <- getverticeshr(kernel.href70, percent=95, unin="m", unout="km2")
plot(ver70)
title(main="href*0.7")

#60% href
kernel.href60 <- kernelUD(relos.spdf, h[href_60, grid=grid])
kernel.href60[[1]]@h  #identify what href calculated as bandwidth
ver60 <- getverticeshr(kernel.href60, percent=95, unin="m", unout="km2")
plot(ver60)
title(main="href*0.6")

#50% href
kernel.href50 <- kernelUD(relos.spdf, h[href_50, grid=grid])
kernel.href50[[1]]@h  #identify what href calculated as bandwidth
ver50 <- getverticeshr(kernel.href50, percent=95, unin="m", unout="km2")
plot(ver50)
title(main="href*0.5")

#IDENTIFY THE verX WINNER
winner <- ver60

#Calculate size of home range and add to created csv file
areakm <- winner$area
write.table(cbind(filename, "KDE95hadhoc", areakm), file="KDE95hadhocarea_WL.csv", append=TRUE, col.names=FALSE, sep="",)

#Export KDE href outline to shapefile
writeOGR(winner, ".", paste(filename, "_KDE95hadhoc", sep=""), driver = "ESRI Shapefile")
B.18 Estimate HR via Kernel Density Estimate (hlscv) (R)

```r
#Load packages
library(sp)
library(rgdal)
library(raster)
library(adehabitatHR)
library(maptools)

#Set working directory
setwd("C:/Users/jillianrutherford/Documents/Masters/ThesisResults/KDE")

#Get a list of all the files in the directory to be looped over
files <- list.files(path="C:/Users/jillianrutherford/Documents/Masters/MovebankFiles/temp", pattern="*.csv", full.names=TRUE, recursive=FALSE)

#Create a CSV file to add KDE95href area values to
file.create("KDE95hlscv_convergence.csv")

#Create a grid to supply to the kernel function for display
x <- seq(from=203000.00, to=209500.00, by=100) #can change this to be smaller for a smoother looking output, eg 100 -> 10
y <- seq(from=56000.00, to=67800.00, by=100)
grid <- expand.grid(x=x, y=y)
coordinates(grid) <- ~x+y
gridded(grid) <- TRUE
plot(grid)

#Apply functions below to all files
for(i in files)
{
  print(i)
  filename <- substring(i,63)
  filename <- substring(filename,1, nchar(filename)-4)

  #Import data and convert to a SpatialPointsDataFrame
  relos <- read.csv(i, header=TRUE, stringsAsFactors=FALSE)
  relos.spdf <- relos[, c("individual.local.identifier", "location.long", "location.lat")]
  coordinates(relos.spdf) <- c("location.long", "location.lat")
  proj4string(relos.spdf) <- CRS("+proj=longlat +datum=WGS84")
  relos.spdf <- spTransform(relos.spdf, CRS(paste("+proj=utm +zone=36", ellps=WGS84",sep="")))

  #Perform KDE with bandwidth determined by LSCV
  kernel.hlscv <- kernelUD(relos.spdf, h="LSCV", grid=grid)
  image(kernel.hlscv)
  con_stat <- kernel.hlscv[[1]]@h$convergence

  #Add convergence status to csv
  write.table(cbind(filename, "KDE95hlscv", con_stat), file="KDE95hlscv_convergence.csv", append=TRUE, col.names=FALSE, sep="",)
}
```
B.19 Estimate HR via Autocorrelated Kernel Density Estimate (including Select Movement Model and Calculate Effective Sample Size) (R)

```r
#Load libraries
library("ctmm")
library("sp")
library(foreach)
library(doParallel)

#Get a list of all the files in the directory to be looped over
csv_list <- list.files(path="C:/Users/jillianrutherford/Documents/Masters/MovebankFiles/test", pattern="*.csv", full.names=TRUE, recursive=FALSE)

#Set up the parallel
cores=detectCores()
backend <- parallel::makeCluster(cores[1]-1)  #this is assigning the backend
doParallel::registerDoParallel(backend)  #here you’re going to assign the
doParallel to that backend

foreach(i=1:10)%dopar%{
  #Get filename for output
  filename <- substring(csv_list[i], 65)
  filename <- substring(filename,1, nchar(filename)-4)

  #Import data as telemetry object
tel_obj <- ctmm::as.telemetry(csv_list[i], projection="+proj=utm +zone=36 +ellps=WGS84 +datum=WGS84 +units=m +no_defs")

  #Create a variogram fit guess using ctmm.guess
  GUESS <- ctmm::ctmm.guess(tel_obj, interactive=FALSE)

  #Fit the correct movement model
  akde.mods <- ctmm::ctmm.select(tel_obj, CTMM=GUESS, verbose=TRUE, trace=TRUE)
  model <- summary(akde.mods[[1]])$name

  #Apply UNWEIGHTED AKDE
  akdeUD <- ctmm::akde(tel_obj, akde.mods[[1]], res=3)
  area_akdeUD <- summary(akdeUD)$CI[[2]]
  effsamp <- akdeUD$DOF.H
  plot(tel_obj, UD=akdeUD, pch=19, cex=0.25, col="darkorange")
  ctmm::writeShapefile(akdeUD, "C:/Users/jillianrutherford/Documents/Masters/MovebankFiles/output_test", filename)

  #Append effective sample size values and area values to csv
  write.table(cbind(filename, model, effsamp, effsamp_w, area_akdeUD, area_akdeUD_w), file="AKDEresults.csv", append=TRUE, col.names=FALSE, sep=','
}
```
Literature Cited


Hasegawa, T. 1990. “Sex Differences in Ranging Patterns.” In The Chimpanzees of the Mahale Mountains: Sexual and Life History Strategies, University of Tokyo Press.


