From Diffusion to Cognition: Analytical, Statistical and Mechanistic Approaches to the Study of Animal Movement

By
Tal Avgar

A Thesis
Presented to The University of Guelph

In Partial Fulfillment of Requirements for the Degree of Doctor of Philosophy
in
Integrative Biology

Guelph, Ontario, Canada
© Tal Avgar, November, 2012
Abstract

From diffusion to cognition: analytical, statistical and mechanistic approaches to the study of animal movement

Tal Avgar

Advisor:

University of Guelph, 2012

John M. Fryxell

Ecology is the scientific study of processes that determine the distribution and abundance of organisms in space and time. Animal movement plays a crucial role in determining the fates of individuals, populations, communities, and ecosystems. Hence, understanding how and why animals change their spatial location through time is fundamental to ecological research. Animal movement patterns reflect behavioral, physiological and physical interactions between individuals and their environment. Coupling movement and environmental data may thus provide a rich source of information regarding many aspects of animal ecology. In my PhD thesis, I develop and demonstrate different approaches to understanding and predicting animal movement patterns in relation to their environment. In the first chapter, I merge two fundamental ecological models, the functional response and random walk, to formally derive diffusion rates of consumers as function of their handling time and the abundance, distribution and mobility of their resources. This mechanistic null model provides a simple behavior-free explanation to commonly observed negative associations between movement rates and resource abundance, often attributed to area-restricted search behavior. In the second chapter, I use positional data of woodland caribou in Ontario to calculate random walk-based movement expectations for each individual during each month. I then statistically link these expectations to ecologically significant environmental conditions. I show that landscape correlates of forage abundance and habitat permeability explain much of the observed variation in caribou movement characteristics and that residual variability may be attributed to spatial population structure. In the third chapter, I develop a novel state-space approach, enabling simultaneous consideration of resource preference, cognitive capacities and movement limitations, within a simulation model of animal movement across heterogeneous landscapes. The model is designed to enable direct
parameterization based on empirical movement and landscape data. This approach allows one to both theoretically explore the consequences of different cognitive abilities and to predict animal space-use patterns across novel or altered landscapes. Overall, my thesis contributes to the rapidly developing field of movement ecology by formulating mechanistically defendable linkages between animal movement and landscape characteristics.
Acknowledgments

My work was supported by a Natural Sciences and Engineering Research Council Vanier Fellowship.

I would like to thank my advisor, John Fryxell, for his encouragement and support, for his faith in me and my research, and for teaching me how to become a better scientist. I owe thanks to my previous advisor, Ran Nathan, for his contagious enthusiasm for the study of movement ecology and for giving me the best head start a young scientist could get. I would also like to thank my collaborators, Anna Mosser and Oded Berger-Tal, and the members of my advisory committee, Glen Brown, Rob Deardon, Reuven Dukas, Kevin McCann and Ian Thompson, for their help, advice and support.

I thank current and past members of the Fryxell, Nudds and Norris labs for their scientific insights and their friendship. I would especially like to thank my friend and colleague, Daniel Kuefler, for his warm companionship and for countless discussions about science, food and well... life.

Last but not least, I am grateful for my family. I thank my son, Jonathan, for making the past two years more interesting (and challenging) than I could ever imagine. More than all, this work was only made possible thanks to the advice, insights, help and endless love and support from my life-partner, Luba.
# Table of Contents

Abstract ........................................................................................................................................ ii

Acknowledgments ......................................................................................................................... iv

List of Tables ................................................................................................................................ v

List of Figures ............................................................................................................................... vi

Prologue .......................................................................................................................................... 1

References ....................................................................................................................................... 5

Chapter 1: Linking rates of diffusion and consumption in relation to resources ...................... 6

Abstract ........................................................................................................................................ 6

Introduction ..................................................................................................................................... 7

Models ............................................................................................................................................ 8

Results .......................................................................................................................................... 14

Discussion .................................................................................................................................... 16

References ..................................................................................................................................... 19

Figures .......................................................................................................................................... 24

Chapter 2: Environmental and individual drivers of animal movement patterns across a wide geographical gradient ................................................................................................. 28

Abstract ........................................................................................................................................ 28

Introduction ..................................................................................................................................... 29

Materials and methods .................................................................................................................. 32

Results .......................................................................................................................................... 37

Discussion .................................................................................................................................... 39

References ..................................................................................................................................... 44

Tables .......................................................................................................................................... 51
Chapter 3: An empirically parameterized individual based model of animal movement, perception and memory................................................................. 58

Abstract ................................................................. 58

Introduction .................................................................. 59

Methods ....................................................................... 61

Results ........................................................................ 74

Discussion ..................................................................... 76

References .................................................................... 83

Tables ........................................................................... 88

Figures ......................................................................... 90

Epilogue ......................................................................... 99

Supplementary material .................................................. 101

Appendix A (Chapter 1) .................................................. 101

Appendix B (Chapter 1) .................................................. 102

Appendix C (Chapter 1) .................................................. 110

Appendix D (Chapter 1) .................................................. 115

Appendix E (Chapter 2) .................................................. 120

Appendix F (Chapter 3) .................................................. 123

Appendix G (Chapter 3) .................................................. 158
List of Tables

Chapter 2

Table 1 – Caribou movement attributes
Table 2 – Monthly displacements - predictions and results
Table 3 – Biased correlated random walk components

Chapter 3

Table 1 – The model’s parameters
Table 2 – Mean standardized bias in parameter estimations
List of Figures

Chapter 1

Figure 1 – Predictions of the closed form Scenario I model……………………………………24
Figure 2 – Predictions of the closed form Scenario II model……………………………………25
Figure 3 – Simulated consumption and diffusion rates function of resource aggregation__26
Figure 4 – Simulated consumption and diffusion rates as function of resource density___27

Chapter 2

Figure 1 – Observed vs. expected monthly displacements______________________________54
Figure 2 – Seasonal means of observed and expected monthly displacements________55
Figure 3 – Individual effects on expected displacement______________________________56
Figure 4 – Observed vs. predicted monthly displacements for validation____________57

Chapter 3

Figure 1 – A graphical illustration of the translation the redistribution kernel___________90
Figure 2 – A graphical illustration of simulated movement paths___________________92
Figure 3 – A simulated trajectory used for assessing scale-dependent parameterization 93
Figure 4 – Graphical illustrations of prior distributions____________________________94
Figure 5 – Marginal posterior parameter distributions______________________________95
Figure 6 – Parameter estimation bias as function of sampling scales________________96
Figure 7 – Path sinuosity distributions as function of sampling scales____________97
Figure 8 – Forage selection coefficient distributions as function of sampling scales____98
Prologue

Movement patterns emerge from diverse individual behaviors and have ramifications for numerous population and community level processes. The rapid growth in movement ecology research in recent years has been motivated by increased awareness of the conceptual significance of movement processes (Nathan et al. 2008). This scientific proliferation has been accompanied, and in part stimulated, by significant technological advances that enable tracking individuals with high spatial and temporal resolution over an extended period, across remote landscapes, and at an affordable cost (Cagnacci et al. 2010). These technologies lead to the accumulation of massive amounts of data which can then be used to resolve applied challenges (e.g., identifying critical habitat requirements or assessing invasive species spread) and to advance our understanding of basic ecological relationships (e.g., linking movement behavior to life history characteristics or to models of optimal foraging).

The majority of questions studied by movement ecologists, whether applied or basic, involve understanding at least some aspects of the interface between the animal and its surroundings (Schick et al. 2008). From an evolutionary adaptive perspective, animal movement is motivated by the need to survive and reproduce. According to this view, animals should tend to move away from localities with unfavorable conditions, such as intense competition or predation, while tending to move toward localities that offer improved access to food, shelter or mating opportunities. Hence, understanding the processes giving rise to animal movement patterns requires knowledge of landscape variation in fitness-influencing factors and potential constraints on the behavioral response. The latter includes biomechanical and energetic constraints (e.g., fish movement towards an upstream food source may be compromised due to currents and energy requirements) as well as cognitive constraints (e.g., a naive animal does not know the true value of a distant food patch due to sensory or computational limitations). Gaining insights into these motivations and limitations is a major facet in the study of movement ecology (Nathan et al. 2008).

The random walk model, which focuses on the statistical properties of the movement path, is a central pillar in the study of animal movement ecology due to its simplicity, flexibility and straightforward link to Eulerian diffusion (Kareiva & Shigesada 1983; Codling, Plank, &
Benhamou 2008). In its simplest form, a random walk process is governed by a single parameter — determining the shape of the step-length distribution. The flexibility of the random walk model is reflected in its many variants, including, among others, locally- and globally-biased random walks, correlated random walks and composite random walks (Codling et al. 2008). For example, relying on a composite correlated random walk model, Morales et al. (2004) developed an elegant method for likelihood-based assignment of discrete behavioral states to segments of an observed movement path. This general approach was later extensively developed using hidden Markov models and is now an essential part of the movement ecologist’s toolkit. At the population level, many random walk processes are approximated by simple diffusion, offering a crucial link between the behavior of individuals and the spatial pattern of the population. This point was exemplified in a seminal paper by Turchin (1991), where he reintroduced Patlak’s (1953) model for spatially explicit diffusion. Turchin demonstrated, both theoretically and empirically, that the probabilistic movement rules employed by individuals in response to local environmental conditions could be used to derive the population’s distribution across a patchy landscape.

Central to the use of random walk models in the study of animal movement ecology is the link between the characteristics of the random walk and local resource abundance. It has long been recognized that animals tend to move at a slower rate through seemingly profitable areas of the landscape (Kuefler, Avgar, & Fryxell 2012 and references therein). This observation fits well with the notion of movement as an adaptive behavior and can be modeled within a random walk framework as a behavioral adjustment of the step and/or turning distributions in response to recent resource encounter events (Benhamou 2007). Under this approach, behavioral parameters determine the quantitative relationship between movement rate and resource abundance.

In the first chapter of my thesis, I derive a synthetic model of consumer movement and intake rates based on the common logical premise underlying two fundamental ecological models. My model offers a parsimonious alternative to the behavioristic approach, suggesting that the effect of resource abundance on consumer diffusion rates may be derived directly from the consumers’ functional response. I demonstrate how diffusion and consumption rates of consumers depend on elementary attributes of naturally occurring consumer-resource
interactions: the abundance, spatial aggregation, and mobility of resources as well as consumer handling-time and directional persistence.

Whether the response is behavioral or mechanistic, animal movement patterns may be indicative of properties of the environment and their ecological significance. In the second chapter, I demonstrate the utility of this notion in increasing our ecological understanding of a cryptic wildlife species, woodland caribou (*Rangifer tarandus caribou*). I focus on investigating the environmental drivers of movement by comparing the movement of multiple individual caribou across a variety of environmental conditions. Landscape variables are classified based on their hypothesized association with habitat permeability, predation risk and forage availability. I link much of the observed variability in caribou movement patterns across space and time to these environmental conditions and show how residual individual differences may reflect spatial population structure. Overall, my second chapter demonstrates the application of abstract analytical models to complex observational data through powerful statistical techniques.

In the first two chapters of my thesis, I focus on a conceptually simplistic (although useful) view of the interface between the moving animal and the landscape. An alternative approach views animal movement as a sequential decision-making process where each step is modeled as a discrete decision regarding the next location. To exemplify, Fortin *et al.* (2005) have employed an elegant ‘step selection analysis’ to describe landscape-dependent habitat selection by elk in Yellowstone National Park. This approach was based on a comparison of each observed relocation step with a set of possible steps the animal could have taken from the same starting point. The set of potential steps is determined using the statistical properties of the movement path and is then used to more accurately define the availability of potential habitats at any point in space and time. This powerful approach enabled Fortin *et al.* (2005) to provide a spatially-explicit behavioral explanation for the emergence of a trophic cascade between wolf, elk and aspen. Useful as it may be, viewing movement patterns as a series of decisions requires a more explicit consideration of the different behavioral process, motivations and constraints shaping this pattern.

Determining what an animal knows about its environment, and how that information is translated into specific movement decisions, is a major conceptual challenge faced by movement ecologists (Lima & Zollner 1996; Schick *et al.* 2008). The modeling framework I present in my
third chapter is designed to evaluate the likelihood of alternative cognitive processes based on positional and environmental data. It is centered on a flexible cognitive algorithm that provides the framework for an adaptive movement kernel. My algorithm reflects a novel approach to conceptualizing cognitive processes by relying on hierarchical relationships between three major components: sensory perception, memory and innate belief. In addition to describing the model and explaining the underlying logic, I demonstrate its potential for parameterization using simulated data and investigate the robustness of predictions over a wide range of temporal and spatial sampling scales. I show that, despite scale-dependent biases in parameter estimates, my model can reproduce descriptive probes of movement paths with little sensitivity to the scale at which these paths were sampled.

My thesis spans the range of complexities characterizing the study of animal movement ecology, from a simple analytical model, through more complex statistical methods, to detailed mechanistic simulations. These approaches reflect a complexity scale in how scientists conceptualize animal movement and its interaction with the environment. At one end of this continuum, the diffusion approximation relies on a highly simplistic view of animal movement. On the other extreme, my individual-based simulation model relies on a complex representation of moving animals as decision-making cognitive beings. In sum, my thesis demonstrates the utility of both extremes in gaining deeper understanding of animal movement across heterogeneous landscapes.
References


Chapter 1: Linking rates of diffusion and consumption in relation to resources

Abstract

The functional response is a fundamental model of the relationship between consumer intake rate and resource abundance. The random walk is a fundamental model of animal movement, well approximated by simple diffusion. Both models are central to our understanding of numerous ecological processes but are rarely linked in ecological theory. Here, I draw on the common logical premise underlying these models to derive a synthetic model, showing how the diffusion and consumption rates of consumers depend on elementary attributes of naturally occurring consumer-resource interactions: the abundance, spatial aggregation and traveling speed of resources as well as consumer handling-time and directional persistence. I show that resource aggregation may lead to increased consumer diffusion and, in the case of mobile resources, reduced consumption rate. Resource-dependent movement patterns have traditionally been attributed to area-restricted search, reflecting adaptive decision making by the consumer. My synthesis provides a simple alternative hypothesis that such patterns could also arise as a byproduct of statistical movement mechanics.

1 Published in the American Naturalist (2011, 178: 182–190) and co-authored by D. Kuefler and J. M. Fryxell.
Introduction

A major purpose of animal movement is acquiring resources. Indeed, several recent theoretical studies of movement, such as studies of Lévy walks or multiphasic random walks, focus on adaptive movement strategies that increase, and perhaps even maximize, the rate of resource encounter (Grünbaum 1998; Benhamou 2007; Bartumeus et al. 2008; James, Plank, & Brown 2008; Reynolds & Rhodes 2009). Despite this intimate relation between the study of animal movement and that of consumer-resource interactions, formal links between the two fields are rare, perhaps due to the enormous complexity inherent in spatially explicit processes.

Simple kinetic mechanisms (e.g., klinokinesis and orthokinesis; Gunn 1975; Benhamou & Bovet 1989) are extremely useful in linking consumer movement rates to local resource abundance and thus predicting the spatial distribution of consumers in a heterogeneous environment (Patlak 1953; Turchin 1991; Wilson & Richards 2000). However, the explicit mechanism underlying the regulation of speed or turn frequency in response to resources is often not postulated, thus hampering our ability to derive the process from first principles. An exception is a study by Kareiva & Odell 1987; followed by a theoretical investigation by Grünbaum (1998) which provided an empirically derived functional relationship between turn frequency and satiety. Here I take an alternative approach, based on the common logic underlying two fundamental ecological models (Turchin 2001): the functional response and the random walk. My aim is to provide a fresh perspective on the mechanistic relationship between rates of movement and consumption and to argue that naturally occurring movement patterns might be usefully viewed in light of this relationship.

The functional response relates a consumer’s intake rate to the availability of resources (Solomon 1949). This concept is fundamental to our understanding of many ecological and evolutionary processes, such as optimal foraging, population regulation and food web stability (Oaten & Murdoch 1975; Dunbrack & Giguere 1987; Turchin 2001). An explicit mathematical form of the functional response was formulated independently by Holling (1959) and Rashevsky (1959). Although many alternative mathematical formulations have been suggested (e.g., Abrams & Ginzburg 2000; Jeschke, Kopp, & Tollrian 2002 and references therein), Holling’s type I and II are the most common forms seen in real systems, having been documented dozens of times (Jeschke et al. 2002). Both functional response equations rely on the implicit
assumption that the rate of resource encounter (per unit search time) is constant over space and time. Hence, consumers are assumed to travel through a field of randomly distributed resources at a constant speed, with straight, randomly oriented, trajectories between sequential resource encounters (McKenzie, Lewis, & Merrill 2009). Indeed, straight movement paths for an appreciable distance have been documented in several organisms in a variety of different habitats (Turchin 1998), including army ants in Panama’s rain forest (Franks & Fletcher 1983), elk in Canada’s boreal forests (Morales et al. 2004) and various marine predators (Sims et al. 2008).

The most fundamental model of animal movement is the random (or Brownian) walk (Nathan et al. 2008; Codling et al. 2008), which assumes that animals move through space in a series of straight, randomly oriented steps (Berg 1983; Kareiva & Shigesada 1983; Turchin 1998; Codling et al. 2008). The assumptions underlying the random walk model are therefore identical to those governing consumer search rate in functional response models. Diffusion is a continuum approximation of the discrete random walk process, where the diffusion coefficient is the determining metric describing the rate of population spread (Turchin 1998). In many applications, the diffusion coefficient is assumed constant. However, if diffusion rates vary with local environmental conditions, the resultant spatial distribution of the population would reflect the underlying pattern of environmental heterogeneity (Patlak 1953; Cosner 2005). Hence, the diffusion approximation provides a crucial link among the behavior of individual consumers, spatial heterogeneity of resources and the resulting spatial distribution of the consumer population (Turchin 1991; Cosner 2005). I use the logical linkage between random walk and functional response models to derive the diffusion coefficient of a consumer as functions of the abundance, spatial aggregation and traveling speed of resources as well as of consumer directional persistence.

Models

Using both analytical and simulation methods, I explore several possible scenarios involving consumer-resource interactions that are broadly representative of many real organisms. In the first scenario, simple enough to allow closed-form solution of the model, I consider static resources that are distributed randomly in space. Consumers are assumed to travel along straight paths until they encounter a resource item, at which point a new travel direction is randomly sampled from a uniform distribution. In the second scenario, also allowing a closed-form
solution, I assume that both resources and consumers are mobile, moving in randomly oriented, but straight, paths across a featureless landscape. The third scenario, requiring computer simulation rather than a closed-form solution, relaxes the assumption of a uniform random resource distribution, comparing outcomes for various levels of spatial aggregation of resources. The fourth scenario relaxes the assumption of straight movement paths by the consumer, assuming instead that consumers travel according to a correlated random walk with varying degrees of directional persistence. This scenario reflects the realistic attribute that animal movement paths are often rather convoluted even when foraging is not involved (Turchin 1998; Nathan et al. 2008).

**Scenario I: randomly distributed static resources**

Simple random walks are characterized by a linear increase of an individual’s squared displacement over time. The expectation for the squared displacement, $E(R^2)$, after time $t$, can be expressed as a function of the mean duration of a single step, $E(\Delta t)$, and the mean squared step length, $E(x^2)$:

$$E(R^2) = \frac{t}{E(\Delta t)} E(x^2).$$  \hspace{1cm} \text{Eq. 1}

I accordingly derive the expected squared displacement of a consumer by expressing $E(\Delta t)$ and $E(x^2)$ as functions of resource density.

As in the Holling’s (1959) functional response models, I assume that consumption events occur whenever a resource item falls within the perception range of a consumer moving in straight lines. I further assume that the consumer’s trajectory is truncated once it encounters a resource item and thus it performs an instantaneous random reorientation after each encounter.

The elapsed time between consecutive encounters of such a consumer, moving at a constant speed, with randomly distributed immobile resources, is exponentially distributed with rate parameter $\lambda_{\Delta t}$. This rate parameter is the product of the effective radius (or the perception range) $r$, the speed of the consumer $v$, and the resource density $\rho$:

$$\lambda_{\Delta t} = 2rv\rho.$$  \hspace{1cm} \text{Eq. 2}
Note that, $2rv$ is also known as the attack rate, sensu Holling (1959). This derivation holds even if the speed is variable, as long as it is sampled from a Maxwell-Boltzmann distribution with mean $v$ (Hutchinson & Waser 2007).

The expected time between successive encounters is

$$E(\Delta t) = \frac{1}{2rv\rho}.$$  \hspace{1cm} \text{Eq. 3}

If the consumer handles each encountered resource item for time $h$ before renewing its search, the expected time between successive encounters is

$$E(\Delta t) = \frac{1}{2rv\rho} + h.$$  \hspace{1cm} \text{Eq. 4}

Note that, the inverse of the step duration is the expected consumption rate, so the reciprocals of Eq. 3 and 4 are in fact type I and II functional responses (Holling 1959), respectively.

Similarly, the distance traveled between successive encounters (i.e., the step length) is exponentially distributed with rate parameter $\lambda_x$:

$$\lambda_x = 2r\rho.$$  \hspace{1cm} \text{Eq. 5}

The expectation for the step length is thus:

$$E(x) = \frac{1}{2r\rho},$$  \hspace{1cm} \text{Eq. 6}

and the expectation for the squared step length is:

$$E(x^2) = \frac{1}{2r^2\rho^2}.$$  \hspace{1cm} \text{Eq. 7}

Note that, $h$ is not considered time spent traveling and so $E(x)$ and $E(x^2)$ are identical for type I and type II consumers.

By substituting Eq. 3 and 7 into Eq. 1, we can express the expected mean squared displacement $E(R^2)$, of a type I consumer, moving through (and interacting with) a field of randomly distributed resources, as a function of the resource density $\rho$:

$$E(R^2) = \frac{tv}{\rho r}.$$  \hspace{1cm} \text{Eq. 8}
Similarly, the expected mean squared displacement of a type II consumer is expressed by substituting Eq. 4 and 7 into Eq. 1:

\[ E(R^2) = \frac{tv}{pr(1+2pr vh)} \]  
Eq. 9

Thus, under the assumptions outlined above, resource locations are synonymous with potential turning points of a consumptive random walker.

Einstein (1905) linked the Lagrangian mean squared displacement of randomly moving particles to the corresponding Eulerian diffusion rate (in two-dimensions):

\[ E(R^2) = 4tD, \]  
Eq. 10

where \( D \) is the diffusion coefficient. It is thus possible to express the diffusion coefficient of ideal consumers as function of resource density, by substituting Eq. 8 (for type I consumers) or Eq. 9 (for type II consumers) into Eq. 10:

\[ D = \frac{v}{4pr}, \]  
Eq. 11

\[ D = \frac{v}{4pr(1+2pr vh)}. \]  
Eq. 12

Hence, my model predicts that diffusion rates should be inversely related to resource density, regardless of whether the consumer has a linear (type I; \( h = 0; \) Eq. 11) or hyperbolic (type II; \( h > 0; \) Eq. 12) functional response.

**Scenario II: randomly distributed mobile resources**

If resources travel at exactly the same constant speed as the consumer, the only required modification is the multiplication of the relative speed (the speed of the consumer relative to the resources) by the constant \( 4/\pi \) (Hutchinson & Waser 2007). Hence, for a type II consumer, the functional response is now expressed as

\[ \frac{1}{E(\Delta t)} = \frac{8pr r}{1+2pr rh}, \]  
Eq. 13

whereas the diffusion coefficient becomes
\[ D = \frac{\pi v}{16 \rho r (1 + \frac{3}{n} \rho r v h)}. \]  

Eq. 14

Alternatively, both consumers and resources may have variable movement velocities sampled from independent Maxwell-Boltzmann distributions with expectations given by \( v_{\text{consumer}} \) and \( v_{\text{resource}} \) respectively (Skellam 1958; Hutchinson & Waser 2007). Under these conditions, the mean relative speed \( \bar{v} \) is defined as

\[ \bar{v} = \sqrt{v_{\text{consumer}}^2 + v_{\text{resource}}^2} \]  

Eq. 15

and the type II functional response is

\[ \frac{1}{E(\Delta t)} = \frac{2 r \bar{v} \rho}{1 + 2 r \bar{v} \rho h}. \]  

Eq. 16

As long as there is no correlation between speed and step lengths, the diffusion coefficient can be expressed as

\[ D = \frac{v_{\text{consumer}}^2}{4 r \bar{v} \rho (1 + 2 r \bar{v} \rho h)} \]  

Eq. 17

Scenario III: aggregated resources

I used computer simulations to consider modifications to the ideal gas based model presented thus far. The first modification relates to the effect of resource spatial distribution (rather than just mean density) on rates of diffusion and consumption of a type II consumer (see Appendix B for the full Matlab simulation code). I employed a Neyman-Scott point-process (Neyman & Scott 1952; Ripley 1977) to generate non-homogeneous distributions of resources with different levels of spatial aggregation. Two variables determine the aggregation level in a Neyman-Scott point-pattern: the standard deviation of the Gaussian location of resource items within a clump, and the mean of the Poisson distributed clump size (i.e., the number of resource items per clump). In the results presented here, aggregation was varied by changing the latter while keeping the former constant (= 0.01 spatial units).

My simulation procedure is designed to estimate the distance travelled by a mobile consumer between consecutive consumption events across non-homogeneous resource landscapes. Each simulation begins with the generation of a Neyman-Scott point pattern of
resources within a two dimensional domain. Neyman-Scott clusters are added at randomly placed positions in space until the overall point density in the entire domain reaches the desired value (= 100 resource items per unit area). The focal simulation arena is located in the center of this domain. In accordance with the assumptions of the analytical model, in which steps always begin and end with consumption events (James, Pitchford, & Plank 2010), a randomly chosen resource within the simulation arena is assigned as the starting point for the consumer. The consumer is then assumed to travel along a vertical corridor of width $2r$. The nearest occurrence at which a resource item falls within this corridor is noted, the distance travelled is recorded and the simulation is terminated. Should the consumer reach the boundary of the simulation arena before encountering a resource, the corridor is extended into a new focal arena embedded within an extension of the spatial domain. This process is repeated for 10,000 replicated search trajectories.

To calculate diffusion rates I focus on a type II consumer traveling at a constant speed in accordance with the ideal gas assumptions. For each simulated step length distribution, I estimated $E(\Delta r)$ and $E(x^2)$, substituted these values into Eq. 1 to determine $E(R^2)$, and then used Eq. 10 to calculate the diffusion rate. To verify that the values derived through Eqs. 1 and 10 accurately describe simulated diffusion rates, I independently simulated 10 trajectories of a 1000 consumption events each. The diffusion coefficient of each trajectory was calculated directly by dividing the observed squared displacement by four times the duration of the trajectory, and the resulting values were plotted against the mean field approximations derived through Eqs. 1 and 10 (see Appendix A).

To address the possibility that mobile resources have a heterogeneous distribution, I assume that resource mobility is sufficient to allow for complete re-mixing of the system during the time required to handle a single resource. By the time the consumer renews search, its location is accordingly independent of the location of all resource clusters. Hence, instead of originating from a resource item, consumers in these simulations originated from a random point in space. Consumer speed was multiplied by $4/\pi$ to account for resource mobility (see scenario II above). Otherwise, steps were simulated in the exact same manner as described above. Note that, this simulation approach is only suitable when the position of the consumer is assumed independent of all other system components at the beginning of each step (James et al. 2010).
Scenario IV: CRW consumers

So far I have assumed that animals move along perfectly straight paths until they encounter a resource. Here, I use computer simulations to consider a more realistic pattern of consumer movement (see Appendix C for the full Matlab simulation code). Consumer movement between resource encounters was simulated as a correlated random walk with steps of constant length (= 0.001 spatial units) across an infinite two-dimensional space. The direction of each step was randomly drawn from a Von-Mises distribution with concentration parameter $k$, where the value of $k$ determines the directionality of the movement. $k$ is inversely proportional to the sinuosity of the movement path (sensu Benhamou 2004) so that $k = \infty$ corresponds to pure straight line motion between resource encounters (as in all previous scenarios). As before, a consumption event occurs whenever a resource item is within the consumer’s detection range (= 0.0001 spatial units) and each consumption event leads to reorientation. Each consumed resource was immediately replaced by a new randomly positioned resource item. Diffusion and consumption rates were calculated using Eqs. 1 and 10 based on a sample of 1000 consumption events for different resource densities and different $k$ values.

Results

Scenario I: randomly distributed static resources

Consumption rates increased with resource density (Eq. 3 and 4). The form of this response was linear in the case of a type I functional response, or decelerating as the consumption rate approached the asymptote ($h^{-1}$), in the case of a type II functional response (Fig. 1a). An increase in the rate of resource acquisition was accompanied by increased tortuosity of the movement path, resulting in an exponentially declining function for consumer diffusion rates, asymptotically approaching zero at high resource densities (Fig. 1b). Increased resource density dampened movement rates more severely in type II than in type I functional responses (Fig. 1b), simply because of the additional time spent handling resources (Eq. 11 and 12). Hence, high resource density enhanced consumption, whereas it suppressed diffusion rates due to the mechanical truncation of the movement trajectory.
Scenario II: randomly distributed mobile resources

The incorporation of resource mobility (Eq. 13-17) did not fundamentally change these predictions. While the consumption rate of type I consumers was extremely sensitive to changes in resource speed, this was rarely the case for type II consumers because consumption rates were usually limited by handling time (Fig. 2a). Diffusion rates of both consumer types decreased with resource speed due to increased resource encounter rates (Fig. 2b). In any case, resource mobility per se reinforced but otherwise had no qualitative effect on the trends predicted by my model and presented in Fig. 1.

Scenario III: aggregated resources

When resources were not aggregated (i.e., a Neyman-Scott point-pattern with a constant clump size of 1; the left hand side of Figs. 3a and 3b), my simulation results were in full agreement with the results of the analytical models (Scenarios I and II; Fig. 1). However, once resources became aggregated, consumer diffusion rates increased with the level of resource aggregation (Fig. 3b). This increase is the result of the modified distribution of step lengths, with increased frequency of very short steps (within resource clusters) and very long steps (between resource clusters). This increased the variance of the step length distribution, resulting in increased $E(x^2)$ and thus increased $D$.

Despite strong effects of resource aggregation on consumer diffusion rates, consumption rates were insensitive to resource aggregation, as long as resources were immobile (Fig. 3a). While the consumption rate of non-aggregated mobile resources (the left hand side of Fig. 3a) was well predicted by the analytical model (Eq. 13), consumption rates of mobile resources decreased as the level of resource aggregation increased (Fig. 3a).

Scenario IV: CRW consumers

Results from my numerical simulations indicated that consumption rates were relatively insensitive to consumer directionality, as reported in some previous studies (e.g., Hutchinson & Waser 2007; McKenzie et al. 2009; James et al. 2010; but see Scharf et al. 2006; Bartumeus et al. 2008 for different model variants). Only for highly tortuous paths (i.e., low $k$ value) were there appreciable reductions in consumption rates compared to those predicted by the analytical
model (Fig. 4a and 4b). For a type I consumer, the suppressive effect of tortuous movement on consumption rates was proportionate to resource density (Fig. 4a). However, this was not the case for a type II consumer, because the asymptotic limit imposed by handling time buffered the effects of movement directionality (Fig. 4b).

On the other hand, consumer diffusion rates were extremely sensitive to movement directionality. For a consumer traveling along highly directional paths, diffusion rates declined exponentially with resource abundance, in a manner well predicted by my scenario I analytical model (Fig. 4c and 4d). However, even mild deviation from straight line movement resulted in depressed diffusion rates that were much less sensitive to resource density (Fig. 4c and 4d). Regardless of the functional response type, at low resource densities or for consumers with highly tortuous movement paths, the diffusion rate was determined solely by directionality. It was only when either directionality or resource density were very high that consumer diffusion became sensitive to resource density (Fig. 4c and 4d).

Discussion

Consumers ranging from bacteria (Berg 2000) to wildebeest (Holdo, Holt, & Fryxell 2009) tend to linger in areas of high resource abundance. Movement rates have been accordingly suggested as a proxy for delineating profitable areas within the landscape (Barraquand & Benhamou 2008). Numerous empirical studies have demonstrated slower, more tortuous movements in habitats with abundant resources (Kuefler & Haddad 2006; Klaassen, Nolet, & Bankert 2006; Weimerskirch et al. 2007; de Knegt et al. 2007; Fryxell et al. 2007; Westerberg et al. 2008; Fryxell et al. 2008; Dias, Granadeiro, & Palmeirim 2009; Kuefler et al. 2010). It is often assumed that such non-random movement patterns are the result of an adaptive behavioral response to local resource abundance, commonly termed area-restricted search (Tinbergen, Impekoven, & Franck 1967), in which the consumer decides to decrease speed (orthokinesis) or increase turning angles (klinokinesis) after encountering a resource patch (Jander 1975; Pyke, Pulliam, & Charnov 1977; Kareiva & Odell 1987; Benhamou & Bovet 1989; Biesinger & Haefner 2005 and references therein). My model offers an alternative mechanistic explanation.

I have demonstrated that under a wide range of conditions, the rate of diffusion should be strongly influenced by resource density even without any decisive behavioral response to
resource abundance or spatial heterogeneity. It is an inevitable by-product of statistical movement mechanics, provided that new directions are sampled at each resource encounter. Movement trajectories generated by my model result in resource-dependent shifts in step length and turn angle distributions, similar to those commonly interpreted as an adaptive decision-based foraging strategy. As such, my model might be regarded as a null model against which behavioral decision-making hypotheses might be evaluated.

Increasing sinuosity following resource encounters is an efficient adaptive strategy for the utilization of patchy resources (Benhamou 2007). In terms of maximizing intake rate, such a strategy is expected to outperform orthokinesis (due to the correlation between speed and encounter rate) and is certainly superior to the simple mechanistic effect demonstrated here. I am not rejecting the notion of area restricted search via klinokinesis as an adaptive behavioral strategy. Rather, I suggest that inferring such a process based on empirical data requires more substantial evidence than simple demonstration that turn angle or step length distributions vary with local resource abundance. It is by coupling movement data with secondary information, such as the actual locations of resources or foraging activity (e.g., Heinrich 1979; White, Tobin, & Bell 1984; Ward & Saltz 1994; Fortin 2003; Fryxell et al. 2008) or the giving-up density (e.g., Brown 1999; Kotler, Brown, & Bouskila 2004), that the behavioral or cognitive processes underlying emerging movement properties should be investigated.

My examination of the effects of resource aggregation suggests that consumption rates should be substantially reduced when resources are both aggregated and mobile (e.g., fish schools or ungulate herds). Interestingly, this result appears to disagree with previous theoretical findings by James et al. (2010), who conclude that the functional response should be insensitive to resource spatial aggregation (note that, for immobile resources, my results support this conclusion). This apparent discrepancy stems from my explicit inclusion of a handling time, during which the consumer is immobile (a component that was not included in James et al. 2010). As resource groups continue to move during this period of consumer immobility, the consumer effectively initiates each new search from a random position in space relative to other resource groups, including the one that was encountered last. Consequently, the consumer experiences a lower de-facto resource density than the overall mean density (clusters are by definition less abundant than isolated individuals). This pattern, intuitive as it may be, may play...
an important role in stabilizing consumer-resource interactions. For example, group formation reduces the overall predation pressure experienced by Serengeti wildebeest, potentially stabilizing their interaction with lions (Fryxell et al. 2007). My work provides simple mechanistic understanding of such group-dependent functional responses.

While the analytical null model presented here relies on a highly simplified view of animal movement, it is consistent with the implicit assumptions underlying Holling’s functional response models and the ideal gas law. In nature, many animals do not travel along straight trajectories, but instead alter their course for reasons other than resource encounter. My simulations incorporating more realistic movement models, based on correlated random walks, suggest that the degree to which diffusion and consumption rates vary with consumer directionality should depend on resource density. When resources are abundant, movement patterns are largely governed by resource encounters. On the other hand, when resources are sparse, movement patterns are determined by the internal capacity to maintain directionality.

To conclude, my models incorporate fundamental attributes of naturally occurring consumer-resource interactions such as resource aggregation and mobility and consumer directionality. I provide simple null mechanisms explaining several empirically observed phenomena including decreased movement rates in preferred habitats (e.g., Kuefler & Haddad 2006), selective use of resource rich patches (e.g., Cameron & Spencer 2008) and reduced consumption rates of mobile aggregated resources (Fryxell et al. 2007). I show that diffusion rates and consumption rates are inextricably linked, through common effects superimposed by resource density, heterogeneity and mobility. Hence, all three interacting factors are consequently important determinants of ecosystem dynamics.
References


Figures

Figure 1 – Predictions of the closed form Scenario I model for (a) consumption and (b) diffusion rates, as function of resource density (dashed and solid lines represent type I and type II consumers respectively). Empty and filled square markers represent results of numerical simulations of the diffusion-consumption process for type I and type II consumers (respectively) searching for randomly distributed immobile resources. Parameter values are: $r = 0.0001$, $h = 1$, $v_{consumer} = 100$, $v_{resource} = 0$ (see Appendix B for simulation code).
Figure 2 – Predictions of the closed form Scenario II model for (a) consumption and (b) diffusion rates as function of resource speed. Dashed and solid lines represent type I and type II consumers respectively. Empty and filled square markers represent results of numerical simulations of the diffusion-consumption process for type I and type II consumers (respectively) searching for randomly distributed mobile resources with Maxwell-Boltzmann distributed velocities. Parameter values are: $r = 0.0001$, $h = 1$, $v_{\text{consumer}} = 100$, $\rho = 100$ (see Appendix D for simulation code).
Figure 3 – Simulated consumption (a) and diffusion (b) rates for type II consumers, as a function of resource aggregation. Variable levels of resource aggregation were generated by varying the mean clump size of the Neyman-Scott process, while holding constant within-clump dispersal standard-deviation and mean resource density over the entire landscape ($\sigma = 0.01$ and $\rho = 100$ respectively). A null clump size (left hand side) corresponds to a random distribution of resources as assumed in the ideal gas based model (see also Figs. 1 and 2). Circles and X’s represent results for immobile ($v_{resource} = 0$) and mobile ($v_{resource} = 100$) resources, respectively. Dotted and dashed trend lines represent linear regressions based on results for immobile and mobile resources, respectively. Other parameter values are: $r = 0.0001$, $h = 1$, $v_{consumer} = 100$ (see Appendix B for simulation code).
Figure 4 – Simulated consumption (a and b) and diffusion (c and d) rates as function of resource density for type I (a and c) and type II (b and d) consumers with four levels of directionality: $k = \infty$ (straight line movement; squares), $k = 5000$ (high directional autocorrelation; triangles), $k = 1$ (low directional autocorrelation; x’s), and $k = 0$ (pure random walk; circles). Note that, all axes are log-scaled. Other parameter values are: $h = 10$, $r = 0.0001$, $v_{\text{consumer}} = 100$, $v_{\text{resource}} = 0$ (see Appendix C for simulation code).
Chapter 2: Environmental and individual drivers of animal movement patterns across a wide geographical gradient

Abstract

Within the rapidly developing field of movement ecology, much attention has been given to studying the movement of individuals within a subset of their population’s occupied range. Our understanding of the effects of landscape heterogeneity on animal movement is still fairly limited as it requires studying the movement of multiple individuals across a variety of environmental conditions. Gaining deeper understanding of the environmental drivers of movement is a crucial component of predictive models of population spread and habitat selection and may help inform management and conservation. In Ontario, woodland caribou (*Rangifer tarandus caribou*) occur along a wide geographical gradient ranging from the boreal forest to the Hudson Bay floodplains. I used high resolution GPS data, collected from 114 individuals across a 450,000 km$^2$ area in northern Ontario, to link movement behavior to underlying local environmental variables associated with habitat permeability, predation risk and forage availability. I show that a great deal of observed variability in movement patterns across space and time can be attributed to local environmental conditions, with residual individual differences that may reflect spatial population structure. I discuss my results in the context of current knowledge of movement and caribou ecology and highlight potential applications of my approach to the study of wide-ranging animals.

---

2 Published in the Journal of Animal Ecology (2012, in press) and co-authored by A. Mosser, G. S. Brown and J. M. Fryxell
Introduction

Movement ecology is a rapidly growing field of research aimed at understanding the "causes, mechanisms, and spatiotemporal patterns of movement and their role in various ecological and evolutionary processes" (Nathan et al. 2008). One potential way to test putative causal mechanisms is to link observed movement patterns with the spatial and temporal variability in their underlying environmental conditions. As experimental manipulation is often unpractical in many ecological systems, a promising observational approach is to use "space as a surrogate" (McIntire & Fajardo 2009) – comparing the movement behavior of multiple individuals across wide geographical and seasonal gradients, thus encompassing a variety of environmental conditions. Despite the benefits of such large scale comparative studies, the high cost and technical challenges involved in tracking animals in the wild have heretofore resulted largely in studies focused on local populations within a limited portion of their occupied range.

The expected squared displacement of an animal’s path, \( E(R^2) \), has long been a central concept in the study of animal movement (Kareiva & Shigesada 1983). It is directly proportional to the population rate of spread (Turchin 1998) and may serve to quantify migration and dispersal patterns (Bunnefeld et al. 2011; Börger & Fryxell 2012). Furthermore, \( E(R^2) \) has been nominated recently as a single summary statistic that best captures key statistical properties of animal movements (Nouvellet et al. 2009). Here I demonstrate the value of this summary statistic in linking local environmental conditions to animal movement rates. While the observed displacement of an individual is but a single realization of a stochastic movement process, \( E(R^2) \) reflects the mean displacement expected under a set of assumptions regarding the statistical properties of the process. If these assumptions hold true, \( E(R^2) \) allows one to better assess the general patterns and accordingly identify the ecological drivers of animal movement. This approach holds great potential for dealing successfully with a central challenge in movement ecology – scaling up animal movement across heterogeneous landscapes (Morales & Ellner 2002).

Movement patterns result from interactions between animals and their environments (Johnson et al. 1992; Schick et al. 2008). Theory suggests that movement rates should decrease when animals travel through resource rich areas (Pyke, Pulliam & Charnov 1977; Avgar, Kuefler & Fryxell 2011), supported by numerous empirical studies (e.g., Klaassen et al. 2006; Fryxell et
Moreover, physical impediments, such as thick undergrowth or deep snow, may slow movement rates (Morales & Ellner 2002; Schooley & Wiens 2004). Conversely, unprofitable local conditions, such as sites with low forage availability or high predation risk, may stimulate increased movement rates (e.g., Englund & Olsson 1996; Gilliam & Fraser 2001; Frair et al. 2005), as would physical properties that increase habitat permeability, such as roads or waterways (e.g., James 1999). Analyzing movement attributes in light of local environmental conditions may thus help identify habitat preferences of free ranging animals (Fauchald & Tveraa 2003; Barraquand & Benhamou 2008; Bastille-Rousseau, Fortin & Dussault 2010) while controlling for the mechanical effects of habitat permeability. Such an approach presents an attractive alternative to resource selection functions that have come into popular use in recent years (Boyce & McDonald 1999), because local trajectory responses may be less sensitive to the arbitrary definition of available resources (Beyer et al. 2010).

Movement patterns, as well as their relationship with underlying environmental conditions, are often scale dependent (Fryxell et al. 2008; Rivrud, Loe & Mysterud 2010). Animals may respond to their environment at specific spatial scales and the resulting movement patterns may be only discernible at a narrow range of temporal resolutions. Consequently, linking movement patterns to their underlying environmental drivers is often a complex task. Here I deal with this task by re-sampling animal movement paths at three different temporal resolutions and matching the resulting movement data with environmental variables sampled at the appropriate spatial resolution (as determined by the spatial scale of movement at each temporal resolution).

I explore the connection between landscape attributes and animal movement patterns for a population of woodland caribou (*Rangifer tarandus caribou*) that inhabit a vast geographical range which encompasses a variety of environmental conditions. Woodland caribou occupy the southern range of caribou in North America and are considered a threatened subspecies over most of this range (COSEWIC 2002). Whereas several previous studies have focused on comparative analysis of caribou movement characteristics (e.g., Bergman, Schaefer & Luttich 2000), and their temporal dynamics (e.g., Ferguson & Elkie 2004a; Rettie & Messier 2001), few studies have investigated possible environmental drivers of these characteristics. Noteworthy exceptions are Johnson et al.'s studies (2002a; 2002b) that estimate scale-specific effects of
cover type on caribou movement in north-central British Columbia. The extensive geographic range of caribou in Ontario, from the boreal forest in the south-west to the Hudson Bay floodplains at the north-eastern extent, provides a unique opportunity to investigate the attributes and environmental correlates of movement in a wide-ranging herbivore across a broad range of ecological conditions.

I evaluate the relative importance of spatiotemporal variables associated with three main functional aspects of the landscape (forage availability, predation risk and habitat permeability) as potential drivers of caribou movement patterns. Interpretation of the effects of different habitat variables on biological processes is often hampered by the covariation characterizing spatial and temporal data (Rossi et al. 1992; Bini et al. 2009). For example, in my system, areas regenerating after natural or anthropogenic disturbance often support high population densities of moose (Alces alces; Rempel et al. 1997), the preferred prey species for wolves (Canis lupus; Boertje, Valkenburg & McNay 1996), which in turn are the major predator of woodland caribou (Bergerud & Elliott 1986; Seip 1992; McLoughlin et al. 2003). It is accordingly plausible that caribou might avoid or travel faster through regenerating forest stands, as supported by some field evidence (Cumming, Beange & Lavoie 1996; James et al. 2004; Vors et al. 2007; Leblond et al. 2011). On the other hand, during the growing season regenerating forest stands often have abundant annual vesicular plants and may thus be attractive as foraging sites (Schaefer & Pruitt 1991; Leblond et al. 2011) in which caribou tend to linger. Here I deal with such discrepancies by classifying environmental variables into functional groups and making a priori predictions of their effects, based on current knowledge of caribou ecology. This approach increases my confidence in the biological interpretation of my results and allows to identify fundamental relationships between local environmental conditions and animal movement patterns. Consequently I am able to account for a great deal of the variability observed in caribou movement patterns across space and time and gain insights into habitat preference and population structure. I discuss implications of my results for caribou conservation and major applications of my approach in the context of movement ecology.
Materials and methods

Data collection

The telemetry data employed in this study were collected by the Ontario Ministry of Natural Resources (OMNR). During February-April of 2009 and 2010, 165 adult female caribou across Northern-Ontario were net-gunned from a helicopter and fitted with a GPS-Argos telemetry collars (Telonics Inc.). Capturing and handling were performed according to Ontario animal care regulations. GPS locations were recorded every 5 hours during the first 13 months after collaring, and every 25 hr thereafter, and were transmitted to OMNR via an Argos satellite uplink every 5 days. Telemetry points collected during the first 24 hr after capturing, the last 24 hr before a mortality event, and those with obvious errors (e.g., a roundtrip to a remote location performed over a short time; less than 0.1% of all fixes) were excluded from my analysis. Overall, the telemetry dataset was composed of more than 200,000 GPS points, taken over a period of three years, across a 450,000 km² area.

Each telemetry point was affiliated with eight environmental variables characterizing that locality at that time. Environmental variables were collected from three main sources. MODIS-based Normalized Difference Vegetation Index (NDVI) values (temporal resolution: 16 days, spatial resolution: 250 m pixels) were obtained via the Land Processes Distributed Active Archive Center at the U.S. Geological Survey Earth Resources Observation and Science Center. Values of ΔNDVI (indicative of rate of vegetative growth rather than standing biomass; Pettorelli et al. 2005) were calculated as the difference between the current NDVI values and those 16 days previous. Snow depth data (temporal resolution: 3 hr, spatial resolution: 40 km pixels) were obtained via the North America Regional Reanalysis data (NARR dataset DSI-6175), from NOAA Operational Model Archive and Distribution System. Land cover types and distances to different landscape features (spatial resolution: 25 m pixels) were obtained from the Ontario Provincial Land Cover Database. This database is based on the Thematic Mapper sensor on Landsat Earth-Resource satellites using data frames recorded between 1999 and 2002 (Spectranalysis Inc. 2004). Annual forest harvest and natural disturbance spatial layers available from the OMNR were used to update the land cover classification to ensure it accurately reflected habitat conditions during the years of animal monitoring.
Environmental variables were recorded for each GPS point at three spatial scales, within radii of 250, 950 and 9300 m. These radii correspond to half of the median displacements (i.e., half the length of a typical step) of caribou over the entire study period at three temporal resolutions: 5 hr, 25 hr and 30 days, respectively. Temporally dynamic variables were averaged over the relevant temporal steps (e.g., a 25 hr step is affiliated with the average of eight snow depth values occurring at that locality during that time). All spatial processing was conducted using ArcGIS desktop 9.2 and 9.3 (ESRI Inc.).

Predictions

Explanatory variables were classified according to their expected impact on three functional aspects of the landscape. To account for potential interactions between environmental variables and time of year (e.g., lakes may enhance movement when frozen during winter yet impede it during the summer), all explanatory variables were separated into summer (May-October) and winter (November-April) effects.

The first functional aspect of the landscape was habitat permeability. Permeability should be positively correlated with water cover during the winter months (when lakes are frozen and may serve as movement corridors), and with open habitat (a combination of all treeless land cover classes) year-round. Hence, these variables were expected to have a positive effect on movement rates. Note however that treeless areas may still have dense vegetation cover impeding movement and hence weakening this positive effect. Conversely, permeability should be negatively correlated with winter snow depth, which was thus expected to have a negative effect on movement rates. Snow has been shown to inhibit caribou movement in Alberta (Stuart-Smith et al. 1997) and as a factor affecting feeding habitats in British Colombia (Johnson, Parker & Heard 2001).

The second functional aspect of the landscape was forage availability. During winter months, when caribou feed mainly on lichen (usually associated with coniferous habitats), conifer forest cover (a combination of all land cover classes dominated by coniferous species) should be positively correlated with forage availability (Brown et al. 2006). During summer months, ΔNDVI, reflecting vegetative growth or forage quality, should be positively correlated with forage availability. Finally, NDVI, reflecting vegetative cover or forage quantity (Pettorelli
et al. 2005) should be positively correlated with forage availability year-round. Note that, whereas winter NDVI values do not reflect caribou winter forage (i.e., lichen) per se, it is likely to reflect coniferous habitats (associated with lichen) and may contain information that is missing or not updated, in the conifer forest cover layer. Several studies have used NDVI to account for habitat selection (but not movement) by caribou (Gustine et al. 2006; Thomas, Johnson & Griffith 2006) and reindeer (Hansen et al. 2009a; Hansen et al. 2009b). All three variables were expected to have negative effects on movement rates due to the tendency to linger in resource-rich habitats.

The last functional aspect I considered here was predation risk. The main predators of caribou in my system are wolves, which in turn rely on moose as their major prey. Both deciduous forest cover (a combination of all land cover classes dominated by deciduous species) and regenerating forest cover (cuts and burns that are less than 20 years old) should be positively correlated with moose occurrence, and thus with high predation risk (Vors et al. 2007), and were expected to have positive effects on movement rates year-round. During the summer months, variables associated with refuge habitats (James et al. 2004), namely conifer forest cover and water cover (caribou are excellent swimmers and are known to use islands during spring and summer), should be negatively correlated with predation risk and were thus expected to have negative effects on movement rates. These predictions are summarized in Table 2.

Data processing

Fine-scale movement patterns were characterized by three basic attributes of a biased correlated random walk (Benhamou 2006; Codling et al. 2008) at two temporal resolutions. A single step was defined as two consecutive GPS locations, taken either 5 hr or 25 hr apart. Each step, $i$, at each temporal resolution, is characterized by its length, $l_i$, (in meters), and its heading, $\theta_i$, – its orientation relative to the true north (in radians). The cosine of the angular difference between the headings of two consecutive steps, $\cos(\theta_i - \theta_{i-1})$, is the step’s directional persistence, $c$. The preferred direction, $\gamma$, is defined as the overall heading of the individual’s movement path over an entire calendar month (from the first to the last GPS points). The cosine of the angular difference between the preferred direction and the step’s heading, $\cos(\gamma - \theta_i)$, is the step’s directional bias, $q$. Note that, steps and turns defined this way are not ‘natural’ steps performed
by the animal, but rather derived variables based on a sample of animal locations recorded at regular intervals over time (Turchin 1998).

To reduce the strong autocorrelation stemming from working with sequential observations in space and time, data were aggregated to obtain monthly summaries of movement attributes and environmental variables per individual per month. A monthly minimum of 20 valid 5 hr steps and 5 valid 25 hr steps was imposed. To avoid potential biases, individuals that were represented by less than three summer months, or less than three winter months, were excluded from my analysis (data for 38 excluded individuals were used to test my approach’s predictive abilities – see ‘Results’). The remaining monthly data set included 114 individuals with an average of 11 months per individual (range: 6-13) and was based on, on average, 980 steps of 5 hr per individual (range: 327-1838), 9308 steps of 5 hr per month (range: 7342-10877) and 90 steps of 5 hr per individual per month (range: 25-149).

Movement attributes were summarized for each individual, at each month, for each of the two finer temporal resolutions. The mean directional persistence, $E(c)$, is a measure of the animal’s tendency to maintain its previous movement heading. The mean directional bias, $E(q)$, is a measure of the animal’s tendency to maintain its average monthly movement heading. To account for potential biased estimations resulting from $q$-$l$ or $c$-$l$ cross-correlations (e.g., a tendency to move faster when moving towards a preferred direction), both means were taken as weighted averages based on step lengths. Note that, $E(c)$ and $E(q)$ are not independent, as the directional persistence increases with the directional bias (Benhamou 2006). To enable straightforward calculation of the expected monthly squared displacements, $E(R^2)$, I classified the path of each animal at each month as either a biased random walk (BRW) or a correlated random walk (CRW), based on the correlation between directional persistence and directional bias (Benhamou 2006). If the correlation between $c$ and $q$, $\text{cor}[q,c]$, was positive and significant, the path was classified as a BRW and its $E(R^2)$ was calculated as (Codling et al. 2008 and refs therein):

$$E(R^2) = nE(l^2) + n(n - 1)E(q)^2E(l)^2.$$  

Eq. 1
where $E(l)$ and $E(l^2)$ are the mean step length and squared step length, and $n$ is the number of steps in a month ($n = 144$ for 5 hr steps and $n = 29$ for 25 hr steps). Otherwise, the path was classified as a CRW and its $E(R^2)$ was calculated as (Benhamou 2006 and references therein):

$$E(R^2) = nE(l^2) + E(l)^2 \frac{2E(c)}{1-E(c)} (1 - \frac{1-E(c)^n}{1-E(c)}).$$  

Eq. 2

For each of the two finer temporal resolutions, I recorded $E(R^2)$, $E(l)$, $E(l^2)$, $E(q)$, and $\text{cor}[q,c]$, for each animal, at each month. At the monthly temporal resolution, I recorded the observed monthly displacement, the straight-line distance traveled by each animal during each calendar month, to be coupled with the environmental variables measured at the appropriate spatial resolution (i.e., within a 9300 m radius around the first GPS point of the month) and compared to $E(R^2)$ values obtained at the two finer temporal resolutions.

For each individual, at each month, and at each spatiotemporal resolution, mean values of all environmental variables were calculated by averaging all available values for a specific variable at a specific resolution. Individual monthly range centroids (northing and easting) were calculated as the harmonic mean of all northing and easting values for each individual in each month.

**Statistical analysis**

At each spatiotemporal resolution, the processed data set consisted of 6-13 data points (i.e., months) for each individual. Linear mixed effects models were used to account for the nested data structure and lack of independence among repeated measures. Mixed effect models enable treating individual animal identity as a random effect, thereby assigning a different intercept to each individual in the population, while treating environmental effects as fixed. This allows one to estimate regression coefficients for the entire population while accounting for individual differences. Function *lme* within package *nlme* in program R (version 2.14.0; http://cran.r-project.org/) was used, with an exponential spatial and/or temporal autocorrelation structure for the residuals.

The response variables in my analysis were the individual monthly movement attributes: $\sqrt{E(R^2)}$, $E(l)$, $\sqrt{E(l^2)}$, $E(c)$, $E(q)$, $\text{cor}[q,c]$ and observed monthly displacement. All displacement variables ($\sqrt{E(R^2)}$, $E(l)$, $\sqrt{E(l^2)}$, and observed monthly displacement) were log-transformed to
reduce heteroscedasticity. All correlation variables (E(c), E(q) and cor[q,c]) were logit-transformed. The explanatory variables were the mean environmental variables experienced by each study animal during each month, at the appropriate spatial resolution. All variables were scaled (by subtracting the within-season mean and dividing by the within-season standard deviation) so as to standardize their coefficients (Schielzeth 2010). The same candidate set of explanatory variables (Table 2) was used for all dependent variables. The appropriate residual exponential autocorrelation structure (i.e., temporal, spatial, both or none) was selected for each response variable based on AIC competition using the full set of explanatory variables (semivariograms are provided in Appendix E).

Model selection was performed according to the general guidelines in Zuur et al. (2009; chapter 5). I started by fitting the full model (including 14 explanatory variables; Table 2) for each response variable, then gradually reducing complexity, based on AIC competition, to obtain the best combination of explanatory variables (interactions were not considered). Finally, as an indicator of goodness-of-fit, the portion of variance in each response variable, explained by the selected set of environmental variables (i.e., fixed effects), was calculated as the difference between the variance of the response variable and the variance of the population level-residuals of the best model (i.e., considering only fixed effects), divided by the variance of the response variable (coded in R as: \( \frac{\text{var(response)} - \text{var(resid(best.model, level=0))}}{\text{var(response)}} \)).

Results

Expected monthly displacement values, \( \sqrt{E(R^2)} \), calculated based on either BRW or CRW, matched well with observed monthly displacement values and explained 68% (at the 5 hr resolution) and 65% (at the 25 hr resolution) of observed variability (Fig. 1). However, my BRW/CRW-based expectations tended to overestimate observed monthly displacement values at the higher end of the range (Fig. 1). Overestimations were positively associated with high NDVI values (explaining ~10% of the variation in \( \sqrt{E(R^2)} \)-observed monthly displacement deviations), and had an additional individual (i.e., random) component (explaining an additional ~10%). The frequency of monthly paths classified as BRW was 13% at the 5 hr resolution and 7% at the 25 hr resolution. All movement attributes varied substantially over time and space (Table 1 and Fig. 2). Overall, movement rates were highest in the north and during the winter (Fig. 2).
Most environmental variables considered here were selected as predictors of monthly movement rates during at least one of the seasons, and the direction of their effects matched my predictions (Table 2). The only notable exception was the positive effect of conifer forest cover on observed monthly displacement during winter. Both regenerating forest cover and deciduous forest cover, variables affiliated with predation risk, had negligible effects on monthly movement rates at all resolutions. The same was true for open habitat (affiliated with permeability) during the summer.

My models accounted for most of the observed variability in caribou monthly displacement across space and time, a considerable portion of which was explained by the environmental variables (Table 2). Regression coefficients magnitude indicated that NDVI had the largest effect on monthly displacement during the summer whereas variables affiliated with habitat permeability were most influential during the winter. The observed monthly displacement model included fewer fixed effects, and had a lower goodness-of-fit, than the two \( \sqrt{E(R^2)} \) models.

Biased correlated random walk components (\( E(l) \), \( E(l^2) \), \( E(c) \), \( E(q) \) and \( \text{cor}[q,c] \)) varied in their response to environmental variables but overall reflected the trends observed in \( \sqrt{E(R^2)} \) (while this was true for both the 5 hr and 25 hr resolutions, I report here the results for the former; Table 3). Environmental conditions had the greatest influence on \( E(c) \) whereas \( \text{cor}[q,c] \) was the least sensitive component to the variables considered here. Note that, contradictory to my prediction, habitat variables thought to be affiliated with winter predation risk had negative effects on both \( E(l) \) and \( E(l^2) \) whereas summer \( \Delta NDVI \) had a positive effect on \( E(q) \). Overall, predicted \( \sqrt{E(R^2)} \) values, calculated based on the predicted values for each of the biased correlated random walk components, (CRW was assumed if \( \text{cor}[q,c] \leq 0 \) and BRW was assumed otherwise) matched observed monthly displacement values well (explaining \(~40\%\) of observed variability) and were unbiased (linear regression intercept and slope did not significantly differ from 0 and 1, respectively).

Random effects in my statistical models reflect individual differences that are not explained by the fixed (environmental) variables (Tables 2 and 3). Approximately 20% of the variability in monthly displacement rates was captured by individual identities. Individual effects did not vary with respect to the animal’s estimated age at the time of capture. Individual effects
did show a clear spatial trend however, with a general increase from south to north (Fig. 3a). This trend was best modeled as a piecewise liner model of the individual effects as function of latitude (I found no significant effect of longitude). The model, fitted using function \textit{piecewise.linear} within package \textit{SiZer}, outcompeted any continuous model (based on AIC), indicating a breakpoint in the spatial trend at latitude 13028647 m (Lambert conformal conic projection; Fig. 3b).

I evaluated the predictive power of my approach using movement data previously omitted from my analysis. I used the best models selected for each of the biased correlated random walk components (at 5 hr resolution; table 3), together with latitude-dependent piecewise liner models of their random effects, to predict $\sqrt{E(R^2)}$ for 38 individuals over 160 months, that had been excluded from my statistical models. These predictions were compared against the observed monthly displacement values (Fig. 4). Approximately 22% of the variance in observed values was explained by my predictions. The linear regression intercept was however significantly smaller than 0, whereas the regression slope was significantly larger than 1, indicating a tendency of my models to over-predict at low displacement rates and under-predict at high displacement rates.

Discussion

Several techniques have been developed to help identify profitable foraging areas based on attributes of the animal’s movement path (reviewed in Barraquand & Benhamou 2008). These techniques vary from the intuitive ‘first passage time’ analysis (Fauchald & Tveraa 2003) applied e.g. to bottlenose dolphins in the Inverness Firth (Bailey & Thompson 2006), through to sophisticated state-space models (Morales \textit{et al}. 2004) applied e.g. to elk in the Rocky Mountains (Frair \textit{et al}. 2005), to residence time analysis applied to Arctic fulmars in the Faroe Islands (Barraquand & Benhamou 2008). Recently, Boettiger \textit{et al}. (2011) quantified African elephant movement by calculating the smallest ellipse that contained all GPS points within a 24 hr window and then linking the attributes of these ellipses to habitat variables using linear filtering. My focal response variable, $E(R^2)$, is inversely proportional to the residence time proposed by Barraquand & Benhamou (2008) and may be further used to link spatiotemporal habitat distribution to population distribution and redistribution patterns (Turchin 1991).
Regardless of their methodological approach, the vast majority of current studies in movement ecology focus on a few individuals, occupying a small subset of the population range. This calls into question the degree to which estimated parameters truly reflect movement properties at the population level. The spatial and temporal extent of my study yielded strong statistical leverage for the identification and relative ranking of multiple landscape attributes, giving us a firmer basis for characterizing the study population.

Foragers, from ants (Avgar, Giladi & Nathan 2008) to antelopes (Fryxell, Wilmshurst & Sinclair 2004), change their movement behavior in response to local forage abundance. For ungulates, time spent feeding is inversely, and disproportionally, related to overall movement rate (Owen-Smith et al. 2010). My results clearly support the hypothesis that forage quality and availability (measured by NDVI and ΔNDVI during the summer and conifer forest cover and NDVI during the winter) suppresses movement rates by woodland caribou. This could explain why previous resource selection studies have found that caribou are most commonly found in conifer stands as well as regenerating forest stands (Ferguson & Elkie 2004b, Brown et al. 2007). Other studies of large herbivores have similarly found strong links between food resource availability and probability of habitat use, including gazelles, elk, reindeer and elephants (Fryxell, Wilmshurst & Sinclair 2004; Frair et al. 2005; Mueller et al. 2007; Fryxell et al. 2008; Hansen et al. 2009b; Boettiger et al. 2011).

Support for the predation risk hypothesis was more equivocal. Presumed indicators of winter predation risk (deciduous and regenerating forest cover) had little effect on caribou movement. Indicators of summer refuge habitat (conifer forest and water cover) were more strongly supported, which might indicate greater responsiveness by female woodland caribou to predation pressure during the summer months, when caribou calves are most vulnerable. Previous studies of other large ungulates have similarly found ambiguous results. Plains zebras in Kenya travel faster, yet with sharper turns, in areas of high predation risk (Fischhoff et al. 2007). Elk travel faster through areas of high predation risk in the Rocky Mountains of Canada (Frair et al. 2005). A second study in Yellowstone National Park, however, found high variability in the response of individual elk to local wolf densities, with some individuals showing reduced movement rates through areas of high predation risk and no obvious trend at the population level (Forester et al. 2007).
Such ambiguity might well be expected from a theoretical standpoint, as animals might reduce their movement rate to reduce the probability of encountering a predator or alternatively increase their movement rate to avoid high predation areas all together. Indeed, it has been suggested that woodland caribou avoid predators at the home-range level rather than through more fine-grained patterns of habitat use (Rettie and Messier 2000). Coarse scale range selection would quite likely result in low explanatory power of predation-related habitat variables at a finer spatial scale. Moreover, the dynamic nature of predation risk across space and time likely weakens its correlation with remote-sensed habitat characteristics. Hence, without direct assessment of local habitat utilization by predators at the appropriate spatiotemporal scale, evaluation of movement responses to predation risk based on habitat attributes alone is necessarily tentative.

Physical or structural attributes of the landscape may also affect permeability to mobile herbivores. Due to the inherent spatiotemporal covariation of landscape attributes, accounting for habitat permeability is therefore crucial for unbiased inference regarding the effects of other functional characteristics of the landscape on movement patterns (Gaillard et al. 2010). Previous studies of other large herbivores have demonstrated that habitat permeability can have direct fitness consequences due to its impact on energy balance (e.g. deep snow, thick vegetation or elevation gradients; Parker, Robbins & Hanley 1984; Dailey & Hobbs 1989). While the magnitude of the response of woodland caribou movement rate to habitat permeability does not necessarily translate into absolute energy expenditure, it does allow a comparison of the relative effects of different structural attributes, perhaps providing insights into the energetic consequence of future environmental change (Mysterud et al. 2008).

Temporally dynamic environmental variables are rarely used in studies of animal movement and habitat preference. Movement patterns reflect interactions between animals and their environment (Schick et al. 2008) and understanding them often requires consideration of the temporally dynamic nature of these environments (Mueller et al. 2011). For example, spatial distributions of Thomson’s gazelles in Africa and Mongolian gazelles in Asia continually change from month to month, like a shifting mosaic (Fryxell, Wilmshurst & Sinclair 2004; Mueller et al. 2007). Similarly, home range size of red deer at multiple temporal scales is driven by temporally dynamic variable such as temperature, precipitation, day length and snow cover (Rivrud, Loe &
Mysterud 2010). Hence, studies of animal movement can greatly benefit from the incorporation of temporally dynamic environmental variables, as demonstrated by the relative large effect size of NDVI, ΔNDVI and snow depth in my study. Indeed, comparison between the movement characteristics of different ungulates species suggest that major classes of movement, such as migration and nomadism, might reflect an adaptation to the spatiotemporal dynamics of resource distribution across the landscape (Mueller and Fagan 2008; Mueller et al. 2011).

My analysis was conducted at three spatiotemporal scales to account for possible scale-specific relationship between movement patterns and environmental variables. Additional analysis (not reported here) indicates that the portion of variance explained by the fixed effects decline whenever there is a mismatch between the temporal resolution of the response variable and the spatial resolution of the explanatory variables. While my results do not indicate substantial difference between the two finer temporal resolutions (i.e., 5 hr and 25 hr), inference of environmental drivers of movement at a monthly resolution was slightly weaker, possibly due to the coarse spatial resolution of the environmental variables.

Analysis of telemetry data is often confounded by the nested nature of such data, where each individual is represented by multiple data points. Here I have dealt with the resulting pseudoreplication by using mixed effects models, allowing us to infer environmental effects (i.e., the fixed effect) on caribou movement while taking into account the hierarchical nature of the data. Additional useful information may lay in the estimated individual intercepts (i.e., the random effects). Spatial trends evident in the random effects (Fig. 3) might indicate environmental gradients (other than the ones accounted for here) affecting movement attributes. Alternatively, such trends could be attributed to innate behavioral differences among individuals, either in the propensity to move or in the response to local conditions, reflecting underlying spatial population structure.

Woodland caribou in Ontario is commonly delineated into a sedentary forest-dwelling ecotype and a migratory forest-tundra ecotype (Thomas 1995; Harris 1999; COSEWIC 2002; OWCRT 2008; Nagy et al. 2011). Previous distinctions between these woodland caribou ecotypes have been based on qualitative characterizations of migratory and calving behaviors. The spatial patterns of individual effects reflected in Fig. 3 provide a more rigorous quantitative corroboration of ecotype delineation based on fine scale movement behavior. The latitudinal
breakpoint detected in this pattern may indicate the transition zone between these two ecotypes and thus inform management policy.

I have demonstrated the use of an extensive data set of telemetry and remote sensed environmental variables to link patterns of movement to local conditions experienced by individual animals across a vast landscape. The approach presented here has several potential applications. First, as a behavior-based indicator for habitat preference, my approach might complement traditional analysis based on resource selection functions, providing the basis for predictive models of animal occurrence (Moorcroft & Barnett 2008; see Bastille-Rousseau, Fortin & Dussault 2010 for a detailed discussion on residency time analysis). Unlike most resource selection analyses (relying on a comparison of habitat use versus availability) my approach does not depend on a definition of availability, enabling a straightforward consideration of temporally dynamic variables and allowing the incorporation of random effects (but see Gillies et al. 2006; Fieberg et al. 2010). Second, habitat dependent movement rates could help with development of spatially-explicit models of gene-flow or range-expansion across novel or altered landscapes. Third, variables affecting habitat permeability are often crucial components in spatially-explicit energetic models and may serve to infer the effects of future climate change on energy expenditure by animals in the wild (Parker, Robbins & Hanley 1984; Dailey & Hobbs 1989; Mysterud et al. 2008). Lastly, residual random effects may be useful in identifying sub-population structure. Fuelled by rapid advancement in both telemetry and remote sensing technologies, these applications will advance our understanding and conservation of wide-ranging wildlife species.
References


### Tables

**Table 1 – Caribou movement attributes.**

<table>
<thead>
<tr>
<th>temporal resolution</th>
<th>movement attribute*</th>
<th>lower 95% CI</th>
<th>median</th>
<th>upper 95% CI</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>E(l) (m/hr)</td>
<td>46.14</td>
<td>196.95</td>
<td>615.90</td>
</tr>
<tr>
<td></td>
<td>√E(l²) (m/hr)</td>
<td>87.30</td>
<td>341.65</td>
<td>948.57</td>
</tr>
<tr>
<td>5 hr</td>
<td>E(q)</td>
<td>-0.12</td>
<td>0.14</td>
<td>0.66</td>
</tr>
<tr>
<td></td>
<td>E(c)</td>
<td>-0.22</td>
<td>0.25</td>
<td>0.77</td>
</tr>
<tr>
<td></td>
<td>√E(R²) (km/month)</td>
<td>5.17</td>
<td>23.00</td>
<td>183.05</td>
</tr>
<tr>
<td></td>
<td>E(l) (m/hr)</td>
<td>23.93</td>
<td>124.18</td>
<td>511.02</td>
</tr>
<tr>
<td></td>
<td>√E(l²) (m/hr)</td>
<td>33.85</td>
<td>184.71</td>
<td>661.66</td>
</tr>
<tr>
<td>25 hr</td>
<td>E(q)</td>
<td>-0.26</td>
<td>0.22</td>
<td>0.79</td>
</tr>
<tr>
<td></td>
<td>E(c)</td>
<td>-0.70</td>
<td>0.13</td>
<td>0.84</td>
</tr>
<tr>
<td></td>
<td>√E(R²) (km/month)</td>
<td>4.08</td>
<td>26.93</td>
<td>192.89</td>
</tr>
<tr>
<td>30 d</td>
<td>observed monthly displacement (km/month)</td>
<td>0.86</td>
<td>20.30</td>
<td>197.67</td>
</tr>
</tbody>
</table>

*By order of appearance these attributes are: the mean step length, the mean squared step length, the mean directional bias, the mean directional persistence and the mean squared displacement.
Table 2 – Monthly displacements - predictions and results.

A priori predictions and results for the effects of different landscape variables on woodland caribou movement across space and time at three different spatiotemporal resolutions. As all variables were scaled, the regression coefficients presented here are standardized and their magnitudes indicate their effect size.

<table>
<thead>
<tr>
<th>functional aspect</th>
<th>environmental variable</th>
<th>predicted effect</th>
<th>observed monthly displacement (30 d)</th>
<th>E($R^2$) (25 hr)</th>
<th>E($R^2$) (5 hr)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>summer</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>habitat permeability</td>
<td>open</td>
<td>↑</td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>forage availability</td>
<td>NDVI</td>
<td>↓</td>
<td>-0.71</td>
<td>-0.64</td>
<td>-0.52</td>
</tr>
<tr>
<td>forage availability</td>
<td>ΔNDVI</td>
<td>↓</td>
<td>-</td>
<td>-0.16</td>
<td>-0.15</td>
</tr>
<tr>
<td>predation risk</td>
<td>conifer</td>
<td>↓</td>
<td>-</td>
<td>-0.17</td>
<td>-0.21</td>
</tr>
<tr>
<td>predation risk</td>
<td>water</td>
<td>↓</td>
<td>-0.22</td>
<td>-0.17</td>
<td>-0.10</td>
</tr>
<tr>
<td>predation risk</td>
<td>deciduous</td>
<td>↑</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>predation risk</td>
<td>regenerating</td>
<td>↑</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td><strong>winter</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>habitat permeability</td>
<td>open</td>
<td>↑</td>
<td>0.25</td>
<td>0.39</td>
<td>0.36</td>
</tr>
<tr>
<td>habitat permeability</td>
<td>water</td>
<td>↑</td>
<td></td>
<td>0.23</td>
<td>0.22</td>
</tr>
<tr>
<td>habitat permeability</td>
<td>snow depth</td>
<td>↓</td>
<td>-0.29</td>
<td>-0.24</td>
<td>-0.31</td>
</tr>
<tr>
<td>forage availability</td>
<td>NDVI</td>
<td>↓</td>
<td>-0.29</td>
<td>-0.25</td>
<td>-0.26</td>
</tr>
<tr>
<td>forage availability</td>
<td>conifer</td>
<td>↓</td>
<td>0.13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>predation risk</td>
<td>deciduous</td>
<td>↑</td>
<td></td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>predation risk</td>
<td>regenerating</td>
<td>↑</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>intercept</td>
<td></td>
<td></td>
<td>0.29</td>
<td>0.26</td>
<td>0.22</td>
</tr>
<tr>
<td>random effects standard deviation</td>
<td></td>
<td></td>
<td>0.40</td>
<td>0.36</td>
<td>0.38</td>
</tr>
<tr>
<td>variance explained by fixed effects</td>
<td></td>
<td></td>
<td>30%</td>
<td>43%</td>
<td>42%</td>
</tr>
<tr>
<td>total explained variance</td>
<td></td>
<td></td>
<td>50%</td>
<td>59%</td>
<td>60%</td>
</tr>
</tbody>
</table>
Table 3 – Biased correlated random walk components (5 hr).

The effects of different landscape variables on the spatiotemporal variability in the components of caribou biased correlated random walk at 5 hr resolution. As all variables were scaled, the regression coefficients presented here are standardized and their magnitudes indicate their effect size.

<table>
<thead>
<tr>
<th>functional aspect</th>
<th>environmental variable</th>
<th>E(l)</th>
<th>E(l^2)</th>
<th>E(q)</th>
<th>E(c)</th>
<th>cor[q,c]</th>
</tr>
</thead>
<tbody>
<tr>
<td>habitat permeability</td>
<td>open</td>
<td>-</td>
<td>0.33</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>forage availability</td>
<td>NDVI</td>
<td>-0.27</td>
<td>-0.44</td>
<td>-0.39</td>
<td>-0.72</td>
<td>-0.30</td>
</tr>
<tr>
<td></td>
<td>ΔNDVI</td>
<td>-</td>
<td>-0.11</td>
<td>0.12</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>summer</td>
<td>conifer</td>
<td>-0.37</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>water</td>
<td>-0.15</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>deciduous</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>regenerating</td>
<td>-</td>
<td>0.17</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>predation risk</td>
<td>open</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.28</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>water</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.21</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>snow depth</td>
<td>-0.44</td>
<td>-0.41</td>
<td>-0.24</td>
<td>-</td>
<td>-0.21</td>
</tr>
<tr>
<td>winter</td>
<td>NDVI</td>
<td>-0.32</td>
<td>-0.28</td>
<td>-0.43</td>
<td>-0.19</td>
<td>-0.34</td>
</tr>
<tr>
<td>forage availability</td>
<td>conifer</td>
<td>-0.54</td>
<td>-0.49</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>deciduous</td>
<td>-</td>
<td>-0.11</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>regenerating</td>
<td>-0.16</td>
<td>-0.16</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>residuals autocorrelation range</td>
<td></td>
<td>1.2 months</td>
<td>0.6 months</td>
<td>1.5 km</td>
<td>0.6 months</td>
<td>0.39 km</td>
</tr>
<tr>
<td>variance explained by fixed effects</td>
<td></td>
<td>42%</td>
<td>43%</td>
<td>17%</td>
<td>39%</td>
<td>8%</td>
</tr>
<tr>
<td>total explained variance</td>
<td></td>
<td>54%</td>
<td>56%</td>
<td>25%</td>
<td>56%</td>
<td>16%</td>
</tr>
</tbody>
</table>
Figures

Figure 1 – observed vs. expected monthly displacements (observed monthly displacement vs. $\sqrt{\mathbb{E}(R^2)}$). Circles represent expectations based on a CRW while triangles represent expectations based on a BRW. Small markers represent expectations based on 5 hr steps while large markers represent expectation based on 25 hr steps. The solid line represents a perfect match (for reference).
**Figure 2** – seasonal means of observed (blue) and expected (at 5 hr resolution; red) monthly displacements. Points represent the centroids of the seasonal range of each individual. Color gradient represent interpolated (krigged) values of monthly displacements. The grey area in the north-west is Hudson Bay. Expected monthly displacements at 25 hr resolution show very similar trends.
**Figure 3** – individual effects on $\sqrt{E(R^2)}$ at 5 hr resolution. (a) a map illustration of spatial trends in estimated individual movement rates (i.e., random effects), possibly reflecting spatial population structure (see ‘Discussion’). Dots represent the centroids of the annual range of each individual. Color gradient represent interpolated (krigged) values of individual $\sqrt{E(R^2)}$ in km. (b) Individual movement rates as function of range latitude. Lines represent a fitted piecewise linear model.
Figure 4 – observed vs. predicted (at 5 hr resolution) monthly displacements for animal-months omitted from my analysis. The black line represents a perfect match (for reference).
Chapter 3: An empirically parameterized individual based model of animal movement, perception and memory

Abstract

Our capacity to predict patterns of animal movement behavior is limited by our understanding of the underlying cognitive process. Determining what an animal knows about its environment, and how that information is translated into specific movement behaviors, is a conceptual challenge faced by movement ecologists. The modeling framework presented here is designed to evaluate the likelihood of alternative processes of perception, memory and decision making, based on readily available positional data and environmental metrics. The model is based on a flexible cognitive algorithm that provides the framework for an adaptive movement kernel. This enables a straightforward methodology for estimating key parameters for sensory perception, memory and movement while providing testable predictions of animal resource selection and space use patterns. In addition to describing the model and explaining the underlying logic, I demonstrate its parameterization potential using simulated data and investigate the robustness of its predictions over a wide range of temporal and spatial sampling scales. I show that the model can reproduce descriptive probes of movement paths with little sensitivity to the scale at which these paths were sampled and I discuss the merits of my approach in the context of movement- and cognitive-ecology and evolution.

In press in Ecological Modelling and co-authored by R. Deardon and J. M. Fryxell.
Introduction

Overview

I model animal movement as a series of discrete relocation decisions performed over a discrete hexagonal gridded landscape. Each probabilistic decision is shaped by the distribution of relative costs versus benefits of nearby grid cells. The probabilities of use of these cells, termed the redistribution kernel, are calculated at each step based on the state of the landscape and the hidden informational state of the animal. The informational state regarding the attributes of each landscape cell is continuously updated using a fully parametric cognitive algorithm incorporating sensory perception, memory and innate beliefs. The redistribution kernel allows simulation of movement patterns based on specific cognitive abilities and limitations. More importantly, it provides the means for likelihood-based inference of the behavioral process underlying movement patterns from real data.

Background

Inferring behavioral processes from observed patterns is a central challenge in behavioral ecology. This challenge is a central theme in the emerging field of movement ecology (Nathan et al. 2008). In recent years, detailed data on animal movement for a variety of taxa and ecosystems have become readily available due to better, lighter and more affordable, telemetry systems (Hebblewhite & Haydon, 2010). Despite these technical advances, linking such relocation data to explicit behavioral processes is often difficult using traditional inferential tools and requires the use of sophisticated modeling approaches. One such approach is state-space modeling (Morales et al. 2004; Forester et al. 2007; Patterson et al. 2008). A state-space model allows simultaneous estimation of movement attributes (e.g. velocity, turn frequency), underlying behavioral states (e.g. foraging, resting), and their environmental drivers (e.g. food abundance, predation).

It has long been recognized that habitat or resource selection could make a major contribution to patterns of animal movement. Movement attributes can be used to identify profitable areas in the landscape (Barraquand & Benhamou 2008) and be integrated into resource selection analysis (Rhodes et al. 2005), while environmental covariates, such as indices of resource abundance, can help explain observed movement patterns of individuals and populations (Avgar et al. 2012; Kuefler et al. 2012). Recently, the integration of resource
selection, mechanistic movement, and home range models has been advocated (Börger, Dalziel, & Fryxell 2008; Moorcroft & Barnett 2008; Schick et al. 2008).

Here I present a mechanistic state-space model providing a flexible, fully parametric formulation of the process of animal movement through heterogeneous environments. Animal movement is modeled as a stochastic relocation process motivated by resource selection – through attraction to or repulsion from local habitats. The main novelty of my approach is in replacing the concept of a discrete latent ‘behavioral state’ by that of a continuous latent ‘informational state’ (sensu Blumstein & Bouskila 1996) that is explicitly modeled as function of the animal’s position in space and time. While my modeling framework enables a straightforward incorporation of behavioral or internal states, my focus here is on modeling/inferring what the individual knows about its environment as it travels through it.

The study of animal behavior is, in many cases, the study of animal decision making. Behavior is the observed manifestation of a complex array of processes occurring in the nervous system, commonly referred to as cognition (Real 1993). Cognitive processes include the perception of external sensory information, the reduction and manipulation of this information, and its integration to form a subjective representation of the organism’s environment (Shettleworth 2001). This representation is the determinant of organismal behavior and its understanding is obviously central to the study of animal behavior, yet it is rarely incorporated into behavioral ecology studies (Real 1993; Dukas 1998; Shettleworth 2001). Multiple theoretical studies have considered the effects of cognitive abilities on animal movement and space-use patterns (Benhamou 1997; Farnsworth & Beecham 1999; Beecham 2001; Vuilleumier & Perrin 2006; Mueller & Fagan 2008; Van Moorter et al. 2009; Berger-Tal & Avgar 2012). Cognitive functions and abilities are, however, difficult to measure in a laboratory setting and are never observed directly in the field. As a result, cognition is rarely considered in empirical studies of animal movement (some exceptions can be found in (Dalziel, Morales, & Fryxell 2008; Selonen & Hanski 2009; MacIver, Patankar, & Shirkgaonkar 2010). Whereas many animals are presumed to move in response to environmental factors, we rarely have two key pieces of the puzzle: what is the information the animal possesses about its environment, and how is that information incorporated in the decision-making process (Blumstein & Bouskila 1996; Lima & Zollner 1996). I propose to bridge this gap by explicitly modeling the informational state of the
animal using a flexible cognitive algorithm that can be empirically parameterized based on readily available positional (i.e., telemetry) and environmental data.

After describing my model, I demonstrate its utility in dealing with scale discrepancies, a major complication in identifying behavioral or cognitive processes from a sample of spatial and/or temporal data. If animals perceive and respond to different stimuli at different scales, biological processes might give rise to patterns that are only discernible at specific temporal and spatial scales (Levin 1992). Movement data almost always comprise a discrete set of snapshots of animal locations over time. The temporal scale of these snapshots may not agree with the scale at which movement decisions are actually made (Codling & Plank 2011). Moreover, the spatial scale at which animals perceive and respond to their environment may differ from the scale at which environmental covariates are measured and mapped (Hebblewhite & Haydon 2010). Similar arguments can be made regarding the spatial resolution of movement data or the temporal resolution of temporally dynamic environmental variables. Our ignorance of the true scale of decision making processes may lead to misidentification of key variables and biased parameter estimates.

Despite being a potential source of error, the effects of scale discrepancies on parameter estimations and predictive abilities are seldom tested (some exceptions can be found in Pinaud 2008; Baasch et al. 2010). I propose that the problem of scale can be dealt with through the use of flexible scaling functions within the movement model itself, thus allowing simultaneous estimations of movement and resource selection at the appropriate scales. I test this premise using factorial sampling of a large set of simulated data at different spatial and temporal scales.

**Methods**

Two main topics are presented. First, a detailed explanation of the model and its underlying rational and assumptions is provided. This includes sections discussing the utility, assessment and parameterization of movement simulation models and a demonstration of space use patterns simulated by my model using extreme cognitive scenarios. Second, the details of a simulation experiment designed to test the parameterization’s sensitivity to sampling resolution are provided.
Modeling Framework

In my framework, the movement process is modeled as a locally-biased random walk (BRW) in discrete space and time (Farnsworth & Beecham 1999). At each time step, the model generates a truncated redistribution kernel summarizing the animal’s probability of occurrence in each of the available spatial cells at the next time step. This redistribution kernel is a function of the subjective landscape – a representation of the landscape as it is perceived by the animal at that particular point in space and time (Dalziel et al. 2008; Christ, Ver Hoef, & Zimmerman 2008; Schick et al. 2008; Getz & Saltz 2008; Hooten et al. 2010).

The Landscape

Landscape layers can be classified into two types: vector data, representing discrete features (e.g., points, lines, or polygons), and raster data, representing a regular grid of cells. Raster cells are characterized by a single value of at least one habitat attribute (either continuous or discrete). In my framework, the landscape is modeled as a heterogeneous hexagonal grid of cells. Such a grid is technically advantageous compared to the commonly used rectangular grid (Birch, Oom, & Beecham 2007) and is also consistent with the neurological architecture of spatial orientation in the mammalian brain (Hafting et al. 2005). Each cell is characterized by its spatial coordinates and the values of several habitat components, which may be either binary (i.e., presence-absence) or continuous. Continuous habitat components are scaled in the range [0, 1] and are assumed to have a monotonic attractive/repulsive effect on the animal. For example, local food availability has a positive monotonic effect (the more food there is at a site, the more attracted the animal is to that location), whereas local predation risk has a negative monotonic effect (the more risky a site is, the more repelled the animal is from that location). The landscape can be either simulated or imported from a geographical information system (GIS). The scale of the hexagonal grid (i.e., the size of each cell) is determined by the spatial resolution of the available landscape data, as well as by the temporal scale of the movement process and the corresponding rate of displacement at that temporal scale.
The Redistribution Kernel

A redistribution kernel is a probability density function characterizing the likelihood of an individual occurring at each particular point in space and time. Here I use a mechanistic cognitive algorithm to calculate the redistribution kernel within each spatial cell after each temporal step (Fig. 1). Non-random animal movement is characterized by attraction or repulsion to specific landscape features. Here, I recognize that cognitive processes leading to a specific subjective representation of the landscape in the mind of the animal are crucial components of this interaction between the animal and the landscape. As we have no way of knowing the true nature of these processes, I model cognition using a simple, yet flexible, learning algorithm. Note that, the algorithm does not assume any particular type of navigation and both spatial learning and cued learning (sensu chapter 2 in Carew 2000) may be considered as plausible mechanisms.

I assume that an animal assesses the quality, $q$, of surrounding localities based on sensory information which attenuates with distance from the source (i.e., the distance between the animal and the assessed location) but accumulates with time. Hence, the proportion of information perceived over distance $d$ during time $\Delta t$ is modeled as $e^{-\alpha-d/\Delta t}$, where $\alpha (> 0)$ is the sensory attenuation coefficient. I further assume that sensory information is committed to the animal’s memory which decays with time. The proportion of information retained in the memory is thus modeled as $e^{-\beta \Delta t}$, where $\beta (\geq 0)$ is the memory decay coefficient and $\Delta t$ is the time elapsed since the memory was last updated. I further assume that whatever the animal does not sense, or otherwise remember, is appended by a default expectation, $q^*$, reflecting a context- and time-independent innate belief. Such default expectations relative to true values across the landscape (i.e. bias), may have substantial adaptive consequences (Berger-Tal & Avgar 2012).

Exponential decay functions were employed for several reasons. First, the intensity of a signal transferred through any channel decays exponentially with distance (Dusenbery 1992). Moreover, ample empirical evidence suggests that spatial memory decays exponentially with time (Panakhova, Buresova, & Bures 1984). Second, the attenuation of information with either space or time allows the animal to focus its attention on proximate objects and is likely to evolve as an adaptive cognitive trait due to limited capacity to processes and retain perceived information (Dukas 1998; Farnsworth & Beecham 1999). Lastly, the exponential forms can be
readily integrated over time, allowing straightforward incorporation of variable time steps (Δt) in the model. Hence, in principle, my model can be fitted to positional data sets with temporal gaps or variable time steps, as often occur with field telemetry data (see ‘Discussion’ for more details).

Based on these assumptions, I derive the following recursion equation for the perceived habitat quality, \(q_{i,j,t}\), with respect to habitat component \(i\), in spatial cell \(j\), at time \(t\), as function of the previously perceived habitat quality at that cell, \(q_{i,j,t-\Delta t}\):

\[
q_{i,j,t} = e^{-\alpha d_{j,t}/\Delta t} Q_{i,j,t} + (1 - e^{-\alpha d_{j,t}/\Delta t})(e^{-\beta \Delta t} q_{i,j,t-\Delta t} + (1 - e^{-\beta \Delta t})q^*_i),
\]

Eq. 1

where \(d_{j,t}\) is the distance between the animal’s current location and cell \(j\), and \(Q_{i,j,t} (\geq 0)\) is the true habitat quality within cell \(j\) at time \(t\).

For example, imagine that an animal is standing at location \(j\). Since \(d_{j,t} = 0\), the animal is fully informed about the quality of location \(j\) \((q_{i,j,t} = Q_{i,j,t})\). As the animal moves away from \(j\), however, it perceives less and less information about \(j\) and relies more on its memory. If true habitat quality changes over time, current sensory information of \(Q_{i,j,t}\), is complimented by memorized information of \(Q_{i,j,t-\Delta t} (\neq Q_{i,j,t})\) and its preceding values. Consequently, memory always lags behind dynamic variation in habitat quality. Over extended intervals with little sensory input, memory decays, approaching the default expectation \((q_{i,j,t} \approx q^*_i)\).

Perceived landscape values \((q_{i,j,t})\) are exponentiated by an attraction coefficient, \(\omega_i\), which either amplifies \((|\omega_i| > 1)\) or reduces \((|\omega_i| < 1)\) the magnitude of perceived differences between different locations. Negative \(\omega_i\) values result in a repulsive effect whereas positive values result in an attractive effect. The magnitude of \(\omega_i\) determines the magnitude of the effect (a null value reflects indifference to the \(i^{th}\) habitat component). In a sense, \(\omega_i\) is the cognitive analog of an animal’s resource selection function (Boyce & Mcdonald, 1999).

This cognitive algorithm is designed to accommodate multiple habitat components \((i = 1:n)\) with varying degrees of attraction or repulsion. At each time step, Eq. 1 is applied independently to each habitat component, thus generating multiple perceived landscape layers. Different perceived landscape layers are then exponentiated by their respective attraction
coefficients and combined via multiplication (see Eq. 2 below). However, the attractiveness of
different locations in the landscape, $A_{j,t}$, should depend not only on the perceived habitat quality
at each locality but also on traveling propensity, modeled here as an exponential function of a
friction coefficient, $\gamma$ ($\geq 0$; a function of morphological and physiological constraints as well as
habitat permeability), multiplied by distance $d_{j,t}$ and divided by the step duration $\Delta t$. The
attractiveness of location $j$, of $n$ habitat qualities, $Q_{i=1,t}$ to $Q_{i=n,t}$, at time $t$, is accordingly
calculated by

$$A_{j,t} = e^{-\gamma d_{j,t}/\Delta t} \prod_{i=1}^{n} q_{i,j,t}^{\omega_{i}}.$$  

Eq. 2

Hence, the attractiveness of any location is a product of the propensity to travel the required
distance during the available time (ranging between 0 and 1) and perceived habitat qualities at
that location, modified by their respective attraction coefficients. The resulting map is the
subjective landscape – the landscape as it is viewed from the animal’s perspective at a particular
point in space and time.

Finally, the redistribution kernel can be calculated based on the subjective landscape. To
reduce the number of calculations needed to produce the redistribution kernel, I arbitrarily
truncate the kernel wherever traveling propensity drops below 1% (i.e., where $e^{-\gamma d_{j,t}/\Delta t} < 0.01$).
The probability of the animal occurring at location $j$ at time $t$ is thus given by the truncated
redistribution kernel:

$$p_{j,t} = \frac{\mathcal{I}(d_{j,t} \leq d_{\text{max}})A_{j,t}}{\sum_{j=1}^{\infty} \mathcal{I}(d_{j,t} \leq d_{\text{max}})A_{j,t}}$$  

Eq. 3

where $\mathcal{I}$ is an indicator function (having the value 1 when the argument is true and 0 when it is
false) and $d_{\text{max}}$ is given by $-\Delta t \ln(0.01)/\gamma$. The redistribution kernel provides us with the likelihood
of observing an animal in a specific spatial cell given a specific landscape grid, parameter values,
and the relocation trajectory to this point (Fig. 1).

Simulating Movement

I model animal movement behavior as a series of discrete decisions, some resulting in
relocation. At each time step, the truncated redistribution kernel is recalculated based on Eqs. 1-3
and is then used as a decision function (sensu Getz & Saltz 2008). The decision rule may be
deterministic (i.e., always move to the cell where $p_{j,t}$ is at its maximal value), or probabilistic (i.e., stochastically move to any cell with probability $p_{j,t}$; Getz & Saltz 2008). Note that, in the probabilistic case, the magnitudes of the attraction coefficients ($|\omega_i|$) influence the level of stochasticity in a manner similar to the randomness parameter used by Farnsworth & Beecham (1999). If any $|\omega_i|$ is very large, the maximal value of $p_{j,t}$ approaches 1 and movement is highly deterministic, whereas if all $|\omega_i|$ are very small the redistribution kernel ‘flattens-out’ and movement decisions are highly stochastic (Getz & Saltz 2008).

Here I use simulations to demonstrate how the model behaves under different cognition scenarios. Four trajectories were simulated across the same hexagonal landscape grid (cell size = 8660 spatial units; landscape size = 4·10^6 cells), characterized by patchy habitat quality, ranging in value between 0 and 1 (Fig. 2). Simulated animals were initially placed in the center of the landscape and their perceived landscape layer (i.e., the map of the habitat initially stored in their memory) was set to their unbiased default expectation ($q_{t=0} = q^* = 0.5$). Trajectories were simulated based on a probabilistic decision rule. The location at the next time step ($\Delta t = 1$ temporal units) was determined according to the current redistribution kernel by calculating the cumulative sum across the kernel, and relocating into the first cell with cumulative sum of probabilities exceeding a pseudo-random number (= $U(0,1)$). Simulated animals had limited traveling propensity ($\gamma = 0.01$) and were strongly attracted to localities with high habitat quality ($\omega = 10$).

The four simulated animals differed, however, in their cognitive ability, reflecting four extreme cognitive types captured by my model (Fig. 2). The first trajectory belongs to a random walker. Perceptual ability was set to 0 ($\alpha = \infty$) and the animal was indifferent to habitat quality quality (note that memory abilities are irrelevant when there is no sensory input). Whereas path-tortuosity varies substantially along this simulated trajectory, it bears no relation to the underlying habitat variation across the landscape (Fig. 2a). The second trajectory belongs to an animal with limited sensory perception ($\alpha = 0.01$) and no memory ($\beta = \infty$). Here avoidance of low quality areas is evident, but the movement is still fairly diffuse with only weak habitat association (Fig. 2b). The third trajectory belongs to an animal with limited sensory perception ($\alpha = 0.01$) and perfect memory ($\beta = 0$). Shortly after initiating its movement, the animal is ‘trapped’ in a (locally) high-quality region and spends the rest of the simulation confined to that
region (Fig. 2c). The fourth trajectory belongs to an animal with limitless sensory perception ($\alpha = 0$). Obtaining full information about its environment allowed this animal to travel efficiently to the highest quality area in the landscape (Fig. 2d).

**Characterizing Movement Trajectories**

Movement trajectories result from a complex mixture of deterministic and stochastic processes that are challenging to quantify and compare. Many different summary statistics have been used to characterize movement trajectories, including the frequency distribution of displacement (Dalziel et al. 2008) the mean squared displacement (Nouvellet, Bacon, & Waxman 2009) and various measures of path tortuosity (Benhamou 2004). Additional complications are added once we start to consider the attributes of the landscape and its possible effects on the movement trajectory. Such interactions are commonly assessed using resource selection functions (Boyce & Mcdonald 1999), based on selective space-usage (i.e., occurrence), but may also be addressed by quantifying the effects of habitat on animal movement parameters (Avgar et al. 2012). When assessing movement model performance, such summary statistics can be used as descriptive probes (Dalziel et al. 2008) allowing one to compare the predictive abilities of a given set of models. In a Bayesian framework, such summary statistics may be used in calculating formal goodness-of-fit measures via posterior predictive assessments of the model (Gelman, Meng, & Stern 1996)

**Fitting Complex Models**

Stochastic simulation models, like the one presented above, are notoriously difficult to parameterize. Due to non-linear interactions between multiple processes (some of which might be hidden), likelihood assessment is often intractable in stochastic simulation models, so parameterization must be based on likelihood-free methods (Hartig et al. 2011). In their simplest form, likelihood-free parameterizations rely on target functions derived from the divergence between patterns generated by the model and observed patterns. Patterns are quantified using summary statistics, such as those presented in the previous section. Note that, inference based on such methods may be somewhat limited by sensitivity to the information content of the specific summary statistics employed (Robert et al. 2011). If on the other hand, the likelihood of data given the model’s parameters can be explicitly calculated, this likelihood serves as the target
function for the parameterization process. Once a target function is defined, different optimization algorithms (e.g., simulated annealing, genetic algorithms, stochastic tunneling, etc.) can be used to search the parameter space for target-maximizing parameter values.

Modeling the redistribution kernel, as done here, provides a straightforward derivation of the likelihood of observed movement data given landscape data and a specific set of parameters. The kernel specifies the probability of the animal occurring at its observed location as function of its trajectory thus far. By taking the product of these probabilities along the entire trajectory, I obtain the likelihood of the trajectory given the habitat layers and parameter values used.

**Simulation Experiments**

I used simulation experiments to evaluate my ability both to estimate the parameters of the model and to capture fundamental aspects of animal movement as function of the spatiotemporal resolution of the data employed in the parameterization process.

*Simulating Trajectories*

Simulated trajectories were stochastically generated across simulated landscapes by iteratively sampling from the multinomial distribution corresponding to the truncated redistribution kernel at each step (see ‘Simulating Movement’ for details). Landscapes were large enough so that simulated trajectories never approached the edge of the landscape, so there was no need to consider boundary conditions. Simulated landscapes were composed of two temporally-stationary independent habitat layers, one representing ‘forage abundance’ \( Q_1 \) and the other representing ‘predation risk’ \( Q_2 \). Each of these layers was characterized by mild spatial autocorrelation (Moran’s I = 0.3), generated by exponentially weighted spatial averaging. The values for each habitat layer varied between zero (corresponding to complete absence of forage or insignificant risk of predation) and one (corresponding to high forage abundance or high predation risk). The attraction coefficients affiliated with each one of these habitat components were accordingly positive for the ‘forage abundance’ layer \( \omega_1 = 2 \); reflecting attraction) and negative for the ‘predation risk’ layer \( \omega_2 = -2 \); reflecting repulsion). All simulation parameter values are reported in Table 1. Twenty independent trajectories, each 8760 time steps long, were simulated, each on an independent landscape realization (for an example of one such trajectory see Fig. 3). To maintain similarity to field positional data, I omitted the first
half of my simulated data (4376 time steps) so as to mimic having some unknown historical space-use patterns preceding each data set.

Resampling Trajectories

Positional data collected in the field are typically samples of the true trajectory of the animal and their temporal resolution is undoubtedly much coarser than the true behavioral process. Similarly, the spatial resolution of available landscape rasters might well be coarser than the true grain to which animals respond. To test the sensitivity of my parameter estimations to the temporal sampling scale, I sampled the simulated movement trajectories at five different temporal scales (x1, x2, x4, x8, and x16) with 1 being the true time scale of the modeled movement process (Δt = 1 temporal units). To test the sensitivity of my parameter estimations to the spatial sampling scale, I aggregated cells across each of my hexagonal gridded landscape realization into five different spatial scales (x1, x2, x4, x8, and x16) with 1 being the true spatial scale of the modeled landscape (cell size = 8660 spatial units). The extremes of these different spatiotemporal scales are illustrated in Fig. 3. Overall I ran 500 parameter estimation procedures: 20 simulated trajectories sampled at five different temporal resolutions and five different spatial resolutions. To equalize sample size across different temporal scales, each of the 500 data sets was reduced to its first 274 relocation points.

One major challenge in estimating perceptual and memory capacities based on an observed space-use pattern is a potential bias resulting from unobserved previous space-usage. We do not know where the animal has been before we initiated our observation and hence cannot infer what it knows. Here I cope with this challenge by using part of the data to initialize the memory within the fitting procedure. Each of the 500 simulated trajectories was evenly divided into a memory build-up set comprising the first 137 relocation points and a parameterization set comprising the last 137 relocation points. The memory build-up set was used to construct the perceived landscape at the end of the memory build-up period (i.e. at the end of the first 137 steps). Perceived landscape layers (one for each habitat component) were initialized with the values of their corresponding default habitat quality (q*), and then updated iteratively for the first 137 steps based on the cognitive algorithm (without calculating the redistribution kernel). The likelihood of the entire parameter set was then calculated based on the 137 redistribution
kernels of the parameterization set (affected by the information perceived and retained during the memory build-up period). Note that, application of my model to dispersing or introduced individuals (i.e., naive individuals with no previous experience with the landscape), would require no memory build-up, so parameterization could be based on the entire dataset.

Parameterization

I chose to demonstrate the likelihood-based parameterization of my model (see ‘Fitting Complex Models’) within a Bayesian framework (Gelman et al. 2004) with a Markov-chain Monte-Carlo (MCMC) procedure being used to obtain samples from the multivariate posterior distribution (Link et al. 2002). A Bayesian MCMC framework was found to be particularly useful because of its flexibility in defining informative priors that can serve as null models (see details below). Nonetheless, any global optimization algorithm capable of operating on complex multidimensional target function surfaces could in principle be employed to estimate parameters for our model. To efficiently explore the model state-space, we employed a random walk Metropolis-Hastings sampler with block updates. A Metropolis-Hastings sampler obtains a sequence of random samples from an unknown distribution by iteratively proposing new values and stochastically accepting them based on their posterior likelihood (Gelman et al. 2004). Starting from a set of initial parameter values (typically the medians of their priors – see below), a proposed value was drawn from the vicinity of the set by adding residual parameter deviates drawn from a multivariate zero-centered normal distribution and acceptance for the set was determined stochastically according to the posterior ratio (Link et al. 2002). The initial variance-covariance matrix for the proposal distribution was populated based on a preliminary MCMC process (see MATLAB code in Appendix 1). This procedure was repeated until a chain 20,000 values long had been generated. Geweke’s convergence diagnostics (Geweke, 1992; MATLAB code written by James P. LeSage) was then used to determine whether the chain is stationary. If there was evidence that the chain did not converge, a new chain was generated starting from the last values in the previous chain. However, in this new chain, the multivariate normal residuals employed in the Metropolis-Hastings sampler (i.e., the proposal distribution) were generated based on the variance-covariance matrix of the previously obtained posterior chain. In this way, correlation of parameters could be accounted for, leading to a more efficient MCMC process.
Once the convergence criterion had been met, the chain was considered stationary and assumed to represent an unbiased sample of the multivariate posterior distribution.

To calculate the posterior, a prior distribution must be defined. Here, apriori parameter independence was assumed and various prior distributions were employed. These distributions were used to define parameter range boundaries and null models within the MCMC process (see Fig. 4 for graphical illustrations of prior distributions). I used Beta distributions as priors for the default expectations of habitat quality ($q^*_i \sim Beta(2,2)$) so that the likelihood of $q^*_i$ greater-than-the-maximum or less-than-the-minimum values of the focal habitat component across the entire landscape is null. These prior Beta distributions were weakly informative (mode = mean = 0.5) so that, in the absence of any information in the data, the posterior distribution of $q^*_i$ would be centered on the middle of the range of values for that component. I used normal distributions as informative priors for the attraction coefficients ($\omega_i \sim N(0,5)$) so that, in the absence of any information in the data, the posterior distribution of $\omega_i$ would be centered on 0, reflecting indifference towards the focal habitat component (Christ et al. 2008).

I used informative spike-and-slab priors (Mitchell & Beauchamp 1988) for the sensory attenuation and the memory decay coefficients. Null values of these coefficients imply infinite perception and/or memory. While infinite perception is biologically unreasonable, lifelong memory is perhaps more conceivable. I accordingly set a spike at 0 for the memory decay coefficient prior. Due to the exponential nature of their effect, extremely high values of these parameters (arbitrarily defined here as >1) translate into no perception and/or memory. Both are considered here as valid null models. Hence, both coefficients’ priors have spikes at 1. Note that, for other applications, these upper boundaries might have to be adjusted according to the spatiotemporal scale of the data and the biology of the organism under study. Each spike is assigned a mass twice as big as the slabs, so that, in the absence of information in the data, each of the null models is twice as likely as the fully parameterized process model.

To implement spike-and-slab priors within my parameterization procedure, I used a mixture of uniform distributions (Fig. 4). The random walk Metropolis-Hastings sampler must be allowed to ‘step into’ the spike, which necessitate distributing the spike’s mass over a segment of the parameter space (i.e., replacing the spike with an additional, high-density, slab). High-density spike-representing slabs of various widths (but equal mass) were tested and found to have
indiscernible effects upon resulting posterior distributions. The width of these high-density slabs was therefore adjusted so as to facilitate chain convergence. When parameter values were within the process range (between 0 and 1; the low-density slab), prior probabilities were set at 0.5. Spikes were represented as bounded segments of the parameter space (high-density slabs between -1 and 0 for $\beta$ and between 1 and 2 for both $\alpha$ and $\beta$) where prior probabilities were 1. Otherwise, prior probabilities were set at 0 (Fig. 4):

$$prior(\alpha) = I(0 < \alpha \leq 1)u(0,1)0.5 + I(1 < \alpha \leq 2)u(1,2); \quad \text{Eq. 4}$$

$$prior(\beta) = I(-1 < \beta \leq 0)u(-1,0) + I(0 < \beta \leq 1)u(0,1)0.5 + I(1 < \beta \leq 2)u(1,2). \quad \text{Eq. 5}$$

Note that, whenever $\beta \leq 0$, the memory decay coefficient was assigned a null value in the cognitive algorithm (i.e., lifelong memory) and similarly, whenever $\alpha$ or $\beta$ were larger than 1, $e^{-\alpha}$ or $e^{-\beta}$ were assigned 0 in the cognitive algorithm (i.e., no perception and/or memory). The friction coefficient, $\gamma$, was given a flat bounded prior ($\gamma \sim u(0,1)$).

**Measuring Performance**

Once the MCMC algorithm had converged, the resulting output matrix contained 20,000 samples from each of the seven marginal posterior distributions, one for each of the seven parameters. To assess the goodness-of-fit of these parameter estimates I calculated a standardized measure of the absolute deviation from the true value (i.e., the value used to simulate the trajectory):

$$\frac{SSD_{\text{posterior}}}{SSD_{\text{prior}}}, \quad \text{Eq. 6}$$

where $SSD_{\text{posterior}}$ is the sum of squared differences between the true value and the entire MCMC posterior set for that parameter and $SSD_{\text{prior}}$ is the sum of squared differences between the true value and a 20,000 sample set of the prior distribution for that parameter. I then classified these standardized bias values as positive or negative, according to whether the posterior median was larger or smaller then the true parameter value.

I employed two types of descriptive probes (see ‘Characterizing Movement Trajectories’ above) as independent measurements of space-use patterns emerging from my model. The first probe was the sinuosity, $S$, an unbiased estimator of path tortuosity (Benhamou 2004),
characterizing the movement trajectory independent of the underlying habitat features. For a path composed of \( n \) steps (i.e., \( n + 1 \) spatiotemporal position) of the same duration but different lengths, \( l \), \( S \) is calculated as

\[
S = 2 \sqrt{E(l) \left(1 + \frac{1}{1-c} + \frac{E(l^2)}{E(l)^2} - 1\right)}.
\]

Eq. 7

Here \( E(l) \) is the expected value of \( l \) and \( c \) is the \( l \)-weighted directional correlation of the path:

\[
c = \frac{\sum_{i=1}^{n} l_i \cos(\theta_i-\theta_{i-1})}{\sum_{i=1}^{n} l_i},
\]

Eq. 8

where \( \theta_i \) is the heading of the \( i \)th step. Note that, \( S \) is directly related to other statistical attributes of random paths, such as mean squared displacement and diffusion rate (Benhamou 2004).

The second descriptive probe I employed was the resource selection function (RSF), commonly used in wildlife research to characterize habitat selectivity independent of the movement process. An RSF is a model yielding values that are proportional to the probability of selection of a resource type (Boyce and McDonald 1999). I estimated three RSF coefficients (an intercept and a selection coefficient for each habitat component) for each simulated trajectory by fitting a logistic regression and interpreting the fit as a logistic discriminant function between a distribution of used cells and a distribution of available cells (Johnson et al. 2006). In my logistic regression, cell usage is a binomial response variable, whereas the habitat qualities in available landscape cells (including those used) are the explanatory variables. Cells were considered available if they were located within the minimum convex polygon encompassing all locations in the simulated trajectory. Note that, while several more restrictive definitions of availability have been recently promoted, the minimum convex polygon is still commonly used in RSF studies. I employed it here purely for its simplicity and its lack of assumptions regarding the underlying movement process.

The true values of probes were recorded by first simulating 1000 movement trajectories, using the original true parameter values, across a single independent landscape realization. These trajectories were then resampled at five temporal and five spatial resolutions (as described above) and a sample of 274 relocation points (starting from the midpoint) was used to calculate sinuosity and RSF coefficients, resulting in three probe vectors, each 1000 elements long.
To evaluate the agreement between probe values generated using the true parameters and probe values generated using the estimated parameters, I generated simulated probe distributions. I sampled 50 sets of parameter values out of each of the 20 posterior distributions generated by the parameterization process for each of the 25 spatiotemporal scales. Each parameter set was then used to simulate a movement trajectory across the same landscape, at the appropriate spatiotemporal scale. Thus, for each spatiotemporal scale, I estimated vectors of 1000 values of sinuosity and RSF coefficients (based on a sample of 274 relocation points starting midpoint). These vectors were then graphically compared to those generated by the true parameter values to allow posterior predictive assessment (Gelman et al. 1996) of my model at the different spatiotemporal scales.

Results

The MCMC algorithm performed well, with 96% of parameterization procedures reaching full convergence, resulting in a stationary 20,000 values-long posterior chain, within 10 iterations (an arbitrarily set cutoff) of adapting the proposal distribution. The frequency of convergence failure increased with the spatial scale of sampling and was most common (20%) at the coarsest spatial and finest temporal scales. Parameterizations that failed to converge were excluded from any further analysis.

Parameter Estimations

Posterior distributions obtained through my parameterization process were mostly unimodal (e.g., Fig. 5). At the original temporal and spatial scales (x1), posterior parameter estimates showed little bias (Fig. 5 and Table 2) for all parameters except for the memory decay coefficient, which often converged to the null model of no memory decay (see below). Comparing standardized bias values between parameters (Table 2) revealed that the estimates of some parameters were highly accurate (the friction and sensory attenuation coefficients) whereas others (the memory decay coefficient and default habitat quality) were less so. The standardized biases are lower for parameters affiliated with predation risk (repulsion) than for those affiliated with forage abundance (attraction; see below). Multivariate bias (the average of the standardized absolute bias of all parameters over all realizations) increased with spatial scale, but showed little sensitivity to the temporal scale of sampling (Fig. 6).
I defined failure to detect an effect of a habitat component when the 95% percentiles of the posterior distribution of its attraction coefficient overlapped zero (for example, see the posterior distribution for the attraction coefficient of the forage abundance layer, $\omega_1$, under coarse sampling scales in Fig. 5). Failure to detect any environmental effects implies convergence to the null model (a pure random walk). The frequency of such failures ranged from 0 at fine spatial scales, to 45% at the coarsest spatial and finest temporal scales. At coarse spatial scales convergence to the null model occurred less frequently as the temporal scale increased. Out of all parameterizations converging to a locally biased random walk (meaning at least one environmental effect is detected), failure to detect the effect of the predation risk layer (repulsion) rarely occurred and only did when both temporal and spatial scales were coarse. Failure to detect an effect of the forage abundance layer (attraction) was more common (0-60%) and increased with spatial scale, with no clear effect of the temporal scale. Note that, a negative attraction coefficient ($\omega_2$) has a greater effect, and is hence more likely to be accurately detected, than a positive attraction coefficient ($\omega_1$) of the same magnitude. This is due to the increased spatial variability in habitat attractiveness caused by the latter being an inverse function of the habitat component’s level (Jensen’s inequality; Ruel & Ayres 1999).

Prevalence of extreme posterior values of the memory decay coefficient indicates convergence to one of two extreme null models, represented by attractors (i.e., spikes) at the boundaries of its prior distribution (see ‘Parameterization’ above and Fig. 4). Whenever $\beta \leq 0$, the memory decay coefficient within the cognitive algorithm is set to 0 and memory never decays. On the other hand, if the memory decay coefficient is very large (defined here as $\beta > 1$), the exponent of the memory decay coefficient within the cognitive algorithm is set to zero and no information is retained in the memory – movement decisions are made based on current sensory perceptions of the landscape. For parameterizations converging to a locally-biased random walk (i.e., at least one environmental effect is detected; $\beta$ always converges to its prior otherwise), frequency of convergence to no-memory-decay ranged from 0 to 40% and decreased with both temporal and spatial scales. The frequency of convergence to no-memory was very low across most temporal and spatial scales but spiked (~70%) at the finest temporal and coarsest spatial scales. The frequency of convergence to the full prior (Fig. 4; indicating not enough information to support any explicit memory model) was quite high (30% on average) but did not display any clear trend with the spatial or temporal scales of sampling.
Descriptive Probes

Descriptive probes reflected an overall high predictive ability of my model at all spatial and temporal scales of sampling (Figs. 7 and 8). Estimation of path sinuosity was little affected by the spatial scale of sampling and matched well with validation values (those measured on paths simulated using the true parameter values). The only exception was a slight tendency to underestimate sinuosity at coarse spatial scales. The temporal scale of sampling did have a marked effect on sinuosity values where departures from the validation distribution increased with scale. At the coarsest temporal scale all estimated distributions showed strong departures (overestimations) from the validation distribution, regardless of the spatial scale of sampling (Fig. 7).

Estimations of resource selection were more variable than the corresponding validation distributions but were otherwise unbiased under most temporal and spatial scales of sampling (Fig. 8 and Appendix G). Marked departures between the estimated and validation distribution occurred only under the coarsest spatial scale and only at fine temporal scales of sampling. Once temporal resolutions became coarse enough, these departures disappeared.

Discussion

The framework presented here is based on a rarely used approach in the study of animal movement – parameterization of an explicit model of decision-making relating an animal’s position at any given time to its past and current perception of the landscape. Recent studies, at the forefront of movement data analysis, have focused on quantifying the properties of the sampled paths as a correlated random walk, delineating discrete behavioral states based on the walk’s attributes, and linking these behaviors to underlying environmental conditions (Jonsen, Myers, & Flemming 2003; Morales et al. 2004; Forester et al. 2007; Gurarie, Andrews, & Laidre 2009). However, animals may not just respond to local conditions but also orient their movements towards attractors in their surroundings (Benhamou 2006), and hence perform a locally-biased random walk (Mueller & Fagan 2008; Getz & Saltz 2008). Modeling locally-biased random walks requires consideration of the information available to the animal at any point in space and time and how that information is integrated in the decision-making process (Blumstein & Bouskila 1996; Lima & Zollner 1996). Hence, it requires explicit modeling of
cognitive aspects of the movement process, such as sensory perception, memory and prioritization. To my knowledge, mine is the first demonstration of how these cognitive processes could be estimated from routinely collected positional data.

**The Redistribution Kernel**

My framework relies on calculating the redistribution kernel at each point in space and time. Analytically tractable redistribution kernels are often employed in the study of population spread and invasion ecology (Kot, Lewis, & van den Driessche 1996). More recently, redistribution kernels have been applied to individual movement behavior (Dalziel et al. 2008; Schick et al. 2008). Redistribution kernels enable straightforward integration of multiple landscape features into a likelihood assessment of a given model based on observed relocation data. Direct linkage of probabilistic measures of animal occurrence with mechanistic drivers of the relocation process (foraging, predator avoidance, spatial memory and perception, territoriality, etc.) helps unify different aspects of animal space use: home range behavior, movement behavior and resource selection analysis (Börger et al. 2008; Mueller & Fagan 2008; Moorcroft & Barnett 2008; Schick et al. 2008).

Simulating movement trajectories based on the redistribution kernel allows exploration of the consequences of different model parameters values, landscape characteristics, and their interactions (e.g., Fig. 2). Such exploration may focus on emerging properties of the resulting movement or space-use patterns (see ‘Characterizing movement trajectories’), or on inter- and intra-specific interactions. For example, one can test whether cognitive abilities are reflected in statistical properties of the movement path or quantify the effects of different landscape configurations on predator-prey encounter rate as function of their cognitive abilities. Moreover, by defining an appropriate fitness function, simulations based on my model could be used to explore the evolution of cognitive abilities and limitations or to predict population dynamics. For example, one may ask how the optimal memory decay rate responds to spatiotemporal environmental dynamics or what is the invasability of a population across a specific landscape as function of the individuals’ cognitive abilities and limitations.

**Memory**
The majority of studies considering animal movement and cognition to date assumed that the effect of memory on the animal’s attraction to a landscape locality does not depend on the quality of this locality (see Smouse et al. 2010 and references therein). My approach is fundamentally different as I explicitly consider the informational state of the animal. Hence, in my model, a previously perceived and memorized locality serves as an attractor only if it is of relatively high quality; if it is of relatively low quality it will repel the animal rather than attract it. That said, my mechanistic model can easily accommodate simple memory-based drivers (independent of the underlying habitat quality) by including a homogeneous landscape layer in the simulation/fit process (where $Q$ has the same value over all $j$’s and $t$’s). If the expectation of habitat quality for this pseudo habitat component is low ($q^* < Q$) and the attraction coefficient is positive ($\omega > 0$; or vice versa) the animal is attracted to memorized localities simply because they are memorized. If on the other hand the expectation of habitat quality is low and the attraction coefficient is negative ($\omega < 0$; or vice versa) the animal is repelled by memorized localities. Such avoidance of previously visited localities may be expected due to predator avoidance (if probability of detection increases with usage) or resource depletion (Van Moorter et al., 2009). Hence, my model includes previous modeling perspectives of animal movement and memory within a more general framework and enables straightforward comparisons of their utility.

From a theoretical standpoint, my formulation provides a fresh perspective on the adaptive value of memory. In a temporally invariant environment memory capacity should be limited solely by its physiological or developmental costs (e.g., the metabolic cost of maintaining a large brain). However, if the environment is temporally dynamic, due e.g. to resource phenology or depletion and regeneration, some memory decay may be advantageous regardless of metabolic costs. If memorized information does not reflect the current state of the landscape, forgetting may reduce harmful bias due to previous experience. Hence, memory capacity should, to some extent, be negatively correlated with the rate of environmental change. Moreover, if different habitat components have different rates of environmental change, it is reasonable to expect different components-specific memory decay rates (which can easily be incorporated in my framework; $\beta_i$).

**Model Selection**
Model competition, the relative ranking of alternative models based on some measure of goodness-of-fit, is increasingly favored as a means of hypothesis evaluation in ecology and evolution. This approach requires fitting different models (reflecting different biological hypotheses) to the same data and then ranking competing models based on how well they fit the data relative to the number of parameters. Such rankings are typically based on information theory, penalizing for model complexity. A Bayesian alternative to the information theoretical approach is the deviance information criterion (Plummer 2008).

Reversible-jump MCMC, used within a Bayesian framework, allows for simultaneous model parameterization and selection by exploring the joint posterior model and parameter space, and thus giving estimates of model posterior probabilities (e.g., Pagel and Meade, 2006). However, such schemes can be extremely computationally intensive. The parameterization scheme presented here allows us to choose between models in a far less computationally burdensome manner.

In the slab-and-spike method used here, model competition is an integral part of the parameter estimation process and multiple hypotheses are assessed simultaneously within a single framework. Instead of penalizing for model complexity post-facto (as in the information criterion approach), I use informative prior distributions to reflect the null expectations for each parameter. Hence, if the data fail to support a specific hypothesis, the estimate of the parameter affiliated with that hypothesis converges to the prior distribution. The strength of the prior reflects the degree of conservatism regarding the hypothesized process. While a strong prior would act to support the null hypothesis if information is lacking (Christ et al. 2008), it would also bias the posterior and increase its variance, despite evidence of support. In essence, the posterior ratios function as Bayes’ factors (Kass & Raftery 1995), evaluating the weight of evidence in favor of alternative hypotheses nested within the parameterization process itself. While my approach does not constitute a formal model comparison, and is more subjective than a full reversible-jump MCMC exploration of the parameter and model posterior, it is far less computationally demanding and offers a simple and intuitive way of choosing between models. The resulting posterior distributions reflect the tension between the information encoded in the data and the appropriate null hypothesis, the latter apriori weighted by the user.

**Scale Sensitivity**
I have investigated the sensitivity of my parameter estimations and model predictions to sampling scale. My numerical experiments demonstrate the adequacy of my modeling approach for inferring resource selectivity and cognitive capabilities, based on a limited set of telemetry data sampled at different spatial and temporal scales. Despite a clear increase in parameter estimation bias with spatial scale of sampling, including an increased frequency of null model convergence, this had little effect on the descriptive probes. Resource selection patterns predicted by my model were robust to both temporal and spatial scales of data sampling, while predicted patterns of path sinuosity were sensitive only to the temporal scale of sampling. This suggests that parameter estimates and model predictions may substantially differ in their sensitivity to temporal and spatial scales of sampling.

I believe the robustness of the model’s predictions to the temporal scale of sampling also may reflect its ability to appropriately deal with temporal gaps in the telemetry data. Changes in movement attributes (e.g., speed, directional persistence, orientation components, etc.) are inherently non-linear with respect to time, making them incomparable when measured over different time intervals (Gurarie et al. 2009). Yet, most relocation data have gaps, due to either sampling design (i.e., more intense sampling during specific seasons or time of day) or technical failure to obtain some of the positional data.

Gaps can be dealt with by either resampling, thus losing part of the data, or by data interpolation, using anything from simple discrete random walks to continuous-time random walks and sophisticated state-space models (Johnson et al. 2008). Further, in a Bayesian framework the parameter vector can be augmented to include missing information as parameters to be estimated (e.g., O’Hara et al. 2002), although this so-called data-augmented MCMC approach would greatly increase the computational burden. Alternatively, interpolation can be avoided by explicitly considering, within the model, the autocorrelation in the movement time-series (Gurarie et al. 2009). I extend this latter approach by employing exponential decay functions for several components of the movement process. While not directly tested here, my model’s accurate predictions of space use patterns based on coarsely sampled positional data may testify to the potential usefulness of such an approach in dealing with temporal gaps.

Concluding Remarks
Resource selection functions are a fundamental ecological tool (Boyce & Mcdonald 1999). Appropriate assessment of resource selection should take into account resource accessibility (Matthiopoulos 2003; Rhodes et al. 2005; Moorcroft & Barnett 2008) as well as individual variation (Duchesne, Fortin, & Courbin 2010). These issues may be resolved using individual based models and an integration of the movement and resource selection processes (Moorcroft & Barnett 2008; Schick et al. 2008; Fieberg et al. 2010). I further argue that an additional crucial component is an explicit model of information perceived and memorized by the animal at each point in space and time. My resource selection estimates are robust to temporal and spatial scales of sampling as well as to the details of landscape configuration. Such methodology has the potential to allow prediction of resource selection patterns in novel landscapes.

My model relies on raw relocation data, rather than summary statistics of the path. Relocation data sets are typically multidimensional (two to three spatial dimensions and one temporal dimension) and are thus complex, especially when coupled with underlying environmental conditions. Movement studies are often aimed at reducing the complexity of the relocation time series into simple summary statistics (Nouvellet et al. 2009). I take a different tack aimed at extracting the maximum information out of the available data by using the actual relocation time series as model input. Hence, in my model, the only information loss occurs in the assignment of observed locations in continuous space to discrete grid cells (Hooten et al. 2010). I believe such an approach is necessary for testing explicit mechanistic hypotheses regarding the movement process.

A major limitation of the formulation presented here is the invariant nature of the friction coefficient. Habitat permeability, the ease with which an animal can travel through a specific habitat, may have profound effects on both the choice of destination and the time it takes to reach that destination. Accounting for such effects requires explicit consideration of all possible paths connecting each pair of consecutive positions (e.g., Matthiopoulos 2003) which may be exceptionally computationally demanding and beyond the scope of the current work. Alternatively, spatially explicit effects of habitat permeability may be approximated by modeling the friction coefficient as function of local permeability in the current position or the average permeability across the redistribution range. The propensity to move might also change with time.
of day (reflecting a diurnal activity cycle; Boyce et al. 2010) or year (reflecting behavioral seasonality, such as migration). Such temporal dynamics in the propensity to travel can be easily incorporated into my model using simple trigonometric functions to account for seasonal or diurnal cycles. Similarly, the discrete nature of my model enables straightforward inclusion of functions accounting for additional biological processes, such as resource depletion, directional persistence and/or bias, and temporally dynamic resource preference.

Summary

Neuronal processes are extremely difficult to measure directly, particularly in free-ranging animals. Nevertheless, some reasonable assumptions can be made regarding cognitive abilities and limitations. The capacity of any cognitive system to perceive, store and process information is limited. Finite computational capacity must result in a tradeoff between scope and resolution – an increase in the number of perceived items (or spatial locations) necessitates a decrease in the information regarding each item (Wiens 1989). Pathways of information perception and processing are likely to undergo natural selection that favors the most proficient variants (Dukas 1998). Here I relied on such assumptions in the formulation of a cognitive algorithm that models the decision making process for a moving animal. As any interpretation of the processes underlying animal movement patterns critically depends on understanding of how animals perceive their environment, I believe that my methodology offers a significant step forward in the study of animal movement. It also provides a unique opportunity to combine theoretical questions of animal cognition with empirical questions. By defining an appropriate fitness function, my simulation could be used to explore the adaptive value of different cognitive strategies as function of landscape structure, competition and predation. Several previous studies have focused on theoretical investigations into the evolution of cognitive abilities (e.g., Dukas 1998; Beecham 2001; Dall et al. 2005; Berger-Tal & Avgar 2012). It is currently difficult to test such models, because their parameters cannot be directly assessed from empirical data. My framework bridges this important gap by relying on empirically derived parameters, based on biologically plausible models of cognition.
References


### Tables

**Table 1** - The model’s parameters. True values are those used to simulate trajectories across the 20 different landscape realizations.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Friction</th>
<th>Sensory attenuation</th>
<th>Memory decay</th>
<th>Forage abundance</th>
<th>Predation risk</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\gamma$</td>
<td>$\alpha$</td>
<td>$\beta$</td>
<td>$q^*_1$</td>
<td>$\omega_1$</td>
</tr>
<tr>
<td></td>
<td>$q^*_2$</td>
<td>$\omega_2$</td>
<td></td>
<td>$q^*_2$</td>
<td>$\omega_2$</td>
</tr>
<tr>
<td>Prior</td>
<td>$u(0,1)$</td>
<td>$p(0&lt;\alpha&lt;1) = 0.5$</td>
<td>$p(0&lt;\beta&lt;1) = 0.5$</td>
<td>$Beta(2,2)$</td>
<td>$N(0,5)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$p(1&lt;\alpha&lt;2) = 1$</td>
<td>$p(1&lt;\beta&lt;2) = 1$</td>
<td></td>
<td>$Beta(2,2)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$p(2&lt;\alpha \mid \alpha &lt; 0) = 0$</td>
<td>$p(-1&lt;\beta&lt;0) = 1$</td>
<td>$N(0,5)$</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$p(2&lt;\alpha \mid \alpha &lt; -1) = 0$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>True</td>
<td>0.005</td>
<td>0.005</td>
<td>0.005</td>
<td>0.7</td>
<td>2</td>
</tr>
<tr>
<td>value</td>
<td></td>
<td></td>
<td></td>
<td>0.3</td>
<td>-2</td>
</tr>
</tbody>
</table>
**Table 2** - Mean standardized bias for each parameter at the original temporal and spatial scales (x1, x1). Each value is an average of 20 realizations.

<table>
<thead>
<tr>
<th>Friction ($\gamma$)</th>
<th>Sensory attenuation ($\alpha$)</th>
<th>Memory decay ($\beta$)</th>
<th>Forage abundance</th>
<th>Predation risk</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Default quality ($q^*_1$)</td>
<td>Attraction ($\omega_1$)</td>
</tr>
<tr>
<td>-1 x10^{-6}</td>
<td>3 x10^{-7}</td>
<td>-2 x10^{-1}</td>
<td>-4 x10^{-1}</td>
<td>4 x10^{-2}</td>
</tr>
</tbody>
</table>
Figures

**Figure 1** – A graphical illustration of the translation of different landscape layers (the two upper panels) into a single redistribution kernel (the lowermost panel) in a specific point in space and time (for details see ‘The Redistribution Kernel’). Parameter values are those used for the simulation experiments (see Table 1). The animal is located in the central hexagonal cell. Whereas the upper right side of area plotted was already explored by the animal (the actual trajectory is not plotted for visual clarity), the lower left side is yet unexplored and its perceived quality is thus mostly determined by $q_i$. Note that, the different maps differ in the values represented by the grey scale. The dark cells in the last panel are those most likely to be occupied by the animal at time $t$. 
Cognition (perception, memory, preference)

$$
(e^{-\alpha d_{j,t} / \Delta t} Q_{i,j,t} + (1-e^{-\alpha d_{j,t} / \Delta t}))(e^{-\beta \Delta t} q_{i,j-t,\Delta t} + (1-e^{-\beta \Delta t})q^*_i) ^\omega_i
$$

$$
\frac{q_{i=1,j,t} \omega_{i=1} q_{i=2,j,t} \omega_{i=2} e^{-\gamma d_{j,t} / \Delta t}}{\sum q_{i=1,j,t} \omega_{i=1} q_{i=2,j,t} \omega_{i=2} e^{-\gamma d_{j,t} / \Delta t}}
$$

Redistribution kernel
Figure 2 – A graphical illustration of simulated movement paths across the same patchy landscape (see ‘Simulating Movement’ for details). All trajectories originated from the same position (marked with a star on panel 2d). The landscape’s quality ranges from 0 (white) to 1 (black). The four trajectories belong to simulated animals with: no sensory ability ($\alpha = \infty$; a), limited sensory and no memory abilities ($\alpha = 0.01$, $\beta = \infty$; b), limited sensory and endless memory abilities ($\alpha = 0.01$, $\beta = 0$; c), and limitless sensory ability ($\alpha = 0$; d). Other parameter values used in these simulations were $\omega = 10$ and $\gamma = 0.01$. 
Figure 3 – One of 20 simulated trajectories used for assessing scale-dependent parameterization and prediction (see ‘Simulation Experiments’ for details). The landscapes quality, calculated here as the product of the two habitat components raised to the power of their respective attraction coefficients (i.e., $Q_1^{\omega_1} Q_2^{\omega_2}$), is scaled between 0 (white) and 1 (black). The four panels display the same trajectory and landscape at different spatiotemporal scales: the scales used to simulate the trajectory (cell size ≈ 8660, step duration = 1; a), the spatial scale used to simulate the trajectory with a temporal scale 16 time coarser (cell size ≈ 8660, step duration = 16; b), the temporal scale used to simulate the trajectory with a spatial scale 16 time coarser (cell size ≈ 138564, step duration = 1; c), and both scales 16 times coarser than the original ones (cell size ≈ 138564, step duration = 16; d). Note that, due to spatial averaging, a coarse spatial scale (panels c and d) results in more intermediate habitat quality values.
**Figure 4** - Graphical illustrations of prior distributions. Informative priors allow the parameterization procedure to converge to appropriate null models in the absence of data-supported alternatives (see ‘Parameterization’ for details and Appendix F for code implementation).
**Figure 5** - Marginal posterior distributions for the seven parameters (black and white histograms in the same frame indicate values for the predation risk and forage abundance layers, respectively) for one out of the 20 realizations. The left column reflects estimates under the original temporal and spatial scales ($x_1, x_1$) while the right column reflects estimates at the coarsest temporal and spatial scales ($x_{16}, x_{16}$). Dashed lines indicate parameter values used to simulate the data (i.e., ‘true’ values).
Figure 6 - The mean, multivariate, absolute, standardized bias as function of spatial and temporal scales of sampling. Shades of grey reflect increasing temporal scales (from 1 to 16).
**Figure 7** - Path sinuosity. Each plot represents a kernel density estimate based on 1000 sinuosity values from 1000 path simulated using the true parameter values (solid lines) or 1000 samples out of the joint posterior distributions (dotted lines) at each of the five temporal and five spatial scales.
**Figure 8** - Forage abundance selection coefficient. Each plot represents a kernel density estimate based on 1000 selection coefficient values from 1000 path simulated using the true parameter values (solid lines) or 1000 samples out of the joint posterior distributions (dotted lines) at each of the five temporal and five spatial scales. Dashed gray lines indicate a null value (i.e., apparent indifference to forage abundance). Results for the predation risk selection coefficient are very similar and are presented in Appendix G.
Epilogue

Animals live in spatially and temporally heterogeneous environments where survival and reproduction often necessitate active relocation through space and time. Natural selection would favor movement patterns that improve access to fitness-enhancing resources (food or mates) or avoidance of fitness-compromising situations, whether these are biotic (predation, parasitism or competition) or abiotic (pollutants or weather). This should not imply however, that animal movement is an optimal, fitness-maximizing behavior. Movement decisions (where to move and when to do so) may be critically constrained by the information available to the animal regarding the current and future states of its environment, as well as its own internal state. Moreover, the execution of these decisions is constrained by the movement capacity of the animal and the physical permeability of its surroundings. Animal movement patterns are the observed manifestations of these motivations and constraints and thus offer movement ecologists a potentially rich and rewarding source of information.

Recent years have seen the development of diverse analytical approaches aimed at extracting and interpreting information from observed patterns of animal movement. This rapid development echoes increased availability of advanced and affordable tracking technology leading to the collection of massive amounts of high resolution data. Despite, and in part due to, these rapid advancements, much confusion still exists regarding what can and cannot be learned from the analysis of animal movement data.

In the first chapter of my thesis, I developed a simple null mechanism capable of reproducing multiple naturally-observed phenomena such as decreased movement rates in preferred habitats, selective use of resource rich patches and reduced consumption of mobile, grouped resources. I have shown that patterns often attributed to a decisive behavioral response of consumers may arise as a by-product of statistical movement mechanics, and that rates of diffusion and consumption are inextricably linked through common effects of resource density, heterogeneity and mobility. The main contribution of this work is in highlighting the crucial role of mechanistic null models in defining which patterns are truly indicative of specific behavioral processes.

In the second chapter, I used an extensive data set of telemetry and remote-sensed environmental variables to link patterns of woodland caribou movement to local conditions
experienced by individual animals across a vast landscape. I was successful at capturing much of the variability in caribou movement across space and time, and my results offer some support to the notion that two behaviorally distinct caribou sub-populations inhibit the study region. Interestingly, despite the large sample size and the geographical extent of the data, my statistical models performed rather poorly in predicting the movement rates of an independent set of animals. Hence, whereas much value could come from appropriate statistical modeling of movement data, predicting animal movement may require a more mechanistic consideration of the underlying behavioral processes.

In my last chapter, I developed a novel state-space model of animal movement accounting for sensory perception, memory and decision-making. Unlike previous works investigating the role of cognition in animal movement, my model can be parameterized directly from positional and landscape data. This model demonstrates a promising step forward in our ability to interpret animal space use patterns and understand the drivers of movement behavior across heterogeneous landscapes. Accounting for the capacity to perceive and remember the environment may be crucial for predicting future space use patterns across novel or rapidly changing landscapes. Substantial traction in understanding cognitive abilities and limitations could be made by applying this approach to fine resolution positional and environmental data of relocated animals. Confidence in the naivety of the animal at the onset of the observation would enable stronger inference about the cognitive processes underlying movement behavior. Animal movement data, when coupled with adequate environmental information, could provide an exceptionally rich source of information about the behavioral ecology. My approach offers a powerful tool with which this information could be extracted.

To conclude, my thesis has dealt with various aspects of the interface between moving animals and the heterogeneous landscapes they traverse, a central theme in the emerging field of movement ecology. The wide range of approaches and methodologies I have employed reflects the current breadth of this field of research. I have no doubt that the near future will bring major scientific advances in our understanding of animal movement ecology and, consequently, in our ability to predict animal space use patterns across heterogeneous landscapes.
Supplementary material

Appendix A (Chapter 1)

Diffusion rates of a type I consumer searching for randomly distributed immobile resources, as function of resource density (on a log-log scale). Simulated values (small filled diamonds) were derived by dividing the squared displacements of 10 independent trajectories, each consisting of 1000 consumption events, by 4 times their durations. Simulated values are in good agreement with simulation based mean field approximations (empty squares – also presented in Fig. 1 of the main text) and the predictions of the closed form Scenario I model (dashed line). Note the high variance of the simulated values around the mean field approximations, hence the use of the latter in the main text. Parameter values are: $r = 0.0001$, $h = 1$, $v_{\text{consumer}} = 100$, $v_{\text{resource}} = 0$ (see appendix B for simulation code).
Appendix B (Chapter 1)

Matlab simulation code for Scenarios I and III and Appendix A

```matlab
%% this code simulates distance-to-detection events for a consumer moving through a field of aggregated resources. to simplify the calculations consumers always travel upwards so that detected resources must be within a vertical corridor of width 2*r originating from a random resource (in the case of immobile resources) or a random point (in the case of a type II consumer with mobile resources). The focal arena is a square surrounded by 8 other squares to minimise any edge effects.

clear;
%the distance at which detection (and hence consumption) occurs
effective_radius=0.0001;
sqrd_effective_radius=effective_radius^2;
%the mean density of resources on the domain
density=100;
%the mean clump size determines the level of aggregation where a null value corresponds to a random distribution of resources
clump_size_lambda=50;
%determines the spacing of resources within a clump
dispersal_std=0.01;
%parameters used in the calculation of consumption and diffusion rates
customer_speed=100;
handling_time=1;
%the number of independently simulated steps
repetitiones=10000;
%these arrays will contain the results
point_distances=NaN(repetitiones,1);
event_distances=NaN(repetitiones,1);

for n=1:repetitiones
    %initialising Matlab's random number generator
    rand('twister', sum(100*clock));
    %generate the landscape
    %passive_events are events located in and around the central focal arena (1<(x,y)<2) and are treated as resources.
    %active_events are the subset of these that are potential starting points for the consumer
    active_events_number=0;
    while ~active_events_number
        %the function generate_NeymanScott_events returns an array of the x and y locations of points scattered between 0 and 3
        [events heritability]
        highest_heritable_source]=generate_NeymanScott_events(density,clump_size_lambda,dispersal_std,effective_radius);
        %out of these events, only a subset can be detected by a consumer located somewhere in the focal domain
```

102
passive_participants=(events(:,1)>(1-effective_radius))&(events(:,1)<=(2+effective_radius))&...
    (events(:,2)>(1-effective_radius))&(events(:,2)<=(2+effective_radius));
passive_events=events(passive_participants,:);
events_number=sum(passive_participants);

%out of these, only a subset (located in the focal domain) can serve as starting points for the consumer

active_participants=((passive_events(:,1)>1)&(passive_events(:,1)<=2)&...
    (passive_events(:,2)>1)&(passive_events(:,2)<=2));
active_events=passive_events(active_participants,:);
active_events_number=sum(active_participants);
end

%% check for consumptions in the focal domain

%the consumer's starting point for immobile resources
    focal_event=active_events(ceil(rand*active_events_number),:);
%the available resources

focal_event_index=find((passive_events(:,1)==focal_event(1))&(passive_events(:,2)==focal_event(2)));
    all_other_events=passive_events([1:(focal_event_index-1) (focal_event_index+1):events_number],:);
%check if there are any resources within the consumers movement corridor
    sqrd_x_diffs=(all_other_events(:,1)-focal_event(1)).^2;
    relevant_events=find(sqrd_x_diffs<sqrd_effective_radius);
%if there are, calculate the step length
    if relevant_events
        sqrd_x_diffs=sqrd_x_diffs(relevant_events);
        sqrd_y_diffs=(all_other_events(relevant_events,2)-focal_event(2)).^2;
%check if there are any resources within the detection range at time 0
    within_initial_disc=sqrd_y_diffs<sqrd_effective_radius;
    if any(within_initial_disc)
        event_distances(n)=min(sqrt(sqrd_x_diffs(within_initial_disc)+sqrd_y_diffs(within_initial_disc)));
    else
%since the consumer travels vertically, only resources located above the starting point are available
        upper_relevant_events=all_other_events(relevant_events,2)>focal_event(2);
            if any(upper_relevant_events)
                sqrd_y_diffs=sqrd_y_diffs(upper_relevant_events);
                sqrd_x_diffs=sqrd_x_diffs(upper_relevant_events);
                distance2event=sqrt(sqrd_y_diffs)-sqrt(sqrd_effective_radius-sqrd_x_diffs)+effective_radius;
                event_distances(n)=min(distance2event);
%the starting point for mobile resources
random_x=rand+1;
sqrd_x_diffs=(all_other_events(:,1)-random_x).^2;
relevant_events=find(sqrd_x_diffs<sqrd_effective_radius);
if relevant_events
    sqrd_x_diffs=sqrd_x_diffs(relevant_events);
sqrd_y_diffs=(all_other_events(relevant_events,2)-1).^2;
    %check if there are any resources within the detection range at time 0
    within_initial_disc=sqrd_y_diffs<sqrd_effective_radius;
    if any(within_initial_disc)
        point_distances(n)=min(sqrt(sqrd_x_diffs(within_initial_disc)+sqrd_y_diffs(within_initial_disc)));
    else
        %since the consumer travels vertically, only resources located above the starting point are available
        upper_relevant_events=all_other_events(relevant_events,2)>1;
        if any(upper_relevant EVENTS)
            sqrd_y_diffs=sqrd_y_diffs(upper_relevant_events);
sqrd_x_diffs=sqrd_x_diffs(upper_relevant_events);
distance2events=sqrt(sqrd_y_diffs)-sqrt(sqrd_effective_radius-sqrd_x_diffs)+effective_radius;
        end
    end
end

% repeat until the original trajectory terminates with a detection event
%the counter counts the number of times the consumer crossed over to a new domain
counter=0;
while isnan(point_distances(n))||isnan(event_distances(n))
    %the function generate_NeymanScott_events_2 returns an array of the x and y locations of points scattered between 0 and 3, including those events generated in the previous landscape and overlapping with the current one
    [events heritability highest_heritable_source]=...
    generate_NeymanScott_events_2(density,clump_size_lambda,dispersal_std,events(heritability,:),effective_radius,highest_heritable_source);
    passive_participants=(events(:,1)>=(1-effective_radius))&(events(:,1)<=(2+effective_radius))&...
    (events(:,2)>=(1+effective_radius))&(events(:,2)<=(2+effective_radius));
    passive_events=events(passive_participants,:);
%find detections in the case of immobile resources
if isnan(event_distances(n))
    sqrd_x_diffs=(passive_events(:,1)-focal_event(1)).^2;
    relevant_events=find(sqrd_x_diffs<sqrd_effective_radius);
    if relevant_events
        y_diffs=passive_events(relevant_events,2)+(2-focal_event(2))+counter;
        distance2events=y_diffs-sqrt(sqrd_effective_radius-sqrd_x_diffs(relevant_events))+effective_radius;
        event_distances(n)=min(distance2events);
    end
end

%find detections in the case of mobile resources
if isnan(point_distances(n))
    sqrd_x_diffs=(passive_events(:,1)-random_x).^2;
    relevant_events=find(sqrd_x_diffs<sqrd_effective_radius);
    if relevant_events
        y_diffs=passive_events(relevant_events,2)+1+counter;
        distance2events=y_diffs-sqrt(sqrd_effective_radius-sqrd_x_diffs(relevant_events))+effective_radius;
        point_distances(n)=min(distance2events);
    end
end

% calculate the corresponding expectancies for the consumption and diffusion rates
mean_step_duration_1_immobile=mean(event_distances./consumer_speed);
mean_step_duration_2_immobile=mean((event_distances./consumer_speed)+handling_time);
mean_step_duration_2_mobile=mean((point_distances./(consumer_speed*4/pi))+handling_time);
mean_sqrd_step_length_immobile=mean(event_distances.^2);
mean_sqrd_step_length_mobile=mean(point_distances.^2);
consumption_rate_1_immobile=1/mean_step_duration_1_immobile;
expected_diffusion_rate_1_immobile=mean_sqrd_step_length_immobile/(4*mean_step_duration_1_immobile);
consumption_rate_2_immobile=1/mean_step_duration_2_immobile;
expected_diffusion_rate_2_immobile=mean_sqrd_step_length_immobile/(4*mean_step_duration_2_immobile);
consumption_rate_2_mobile=1/mean_step_duration_2_mobile;
expected_diffusion_rate_2_mobile=mean_sqrd_step_length_immobile/(4*mean_step_duration_2_mobile);

% calculate simulated diffusion rates as the squared displacement of an individual trajectory divided by four times the duration of this trajectory
headings_immobile=rand(1,repetitiones).*2*pi);
final_x_position_immobile=sum(cos(headings_immobile).*event_distances');
final_y_position_immobile=sum(sin(headings_immobile).*event_distances');
total_duration_immobile=sum(event_distances)/consumer_speed;
simulated_diffusion_rate_1_immobile=((final_x_position_immobile.^2)+(final_y_position_immobile.^2))./(4.*total_duration_immobile);
simulated_diffusion_rate_2_immobile=((final_x_position_immobile.^2)+(final_y_position_immobile.^2))./(4.*(total_duration_immobile+(handling_time*repetitiones)));

headings_mobile=rand(1,repetitiones).*2*pi);
final_x_position_mobile=sum(cos(headings_mobile).*point_distances');
final_y_position_mobile=sum(sin(headings_mobile).*point_distances');
total_duration_mobile=sum(point_distances)/((4/pi)*consumer_speed);
simulated_diffusion_rate_2_mobile=((final_x_position_mobile.^2)+(final_y_position_mobile.^2))./(4.*(total_duration_mobile+(handling_time*repetitiones)));
function [events heritability highest_heritable_source]=...

generate_NeymanScott_events(density,clump_size_lambda,dispersal_std,effective_radius)
%to avoid any edge effects, the landscape is generated over an area 9 times larger than the simulation domain so that the domain is surrounded from all sides
%the overall number of events to be generated is thus 9 times the density
N=9*density;
%heritable sources are those affiliated with events located above the domain (y>1). if the consumer exits the domain these clumps are considered in the generation of the new landscape.
highest_heritable_source=(2+effective_radius);

%% this loop generates Neyman-Scott events, clump by clump, until the landscape is saturated
%the number of generated events
n=0;
heritability=[];
events=[];
while n<N
%the number of events in each aggregate is sampled from a Poisson distribution with mean clump_size_lambda. A null value represents a random distribution and thus exactly 1 resource per clump
if clump_size_lambda
  clump_size=poissrnd(clump_size_lambda);
else
  clump_size=1;
end
n=n+clump_size;
%the distance of events within an aggregate from the center of the aggregate is sampled from a normal distribution with standard deviation dispersal_std
dispersal_distances=randn(clump_size,1).*dispersal_std;
%other than that, the location of the events within a clump is random
dispersal_azimuth=rand(clump_size,1).*(2*pi);
[x,y]=pol2cart(dispersal_azimuth,dispersal_distances);
%the location of the center of the clump is sampled from a uniform-random distribution between 0 and 3 (the focal domain is located between 1 and 2)
source=rand(1,2)*3;
%events that fall outside of the 9*domain boundaries are discarded
current_events=([x+source(1)) (y+source(2))];
valid_events=(current_events(:,1)<=3)&(current_events(:,1)>0)&(current_events(:,2)>0)&(current_events(:,2)<=3);
current_events=current_events(valid_events,:);

%if a clump occur in both the focal (i.e., center) domain and the one above it (the one the consumer will travel to after leaving the focal domain), the events in that clump are saved and will be used in generating the next landscape
if any((current_events(:,2)>1)&(current_events(:,2)<=(2+effective_radius)))
    current_heritability=current_events(:,2)>1;
    highest_heritable_source=max(highest_heritable_source,source(2));
else
    current_heritability=zeros(sum(valid_events),1);
end
heritability=[heritability; current_heritability];
events=[events; current_events];
end
%the order of events is randomised and excess events are removed
randomised_indices=randperm(length(heritability));
randomised_indices(N+1:end)=[];
events=events(randomised_indices,:);
heritability=logical(heritability(randomised_indices));
function  [events  heritability  highest_heritable_source]=...

generate_NeymanScott_events_2(density,clump_size_lambda,dispersal_std,previus_events,effective_radius,highest_heritable_source)

N=9*density;
previus_events(:,2)=previus_events(:,2)-1;
min_height=highest_heritable_source-1;
events=previus_events;
n=length(events);
heritability=events(:,2)>1;
highest_heritable_source=(2+effective_radius);

while n<N
    if clump_size_lambda
        clump_size=poissrnd(clump_size_lambda);
    else
        clump_size=1;
    end
    n=n+clump_size;
dispersal_distances=randn(clump_size,1).*dispersal_std;
dispersal_azimuth=rand(clump_size,1).*(2*pi);
[x,y]=pol2cart(dispersal_azimuth,dispersal_distances);
%the minimum height (y position) for a new source is the height of the highest inherited source
source=rand(1,2);
source(1)=source(1)*3;
source(2)=(source(2)*(3-min_height))+min_height;
current_events=[(x+source(1)) (y+source(2))];
valid_events=(current_events(:,1)<=3)&(current_events(:,1)>0)&(current_events(:,2)>0)&(current_events(:,2)<=3);
current_events=current_events(valid_events,:);
if any((current_events(:,2)>1)&(current_events(:,2)<=(2+effective_radius)))
    current_heritability=current_events(:,2)>1;
highest_heritable_source=max(highest_heritable_source, source(2));
else
    current_heritability=zeros(sum(valid_events),1);
end
heritability=[heritability; current_heritability];
events=[events; current_events];
end
randomised_indices=randperm(length(heritability));
randomised_indices(N+1:end)=[];
events=events(randomised_indices,:);
heritability=logical(heritability(randomised_indices));
Appendix C (Chapter 1)

Matlab simulation code for Scenario IV

```matlab
%% this code simulates the diffusion-consumption process for correlated random walkers consuming immobile, randomly distributed, resources

clear;

densities=[1 5 10 50 100 500 1000 5000 10000];
consumption_events=1000;%the number of consumption events to be recorded (i.e., the sample size)
displacements=NaN(consumption_events,length(densities));%this matrix will hold the displacement affiliated with each consumption event
path_lengths=NaN(consumption_events,length(densities));%this matrix will hold the actual distances traveled
consumer_step=0.001;%the length of a single step
detection_range=0.0001;%the distance at which detection will occur
k=1;%the von-Mises concentration parameter defines the degree of autocorrelation in movement direction. k>=0.0001 where 0.0001 corresponds to pure diffusion and inf corresponds to pure straight line motion.
angles_n=1000000;%the size of the turn angle vector
rand('twister', sum(100*clock));%reset the random number generator seed
%randraw is an efficient random variates generator available through Matlab Central File Exchange
turn_angles=randraw('vonmises',[0, k],angles_n);%a vector of randomly generated turn angles sampled from a von-Mises distribution with concentration parameter k

for s=1:length(densities)
    density=densities(s)%the corners of the simulation domain
    max_x=500/sqrt(density);
    min_x=-500/sqrt(density);
    max_y=500/sqrt(density);
    min_y=-500/sqrt(density);
    x_side_length=max_x-min_x;
    y_side_length=max_y-min_y;
%placing a million resources
    resource_n=round(x_side_length*y_side_length*density);
    resources_x=rand(resource_n,1).*(x_side_length) + min_x;
    resources_y=rand(resource_n,1).*(y_side_length) + min_y;
%to minimise run-time, the domain is divided into circular zones so that the consumer searches for resources located in its current zone
    zones_spacing=x_side_length*(consumer_step+detection_range);

    unique_x_zones=(min_x+zones_spacing):zones_spacing:(zones_spacing*floor(x_side_length/(2*zones_spacing)));
    x_zones=repmat(unique_x_zones,1,length(unique_x_zones));
```
y_zones=sort(x_zones);
zone_radius=2*zones_spacing;
%the consumer starts at the center of the domain
initial_consumer_x=0;
initial_consumer_y=0;
%initiate all relevant arrays
consumer_x=initial_consumer_x;
consumer_y=initial_consumer_y;
[temp,current_zone]=min(sqrt((consumer_x-x_zones).^2 + (consumer_y-y_zones).^2));%assign the right current zone
relevant_resources=(sqrt((resources_x-x_zones(current_zone)).^2 + (resources_y-y_zones(current_zone)).^2))<=zone_radius;%the relevant resources are those with the radius of the current zone
relevant_resources_x=resources_x(relevant_resources);
relevant_resources_y=resources_y(relevant_resources);
counter=0;%counts steps
displacement_thus_far=0;
detected_resources=sqrt((relevant_resources_x-consumer_x).^2 + (relevant_resources_y-consumer_y).^2)<=detection_range;%check for detections at the starting point
detection=sum(detected_resources);
%
%run over all events
for i=1:consumption_events
heading=rand*2*pi;%initial heading
x_step=0;
y_step=0;
%here the simulation begins
while ~detection%this loop will run as long as the consumer is within the domain and hasn’t detected a resource
counter=counter+1;
%updating the current location of the consumer
consumer_x=consumer_x + x_step;
consumer_y=consumer_y + y_step;
%making sure the consumer does not exit the simulation domain
if (consumer_x>max_x || consumer_x<min_x || consumer_y>max_y || consumer_y<min_y)
displacement_thus_far=displacement_thus_far + sqrt((consumer_x-initial_consumer_x).^2 + (consumer_y-initial_consumer_y).^2);
initial_consumer_x=0;
initial_consumer_y=0;
consumer_x=initial_consumer_x;
consumer_y=initial_consumer_y;
resources_x=rand(resource_n,1).*(x_side_length) + min_x;
resources_y=rand(resource_n,1).*(y_side_length) + min_y;
end
%assign a new heading based on a randomly drawn turn angle
heading=heading+turn_angles(ceil(rand*angles_n));
heading=heading-((heading>(2*pi))*(2*pi))+((heading<0)*(2*pi));
x_step=consumer_step*cos(heading);
y_step=consumer_step*sin(heading);

%update the current zone
    if sqrt((consumer_x-x_zones(current_zone))^2+(consumer_y-y_zones(current_zone))^2)>(zone_radius/2)
        [temp,current_zone]=min(sqrt((consumer_x-x_zones).^2+(consumer_y-y_zones).^2));
        relevant_resources=(sqrt((resources_x-x_zones(current_zone)).^2+(resources_y-y_zones(current_zone)).^2))<=zone_radius;
        relevant_resources_x=resources_x(relevant_resources);
        relevant_resources_y=resources_y(relevant_resources);
    end

%define the slope and the intercept of the line equation characterising the current step
    slope=tan(heading);
    intercept=consumer_y-(slope*consumer_x);

%search for detected resources by identifying resources that are projected on the line between the step start point and the end point and that are within detection range

    resources_x_projections=((slope.*relevant_resources_y)+relevant Resources_x-(slope*intercept))./(1+(slope^2));
    resources_y_projections=(slope.*resources_x_projections)+intercept;

detected_resources=((resources_x_projections<consumer_x)&&(resources_x_projections>(consumer_x+x_step)) || ...
                        (resources_x_projections>consumer_x)&&(resources_x_projections<(consumer_x+x_step)) && ...
                        (resources_y_projections<consumer_y)&&(resources_y_projections>(consumer_y+y_step)) || ...
                        (resources_y_projections>consumer_y)&&(resources_y_projections<(consumer_y+y_step)) && ...
                        (sqrt((resources_x_projections-relevant_resources_x).^2+(resources_y_projections-relevant_resources_y).^2)<=detection_range);

%adding those that are within detection range from the end-point
    detected_resources=detected_resources | ...
                        (sqrt((relevant_resources_x-(consumer_x+x_step)).^2+(relevant_resources_y-(consumer_y+y_step)).^2)<=detection_range);
    detection=sum(detected_resources);
end
if detection==1
    approach_dist=sqrt((relevant_resources_x(detected_resources)-
    consumer_x).^2 + (relevant_resources_y(detected_resources)-consumer_y).^2);
    if approach_dist>detection_range
        if the detected resource is out
            of detection range the consumer will move a certain distance (temp2-temp1)
            before turning toward the resource
                temp1=sqrt((detection_range^2) -
                ((resources_x_projections(detected_resources)-
                relevant_resources_x(detected_resources))^2 +...
                (resources_y_projections(detected_resources)-
                relevant_resources_y(detected_resources))^2));
                temp2=sqrt((resources_x_projections(detected_resources)-
                consumer_x)^2 + (resources_y_projections(detected_resources)-consumer_y)^2);
                approach_dist=detection_range+temp2-temp1;
            end
        % the consumer is placed at the detected resource
        consumer_x=relevant_resources_x(detected_resources);
        consumer_y=relevant_resources_y(detected_resources);
    else % this handles the case of more than one detected resources
        places=find(detected_resources);
        [approach_dist,I]=min(sqrt((relevant_resources_x(detected_resources)-
        consumer_x).^2 + (relevant_resources_y(detected_resources)-consumer_y).^2));
        closest_resource=places(I);
        if approach_dist>detection_range
            temp1=sqrt((detection_range^2) -
            ((resources_x_projections(closest_resource)-
            relevant_resources_x(closest_resource))^2 +...
            (resources_y_projections(closest_resource)-
            relevant_resources_y(closest_resource))^2));
            temp2=sqrt((resources_x_projections(closest_resource)-
            consumer_x)^2 + (resources_y_projections(closest_resource)-consumer_y)^2);
            approach_dist=detection_range+temp2-temp1;
        end
        consumer_x=relevant_resources_x(closest_resource);
        consumer_y=relevant_resources_y(closest_resource);
    end
% update the results matrices and save
    displacements(i,s)=displacement_thus_far + sqrt((consumer_x-
    initial_consumer_x)^2 + (consumer_y-initial_consumer_y)^2);
    path_lengths(i,s)=((counter-(counter>0))*consumer_step)+approach_dist;
    save;
% set the grounds for the next move
    counter=0;
    displacement_thus_far=0;
    initial_consumer_x=consumer_x;
    initial_consumer_y=consumer_y;
detected_resource = find((resources_x == consumer_x) & (resources_y == consumer_y));
resources_x(detected_resource) = rand*(x_side_length) + min_x;
resources_y((detected_resource)) = rand*(y_side_length) + min_y;
[temp, current_zone] = min(sqrt((consumer_x - x_zones).^2 + (consumer_y - y_zones).^2));
relevant_resources = (sqrt((resources_x - x_zones(current_zone)).^2 + (resources_y - y_zones(current_zone)).^2)) <= zone_radius;
relevant_resources_x = resources_x(relevant_resources);
relevant_resources_y = resources_y(relevant_resources);
detected_resources = sqrt((relevant_resources_x - consumer_x).^2 + (relevant_resources_y - consumer_y).^2) <= detection_range;
detection = sum(detected_resources);
end
save;
end

%%% calculate the estimated consumption and diffusion rates
handling_time = 10;
consumer_speed = 100;
mean_step_duration_1 = mean(path_lengths./consumer_speed);
mean_step_duration_2 = mean((path_lengths./consumer_speed)+handling_time);
mean_sqrd_step_length = mean(displacements.^2);
consumption_rate_1 = 1./mean_step_duration_1;
consumption_rate_2 = 1./mean_step_duration_2;
diffusion_rate_1 = mean_sqrd_step_length./(4.*mean_step_duration_1);
diffusion_rate_2 = mean_sqrd_step_length./(4.*mean_step_duration_2);
Appendix D (Chapter 1)

Matlab simulation code for Scenario II

```
%% this code simulates a diffusion-consumption process where resources are
moving with Maxwell-Boltzmann distributed velocities. The domain is a
circular arena centred at (0,0) with radius r, the consumer is located at the
center of the domain and travels upwards. The domain "follows" the consumer
while it moves and the velocities of the resources are adjusted accordingly

clear;
resource_density=100;
effective_radius=0.0001;
consumer_speed=100;
domain_radius=100;
handling_time=1;
%the total number of resource items in the domain
resource_number=round(pi*(domain_radius^2)*resource_density);
%the mean Maxwell-Boltsman velocities to be simulated
resource_velocities=0:50:1000;
%the number of encounters to be recorded for each speed
sample_size=1000;
results=NaN(length(resource_velocities),5);

%% run over all resource speed values
for v=1:length(resource_velocities)
    mean_resource_speed=resource_velocities(v);
    %this array will accommodate the results for the current speed
durations=NaN(1,sample_size);
    % here the simulation begins
    for i=1:sample_size
        %set the initial distribution of resources in the domain
        headings=rand(1,resource_number).*2*pi;
        distances=sqrt(rand(1,resource_number)).*domain_radius;
        resources_x0=sin(headings).*distances;
        resources_y0=cos(headings).*distances;
        clear headings distances

        %set the X and Y velocities of each resource item. the Y speed is adjusted to
the motion of the domain (i.e., the consumer) upwards
        resources_x_speed=randn(1,resource_number).*(mean_resource_speed/sqrt(pi/2));
        resources_y_speed=(randn(1,resource_number).*(mean_resource_speed/sqrt(pi/2))
        )-consumer_speed;

        %find the closest encounter by solving for the times at which each resource
will enter the detection range of the consumer
        quadratic_coefficients=(resources_x_speed.^2)+(resources_y_speed.^2);
```

linear_coefficients = 2.0 * ((resources_x_speed.*resources_x0)+(resources_y_speed.*resources_y0));
constant_terms = (resources_x0.^2)+(resources_y0.^2)-(effective_radius^2);
roots = (linear_coefficients.^2)-(4.*quadratic_coefficients.*constant_terms);
intercept_times_1 = inf(1,resource_number);
positive_roots = roots >= 0;
intercept_times_1(positive_roots) = (-linear_coefficients(positive_roots)+sqrt(roots(positive_roots)))./(2.*quadratic_coefficients(positive_roots));
intercept_times_2 = inf(1,resource_number);
intercept_times_2(positive_roots) = (-linear_coefficients(positive_roots)-sqrt(roots(positive_roots)))./(2.*quadratic_coefficients(positive_roots));
intercept_times = intercept_times_1;
intercept_times(intercept_times_2<intercept_times_1) = intercept_times_2(intercept_times_2<intercept_times_1);
clear intercept_times_2 intercept_times_1 roots positive_roots

% calculate the times at which each resource will exit the domain
quadratic_coefficients = (resources_x_speed.^2)+(resources_y_speed.^2);
linear_coefficients = 2.0 * ((resources_x_speed.*resources_x0)+(resources_y_speed.*resources_y0));
constant_terms = (resources_x0.^2)+(resources_y0.^2)-(domain_radius^2);
root_times = (linear_coefficients.^2)-(4.*quadratic_coefficients.*constant_terms);
negative_exit_times = root_times<0;
negative_exit_times(negative_exit_times) = (-linear_coefficients(negative_exit_times)+sqrt(root_times(negative_exit_times)))./(2.*quadratic_coefficients(negative_exit_times));
clear root_times quadratic_coefficients linear_coefficients constant_terms negative_exit_times
% to keep density constant while resources are mobile, each resource item exiting the domain before the consumer-resource encounter is replaced by a new resource item entering the domain in a random direction and with a random speed. The interception time for each new resource item is then used to update the closest interception time. This procedure is repeated until the closest interception time precedes any resource exiting event.

```matlab
out=(resources_exit_times<closest_intercept_time);
sum_out=sum(out);
while sum_out

% set the entry points and velocities of all replacement resource items

resources_x_speed=randn(1,sum_out).*(mean_resource_speed/sqrt(pi/2));
resources_y_speed=(randn(1,sum_out).*(mean_resource_speed/sqrt(pi/2)))-
consumer_speed;

quarter_1=(resources_x_speed<0)&(resources_y_speed<0);
quarter_2=(resources_x_speed<0)&(resources_y_speed>=0);
quarter_3=(resources_x_speed>=0)&(resources_y_speed>=0);
quarter_4=(resources_x_speed>=0)&(resources_y_speed<0);
headings=rand(1,sum_out);
headings(quarter_1)=headings(quarter_1).*(pi/2);
headings(quarter_2)=(headings(quarter_2).*(pi/2))+(pi/2);
headings(quarter_3)=(headings(quarter_3).*(pi/2))+(pi);
headings(quarter_4)=(headings(q

% find the closest encounter by solving for the times at which each resource will enter the detection range of the consumer

quadratic_coefficients=(resources_x_speed.^2)+(resources_y_speed.^2);
linear_coefficients=2.*((resources_x_speed.*resources_x0)+(resources_y_speed.*resources_y0));
constant_terms=(resources_x0.^2)+(resources_y0.^2)-(effective_radius^2);
roots=(linear_coefficients.^2)-(4.*quadratic_coefficients.*constant_terms);
intercept_times_1=inf(1,sum_out);
positive_roots=roots>=0;
intercept_times_1(positive_roots)=(-linear_coefficients(positive_roots)+sqrt(roots(positive_roots))))./(2.*quadratic_coefficients(positive_roots)));
intercept_times_1(intercept_times_1<0)=inf;
intercept_times_2=inf(1,sum_out);
```

117
\[ \text{intercept\_times\_2(positive\_roots)} = \frac{-\text{linear\_coefficients(positive\_roots)} - \sqrt{\text{roots(positive\_roots)}}}{2 \cdot \text{quadratic\_coefficients(positive\_roots)}}; \]

\[ \text{intercept\_times\_2(intercept\_times\_2<0)} = \text{inf}; \]

\[ \text{intercept\_times} = \text{intercept\_times\_1}; \]

\[ \text{clear intercept\_times\_2 intercept\_times\_1 roots positive\_roots quadratic\_coefficients linear\_coefficients constant\_terms} \]

\[ \text{intercept\_times} = \text{intercept\_times} + \text{resources\_exit\_times(out)}; \]

\[ \text{finite\_intercept\_times} = \text{intercept\_times} < \text{inf}; \]

\[ \text{if any(finite\_intercept\_times)} \]

\[ \text{closest\_intercept\_time} = \min(\{\text{intercept\_times(finite\_intercept\_times)}\}); \]

\[ \text{end} \]

\[ \text{clear intercept\_times finite\_intercept\_times} \]

% calculate the times at which each resource will exit the arena

\[ \text{quadratic\_coefficients} = (\text{resources\_x\_speed}^2) + (\text{resources\_y\_speed}^2); \]

\[ \text{linear\_coefficients} = 2 \cdot ((\text{resources\_x\_speed} \cdot \text{resources\_x\_0}) + (\text{resources\_y\_speed} \cdot \text{resources\_y\_0})); \]

\[ \text{constant\_terms} = (\text{resources\_x\_0}^2) + (\text{resources\_y\_0}^2) - (\text{domain\_radius}^2); \]

\[ \text{clear resources\_x\_speed resources\_y\_speed resources\_x\_0 resources\_y\_0} \]

\[ \text{roots} = (\text{linear\_coefficients}^2) - (4 \cdot \text{quadratic\_coefficients} \cdot \text{constant\_terms}); \]

\[ \text{temp\_resources\_exit\_times} = (-\text{linear\_coefficients} + \sqrt{\text{roots}})/(2 \cdot \text{quadratic\_coefficients}); \]

\[ \text{negative\_exit\_times} = \text{temp\_resources\_exit\_times} <= 0; \]

\[ \text{temp\_resources\_exit\_times(negative\_exit\_times)} = (-\text{linear\_coefficients(negative\_exit\_times)} - \sqrt{\text{roots(negative\_exit\_times)}})/(2 \cdot \text{quadratic\_coefficients(negative\_exit\_times)}); \]

\[ \text{resources\_exit\_times} = \text{temp\_resources\_exit\_times} + \text{resources\_exit\_times(out)}; \]

\[ \text{clear roots quadratic\_coefficients linear\_coefficients constant\_terms negative\_exit\_times temp\_resources\_exit\_times} \]

\[ \text{out} = (\text{resources\_exit\_times(<closest\_intercept\_time)}); \]

\[ \text{sum\_out} = \text{sum(out)}; \]

\[ \text{end} \]

% if no more resource items are scheduled to leave the domain before the next intercept, the duration of the step is the time to the closest intercept

\[ \text{durations(i)} = \text{closest\_intercept\_time}; \]

\[ \text{end} \]
%% calculate the estimated consumption and diffusion rates
mean_step_duration_1 = mean(durations);
mean_step_duration_2 = mean(durations + handling_time);
mean_sqrd_step_length = mean((durations.*consumer_speed).^2);
consumption_rate_1 = 1./mean_step_duration_1;
diffusion_rate_1 = mean_sqrd_step_length./(4.*mean_step_duration_1);
consumption_rate_2 = 1./mean_step_duration_2;
diffusion_rate_2 = mean_sqrd_step_length./(4.*mean_step_duration_2);
results(v,:) = [mean_resource_speed consumption_rate_1 consumption_rate_2
diffusion_rate_1 diffusion_rate_2];
save;
end
Appendix E (Chapter 2)

Semivariograms for the different response variables discussed in the main text. Plots were produced in package *nlme* by plotting the semivariograms of the full model (i.e., with all candidate fixed effects) against space, time, or both (depending on which option minimises the AIC score). Plots reflect autocorrelation between residuals within the random effects. Euclidian spatiotemporal distance matrices were generated by pointwise multiplication of the two spatial dimensions, \(x\) and \(y\), by the temporal dimension, \(t\):

\[
d_{i,j} = \sqrt{(t_i x_i - t_j x_j)^2 + (t_i y_i - t_j y_j)^2},
\]

coded in *nlme* as: `correlation=corExp(form=~T*(X+Y))`.
% folder and filename: simulator\simulator.m

clear;

% time steps can have variable duration
step_duration=1;
% the simulation duration
steps_number=365*(24/step_duration);
% the hexagonal cell size is also the distance between the centroids of adjacent cells
cell_dim=100;
% the total number of cells in the landscape; if it is too small, the simulation might terminate because of hitting the edge.
% note however that the run time increase with the doamin size.
landscape_size=4000000;
% the spatial smoothing coefficient determines the level of spatial autocorrelation in the simulated landscape (for both habitat components)
spatial_smoothing_coefficient=50;

%% Parametrizing the simulation
% the default value for the quality of any location in the landscape is used when no other information is available (values must be positive numerics; % units are the same as for the landscape variable). for an attractor, high values will lead to an over estimation of habitat quality when perceptual or memory information % are lacking (i.e., assuming a hospitable surroundings). Low values will lead to an under estimation of habitat quality when perceptual or memory % information is lacking (i.e., assuming a hostile surroundings). values must be bounded between the minimum and maximum habitat quality values
prior_habitat_quality_1=0.7;
prior_habitat_quality_2=0.3;
% the sensory input from the landscape declines as an exponential function of the Euclidian distance yet increases with exposure time.
% in a single instant, the amount of information perceived by the animal is some fraction [0,1] of the "truth". The sensory decay % coefficient determines the exponential rate of information reduction with distance (units are time/distance; values must be positive numerics; % 0 corresponds to no sensory limitations; inf corresponds to no sensory input). value must be larger than 0.0001.
sensory_decay_coefficient=0.005;
% the maximal perceptual distance is set according to the values of the sensory decay coefficients and the step duration.
max_percep_dist=step_duration*(-log(0.01)/sensory_decay_coefficient)+cell_dim;
% landscape information stored in memory decay as an exponential function of time; in a single instant, the amount of information retained % is some fraction [0,1] of the information held in the previous instant. The memory decay coefficient determines the exponential rate of information % reduction with time (units are 1/time; values must be positive numerics; 0 corresponds to no memory lost; inf corresponds to no memory input).
memory_decay_coefficient=0.005;
% the memory loads are the fraction of the cognitive landscape that is remembered in the next time step
memory_load=exp(-memory_decay_coefficient.*step_duration);
%the perceived landscape may be further manipulated in the cognitive system
%by enhancing or reducing the magnitude of the spatial differences;
%negative values correspond to repulsion rather than attraction; values
%between -1 and 1 decrease the perceived quality differences while values
%larger than 1 or smaller than -1 enhance the perceived differences in
%habitat quality, if the value is null there will be no response to habitat
quality
enhancement_coefficient_1=2;
enhancement_coefficient_2=-2;
%assuming that movement is costly and that this cost increases with distance
%and decreases with time, the travel cost coefficient defines
%the shape of the decline in profitability with distance (values must be
%positive numerics; a null value results in a null effect).
travel_cost_coefficient=0.005;
%the maximal displacement distance is a function of the step duration and the
%travel cost coefficient
max_displacement=step_duration*(-log(0.01)/travel_cost_coefficient)+
cell_dim;

%% 
for realisation=1:20
% generate a simulated landscape
%the landscape matrix includes 4 columns: X position, Y position, habitat
quality 1 and habitat quality 2 (all values must be positive numerics).
%in addition the landscape generator returns the landscape matrix which
%contains all landscape cell numbers in the spatial context they appear in
%the landscape (row are correspond to X values and columns correspond to Y).
[landscape
landscape_matrix]=landscape_generator(cell_dim,landscape_size,spatial_smoothing_coefficient);
%the X positions of all landscape cells
X=single(landscape(:,1));
landscape_size=length(X);
all_cell_indices=1:landscape_size;
max_X=max(X);
%the Y positions of all landscape cells
Y=single(landscape(:,2));
max_Y=max(Y);
%habitat component 1 value for each cell in the landscape
habitat_quality_1=single(landscape(:,3));
%habitat component 2 value for each cell in the landscape
habitat_quality_2=single(landscape(:,4));
%the initial location of the animal is the center of the domain
[temp initial_focal_cell]=min(((max_X/2)-X).^2)+(((max_Y/2)-Y).^2));

%% Initializing the simulation
%initialise the random number generator seed
rand('twister', sum(100*clock));
%this array will contain the cell indices corresponding to the simulated path
(updated stepwise):
path_cells=single(NaN(steps_number,1));

%% the main loop of the simulation
%if no cognitive capacities exist, only a simplified version of the algorithm
%is used - a simple random walk, otherwise the full algorithm is used
if enhancement_coefficient_1||enhancement_coefficient_2
while isnan(path_cells(end))
% initialising the cognitive landscapes (the entire landscape as it is perceived in the animal's mind)

cognitive_landscape_1=ones(landscape_size,1).*prior_habitat_quality_1;
cognitive_landscape_2=ones(landscape_size,1).*prior_habitat_quality_2;
memorised_cells=[];
path_cells(1)=initial_focal_cell;
focal_cell=0;
for step=1:steps_number-1
% update distances to all perceived cells in case of a change in location or step duration
    if focal_cell~=path_cells(step)
        moved=true;
        focal_cell=path_cells(step);
    end
% to avoid edge effects the simulation keeps track of the distance between the current position of the animal and the edge of the simulation domain:
    focal_cell_distance_from_edge=min([X(focal_cell) (max_X-X(focal_cell)) Y(focal_cell) (max_Y-Y(focal_cell))]);
    if focal_cell_distance_from_edge<=max_displacement;
        error('too close to the edge');
    end
% if the maximal perceptual distance is larger than the maximal relocation distance, the available landscape is a sub-set of the perceived landscape
    if max_percep_dist>max_displacement
        % define the coarse landscape window currently relevant for perception
        percived_cells=sub_landscape_calculator(X(focal_cell),Y(focal_cell),max_percep_dist,cell_dim,landscape_matrix);
        % calculate the Euclidian distances between the current location and all other cells and refine the selection
        distances=sqrt((X(percived_cells(:))-X(focal_cell)).^2 + (Y(percived_cells(:))-Y(focal_cell)).^2);
        within_percep_bounds=(distances<=max_percep_dist);
        percived_cells=percived_cells(within_percep_bounds);
    % the distances to all perceived cells
        percived_cells_distances=distances(within_percep_bounds);
        clear within_percep_bounds distances
    % define the cells currently available for relocation
        within_availability_bounds=(percived_cells_distances<=max_displacement);
        available_cells=percived_cells(within_availability_bounds);
        % the distances to all currently available cells
        available_cells_distances=percived_cells_distances(within_availability_bounds);
        clear within_availability_bounds
    else
        % the perceived landscape is a sub-set of the available landscape
    end
% define the coarse landscape window currently available for relocation
   available_cells=available_cells_calculator(X(focal_cell),Y(focal_cell),max_displacement,cell_dim,landscape_matrix);
% calculate the Euclidean distances between the current location and all other cells and refine the selection
    distances=sqrt((X(available_cells(:))-X(focal_cell)).^2 + (Y(available_cells(:))-Y(focal_cell)).^2);

within_availability_bounds=(distances<=max_displacement);

available_cells=available_cells(within_availability_bounds);
% the distances to all currently available cells
available_cells_distances=distances(within_availability_bounds);

% define the cells currently perceived
within_percep_bounds=(available_cells_distances<max_percep_dist);
perceived_cells=available_cells(within_percep_bounds);
% the distances to all perceived cells
percived_cells_distances=available_cells_distances(within_percep_bounds);

% to avoid assuming perfect (i.e., 100%) sensory information input from the current location, the distance of the focal cell is set to % half way from the center to the edge
percived_cells_distances(perceived_cells==focal_cell)=cell_dim/4;
% the information load is the fraction of information perceived by the animal during a given time step from each of the spatial cells % in the landscape and it is proportional to the step duration
information_loads=exp(-sensory_decay_coefficient.*percived_cells_distances./step_duration);

if enhancement_coefficient_1
% the sensed (as opposed to remembered) quality of each cell is a product of the information load (a fraction) % and the real quality of the cell
sensed_qualities_1=information_loads.*habitat_quality_1(percived_cells);
end

end

if enhancement_coefficient_2
sensed_qualities_2=information_loads.*habitat_quality_2(percived_cells);
end

% travel costs are proportional to distance and decrease with time available for travel (i.e., step duration)
travel_costs_factor=exp(-travel_cost_coefficient.*available_cells_distances./step_duration);

end

%% the cognitive landscape is the landscape as it is currently perceived by the animal and it is updated at each time step by the cognitive algorithm. % the cognitive landscape is a product of viewing the habitat quality via the "lenses" of sensory and memory inputs. This algorithm was inspired by ideas formulated in Bush 1951, McNamara and Houston 1987, Bernstein 1988 and Lange and Dukas 2009.
% what the animal knows about a specific cell is the sum of what it senses and what it remembers; these are then complimented by a default value % and modified by the perceived depletion
if enhancement_coefficient_1
if memory_load
    if memory_load<1
        cognitive_landscape_1(memorised_cells)=memory_load.*cognitive_landscape_1(memorised_cells) + (1-memory_load)*prior_habitat_quality_1;
    end

cognitive_landscape_1(percived_cells)=sensed_qualities_1 + ((1-information_loads).*cognitive_landscape_1(percived_cells));
available_landscape_1=cognitive_landscape_1(available_cells).^enhancement_coefficient_1;
else
    cognitive_landscape_1(percived_cells)=sensed_qualities_1 + ((1-information_loads).*prior_habitat_quality_1);
    available_landscape_1=cognitive_landscape_1(available_cells).^enhancement_coefficient_1;
    cognitive_landscape_1(percived_cells)=prior_habitat_quality_1;
end
end
if enhancement_coefficient_2
    if memory_load
        if memory_load<1
            cognitive_landscape_2(memorised_cells)=memory_load.*cognitive_landscape_2(memorised_cells) + (1-memory_load)*prior_habitat_quality_2;
        end
    end
    cognitive_landscape_2(percived_cells)=sensed_qualities_2 + ((1-information_loads).*cognitive_landscape_2(percived_cells));
    available_landscape_2=cognitive_landscape_2(available_cells).^enhancement_coefficient_2;
else
    cognitive_landscape_2(percived_cells)=sensed_qualities_2 + ((1-information_loads).*prior_habitat_quality_2);
    available_landscape_2=cognitive_landscape_2(available_cells).^enhancement_coefficient_2;
    cognitive_landscape_2(percived_cells)=prior_habitat_quality_2;
end
end

%if the animal moved, the memorised cells list needs updating
if moved
    moved=false;
    memorised_cells=unique([memorised_cells percived_cells']);
end
integrating the two landscape variables into a single perceived map of available cells

```matlab
if enhancement_coefficient_1&&enhancement_coefficient_2
    available_landscape=available_landscape_1.*available_landscape_2;
else
    available_landscape=available_landscape_1;
end
```

the attraction landscape is a function of the integrated landscape and the decline in attractiveness due to travel costs:
traveling is costly and this cost increases with the distance traveled and decreases with the time available for traveling.

```matlab
attraction_landscape=available_landscape.*travel_costs_factor;
```

declining attractiveness due to travel costs: traveling is costly and this cost increases with the distance traveled and decreases with the time available for traveling.

```matlab
probabilities_of_occurrence=attraction_landscape./sum(attraction_landscape);
```

the redistribution kernel is produced by standardization of the attraction landscape

```matlab
probabilities_of_occurrence=attraction_landscape./sum(attraction_landscape);
```

the next location is stochastically determined according to the redistribution kernel

```matlab
cumsum_probabilities=cumsum(probabilities_of_occurrence);
cumsum_probabilities(end)=1;
next_cell_index=find(cumsum_probabilities>=rand,1);
path_cells(step+1)=available_cells(next_cell_index);
```

either the allocation or the capacity of the environment.

```matlab
 available_cells=available_cells(stdiardage_calculator(X(focal_cell)),cell_dim,landscape_matrix);
```

calculate the Euclidean distances between the current location and all other cells and refine the selection
distances=sqrt((X(available_cells(:))-X(focal_cell)).^2 + (Y(available_cells(:))-Y(focal_cell)).^2);
within_availability_bounds=(distances<=max_displacement);

available_cells=available_cells(within_availability_bounds);
%the distances to all currently available cells

available_cells_distances=distances(within_availability_bounds);
%the only factor determining the attractively of the different landscape cells is the traveling costs (i.e., distance and time)
attraction_landscape=exp(-travel_cost_coefficient.*available_cells_distances./step_duration);
%the redistribution kernel is produced by standardization of the attraction landscape

probabilities_of_occurrence=attraction_landscape./sum(attraction_landscape);
clear attraction_landscape
%the next location is stochastically determined according to the redistribution kernel
cumsum_probabilities=cumsum(probabilities_of_occurrence);
clear probabilities_of_occurrence
cumsum_probabilities(end)=1;
next_cell_index=find(cumsum_probabilities>=rand,1);
path_cells(step+1)=available_cells(next_cell_index);
end

% save the resulting trajectory
observed_path=[path_cells (step_duration*ones(length(path_cells),1))];
filename=['data_' num2str(realisation) '_' num2str(step_duration) '_1'];
save(filename,'observed_path','landscape');
end

% folder and filename: simulator\sub_landscape_calculator.m

%this function returns an rectangular chunk of the landscape containing all cells within a given range of some given location/s
function [sub_landscape_matrix] =
sub_landscape_calculator(x,y,range,cell_dim,landscape_matrix)

orthogonal_x_conversion_factor=cell_dim*sqrt(3)/2;%as the simulation operates on a hexagonal grid, the horizontal distance between neighboring cells is smaller than the Euclidian distance
x_side_length=size(landscape_matrix,1);
y_side_length=size(landscape_matrix,2);
%% define the four corner cells of the sub-landscape surrounding the input point/s
min_row_index=floor((min(x)-range)/orthogonal_x_conversion_factor)-1;
if min_row_index<1
    min_row_index=1;
end
max_row_index=ceil((max(x)+range)/orthogonal_x_conversion_factor)+1;
if max_row_index>x_side_length
    max_row_index=x_side_length;
end
min_col_index=floor((min(y)-range)/cell_dim)-1;
if min_col_index<1
    min_col_index=1;
end
max_col_index=ceil((max(y)+range)/cell_dim)+1;
if max_col_index>y_side_length
    max_col_index=y_side_length;
end

%% output
sub_landscape_matrix=landscape_matrix(min_row_index:max_row_index,min_col_index:max_col_index);

% folder and filename: simulator\landscape_generator.m

function [landscape landscape_matrix] = landscape_generator(cell_dim,n,spatial_smoothing_coefficient)

%cell_dim: by defining the cell dimension (i.e., the cross section length),
one defines the spatial scale of the simulation
%n is the number of cells in the landscape
%the landscape can be smoothed by spatial averaging where the smoothing
coefficient determines the extant

%the width and length of the landscape grid are assumed to be equal:
side_length=sqrt(n);
%define the vectors holding the spatial locations of the centroids of cells 1
to n:
unique_x_values=0:cell_dim:cell_dim*(side_length-1);
X=repmat(unique_x_values,1,length(unique_x_values));
clear unique_x_values
Y=sort(X);
%transform the spatial relationship into hexagonal ones:
X=X.*(sqrt(3)/2);
additions=repmat([0 cell_dim/2]',ceil(side_length/2),side_length);
if size(additions,1)>side_length
    additions((side_length+1):end,:)=[];
end
Y=Y + additions(:)';
clear additions
%output 1
landscape_matrix=uint32(reshape(1:n,side_length,side_length));
generate vectors holding the values of the landscape quality for each of the landscape cells

% initialise the seed for the random number generator
rand('twister', sum(100*clock));
habitat_variable_1=rand(1,n);
habitat_variable_2=rand(1,n);
% assuming a 99% reduction as a threshold value enables determining a finite spatial limit to smoothing:
max_smoothing_distance=-log(0.01)*spatial_smoothing_coefficient;
if max_smoothing_distance >= cell_dim
    habitat_quality_1=zeros(1,n);
    habitat_quality_2=zeros(1,n);
% this loop is iterating over all cells in the landscape and calculate the weighted spatial average of quality per cell:
    for i = 1:n
        relevant_cells=sub_landscape_calculator(X(i),Y(i),max_smoothing_distance,cell_dim,landscape_matrix);
        % calculate the Euclidian distances between the current location and all other cells and refine the selection
        distances=sqrt((X(relevant_cells(:))-X(i)).^2 + (Y(relevant_cells(:))-Y(i)).^2);
        within_bounds=(distances<=max_smoothing_distance);
        relevant_cells_distances=distances(within_bounds);
        relevant_cells=relevant_cells(within_bounds);
        weights=exp(-relevant_cells_distances./spatial_smoothing_coefficient);
        habitat_quality_1(i)=sum(weights.*habitat_variable_1(relevant_cells))./sum(weights);
        habitat_quality_2(i)=sum(weights.*habitat_variable_2(relevant_cells))./sum(weights);
    end
else
    habitat_quality_1=habitat_variable_1;
    habitat_quality_2=habitat_variable_2;
end

clear weights relevant_cells max_smoothing_distance habitat_variable_1

habitats_quality_1=(habitat_quality_1-min(habitat_quality_1))./(max(habitat_quality_1)-min(habitat_quality_1));
habitats_quality_2=(habitat_quality_2-min(habitat_quality_2))./(max(habitat_quality_2)-min(habitat_quality_2));

output 2
landscape = [X' Y' habitat_quality_1' habitat_quality_2'];

folder and filename: simulator\Morans_I_calculator.m

this code calculates Moran's I for a simulated landscape with two independent landscape variables

simulate the landscape
cell_dim=100;
landscape_size=1000000;
spatial_smoothing_coefficient=50;

[landscape
landscape_matrix]=landscape_generator(cell_dim,landscape_size,spatial_smoothing_coefficient);

%% calculate Moran's I
mean_quality_1=mean(landscape(:,3));
mean_quality_2=mean(landscape(:,4));
diffs_1=landscape(:,3)-mean_quality_1;
diffs_2=landscape(:,4)-mean_quality_2;

results_1=0;
results_2=0;
sumed_weights=0;

for i=1:landscape_size

sub_matrix=sub_landscape_calculator(landscape(i,1),landscape(i,2),cell_dim,cell_dim,landscape_matrix);
relevant_cells=sub_matrix(:);
relevant_cells(relevant_cells==i)=[
];
distances=sqrt((landscape(i,1)-landscape(relevant_cells,1)).^2 +
(landscape(i,2)-landscape(relevant_cells,2)).^2);
% spatial weights can be calculated as either a Boolean or a continuous variable
weights=distances<=cell_dim;
results_1=results_1+sum(weights.*diffs_1(i).*diffs_1(relevant_cells));
results_2=results_2+sum(weights.*diffs_2(i).*diffs_2(relevant_cells));
weights=1./sqrt((landscape(i,1)-landscape((i+1):landscape_size,1)).^2 +
(landscape(i,2)-landscape((i+1):landscape_size,2)).^2);
results_1=results_1+sum(weights.*diffs_1(i).*diffs_1((i+1):landscape_size));
results_2=results_2+sum(weights.*diffs_2(i).*diffs_2((i+1):landscape_size));
sumed_weights=sumed_weights+sum(weights);
end
morans_I_1=(landscape_size/sumed_weights)*(results_1/sum(diffs_1.^2));
morans_I_2=(landscape_size/sumed_weights)*(results_2/sum(diffs_2.^2));

------------------------------------------
% folder and file name: data processing\data_processing.m

clear;
% this code resamples the simulated trajectories in 5 different spatial scales and 5 different temporal scales
%
for realisation=1:20
for time_gap=[2 4 8 16]
filename=['data_' num2str(c) '_1_1.mat'];
load(filename)
sampling_times=1:time_gap:length(observed_path);
observed_path=observed_path(round(sampling_times),:);
observed_path(:,2)=time_gap;
filename=['data_' num2str(c) '_' num2str(time_gap) '_1.mat'];
save(filename,'observed_path','landscape');
end
end

%the sample size is half of the simulated trajectory. the first half of the
simulated trajectory is not use to mimic unknown history
sample_size=round(((365*24)/2)/16);
for temporal_resolution=[1 2 4 8 16]
    for realisation=1:20
        %the data file includes the 'observed_path' array (containing the cells
        %occupied at each time step and the time step duration)
        %and the 'landscape' array (containing the coordinates and habitat value of
        %each cell in the landscape)
        filename=['data_' num2str(realisation) '_' num2str(temporal_resolution) '_1.mat'];
        load(filename)
        %the number of steps
        n=size(observed_path,1);
        %only the later half of the simulated trajectory is used
        observed_path=observed_path(round(n/2):round(n/2)+sample_size-1,:);
        %occupied cell numbers
        path_cells=uint32(observed_path(:,1));
        %time elapsed between relocations
        step_durations=single(observed_path(:,2));
        clear observed_path
        %the X positions of all landscape cells
        X=single(landscape(:,1));
        %the Y positions of all landscape cells
        Y=single(landscape(:,2));
        %habitat component 1 value for each cell in the landscape
        habitat_quality_1=single(landscape(:,3));
        %habitat component 2 value for each cell in the landscape
        habitat_quality_2=single(landscape(:,4));
        clear landscape
        %the diameter of a single hexagonal cell or the distance between adjacent
cells
        cell_dim=round(sqrt((X(1)-X(2))^2 + (Y(1)-Y(2))^2));
        %generate a rectangular matrix containing the indices of all cells in the
        landscape
        landscape_matrix=uint32(reshape(1:length(X),sqrt(length(X)),sqrt(length(X))));
        %the maximum distance traveled during a single step
        max_step_length=max(sqrt((X(path_cells(1:end-1))-X(path_cells(2:end))).^2 + (Y(path_cells(1:end-1))-Y(path_cells(2:end))).^2));
        % redefine the landscape so that it is the minimal block encompassing all
        %locations
        %the function 'sub_landscape_calculator' returns and n by m matrix containing
        the numbers of all spatial cell within the rectangle encompassing all
        %observed path cells + a buffer zone equal to the maximum step length. n is
        the number of unique x values; m is the number of unique y values.
[landscape_matrix]=sub_landscape_calculator(X(path_cells),Y(path_cells),max_step_length,cell_dim,landscape_matrix);
%this loop updates the path cells with their index in the new landscape matrix
    for i = 1:sample_size
        path_cells(i)=find(landscape_matrix(:)==path_cells(i));
    end
% updating X values according to the new landscape matrix
    X=X(landscape_matrix(:))-min(X(landscape_matrix(:)));
% updating Y values according to the new landscape matrix
    Y=Y(landscape_matrix(:))-min(Y(landscape_matrix(:)));
    habitat_quality_1=habitat_quality_1(landscape_matrix(:));
    habitat_quality_2=habitat_quality_2(landscape_matrix(:));
% creating a rectangular matrix containing all cells indices from 1 to the number of cells in the newly defined landscape along dimensions n and m.
    landscape_matrix=uint32(reshape(1:length(landscape_matrix(:)),size(landscape_matrix,1),size(landscape_matrix,2)));
    save(filename);
end
end

for spatial_resolution=[2 4 8 16]
    for realisation=1:20
        for temporal_resolution=[1 2 4 8 16]
            filename=['data_' num2str(realisation) '_l.mat'];
            load(filename)
            new_filename=['data_' num2str(realisation) '_num2str(temporal_resolution) '_num2str(spatial_resolution) '.mat'];

            % generate the centroids of the new cells
            % the new distance between the centroids of adjacent cells
            cell_dim=cell_dim*sqrt(spatial_resolution);
            %cell_dim=sqrt(10000*spatial_resolution);
            % the number of cells along one side of the new landscape grid
            side_length=ceil(max(size(landscape_matrix))/sqrt(spatial_resolution));
            % the new total number of cells in the landscape.
            new_landscape_size=side_length^2;
            % define the vectors holding the spatial locations of the centroids of cells 1 to n:
            unique_x_values=0:cell_dim:cell_dim*(side_length-1);
            new X=repmat(unique_x_values,1,length(unique_x_values));
            clear unique x values
            new Y=sort(new X);
            % transform the spatial relationship into hexagonal ones:
            new X=new X.*(sqrt(3)/2);
            additions=repmat([0 cell_dim/2]',ceil(side_length/2),side_length);
            if size(additions,1)>side_length
                additions((side_length+1):end,:)=[];
            end
            new Y=new Y+additions(:);
clear additions

landscape_matrix=uint32(reshape(1:new_landscape_size,side_length,side_length));

%% affiliate each existing cell with the closest new centroid
affiliations=NaN(1,length(X));
    for cell=1:length(X)
        candidate_cells=sub_landscape_calculator(X(cell),Y(cell),cell_dim,cell_dim,landscape_matrix);
            [C,I]=min(((new_X(candidate_cells(:))-X(cell)).^2)+((new_Y(candidate_cells(:))-Y(cell)).^2));
            affiliations(cell)=candidate_cells(I);
        end
    path_cells=affiliations(path_cells)

%% calculate the habitat quality in each of the new cells as the mean of all affiliated cells
new_habitat_quality_1=NaN(new_landscape_size,1);
new_habitat_quality_2=NaN(new_landscape_size,1);
    for new_cell=1:new_landscape_size
        affiliated_cells=find(affiliations==new_cell);
            if isempty(affiliated_cells)
                new_habitat_quality_1(new_cell)=mean(habitat_quality_1);
                new_habitat_quality_2(new_cell)=mean(habitat_quality_2);
            else
                new_habitat_quality_1(new_cell)=mean(habitat_quality_1(affiliated_cells));
                new_habitat_quality_2(new_cell)=mean(habitat_quality_2(affiliated_cells));
            end
        end

%% redefine the landscape so that it is the minimal block encompassing all locations
X=single(new_X');
Y=single(new_Y');
habitat_quality_1=single(new_habitat_quality_1);
habitat_quality_2=single(new_habitat_quality_2);
clear C cell new_cell affiliated_cells new_landscape_size affiliations new_X new_Y new_habitat_quality_1 new_habitat_quality_2
max_step_length=max(sqrt((X(path_cells(1:end-1))-X(path_cells(2:end)))).^2 + (Y(path_cells(1:end-1))-Y(path_cells(2:end)))).^2));

%the function 'sub_landscape_calculator' returns and n by m matrix containing the numbers of all spatial cell within the rectangle encompassing all observed path cells + a buffer zone equal to the maximum step length. n is the number of unique x values; m is the number of unique y values.

[landscape_matrix]=sub_landscape_calculator(X(path_cells),Y(path_cells),max_step_length,cell_dim,landscape_matrix);
%this loop updates the path cells with their index in the new landscape matrix
    for i = 1:sample_size
path_cells(i)=find(landscape_matrix(:)==path_cells(i));
end
% updating X values according to the new landscape matrix
X=X(find(landscape_matrix(:)))-=min(X(find(landscape_matrix(:))));
% updating Y values according to the new landscape matrix
Y=Y(find(landscape_matrix(:)))-=min(Y(find(landscape_matrix(:))));
% habitat quality 1
habitat_quality_1=habitat_quality_1(find(landscape_matrix(:)));
% habitat quality 2
habitat_quality_2=habitat_quality_2(find(landscape_matrix(:)));
% creating a rectangular matrix containing all cells indices from 1 to the number of cells in the newly defined landscape along dimensions n and m.
landscape_matrix=uint32(reshape(1:length(landscape_matrix(:)),size(landscape_matrix,1),size(landscape_matrix,2)));

save(new_filename)
end
end
end
end

% folder and filename: data processing\sub_landscape_calculator.m

% this function returns an rectangular chunk of the landscape containing all cells within a given range of some given location/s
function [sub_landscape_matrix] = sub_landscape_calculator(x,y,range,cell_dim,landscape_matrix)

orthogonal_x_conversion_factor=cell_dim*sqrt(3)/2; % as the simulation operates on a hexagonal grid, the horizontal distance between neighboring cells is smaller than the euclidian distance
x_side_length=size(landscape_matrix,1);
y_side_length=size(landscape_matrix,2);

% define the four corner cells of the sub-landscape surrounding the input point/s
min_row_index=floor((min(x)-range)/orthogonal_x_conversion_factor)-1;
if min_row_index<1
    min_row_index=1;
end
max_row_index=ceil((max(x)+range)/orthogonal_x_conversion_factor)+1;
if max_row_index>x_side_length
    max_row_index=x_side_length;
end
min_col_index=floor((min(y)-range)/cell_dim)-1;
if min_col_index<1
    min_col_index=1;
end
max_col_index=ceil((max(y)+range)/cell_dim)+1;
if max_col_index>y_side_length
    max_col_index=y_side_length;
end

% sub_landscape_matrix=landscape_matrix(min_row_index:max_row_index,min_col_index:max_col_index);
% folder and filename: calibration\mcmc_module.m

function [chains acceptance_rate median_NLL] = mcmc_module(steps_number,path_cells,step_durations,last_training_step,X,Y,habitat_quality_1,habitat_quality_2,...
cell_dim,landscape_matrix,max_step_length,free_parameters,param_values,cov_matrix,boundries,chain_length)

[prior_loglikelihood]=prior_likelihood_generator(param_values,boundries);
[loglikelihood]=likelihood_generator(param_values,steps_number,path_cells,step_durations,last_training_step,X,Y,...
habitat_quality_1,habitat_quality_2,cell_dim,landscape_matrix,max_step_length,boundries);
if ((loglikelihood+prior_loglikelihood)==-inf) || isnan(loglikelihood+prior_loglikelihood)
    error('initial values are invalid');
end

%initialise the chains array
chains=NaN(chain_length,free_parameters);
chains(1,:)=param_values;
%initialise the random number generator seed
rand('twister', sum(100*clock));
%generate a vector of white noise used as a random acceptance criteria
uniform_randoms=rand(chain_length*free_parameters,1);
%generate a matrix of noise according a multivariate normal distribution.
these residuals are used to produce new proposed values at each iteration
residuals=mvnrnd(zeros(chain_length,free_parameters),(cov_matrix./10));
%count the number of proposed parameter combinations that are integrated into
the chains (i.e., have high enough likelihood)
acceptance=1;
for i=1:chain_length-1
    %the proposal values are variations of the previous values in the chains
    proposed_values=chains(i,:)+residuals(i,:);
    [proposal_prior_loglikelihood]=prior_likelihood_generator(proposed_values,boundries);
    %if one of the proposal values is outside its prior boundaries the proposals
likelihood is zero and the proposal is rejected
    if proposal_prior_loglikelihood==-inf
        chains(i+1,:)=chains(i,:);
    else
        %generate a loglikelihood value for the proposal
        [proposal_loglikelihood]=likelihood_generator(proposed_values,steps_number,path_cells,step_durations,last_training_step,X,Y,...
habitat_quality_1,habitat_quality_2,cell_dim,landscape_matrix,max_step_length,boundries);
        %calculate the ratio between the current and previous likelihoods
        posterior_ratio=exp((proposal_loglikelihood+proposal_prior_loglikelihood)-(loglikelihood+prior_loglikelihood));
%if the likelihood of the proposal is higher than that of the previous parameter combination, the proposed values are accepted. otherwise, the probability of acceptance is proportional to the posterior ratio

if (posterior_ratio>=uniform_randoms(i))
    acceptance=acceptance+1;
    chains(i+1,:)=proposed_values;
    prior_loglikelihood=proposal_prior_loglikelihood;
    loglikelihood=proposal_loglikelihood;
else
    %if the proposed value were not accepted the chain is updated by the previous values
    chains(i+1,:)=chains(i,:);
end
end

%the acceptance rate is the proportion of accepted proposals throughout the MCMC run and it depends on magnitude of variance-covariance values. for best results the acceptance rate should be around 40%-50%
acceptance_rate=acceptance./chain_length;

[median_prior_loglikelihood]=prior_likelihood_generator(median(chains),boundries);
[median_loglikelihood]=likelihood_generator(median(chains),steps_number,path_cells,step_durations,last_training_step,X,Y,...
habitat_quality_1,habitat_quality_2,cell_dim,landscape_matrix,max_step_length,boundries);
median_NLL=-(median_prior_loglikelihood+median_loglikelihood);

% folder and filename: calibration\momentg.m

function results = momentg(draws)
% PURPOSE: computes Gewke's convergence diagnostics NSE and RNE (numerical std error and relative numerical efficiencies)
% USAGE: result = momentg(draws)
% where: draws = a matrix of Gibbs draws (ndraws x nvars)
% RETURNS: a structure result:
% result.meth = 'momentg'
% result.ndraw = # of draws
% result.nvar = # of variables
% result(i).pmean = posterior mean for variable i
% result(i).pstd = posterior std deviation
% result(i).nse = nse assuming no serial correlation for variable i
% result(i).rne = rne assuming no serial correlation for variable i
% result(i).nse1 = nse using 4% autocovariance tapered estimate
% result(i).rne1 = rne using 4% autocovariance taper
% result(i).nse2 = nse using 8% autocovariance taper
% result(i).rne2 = rne using 8% autocovariance taper
% result(i).nse3 = nse using 15% autocovariance taper
% result(i).rne3 = rne using 15% autocovariance taper
% SEE ALSO: coda(), apm()
% REFERENCES: Geweke (1992), 'Evaluating the accuracy of sampling-based
% approaches to the calculation of posterior moments', in J.O. Berger,
% J.M. Bernardo, A.P. Dawid, and A.F.M. Smith (eds.) Proceedings of
% the Fourth Valencia International Meeting on Bayesian Statistics,
% pp. 169-194, Oxford University Press
% Also: 'Using simulation methods for Bayesian econometric models:
% Inference, development and communication', at: www.econ.umn.edu/~bacc
% written by:
% James P. LeSage, Dept of Economics
% University of Toledo
% 2801 W. Bancroft St,
% Toledo, OH 43606
% jpl@jpl.econ.utoledo.edu
% NOTE: this code draws heavily on MATLAB programs written by
% Siddartha Chib available at: www.econ.umn.edu/~bacc
% I have repackaged it to make it easier to use.

[ndraw nvar] = size(draws);
results.ndraw = ndraw;
results.meth = 'momentg';
results.nvar = nvar;

NG=100;
if ndraw < NG
error('momentg: needs a larger number of ndraws');
end

ntaper=[4 8 15];
ns = floor(ndraw/NG);
ns*NG;

for jf = 1:nvar; % loop over all variables
    cnt = 0;
    cn = zeros(NG);
    cd = zeros(NG);
    cdn = zeros(NG);
    cdd = zeros(NG);
    cnn = zeros(NG);
    cvar = zeros(NG);

    % form sufficiency statistics needed below
    td=0; tn=0; tdd=0; tnn=0; tdn=0; tvar=0;
    for ig=1:NG;
        gd=0; gn=0; gdd=0; gdn=0; gnn=0; gvar=0;
        for is = 1:ns;
            cnt = cnt + 1;
            g = draws(cnt,jf);
            ad = 1;
            an = ad*g;
            gd = gd+ad;
\[ gn = gn + an; \]
\[ gdn = gdn + ad*an; \]
\[ gdd = gdd + ad*ad; \]
\[ gnn = gnn + an*an; \]
\[ gvar = gvar + an*g; \]
\[ \text{end}; \quad \% \text{end of for is} \]

\[ td = td + gd; \]
\[ tn = tn + gn; \]
\[ tdn = tdn + gdn; \]
\[ tdd = tdd + gdd; \]
\[ tnn = tnn + gnn; \]
\[ tvar = tvar + gvar; \]

\[ cn(ig) = gn/ns; \]
\[ cd(ig) = gd/ns; \]
\[cdn(ig) = gdn/ns; \]
\[ cdd(ig) = gdd/ns; \]
\[ cnn(ig) = gnn/ns; \]
\[ cvar(ig) = gvar/ns; \]
\[ \text{end}; \quad \% \text{for ig} \]

\[ eg = tn/td; \]
\[ varg = tvar/td - eg^2; \]
\[ sdg = -1; \]
\[ \text{if} \ (varg>0); \ sdg = sqrt(varg); \text{end}; \]
\[ \% \text{save posterior means and std deviations to results structure} \]
\[ \text{results(jf).pmean} = eg; \]
\[ \text{results(jf).pstd} = sdg; \]

\[ \% \text{numerical standard error assuming no serial correlation} \]
\[ \text{varnum} = (tnn-2*eg*tdn+tdd*eg^2)/(td^2); \]
\[ \text{sdnum} = -1; \quad \text{if} \ (\text{varnum}>0); \ sdnum = sqrt(\text{varnum}); \text{end}; \]
\[ \% \text{save to results structure} \]
\[ \text{results(jf).nse} = \text{sdnum}; \]
\[ \text{results(jf).rne} = \text{varg}/(\text{nuse}*\text{varnum}); \]

\[ \% \text{get autocovariance of grouped means} \]
\[ \text{barn} = \text{tn/nuse}; \]
\[ \text{bard} = \text{td/nuse}; \]
\[ \text{for} \ ig=1:\text{NG}; \]
\[ \quad \text{cn}(ig) = \text{cn}(ig) - \text{barn}; \]
\[ \quad \text{cd}(ig) = \text{cd}(ig) - \text{bard}; \]
\[ \text{end}; \]
\[ \% \text{index 0 not allowed, \ lag+1 stands for lag} \]
\[ \text{rnn}(lag+1) = \text{ann}/\text{NG}; \]
\[ \text{rdd}(lag+1) = \text{add}/\text{NG}; \]
\[ \text{rnd}(lag+1) = \text{and}/\text{NG}; \]
\[ \text{rdn}(lag+1) = \text{adn}/\text{NG}; \]
end; %lag

% numerical standard error with tapered autocovariance functions
for mm=1:3;
    m=ntaper(mm);
    am=m;
    snn=rnn(1); sdd=rdd(1); snd=rnd(1);
    for lag=1:m-1;
        att=1-lag/am;
        snn=snn+2*att*rnn(lag+1);
        sdd=sdd+2*att*rdd(lag+1);
        snd=snd+att*(rnd(lag+1) + rnd(lag+1));
    end; %lag
    varnum=ns*nuse*(snn-2*eg*snd+sdd*eg^2)/(td^2);
    sdnum=-1;
    if (varnum>0); sdnum=sqrt(varnum); end;
end; % end of for mm loop
end; % end of loop over variables

% folder and filename: calibration\prior_likelihood_generator.m

function [sum_loglikelihood]=prior_likelihood_generator(param_values,boundries)

likelihoods=NaN(length(param_values),1);

adj_param_values=(param_values-boundries(2,:))/(boundaries(1,:)-boundries(2,:));
%The prior for the default habitat value is distributed Beta:
likelihoods(1:2)=betapdf(adj_param_values(1:2),2,2);
%The prior for the sensory attenuation is 0.5 when between 0 and 1, 1 when between 1 and 2 and 0 elsewhere.
if (adj_param_values(3)>2)||(adj_param_values(3)<0)
    likelihoods(3)=0;
elseif adj_param_values(3)>1
    likelihoods(3)=1;
else
    likelihoods(3)=0.5;
end

end
end
%the prior for the memory decay is 0.5 when between 0 and 1, 1 when between 1 and 2 or -1 and 0, and 0 elsewhere.
if (adj_param_values(4)>2)||(adj_param_values(4)<-1)
    likelihoods(4)=0;
elseif adj_param_values(4)>1||(adj_param_values(4)<0)
    likelihoods(4)=1;
else
    likelihoods(4)=0.5;
end
%the prior for attraction is Normal
likelihoods(5:6)=normpdf(param_values(5:6),0,5);
%the prior for the travelcost is 1 when between 0 and 1, and 0 Elsewhere.
if (adj_param_values(7)>1)||(adj_param_values(7)<0)
    likelihoods(7)=0;
else
    likelihoods(7)=1;
end
sum_loglikelihood=sum(log(likelihoods));

% folder and filename: calibration\sub_landscape_calculator.m

% this function returns an rectangular chunk of the landscape containing all cells within a given range of some given location/s
function [sub_landscape_matrix] = sub_landscape_calculator(x,y,range,cell_dim,landscape_matrix)

    orthogonal_x_conversion_factor=cell_dim*sqrt(3)/2;%as the simulation operates on a hexegonal grid, the horizontal distance between neighoring cells is smaller than the euclidian distance
    x_side_length=size(landscape_matrix,1);
    y_side_length=size(landscape_matrix,2);

    % define the four corner cells of the sub-landscape surrounding the input point/s
    min_row_index=floor((min(x)-range)/orthogonal_x_conversion_factor)-1;
    if min_row_index<1
        min_row_index=1;
    end
    max_row_index=ceil((max(x)+range)/orthogonal_x_conversion_factor)+1;
    if max_row_index>x_side_length
        max_row_index=x_side_length;
    end
    min_col_index=floor((min(y)-range)/cell_dim)-1;
    if min_col_index<1
        min_col_index=1;
    end
    max_col_index=ceil((max(y)+range)/cell_dim)+1;
    if max_col_index>y_side_length
        max_col_index=y_side_length;
    end

    %
sub_landscape_matrix=landscape_matrix(min_row_index:max_row_index,min_col_index:max_col_index);

% folder and filename: calibration\apm.m

function result = apm(results1,results2)
% PURPOSE: computes Geweke's chi-squared test for two sets of MCMC sample draws
% ---------------------------------------------------------------
% USAGE: result = apm(results1,results2)
% where: results1 = a structure returned by momentg
% results2 = a structure returned by momentg
% ---------------------------------------------------------------
% RETURNS: a structure:
% results.meth  = 'apm'
% results.ndraw = ndraw1+ndraw2
% results.nvar  = # of variables
% results.p1    = ndraw1/(ndraw1+ndraw2)
% results.p2    = ndraw2/(ndraw1+ndraw2)
% results(i).pmean(k)  = posterior mean for variable i
% for k = nse, nse1,nse2,nse3
% results(i).nse(k)    = nse for variable i
% for k = nse, nse1,nse2,nse3
% results(i).prob(k)   = chi-sq test prob for variable i
% for k = nse, nse1,nse2,nse3
% ---------------------------------------------------------------
% SEE ALSO: coda(), prt()
% ---------------------------------------------------------------
% Also: 'Using simulation methods for Bayesian econometric models: Inference, development and communication', at: www.econ.umn.edu/~bacc
% ---------------------------------------------------------------

% written by:
% James P. LeSage, Dept of Economics
% University of Toledo
% 2801 W. Bancroft St,
% Toledo, OH 43606
% jpl@jpl.econ.utoledo.edu

% NOTE: this code draws heavily on MATLAB programs written by Siddartha Chib available at: www.econ.umn.edu/~bacc
% I have repackaged it to make it easier to use.

if ~isstruct(results1)
    error('apm: requires a structure from momentg as input');
elseif ~isstruct(results2)
    error('apm: requires a structure from momentg as input');
end;
nvar = results1(1).nvar;
nvar2 = results2(1).nvar;
if nvar ~= nvar2;
    error('apm: structure arguments have different # of variables');
end;

ndraw1 = results1(1).ndraw;
ndraw2 = results2(1).ndraw;
result.p1 = ndraw1/(ndraw1+ndraw2);
result.p2 = ndraw2/(ndraw1+ndraw2);
result.ndraw = ndraw1+ndraw2;
result.meth = 'apm';
result.nvar = nvar;

ng = nvar;
f = 2;

% pull out information
for i=1:nvar;
    j=1;
    g(j,i) = results1(i).pmean;
sdnum1(j,i) = results1(i).nse;
sdnum2(j,i) = results1(i).nse1;
sdnum3(j,i) = results1(i).nse2;
sdnum4(j,i) = results1(i).nse3;
    j=2;
    g(j,i) = results2(i).pmean;
sdnum1(j,i) = results2(i).nse;
sdnum2(j,i) = results2(i).nse1;
sdnum3(j,i) = results2(i).nse2;
sdnum4(j,i) = results2(i).nse3;
end;

for i=1:nvar;
    for k=1:4;
        eg=0; nse=0; wtsum=0;
        if k==1; sdnum=sdnum1;
            elseif k==2; sdnum=sdnum2;
            elseif k==3; sdnum=sdnum3;
            elseif k==4; sdnum=sdnum4;
        end;
        gvar=zeros(nf-1);
        for j=1:nf;
            eg=eg+g(j,i)/(sdnum(j,i))^2;
            wtsum=wtsum+1/(sdnum(j,i))^2;
        end;
        eg=eg/wtsum;
        nse=1/sqrt(wtsum);
        for j=1:nf-2;
            gvar(j,j)=(sdnum(j,i))^2+(sdnum(j+1,i))^2;
            gvar(j,j+1)=-(sdnum(j+1,i))^2;
            gvar(j+1,j)=gvar(j,j+1);
        end;
        gvar(nf-1,nf-1)=(sdnum(nf-1,i))^2+(sdnum(nf,i))^2;
        ginv=inv(gvar);
    end;
end;
gl = g(1:nf-1,i); g2 = g(2:nf,i);
cstat = (g2 - gl)' * ginv * (g2 - gl);
df = nf - 1;
p = 1 - chis_prb(cstat, df);
result(i).pmean(k) = eg;
result(i).nse(k) = nse;
result(i).prob(k) = p;
end;
end;

% folder and filename: calibration\calibration_module.m

clear;
% the number of parameters to be calibrated
free_parameters = 7;
% the MCMC sample size for the initial chain (each link in the chain holds a sample of the values of all free parameters)
initial_chain_length = 1000;
% the full MCMC sample size
chain_length = 20000;

% load and construct all necessary data structures
for realisation = 1:20
% the data file includes: X, Y, cell_dim, habitat_quality_1, habitat_quality_2, landscape_matrix, max_step_length, path_cells, sample_size and step_duration
    filename = ['data_' num2str(realisation) '_1_1.mat'];
    load(filename);
    filename = ['results_' num2str(realisation) '_1_1']
% the first half of the sample is used to construct the cognitive landscape and the second half is used for the calibration process.
    last_training_step = round(sample_size/2);

% initialise the parameters and variance-covariance matrices
    initial_values = zeros(1, free_parameters);
    cov_matrix = zeros(free_parameters, free_parameters);
    boundries = zeros(2, free_parameters);

% the parameter index in the parameters' vector
    prior_habitat_quality_1 = 1;
    prior_habitat_quality_2 = 2;
% the first value sent to the MCMC module
    initial_values(prior_habitat_quality_1) = 0.5;
    initial_values(prior_habitat_quality_2) = 0.5;
% the variance of the Gaussian residual distribution used to generate the MCMC. At this stage all covariance terms are null as there is no prior knowledge regarding the covariance structure
    cov_matrix(prior_habitat_quality_1, prior_habitat_quality_1) = 0.01;
    cov_matrix(prior_habitat_quality_2, prior_habitat_quality_2) = 0.01;
% the boundreis of the beta prior used in the MCMC proposal
    boundries(:, prior_habitat_quality_1:prior_habitat_quality_2) = [1 1; 0 0];
sensory_decay_coefficient = 3;
% for both the sensory decay and memory decay coefficients any value larger than 1 is handled as inf (i.e., no sensory and memory abilities)
    initial_values(sensory_decay_coefficient)=0.01;
    cov_matrix(sensory_decay_coefficient,sensory_decay_coefficient)=0.001;
% the boundaries of the uniform prior used in the MCMC proposal. the upper boundary is set to twice the actual upper boundary to allow convergence to inf
    boundries(:,sensory_decay_coefficient)=[1; 0];

    memory_decay_coefficient=4;
% for the memory decay coefficient any value smaller than 0 is handled as 0 (i.e., no memory decay)
    initial_values(memory_decay_coefficient)=0.01;
    cov_matrix(memory_decay_coefficient,memory_decay_coefficient)=0.001;
% the boundaries of the uniform prior used in the MCMC proposal. the lower boundary is set to twice the actual lower boundary to allow convergence to 0
    boundries(:,memory_decay_coefficient)=[1; 0];

    enhancement_coefficient_1=5;
    enhancement_coefficient_2=6;
% the enhancement coefficient can have both negative (in case of repulsion) and positive (in case of attraction) values. a null value results in the cognitive algorithm being deactivated and the model operating as a pure random walk.
    initial_values(enhancement_coefficient_1)=0;
    initial_values(enhancement_coefficient_2)=0;
    cov_matrix(enhancement_coefficient_1,enhancement_coefficient_1)=0.1;
    cov_matrix(enhancement_coefficient_2,enhancement_coefficient_2)=0.1;
% the boundaries of the Beta prior used in the MCMC proposal
    boundries(:,enhancement_coefficient_1:enhancement_coefficient_2)=[inf, inf; -inf, -inf];

    travel_cost_coefficient=7;
    initial_values(travel_cost_coefficient)=0.01;
    cov_matrix(travel_cost_coefficient,travel_cost_coefficient)=0.001;
% the boundaries of the uniform prior used in the MCMC proposal
    boundries(:,travel_cost_coefficient)=[1; 0];

% initial (shorter) MCMC chains are generated so as to populate the covariance matrix
% the acceptance rate is the proportion of accepted proposals throughout the MCMC run and it depends on magnitude of variance-covariance values.
% for best results the acceptance rate should be around 40%-50%
    acceptance_rate=0;
    while (acceptance_rate<0.01)||(acceptance_rate>0.7)
        [chains acceptance_rate]=mcmc_module(sample_size,path_cells,step_durations,last_training_step,X,Y,habitat_quality_1,habitat_quality_2,...
            cell_dim,landscape_matrix,max_step_length,free_parameters,initial_values,cov_matrix,boundries,initial_chain_length);
        if acceptance_rate<0.01
            cov_matrix=cov_matrix./2;
        elseif acceptance_rate>0.7
            cov_matrix=2.*cov_matrix;
    end
end

%the initial values and the variance-covariance matrix are updated based on the previous chain
initial_values=chains(end,:);
cov_matrix=cov(chains);
acceptance_rate=0;
while (acceptance_rate<0.1)||(acceptance_rate>0.7)
    [chains acceptance_rate]=mcmc_module(sample_size,path_cells,step_durations,last_training_step,X,Y,habitat_quality_1,habitat_quality_2,...
cell_dim,landscape_matrix,max_step_length,free_parameters,initial_values,cov_matrix,boundries,initial_chain_length);
if acceptance_rate<0.1
    cov_matrix=cov_matrix./2;
elseif acceptance_rate>0.7
    cov_matrix=2.*cov_matrix;
end
initial_values=chains(end,:);
cov_matrix=cov(chains);

%%

%the counter keeps track of the number of iteration
counter=0;
%the number of converged parameters
converged=0;
%This loop will continue until the MCMC has converged or until the maximum number of iterations is reached
while (converged<free_parameters)&&(counter<10)
    [chains acceptance_rate]
median_NLL=mcmc_module(sample_size,path_cells,step_durations,last_training_step,X,Y,habitat_quality_1,habitat_quality_2,...
cell_dim,landscape_matrix,max_step_length,free_parameters,initial_values,cov_matrix,boundries,chain_length);
%the initial values and the variance-covariance matrix are updated based on the previous chain
initial_values=chains(end,:);
cov_matrix=cov(chains);
%the acceptance rate is the proportion of accepted proposals throughout the MCMC run and it depends on magnitude of variance-covariance values.
%for best results the acceptance rate should be around 40%-50%
if acceptance_rate<0.1
    cov_matrix=cov_matrix./2;
elseif acceptance_rate>0.7
    cov_matrix=2.*cov_matrix;
else
    counter=counter+1;
    converged=0;
if the acceptance rate is acceptable, test for convergence; the test is based on comparing the first 20% of the chain with the last 50%
first_batch_results=momentg(chains(1:round(0.2*chain_length),:));
second_batch_results = momentg(chains(round(0.5*chain_length):end,:));
results=apm(first_batch_results,second_batch_results);
for
    i=1:free_parameters
        converged=converged+(results(i).prob(2)>0.05);
    end
end
converged
counter
save(filename);
end
end
plot(chains./repmat(mean(chains),length(chains),1))

% folder and filename: calibration\chis_prb.m

function prob = chis_prb(x,v)
% PURPOSE: computes the chi-squared probability function
%------------------------------------------------------------------
% USAGE: prob = chis_prb(x,v)
% where: x = the value to test
%       (may be a matrix size(x), or a scalar)
%       v = the degrees of freedom
%       (may be a matrix size(x), or a scalar)
%------------------------------------------------------------------
% RETURNS:
% prob = the probability of observing a chi-squared
% value <= x, i.e., prob(x | v).
%------------------------------------------------------------------
% SEE ALSO: chis_d, chis_pdf, chis_cdf, chis_inv, chis_rnd
%------------------------------------------------------------------

% written by:
% James P. LeSage, Dept of Economics
% University of Toledo
% 2801 W. Bancroft St,
% Toledo, OH 43606
% jlesage@spatial-econometrics.com

if nargin ~= 2
    error('Wrong # of arguments to chis_prb');
end;
prob = gammainc(x/2, v/2);

% folder and filename: calibration\likelihood_generator.m

function [sum_loglikelihood] =
likelihood_generator(param_values,steps_number,path_cells,step_durations,last_training_step,...
%% Parameterising the simulation
%the default value for the quality of any location in the landscape is used
when no other information is available (values must be positive numerics;
%units are the same as for the landscape variable). High values will lead to
an over estimation of habitat quality when perceptual or memory information
%are lacking (i.e., assuming a hospitable surroundings). Low values will lead
to an under estimation of habitat quality when perceptual or memory
%information is lacking (i.e., assuming a hostile surroundings).
prior_habitat_quality_1=param_values(1);
prior_habitat_quality_2=param_values(2);

% the sensory input from the landscape declines as an exponential function of
the Euclidian distance yet increases with exposure time. 
% in a single instant, the amount of information perceived by the animal is
some fraction [0,1] of the "truth". The sensory decay
% coefficient determines the exponential rate of information reduction with
distance (units are hours/meters; values must be positive numerics;
% inf corresponds to no sensory input).
if param_values(3)>=boundries(1,3)
   sensory_decay_coefficient=inf;
else
   sensory_decay_coefficient=param_values(3);
end

% landscape information stored in memory decay as an exponential function of
time; in a single instant, the amount of information retained
% is some fraction [0,1] of the information held in the previous instant. The
memory decay coefficient determines the exponential rate of information
% reduction with time (units are 1/hours; values must be positive numerics; 0
% corresponds to no memory lost; inf corresponds to no memory input).
if param_values(4)>=boundries(1,4)
   memory_decay_coefficient=inf;
elseif param_values(4)<=boundaries(2,4)
   memory_decay_coefficient=0;
else
   memory_decay_coefficient=param_values(4);
end

% the perceived landscape may be further manipulated in the cognitive system
by enhancing or reducing the magnitude of the spatial differences;
% negative values correspond to repulsion rather than attraction; values
between -1 and 1 decrease the perceived quality differences while values
% larger than 1 or smaller than -1 enhance the perceived differences in
habitat quality.
if sensory_decay_coefficient==inf
   attraction_coefficient_1=0;
   attraction_coefficient_2=0;
else
   % if the maximum gradient across the landscape is less then 1%, the attraction
   % is set to 0
enhanced_habitat_quality_1=habitat_quality_1((habitat_quality_1~=0)&(habitat_quality_1~=1)).^param_values(5);
  max_1=max(enhanced_habitat_quality_1);
  min_1=min(enhanced_habitat_quality_1);
  max_gradient_1=(max_1-min_1)/max_1;
  clear enhanced_habitat_quality_1
  if (max_gradient_1<0.01)
    attraction_coefficient_1=0;
  else
    attraction_coefficient_1=param_values(5);
  end

enhanced_habitat_quality_2=habitat_quality_2((habitat_quality_2~=0)&(habitat_quality_2~=1)).^param_values(6);
  max_2=max(enhanced_habitat_quality_2);
  min_2=min(enhanced_habitat_quality_2);
  max_gradient_2=(max_2-min_2)/max_2;
  clear enhanced_habitat_quality_2
  if (max_gradient_2<0.01)
    attraction_coefficient_2=0;
  else
    attraction_coefficient_2=param_values(6);
  end
end

% Assuming that movement is costly and that this cost increases with distance and decreases with time, the travel cost coefficient defines the shape of the decline in profitability with distance (values must be positive numerics; a null value results in a null effect).
travel_cost_coefficient=param_values(7);

% Initializing the simulation
% this array will contain the probabilities assigned to each step along the path (updated stepwise):
likelihoods=NaN(1,steps_number);
% the probability of the first location is, by definition, 1
likelihoods(1)=1;
% initiating the current location
focal_cell=0;
current_step_duration=0;

% if no cognitive capacities exist, only a simplified version of the algorithm is used - a simple random walk, otherwise the full algorithm is used
if attraction_coefficient_1||attraction_coefficient_2
% initialising the cognitive landscapes (the entire landscape as it is perceived in the animal's mind)
initial_cognitive_landscape_1=ones(length(landscape_matrix(:,)),1).*prior_habitat_quality_1;
cognitive_landscape_1=initial_cognitive_landscape_1;
initial_cognitive_landscape_2=ones(length(landscape_matrix(:,)),1).*prior_habitat_quality_2;
cognitive_landscape_2=initial_cognitive_landscape_2;
max_percep_dist=-log(0.01)/sensory_decay_coefficient;

%first, the memorised map has to be initialised; this loop runs over an
initial segment of the path to generate an initial cognitive map
if memory_decay_coefficient<inf
memorised_cells=[];
for step=1:last_training_step
%update distances and sensed qualities in all perceived cells in case of a
change in location or step duration
if (focal_cell~=path_cells(step))||(step_durations(step)~=current_step_duration)
    moved=true;
    if focal_cell~=path_cells(step)
        focal_cell=path_cells(step);
    end
    if step_durations(step)~=current_step_duration
        current_step_duration=step_durations(step);
%the maximal perceptual distance is set according to the values of the
sensory decay coefficients and the current step duration.
%the algorithm assumes no sensory information is perceived beyond this
distance
    current_max_percep_dist=(current_step_duration*max_percep_dist) + cell_dim;
%the memory loads are the fraction of the cognitive landscape that is
remembered in the next time step
    memory_load=exp(-memory_decay_coefficient.*current_step_duration);
    end
%define the current coarse landscape window relevant for perception
    percived_cells=sub_landscape_calculator(X(focal_cell),Y(focal_cell),current_max_percep_dist,cell_dim,landscape_matrix);
%calculate the Euclidian distances between the current location and all other
cells and refine the selection
    distances=sqrt((X(percived_cells(:))-X(focal_cell)).^2 +
(Y(percived_cells(:))-Y(focal_cell)).^2);
    within_percep_bounds=(distances<current_max_percep_dist);
    percived_cells=percived_cells(within_percep_bounds);
%the distances to all percived cells
    percived_cells_distances=distances(within_percep_bounds);
%to avoid assuming perfect (i.e., 100%) sensory information input from the
current location, the distance of the focal cell is set to
%half way from the center to the edge
    percived_cells_distances(percived_cells==focal_cell)=cell_dim/4;
%the information load is the fraction of information perceived by the animal
during a given time step from each of the spatial cells
%in the landscape and it is proportional to the step duration
    information_loads=exp(-sensory_decay_coefficient.*percived_cells_distances./current_step_duration);
if attraction_coefficient_1
% the sensed (as opposed to remembered) quality of each cell is a product of the information load (a fraction) and the real quality of the cell

sensed_qualities_1 = information_loads.*habitat_quality_1(perceived_cells);
end
if attraction_coefficient_2
sensed_qualities_2 = information_loads.*habitat_quality_2(perceived_cells);
end

% the cognitive landscape is the landscape as it is currently perceived by the animal and it is updated at each time step by the cognitive algorithm. The cognitive landscape is a product of viewing the habitat quality via the "lenses" of sensory and memory inputs. This algorithm was inspired by ideas formulated in Bush 1951, McNamara and Houston 1987, Bernstein 1988 and Lange and Dukas 2009.

% what the animal knows about a specific cell is the sum of what it senses and what it remembers; these are then complimented by a default value and modified by the perceived depletion
if attraction_coefficient_1
    if memory_load < 1
        cognitive_landscape_1(memorised_cells) = memory_load.*cognitive_landscape_1(memorised_cells) + (1-memory_load).*prior_habitat_quality_1;
    end
    cognitive_landscape_1(perceived_cells) = sensed_qualities_1 + ((1-information_loads).*cognitive_landscape_1(perceived_cells));
end
if attraction_coefficient_2
    if memory_load < 1
        cognitive_landscape_2(memorised_cells) = memory_load.*cognitive_landscape_2(memorised_cells) + (1-memory_load).*prior_habitat_quality_2;
    end
    cognitive_landscape_2(perceived_cells) = sensed_qualities_2 + ((1-information_loads).*cognitive_landscape_2(perceived_cells));
end

% if the animal moved, the memorised cells list needs updating
if moved
    moved = false;
    memorised_cells = unique([memorised_cells percived_cells']);
end
end

% this loop runs over the later segment of the path to calculate the likelihood of the model
focal_cell = 0;
for step = last_training_step:steps_number-1
% update distances to all perceived cells and available cells in case of a change in location or step duration
    if (focal_cell ~= path_cells(step)) || (step_durations(step) ~= current_step_duration)
moved=true;
if focal_cell~=path_cells(step)
    focal_cell=path_cells(step);
end
if step_durations(step)~=current_step_duration
    current_step_duration=step_durations(step);
end
% the maximal perceptual distance is set according to the values of the sensory decay coefficients and the current step duration.
% the algorithm assumes no sensory information is perceived beyond this distance

current_max_percep_dist=(current_step_duration*max_percep_dist) + cell_dim;
% the memory loads are the fraction of the cognitive landscape that is remembered in the next time step
memory_load=exp(-memory_decay_coefficient.*current_step_duration);
end
% if the maximal perceptual distance is larger than the maximal relocation distance, the available landscape is a sub-set of
% the perceived landscape and vice-verse
if current_max_percep_dist>max_step_length
% define the coarse landscape window currently relevant for perception
percived_cells=sub_landscape_calculator(X(focal_cell),Y(focal_cell),current_max_percep_dist,cell_dim,landscape_matrix);
% calculate the Euclidian distances between the current location and all other cells and refine the selection
distances=sqrt((X(percived_cells(:))-X(focal_cell)).^2 + (Y(percived_cells(:))-Y(focal_cell)).^2);
within_percep_bounds=(distances<current_max_percep_dist);
percived_cells=percived_cells(within_percep_bounds);
% the distances to all perceived cells
percived_cells_distances=distances(within_percep_bounds);
clear within_percep_bounds distances
% define the cells currently available for relocation
within_availability_bounds=(percived_cells_distances<=max_step_length);
available_cells=percived_cells(within_availability_bounds);
% the distances to all currently available cells
available_cells_distances=percived_cells_distances(within_availability_bounds);
clear within_availability_bounds
else
% define the coarse landscape window currently available for relocation
available_cells=sub_landscape_calculator(X(focal_cell),Y(focal_cell),max_step_length,cell_dim,landscape_matrix);
% calculate the Euclidian distances between the current location and all other cells and refine the selection
distances=sqrt((X(available_cells(:))-X(focal_cell)).^2 + (Y(available_cells(:))-Y(focal_cell)).^2);
within_availability_bounds=(distances<=max_step_length);
available_cells=available_cells(within_availability_bounds);
% the distances to all currently available cells
available_cells_distances=distances(within_availability_bounds);
clear within_availability_bounds distances
%define the cells currently perceived
within_percep_bounds=(available_cells_distances<current_max_percep_dist);
    percived_cells=available_cells(within_percep_bounds);
%the distances to all perceived cells
percived_cells_distances=available_cells_distances(within_percep_bounds);
clear within_percep_bounds
end
%to avoid assuming perfect (i.e., 100%) sensory information input from the current location, the distance of the focal cell is set to %half way from the center to the edge
    percived_cells_distances(percived_cells==focal_cell)=cell_dim/4;
%the information load is the fraction of information perceived by the animal during a given time step from each of the spatial cells %in the landscape and it is proportional to the step duration
information_loads=exp(-
    sensory_decay_coefficient.*percived_cells_distances./current_step_duration);

if attraction_coefficient_1
%the sensed (as opposed to remembered) quality of each cell is a product of the information load (a fraction) %and the real quality of the cell
    sensed_qualities_1=information_loads.*habitat_quality_1(percived_cells);
end

sensed_qualities_2=information_loads.*habitat_quality_2(percived_cells);
end
%travel costs are proportional to distance and decrease with time available for travel (i.e., step duration
    travel_costs_factor=exp(-
    travel_cost_coefficient.*available_cells_distances./current_step_duration);
end

% the cognitive landscape is the landscape as it is currently perceived by the animal and it is updated at each time step by the cognitive algorithm. %the cognitive landscape is a product of viewing the habitat quality via the "lenses" of sensory and memory inputs. This algorithm was inspired %by ideas formulated in Bush 1951, McNamara and Houston 1987, Bernstein 1988 and Lange and Dukas 2009.
what the animal knows about a specific cell is the sum of what it senses and what it remembers; these are then complimented by a default value %and modified by the perceived depletion
if attraction_coefficient_1
    if memory_load
        if memory_load<1
    cognitive_landscape_1(memorised_cells)=memory_load.*cognitive_landscape_1(memorised_cells) + (1-memory_load)*prior_habitat_quality_1;
end
    cognitive_landscape_1(percived_cells)=sensed_qualities_1 + ((1-information_loads).*cognitive_landscape_1(percived_cells));
available_landscape_1 = cognitive_landscape_1(available_cells).^attraction_coefficient_1;
    else
        cognitive_landscape_1(percived_cells) = sensed_qualities_1 + 
        ((1-information_loads).*prior_habitat_quality_1);
    end
    cognitive_landscape_1(percived_cells) = prior_habitat_quality_1;
end

if attraction_coefficient_2
    if memory_load
        if memory_load<1

    end
    cognitive_landscape_2(memorised_cells) = memory_load.*cognitive_landscape_2(memorised_cells) + 
    (1-memory_load)*prior_habitat_quality_2;
    end
    cognitive_landscape_2(percived_cells) = sensed_qualities_2 + 
    ((1-information_loads).*cognitive_landscape_2(percived_cells));
end

available_landscape_2 = cognitive_landscape_2(available_cells).^attraction_coefficient_2;
    else
        cognitive_landscape_2(percived_cells) = sensed_qualities_2 + 
        ((1-information_loads).*prior_habitat_quality_2);
    end
    cognitive_landscape_2(percived_cells) = prior_habitat_quality_2;
end

% if the animal moved, the memorised cells list needs updating
if moved
    moved=false;
    memorised_cells=unique([memorised_cells percived_cells']);
end

% integrating the two landscape variables into a single perceived map of available cells
if attraction_coefficient_1&&attraction_coefficient_2
    available_landscape=available_landscape_1.*available_landscape_2;
elseif attraction_coefficient_1
    available_landscape=available_landscape_1;
else
    available_landscape=available_landscape_2;
end

% the attraction landscape is a function of the integrated landscape and the
decline in attractiveness due to travel costs:
% traveling is costly and this cost increases with the distance traveled and
decreases with the time available for traveling.
attraction_landscape=available_landscape.*travel_costs_factor;
clear available_landscape

%%% the redistribution kernel is produced by standardization of the attraction landscape
probabilities_of_occurrence=attraction_landscape./sum(attraction_landscape);
clear attraction_landscape
% the likelihood of observing the animal in a given cell in the next time step is the value of the redistribution kernel at that cell
likelihoods(step+1)=probabilities_of_occurrence(available_cells==path_cells(step+1));
% to avoid nullifying the entire likelihood chain due to floating point rounding to zero, null values are set to the minimal non-zero value
if likelihoods(step+1)<eps(0)
    likelihoods(step+1)=eps(0);
end
clear probabilities_of_occurrence
end
else
% in case there is no response to the landscape (pure random walk)
    for step=last_training_step:steps_number-1
        if (focal_cell~=path_cells(step))||(step_durations(step)~=current_step_duration)
            if focal_cell~=path_cells(step)
                focal_cell=path_cells(step);
            end
            % define the coarse landscape window currently available for relocation
            available_cells=sub_landscape_calculator(X(focal_cell),Y(focal_cell),max_step_length,cell_dim,landscape_matrix);
            % calculate the Euclidian distances between the current location and all other cells and refine the selection
            distances=sqrt((X(available_cells(:))-X(focal_cell)).^2 + (Y(available_cells(:))-Y(focal_cell)).^2);
            within_availability_bounds=(distances<=max_step_length);
            available_cells=available_cells(within_availability_bounds);
            % the distances to all currently available cells
            available_cells_distances=distances(within_availability_bounds);
            clear within_availability_bounds distances
        end
        if step_durations(step)~=current_step_duration
            current_step_duration=step_durations(step);
        end
    % the only factor determining the attractiveness of the different landscape cells is the traveling costs (i.e., distance and time)
    attraction_landscape=exp(-travel_cost_coefficient.*available_cells_distances./current_step_duration);
% the redistribution kernel is produced by standardization of the attraction landscape
    probabilities_of_occurrence=attraction_landscape./sum(attraction_landscape);
end
%the likelihood of observing the animal in a given cell in the next time step
is the value of the redistribution kernel at that cell

likelihoods(step+1)=probabilities_of_occurrence(available_cells==path_cells(step+1));
%to avoid nullifying the entire likelihood chain due to floating point
rounding to zero, null values are set to the minimal non-zero value
   if likelihoods(step+1)<eps(0)
      likelihoods(step+1)=eps(0);
   end
end
%the log-likelihood values are summed over all steps other than those used to
initialise the cognitive landscape
sum_loglikelihood=sum(log(likelihoods(~isnan(likelihoods))));
Appendix G (Chapter 3)

Predation risk selection coefficient. Each plot represents a kernel density estimate based on 1000 selection coefficient values from 1000 path simulated using the original, true parameter values (solid lines) or 1000 samples out of the joint posterior distributions (dotted lines) at each of the five temporal and five spatial scales. Dashed gray lines indicate a null value (i.e., apparent indifference to predation risk).