The Dissertation Committee for Colin Robert Addis
certifies that this is the approved version of the following dissertation:

Causes and Consequences of Movement: the Interaction between Foraging and Landscape Patterns

Committee:

Timothy H. Keitt, Supervisor

Michael Daniels

Anthony Di Fiore

Shalene Jha

Lauren Meyers

Causes and Consequences of Movement: the Interaction between Foraging and Landscape Patterns

by

Colin Robert Addis

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

December 2017

To Parker for giving me a place to write

and to my parents for giving me everything else.

Acknowledgments

I would like to thank members of the Keitt Lab, particularly Daniel Mitchell, Andria Salas and Gautham Surya, for their input, as well as Dr. Rebecca Lieser, Dr. Erin Zoller, and Graham Rehrig for their help with editing this manuscript.

I would also like to thank my collaborators: Tony Di Fiore, Andres Link, Laura Abondano, Sara Alvarez, Ana Palma, and everyone at Proyecto Primates and the Tiputini Biodiversity Station (Chapter 3); Shalene Jha (Chapter 2); and my advisor Tim Keitt for his guidance on every part of this work.

Causes and Consequences of Movement: the Interaction between Foraging and Landscape Patterns

Publication No. _____

Colin Robert Addis, Ph.D.
The University of Texas at Austin, 2017

Supervisor: Timothy H. Keitt

Foraging animals must move to search their environment for food. The given pattern of resources on the landscape, in space and time, shapes the resulting movement patterns of foragers: over evolutionary time, search strategies adapt to the environment, but foragers also respond to short-term changes in the resource distribution resulting from, for example, phenology or plant demographics. Foragers can also alter the landscape as they move, either physically or by facilitating gene flow, such that patterns of alteration in the landscape reflect the movement patterns of foragers. This two-way interaction between movement and landscape patterns sets up a feedback loop: the movement of dispersers alters the landscape upon which they likely base their subsequent movement decisions. In this dissertation, we investigate three aspects of this cyclical interaction in two different systems: crop-pollinating solitary bees in agroecosystems; and frugivorous, seed-dispersing primates in neotropical forests. First, we add phenological variation to an existing, spatially explicit simulation model of pollination services on agricultural landscapes in order to examine how mass-flowering events affect plant-pollinator interactions. Simulation results show that mass-flowering phenology benefits both crop pollination and

specialist pollinators, but only when crop phenology and pollinator life-cycles are well synchronized. Second, we examine whether spider monkeys use topography in choosing their foraging routes between fruiting resources by comparing the power of two different movement models—a pure random walk and a model based on elevation—to explain observed foraging movements of spider monkeys in the western Amazon. Our elevation-based model explains the data significantly better than the null model, demonstrating that when given a choice between different routes, monkeys prefer higher-elevation pathways. Finally, in order to understand how spider monkeys shape forest structure through seed dispersal, we present a spatially explicit, agent-based simulation model that links foraging, seed dispersal, and forest growth in a closed cycle. Our results suggest that dispersal acts to condense tree distributions over the long-term, but at intermediate time-scales, it produces network-like landscape patterns that coincide with the spider monkeys' travel routes.

Table of Contents

Acknowledgments v

Abstract vi

List of Tables x

List of Figures xi

Chapter 1. Introduction 1

Chapter 2. Effect of Monoculture Phenology on Native Pollinator Persistence and Pollination Services in Agroecosystems 4

2.1 Introduction . . . 4
2.2 Methods . . . 7
2.2.1 Overview . . . 8
2.2.2 Spatio-Temporal Structure . . . 9
2.2.3 State Variables . . . 11
2.2.4 Processes . . . 11
2.2.5 Summary of Simulated Scenarios . . . 14
2.2.6 Initial Conditions . . . 15
2.2.7 Output and Analysis . . . 15
2.3 Results . . . 18
2.4 Discussion . . . 21

Chapter 3. The Effect of Topography on Path Choice of a Frugivorous Neotropical Primate, *Ateles belzebuth* 26

3.1 Introduction . . . 26
3.2 Methods . . . 30
3.2.1 Study Site and Species . . . 30
3.2.2 Data Processing . . . 32
3.2.3 Analysis . . . 33
3.3 Results . . . 37
3.4 Discussion . . . 38

Chapter 4. Foraging Behavior, Agent-Based Seed Dispersal, and Tropical Forest Community Structure **44**
4.1 Introduction 44
4.2 Methods 48
4.2.1 Model Structure 48
4.2.2 Parameterization 52
4.2.3 Analysis 53
4.3 Results 55
4.4 Discussion 59

Appendices **63**

Appendix A. Pollination Model (Ch. 2) Source Code **64**
A.1 model.cpp 64
A.2 stats.hpp 66
A.3 defs.hpp 68

Appendix B. Derivation of Equations (Ch. 3) **79**

Appendix C. Methods for Defining Inter-bout Tracks (Ch. 3) **81**

Appendix D. Seed Dispersal Model (Ch. 4) Source Code **83**
D.1 model.cpp 83
D.2 io.hpp 85
D.3 defs.hpp 89
D.4 lattice.hpp 98
D.5 random.hpp 102

Bibliography **105**

List of Tables

2.1 State Variables 11
2.2 Parameter Descriptions and Values 15

4.1 Parameter descriptions and default values 50
4.2 Partial Spearman Correlation Results 59

List of Figures

2.1 Examples of Gaussian Wavelet-Generated Landscapes 10
2.2 Example Output from One Simulated Year 12
2.3 Effect of Peak Mismatch on Pollinators and Pollination for Different Landscapes 17
2.4 Flowering Variance vs. Optimal Peak Mismatch 19
2.5 Maximum Pollinator and Pollination Densities for Different Landscapes 20

3.1 Elevation Map of Tiputini Biodiversity Station 33
3.2 Examples of Circuitscape Output on a Ridge 36
3.3 Log-Likelihood of Alternative and Null Models 39

4.1 Concept Map 46
4.2 Focal Pattern Extent at End of Simulation 56
4.3 Example Output of Single Simulation Run 57
4.4 Timing vs. Intensity of Peak Reticulation 58

Chapter 1

Introduction

Movement is a fundamental process across the animal kingdom. While some animals move passively—along with ocean currents or as parasites on other animals, for example—many undertake active locomotion. This kind of movement requires a broad complex of capabilities, including the capacity to collect to information about the environment, make decisions about where to go, and physically execute those decisions. As such, the study of animal movement has traditionally been spread across different disciplines and sub-fields including landscape ecology, foraging theory, animal behavior and cognitive science. Recently, the emerging field of Movement Ecology has sought to combine this diffuse topic into a single framework, where the movement path of an animal is the outcome of the interaction of four fundamental processes: external factors, or the sensing of environmental variables; the animal's internal state, including motivation and behavior; the animal's navigation capacity, or capacity for spatial cognition; and the animal's motion capacity, or its physical ability to move.

Along with the need for mates and evading predation, acquiring food is among the most important internal motivations for movement. The resources in an animal's habitat are, however, often distributed heterogeneously in both space and time. To survive long-term, an animal must consume more energy than it expends in traveling between resources. Foraging theory assumes that a species' movement capacity, including its sensory, cognitive, and physical capabilities, has evolved to maximize the

ratio of energy intake to expenditure for a particular environment. The best foraging strategies take advantage of efficiencies afforded by distinct spatio-temporal patterns in resource distribution. As a result, the patterns of foraging movement often reflect the particular resource patterns on a landscape.

At the same time, foraging animals can also have a significant impact on the landscapes over which they move. As animals move through a site, they may physically change it, such as by degrading vegetation or eroding soil, creating landscape-level patterns mirroring the scale and pattern of their movements. By depositing seeds or pollen collected from distant locations, they can also alter the community composition or gene pool at a site, creating landscape patterns in genetic structure or species distribution.

This two-way interaction between movement and landscape sets up a feedback loop: the movement of dispersers alters the landscape upon which they likely base their subsequent movement decisions. Thus far, the formation of movement and landscape patterns have not commonly been studied as an integrated cycle. This may be because of the complexity of such a framework, or because in some systems, the strength or scale of the two processes is mismatched, effectively reducing the loop to a one-way interaction for practical purposes.

In this dissertation, I examine both the causes and consequences of foraging movements in different systems as elements of this larger, cyclical interaction. In chapter 2, I use simulation modeling to look at how crop-pollinating solitary bees interact with the spatial and temporal availability of crops on agricultural landscapes, and how this interaction affects pollination services. Here, movement patterns are static in the sense that simulated individuals do not respond to changes in the landscape. Spatial patterns of habitat conversion to cropland do, however, affect nest establish-

ment and therefore alter the projection of pollination services, meaning landscape patterns ultimately feedback on themselves through the movement process.

In chapters 3 and 4, I investigate two possible explanations for the use of habitual pathways by frugivorous, seed-dispersing spider monkeys in neotropical forests. First, in chapter 3, I examine whether their movements might be driven by static landscape patterns, specifically topography, by comparing the power of two different movement models—a pure random walk and a model based on elevation—to explain observed data. Second, in chapter 4, I use agent-based simulation modeling to test whether spider monkeys can, through the deposition of seeds, alter forest structure to reflect, and possibly reinforce, their travel routes. This final chapter incorporates the entire interaction loop between movement and landscape patterns that are both dynamic and responsive to one another through time.

Chapter 2

Effect of Monoculture Phenology on Native Pollinator Persistence and Pollination Services in Agroecosystems

2.1 Introduction

Natural populations are buffeted by external forces ranging from weather to disease. Primary among these are temporal variations in resource supply rates with concomitant impacts on fecundity and survival. Diverse taxa from a wide range of systems have adapted to resource cycles with both annual (Fryxell and Avgar, 2012) and multi-year components (Yang, 2004; Kelly, 1994) ranging from broad, regular patterns to short, unpredictable pulses (Yang et al., 2008; Holt, 2008; Ostfeld and Keesing, 2000).

Pollinator species are especially subject to fluctuations in resource supply owing to the ephemeral nature of flowering events. Individual plants generally provide nectar and pollen resources for a period of days or weeks, triggered by seasonal environmental cues such as photoperiod or temperature (Rathcke and Lacey, 1985). At the population level, plant phenology is an aggregation of the phenology of many individual plants, and so the duration of a flowering event is a function of not only the average flowering duration of a single plant, but also the variance in starting dates (Elzinga et al., 2007; Rathcke and Lacey, 1985). At the landscape level, the temporal distribution of flowering resources is the sum of an entire community of overlapping phenologies, with different species peaking and receding at different times (Ogilvie

and Forrest, 2017). In temperate climates, flowering of the community as a whole is restricted to the growing season, generally spring and summer (Rathcke and Lacey, 1985).

Pollinators typically time the most energy-intensive periods of their life cycle to correspond with these phenological cycles of pollen or nectar production. In particular, specialist pollinators, such as many solitary bees, tend to have relatively short flight seasons, when compared with those of generalist pollinators, which coincide with the peak flowering time of the narrow collection of related plant species for which they forage (Ogilvie and Forrest, 2017). This delicate synchrony between plants and pollinators may, however, be disrupted by idiosyncratic local variation, such as a late frost (Inouye, 2008), or from longitudinal trends in phenology caused, for example, by climate change (Memmot et al., 2007; Hegland et al., 2009).

It is particularly important to understand these phenological interactions in agroecosystems, where pollinators provide vital ecosystem services (Klein et al., 2007) and management practices have direct, but complex, effects on the timing of resources available to pollinators. The conversion of wild habitat to farmland generally replaces the native plant community with a less diverse assemblage of crops, or even a monoculture, such that the phenological profile of these crops begins to dominate the landscape. As a result, specialist pollinators whose life cycles do not coincide with the flowering time of these now dominant species may suffer. The duration of the resource availability can also play a role. Some crop species, known as mass-flowering crops (MFCs), bloom all at once as a population, such that variance in individual flowering times is low and resources are available in a single, short pulse (Jauker et al., 2012; Holzschuh et al., 2013). Therefore, when land managers replace wild habitat, or even other crops, with an MFC, the landscape's total annual resources available

to pollinators become tightly clustered in time.

At the same time, agricultural management decisions can have profound spatial consequences for pollinators. As agricultural intensification results in the loss of nesting habitat for native pollinators, their populations suffer, particularly cavity-nesting bees which cannot nest in disturbed land as ground nesting bees can (Williams et al., 2010). Beyond a certain point, adding tillable land area no longer increases crop yields because depressed wild pollinator populations cannot provide robust pollination services (Deguines et al., 2014). The particular spatial pattern of this habitat conversion, however, is also important. Compared to landscapes with separate, contiguous reserves of natural habitat, crops benefit more from fragmentation in which land-use types are interspersed and crops are within easy foraging distance of the pollinators' nesting grounds (Deguines et al., 2014; Kremen et al., 2004; Ricketts et al., 2008). The effects of fragmentation on pollinator health are less uniform across species, depending in part on whether the pollinators are able to utilize crop resources in addition to native ones (Ricketts et al., 2008).

Given these spatiotemporal dyamics, *how does resource pulsing affect plant-pollinator interactions in agroecosystems?* Empirical studies suggest that the presence of oilseed rape (*Brassica napus*), a common MFC, boosts reproduction and survival of foraging bees during the pulse, but decreases them at other times of the year when resources are more scarce (Jauker et al., 2012; Riedinger et al., 2014; Rundlöf et al., 2014; Westphal et al., 2009). For solitary bees, this appears to be a net positive for population growth (Holzschuh et al., 2013; Jauker et al., 2012), with the benefits outweighing the disadvantages, but these studies do not control for total available resources. Oilseed rape is a highly productive pollen source for bees compared to the native habitat or other crops it might replace, making it difficult to isolate the purely

temporal effect of phenological concentration. Simulation modeling would be an ideal tool for isolating these temporal effects, allowing us to redistribute resources in time while keeping total resources constant. Previous modeling studies on pollinators in agroecosystems have investigated spatial effects (Brosi et al., 2008; Keitt, 2009), but few have considered the additional key role of pulsed resource phenology on pollinator success and pollination services.

The purpose of this study is to investigate how the phenological concentration of pulsed flowering resources, combined with known spatial dynamics, affects pollinator abundance and pollination services in an agricultural system. Here we extend a previous landscape model of pollination services (Keitt, 2009) to include pulsed plant production and a more realistic treatment of pollinator life-history events with finer temporal resolution and a focus on solitary bees. We use this model to evaluate different scenarios of plant-pollinator mismatch and mass-flowering phenology under different spatial patterns of habitat conversion.

2.2 Methods

In constructing our model, we chose the simplest possible structure that still captures the essential processes under investigation: the spatial patterns of habitat conversion and the temporal patterns of solitary bee life-cycles and crop phenology. Our goal is not a model that could be used to manage any particular pollinator-agroecosystem. Rather we seek to uncover general principles of phenological mismatch in a landscape with realistic spatial structure and reasonable, albeit necessarily simplified, plant-pollinator biology.

2.2.1 Overview

Our model is a spatially explicit, individual-based simulation that proceeds over discrete daily time steps on a spatially discrete cellular landscape. We have taken the spatial processes from the Keitt (2009) model and added both intra-annual resource variation and more detailed pollinator demographics to specifically capture the life histories of solitary bees foraging on mixed agricultural landscapes.

The simulation landscape consists of wild pollinator habitat, some of which has been converted to agricultural use. The spatial pattern of this habitat conversion is characterized both by the total proportion of cropland on the landscape and the size of the individual fields or patches. Each cell, regardless of habitat type, contains a single plant. These plants are modeled as annuals, and we assume no seed limitation such that they are all replaced by a new plant at the beginning of each year (see Keitt (2009) for a model with direct seedbank dynamics).

Individual solitary bees nest in the wild habitat but forage among both wild and crop plants within the vicinity of their nests. When visiting a plant, foragers trade pollination services for nectar and pollen resources, but the availability of plant resources is limited by the timing of flowering and competition with other pollinators. Each plant is only in bloom during a single flowering period each year assigned to each plant at random. During this time, the first pollinator to visit will pollinate the plant and receive the resources. After that, the flower is exhausted and pollinator visits no longer have any effect. The flowering periods are scheduled such that, on the whole, wild plants, which represent a diverse community, are available uniformly throughout the year, while crops, which represent a mono-culture, are available in an annual mass-flowering pulse.

Successful foragers use their collected resources to provision their brood at the

nest. Foragers do not survive year to year, but the summer's brood persists through the winter to subsequently emerge as new foragers. Emergence occurs as a pulse the following year. By varying the relative timing of the flowering and emergence pulses and their concentrations in time, we can simulate a wide-range of congruence, intensity and mismatch between pollinator activity and pollen resources. We test the effect of these different temporal patterns on pollinator abundance and crop pollination services, as well as the effect of two spatial parameters, the degree of habitat conversion from wild to cultivated states and the characteristic patch size of wild and cultivated parts of the landscape.

2.2.2 Spatio-Temporal Structure

Our simulation proceeds over an annual cycle with $T = 180$ daily time steps per year. The simulation landscape consists of a grid of 256 x 256 discrete cells. Cells are arranged on a torus in order to emulate a larger landscape while maintaining computational feasibility. Each cell is designated as either cropland (at proportion $1 - \Lambda$) or wild habitat (at proportion Λ) for the duration of the simulation, and contains a single perennial plant. The spatial pattern of cropland and wild habitat was randomly generated using a Gaussian wavelet model with a Haar kernel (see Keitt (2000); Lasky and Keitt (2013) for details):

$$Var(\phi_i, \sigma) \propto e^{-2\sigma^2\omega_i^2} \tag{2.1}$$

where ϕ_i is the set of wavelet coefficients and ω_i is the central frequency of the wavelet kernel at transform level i, and σ determines the characteristic patch size in cells (Figure 1). For the Haar wavelet, $\omega_i = \pi/2^i$ radians where $i = 0$ is the finest scale (highest frequency) of analysis.

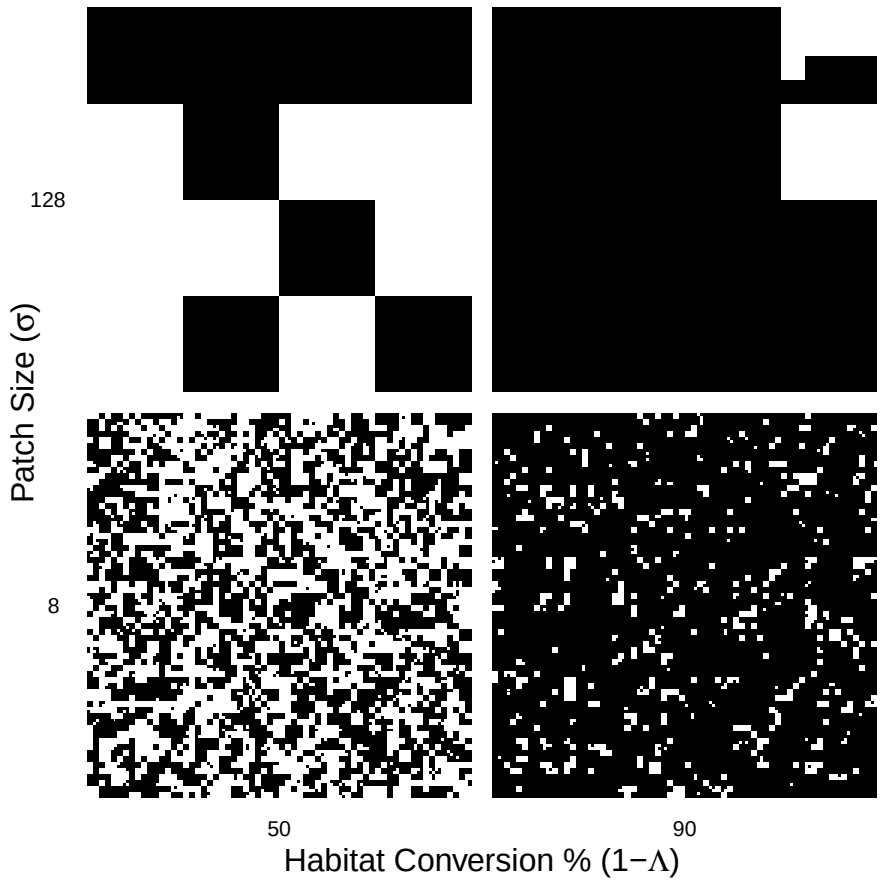

Figure 2.1: Examples of Gaussian Wavelet-Generated Landscapes

Landscapes are randomly generated from a Gaussian wavelet transform at different patch sizes and habitat conversion rates. White area represents habitat while black area represents cropland.

2.2.3 State Variables

Four dynamic state variables are associated with each cell i and updated daily (Table 2.1). The total number of foragers nesting in a cell is given by n_i, where each forager is an independent solitary bee. The number of pupating brood in a cell is given by b_i, although each forager contributes to this pool by mass provisioning and sealing its own brood cells without coordination with other foragers. Foragers may only nest in habitat cells, and therefore n_i and b_i are always zero for all crop cells. There is no limit on the number of foragers or brood in a cell at any given time.

On a given day, the state of a cell's plant is described by two boolean characteristics: flowering status, given by f_i, and pollination status, given by p_i. A plant's flowering status is only true during its plant-specific annual flowering period, the set of consecutive days in which the plant is potentially receptive to pollinator visits. Pollination status simply describes whether or not a plant has been pollinated yet in a given year. A pollinated plant can yield no resources to pollinators for the remainder of the year.

n_i	number of foragers nesting within cell i
b_i	number of brood developing in cell i
f_i	boolean flowering status of the plant in cell i
p_i	boolean pollination status of the plant in cell i

Table 2.1: State Variables

2.2.4 Processes

Each simulation day, the following processes take place, in order:

1. *Phenology* Flowering status f is updated for plants entering or exiting their flowering period. For a plant in cell i, this period lasts for π days beginning on

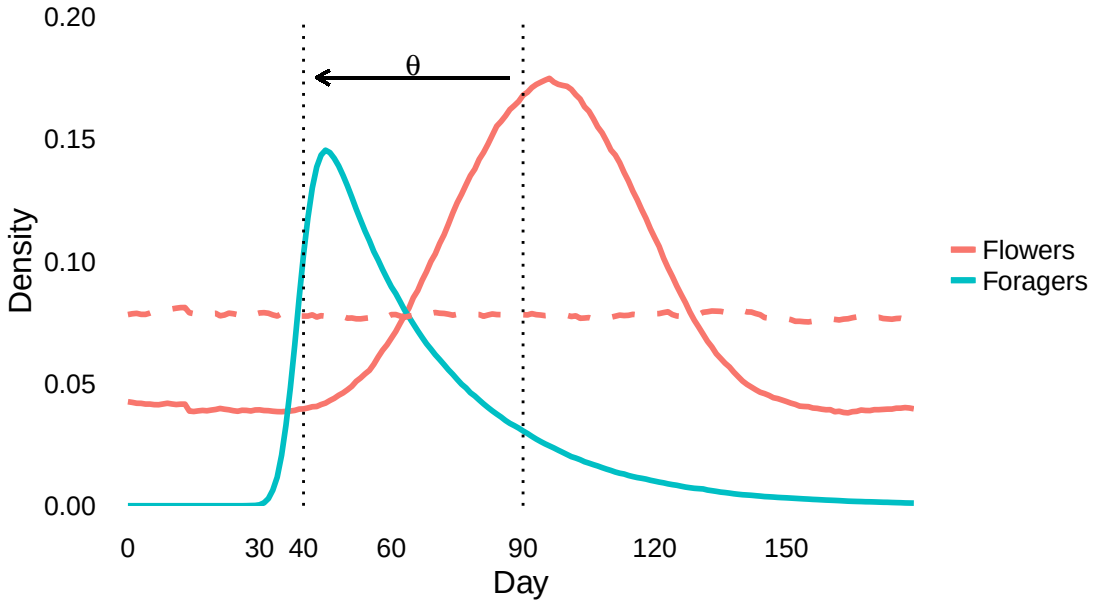

Figure 2.2: Example Output from One Simulated Year

Daily time series over a single simulation year: The solid lines show daily nest density and the density of plant (both wild and crop) that are flowering on any given day ($\Lambda = 50\%, \sigma = 128, \alpha = 10$). The horizontal dashed line shows the flowering plant density under uniform phenology ($\Lambda = 50\%, \sigma = 128, \alpha = 1$). The vertical lines, from left to right, show the peak-emergence date ($t'' = 40$) and peak-flowering date ($t' = 90$), where the difference between them is the peak mismatch, θ. In this case, Peak Emergence occurs before Peak Flowering, so θ is negative.

day t_i where

$$t_i/T \sim Beta(\alpha, \beta) \tag{2.2}$$

such that t_i ranges from day 1 to T, and the peak flowering day across the landscape is given approximately by

$$t' \approx T \cdot \frac{\alpha - 1}{\alpha - \beta - 2} \tag{2.3}$$

For all plants, this distribution is symmetrical ($\alpha = \beta$). For non-crop plants, $\alpha = \beta = 1$ and the distribution of flowering dates is uniform across the year, on average. For crops, both α and β are greater than one, with the variance of flowering times decreasing as α or β increase.

2. *Foraging and Reproduction* Each cell, in random order, emits n_i foraging trips (one per forager) in random directions, where the distance traveled is drawn from a negative exponential distribution with mean $1/\vartheta$ (hereafter, simply the 'dispersal distribution'). If the cell in which the forager lands has an unpollinated ($p_i = False$), flowering plant ($f_i = True$), then the forager pollinates this plant (sets $p_i = True$), returns to its nest cell, and adds a single brood to b_i.

3. *Mortality* All pollinators are subjected to a constant daily rate of mortality, δ, producing a negative exponential survival curve where the average lifespan is $1/\delta$ days.

4. *Emergence* Brood provisioned the previous year emerge individually on dates drawn from a distribution identical in form to the phenology distribution (Equation 2.2), but with shape parameters μ and η and peak-emergence date t''. The emergence date of an individual does not depend on the date on which that individual was provisioned in the previous year. After emerging, the new foragers

immediately disperse from their natal cells according to the dispersal distribution. If they land in habitat, they are added to the forager population in that cell (n_i) and remain nesting there until their death. If they land in crops, they immediately die.

At the end of each year, all foragers and plants die ($n_i = 0$ for all i), while the brood persist unaltered. Each previous plant is replaced by a new, un-pollinated plant that survives throughout the next season and has a new flowering date drawn from the distribution described previously.

2.2.5 Summary of Simulated Scenarios

We varied α, t'', σ and Λ to test the effect of phenology, emergence date, patch size and habitat conversion on both nest density and crop pollination services (see Table 2.2 for value ranges). We chose reasonable constant values for the remaining parameters to reduce the parameter space to be explored (Table 2.2). The daily nest mortality rate (δ) corresponds to an average lifespan of 28 days, while the dispersal distribution parameter ϑ corresponds to an average dispersal distance of 16 cells.

Because the mismatch between the peak-emergence date and peak-flowering date was likely to be more important than the dates themselves, we held the peak flowering date constant at midyear ($t' = day\ 90$) and varied the peak emergence date (t'') with respect to t' rather than varying both. We also decided to set $\mu = 100$ for all brood, such that the variance of emergence dates was low. This implies that all our pollinators are specialists that have evolved to emerge during a short window that presumably corresponds to the flowering period of a specific host-plant species. We did not independently set η, but rather calculated it from μ and the desired peak date, t" (see Equation 2.3). Crops always emerge at midyear ($t' = 90$), so $\alpha = \beta$

ensures a centered, symmetrical distribution.

Parameter	Description	Value
T	Length of Growing Season	180 day
π	Flowering Period	14 days
δ	Daily Forager Mortality Rate	0.036
ϑ	Dispersal Parameter	0.0625
σ	Characteristic Patch Size	$2 \rightarrow 128$ cells
Λ	Habitat Coverage Rate	$0.05 \rightarrow 1$
$1 - \Lambda$	Habitat Conversion Rate / Crop Density	$0.95 \rightarrow 0$
α	Flowering Concentration Parameter	$1 \rightarrow 1000$
t'	Peak Flowering Date	90
μ	Emergence Distribution Shape Parameter	100
η	Emergence Distribution Shape Parameter	$\frac{T \cdot (\mu - 1)}{t"} - \mu + 2$
t''	Peak Emergence Date	$1 \rightarrow 180$

Table 2.2: Parameter Descriptions and Values

2.2.6 Initial Conditions

We ran each simulation on a different randomly generated landscape for 100 years. At the beginning of the simulation, all plants are unpollinated ($p_i = False$) and not flowering ($f_i = False$), and the landscape contains no foragers ($n_i = 0$). Brood are distributed randomly among the habitat cells such that the brood density across the entire landscape is equivalent to 0.25 brood per cell. We randomly assign each brood an emergence date in the current year, drawn from the emergence distribution, as if a forager population had provisioned them in the previous year, before the simulation began.

2.2.7 Output and Analysis

For each simulated day, we recorded n_{tot}, the total forager density, or the average of n_i over all cells. For each year, we calculated N, the maximum annual

value of n_{tot}. We chose the maximum forager density, N, as our annual measure of population over time because it is always non-zero for extant populations and reflects the actual population density one might observe during a pollinator's period of peak activity.

At the end of each year, before pollination status is reset, we also recorded the density of pollinated crops (crop cells where $p_i = True$) as P. This reflects the sum of all crops pollinated throughout the year and therefore the annual pollination services directly provided by the pollinator, allowing us to evaluate spatio-temporal patterns from a yield management perspective.

We averaged these annual values, N and P, for the final 25 years of a simulation to obtain its equilibrium values $\bar{N}$ and $\bar{P}$, respectively. Additionally, for each parameter set, the $\bar{N}$ and $\bar{P}$ values reported represent the average of 10 simulations. Simulations generally reached steady state well before year 75. We chose to analyze equilibrium values because they are less sensitive to initial conditions and simpler to analyze than a time series of transient states.

We define peak mismatch as the difference, in days, between the peak crop-flowering date and the peak brood-emergence date:

$$\theta = t'' - t' \tag{2.4}$$

The values of θ which optimize $\bar{N}$ and $\bar{P}$ ($\bar{N} = \bar{N}_{max}$ and $\bar{P} = \bar{P}_{max}$) for a given combination of σ, Λ and α values, are given as θ_N and θ_P, respectively (see Figure 3).

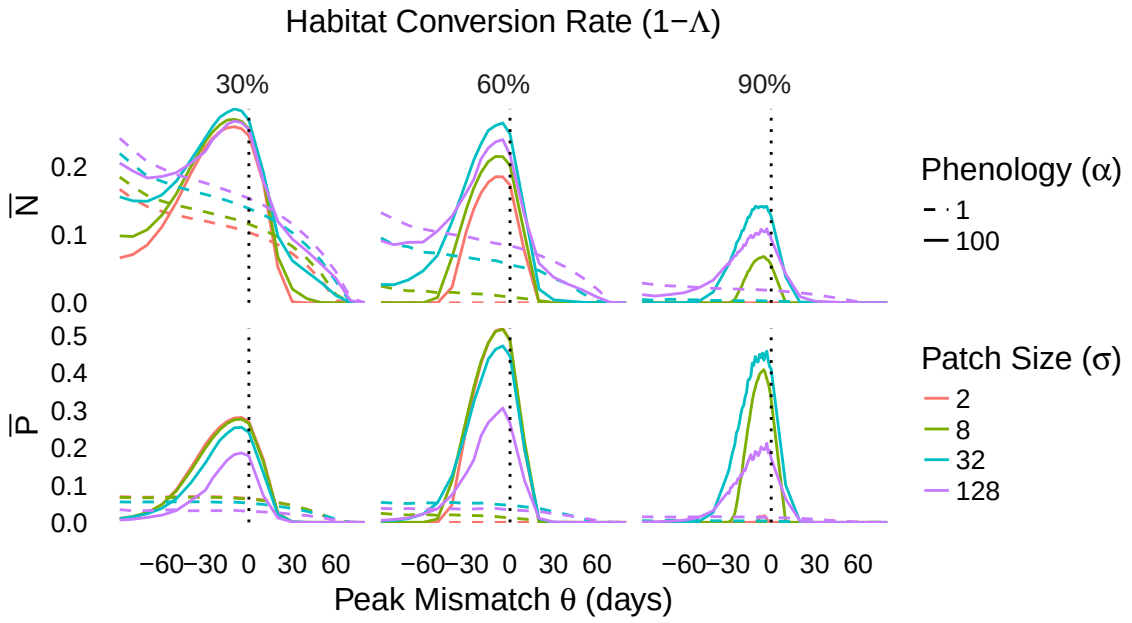

Figure 2.3: Effect of Peak Mismatch on Pollinators and Pollination for Different Landscapes

Pollination density ($\bar{N}$) and crop-pollination density ($\bar{P}$) under mass-flowering phenology (solid) and uniform phenology (dashed), at different values of peak mismatch (θ), habitat conversion ($1 - \Lambda$) and patch size (σ): The vertical dotted lines show $\theta = 0$, where brood and flowers both peak on day $t = 90$. Points to the left of this line represent simulations in which brood emerge before flowering, and to the right of this line, after.

2.3 Results

Our results highlight a series of complex and interrelated effects of both spatial and temporal variation. For a given landscape, the peak mismatch, θ, strongly affects both pollinator density ($\bar{N}$) and crop-pollination density ($\bar{P}$). Under uniform phenology ($\alpha = 1$), $\bar{N}$ is maximized when brood emergence peaks on the first day of the year ($\theta = -90$) and steadily declines to extinction as peak mismatch (θ) increases (Figure 2.3, dashed lines). $\bar{P}$ remains nearly constant for negative values of θ and then also declines to 0 as θ becomes positive. For the same landscapes under high-intensity mass-flowering phenology ($\alpha = 100$), however, $\bar{N}$ and $\bar{P}$ are maximized when brood emergence peaks just before flowering does, such that θ is slightly less than zero (Figure 2.3, solid lines). Negative θ values produce higher $\bar{N}$ and are less likely to lead to pollinator extinction than an equal peak mismatch in the positive direction, while $\bar{P}$ is near zero for both extremely low and high values of θ (Figure 2.3).

Under mass-flowering phenology ($\alpha > 1$), the values of peak mismatch (θ) which maximize pollinator density ($\bar{N}$) and crop-pollination density ($\bar{P}$) are not significantly different from one another, such that we do not distinguish between them here, referring to both collectively as θ_{max}. This θ_{max} is not affected by patch size (σ) or habitat conversion rate ($1 - \Lambda$) but it is positively related to the concentration of the flowering distribution (α), approaching 0 as α increases (Figure 2.4). It is important to note that within a single year, the forager population (n_{tot}) and total flower density actually peak several days after the day designated by parameters t' and t'', respectively, although this lag does not appear to significantly affect peak mismatch, θ (Figure 2.2).

The landscape pattern also affects pollinator density ($\bar{N}$), although this re-

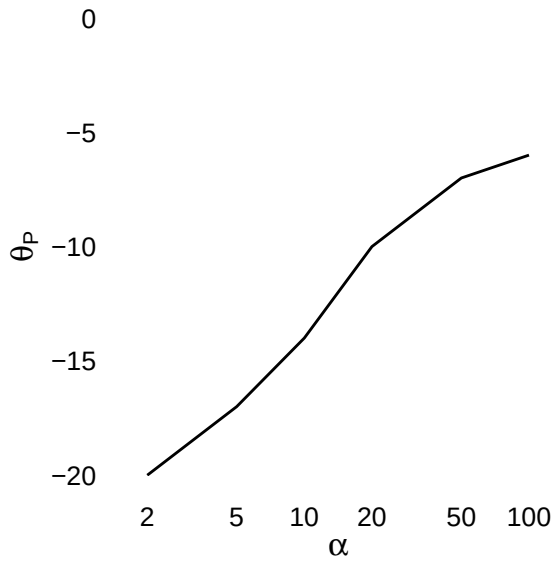

Figure 2.4: Flowering Variance vs. Optimal Peak Mismatch

Optimal peak mismatch (θ_P) for different values of the concentration parameter (α) on a logarithmic scale ($\Lambda = 40\%, \sigma = 16$). As α increases, crop phenology becomes more concentrated in time. Only θ_P is shown here, but θ_N produced similar results.

lationship can depend on the peak mismatch and phenology. For all values of peak mismatch (θ), low patch size ($\sigma = 2$) is more likely to lead to pollinator extinction (Figure 2.3). Under uniform phenology ($\alpha = 1$), $\bar{N}$ increases uniformly as patch size (σ) increases from 2 to 128. Under mass-flowering phenology, $\bar{N}$ does the same, but only when brood emerge early in the year ($\theta << 0$) or late in the year ($\theta > 0$; Figure 2.3). When brood emerge at θ_{max}, $\bar{N} = \bar{N}_{max}$ only increases up to $\sigma = 32$ and then declines (Figures 2.3 and 2.5). The effect of habitat conversion is more consistent across values of θ and α. As habitat is converted to crops, or habitat coverage (Λ) goes to 0, $\bar{N}$ decreases uniformly and is more likely to collapse (Figure 2.3).

Like pollinator density ($\bar{N}$), crop-pollination density ($\bar{P}$) is nearly zero when foragers emerge late, but unlike $\bar{N}$, it is also nearly zero when foragers emerge early (Figure 2.3), so the effect of landscape pattern on $\bar{P}$ can only be assessed at θ_{max}, when

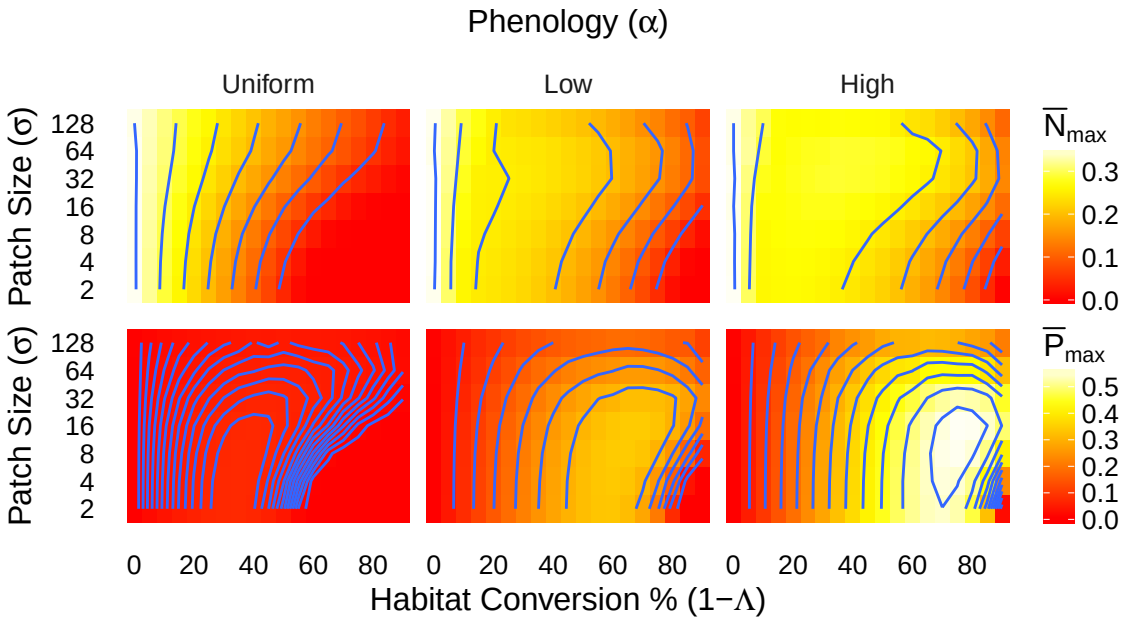

Figure 2.5: Maximum Pollinator and Pollination Densities for Different Landscapes

Contours of $\bar{N}_{max}$ and $\bar{P}_{max}$ for different combinations of habitat conversion $(1 - \Lambda)$ and patch size (σ). Panels show different levels of mass-flowering phenology: uniform $(\alpha = 1)$, low $(\alpha = 10)$ and high $(\alpha = 100)$.

$\bar{P} = \bar{P}_{max}$. Under uniform phenology, $\bar{P}_{max}$ is universally low, although it is slightly greater when habitat conversion is roughly 35% and patch size is small ($\sigma \leq 16$) (Figure 2.5). As the flowering pulse becomes more concentrated (α increases to 100), the patch size (σ) and habitat conversion ($1 - \Lambda$) that produces the greatest $\bar{P}_{max}$ increase to 16 and about 75%, respectively. In addition, as α increases, the value of $\bar{P}_{max}$ resulting from this ideal landscape increases in both absolute terms and in prominence with respect to other landscapes (Figure 2.5, bottom row).

If we look at $\bar{N}_{max}$ from the same perspective (Figure 2.5, top row), we see that as α increases, the ideal landscape for forager populations at θ_{max} decreases in patch size from 128 to 32. Although $\Lambda = 0$ always results in high forager density, regardless of phenology, the ideal landscape also includes higher rates of habitat conversion as α increases. The highest achievable $\bar{N}_{max}$ is always roughly 0.3, however, and does not change with α.

2.4 Discussion

The spatial dynamics of our model are similar to those in Keitt (2009). Pollinators generally thrive when habitat conversion is low because the total area available for nesting is high. Pollinators also benefit from large patch sizes because when newly emerged foragers disperse locally from natal cells in the middle of uninterrupted habitat, they are more likely to land in habitat, resulting in a higher survival rate.

Pollination services are, however, maximized when both habitat conversion rate and patch size are intermediate. This is because high crop-pollination density requires both high pollinator density and adequate access to receptive crops, and these factors favor opposite landscape patterns, respectively. A predominantly wild landscape yields high pollinator density, but there are few crops, so most of the

pollination activity goes to wild plants instead. Conversely, at high habitat conversion rates, there are plenty of available crops, but few foragers to pollinate them. Similarly, large patches are good for pollinators, but crops in the center of such fields are out of range of foragers nesting on the edge. Alternatively, small patches make these crops more accessible, but at the cost of higher dispersal mortality. A compromise between the extremes of both spatial parameters yields the maximum crop pollination density, specifically when patch size is equal to the average foraging distance.

These spatial effects dominate the uniform model because there is no temporal pattern of resource availability, with crops and wild plants both flowering randomly throughout the year. Furthermore, the temporal dynamics are one-dimensional: the earlier the brood emerges, the better pollinators perform. This is because of the negative exponential shape of the forager survival curve, by which some foragers from the emergence pulse can survive the entire year. This gives the intra-annual pollinator density curve (n_{tot}, Figure 2.2) a long right-sided tail. The area under this n_{tot} curve is equivalent to the total number of annual foraging trips, in units $forager \cdot days$. A later emergence pulse means winter cuts off more of this tail area, reducing the total annual foraging trips, leaving less total $forager \cdot days$ to pollinate crops or provision brood.

The mass-flowering model retains the spatial and temporal dynamics of the uniform model, but by concentrating crop resources in time, this model subjects crop access to a temporal filter in addition to the spatial one. The overlap between the curves for n_{tot} and total available flowers (Figure 2.2) represents the exposure of foraging trips to available resources. When the emergence and flowering pulses coincide, this overlap area is large, as the majority of foraging trips are successfully matched with available crops, but when they are asynchronous, many of the foraging

trips are wasted on unreceptive crops. As the width of the resource pulse narrows, foragers must emerge even closer to the flowering peak to take full advantage. The temporal and spatial dynamics also interact with one another, such that the habitat conversion rate regulates the impact of a given phenology pattern. When conversion is low, even a highly concentrated phenology pattern can only affect crops, which are few, and total resources remain uniform throughout the year, on average. As more habitat is converted to cropland, a larger proportion of total resources is subject to the annual pulse. Random flowering dates are traded for concentrated flowering dates, and the difference in resource levels between peak and off-peak times becomes more extreme.

Conversely, concentrated phenology patterns can mitigate the negative effects of habitat conversion on pollinators. From a purely spatial perspective, adding cropland up to a certain threshold increases the total number of pollinated crops, but beyond this, pollinators suffer too much from habitat loss to pollinate effectively, and crop yields decline. When crops and pollinators are perfectly matched in time, mass-flowering phenology can increase this threshold compared to uniform phenology, which allows more habitat conversion before pollination falls off and increases the maximum attainable yield of the system. This effect, however, cuts both ways: if crops and pollinators are mismatched, crop yields are far worse than under uniform phenology. In this case, the displacement of resources in time means pollinators decline from lack of resources and the few pollinators that remain cannot access crops at the proper time to pollinate them.

These results suggest that planting MFCs can benefit agricultural systems, but managers must carefully consider the phenological match between crop phenology and the life cycles of those domestic or wild pollinators expected to provide the

majority of pollination services. Even when phenologies are well-matched, however, over-reliance on MFCs leaves pollinators more susceptible to disturbance or inter-annual variation. Phenological shifts due to climate change or loss of crops to disease could be catastrophic for pollinators, with little resource redundancy to buffer against disturbance. These management decisions are even more consequential on intensively farmed landscapes, as temporal effects are magnified by larger cropland area.

We do not propose, however, that these results represent comprehensive or universally applicable principles for managing the temporal effects of agricultural intensification. Our model does reveal basic temporal and spatial dynamics of a single plant-pollinator interaction in the absence of confounding processes, but real agroecosystems are, of course, far more complex. Pollinators with life history traits different from the specialist solitary bees modeled here could behave differently, and we hope to incorporate this diversity into future modeling studies. For instance, ground-nesting bees, which can nest more effectively in disturbed land, may be able to withstand higher rates of habitat conversion (Williams et al., 2010). Generalist pollinators, which tend to live longer than specialists, would probably not be as sensitive to shifts in crop phenology. Even a specialist, if able to adapt its emergence time in response to flowering changes, could better accommodate large phenological changes, although we have little information on the plasticity of pollinator emergence in the wild.

Another important extension to the model would be to include social bees (i.e., honey bees and bumblebees), which we expect would perform worse than solitary bees. Studies suggest that mass-flowering phenology is, on balance, detrimental to social bees because their colonies require resources more consistently throughout the year than solitary bees (Williams et al., 2012; Westphal et al., 2009). MFCs that flower

early in the season boost colony size at the expense of late-season resources needed for reproduction (Rundlöf et al., 2014). There exist excellent models of complex colony demographics responding to variable resource-intake rates (Khoury et al., 2011, 2013; Russell et al., 2013), and while properly parameterizing them is difficult, combining them with our landscape model would be a promising next step.

Chapter 3

The Effect of Topography on Path Choice of a Frugivorous Neotropical Primate, *Ateles belzebuth*

3.1 Introduction

According to the Movement Ecology framework, an individual animal's movement results from the response of its internal behavioral state to its external environment, mediated by the animal's locomotive and navigational capabilities (Nathan et al., 2008). Foraging animals, whose internal motivation for movement is the acquisition of food, must search their environment for resources in space and time and the search strategy they employ is constrained by their capacity for locomotion and navigation, including sensory input and spatial cognition. Optimal Foraging Theory assumes that this strategy and the traits that constrain it have evolved to optimize the ratio of resource intake to energy expenditure for the given distribution of resources in its habitat (Charnov, 1976; Pyke et al., 1977).

The standard null model in studies of foraging movement is the *random walk* (Codling et al., 2008). In its simplest form, a naive random walk assumes no spatial memory or even sensory capacity on the part of the forager. In wholly unpredictable environments, random search can be effective, but when memory or sensory input gives a forager information about the distribution of resources, more directed search methods are favored (Bartumeus et al., 2005). The movement patterns of foragers reflect the interaction between search strategy and resource patterns, and the use of non-random strategies on structured landscapes often produces structured move-

ment patterns. These include patterns in space, such as clustered or highly directed movements (Morales et al., 2004), but also temporal patterns, such as repetitive or cyclical movements (Thomson et al., 1997). The foraging movements of spider monkeys (genus *Ateles*) in particular, contain pronounced non-random structure: spider monkeys show high fidelity to certain routes, repeatedly using the same paths over time to move between fruiting resources (Di Fiore and Suarez, 2007). What drives the formation of these patterns? In other words, how do monkeys choose their foraging paths, and why do they use the same routes again and again?

As tropical frugivores, spider monkeys are confronted with a particularly complex spatio-temporal resource distribution when searching the forest for ripe fruit. As generalists, their diet can include more than 150 different tree species—which can fruit at different times of the year, yield different nutritional values and ripen at different rates (Link and Di Fiore, 2006). Fruiting times can also be extremely unpredictable in the neotropical forests in which spider monkeys are found, particularly in equatorial forests that lack distinct seasonality (Schaik, 1993). Additionally, resources are simultaneously being depleted by a complex community of competing frugivores, including birds, bats and other primates, that can denude an entire tree without warning (Chapman, 2013; Riba-Hernández et al., 2003; Suarez et al., 2014). Finally, the fruiting trees that support a group of spider monkeys can be distributed patchily across hundreds of hectares of dense forest with low visibility that prevents monitoring or recognizing resources from afar. This makes their resources both unpredictable in time and difficult to find in space, yet frugivorous primates search this kind of landscape more efficiently than a random walk would, suggesting they possess a capacity for spatial memory that allows them to process the available information and formulate an efficient search plan (Garber and Hannon, 1993; Schreier and Grove,

2010; Porter and Garber, 2013; Hopkins, 2016).

The exact structure of spatial memory in primates and other taxa is still the subject of intense debate and research (Fagan et al., 2013). Ideally, a foraging primate would possess perfect information about the landscape and a mental map of the area known as a vector map, or Euclidean map (Fagan et al., 2013). In this model of spatial cognition, the forager encodes its current location and known resources as coordinates on a Cartesian plane and is able to chart novel direct routes between two points, a process similar to using a map and compass (Tolman, 1948; Rodrigo, 2002). Simulations of spider monkeys foraging show that search strategies based on this kind of mental map can produce route fidelity as an emergent property of repeatedly calculating the same optimal route between valuable resources (Boyer and Walsh, 2010). There is also some empirical evidence suggesting that primates use vector maps to travel in efficient straight-line paths (Normand and Boesch, 2009; Presotto and Izar, 2010; Asensio et al., 2011), but not spider monkeys, whose paths appear more circuitous (Di Fiore and Suarez, 2007). The use of vector maps is also difficult to prove conclusively, because the complexity or efficiency of observed movement patterns can often be explained equally well by simpler cognitive models (Bennett, 1996).

One such alternative cognitive model is a network map, also known as a topological map, where locations are remembered with respect to a set of discrete, intersecting routes, analogous to a street map (Poucet, 1993). These routes and their crossing points are recognized by discrete landmarks in the environment, which are often, but not always, visual cues (Rodrigo, 2002). Compared with a vector map, this kind of mental map requires storing fewer spatial data points to cover the same territory, thus reducing learning time as well as the metabolic costs and sleep time

associated with memory formation (Fagan et al., 2013). Foraging with a network map does not, however, allow foragers to forge novel straight-line paths to a resource. Instead, foragers must reuse existing paths or retrace their steps to reach their destination, which would naturally produce route fidelity.

Because of these characteristics, studies have suggested network maps as a likely explanation for the path fidelity observed in *Ateles* and other primates, but it remains unclear what determines the exact location of the pathways (Di Fiore and Suarez, 2007; Noser and Byrne, 2007; Presotto and Izar, 2010; Porter and Garber, 2013; Suarez et al., 2014). One possibility is that, over generations of exploration and social learning, the paths have developed into corridors that connect the most resources for efficient monitoring and collection (Hopkins, 2011).

Topography may also play an important role in the selection of habitual routes. Di Fiore and Suarez (2007) suggests that spider monkeys may use ridge-lines as travel routes because they provide greater visibility for monitoring resources or predation risk over long distances. This need not mean that monkeys only feed on high-elevation resources. Although there are conceivably different ways to use topography, high-elevation contours might be used as highways for high-speed or long-distance travel, where monkeys can depart from them for short forays into low-lying resource patches or to cross unavoidable valleys to link up with other ridges. We are, however, not aware of any studies that explicitly and quantitatively examine the link between topography and the location of travel routes in primates.

The objective of this study is to test whether, and to what extent, spider monkeys use topography in choosing their routes between fruiting resources. While we know that the decision-making process behind primate movement is exceedingly complex, we do not have the necessary data to parameterize a fully behavioral model

that incorporates the variety of dynamic environmental cues that drive decisions in real time. Instead, we test here the hypothesis that monkeys prefer higher elevation routes when traveling between feeding locations, where this preference is the product of a more complex underlying process driven by unknown variables. To do so, we compare the explanatory power of two different movement models, a pure random walk and a model based on elevation, when confronted with actual data on the foraging movements of spider monkeys in the western Amazon. For the elevation-based movement model, we use a biased random walk where monkeys prefer to move uphill in a elevation gradient. This kind of gradient-based random walk assumes no spatial memory other than a recognition of the terminal destination (Garber and Hannon, 1993; Fagan et al., 2013). Previous studies strongly support the existence of some kind of spatial memory in primates, but we chose a simpler movement model with fewer cognitive assumptions in order to focus on the influence of elevation. The purpose of this study is not to provide direct support for any particular model of spatial cognition, although we hope our results will inform the formulation of more sophisticated models of that kind in future studies.

3.2 Methods

3.2.1 Study Site and Species

The data were collected at the Tiputini Biodiversity Station (TBS) within the Yasuni Biosphere Reserve in eastern Ecuador ($0°37'S, 76°10'W$). The station covers roughly $650ha$ of lowland tropical forest in the western Amazon basin with an elevation range of $190 - 270m$ and receiving 2.74m of rainfall annually, on average (Karubian et al., 2005).

The station is inhabited by ten primate species, including a single species of

spider monkeys, the white-bellied spider monkey (*Ateles belzebuth*). Spider monkeys are highly social, forming territorial groups of 20-30 animals that are characterized by internal fission-fusion dynamics, such that individuals form small subgroups which constantly change in size and membership (Symington, 1990; Link and Di Fiore, 2006). Spider monkeys are also primarily arboreal, only occasionally descending to the ground and risking increased predation to eat mineral-rich clay (Blake et al., 2010). Several spider monkey groups inhabit the station, each with persistent, exclusive territories enforced by the adult males through periodic border patrols. The monkeys spend up to 12 hours per day moving, foraging, resting, and socializing before ending the day at one of several frequently used sleeping trees (Russo and Augspurger, 2004; Karubian et al., 2015).

The data analyzed here consist of focal samples collected between 2009 and 2015. In each focal sample, an observer on the ground followed a focal individual for a period ranging from 5 minutes to 12 hours. Focal animals were chosen opportunistically but with an effort to evenly sample the group over long periods of time. Once an individual was chosen, the focal sample continued until the animal was lost or went to sleep for the night. Feeding bouts were defined as at least 5 continuous minutes of feeding by the focal animal on the fruit of a single tree, or at least 5 $animal \cdot min$ of feeding by a group that includes the focal animal (e.g., 5 animals feeding for 1 minute). Every 20 seconds, a Garmin handheld GPS unit (models 60Cx and 76Cx) recorded the altitude and x and y coordinates of the ground observer using the World Geographic System 1984 datum (WGS84) and Universal Trans-Mercator projection (UTM 18S). An experienced observer can generally stay within $5m$ of a focal animal.

3.2.2 Data Processing

The data analyzed here are drawn from a complex, hierarchical, long-term data set collected in challenging field conditions by different technicians of varying levels of experience. To reduce human error, in both observation and data entry, we discarded the entirety of any focal samples that contained incomplete data, logical disagreements, or any other departures from the data model. We also restricted the data to those collected by a select group of three experienced field technicians known to make reliable observations.

Due to the thick forest cover, the location data come with significant horizontal error. During feeding bouts, when an individual is known to be relatively stationary, the residual errors in both x and y have a standard deviation of about 7m and are normally distributed within 2 standard deviations. The fat tails of the distribution beyond this point probably reflect the prevalence of extreme events in which the GPS fails to find a satellite at all and can report coordinates hundreds of meters from the true location. The Distance Root Mean Squared (DRMS) accuracy is 10.5m, meaning approximately 68% of fixes fall within 10.5m of the true location (Chin, 1987). We removed any fixes that suggested unrealistically fast movement by the observer ($> 10m/s$) after accounting for error in sequential locations.

We also reduced the location frequency from every $20s$ to every $2.5min$ by averaging all locations in $2.5min$, where the window size was selected to balance between reducing error and maintaining resolution. This process reduced the DRMS to $8.2m$. Each resulting location is designated as occurring between two feeding bouts or during one (see Appendix C for details). We define an *inter-bout track* as the set of consecutive locations between the final location of a feeding bout and the first location of the next bout. If any fixes were missing, we discarded the entire inter-

bout track. In a single day, an observer may record multiple focal samples, spanning multiple individuals, where each focal sample may contain multiple inter-bout tracks.

We created a high-resolution elevation map (Fig. 1) using the altitude measurements attached to each of more than four million GPS fixes collected at the station as part of a larger project from 2004-2015. We removed unrealistic values outside the known elevation range (190 to 270m), superimposed a 20m raster grid onto the study area, and calculated the resulting elevation (in meters above sea level) in each cell as the average of all the altitude measurements falling within it.

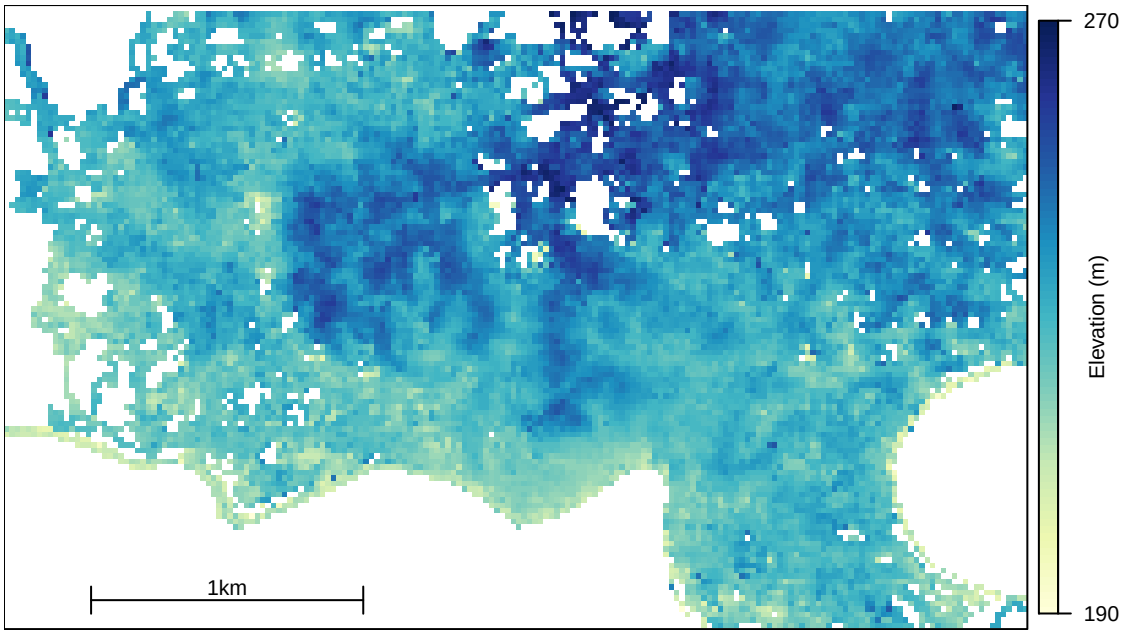

Figure 3.1: Elevation Map of Tiputini Biodiversity Station

The Tiputini River flows along the irregular border at the bottom of this elevation map and then wraps around to the cutout in the bottom right corner.

3.2.3 Analysis

The objective of our analysis is to determine whether an alternative model, in which monkeys prefer high elevation routes, can explain the observed movement

data significantly better than a null model, in which elevation has no effect on path selection. Both the null and alternative models are based on the same gradient-based random walk, where animals wander randomly over a grid of elevation values with a certain level of movement bias towards higher terrain. The null and alternative models differ only in the level of preference for elevation set by the parameter θ.

To calculate these random walks, we used a program called Circuitscape (Shah and McRae, 2008; McRae and Shah, 2013), which models the movement of animals over a landscape as electrons moving through a circuit (see McRae et al. (2008) for a detailed explanation of circuit theory and its application). In this framework, the landscape grid is treated as a lattice network in which each cell is a node connected to its eight neighbors. According to some landscape attribute, each cell, x, is assigned a conductance value, G_x (the reciprocal of resistance, R_x), which is analogous to its permeability, or the ease with which animals move across it. Current is applied to the cell in which the animal begins, while the destination cell is connected to ground, such that all electrons eventually flow from start to finish. As in a real circuit, the electrons flow randomly across the landscape from their origin to their destination, successively choosing neighboring cells with probabilities roughly proportional to their conductance (see Appendix 2 for details). The resulting current, I_x, across cell x is equal to the net number of times any electron crosses that cell during its walk, where each electron represents a possible route that an animal could take from the origin to the destination. Rather than actually simulating all possible random walks, however, Circuitscape solves for the current in each cell analytically, which dramatically reduces computation time. When the total current across the circuit is equal to $1A$, the resulting current across a cell is equal to the probability that the animal's true path passes through that cell en route to its destination.

In our analysis, conductance is related to the elevation as:

$$G_x = \theta^k \ , \ k = \frac{\alpha_x - \alpha_{min}}{\alpha_{max} - \alpha_{min}} \tag{3.1}$$

where G_x is the conductance and α_x is the elevation in cell x, respectively; $\alpha_{min} = 190$ and $\alpha_{max} = 270$ are the minimum and maximum elevations on the landscape, respectively; and θ is the ratio of G_{max} to G_{min} on the resulting conductance map. Scaled exponentially in this way, conductance increases by the same multiplicative factor per meter of elevation at any point in the range.

To compare the likelihood of the different models, we calculated the probability of observing the movement data collected at TBS given the model parameter, θ. Let us assume that for each inter-bout track, there exists a true path expressed as a sequence of grid cells, by which the monkey travels from its origin at the location of the previous feeding bout to its ultimate destination at the location of the next feeding bout. Let $X = x_1, x_2, ..., x_n$ be the observed subset of this true path, where the origin is x_0 and the destination is x_{n+1}. The probability of observing X given the model parameter θ is equal to the joint probability of observing the monkey in each cell x in X:

$$P(X|\theta) = \prod_{m=1}^{n} P(x_m|x_0, ..., x_{m-1}) \approx \prod_{m=1}^{n} P(x_m|x_{m-1}) \tag{3.2}$$

which we can approximate as a simpler expression if we assume the data is a first-order Markov process, where each location depends only on the previous one. To calculate this sequence of probabilities for a given inter-bout track, we ran a separate Circuitscape analysis for each x_m in X, where the destination cell is always x_{n+1}, but the origin cell, x_{m-1}, is always one step behind x_m. In this way, the probability of passing through x_m is only conditioned upon having been observed in x_{m-1} in the immediately previous step, and the resulting current through x_m is equal to $P(x_m|x_{m-1})$.

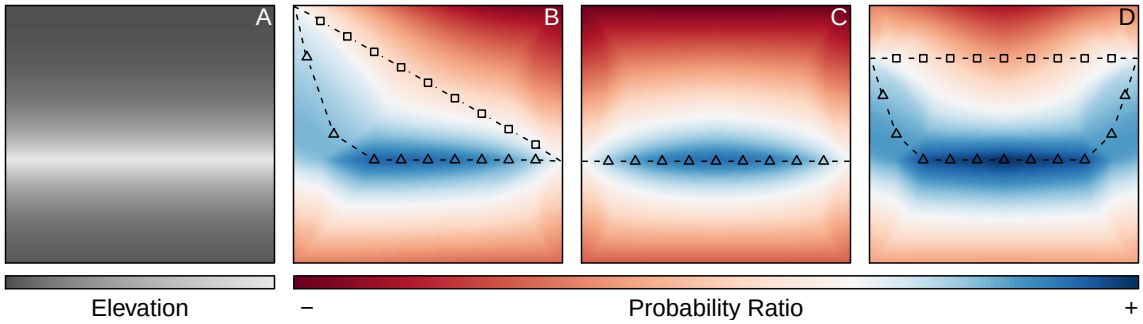

Figure 3.2: Examples of Circuitscape Output on a Ridge

A) Landscape with a horizontal ridge. White represents high elevation and therefore high conductance in the alternative model. **B)** Map of the output probability of the alternative model divided by that of the null model where the input landscape is the ridge in Panel A. In blue cells, the ratio is greater than 1, such that the true path is more likely to pass through that cell under the alternative model than under the null model; in red cells, the ratio is less than and the alternative is less likely than the null; and in white cells the ratio is equal to 1, and the models are equally likely. Two possible sets of observed locations X are shown, each starting on top of the ridge at right, and ending at the bottom of the slope at top left. Open triangles would provide evidence for the alternative model: they follow the ridge for as long as possible, before descending to the destination. Open squares represent evidence for the null model: they follow a direct route irrespective of topography. **C)** Same as Panel B, except that both origin and destination lie on the ridge. Here, the direct path provides evidence for the alternative model because the direct path follows the topography. **D)** Same as panel B and C, except paths start and end at low elevation. Diverting to the ridge would support the alternative model, while taking the direct route through the valley would support the null model.

If we assume each inter-bout track is independent, then the log-likelihood of a given model is equivalent to the sum of $logP(X|\theta)$ over all inter-bout tracks. This is not, however, a true likelihood because the Circuitscape is deterministic, not stochastic. As a result, parametric methods for comparing model log-likelihoods, such as the Likelihood Ratio Test, which assumes the total log-likelihood is chi-squared distributed, are not appropriate for making inference. Instead, because the underlying distribution of probabilities is unknown, we conducted a non-parametric permutation

test to determine whether the log-likelihood of each alternative model is significantly greater than that of the null model. Specifically, we calculated the log-likelihood of each track given the alternative model (i.e., the sum of the log-probabilities of all conditional steps in that track) and then calculated a corresponding list given the null model. We randomly swapped corresponding elements for the same tracks between the two lists and then compared the total log-likelihood of the two lists. We repeated this 10,000 times to generate the distribution of log-likelihood differences between the null and alternative models due to random chance. This gave us a context in which to place the actual observed difference between the null and alternative models and calculate a p-value.

The value of θ is arbitrary, so we conducted the analysis for a range of θ values. To present this range of model-specific θ values on a meaningful scale, we define Δ_{10} as the probability ratio of climbing 10m versus remaining at the same elevation:

$$\Delta_{10} = \frac{P(\alpha_x + 10)}{P(\alpha_x)} = \frac{2 \cdot \theta^{0.125}}{(1 + \theta^{0.125})} \tag{3.3}$$

where α_x is the elevation in a given cell x, $P(\alpha_x)$ is the probability of moving from cell x to a neighboring cell with the same elevation, and $P(\alpha_x + 10)$ is the probability of moving to a neighboring cell that is 10m higher in elevation. This ratio can then be expressed as a function of θ (see Appendix B for derivation). We conducted our analysis over a range of θ values such that Δ_{10} varies from 1 in the null model to a maximum of 5.

3.3 Results

Our data processing yielded 287 focal samples from 21 individual monkeys (7 males and 14 females) where the median duration of a focal sample is $15min$ and each

sample contains, on average, 4.35 inter-bout tracks. This resulted in a total of 1,249 distinct inter-bouts tracks where the median number of observed locations in each track is 6 (including the origin and destination). The median track displacement, or the distance from the origin cell to the destination cell, is $84.85m$, while the median track length, or the sum of all sequential distances in the track, is $161.29m$. We chose the median to describe the central tendency because all of these distributions are skewed left, with long right-handed tails that strongly bias the mean upwards.

Given the observed data, the likelihood of the alternative model is significantly greater than that of the null model for all values tested here ($1.05 \leq \Delta_{10} \leq 1.8$), and therefore we reject the null model. The p-values resulting from the Permutation Tests were effectively zero for all Δ_{10}: in none of the 10,000 random permutations was the difference between the alternative and null model greater than the observed difference between them. The likelihood of the model increased with Δ_{10}, reaching a maximum at $\Delta_{10} = 1.6$ and then decreasing with higher values of Δ_{10} (Fig. 3.3).

3.4 Discussion

Our results demonstrate that elevation plays a significant role in path selection in spider monkeys. Models that include a preference for high-elevation routes explain the observed movements significantly better than a pure random walk model. The model best supported by the data is the one in which, when given the choice between two neighboring cells, an animal is 1.6 times more likely to move up $10m$ than remain at its current elevation.

There are different ways elevation could conceivably shape movement decisions, so it is important to clarify the implications of Circuitscape's structure on the kind of inference that can be made here. Circuitscape does not identify a single ideal path;

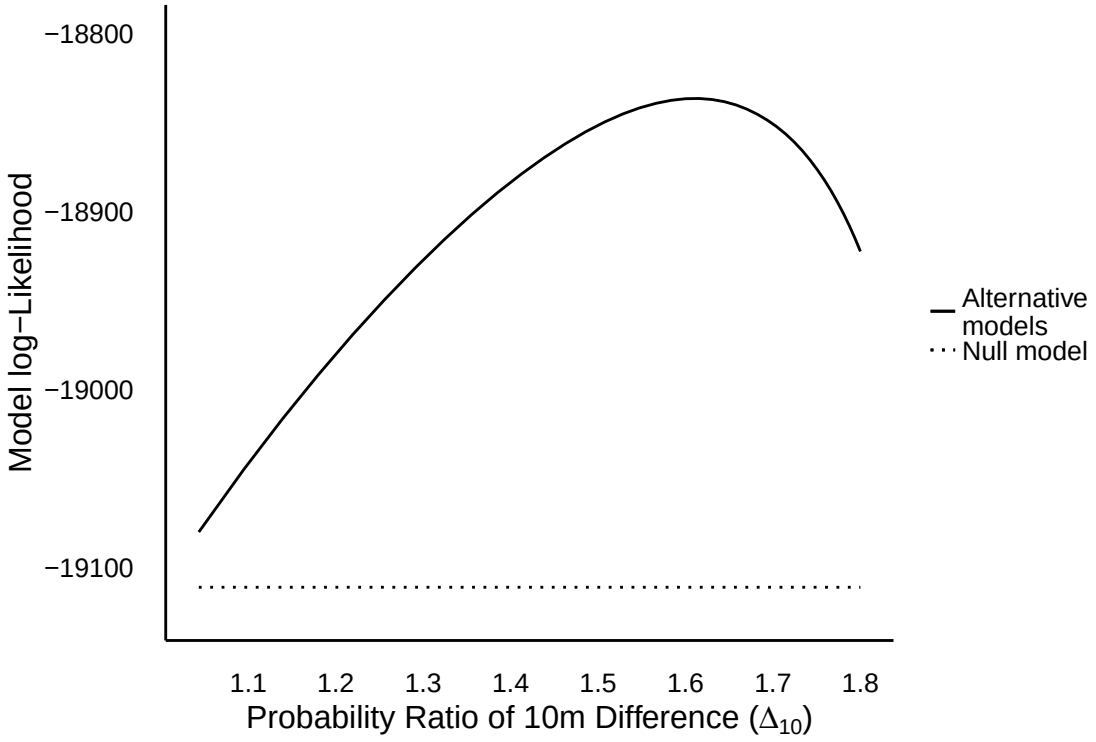

Figure 3.3: Log-Likelihood of Alternative and Null Models

The x-axis shows the value of the input parameter of the alternative model, Δ_{10}, the probability ratio of moving to the higher of two neighboring cells that are 10m apart in elevation. The y-axis shows the log-likelihood of the model given all the data; i.e., the sum of the log-likelihoods of all inter-bout tracks. The log-likelihood of the null model ($\Delta_{10} = 1$) is given as a reference (dotted line).

instead, it aggregates all possible random walks from origin to destination, where the probability of each cell is the normalized frequency of walks through that cell. In the null model ($\Delta_{10} = 1$), monkeys have no preference for elevation, treating the landscape as if it were perfectly flat. As a result, their paths are perfectly random and probability is spread widely across the landscape. The likelihood of the alternative model is not being compared with likelihood of a direct route, but rather with the totally random movement of this null model.

The null model does not, however, produce a perfectly uniform probability map: there is an buildup of probability in the broad swath of cells between the origin and destination. This means a direct route is slightly more probable than an indirect one, even when movement is perfectly random (from an electron's perspective, if resistance is uniform everywhere, the path of least resistance is the shortest path). In the alternative model ($\Delta_{10} > 1$), as preference for elevation increases, probability becomes more concentrated along ridges at the expense of valleys as more paths are diverted from the direct route to pass through higher terrain. Paths will also divert further from the direct route as the attraction to high-elevation cells increases (see Figure 3.2D).

Our results, therefore, indicate that when faced with a choice between different routes to a destination that are equally direct, the routes through high elevation terrain are more probable. This does not necessarily mean that monkeys spend significantly more time at high elevation or that they do not visit low-lying areas. This would be difficult at TBS, where the ridge-lines do not form a single connected network and monkeys would commonly be forced to cross a valley to reach another ridge. Furthermore, when a feeding location is located in a valley, the path must descend to it at some point, even if it follows a ridge-line a far as possible (see Figure 3.2B).

One of the limitations of our methodology is that a map of ground elevation like the one used here does not necessarily reflect the vertical changes in terrain as perceived by monkeys because they travel through the canopy rather than at ground level. For example, a monkey may actually have to descend when moving from a tall, low-lying tree to a short tree on higher ground. A raster map of canopy height could be collected with an aerial Light Detection and Ranging (LIDAR) scan of the site, but this would be expensive and difficult to obtain, and furthermore, the topography of

the treetops may be no more accurate a reflection of route elevation than the ground. The monkeys' true paths lie somewhere between the ground and the treetops, where the actual path elevation is a product of the 3-dimensional branch geometry of the canopy, which is far beyond our ability to map. The existing elevation maps of the ground used in this study are as good a proxy for route height as any other feasible data source.

Our model also made strong simplifying assumptions about the nature of decision-making and memory in spider monkeys. Other than the ability to recognize and stop at their destination, the animals in our Circuitscape models have no spatial memory, while the literature points strongly to some level of spatial memory and awareness in primates. Furthermore, we assumed that inter-bout tracks are independent, such that monkeys have only one destination at a time which is completely forgotten upon arrival and replaced with a new, unrelated target. In reality, spider monkeys probably plan their routes to include multiple destinations. For example, they may choose their path between major feeding trees so as to collect specific information about fruit phenology or harvest minor food sources. Spider monkeys are also highly social, and unexpected social interactions in the middle of a track might divert a monkey from its destination. Monkeys could also have other destinations besides feeding trees, such as sleeping trees or social rendezvous points.

The purpose of this study, however, is to test whether elevation plays a significant role in path choice and our simplifying assumptions allowed us to apply this limited question to a complex data-set. Furthermore, a model in which monkeys construct paths that connect multiple destinations, or that uses an explicitly defined travel network, is far more complex and would not have allowed us to take advantage of the Circuitscape framework. Our results should not, therefore, be interpreted as

supporting the notion that spider monkeys do not possess spatial cognition or the ability to plan multi-destination routes. Instead, our results simply indicate that elevation is an important factor in making movement decisions.

It is still unclear, however, why spider monkeys prefer higher ground. As suggested by Di Fiore and Suarez (2007), ridges may provide spider monkeys with better visibility for monitoring resources from afar. Ridges may also be the most efficient way to avoid the costly energy expenditure associated with crossing a deep valley. The topography at TBS, however, is less extreme than at other sites, which could diminish the advantages of ridge-line travel here. Furthermore, as discussed above, our model conceives of elevation as a useful proxy for a more complex underlying process. Monkeys may not perceive elevation as the primary driver of their decisions, but instead are responding to other landscape variables that are correlated with elevation. For instance, trees that grow at higher elevation may be more desirable for travel or foraging. More detailed data layers would be required to tease apart correlation from causation here.

The value that spider monkeys place on elevation may also depend on the type of mental map they use. If they use a network map, spider monkeys may seek out ridges because they provide easy-to-follow edges in the travel network. This would agree well with observations of spider monkeys using habitual pathways because unlike trees, topographic landmarks are stable through time, thereby encouraging repeated use. If they use a vector map, however, calculating straight-line routes is most efficient. In this situation, rather than simply following a circuitous ridge, they may place greater value on the energy efficiency or visibility afforded by high elevation, combining it with other environmental variables to chart efficient new routes. In truth, however, spider monkeys probably use a combination of both of these mental maps,

each at different times, in different areas, or at different scales, and topography is only one of many factors shaping their choices. For example, spider monkeys may use a network map for relatively long-distance travel, using ridges like a highway, and then use a vector map to descend and forage locally in a restricted area. To understand the role of topography as a part of such a complex decision-making system, future studies need to incorporate elevation into more sophisticated models of primate foraging, such as models that explicitly define a travel network (Suarez, 2014).

Finally, preference for high elevation pathways could also have implications for seed dispersal. Spider monkeys are prolific seed dispersers, in terms of both quantity and diversity of seeds dispersed, and studies show that their movement patterns shape seed deposition patterns in significant ways (see Chapter 4). If spider monkeys deposit seeds preferentially along higher-elevation routes, is the seed deposition pattern significantly related to topography as well? Is there a net gene flow from low to high elevation, and if so how does this interact with tree species' adaptations to specific micro-habitats related to slope or drainage? Elevation could be added to existing models of agent-based seed dispersal to refine our understanding of the impact of foraging decisions on landscape patterns and forest demography.

Chapter 4

Foraging Behavior, Agent-Based Seed Dispersal, and Tropical Forest Community Structure

4.1 Introduction

Understanding what mechanisms maintain species diversity in plant communities is a fundamental question in the field of ecology. This is particularly true in tropical forests, where the diversity of tree species is extraordinarily high, with as many as 1100 species in a 25 hectare plot, and the density of most species is quite low (Valencia et al., 2004). There are a number of empirically supported theories to explain this diversity, but one of the most prominent, proposed by Janzen (1970) and Connell (1971), attributes this diversity to the negative density-dependent effects of plant enemies on seedling survival. Seeds and seedlings underneath the crown of their parent tree suffer high mortality because the high density of genetically-related individuals attracts herbivores and facilitates the spread of disease. Seed dispersal is the primary mechanism by which progeny escape this high-mortality area, and as such, it is vital to the functioning of a tropical forest ecosystem (Howe and Miriti, 2000). This displacement of seeds away from maternal source trees also facilitates gene flow across landscapes (Bacles et al., 2006) and establishes the spatial template of seed deposition upon which the distributions of all subsequent tree cohorts are based (Jordano and Godoy, 2002).

In tropical forests, frugivorous vertebrates are responsible for the vast majority of seed dispersal from fruiting canopy trees (Terborgh, 1990). As these animals

search the landscape for resources, they consume fruit, swallow the seeds, and deposit them intact at subsequent locations along their search path, which is chosen according to their foraging behavior (Côrtes and Uriarte, 2012). Though all animal movement contains a random element (Bartumeus et al., 2005), these foraging patterns are often spatially structured (Morales et al., 2004), especially on heterogeneous resource landscapes, where even simple search strategies have been shown to produce movement patterns with fine scale structure that reflect landscape patterns (Boyer et al., 2006; Nonaka and Holme, 2007). This spatial structure may also persist over time; for example, some foragers repeatedly use established travel routes to monitor well-known, high-quality resources, a behavior called trap-lining (Thomson et al., 1997). If agents repeatedly deposit seeds along these kind of routes as they forage, the pattern of resulting trees might conform to these same routes. Given this, how do the search behaviors of seed-dispersing agents, which determine these routes, affect the spatial distribution of tropical tree species?

Due to the long time scales involved in plant growth and demographics, simulation modelling is an effective tool for exploring this question. In an exhaustive review of agent-based seed-dispersal studies, Côrtes and Uriarte (2012) found that each step of the seed-dispersal process had been studied thoroughly, but in relative isolation. They suggested an integrated modelling approach that combines existing models of seed-dispersal (Clark et al., 1999; Nathan and Muller-Landau, 2000) with movement ecology models that emphasize mechanistic and spatially-explicit movement at fine-scale (Nathan et al., 2008; Schick et al., 2008). Wang and Smith (2002) went further to suggest that ecologists consider forest dynamics as well, in order to "close the seed-dispersal loop" that starts with seed production and begins again with the growth of new fruit-bearing trees (Fig. 4.1).

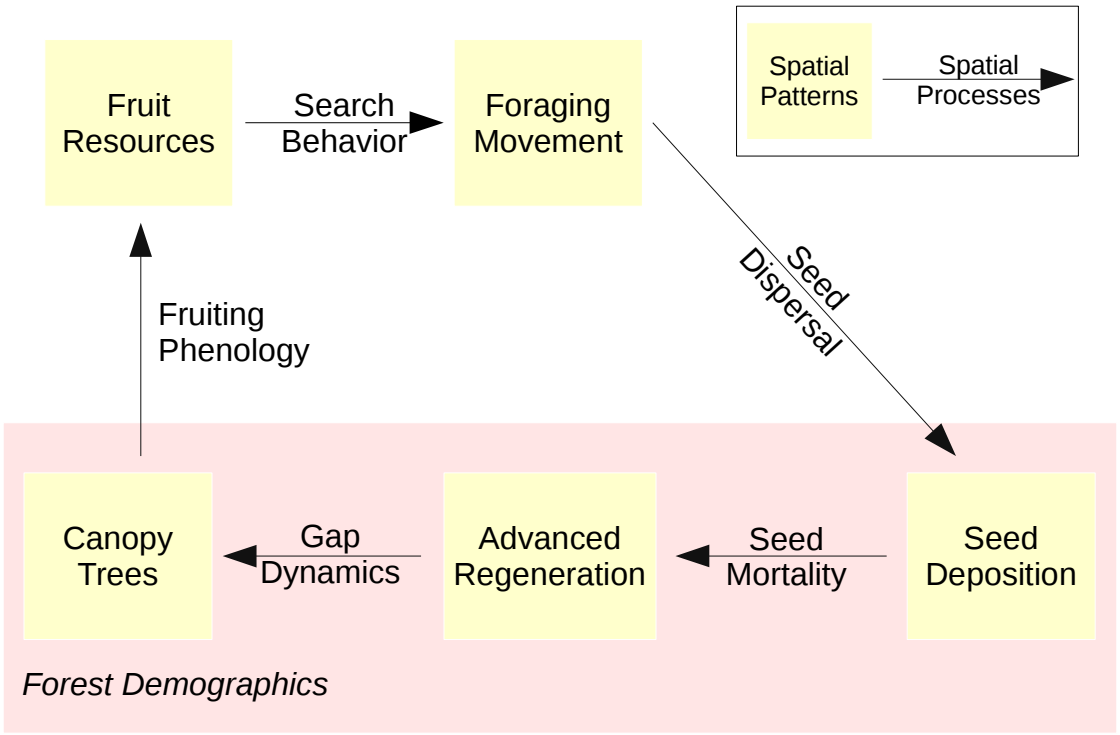

Figure 4.1: Concept Map

The seed-dispersal process as a closed-loop, integrating both forest demographics and the movements of vertebrate dispersers.

Empirical studies demonstrate that the movement patterns of vertebrate dispersers can indeed impart spatial pattern to the density (Fragoso, 1997; Russo and Augspurger, 2004; Karubian et al., 2012) and genetic structure (Jordano and Godoy, 2002; Karubian et al., 2010) of dispersed seeds. Once the seeds are deposited, however, many spatial processes intervene in the development from seeds to adults. Seeds may be destroyed by seed predators on the ground or secondarily dispersed to another location (Andresen, 1999). Many seeds fail to germinate due to micro-habitat quality (Jordano and Schupp, 2000), and many of the established seedlings also die, due to competition, disease, and herbivory (Wang and Smith, 2002). Even then, saplings can

only emerge as canopy trees when treefalls or other disturbances creates forest gaps (Denslow, 1987). Spatial bias in any step of this filtering process would alter the ultimate adult distribution with respect to the original seed pattern. Studies show that if seedling recruitment is dispersal-limited, rather than establishment-limited, disperser movement patterns can carry from the seed distribution into the seedling distribution (Fragoso, 1997; Julliot, 1997; Howe and Miriti, 2004; Russo and Augspurger, 2004), but there is little evidence as to whether these patterns propagate through to the adult distribution in a significant way.

Our integrated model of seed dispersal also contains higher-order feedback between movement and landscape processes: the movement of dispersers alters the landscape upon which they likely base their subsequent movement decisions (Di Fiore and Suarez, 2007). In a real forest system, the time scales involved may be so different that this two-way interaction would simplify to a one-way effect of landscape on movement: agents respond to changes in the forest on a time scale of hours or days, while the forest canopy requires years to incorporate changes in the seed-deposition pattern. To study dispersal as a closed loop, however, we must acknowledge the existence of this feedback and understand its basic spatio-temporal dynamics. Are there any realistic conditions under which foragers and forests could significantly shape one another's spatial patterns through time? How does the temporal scale of forest growth affect the strength of this interaction?

Spider monkeys (g. *Ateles*) are an excellent taxa with which to study the impact of movement on forest structure because they are important and prolific dispersers of neotropical tree seeds (Peres and van Roosmalen, 2002; Russo and Augspurger, 2004; Link and Di Fiore, 2006). The monkeys' repeated deposition of large quantities of seeds along established travel routes create strong patterns that

are more likely to persist to the emergent stage. Their movements are also both highly structured and highly repetitive (Di Fiore and Suarez, 2007), forming narrow, network-like, or *reticulated*, patterns that circumvent and enclose large empty spaces. Boyer and Walsh (2010) developed a simulation model of spider monkey foraging that succeeded, to some extent, in reproducing these search paths from relatively simple behavioral rules. We seek to address the above questions with the aforementioned integrated approach, by combining the B&W model of movement with simple models of forest demography, connected by memory-based foraging behavior and mechanistic, individual-based seed dispersal.

4.2 Methods

4.2.1 Model Structure

We have developed a discrete-time, spatially explicit, agent-based simulation model that links foraging, seed dispersal, and forest growth in a closed cycle. In our model, a population of A individual agents simultaneously forage over a forest landscape of 50 x 50 hexagonal cells, each of which contains an adult fruiting tree and a bank of seedlings (or trees at various sub-canopy stages of advanced regeneration). We use a hexagonal lattice because it lacks directional bias, unlike a square grid in which the distance between diagonal neighbors is longer than between cardinal neighbors. Each tree belongs to one of two species: the focal species, for which the agents forage, and a matrix species, which the agents ignore. The foraging process proceeds over time steps equivalent to roughly 10 minutes, while phenology and forest growth occur at daily intervals of 100 steps, corresponding to about 12 hours of daylight foraging. Each day, fruit are added to each tree according to its size, k, which is drawn from a

power law distribution:

$$f(k) = k^{-\lambda} \qquad k_{min} \leq \ k \leq \ k_{max} \tag{4.1}$$

using a discrete approximation of $f(k)$:

$$f(k) = \left\lfloor \frac{k_{min} - 0.5}{(1-r)^{\frac{1}{\lambda-1}}} + 0.5 \right\rfloor \tag{4.2}$$

where r is a uniform random variable between 0 and 1 (Clauset et al., 2009). Each fruit will stay on the tree for n_{rot} days, after which the fruit rots off and is no longer available for consumption.

Each agent remembers the size and location of focal trees it has visited, updating this list as it forages and the landscape changes. Agents also understand the daily addition of fruit and can therefore predict the number of fruit on a known tree without direct observation, although this prediction may be an overestimate due to unobserved consumption by other agents. During each time step, an agent is either eating or moving. If the agent's current cell has a focal tree with fruit, the agent will consume r_c fruit and remain in that cell. Multiple agents may feed, sequentially, on the same tree during the same step. If the cell contains no consumable fruit (i.e., a matrix tree or an empty focal tree), the agent moves to one of its six neighboring cells. The cell is chosen according to the mode of movement employed by the agent in that time step: random or directed. In directed mode, the agent moves toward the target cell, j, from its memory that maximizes the efficiency function, $e_i(t)$, evaluated from the current cell, i, at time step t, as given by

$$e_i(t) = max_j \left[\frac{F_j(t)}{d_{ij}} \right] \tag{4.3}$$

where $F_j(t)$ is the estimated number of fruit on the focal tree in cell j at time t. In random mode, the agent chooses a neighboring cell randomly, as in a random walk.

Parameter	Description	Default		
λ	power law exp. for tree size distribution	3		$*$
η	random walk factor	0.001		
A	number of agents in population	10	inds.	
g	seed retention time	$2 \rightarrow 20$	steps	
P	initial proportion of focal trees	0.1		
α	density dependence parameter	3		
β	density dependence parameter	$(\frac{1-P}{P})\alpha$		
k_{min}	minimum tree size	7	fruit	
k_{max}	maximum tree size	1350	fruit	
n_{rot}	fruit ripening period	7	days	$*$
r_c	fruit consumption rate	3	fruit/step	$*$
τ_A	average lifespan of adult tree	$10 \rightarrow 30$	yr	
m	daily mortality rate of adult tree	$\frac{1}{365 \cdot \tau_A}$		
τ_s	lifespan of seedling	2	yr	

Table 4.1: Parameter descriptions and default values
Values marked with an asterisk ($*$) are drawn directly from Boyer and Walsh (2010).

In any given time step, the probability of random movement is μ and the probability of directed movement is $1 - \mu$, where

$$\mu = exp\left(\frac{-e_i(t)}{\eta}\right) \tag{4.4}$$

and η controls the frequency of random movement. As η increases $0 \rightarrow \infty$, agents are more likely to ignore the best resources and randomly explore the landscape.

It is important to note that agents cannot leave the 50 x 50 cell area. Because all movements are discrete steps between neighboring cells, this strict boundary condition does not necessarily imply the reflection of movement paths. Instead, agents in edge cells do not have the option to move beyond the landscape because edge cells simply have fewer neighbors (i.e., only those cells within the landscape). This boundary condition is analogous to the territorial boundaries observed in groups of spider monkeys which individuals in the wild rarely cross.

When an agent consumes a fruit, the seed is ingested, carried for exactly g

time steps, and deposited into whichever cell the agent is located at that subsequent time. Upon deposition, the seed immediately germinates and joins the seedling bank in that cell, and remains there for exactly τ_s years. Each day, all the canopy trees on the landscape are subjected to the same mortality rate, m, and when a tree dies, a gap is formed in that cell.

Open gaps are filled with new canopy trees according to a negative density-dependent process that stabilizes the abundance of focal trees around the initial proportion of focal trees on the landscape, P. When a gap forms, a seedling is randomly chosen from among that cell's bank. The probability, $\sigma_{i,corr}$, that the emergent seedling is from the focal species, is given by

$$\sigma_{i,corr} = \frac{\sigma_i \cdot \delta}{\sigma_i \cdot \delta + (1 - \sigma_i)(1 - \delta)} \tag{4.5}$$

where σ_i is the raw proportion of focal seedlings in cell i, and δ is a density dependent factor given by

$$\delta = I_{p(t)}(\alpha, \beta) \tag{4.6}$$

where $p(t)$ is the proportion of focal trees at time t and $I_{p(t)}$ is the incomplete regularized beta function, or CDF evaluated at $p(t)$, such that

$$\frac{\alpha}{\alpha + \beta} = P \tag{4.7}$$

Thus, the chance of a focal seedling emerging to the canopy is equal to the local abundance of focal seedlings biased by the global abundance of focal trees. If the actual abundance of focal trees is less than the target abundance ($p(t) < P$), then the chance that the next tree will be focal increases ($\sigma_{i,corr} > \sigma_i$), and vice versa.

4.2.2 Parameterization

In order to investigate the effects of tree lifespan τ_A and gut-retention time g on the resulting spatial structure of the landscape, we varied the values of τ_A and g. To select the range of tree lifespan values, we began with a realistic annual mortality rate for large canopy trees in tropical forests (Condit et al., 1995) and then increased this value to observe the effect of higher forest turnover. This range of mortality rates (m) translates to an average adult tree lifespan (τ_A) of 30 down to 10 years. Because the survival curve is negative exponential, however, 10% of the trees live more than 23 years at the low end ($\tau_A = 10$) and 69 years at the high end ($\tau_A = 30$).

All other parameters were kept constant. Where our parameters are equivalent to those in the B&W model, we used the same values (Table 4.1). We chose the remaining parameter values to agree reasonably with empirical observations or to facilitate model function. In terms of order of magnitude, our troupe size of $A = 10$ is consistent with wild spider monkey groups (Symington, 1990; Link and Di Fiore, 2006). We also set the extent of the landscape to be roughly consistent with the size of observed spider monkey territories: if each cell represents the area of the crown of a canopy tree, roughly $20m$ across, 50 x 50 cells is equivalent to $1km^2$. The vast majority of tropical seedlings in shade conditions die within a year (Augspurger, 1984), but we gave our seedling bank a slightly longer lifespan (τ_s) because this pool includes more advanced stages of tree development as well.

We set the minimum tree size ($k_{min} = 7$) such that the smallest tree on the landscape receives at least one fruit per day. Given this minimum and the exponent of the tree size distribution ($\lambda = 3$), we set the maximum tree size ($k_{max} = 1350$) such that the average tree size is the same as that in the B&W model. In setting the initial proportion of focal trees, P, we found 10% was ideal for identifying clear spatial

patterns, being neither too sparse nor too crowded. Although resource intake rates are most efficient in the B&W model when agents explore the landscape randomly with some regularity ($\eta \approx 1$), we chose a more deterministic search ($\eta = 0.001$) in order to induce more habitual movement patterns, like those observed in wild spider monkeys (Di Fiore and Suarez, 2007).

4.2.3 Analysis

In order to analyze model output, we quantified the level of reticulation in the configuration of focal trees on the landscape. While classic measures of spatial pattern like Ripley's K (Dixon, 2002) can identify coarse clustering, they are not able to identify the more subtle reticulation pattern described above, in which trees coalesce into elongated, network-like patterns that enclose empty space. Instead, we used a simple metric for reticulation based on the Relative Neighborhood Graph (RNG), a standard method for defining a sparse, planar topology for a set of points, which is intended to mimic the way humans might perceive the pattern (Toussaint, 1980). In the RNG, two points are only connected by an edge if there are no other points that are closer to both of them than they are to each other. As a result, the RNG tends to avoid direct paths between two points that jump over a large empty space if there is an alternative path that uses smaller steps, even if it is longer or more circuitous. This makes it ideally suited to identifying reticulation.

Here, we constructed the RNG as an undirected, weighted graph such that the weight on each edge, $e_{i,j}$, is equal to the Manhattan distance, $M_{i,j}$, between vertices i and j (i.e. the discrete number of steps between cells i and j on the hexagonal lattice). We used Manhattan distance here, rather than Euclidean distance, because it more accurately reflects the actual movement distance experienced by the agents

in the course of the simulation. If $S_{i,j}$ is the shortest path on RNG, accounting for edge length, between each pair of vertices i,j, then the sum of the edge lengths of the *longest* of these shortest paths is the diameter. Our reticulation metric, R is given by

$$R = \frac{D(RNG) - M_{I,J}}{max[M_{i,j}]} \tag{4.8}$$

where $D(RNG)$ is the diameter of RNG, $M_{I,J}$ is the Manhattan distance between the two vertices (I and J) whose shortest path is equal to the diameter, and $max[M_{i,j}]$ is the maximum pairwise Manhattan distance on the landscape. If there are multiple vertex pairs that produce the diameter, then an average of their Manhattan distances is used. In effect, R tells us how much further it is to travel between I and J on RNG than to go directly as the crow flies, as a percentage of the overall size of the landscape. If the landscape is highly reticulated (Fig. 4.3B), then travelling around the empty space on the RNG will take much longer than going directly, and R will be high. In contrast, if trees are more evenly distributed (Figure 4.3A,C), the RNG will not deviate significantly from a direct path, and R will be low.

We ran the simulation for different values of g (2-20 steps) and τ_A (10-30 years), with 10 repetitions for each parameter set. For every simulation run, we calculated reticulation, R, for each year and smoothed this time series using the cubic spline function "smooth.spline" from the "stats" package in R (R Core Team, 2015). The reticulation is subject to significant fluctuations over time as small changes in tree configuration can sometimes significantly alter the connections in the RNG and, consequently, the shortest paths across the RNG. Therefore, to generate a usable time series, we chose the highest smoothing parameter value ($spar = 1$) allowed within the safe usable range indicated in the documentation. From each of these smoothed time series, $R'(t)$, we extracted three values: the final reticulation, $R'_f = R'(t = 500)$,

measured at the end of the simulation (year 500); the maximum reticulation value, R'_{max}; and the year, t_{max}, in which this occurs, such that $R'(t_{max}) = R'_{max}$.

4.3 Results

The spatial extent of the focal-tree distribution, or the maximum pairwise Manhattan distance between focal trees, $max[M_{i,j}]$, decreased over the life of the simulation, without exception: in none of the more than 1000 simulations was the extent greater at the end than the beginning. The extent of this contraction is strongly dependent on adult tree lifespan (τ_A) and, to a lesser extent, gut-retention time (g), with the greatest contraction occurring at high g and low τ_A (Fig. 4.2). Within each simulation time series, the extent of focal trees continually decreased over time: $max[M_{i,j}]$ only increased 312 separate times over a one-year period, in the more than 500,000 total annual time intervals from more than 1000 simulations.

The smoothed time series of Reticulation, $R'(t)$, however, generally peaks at some intermediate year, t_{max}, beginning with a random landscape and low reticulation values (Fig. 4.3A) and increasing to peak reticulation, R'_{max} (Fig. 4.3B), before declining to R'_f at the end of the simulation as the extent contracts (Fig. 4.3C). gut-retention time, g, has a negative relationship with the timing and height of the peak (t_{max} and R'_{max}, respectively), but the relationship saturates at high values of g (Fig. 4.4A). For $g < 13$, g is strongly correlated with R'_{max}, t_{max} and R'_f, when controlling for τ_A, but there is almost no correlation when $g \geq 13$ (Table 4.2). Tree lifespan, τ_A shows a consistent, positive correlation with t_{max} and R'_{max}, when controlling for g, over the entire range of g values (Table 4.2, Fig. 4.4B). There is only a weak positive correlation, however, between τ_A and the final reticulation, R'_f, and only when g is large (Table 4.2).

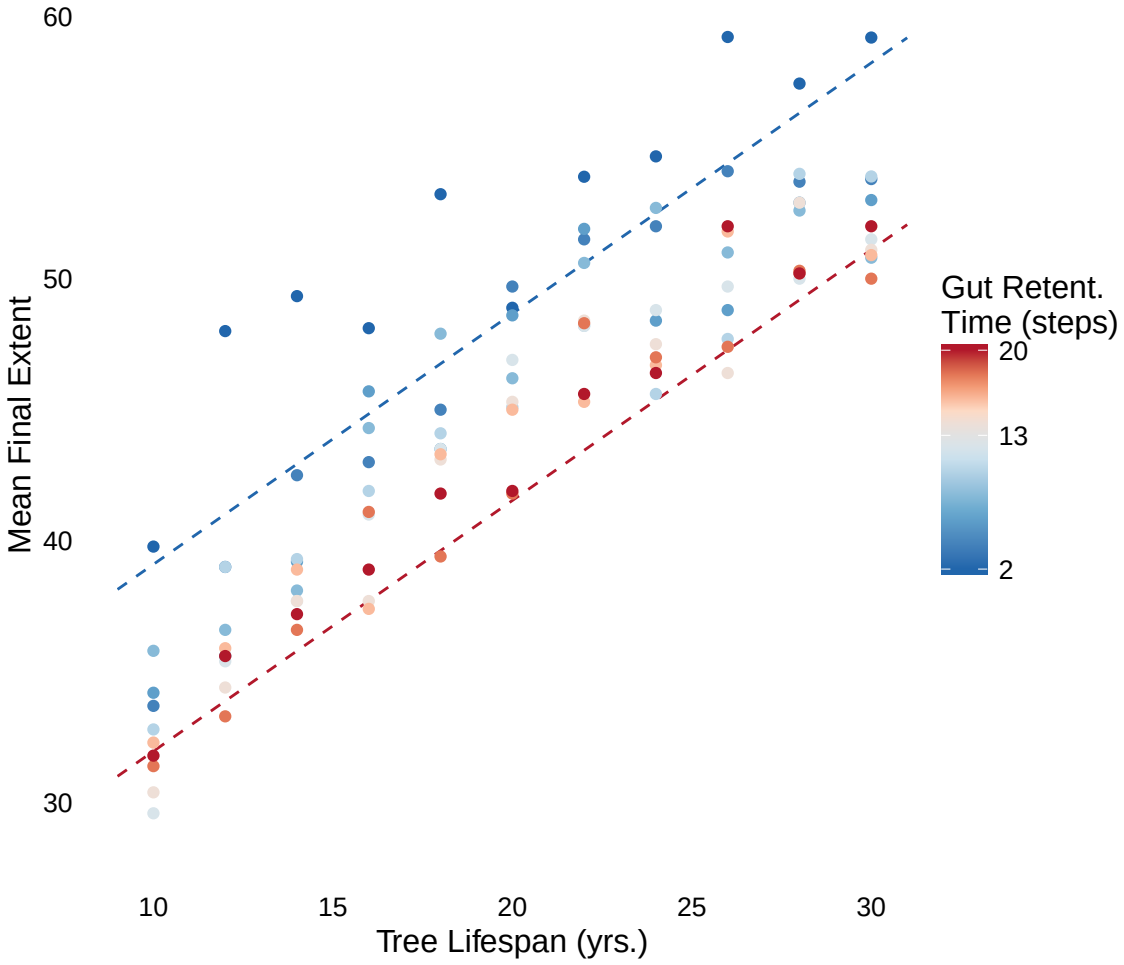

Figure 4.2: Focal Pattern Extent at End of Simulation

Mean final extent is the maximum pairwise Manhattan distance between focal trees in year 500, $max[M_{i,j}]$, averaged over 10 simulations. gut-retention time is g and tree lifespan is τ_A. The dotted lines are trend lines from the multiple regression $max[M_{i,j}] = \beta_0 + \beta_1 g + \beta_2 \tau_A$, for the two extreme values of g, 2 (bottom) and 20 (top). Slope $\beta_1 = 0.957$, intercept $\beta_0 = 30.325$ and $\beta_2 = -0.396$. The effect size of the interaction between g and τ_A was very small and so was not included in this regression. Because the sample sizes of our simulated data are arbitrarily constructed, p-values have no inferential significance and are not reported here. For context, all simulations begin with a random landscape, whose mean extent is 71.618 ($sd = 1.539$).

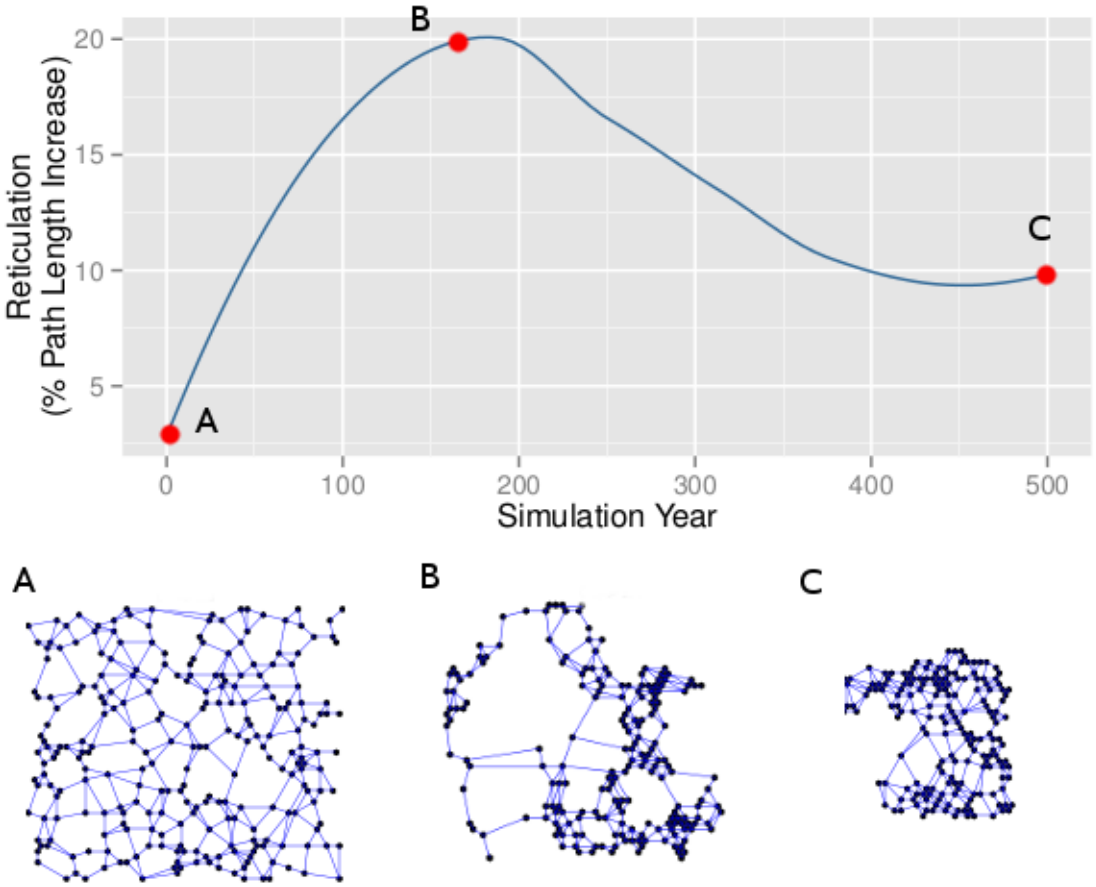

Figure 4.3: Example Output of Single Simulation Run

The time series above, smoothed using a cubic spline, is an example of the output of a single simulation run ($g = 14$ and $\tau_A = 18$). The diagrams below are actual landscapes from the corresponding time points in this simulation, over which their Relative Neighborhood Graphs have been superimposed in blue.

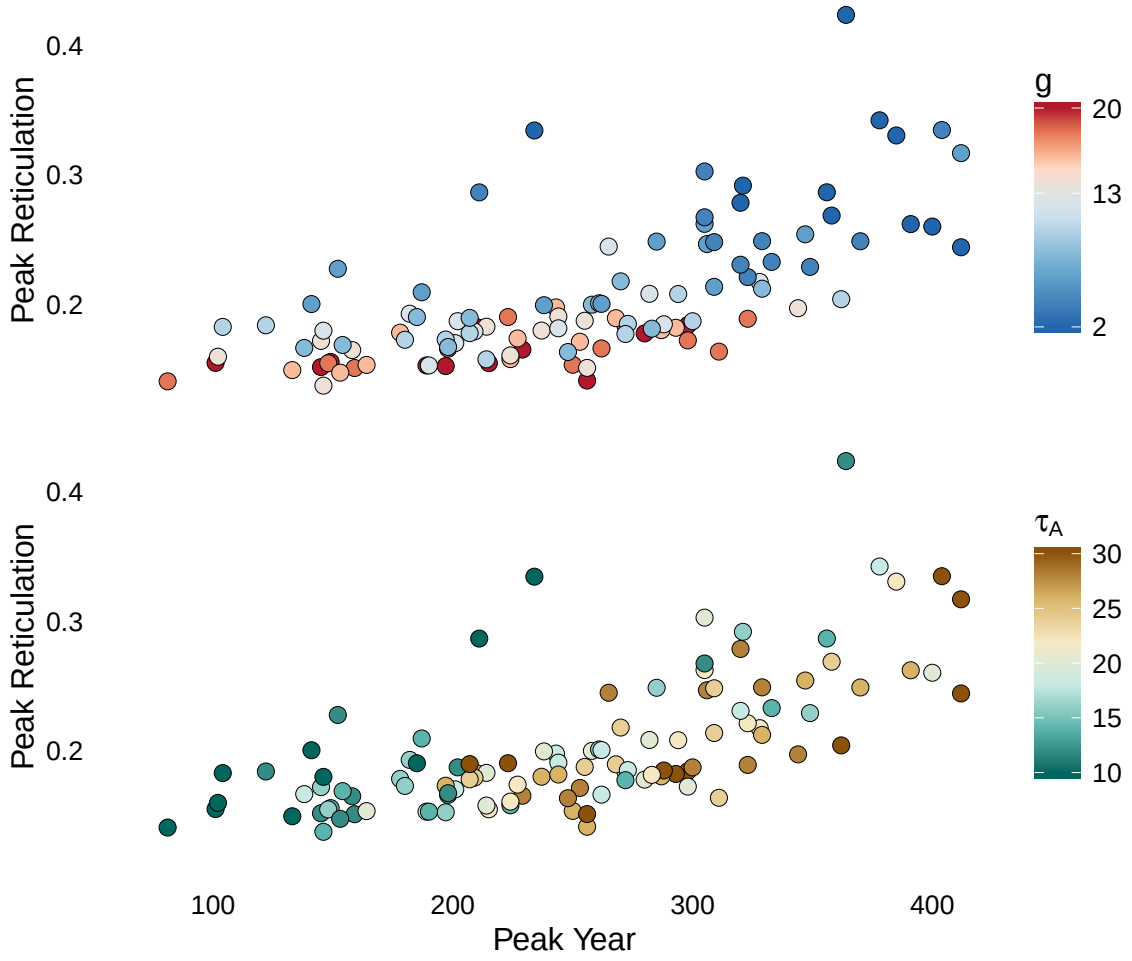

Figure 4.4: Timing vs. Intensity of Peak Reticulation

For each simulation run, we smoothed the annual time series of R using a cubic spline, and extracted the peak R and the year in which that peak occurred. The points above are averages of these values from the 10 simulations run for each set of g and τ_A parameter combinations.

	$g < 13$			$g \geq 13$		
	R'_f	t_{max}	R'_{max}	R'_f	t_{max}	R'_{max}
g	-0.3592	-0.3284	-0.4372	-0.0083	-0.0083	-0.0543
τ_A	0.3175	0.3089	0.0391	0.4021	0.3563	0.1753

Table 4.2: Partial Spearman Correlation Results

Each column shows the coefficient estimates from one of six separate partial Spearman rank correlation tests (Kim, 2015) between gut-retention time, g, tree lifespan, τ_A, and one of the model outputs (R'_f, t_{max} and $R'max$) for low or high values of g. Because the sample sizes of our simulated data are arbitrarily constructed, p-values have no inferential significance and are not reported here.

4.4 Discussion

We observe two major emergent patterns in our simulated tree populations. First, over long periods of time, the dispersal process collapses the tree distribution towards the center of the foraging arena as forager movement increasingly avoids the margins. When agents visit trees on the outer edges of the focal range, they tend to bring those seeds back towards the center of the landscape, drawn by the higher concentration of fruit resources. Over time, those outlying focal trees die and are replaced by matrix trees and the focal distribution contracts. It is important to reiterate that in this model, undispersed seeds fall directly beneath the parent, meaning the trees alone have no power to expand or contract their range. In the absence of dispersing agents, the distribution of focal trees would remain perfectly constant through time.

The rate of the contraction is controlled by the adult tree lifespan, τ_A. When lifespans are short, frequent tree mortality accelerates the process of replacing outlying focal trees with matrix trees. When lifespans are long, forest turnover is low and the canopy tree distribution responds more slowly to changes in the seedling banks laid out by the agents.

The second major emergent pattern we find is that, at intermediate time-scales the spatial placement of trees becomes reticulated owing to trap-lining behavior of foragers. Here, low gut-retention time, g, results in more intense reticulation patterns. This is because short retention times translate to average dispersal distances of only a few steps from the parent tree. This preserves adult tree patterns locally, thereby propagating, and even enhancing, those reticulated patterns into the next generation.

In the opposite direction, longer retention times increasingly disrupt reticulation, although this effect eventually saturates for two reasons. First, the discrete nature of the landscape creates a natural minimum for reticulation. As the focal tree pattern collapses, there is less room for distinct open spaces or reticulated paths and reticulation declines, but trees can get no closer than the width of a single cell. Once the focal tree pattern reaches this fully contracted point, reticulation can go no lower and an increase in retention time can have no further effect. Second, the negative effect of long gut-retention times on reticulation saturates because longer post-consumption paths are more likely to reflect off the borders of the landscape. This path recursion severs the correlation between retention time and dispersal distance, such that a further increase in retention time has little effect on actual seed displacement.

The reticulation at the end of the simulation is a snapshot in time during the process of peak and decline, such that the variation in endpoint values largely reflects the rate of contraction. When tree lifespan (τ_A) is low and turnover is high, the forest has raced through the entire process by year 500, ending with low values for extent and reticulation. At high τ_A values, year 500 may have caught the simulation just after the peak, before it has had a chance to significantly decline, or maybe even before the peak, while reticulation is still rising. The partial correlation analysis

strongly supports this interpretation: rising τ_A shifts the reticulation peak forward in time, but it does not significantly alter the strength of the pattern. Turnover rate is not affecting the pattern itself, only how quickly it forms and then in turn, is washed away.

Our model results demonstrate that a seed dispersal agent can, in theory, influence the spatial forest structure owing to directed foraging movements between high-value rewards. When considering whether a theoretical feedback between movement and landscape could have significant effects in reality, one might imagine that real forest canopies do not turnover quickly enough to reflect the swiftly changing dispersal patterns of the agents. These results suggest the opposite: under fast turnover, the contracting effects of agent-based dispersal overwhelm any small-scale structure, while slow turnover allows time for the reticulating effects of dispersal to take effect, albeit temporarily, before the landscape contracts. There is, theoretically, still some average tree lifespan much higher than 30 years that would grind turnover to a near halt and prevent even reticulation. Our model, however, suggests that realistic, intermediate forest turnover permits subtle, small-scale, though possibly transient, pattern formation.

Our model examines the process of dispersal agent-based pattern formation in relative isolation, but this is an important start to understanding how it might function as one of many competing spatial processes in a real tropical forest. In reality, tropical tree density is generally quite low and trees do not universally collapse to the center of the forest, but that does not necessarily contradict our results. The simulation suggests that agent-based dispersal exerts strong pressure to contract the extent of resources, which in a real forest would be coincident with and counteracted by negative density-dependent processes that spread the distribution. For instance,

we would expect Janzen-Connell effects at a cellular scale to increase the survival of rare focal seedlings at the edges of the landscape and increase mortality of common focal seedlings at the center, thereby preserving sparse distributions and preventing their collapse into a single, homogeneous clump. This situation may actually be more conducive to the formation of reticulation as it provides more space and time for agents to shape the landscape without the pattern collapsing.

The reticulation and contraction processes observed here can both be empirically tested on spatial data of foraging spider monkeys and their target tree species. If contraction is occurring, we might expect to see a significantly higher density of focal trees at the center of the monkeys' home range than at the edges. It is, however, important to note that our model did not include patrolling movements commonly exhibited by spider monkeys in the wild, in which males conduct fast-moving patrols along the edges of their group's territory to enforce the borders and ward off neighboring groups. This behavior may counteract the predicted contraction by bringing seeds to outlying areas. If reticulation is occurring, we might expect to see more focal trees along habitual pathways than in the space between these common routes. In either case, the tree species would need to be carefully chosen such that the spider monkeys are responsible for dispersal of the vast majority of that species' seeds. Finally, we hope our integrated approach to dispersal modeling (i.e., connecting movement, dispersal, and plant growth in a closed loop) can be applied to other agent-based dispersal systems. Results from a wider range of disperser behaviors and resource distributions could help illuminate general principles regarding how system attributes affect the strength of movement-landscape feedback.

Appendices

Appendix A

Pollination Model (Ch. 2) Source Code

The following is the source code for the simulation model presented in Chapter 2, including the parent C++ script (model.cpp) and the supporting header files (defs.hpp).

A.1 model.cpp

```
#include <string>
#include "stats.hpp"

struct pars_type {
  std::string rid;
  std::string version;
  int T; // Number of day in a growing season
  int Y; // Number of years in simulation
  int ncol, nrow; // Dimensions of land
  double pi; // Number of days each flower blooms
  double phi; // Foraging trips per nests per day
  double alpha, beta; // Phenology Distribution Shape Parameters
  double init_P; // Initial density of plants in wild habitat
  double init_N; // Initial density of nests on whole landscape
  double gamma; // Germination fraction
  double delta_s; // Annual seed mortality rate
  double delta_p; // Annual wild plant mortality rate
  double delta_n; // Daily nest mortality
  double lambda; // Brood survival rate
  double vartheta; // Dispersal Distribution Parameter
  double mu, nu; // Eclosion Distribution Shape Parameters

  pars_type(std::string rid, double delta_n, double delta_p,
            double alpha, int phpeak, double pi, double phi,
```

```
        double init_N, double lambda, double mu, int brpeak)
  : rid(rid),
    version("3.3"),
    T(180),
    Y(100),
    ncol(256),
    nrow(256),
    pi(pi),
    phi(phi),
    alpha(alpha),
    beta(stats::mode_to_shape(alpha, phpeak, 180)),
    init_P(1),
    init_N(init_N),
    gamma(0.5),
    delta_s(0.1),
    delta_p(delta_p),
    delta_n(delta_n),
    lambda(lambda),
    vartheta(0.0625),
    mu(mu),
    nu(stats::mode_to_shape(mu, brpeak, 180)) {}
pars_type(std::string rid)
  : rid(rid),
    version("3.3"),
    T(10),
    Y(3),
    ncol(4),
    nrow(4),
    pi(2),
    phi(1),
    alpha(1),
    beta(1),
    init_N(0.25),
    init_P(1),
    gamma(0.5),
    delta_s(0.5),
    delta_p(0.5),
    delta_n(0.5),
    lambda(0.5),
    vartheta(0.0625),
    mu(2),
```

```
        nu(2) {}
};

#include "defs.hpp"
int simulation::get_eclosion_date() {
  return get_epoch(y+1, eclosion_dist(engine));
}
void simulation::winter() {
  nests.clear(); N=0;
}
void simulation::year() {
  for (t=0; t<pars.T; ++t) {day();} // Pollinate
  winter(); // What happens to pollinators over winter?
  for (cell_type& i : land) {
    i.poll = false;
    assign_phen(i.id);
  }
}

int main(int argc, char* argv[]) {
  pars_type pars(argv[1], std::stod(argv[2]), std::stod(argv[3]),
                 std::stod(argv[4]), std::stoi(argv[5]),
                 std::stod(argv[6]), std::stod(argv[7]),
                 std::stod(argv[8]), std::stod(argv[9]),
                 std::stod(argv[10]), std::stoi(argv[11]));
  simulation sim(pars);
  sim.init();
  sim.run();
  return 0;
};
```

A.2 stats.hpp

```
#include <boost/math/distributions/beta.hpp>
#include <random>
#include <algorithm>

namespace stats {

int random_round(double x, std::default_random_engine& engine) {
```

```
int base = std::floor(x);
std::bernoulli_distribution roundup(x-base);
return base + int(roundup(engine));
}

struct discrete_beta_distribution {
boost::math::beta_distribution<double> beta_dist;
std::uniform_real_distribution<double> unif_dist;
int n;
discrete_beta_distribution(double a, double b, int n)
      : beta_dist(a,b),
        n(n) {}
int operator() (std::default_random_engine& engine) {
return std::trunc(
      n*boost::math::quantile(beta_dist, unif_dist(engine)));
}
};

double mode_to_shape(double alpha, double mode, int T) {
  return T/mode*(alpha-1)-alpha+2;
}

double mean(std::vector<int>::iterator begin,
            std::vector<int>::iterator end) {
double sum = std::accumulate(begin, end, 0);
return sum / std::distance(begin, end);
}

double rmse(std::vector<int>::iterator begin,
            std::vector<int>::iterator end) {
double x_bar = mean(begin, end);
double mse = std::accumulate(begin, end, 0,
                             [&](double sum, double x){
                                return sum + std::pow(x-x_bar, 2);})
               / std::distance(begin, end);
return std::sqrt(mse);
}

}
```

A.3 defs.hpp

```
#include <cmath>
#include <random>
#include <algorithm>
#include <iostream>
#include <iterator>
#include <map>
#include <unordered_map>
#include <vector>
#include <cassert>
#include <fstream>
#define PI 3.14159265

struct simulation {
  struct point_type {
    double x, y;
    point_type(double x, double y) : x(x), y(y) {}
  };

  template<class T>
  struct grid {
    typedef std::vector<T> storage_type;
    typedef typename storage_type::value_type value_type;
    typedef typename storage_type::iterator iterator;
    typedef typename storage_type::size_type size_type;
    grid(size_type nr, size_type nc) :
      rows(nr), cols(nc), storage(nr * nc) {}
    iterator operator[](size_type r) {
      return storage.begin() + r * cols;
    }
    value_type& operator[](const point_type& p) {
      return storage[int(p.y)*cols + int(p.x)];
    }
    iterator begin() {return storage.begin();}
    iterator end() {return storage.end();}
    size_type size() {return storage.size();}
    size_type rows, cols;
    storage_type storage;
  };
```

```
struct cell_type {
  int id = -1;
  point_type center = point_type(-1,-1); // Center of cell
  bool crop = false; // Is this cell designated as cropland?
  bool plant = false; // Does this cell have a plant on it
  bool poll = false; // Has this plant been pollinated?
  int flowDate = -1; // First day of flowering [0...T)
  int pi = 0; // Number of flowering days for this plant
  int bank = 0; // seed bank
};

// Creates new point at random angle (uniform)
// and distance (negative exponential)
point_type project(point_type xy, double r, double theta) {
  xy.x += r * std::cos(theta);
  xy.y += r * std::sin(theta);
  return xy;
}

// Enforces torus: points outside grid are
// wrapped around back into the grid
point_type wrap(point_type& xy) {
  if (xy.x < 0) {
    while (xy.x < 0) xy.x += pars.ncol;
  } else if (xy.x > pars.ncol) {
    while (xy.x > pars.ncol) xy.x -= pars.ncol;
  }
  if (xy.y < 0) {
    while (xy.y < 0) xy.y += pars.nrow;
  } else if (xy.y > pars.nrow) {
    while (xy.y > pars.nrow) xy.y -= pars.nrow;
  }
}

int rdisp(int i_orig) {
  point_type dest_xy = project(land.storage[i_orig].center,
                               disp_dist(engine),
                               unif_dist(engine)*2*PI);
  wrap(dest_xy);
  return land[dest_xy].id;
}
```

```
  int pick_one(std::vector<int>& V) {
    std::uniform_int_distribution<int> lottery(0, V.size()-1);
    // Choose the "first visit" randomly
    return *std::next(V.begin(), lottery(engine));
  }


/////////////////////////// NEST DYNAMICS ///////////////////////////

  // GENERATE VISITS : LOOP NESTS
  bool is_late_bloomer(const cell_type& i) {
      return i.flowDate > pars.T-i.pi;
  }

  bool is_flowering(const cell_type& i) {
      if (is_late_bloomer(i)) {
          return i.plant &&
                 (t >= i.flowDate || t < i.pi-(pars.T-i.flowDate));
      } else {
          return i.plant && i.flowDate + i.pi > t && t >= i.flowDate;
      }
  }

  bool is_full(const cell_type& i) {return !i.poll;}

  // Ignores visits with no potential for nectar collection
  // or pollination
  void visit(int id_orig, int id_dest) {
    cell_type& i_dest = land.storage[id_dest];
    // If plant is flowering and unpollinated...
    if(is_flowering(i_dest) && is_full(i_dest)) {
        // Add the nest to the lottery for that cell's nectar
        visits[id_dest].push_back(id_orig);
    }
  }

  void gen_visits() { // Generate and record visits
    for (auto& pair : nests) { // pair : {cell id, number of nests}
      // Each nest launches 1 foraging bout from nest to random cell
      for (int b = 0; b < pair.second; ++b)
```

```
      visit(pair.first, rdisp(pair.first));
  }
}

// PICK WINNERS : LOOP VISITS
int get_epoch(int year, int day) {return year*pars.T + day;}
int get_year(int epoch) {return epoch/pars.T;}
int get_day(int epoch) {return epoch%pars.T;}
int get_eclosion_date();

void pollinate(int id_nest, int id_visit) {
  land.storage[id_visit].poll = true; // Pollinate flower
  // Increment pollination event counter
  if (land.storage[id_visit].crop) {++Cc;} else {++Cw;}
  ++brood[get_eclosion_date()][id_nest]; // Mass provision brood
}

// Pick winners to collect resources for home cell
void pick_winners() {
  // pair : {visited cell id, vector of visiting nest origins}
  for (auto& pair : visits)
    pollinate(pick_one(pair.second), pair.first);
  visits.clear();
}

// NEST MORTALITY : LOOP NESTS
int get_ndead(int n) {
  std::binomial_distribution<int> mort_dist(n, pars.delta_n);
  return mort_dist(engine);
}

bool kill_nests(int& n, int ndead) {
  assert(ndead <= n);
  n -= ndead; // Decrement nests in cell i
  N -= ndead; // Decrement landscape nest counter
  // if no nests left in cell i, tag cell i for removal from "nests"
  return n == 0;
}

void nest_death() {
  std::vector<int> to_be_erased;
```

```
  for (auto& pair : nests) {// pair :{cell id, number of nests}
    if (kill_nests(pair.second, get_ndead(pair.second))) {
      to_be_erased.push_back(pair.first);
    }
  }
  // Remove cells that no longer have nests
  for (auto& i : to_be_erased) nests.erase(i);
}

// NEST DISPERSAL : LOOP BROOD[EPOCH]
bool brood_ready() {
  auto it=brood.begin();
  return it != brood.end() && it->first == get_epoch(y,t);
}

int get_nqueen(int b) {
  std::binomial_distribution<int> eclosion(b, pars.lambda);
  return eclosion(engine);
}

bool is_habitat(int id) {return !land.storage[id].crop;}
bool is_habitat(const cell_type& i) {return !i.crop;}

void disperse_queen_to(int dest_id) {
  ++nests[dest_id]; // Disperse there and create new nest
  ++N; // Add to nest counter
}

void nest_dispersal() {
  // If next brood is schedule to eclose today...
  if (brood_ready()) {
    // pair :{cell id, number of brood}
    for (auto& pair : brood.begin()->second) {
      // For each survivor...
      for (int q = get_nqueen(pair.second); q > 0; --q) {
        // Pick Random destination
        int dest_id = rdisp(pair.first);
        // If can nest in crops OR if destination is wild habitat
        if (is_habitat(dest_id)) disperse_queen_to(dest_id);
      }
    }
```

```
      brood.erase(get_epoch(y,t));
    }
  }

////////////////////////// PLANT DYNAMICS //////////////////////////

  void assign_phen(int id) {
    cell_type& i = land.storage[id];
    if (i.plant) {
      i.pi = stats::random_round(pars.pi, engine);
      if (i.crop) {
        i.flowDate = crop_phen_dist(engine);
      } else {
        i.flowDate = wild_phen_dist(engine);
      }
    }
  }

  void plant_death() {
    for (cell_type& i : land) {
      // only wild plants can die
      if (is_habitat(i) && i.plant && plant_dies(engine)) {
        i.plant = false;
        i.poll = false;
        i.flowDate = -1;
        i.pi = 0;
      }
    }
  }

  void seed_dispersal() {
    for (cell_type& i_orig : land) {
      // only pollinated wild plants produce seeds
      if (is_habitat(i_orig) && i_orig.poll) {
        int id_dest = rdisp(i_orig.id);
        // If seed lands in habitat, then add a seed to the bank
        if (is_habitat(id_dest)) ++land.storage[id_dest].bank;
      }
    }
  }
```

```
void seed_death() {
  for (cell_type& i : land) {
    if (i.bank > 0) {
      std::binomial_distribution<int> mort_dist(i.bank,
                                                pars.delta_s);
      i.bank -= mort_dist(engine);
    }
  }
}

void recruitment(int id) {
  cell_type& i = land.storage[id];
  if (i.bank > 0) {
    std::binomial_distribution<int> germ_dist(i.bank, pars.gamma);
    // Draw number of germinating seeds
    int ngerm = germ_dist(engine);
    // If 1+ seeds germinate in empty habitat...
    if (!i.plant && ngerm>0) {
      i.plant = true; // Recruit 1 seedling to adult plant
      // Assign random flowering date
      i.flowDate = wild_phen_dist(engine);
      // Assign discrete flowering period
      i.pi = stats::random_round(pars.pi, engine);
    }
    i.bank -= ngerm; // Remove germinated seeds from seed bank
  }
}

void kill_plant(int id) {
  cell_type& i = land.storage[id];
  i.plant = false;
  i.poll = false;
  i.flowDate = -1;
  i.pi = 0;
}

void disperse_seed(int id_orig, int id_dest) {
  // If seed lands in habitat, add it to the seed bank
  if (is_habitat(id_dest)) ++land.storage[id_dest].bank;
}
```

```
  void prune_bank(int id) {
    cell_type& i = land.storage[id];
    if (i.bank > 0) {
      std::binomial_distribution<int> mort_dist(i.bank, pars.delta_s);
      i.bank -= mort_dist(engine);
    }
  }

////////////////////////////// INITIALIZE ////////////////////////////

  void import_land() {
    std::ifstream file(
      "/home/colin/projects/bombus/init/land/gaussLand." + pars.rid);
    std::string line;
    int row=0;
    while (std::getline(file, line)) {
      std::istringstream iss(line);
      int col=0;
      for (std::string i; iss >> i; ++col)
        land[row][col].crop = bool(stoi(i));
      ++row;
    }
  }

  void init_nests() {
    std::vector<int> habitat_ids;
    for (auto& i : land)
      if (is_habitat(i)) habitat_ids.push_back(i.id);
    // Convert initial density to absolute number of nests
    int NI = pars.init_N * pars.nrow * pars.ncol;
    for (int n=0; n<NI; ++n)
      ++brood[get_eclosion_date()][pick_one(habitat_ids)];
  }

  void init() {
    import_land(); // Sets crop
    for (cell_type& i : land) i.plant = true; // Sets plant
    for (cell_type& i : land) assign_phen(i.id); // Sets pi, flowDate
    init_nests(); // Builds brood
  }
```

```
simulation(const pars_type& p)
    : t(0),
      y(-1),
      N(0),
      over_I(1.0 / p.nrow / p.ncol),
      Rw(0),
      Cw(0),
      Fw(0),
      Rc(0),
      Cc(0),
      Fc(0),
      pars(p),
      engine(std::random_device{}()),
      unif_dist(0,1), // [0,1)
      land(p.nrow, p.ncol),
      plant_dies(p.delta_p),
      seed_germinates(p.gamma),
      disp_dist(p.vartheta),
      crop_phen_dist(p.alpha, p.beta, p.T),
      eclosion_dist(p.mu, p.nu, p.T),
      wild_phen_dist(0, p.T) {
  // assign cell IDs
  int ID = 0;
  for (cell_type& i : land) i.id = ID++;
  // fill cell center values
  for (int m=0; m!=pars.nrow; ++m){
    for (int n=0; n!=pars.ncol; ++n)
      land[m][n].center = point_type(n+0.5, m+0.5);
  }
}

void reset_CFR() {Cc=0; Fc=0; Rc=0; Cw=0; Fw=0; Rw=0;}

void count_FR() {
  for (cell_type& i : land) {
    if (is_flowering(i)) {
      if (i.crop) {
        ++Fc;
        if (is_full(i)) ++Rc;
      } else {
        ++Fw;
```

```
        if (is_full(i)) ++Rw;
      }
    }
  }
}

void print_output() {
  std::cout << pars.rid <<" "<< y <<" "<< t <<" "<< N <<" ";
  std::cout << Cc <<" "<< Fc <<" "<< Rc <<" "<< Cw <<" "<< Fw <<" ";
  std::cout << Rw << std::endl;
}

void day() {
  reset_CFR();
  count_FR();
  gen_visits();
  pick_winners();
  nest_death();
  nest_dispersal();
  print_output();
}

bool runaway_growth() {
  return N > (pars.ncol*pars.nrow);
}

bool is_extinct() {
  return nests.empty() && brood.empty();
}

void winter();

void year();

void run() {
  y = 0;
  while (y < pars.Y && !is_extinct() && !runaway_growth()) {
    year(); // Year 0->Y-1
    ++y; // Year 1->Y
  }
}
```

```
  pars_type pars;
  int t, y;
  double over_I;
  grid<cell_type> land;
  std::unordered_map<int,int> nests;
  std::unordered_map<int,std::vector<int>> visits;
  std::map<int,std::unordered_map<int,int>> brood;
  std::default_random_engine engine;
  std::bernoulli_distribution plant_dies;
  std::bernoulli_distribution seed_germinates;
  std::uniform_real_distribution<double> unif_dist;
  std::exponential_distribution<double> disp_dist;
  stats::discrete_beta_distribution crop_phen_dist;
  stats::discrete_beta_distribution eclosion_dist;
  std::uniform_int_distribution<int> wild_phen_dist;
  int N, Fc, Rc, Cc, Fw, Rw, Cw;
};
```

Appendix B

Derivation of Equations (Ch. 3)

Although the cells of the raster given to Circuitscape are assigned conductance or resistance values, the cells are actually treated by Circuitscape as nodes in the lattice, with connections between cells as the edges over which electrons, or animals, travel. The resistance of the edge from cell x to cell y is the average of the resistance values in the two cells:

$$R_{xy} = \frac{R_x + R_y}{2} \tag{B.1}$$

Because conductance is the reciprocal of resistance, the conductance of this same edge is given by:

$$G_{xy} = \frac{2G_x \cdot G_y}{G_x + G_y} \tag{B.2}$$

If we think of the Circuitscape model as random walk of animals from cell to cell, the probability of an animal currently in cell x choosing to move to one of its neighbors, cell y, is proportional to the conductance of that edge connecting those two cells, G_{xy}. In other words, the probability of an animal moving to the elevation in cell y (α_y), given its current elevation in cell x (α_x) is:

$$P(\alpha_y|\alpha_x) \propto G_{xy} \tag{B.3}$$

If we consider, instead, two neighbors of cell x, cells a and b, the ratio of their probabilities is given by:

$$\frac{P(\alpha_a|\alpha_x)}{P(\alpha_b|\alpha_x)} = \frac{G_{xa}}{G_{xb}} \tag{B.4}$$

By substituting with Equation B.2, we obtain:

$$\frac{P(\alpha_a|\alpha_x)}{P(\alpha_b|\alpha_x)} = \frac{G_a}{G_b} \cdot \frac{G_x + G_b}{G_x + G_a} \tag{B.5}$$

Given Equation 3.1,

$$\begin{aligned}
\frac{P(\alpha_a|\alpha_x)}{P(\alpha_b|\alpha_x)} &= \frac{\theta^{(\alpha_a-\alpha_{min})/(\alpha_{max}-\alpha_{min})}}{\theta^{(\alpha_b-\alpha_{min})/(\alpha_{max}-\alpha_{min})}} \cdot \frac{(\theta^{(\alpha_x-\alpha_{min})/(\alpha_{max}-\alpha_{min})} + \theta^{(\alpha_b-\alpha_{min})/(\alpha_{max}-\alpha_{min})})}{(\theta^{(\alpha_x-\alpha_{min})/(\alpha_{max}-\alpha_{min})} + \theta^{(\alpha_a-\alpha_{min})/(\alpha_{max}-\alpha_{min})})} \\
&= \theta^{(\alpha_a-\alpha_b)/(\alpha_{max}-\alpha_{min})} \cdot \frac{(\theta^{\alpha_x/(\alpha_{max}-\alpha_{min})} + \theta^{\alpha_b/(\alpha_{max}-\alpha_{min})})}{(\theta^{\alpha_x/(\alpha_{max}-\alpha_{min})} + \theta^{\alpha_a/(\alpha_{max}-\alpha_{min})})} \\
&= \theta^{(\alpha_a-\alpha_b)/80} \cdot \frac{(\theta^{\alpha_x/80} + \theta^{\alpha_b/80})}{(\theta^{\alpha_x/80} + \theta^{\alpha_a/80})}
\end{aligned} \tag{B.6}$$

where $\alpha_{max} - \alpha_{min} = 270 - 190 = 80$. This holds if both a and b are diagonal neighbors of cell x, or both are cardinal neighbors of x. If one of a and b is a diagonal neighbor of x and the other is a cardinal neighbor, however, the result will be different by a factor of $\sqrt{2}$ (R_{xy} is $\sqrt{2}$ greater between diagonal cells).

If we consider the special case in which cell a is $10m$ higher in elevation than cells x and b ($\alpha_a = \alpha_x + 10; \alpha_b = \alpha_x$) we can derive the result for Δ_{10} given in Equation 3.3 from Equation B.6:

$$\begin{aligned}
\Delta_{10} = \frac{P(\alpha_a|\alpha_x)}{P(\alpha_b|\alpha_x)} &= \theta^{(\alpha_x+10-\alpha_x)/80} \cdot \frac{(\theta^{\alpha_x/80} + \theta^{\alpha_x/80})}{(\theta^{\alpha_x/80} + \theta^{(\alpha_x+10)/80})} \\
&= \theta^{10/80} \cdot \frac{2 \cdot \theta^{\alpha_x/80}}{(\theta^{\alpha_x/80} + \theta^{\alpha_x/80} \cdot \theta^{10/80})} \\
&= \theta^{1/8} \cdot \frac{2 \cdot \theta^{\alpha_x/80}}{\theta^{\alpha_x/80}(1 + \theta^{1/8})} \\
&= \frac{2 \cdot \theta^{0.125}}{(1 + \theta^{0.125})}
\end{aligned}$$

Appendix C

Methods for Defining Inter-bout Tracks (Ch. 3)

In our data model, focal samples are time series composed of (x,y) locations at 2.5min intervals and feeding bouts are periods defined by beginning and ending times. Locations are assigned a binary state, either as occurring during a feeding bout or between feeding bouts, when the monkey is presumably moving to the next feeding site. In the simplest case, a location is tagged as "feeding" if its time-stamp falls between the beginning and ending times of a feeding bout, and tagged as "moving" if it does not. If a feeding bout lasts longer than 2.5min, it may contain more than one consecutive location which are all tagged as "feeding".

It is possible, however, that a feeding bout lasting less than 2.5 min does not overlap with any locations, but rather sits between two points in the time series without capturing either one. To mark both of these surrounding locations as "moving" would be misleading for two reasons. First, the feeding bout would be, for the purposes of our analysis, invisible, yielding an inter-bout track that continues uninterrupted through the observed feeding bout when, in fact, the feeding bout should break the track into two parts. Second, these locations actually represent the average of locations within a window of time beginning 1.25min before the location's time-stamp and extending to 1.25min later. Even if the time-stamp of the average location does not fall inside the period of the bout, some of the raw locations, which were collected at 20sec intervals, probably do.

In order to properly represent these "invisible" feeding bouts in the time series

of processed locations, we relax our conditions to include windows. More precisely, if a feeding bout's period does not capture any locations in the time series, then any locations for which the 2.5min window overlaps with the feeding bout period are tagged as "feeding". This can include a single window, or two consecutive windows, but never more than two (otherwise an actual location time-stamp would be captured, and the simpler case would apply)

Appendix D

Seed Dispersal Model (Ch. 4) Source Code

The following is the source code for the simulation model presented in Chapter 4, including the parent C++ script (model.cpp) and the supporting header files (*.hpp).

D.1 model.cpp

```
#include "io.hpp" // Brings in defs.hpp, lattice.hpp, random.hpp

using namespace std;
using namespace keittlab;

int main(int argc, char* argv[]) {
  //MODEL INITIALIZATION
  randomize();
  parameters par(argv[1]);
  lattice hex(par.R, par.Q, par.wrap);
  hex.fillAdj();
  hex.fillNYC();
  binomial_distribution<> ngaps(hex.N, par.amort);
  uniform_real_distribution<double> urd(0,1);
  state counts;
  landscape forest(hex.N); // Vector of empty cells/trees objects
  population foragers(par.A);
  vector<int> indeces;
  for (int i=0; i!=hex.N; ++i) indeces.push_back(i);

  ofstream treesfile(par.resdir+"/trees."+par.rid);
  // Place trees randomly
  InitLandRand(hex, forest, par, counts, urd, treesfile);
  InitPopRand(hex, foragers); // Place agents randomly
```

```
  if (par.naive==false) {
    inform(forest, foragers[0], par);
    for (auto it=foragers.begin()+1; it!=foragers.end(); ++it)
      it->memory = foragers[0].memory;
  }

  // SIMULATION
  for (; counts.day != par.days; ++counts.day) {
    int startFruit = totalFruit(forest, true);
    shuffle(foragers.begin(), foragers.end(), urng());
    for (int step = 0; step != par.steps; ++step) {
      // TIME STEP
      for (auto& A : foragers) takeStep(A, forest, hex, par, counts);
    }
    // END OF THE DAY
    int endFruit = totalFruit(forest, true);
    for (auto& A : foragers) predict(A, forest, par);
    if (par.sleep) {  // Void all seeds at sleeping site
      for (auto& A : foragers) seedflush(A, forest, counts);
    }
    matrixDispersal(forest, counts, hex, par,
                    float(startFruit-endFruit)/startFruit, indeces);
    phen(forest, hex, par, counts);
    grow(hex, forest, ngaps, urd, par, counts, treesfile);
    // Prune seedling bank
    if (counts.day % (365*10) == 1) prune(forest, par, counts);
  }
  // END SIMULATION

  // OUTPUT SNAPSHOT OF FINAL STATE
  // pushBanks(forest, par, true);
  pushMemory(foragers, par);
  pushFruit(forest, par);
  treesfile.close();

  return 0;
};
```

D.2 io.hpp

```
#include "defs.hpp"
#include <iostream>
#include <fstream>
#include <unordered_set>
#include <boost/algorithm/string.hpp>

parameters::parameters(string RID) {
  ifstream parfile("/home/colin/projects/SEED/init/params."+RID);
  string line;
  while (getline(parfile, line)) {
    int eqpos = line.find("=");
    string parname = line.substr(0, eqpos);
    string value = line.substr(eqpos+1);
    if (parname=="rid") {
      rid = value;
    } else if (parname=="years") {
        years = stoi(value);
    } else if (parname=="steps") {
      steps = stoi(value);
    } else if (parname=="version") {
      version = value.substr(1, value.size()-2);
    } else if (parname=="A") {
      A = stoi(value);
    } else if (parname=="cr") {
      cr = stoi(value);
    } else if (parname=="lag") {
      lag = stoi(value);
    } else if (parname=="nu") {
      nu = stod(value);
    } else if (parname=="sleep") {
      sleep = bool(stoi(value));
    } else if (parname=="naive") {
      naive = bool(stoi(value));
    } else if (parname=="pop") {
      pop = value.substr(1, value.size()-2);
    } else if (parname=="aspan") {
      aspan = stod(value);
    } else if (parname=="jspan") {
      jspan = stod(value);
```

```
    } else if (parname=="nrot") {
      nrot = stoi(value);
    } else if (parname=="gr") {
      gr = stod(value);
    } else if (parname=="U") {
      U = stod(value);
    } else if (parname=="C") {
      C = stod(value);
    } else if (parname=="Q") {
      Q = stoi(value);
    } else if (parname=="R") {
      R = stoi(value);
    } else if (parname=="L") {
      L = stod(value);
    } else if (parname=="p") {
      p = stod(value);
    } else if (parname=="kmax") {
      kmax = stoi(value);
    } else if (parname=="beta") {
      beta = stod(value);
    } else if (parname=="land") {
      land = value.substr(1, value.size()-2);
    } else if (parname=="wrap") {
      wrap = bool(stoi(value));
    } else if (parname=="resdir") {
      resdir = value.substr(1, value.size()-2);
    } else {
      cout << "Invalid Input Parameter" << endl;
      exit (EXIT_FAILURE);
    }
  }
  jdays = jspan*365;
  amort = 1 / aspan / 365;
  days = years*365;
  beta2 = (1-p) * beta / p;
}

std::ostream& operator<<(std::ostream& os, const parameters& PAR) {
  os << "rid=" << PAR.rid << endl;
  os << "years=" << PAR.years << endl;
  os << "steps=" << PAR.steps << endl;
```

```
  os << "version=" << PAR.version << endl;
  os << "A=" << PAR.A << endl;
  os << "cr=" << PAR.cr << endl;
  os << "lag=" << PAR.lag << endl;
  os << "nu=" << PAR.nu << endl;
  os << "sleep=" << PAR.sleep << endl;
  os << "naive=" << PAR.naive << endl;
  os << "pop=" << PAR.pop << endl;
  os << "aspan=" << PAR.aspan << endl;
  os << "jspan=" << PAR.jspan << endl;
  os << "nrot=" << PAR.nrot << endl;
  os << "gr=" << PAR.gr << endl;
  os << "U=" << PAR.U << endl;
  os << "C=" << PAR.C << endl;
  os << "Q=" << PAR.Q << endl;
  os << "R=" << PAR.R << endl;
  os << "L=" << PAR.L << endl;
  os << "p=" << PAR.p << endl;
  os << "kmax=" << PAR.kmax << endl;
  os << "beta=" << PAR.beta << endl;
  os << "land=" << PAR.land << endl;
  os << "wrap=" << PAR.wrap << endl;
  os << "resdir=" << PAR.resdir << endl;
  os << "jdays=" << PAR.jdays << endl;
  os << "amort=" << PAR.amort << endl;
  os << "days=" << PAR.days << endl;
  os << "beta2=" << PAR.beta2;
}

template <typename T>
std::ostream& operator<<(std::ostream& os, const std::vector<T>& v) {
    copy(v.begin(), v.end(), std::ostream_iterator<T>(os," "));
    return os;
}

template <typename T>
std::ostream& operator<<(std::ostream& os, const std::deque<T>& d) {
    copy(d.begin(), d.end(), std::ostream_iterator<T>(os," "));
    return os;
}
```

```
template <typename T>
std::ostream& operator<<(std::ostream& os,
                         const std::unordered_set<T>& us) {
    copy(us.begin(), us.end(), std::ostream_iterator<T>(os," "));
    return os;
}

template <typename T>
std::ostream& operator<<(std::ostream& os,
                         const std::unordered_map<int, T>& um) {
    for (auto& i : um) os << i.first<<":"<< i.second <<" ";
    return os;
}

template <typename T>
std::ostream& operator<<(std::ostream& os, const std::array<T, 2> a) {
  os << a[0] << " " << a[1];
  return os;
}

template <typename T>
std::ostream& operator<<(std::ostream& os, const std::array<T, 3> a) {
  os << a[0] << " " << a[1] << " " << a[2];
  return os;
}

void pushBanks(landscape& L, parameters& PAR, bool SP) {
  ofstream file(PAR.resdir+"/banks."+PAR.rid);
  for (auto& C : L) {
    unordered_map<int,int> pool;
    for (auto& S : C.bank)
      if (S.species==SP) ++pool[S.parent];
    for (auto it=pool.begin(); it!=pool.end(); ++it)
      file << C.snake <<" "<< it->first <<" "<< it->second << endl;
  }
  file.close();
}

void pushMemory(population& P, parameters& PAR) {
  ofstream file(PAR.resdir+"/memory."+PAR.rid);
  for (auto& A : P)
```

```
    file << A.memory << endl;
  file.close();
}

void pushFruit(landscape& L, parameters& PAR) {
  ofstream file(PAR.resdir+"/fruit."+PAR.rid);
  for (auto& C : L) {
    file << C.snake <<" "<< C.tree.id <<" "<< C.tree.species <<" ";
    file << C.tree.size<<" "<< C.tree.fruit << endl;
  }
  file.close();
}
```

D.3 defs.hpp

```
#include "lattice.hpp"
#include <cmath>
#include <deque>
#include <random>
#include <boost/math/special_functions/beta.hpp> // Boost 1.58

using namespace std;
using namespace keittlab;

int discretePowerLaw(uniform_real_distribution<double>& URD, int MIN,
                     int MAX, double EXP, double C) {
  int discrete = MAX+1;
  while ( discrete > MAX ) {
    double r = URD(urng());
    double continuous = (MIN-0.5)*pow(1-r, -1.0/(EXP-1)) + 0.5;
    discrete = round(C*continuous);
  }
  return discrete;
}

struct state {
  int tree_id, Nf, day;
  state() : tree_id(0), day(0), Nf(0) {}
};
```

```
struct parameters {
  int A, years, lag, steps, days, nrot, cr, kmax, Q, R;
  double aspan, jspan, jdays, amort, nu, U, gr, C, beta, beta2, p, L;
  bool sleep, naive, wrap;
  string resdir, rid, land, pop, version;
  parameters(string); // Constructor: defined in io.hpp
};

// LANDSCAPE
int fruitFlux(int size, int nrot) {
  return size / nrot + maybe(float(size%nrot)/nrot);
}

struct plant {
  int id; // ID is only assigned when plant reaches adult tree status
  int parent; // ID of parent plant
  bool species; // Focal species is TRUE, matrix species is FALSE
  int size; // How large is this tree?
  int fruit; // How many fruit does the tree have?
  int est; // On what day was this plant established?
  int emrg; // On what day did the plant emerge into the canopy?
  bool disp; // Was this plant dispersed by an agent?
  // To create empty trees when landscape is created
  plant()
      : id(-1), parent(-1), species(false), size(-1), fruit(0),
        est(-1), emrg(-1), disp(false) {}
  // Create new seed during simulation
  plant(int parent, bool species, int est, bool disp)
      : id(-1), species(species), parent(parent), size(-1), fruit(0),
        est(est), emrg(-1), disp(disp) {}
  void emerge(uniform_real_distribution<double>& URD, parameters& PAR,
              state& S) {
    id = S.tree_id++;
    size = discretePowerLaw(URD, PAR.nrot, PAR.kmax, PAR.L, PAR.C);
    fruit = fruitFlux(size, PAR.nrot);
    emrg = S.day;
  }
};

struct cell {
  int snake = -1; // Position of the cell in the landscape vector
```

```
  plant tree; // Each cell has one and only one tree
  deque<plant> bank; // Seed or seedling bank
};

void pushTree(ostream& os, cell& C) {
        os << C.snake <<" "<< C.tree.id <<" "<< C.tree.species <<" ";
        os << C.tree.parent <<" "<< C.tree.emrg << endl;
}

typedef vector<cell> landscape;

// Seed will just fall beneath the crown
// Unless there is secondary dispersal
// Or it is destroyed
void drop(int orig, landscape& L, lattice& H, parameters& PAR,
          state& S) {
  int dest = orig;
  if (maybe(PAR.gr)) dest = *one_of(H.adjlist.find(orig)->second);
  if (maybe(PAR.U)) {
    L[dest].bank.emplace_back(plant(L[orig].tree.id,
                              L[orig].tree.species, S.day, false));
  }
  --L[orig].tree.fruit; // Remove fruit from tree
}

void rot(cell& C, landscape& L, lattice& H, parameters& PAR,
         state& S) {
  if (C.tree.fruit>C.tree.size) {
    //For each rotten fruit, drop it to the ground
    int rotten = C.tree.fruit-C.tree.size;
    for (int i=0; i!=rotten; ++i)
      drop(C.snake, L, H, PAR, S);
  }
}

int totalFruit(landscape& L, bool species) {
  int count = 0;
  for (auto& C : L)
    if (C.tree.species==species) count += C.tree.fruit;
  return count;
}
```

```
// Disperse matrix seeds in equivalence to
// agent-based focal dispersal for the day
void matrixDispersal(landscape& L, state& S, lattice& H,
                     parameters& PAR, double dispeff,
                     vector<int>& indeces) {
  // Shuffle cell order
  shuffle(indeces.begin(), indeces.end(), urng());
  // Calculate matrix fruit to disperse
  int left = round(dispeff * totalFruit(L, false));
  auto it = indeces.begin();
  int dest = -1;
  while (left > 0) {
    if (L[*it].tree.species == false) {
      if (left >= L[*it].tree.fruit) {
        for (int i=0; i!=L[*it].tree.fruit; ++i) {
          dest = H.rwalk(*it, PAR.lag);
          L[dest].bank.emplace_back(plant(L[*it].tree.id,
                                          L[*it].tree.species,
                                          S.day, true));
        }
        left -= L[*it].tree.fruit;
        L[*it].tree.fruit = 0;
      } else {
        left -= L[*it].tree.fruit;
      }
    }
    ++it;
  }
}

void phen(landscape& L, lattice& H, parameters& PAR, state& S) {
  for (auto& C : L) {
    if (C.tree.species) {
      C.tree.fruit += fruitFlux(C.tree.size, PAR.nrot);
      rot(C, L, H, PAR, S); // Rotten fruit fall from tree
    } else {
      C.tree.fruit += fruitFlux(C.tree.size, PAR.nrot);
      rot(C, L, H, PAR, S); // Rotten fruit fall from tree
    }
  }
```

```
}

void grow(lattice& H, landscape& L, binomial_distribution<>& BD,
          uniform_real_distribution<double> URD, parameters& PAR,
          state& S, ostream& F) {
  auto dead = [&](plant& p){return p.est <= (S.day - PAR.jdays);};
  int ngaps = BD(urng());
  assert( ngaps >= 0 );
  double densDepend = 0.5;
  if(ngaps > 0) densDepend = boost::math::ibetac(PAR.beta, PAR.beta2,
                                                 S.Nf*1.0/H.N);
  for (int i=0; i < ngaps; ++i) { // How many gaps are created today?
    auto& C = *one_of(L); // Refer to a random cell
    // Find first dead seedling by searching from back to front
    auto firstDead = find_if(C.bank.rbegin(), C.bank.rend(), dead);
    // If seedling bank is empty (or has no live seeds)...
    if (firstDead == C.bank.rbegin()) {
      // Replace old tree with new matrix tree
      S.Nf -= C.tree.species;
      C.tree = plant(-2, false, -1, false);
      C.tree.emerge(URD, PAR, S);
    } else {
      int nseeds = distance(C.bank.rbegin(), firstDead);
      auto firstFalse = partition(C.bank.rbegin(), firstDead,
                                  [](plant& p){return p.species;});
      int nTrue = distance(C.bank.rbegin(), firstFalse);
      double pTrue = nTrue*1.0/nseeds;
      double ppTrue = pTrue*densDepend /
                      (pTrue*densDepend + (1-densDepend)*(1-pTrue));
      deque<plant>::reverse_iterator winner;
      if (maybe(ppTrue)) {
        winner = one_of_range(C.bank.rbegin(), firstFalse);
      } else {
        winner = one_of_range(firstFalse, firstDead);
      }
      // Update focal tree count
      S.Nf += (winner->species - C.tree.species);
      C.tree = *winner; // Replace old tree with new one
      C.tree.emerge(URD, PAR, S);
      C.bank.clear(); // Clear seedling bank
    }
```

```
    pushTree(F, C);
  }
}

// Remove dead seeds/seedlings from bank to free up memory
void prune(landscape& L, parameters& PAR, state& S) {
  // True when seedling is alive
  auto alive = [&](plant& p){return p.est > (S.day - PAR.jdays);};
  // For each cell on the landscape, find the first live seedling
  //And erase all seedlings up to, but not including, first living one
  for (auto& C : L) {
    auto firstLiving = find_if(C.bank.begin(), C.bank.end(), alive);
    C.bank.erase(C.bank.begin(), firstLiving);
  }
}

// MOVEMENT
struct agent {
  int id = -1;
  int snake = -1;
  deque<pair<int,int>> cache;
  unordered_map<int, int> memory;
};

typedef vector<agent> population;

void predict(agent& A, landscape& L, parameters& PAR) {
  for (auto& loc : A.memory) {
    loc.second += fruitFlux(L[loc.first].tree.size, PAR.nrot);
    if (loc.second>L[loc.first].tree.size) {
      loc.second=L[loc.first].tree.size;
    }
  }
}

double getValue(double nfruit, double distance) {
  return nfruit / distance;
}

void eat(agent& A, cell& C, parameters& PAR) {
  // If the agent's appetite is greater than the fruit on the tree...
```

```
  // Swallow seed
  if (PAR.cr >= C.tree.fruit) {
    for (int i=0; i!=C.tree.fruit; ++i)
      A.cache.emplace_back(make_pair(PAR.lag+1, C.tree.id));
    C.tree.fruit = 0; // All the fruit has been eaten
  } else {
    for (int i=0; i!= PAR.cr; ++i) {
      --C.tree.fruit; // Remove fruit from tree
      A.cache.emplace_back(make_pair(PAR.lag+1, C.tree.id));
    }
  }
}

int best(agent& A, lattice& H, parameters& PAR) {
  vector<int> finalists;
  int maximum = -1;
  int value = -1;
  for (auto& j : A.memory) {
    value = getValue(j.second, H.getNYC(A.snake, j.first));
    if (value > maximum) { // Is that value the highest one so far?
      maximum = value; // If so, reset the maximum
      finalists.clear();
      finalists.push_back(j.first);
    } else if (value == maximum) {
      finalists.push_back(j.first);
    }
  }
  if(finalists.empty()) {
    return -1;
  } else {
    return *one_of(finalists);
  }
}

int move(agent& A, lattice& H, parameters& PAR) {
  int ideal = best(A, H, PAR);
  double p;
  if (ideal == -1) {
    p = 1;
  } else {
    double e = getValue(A.memory[ideal], H.getNYC(A.snake, ideal));
```

```
    p = exp(-1*e / PAR.nu);
  }
  if (maybe(p)) { // Agents takes random step
    A.snake = *one_of(H.adjlist.find(A.snake)->second);
    return -1;
  } else {
    A.snake = H.direction(A.snake, ideal);
    return ideal;
  }
}

int takeStep(agent& A, landscape& L, lattice& H, parameters& PAR,
             state& S) {
  cell& C = L[A.snake];
  //DIGEST
  for (auto& s : A.cache) --s.first; // decrement cache (1 -> 0)
  //DEPOSIT
  auto notready = [](pair<int, int>& s){return s.first>0;}; //
  auto firstnotready = find_if(A.cache.begin(), A.cache.end(),
                               notready);
  for (auto it=A.cache.begin(); it!=firstnotready; ++it)
    C.bank.emplace_back(plant(it->second, true, S.day, true));
  A.cache.erase(A.cache.begin(), firstnotready); //
  // DECIDE:
  if (C.tree.species) { // If the cell contains a focal tree...
    if (C.tree.fruit>0) { // And there is fruit...
      eat(A, C, PAR); // Eat the fruit, and stay
      return -2; // indicates eating
    } else { // But if the focal tree is empty...
      A.memory[A.snake] = 0; // Remember an empty focal tree...
      return move(A, H, PAR); // And then leave
    }
  } else {
    // But if the cell contains a matrix tree
    // And you remember a focal tree here
    if (A.memory.count(A.snake)>0) {
      A.memory.erase(A.snake); // Then forget that focal tree
    }
    return move(A, H, PAR); // Then leave either way
  }
}
```

```
void seedflush(agent& A, landscape& L, state& S) {
  if (!A.cache.empty()) {
    for (auto& s : A.cache)
      L[A.snake].bank.emplace_back(plant(s.second, true, S.day,
                                         true));
    A.cache.clear();
  }
}

// INITIALIZATION FUNCTIONS
void InitLandRand(lattice& H, landscape& L, parameters& PAR,
                  state& S, uniform_real_distribution<double>& URD,
                  ostream& F) {
  for (int i=0; i!=H.N; ++i) { // Loop through cells
    L[i].snake=i; // Cell ID = vector position
    L[i].tree.id = S.tree_id++; // Tree IDs starting from 0
    L[i].tree.species = maybe(PAR.p); // Set species with Bern. trial
    L[i].tree.size = discretePowerLaw(URD, PAR.nrot, PAR.kmax, PAR.L,
                                      PAR.C);
    L[i].tree.fruit = fruitFlux(L[i].tree.size, PAR.nrot);
    S.Nf += L[i].tree.species;
    pushTree(F, L[i]);
  }
}

void InitPopRand(lattice& H, population& P) {
  int id_counter = 0;
  for (auto& A : P) { // Loop through population
    A.id = id_counter++;
    A.snake = pick_a_number(0, H.N-1);
  }
}

void inform(landscape& L, agent& A, parameters& PAR) {
  for (auto& C : L)
    if(C.tree.species) A.memory[C.snake]=C.tree.fruit;
}
```

D.4 lattice.hpp

```
#include <vector>
#include <string>
#include <unordered_map>
#include <array>
#include <cmath>
#include <algorithm>
#include "random.hpp"
#define PI 3.14159265

using namespace std;
using namespace keittlab;

// Complete Hexagonal Lattice covering a rectangular area
struct lattice {
  int ncol;
  int nrow;
  int N;
  double size;
  array<double,2> offset;
  string convention;
  string rotation;
  double height;
  double width;
  double vertical;
  double horiz;
  unordered_map< int, vector<int> > adjlist;
  unordered_map< int, unordered_map<int,int> > NYC;
  lattice(int NCOL, int NROW, double SIZE=1/sqrt(3),
          string CONVENTION="odd-r", string ROTATION="pointy-top") {
    ncol = NCOL;
    nrow = NROW;
    N = NCOL * NROW;
    size = SIZE;
    offset[0] = sqrt(3)/2*SIZE;
    offset[1] = SIZE;
    convention = CONVENTION;
    rotation = ROTATION;
    height = SIZE * 2;
    width = sqrt(3) / 2 * height;
```

```
  vertical = 0.75 * height;
  horiz = width;
}
bool outside(array<int,2> qr) {
  return qr[0]<0 || qr[0]>=ncol || qr[1]<0 || qr[1]>=nrow;
}
array<int,2> qr(int snake, int dir=0) {
  int q = snake % ncol;
  int r = snake / ncol;
  array<int,2> out;
  if (dir==0) {
    out[0]=q; out[1]=r;
  } else if (dir==1) {
    out[0]=q;  out[1]=r+nrow;
  } else if (dir==2) {
    out[0]=q+ncol; out[1]=r+nrow;
  } else if (dir==3) {
    out[0]=q+ncol; out[1]=r;
  } else if (dir==4) {
    out[0]=q+ncol; out[1]=r-nrow;
  } else if (dir==5) {
    out[0]=q; out[1]=r-nrow;
  } else if (dir==6) {
    out[0]=q-ncol; out[1]=r-nrow;
  } else if (dir==7) {
    out[0]=q-ncol; out[1]=r;
  } else if (dir==8) {
    out[0]=q-ncol; out[1]=r+nrow;
  } else {
    out[0]=q; out[1]=r;
  }
  return out;
}
array<int,3> cube(array<int,2> qr) {
  int q = qr[0]; int r = qr[1];
  bool odd = abs(r % 2);
  int x = q - (r - odd) / 2;
  int z = r;
  int y = -x - z;
  array<int,3> output {{ x , y , z }};
  return output;
```

```
}
array<int,2> qr(array<int,3> cube) {
  int odd = abs(cube[2] % 2);
  int q = cube[0] + (cube[2] - odd) / 2;
  int r = cube[2];
  array<int,2> output {{ q, r }};
  return output;
}
int snake(array<int,2> qr) {
  if (outside(qr)) {
    return -1;
  } else {
    return qr[1] * ncol + qr[0];
  }
}
int cubicDist(array<int,3> orig, array<int,3> dest) {
  int dx = abs(orig[0] - dest[0]);
  int dy = abs(orig[1] - dest[1]);
  int dz = abs(orig[2] - dest[2]);
  return (dx+dy+dz) / 2;
}
int manhattan(int orig, int dest) {
  auto origCube = cube(qr(orig));
  if (wrap) {
    vector<int> distances;
    for (int i=0; i!=9; ++i) {
      distances.push_back(cubicDist(origCube, cube(qr(dest, i))));
    }
    return *min_element(distances.begin(), distances.end());
  }
  else return cubicDist(origCube, cube(qr(dest)));
}
array<int,3> project(array<int,3> orig, int dir, int dist) {
  int x = orig[0]; int y = orig[1]; int z = orig[2];
  if (dir == 0) {
    z+=dist; y-=dist;
  } else if (dir == 1) {
    x+=dist;y-=dist;
  } else if (dir == 2) {
    x+=dist;z-=dist;
  } else if (dir == 3) {
```

```
    y+=dist;z-=dist;
  } else if (dir == 4) {
    x-=dist;y+=dist;
  } else if (dir == 5) {
    x-=dist;z+=dist;
  }
  array<int,3> output {{ x, y, z }};
  return output;
}
void fillNYC() {
  for (int i=0; i!=N; ++i) {
    for (int j=i; j!=N; ++j)
      NYC[i][j] = manhattan(i, j);
  }
}
int getNYC(int orig, int dest) {
  if (orig<=dest) {
    return NYC[orig][dest];
  } else {
    return NYC[dest][orig];
  }
}
vector<int> ring(int pos_snake, int R) {
  vector<int> output;
  auto pos_cube = project(cube(qr(pos_snake)), 4, R);
  for (int i=0; i!=6; i++) {
    for (int j=0; j!=R; ++j) {
      pos_cube = project(pos_cube, i, 1);
      pos_snake = snake(qr(pos_cube));
      if (pos_snake != -1) output.push_back(pos_snake);
    }
  }
  return output;
}
void fillAdj() {
  for (int i=0; i!= N; ++i)
    adjlist.emplace(i, ring(i, 1));
}
int direction(int origin, int dest) { //
  auto dist = getNYC(origin, dest);
  auto adj = adjlist[origin];
```

```
    vector<int> choices;
    for (auto& i : adj)
      if (i >= 0 && getNYC(i, dest) < dist) choices.push_back(i);
    return choices[0];
  }
  int rwalk(int snake, int nsteps) {
    for (int i=0; i!= nsteps; ++i)
      snake = *one_of( adjlist.find(snake)->second );
    return snake;
  }
};
```

D.5 random.hpp

```
#ifndef KEITTLAB_RANDOM_HPP
#define KEITTLAB_RANDOM_HPP

#include <cassert>
#include <cstddef>
#include <initializer_list>
#include <iterator>
#include <limits>
#include <random>

namespace keittlab {

inline std::default_random_engine& urng()
{
  static std::default_random_engine engine;
  return engine;
}

inline std::random_device::result_type randomize()
{
  static std::random_device device;
  auto seed = device();
  urng().seed(seed);
  return seed;
}
```

```
inline int pick_a_number(int from, int thru)
{
  static std::uniform_int_distribution<> z;
  return z(urng(), decltype(z)::param_type{from, thru});
}

inline double pick_a_number(double from, double upto)
{
  static std::uniform_real_distribution<> z;
  return z(urng(), decltype(z)::param_type{from, upto});
}

template<typename T>
inline T& one_of(T& x, T& y, double px = 0.5)
{
  static std::bernoulli_distribution z;
  return z(urng(), typename decltype(z)::param_type{px}) ? x : y;
}

template<typename T>
inline T const& one_of(T const& x, T const& y, double px = 0.5)
{
  static std::bernoulli_distribution z;
  return z(urng(), typename decltype(z)::param_type{px}) ? x : y;
}

inline bool maybe(double p = 0.5)
{ return one_of(true, false, p); }

template<typename Forward>
inline Forward one_of_range(Forward begin, Forward end)
{
  if (begin == end) return end;
  return std::next(begin,
                   pick_a_number(0, std::distance(begin, end) - 1));
}

template<typename Container>
inline auto one_of(Container& c) -> decltype(begin(c))
{ return one_of_range(begin(c), end(c)); }
```

```
template<typename Container>
inline auto one_of(Container const& c) -> decltype(begin(c))
{ return one_of_range(begin(c), end(c)); }

template<typename T, std::size_t N>
inline T& one_of(T (&v)[N])
{ return *one_of_range(std::begin(v), std::end(v)); }

template<typename T>
inline T one_of(std::initializer_list<T> z)
{ return *one_of_range(begin(z), end(z)); }

} // namespace keittlab

#endif // KEITTLAB_RANDOM_HPP
```

Bibliography

Andresen, E. 1999. Seed Dispersal by Monkeys and the Fate of Dispersed Seeds in a Peruvian Rain Forest. Biotropica 31:145–158.

Asensio, N., W. Y. Brockelman, S. Malaivijitnond, and U. H. Reichard. 2011. Gibbon travel paths are goal oriented. Animal Cognition 14:395–405. doi: 10.1007/s10071-010-0374-1.

Augspurger, C. K. 1984. Seedling Survival of Tropical Tree Species: Interactions of Dispersal Distance, Light-Gaps, and Pathogens. Ecology 65:1705–1712. doi: 10.2307/1937766.

Bacles, C. F. E., A. J. Lowe, and R. A. Ennos. 2006. Effective Seed Dispersal Across a Fragmented Landscape. Science (New York, N.Y.) 311:628. doi: 10.1126/science.1121543.

Bartumeus, F., M. G. E. Da Luz, G. M. Viswanathan, and J. Catalan. 2005. Animal Search Strategies: A Quantitative Random-Walk Analysis. Ecology 86:3078–3087.

Bennett, A. T. 1996. Do animals have cognitive maps? The Journal of Experimental Biology 199:219–224. doi:10.1.1.318.4634.

Blake, J. G., J. Guerra, D. Mosquera, R. Torres, B. A. Loiselle, and D. Romo. 2010. Use of Mineral Licks by White-Bellied Spider Monkeys (Ateles belzebuth) and Red Howler Monkeys (Alouatta seniculus) in Eastern Ecuador. International Journal of Primatology 31:471–483. doi:10.1007/s10764-010-9407-5.

Boyer, D., and P. D. Walsh. 2010. Modelling the mobility of living organisms in heterogeneous landscapes: does memory improve foraging success? Philosophical Transactions of the Royal Society A 368:5645–59.

Boyer, D., G. Ramos-Fernández, O. Miramontes, J. L. Mateos, G. Cocho, H. Larralde, H. Ramos, and F. Rojas. 2006. Scale-free foraging by primates emerges from their interaction with a complex environment. Proceedings. Biological sciences / The Royal Society 273:1743–50. doi:10.1098/rspb.2005.3462.

Brosi, B. J., P. R. Armsworth, and G. C. Daily. 2008. Optimal design of agricultural landscapes for pollination services. Conservation Letters 1:27–36. doi:10.1111/j.1755-263X.2008.00004.x.

Chapman, C. 2013. Patch Use and Patch Depletion by the Spider and Howling Monkeys of Santa Rosa National Park , Costa Rica. Behaviour 105:99–116.

Charnov, E. 1976. Optimal foraging, the marginal value theorem. Theoretical population biology 9:129136.

Chin, G. Y. 1987. Two-Dimensional Mesaure of Accuracy in Navigational Systems. Technical report, U.S. Department of Transportation, Cambridge, MA.

Clark, J. S., M. Silman, R. Kern, E. Macklin, and J. HilleRisLambers. 1999. Seed dispersal near and far: patterns across temperate and tropical forests. Ecology 80:1475–1494.

Clauset, A., C. R. Shalizi, and M. E. J. Newman. 2009. Power-Law Distributions in Empirical Data. SIAM Review 51:661–703. doi:10.1137/070710111.

Codling, E. A., M. J. Plank, and S. Benhamou. 2008. Random walk models in biology. Journal of The Royal Society Interface 5:813–834. doi:10.1098/rsif.2008.0014.

Condit, R. S., S. P. Hubbell, and R. B. Foster. 1995. Mortality Rates of 205 Neotropical Tree and Shrub Species and the Impact of a Severe Drought. Ecological Monographs 65:419439.

Connell, J. H. 1971. On the role of natural enemies in preventing competitive exclusion in some marine animals and in rain forest trees. In P. J. den Boer, and G. R. Gradwell, eds., Dynamiics of Populations: Proceedings of the Advanced Study Institute on Dynamics of numbers in populations, Oosterbeek, the Netherlands, 7-18 September 1970, pages 298–310. Centre for Agricultural Publishing and Documentation, Wageningen.

Côrtes, M. C., and M. Uriarte. 2012. Integrating frugivory and animal movement: a review of the evidence and implications for scaling seed dispersal. Biological Reviews doi:10.1111/j.1469-185X.2012.00250.x.

Deguines, N., C. Jono, M. Baude, M. Henry, R. Julliard, and C. Fontaine. 2014. Large-scale trade-off between agricultural intensification and crop pollination services. Frontiers in Ecology and the Environment 12:212–217. doi:10.1890/130054.

Denslow, J. S. 1987. Tropical Rainforest Gaps and Tree Species Diversity. Annual Review of Ecology and Systematics 18:431–451.

Di Fiore, A., and S. A. Suarez. 2007. Route-based travel and shared routes in sympatric spider and woolly monkeys: cognitive and evolutionary implications. Animal Cognition 10:317–29. doi:10.1007/s10071-006-0067-y.

Dixon, P. M. 2002. Ripleys K function. In A. H. El-Shaarawi, and W. W. Piegorsch, eds., Encyclopedia of Environmetrics, volume 3, pages 1796–1803. John Wiley & Sons, Chichester. doi:10.1002/9780470057339.var046.

Elzinga, J. A., A. Atlan, A. Biere, L. Gigord, A. E. Weis, and G. Bernasconi. 2007. Time after time: flowering phenology and biotic interactions. Trends in Ecology & Evolution 22:432–9. doi:10.1016/j.tree.2007.05.006.

Fagan, W. F., M. A. Lewis, M. Auger-Methe, T. Avgar, S. Benhamou, G. Breed, L. LaDage, U. E. Schlagel, W.-w. Tang, Y. P. Papastamatiou, J. Forester, and T. Mueller. 2013. Spatial memory and animal movement. Ecology Letters 16:1316–1329. doi:10.1111/ele.12165.

Fragoso, J. M. V. 1997. Tapir-Generated Seed Shadows: Scale-Dependent Patchiness in the Amazon Rain Forest. Journal of Ecology 85:519–529. doi: 10.2307/2960574.

Fryxell, J. M., and T. Avgar. 2012. Catching the wave. Nature 490:182–183.

Garber, P. A., and B. Hannon. 1993. Modeling monkeys: A comparison of computer-generated and naturally occurring foraging patterns in two species of neotropical primates. International Journal of Primatology 14:827–852. doi: 10.1007/BF02220255.

Hegland, S. J., A. Nielsen, A. Lazaro, A.-L. Bjerknes, and O. Totland. 2009. How does climate warming affect plant-pollinator interactions ? Ecology Letters 12:184–195. doi:10.1111/j.1461-0248.2008.01269.x.

Holt, R. D. 2008. Theoretical perspectives on resource pulses. Ecology 89:671–681. doi:10.1890/07-0348.1.

Holzschuh, A., C. F. Dormann, T. Tscharntke, and I. Steffan-Dewenter. 2013. Mass-flowering crops enhance wild bee abundance. Oecologia 172:477–484. doi: 10.1007/s00442-012-2515-5.

Hopkins, M. E. 2011. Mantled Howler (Alouatta palliata) Arboreal Pathway Networks: Relative Impacts of Resource Availability and Forest Structure. International Journal of Primatology 32:238–258. doi:10.1007/s10764-010-9464-9.

———. 2016. Mantled howler monkey spatial foraging decisions reflect spatial and temporal knowledge of resource distributions. Animal Cognition 19:1–17. doi:10.1007/s10071-015-0941-6.

Howe, H. F., and M. N. Miriti. 2000. No question: seed dispersal matters. Trends in Ecology & Evolution 15:434–436.

———. 2004. When Seed Dispersal Matters. BioScience 54:651. doi:10.1641/0006-3568(2004)054[0651:WSDM]2.0.CO;2.

Inouye, D. W. 2008. Effects of Climate Change on Phenology, Frost Damage, and Floral Abundance of Montage Wildflowers. Ecology 89:353–362.

Janzen, D. H. 1970. Herbivores and the Number of Tree Species in Tropical Forests. The American Naturalist 104:501–528.

Jauker, F., F. Peter, V. Wolters, and T. Diekötter. 2012. Early reproductive benefits of mass-flowering crops to the solitary bee Osmia rufa outbalance post-flowering disadvantages. Basic and Applied Ecology 13:268–276. doi:10.1016/j.baae.2012.03.010.

Jordano, P., and J. A. Godoy. 2002. Frugivore-generated Seed Shadows: a Landscape View of Demographic and Genetic Effects. In D. J. Levey, W. Silva,

and M. Galetti, eds., Seed Dispersal and Frugivory: Ecology, Evolution and Conservation, volume 20, pages 305–321. CAB International.

Jordano, P., and E. W. Schupp. 2000. Seed Disperser Effectiveness: the Quantity Component and Patterns of Seed Rain for Prunus mahaleb. Ecological Monographs 70:591–615.

Julliot, C. 1997. Impact of seed dispersal by red howler monkeys Alouatta seniculus on the seedling population in the understorey of tropical rain forest. Journal of Ecology 85:431–440.

Karubian, J., J. Fabara, D. Yunes, J. P. Jorgenson, D. Romo, and T. B. Smith. 2005. Temporal and Spatial Patterns of Macaw Abundance in the Ecuatorian Amazon. The Condor 107:617–626. doi:10.1650/0010-5422(2005)107[0617:TASPOM]2.0.CO;2.

Karubian, J., V. L. Sork, T. Roorda, R. Duraes, and T. B. Smith. 2010. Destination-based seed dispersal homogenizes genetic structure of a tropical palm. Molecular Ecology 19:1745–1753. doi:10.1111/j.1365-294X.2010.04600.x.

Karubian, J., R. Duraes, J. L. Storey, and T. B. Smith. 2012. Mating Behavior Drives Seed Dispersal by the Long-wattled Umbrellabird Cephalopterus penduliger. Biotropica 44:689–698.

Karubian, J., K. Ottewell, A. Link, and A. Di Fiore. 2015. Genetic consequences of seed dispersal to sleeping trees by white-bellied spider monkeys. Acta Oecologica 68:50–58. doi:10.1016/j.actao.2015.07.005.

Keitt, T. H. 2000. Spectral representation of neutral landscapes. Landscape Ecology 15:479–493.

———. 2009. Habitat Conversion, Extinction Thresholds, and Pollination Services in Agroecosystems. Ecological Applications 19:1561–1573.

Kelly, D. 1994. The evolutionary ecology of mast seeding. Trends in Ecology & Evolution 9:465–470.

Khoury, D. S., M. R. Myerscough, and A. B. Barron. 2011. A Quantitative Model of Honey Bee Colony Population Dynamics. PLoS ONE 6:2–7. doi: 10.1371/journal.pone.0018491.

Khoury, D. S., A. B. Barron, and M. R. Myerscough. 2013. Modelling Food and Population Dynamics in Honey Bee Colonies. PLoS ONE 8. doi:10.1371/ journal.pone.0059084.

Kim, S. 2015. ppcor: Partial and Semi-Partial (Part) Correlation.

Klein, A.-M., B. E. Vaissière, J. H. Cane, I. Steffan-Dewenter, S. A. Cunningham, C. Kremen, and T. Tscharntke. 2007. Importance of pollinators in changing landscapes for world crops. Proceedings. Biological sciences / The Royal Society 274:95–96. doi:10.1098/rspb.2006.3721.

Kremen, C., N. M. Williams, R. L. Bugg, J. P. Fay, and R. W. Thorp. 2004. The area requirements of an ecosystem service: crop pollination by native bee communities in California. Ecology Letters 7:1109–1119. doi:10.1111/j.1461-0248. 2004.00662.x.

Lasky, J. R., and T. H. Keitt. 2013. Reserve Size and Fragmentation Alter Community Assembly , Diversity , and Dynamics. The American Naturalist 182. doi:10.1086/673205.

Link, A., and A. Di Fiore. 2006. Seed dispersal by spider monkeys and its importance in the maintenance of neotropical rain-forest diversity. Journal of Tropical Ecology 22:235. doi:10.1017/S0266467405003081.

McRae, B. H., and V. B. Shah. 2013. Circuitscape 4 User Guide.

McRae, B. H., B. G. Dickson, T. H. Keitt, and V. B. Shah. 2008. Using Circuit Theory to Model Connectivity in Ecology, Evolution and Conservation. Ecology 89:2712–2724.

Memmot, J., P. G. Craze, N. M. Waser, and M. V. Price. 2007. Global warming and the disruption of plantpollinator interactions. Ecology Letters 10:710–717. doi:10.1111/j.1461-0248.2007.01061.x.

Morales, J. M., D. T. Haydon, J. Frair, K. E. Holsinger, and J. M. Fryxell. 2004. Extracting More Out of Relocation Data: Building Movement Models As Mixtures of Random Walks. Ecology 85:2436–2445. doi:10.1890/03-0269.

Nathan, R., and H. C. Muller-Landau. 2000. Spatial patterns of seed dispersal, their determinants and consequences for recruitment. Trends in Ecology & Evolution 15:278–285. doi:10.1016/S0169-5347(00)01874-7.

Nathan, R., W. M. Getz, E. Revilla, M. Holyoak, R. Kadmon, D. Saltz, and P. E. Smouse. 2008. A movement ecology paradigm for unifying organismal movement research. Proceedings of the National Academy of Sciences of the United States of America 105:19052–9. doi:10.1073/pnas.0800375105.

Nonaka, E., and P. Holme. 2007. Agent-based model approach to optimal foraging in heterogeneous landscapes: effects of patch clumpiness. Ecography 30:777–788. doi:10.1111/j.2007.0906-7590.05148.x.

Normand, E., and C. Boesch. 2009. Sophisticated Euclidean maps in forest chimpanzees. Animal Behaviour 77:1195–1201. doi:10.1016/j.anbehav.2009.01.025.

Noser, R., and R. W. Byrne. 2007. Mental maps in chacma baboons (Papio ursinus): Using inter-group encounters as a natural experiment. Animal Cognition 10:331–340. doi:10.1007/s10071-006-0068-x.

Ogilvie, J. E., and J. R. K. Forrest. 2017. Interactions between bee foraging and floral resource phenology shape bee populations and communities. Current Opinion in Insect Science 21:75–82. doi:10.1016/j.cois.2017.05.015.

Ostfeld, R. S., and F. Keesing. 2000. Pulsed resources and community dynamics of consumers in terrestrial ecosystems. Trends in Ecology and Evolution 15:232–237. doi:10.1016/S0169-5347(00)01862-0.

Peres, C. A., and M. van Roosmalen. 2002. Primate Frugivory in Two Species-rich Neotropical Forests: Implications for the Demography of Large-seeded Plants in Overhunted Areas. In M. Galetti, D. J. Levey, and W. R. Silva, eds., Seed Dispersal and Frugivory: Ecology, Evolution and Conservation, chapter 27, pages 407–421. CAB International, New York.

Porter, L. M., and P. A. Garber. 2013. Foraging and Spatial Memory in Wild Weddell's Saddleback Tamarins (Saguinus fuscicollis weddelli) When Moving Between Distant and Out-of-Sight Goals. International Journal of Primatology 34:30–48. doi:10.1007/s10764-012-9644-x.

Poucet, B. 1993. Spatial cognitive maps in animals: New hypotheses on their structure and neural mechanisms. Psychological Review 100:163–182. doi:10.1037/0033-295X.100.2.163.

Presotto, A., and P. Izar. 2010. Spatial reference of black capuchin monkeys in Brazilian Atlantic Forest: Egocentric or allocentric? Animal Behaviour 80:125–132. doi:10.1016/j.anbehav.2010.04.009.

Pyke, G., H. Pulliam, and E. Charnov. 1977. Optimal Foraging : A Selective Review of Theory and Tests. Quarterly Review of Biology 52:137–154.

R Core Team. 2015. R: A Language and Environment for Statistical Computing.

Rathcke, B., and E. P. Lacey. 1985. Phenological Patterns of Terrestrial Plants. Annual Review of Ecology and Systematics 16:179–214. doi:10.1146/annurev.es.16.110185.001143.

Riba-Hernández, P., K. E. Stoner, and P. W. Lucas. 2003. The sugar composition of fruits in the diet of spider monkeys (Ateles geoffroyi) in tropical humid forest in Costa Rica. Journal of Tropical Ecology 19:709–716. doi:10.1017/S0266467403006102.

Ricketts, T. H., J. Regetz, I. Steffan-Dewenter, S. A. Cunningham, C. Kremen, A. Bogdanski, B. Gemmill-Herren, S. Greenleaf, A. M. Klein, M. M. Mayfield, L. A. Morandin, A. Ochieng, and B. F. Viena. 2008. Landscape effects on crop pollination services: are there general patterns? Ecology Letters 11:499–515. doi:10.1111/j.1461-0248.2008.01157.x.

Riedinger, V., M. Renner, M. Rundlöf, I. Steffan-Dewenter, and A. Holzschuh. 2014. Early mass-flowering crops mitigate pollinator dilution in late-flowering crops. Landscape Ecology 29:425–435. doi:10.1007/s10980-013-9973-y.

Rodrigo, T. 2002. Navigational strategies and models. Psicológica 23:3–23.

Rundlöf, M., A. S. Persson, H. G. Smith, and R. Bommarco. 2014. Late-season mass-flowering red clover increases bumble bee queen and male densities. Biological Conservation 172:138–145. doi:10.1016/j.biocon.2014.02.027.

Russell, S., A. B. Barron, and D. Harris. 2013. Dynamic modelling of honey bee (Apis mellifera) colony growth and failure. Ecological Modelling 265:158–169. doi:10.1016/j.ecolmodel.2013.06.005.

Russo, S. E., and C. K. Augspurger. 2004. Aggregated seed dispersal by spider monkeys limits recruitment to clumped patterns in Virola calophylla. Ecology Letters 7:1058–1067. doi:10.1111/j.1461-0248.2004.00668.x.

Schaik, C. P. v. 1993. The Phenology of Tropical Forests: Adaptive Significance and Consequences for Primary Consumers. Annual Review of Ecology and Systematics 24:353–377. doi:10.1146/annurev.ecolsys.24.1.353.

Schick, R. S., S. R. Loarie, F. Colchero, B. D. Best, A. Boustany, D. A. Conde, P. N. Halpin, L. N. Joppa, C. M. McClellan, and J. S. Clark. 2008. Understanding movement data and movement processes: current and emerging directions. Ecology letters 11:1338–50. doi:10.1111/j.1461-0248.2008.01249.x.

Schreier, A. L., and M. Grove. 2010. Ranging patterns of hamadryas baboons: Random walk analyses. Animal Behaviour 80:75–87. doi:10.1016/j.anbehav.2010.04.002.

Shah, V. B., and B. H. McRae. 2008. Circuitscape: A Tool for Landscape Ecology. Proceedings of the 7th Python in Science Conference (SciPy 2008) pages 62–66. doi:10.1111/j.1523-1739.2008.00942.x.

Suarez, S. A. 2014. Ecological Factors Predictive of Wild Spider Monkey (Ateles belzebuth) Foraging Decisions in Yasuní, Ecuador. American Journal of Primatology 76:1185–1195. doi:10.1002/ajp.22303.

Suarez, S. A., J. Karro, J. Kiper, D. Farler, B. McElroy, B. C. Rogers, B. Stockwell, and T. Young. 2014. A Comparison of ComputerGenerated and Naturally Occurring Foraging Patterns in RouteNetworkConstrained Spider Monkeys. American Journal of Primatology 76:460–471. doi:10.1002/ajp.22222.

Symington, M. M. 1990. Fission-Fusion Social Organization in Ateles and Pan. International Journal of Primatology 11:47–61. doi:10.1007/BF02193695.

Terborgh, J. 1990. Seed and fruit dispersal - commentary. In K. S. Bawa, and M. Handley, eds., Reproductive Ecology of Tropical Forest Plants, chapter Seed and f, pages 181–190. Parthenon Publishing Group, Paris.

Thomson, J. D., M. Slatkin, and B. A. Thomson. 1997. Trapline foraging by bumble bees: II. Definition and detection from sequence data. Behavioral Ecology 8:199–210.

Tolman, E. C. 1948. Cognitive maps in rats and men. Psychological Review 55:189–208. doi:10.1037/h0061626.

Toussaint, G. T. 1980. The relative neighborhood graph of a finite planar set. Pattern Recognition 12:261–268.

Valencia, R., R. B. Foster, G. Villa, R. Condit, J.-C. Svenning, C. Hernández, K. Romoleroux, E. Losos, E. Magård, and H. Balslev. 2004. Tree species distributions and local habitat variation in the Amazon: Large forest plot in

eastern Ecuador. Journal of Ecology 92:214–229. doi:10.1111/j.0022-0477.2004.00876.x.

Wang, B. C., and T. B. Smith. 2002. Closing the seed dispersal loop. Trends in Ecology and Evolution 17:379–385. doi:10.1016/S0169-5347(02)02541-7.

Westphal, C., I. Steffan-Dewenter, and T. Tscharntke. 2009. Mass flowering oilseed rape improves early colony growth but not sexual reproduction of bumblebees. Journal of Applied Ecology 46:187–193. doi:10.1111/j.1365-2664.2008.01580.x.

Williams, N. M., E. E. Crone, T. H. Roulston, R. L. Minckley, L. Packer, and S. G. Potts. 2010. Ecological and life-history traits predict bee species responses to environmental disturbances. Biological Conservation 143:2280–2291. doi:10.1016/j.biocon.2010.03.024.

Williams, N. M., J. Regetz, and C. Kremen. 2012. Landscape-scale resources promote colony growth but not reproductive performance of bumble bees. Ecology 93:1049–1058. doi:10.1890/11-1006.1.

Yang, L. H. 2004. Periodical Cicadas as Resource Pulses in North American Forests. Science 306:1565–1567.

Yang, L. H., J. L. Bastow, and K. O. Spence. 2008. What Can We Learn from Resouce Pulses? Ecology 89:621–634.