A PROBABILISTIC CHARACTERIZATION OF SHARK MOVEMENT USING LOCATION TRACKING DATA

A Dissertation
Submitted to
the Temple University Graduate Board

in Partial Fulfillment
of the Requirements for the Degree of
DOCTOR OF PHILOSOPHY

by
Samuel S. Ackerman
May 2018

Examining Committee Members:

Dr. Marc Sobel, Advisory chair, Statistical Science
Dr. Richard Heiberger, Statistical Science
Dr. Michael O'Connor, Biodiversity, Earth and Environmental Science (Drexel University)
Dr. Alexandra Carides, Statistical Science
Dr. Richard Souvenir, Computer and Information Sciences

©
by
Samuel S. Ackerman
May 2018
All Rights Reserved

ABSTRACT

A PROBABILISTIC CHARACTERIZATION OF SHARK MOVEMENT USING LOCATION TRACKING DATA

Samuel S. Ackerman
DOCTOR OF PHILOSOPHY

Temple University, May 2018

Professor Marc J. Sobel, Chair

Our data consist of measurements of 22 sharks' movements within a 366-acre tidal basin. The measurements are made at irregular time points over a 16-month interval. Constant-length observation intervals would have been desirable, but are often infeasible in practice. We model the sharks' paths at short constant-length intervals by inferring their behavior (feeding vs transiting), interpolating their locations, and estimating parameters of motion (speed and turning angle) in environmental and ecological contexts. We are interested in inferring regional differences in the sharks' behavior, and behavioral interaction between them.

Our method uses particle filters, a computational Bayesian technique designed to sequentially model a dynamic system. We discuss how resampling is used to approximate arbitrary densities, and illustrate its use in a simple example of a particle filter implementation of a state-space model. We then introduce a particular model formulation that uses conditioning to introduce unobserved parameters for the shark's behaviors. We show how the irregularly-observed shark locations can be modeled by interpolation as a set of movements at constant-length time intervals. We use a spline method for generating approximations of the ground truth at these intervals for comparison with our model. Finally, we demonstrate our model's estimates of the sharks' behavioral and ecological parameters of interest on a subset of the

observed data.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Marc Sobel, and the members of my committee: Dr. Richard Heiberger and Dr. Alexandra Carides of the Temple University Department of Statistical Science, Dr. Michael O'Connor of the Drexel University Department of Biology, for their many hours and invaluable feedback on my work, as well as Dr. Richard Souvenir of the Temple Department of Computer and Information Sciences.

I would also like to acknowledge Dr. Mario Espinoza, Dr. Christopher Lowe, and Thomas Farrugia, who published the data used in the this paper, particularly Dr. Espinoza for his technical assistance, as well as Dr. Noga Neeman, formerly of the Drexel University Department of Biology.

I would also like to acknowledge my family and friends, the staff and faculty of the Department of Statistical Science, and all else who supported me through this long process.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION AND MOTIVATION

A growing area of research in ecology involves the measurement of local movement and migration patterns of animals through positional tracking devices attached to or implanted in these animals. Use of these devices is called telemetry, which in general refers to making measurements remotely. Telemetry has been successfully implemented with a wide variety of species in different habitats, particularly recently in fish, which have historically been difficult to study. Understanding local movement within a habitat can help us learn about the daily behavior (such as feeding) of individual animals, as well as the community structure, and is therefore of tremendous practical importance.

Our dataset contains more than 68,000 two-dimensional position observations of 22 small gray smooth-hound sharks in the full-tidal basin of the Bolsa Chica Wetlands, an area covering about 0.57 mile^2 in Orange County, CA (see Figure 1.1). Based solely on the position observations and the speed and directional data derived from them, we believe we can identify two distinct behaviors: feeding and transiting. Our general approach will be to use particle filters, which are sequential simulations of data, in this case the sharks' movement patterns. The particle filter simulations will distinguish between the two hypothesized behavior types and model the movement patterns characterized by each behavior.

Our particle filter model uses several parameters to describe shark behavior; we estimate their posterior distributions using sequential Bayesian updates. We will answer biologically-relevant questions about the sharks' behaviors. These include:

- Are sharks more likely to display certain kinds of feeding behaviors in certain regions of the tidal basin?
- Do sharks that are near each other tend to exhibit similar behaviors beyond what is expected from regional association? That is, does behavior type exhibit statistically-significant spatial-temporal clustering?

Identifying feeding areas is important in general for learning about the sharks' overall behavior; these areas may also be ecologically sensitive and need additional environmental protection. We will compare the regions identified by our model through shark movements with the regional distribution of prey populations as measured by ecological surveys.

The paper will proceed as follows:

1. In Section 2.2 we describe our dataset in more detail and define the relevant variables used in our analysis.
2. In Section 2.6 we explain the basics of state-space models and particle filters, and present the specification of our model and its Bayesian setup[1]:
3. In Section 3 we show how results from the model simulations can be used for inference about parameters of interest, particularly the regional behavior type probabilities and movement parameters for the two behavior types, to answer the biological questions we are interested in.

[1]See Section 4.1 for an overview of Bayesian statistics.

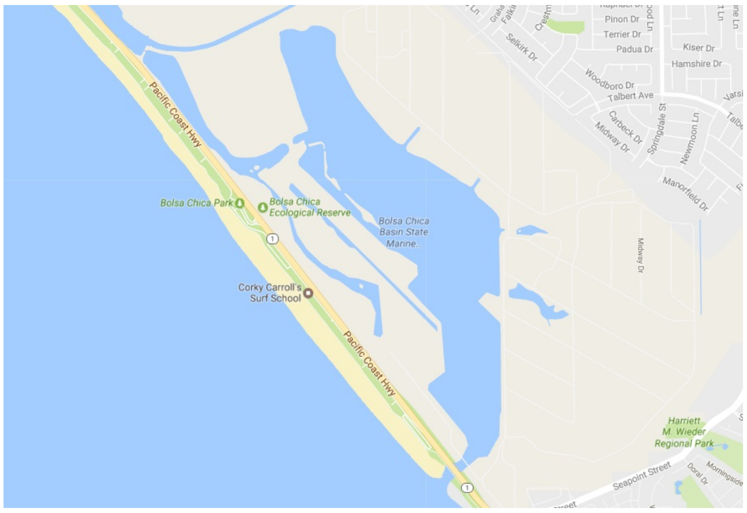

Pacific Coast Hwy
Bolsa Chica Park
Bolsa Chica Ecological Reserve
Bolsa Chica Basin State Marine...
Corky Carroll's Surf School
Pacific Coast Hwy
Harriett M. Wieder Regional Park
Seapoint Street
Talbert Ave
Midway Dr

Figure 1.1: The Bolsa Chica Full Tidal Basin (courtesy of Google Maps). The tidal basin covers about 0.57 mile^2; sharks can exit the tidal basin to the Pacific Ocean through an inlet at the southern end.

4. Lastly, in Section 10, we present an extension of the basic model to jointly infer behavioral interaction effects between sharks.

CHAPTER 2

DATA

2.1 Abbreviations, notation and symbols

iid	independently and identically distributed
m	meters
BCRW	biased correlated random walk
BRW	biased random walk
CDLM	conditional dynamic linear model
CRW	correlated random walk
EKF	Extended Kalman filter
KF	Kalman filter
MCMC	Markov chain Monte Carlo
ML	maximum likelihood
NIG	normal inverse-gamma (prior)
OU	Ornstein–Uhlenbeck process
PF	particle filter
RMSE	root mean squared error
R(R/S)WR	random (re)sampling with replacement
SIR	sequential importance (re)sampliing
SSM	state-space model

UTM	Universal Transverse Mercator coordinate system
VM	von Mises distribution
VPS	VR2-W Positioning System
WN	wrapped normal distribution

$^{(n)}$ or $^{(j)}$	(superscript) variable for particle n or j
$^{(r)}$	(superscript) variable for region r
$\tilde{}$	(top tilde) value after resampling
$\hat{}$	('hat) parameter or value estimate
$\_$	(underline) smoothed value of parameter or variable
$\emptyset$	empty or null set
$x := y$	variable x is defined as 'y'
$x \overset{d}{\rightsquigarrow} f(\cdot)$	the random variable x converges in distribution to density f
$i : j$	sequence of integers $i, \ldots, j$
j'	index of same type as j, but to differentiate it from the particular index j, for instance two particles n and n' which may or may not be different. Also denotes posterior updates of a parameter, such as $\alpha' = \alpha + \ldots$
$\vert \cdot \vert$	set cardinality or absolute value, depending on context
$\Vert \cdot \Vert$	Euclidean distance
$\{\cdot\}_{j=1}^{J}$ or $\{\cdot\}$	set of objects with indices j, or indices dropped for clarity
f	a general function
$\mathbf{f}$	a vector
$\mathbf{f}_{i:j}$	ordered set of vectors $\mathbf{f}$ indexed $i : j$
$\mathbf{f}(\cdot)$	vector-valued function
$\mathbf{F}$	a matrix, possibly time-indexed
$\mathbf{F}(\cdot)$	matrix-valued function, particularly a Jacobian

&	and
$a^{(n)}$	particle rescaling factor

$A(\cdot)$	overall classification accuracy
$A(\cdot \mid k)$	classification accuracy conditional on the true class being k
$\mathcal{Ber}$	Bernoulli distribution
$\mathcal{Beta}$	Beta-distribution
c	time subscript for variables if measured at time intervals of constant length
c_{thresh}	maximum number of constant intervals c allowed between observed times for simulation
(c, s)	ordered pair of time Υ_c index and shark s for joint modeling
C	total number of constant-length time intervals c
$d(\cdot)$	a generic distribution function
$d(x \mid \dots)$	density function $d(\cdot)$ with parameters $\dots$ evaluated at the value x
$\widetilde{d}(\cdot)$	truncated distribution of $d(\cdot)$
$\mathcal{Dir}$ or $\mathcal{Dir}_K$	K-variate Dirichlet distribution
$\mathrm{E}(\cdot)$	expected value
$\mathcal{G}$	Gamma distribution
$\mathcal{G}^{-1}$	inverse-Gamma distribution
$h(s, j)$	clock time of shark s's j^{th} observation
H_t	clock time of observation t
$\mathrm{I}(\cdot)$	indicator function
k, K	indices usually used for categories
$\ell(\cdot)$	density function of a state-space model state/dynamic equation, or a likelihood function
$\boldsymbol{\ell}_\mu$	mean function of a state-space model state/dynamic equation $\ell(\cdot)$
$\mathcal{Lognormal}$	log-normal distribution
$m(\cdot)$	density function of a state-space model measurement equation, or generally a marginal density

$\boldsymbol{m}_\mu$	mean function of a state-space model measurement equation $m(\cdot)$
m_{current}	marginal density of $m(\mathbf{y}_t \mid \mathbf{x}_t)$ where both are observed up to the same time index t
m_{interp_k}, $m_{\text{interp}_{t\mid k}}$	marginal predictive density m_{predict} for irregularly-observed data $\mathbf{y}_t$ (interpolated to constant-length time intervals), conditional on behavioral mode $\lambda = k$, or more specifically, the predictive of the observation $\mathbf{y}$ at time t, given behavior $\lambda = k$.
m_{predict}	marginal density of $m(\mathbf{y}_t \mid \mathbf{x}_{t-1})$ where the next $\mathbf{y}$ is predicted at t based on simulated $\mathbf{x}$ up to $t-1$
m_{predict_k}	marginal density m_{predict} conditional on the intermediate $\lambda_t = k$
n	particle index
n_j	particle index at iteration j of smoothing
$n_{a,b,i\to j}$	cumulative number of transitions between behaviors i and j, between time steps a and b, $1 \leq a \leq b$
$n_{a,b}$	cumulative number of transitions between time steps a and b, $1 \leq a \leq b$. By definition, this equals $b - a$.
$n(s)$	the index of the original particles that is resampled as new index s (in context of low-variance sampling)
N	total number of particles n
N_{eff}	particle filter effective size
N_{thresh}	threshold for resampling particles based on effective size
$\mathcal{N}$ or $\mathcal{N}_k$	univariate or multivariate normal distribution or density function
$X \sim \mathcal{N}(\mu, \sigma)$	random variable X is distributed as normal with mean μ and standard deviation σ
$\mathcal{N}(x \mid \mu, \sigma)$	normal distribution density with mean μ and standard deviation σ evaluated at the value x
$N(\,\cdot\,)$	spatial-temporal neighborhood

$N^{(r)}_{a,b}$	matrix of cumulative counts of behavior transitions $n_{a,b,i\to j}$ for region r between times a and b
$O(\cdot)$	order of
p	a distribution function, particularly for the prior
p_{current}	predictive density of $d(\mathbf{x}_t \mid \mathbf{y}_{1:t})$ where both are observed up to the same time index t
p_{predict}	predictive density of $d(\mathbf{x}_t \mid \mathbf{y}_{1:(t-1)})$ where the next $\mathbf{x}$ is predicted at t based on observations up to $t-1$
p_i	a probability value, such as one drawn from a Dirichlet or Beta distribution
$p_{i\to j}$ or $p_{c,i\to j}$	transition probability between behaviors i and j, or this probability estimate as of time c
$p^{(r)}_0$	true foraging probability for region r
$P^{(r)}_c$	transition probability matrix for region r, at time c
$\Pr(\cdot)$	probability of an event
$X \sim \Pr(\cdot)$	categorical distribution of random variable X
$\mathbf{q}_c$ or $\mathbf{q}_t$	error term of SSM state/dynamic equation, at time c or t
$\mathbf{Q}_c$ or $\mathbf{Q}_t$	covariance matrix of error term of SSM state/dynamic equation, at time c or t
r	region index
R	total number of regions
$\mathbb{R}$	the real number line
$\mathbf{r}_c$ or $\mathbf{r}_t$	error term of SSM measurement equation, at time c or t
$\mathbf{R}_c$ or $\mathbf{R}_t$	covariance matrix of error term of SSM measurement equation, at time c or t
R^2_C	net squared displacement (Marsh-Jones statistic)
s	shark index
s_t	the shark s that corresponds to observation $\mathbf{y}_t$
S	total number of sharks; also index of size of resampling (see low variance sampling)

$\mathbf{S}_t^x$	sufficient statistics of states $\mathbf{x}_t$
t	general time subscript for variables, specifically if measured at time intervals of non-constant length
T	total number of observations, specifically if measured at time intervals of non-constant length
$t(s, c, j)$	index t of pooled observations $\mathbf{y}_{1:T}$ of shark s's j^{th} observation in interval c, if defined; otherise equals $\emptyset$
v_c or v_t	velocity at time c or t
w_c or w_t	particle resampling weights (sometimes with superscript (n)), at time c or t
$w_{j,c}$	particle resampling weights (sometimes with superscript (n)), at time c for smoothing iteration j
$w_{c\|k}$ or $w_{t\|k}$	particle resampling weights (sometimes with superscript (n)), at time c or t, conditional on the behavior λ_c or $\lambda_t = k$
$\tilde{w}_c$, $\tilde{w}_{c\|k}$, etc.	value of particle weights w_c or $w_{c\|k}$, etc. after resampling
$\bar{w}_n$	cumulative sum of particle weights up to index n, i.e., $\bar{w}_n = \sum_{j=1}^{n} w^{(j)}$
$\mathcal{WN}$	wrapped normal distribution
$\mathcal{W}^{-1}$	inverse-Wishart distribution
u	a random draw from a continuous uniform distribution $\mathcal{U}$
$\mathcal{U}$	continuous uniform distribution
$x_{c,i}$	$\mathrm{I}(\lambda_c = i)$, for purposes of likelihood formulation (see Section 6)
$\mathbf{x}_c$ or $\mathbf{x}_t$	unobserved measurements at time c or t
$\underline{\mathbf{x}}_c$ or $\underline{\mathbf{x}}_t$	smoothed value of state $\mathbf{x}$ at time c or t
$\mathbf{y}_t$	observed measurements at time t
$z_t, \boldsymbol{z}_t, z_{1,t}, z_{2,t}$	observed spatial location, either as a single value z_t (1-D), or vector ($\boldsymbol{z}_t$) of horizontal ($z_{1,t}$) and vertical ($z_{2,t}$) coordinates
$\mathbb{Z}$	set of positive integers $\{1, 2, \ldots \infty\}$
α_c or α_t	mean of velocity or log-velocity at time c or t

$\boldsymbol{\alpha}$ or $\alpha^{(r)}$	vector of Dirichlet parameters, or a single Dirichlet regional parameter
β_c or β_t	mean of turn angle, in radians, at time c or t
$\delta_{\boldsymbol{\zeta}}$	spatial radius in meters of neighborhood $N(\cdot)$
δ_{Υ}	temporal length in seconds of neighborhood $N(\cdot)$; all neighbors (c', s') have to be less than δ_{Υ} seconds before
$\Delta_{s,c,j}$	for interpolation, refers to all the observations of shark s in constant-length interval c (see $t(s,c,j)$). If $j = 1$, the gap from the observation time to Υ_c; otherwise, it's the time gap from observation $j-1$ to j in the interval.
Δ_t	the time difference from observation t to $t+1$, for varying interval lengths
Δ_{Υ}	constant time difference between simulated observations
Δ_{MJ}	Marsh-Jones statistic
$\zeta_t, \boldsymbol{\zeta}_t, \zeta_{1,t}, \zeta_{2,t}$	(zeta) unobserved spatial location, either as a single value ζ_t (1-D), or vector ($\boldsymbol{\zeta}_t$) of horizontal ($\zeta_{1,t}$) and vertical ($\zeta_{2,t}$) coordinates; either at t or c
η	various subscripts: degrees of freedom parameter for an inverse Wishart ($\mathcal{W}^{-1}$) distribution
η_k	neighborhood behavioral influence parameter for $\lambda_c = k$
θ_c or θ_t	turn angle (radians) at time c or t
$\boldsymbol{\theta}_c$ or $\boldsymbol{\theta}_t$	vector of state-space model unobserved parameters, at time c or t
λ_c or λ_t	latent variable value (particularly behavior), at time c or t
μ or $\boldsymbol{\mu}$	generally a mean value or vector
$\boldsymbol{\Lambda}$	scale matrix for an inverse Wishart ($\mathcal{W}^{-1}$) distribution
$\Theta(\cdot)$	wrapping function that wraps input values to the interval $[-\pi, \pi]$
$\pi_{c,s,k}$	proportion of spatial-temporal neighbors of (c, s) that have behavior $\lambda_{c,s} = k$

$\boldsymbol{\pi}$	normalized eigenvector of square matrix for modeling stationary probabilities
ρ_c	spatial-temporal rescaling factor
σ_c^2 or σ_t^2	variance of either velocity v or log-velocity $\ln(v)$ at time c or t
$\boldsymbol{\Sigma}$ or Σ	a covariance matrix
τ_c^2 or τ_t^2	variance of turn angle θ (radians) at time c or t
Υ_c	clock time of constant-interval measurement c
ψ_c or ψ_t	bearing angle at time c or t
$\boldsymbol{\Psi}$	hard domain constraint
$\boldsymbol{\Omega}_c$ or $\boldsymbol{\Omega}_t$	prior distribution vector of hyperparameters at time c or t

2.2 Data

The shark telemetry data we use were collected for the published paper by Espinoza, Farrugia, and Lowe ([16]). The gray smooth-hound shark is a small shark which is a major predator–its diet consists mainly of crabs, small fish, scallops, and other shellfish–in the estuary, whose presence is considered an indicator of the level of ecological restoration in such a marine habitat. As part of this paper, the authors detail their efforts to catch several hundred gray smooth-hound sharks in nets for measurement, and to track 22 of them with implanted telemetry equipment. They placed 16 evenly-spaced acoustic receivers in the tidal basin to record the sharks' positions when sharks passed within range of the receivers. Shark positions were calculated by triangulation using the differences in detection times between receivers. Thus, there is significant variation in the time gaps between observations. These gaps may have occurred due to interference from obstacles (such as mud), or if a shark left the tidal basin out of receiver range. This is a significant modeling challenge that we will address.

Figure 2.1 shows the time spans recorded for each shark. Three of the 22 sharks died during data collection. In general, sharks were observed for relatively short time spans, usually around a month or less. The primary reason, according to the study's authors, is that the habitat was relatively new since it was being restored, and food resources could be limited compared to other habitats. Thus individual sharks had not gotten accustomed to it as a more permanent habitat and did not stay for extended periods. If a shark left for another habitat, observation ceased as long as they were out of range of the receivers. Some sharks, such as shark #17, returned after several months absence. Furthermore, though the observation spans of sharks tagged in the same year overlap significantly, the two groups of sharks tagged in different years (nine in 2008, thirteen in 2009) did not overlap at all; this limits our ability to infer the degree of inter-shark influence.

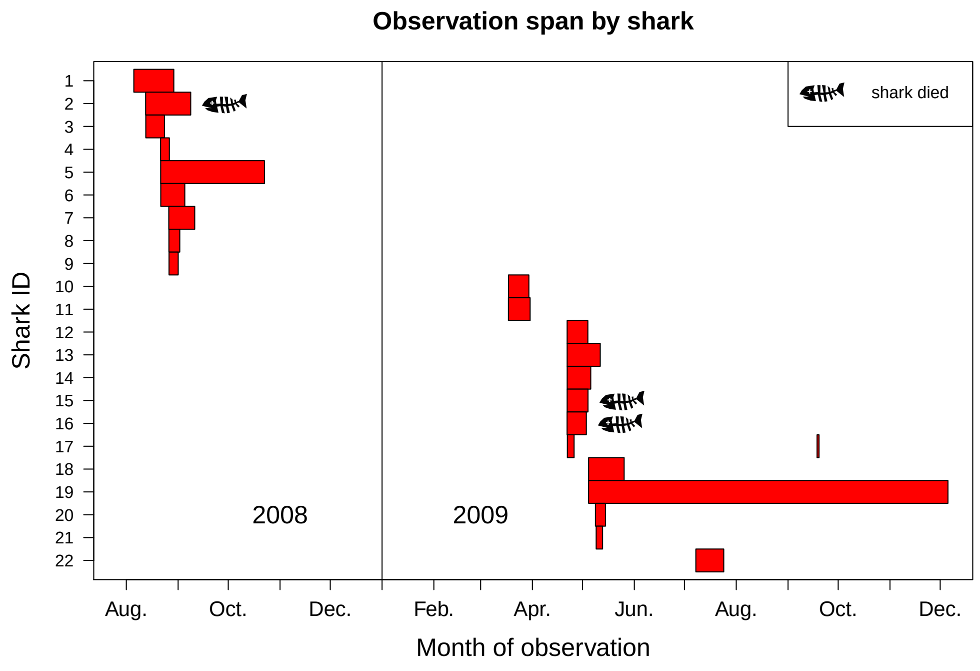


Figure 2.1: Spans of time each of the 22 sharks was observed. Most sharks were only observed for relatively short periods.

2.3 Variables

The following variables will be used to model the sharks' movements. Variables are specific to a given shark s, but we omit this index for clarity here. Depth coordinates were not included because the tidal basin is shallow and we assume depth is not significant in modeling the behavior. In applications involving marine turtles, for instance, which dive, this would not be true. Spatial-temporal covariate variables, such as the temperature, presence of vegetation, or tide level at a coordinate, were not available to assist in modeling behavior; other movement models have successfully incorporated such information. Also, we do not have acceleration data, which would be helpful in modeling speed; devices with accelerometers apparently began to be used in research around the time our data was collected.

- Let t be the time index of a recorded observation for shark s, and let H_t be the actual time of the observation. The index c will later be used to index variables simulated at time steps of equal length.
- At time indexed t, shark s is observed at position $\boldsymbol{\zeta}_t = \begin{bmatrix} \zeta_{1,t} & \zeta_{2,t} \end{bmatrix}^T$ (measured in UTM coordinates) at speed v_t (m/s).
- Let $\psi_t \in [-\pi, \pi]$ be the bearing angle, the shark's direction of movement from $\boldsymbol{\zeta}_t$ (defined relative to a fixed point, here the horizontal axis).
- The turn angle $\theta_t \in [-\pi, \pi]$ is the change in bearing angles ($\psi_t = \psi_{t-1} + \theta_t$). $\theta_t < 0$ means the shark is turning clockwise relative to its previous direction.
- Δ_t is the observed time in seconds between observations t and $t+1$ for a given shark. When observations are pooled, Δ_t is still the time difference between consecutive observations of the same shark, but the next index after t is not necessarily $t+1$ if another shark s' was observed between them.

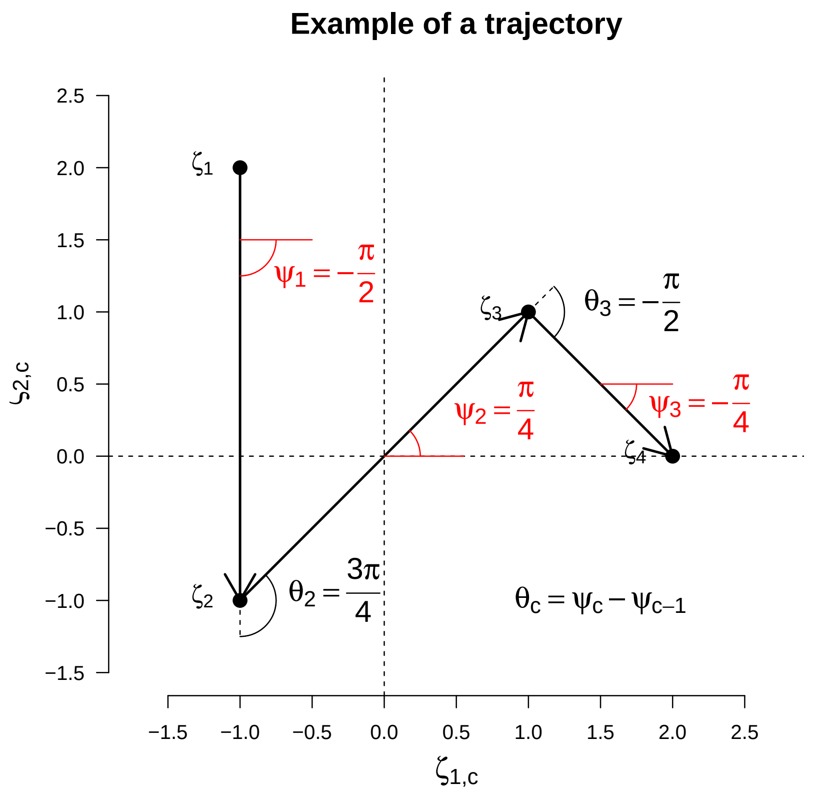


Figure 2.2: Toy illustration of a trajectory in the $\boldsymbol{\zeta}$-coordinate plane with the bearing angle ψ_c and turn angle θ_c in radians.

A toy example of a trajectory using these variables is shown in Figure 2.2.

Noting that animal movement patterns depend on the animal's behavior type (particularly foraging or transiting, i.e., movement between feeding areas), a common approach for modeling animal movement is to model movement separately for each behavior type. For instance, for some marine animals, foraging behavior can be characterized by frequent changes in movement direction (large turn angles between positions) and slower speed, while transiting is often characterized by faster speed and less change in direction. One approach to model the overall trajectory is to model it as a mixture of the two behavior-specific movements. This is illustrated, for instance, in Morales et al ([36]) with elk, Jonsen et al ([24]) with leatherback sea turtles, and Roever et al ([43]) with African elephants; these studies use state-space model (SSM) techniques, discussed in detail in Section 3. Of course, the model specifics need to be customized to each particular species with biological domain knowledge. In our case, we denote the behavior type at time t by λ_t, and use the foraging/transiting paradigm, where

- foraging ($\lambda_t = 0$) is generally when the shark moves with low speed (v_t) and sharp back and forth motion, shown by large turn angles ($|\theta_t|$ is large), and
- transiting ($\lambda_t = 1$) is characterized by higher speed and straighter movement ($|\theta| \approx 0$).

We believe the sharks' movement can be characterized by periods spent foraging in certain areas and then movement (transiting, essentially all other type of movement) between these regions.

2.4 Spline interpolation

One of the major challenges of dealing with telemetry data is that often data are observed at irregularly-occurring time intervals. This is usually due to technical limitations of the telemetry equipment, such as limited range or interference by landscape features, or the behavior of the animals themselves. For instance, certain marine animals travel long distances into open water or dive for long periods of time, during which it may be difficult or impossible to measure their location.

The empirical density of the observed time gaps Δ_t in seconds between observations, for all sharks, is shown in Figure 2.3; the density is restricted to $\Delta_t < 3600$ seconds (1 hour), which is above the empirical 98^{th} percentile of time gaps. In our data, the raw Δ_t have $Q_1 = 68$ seconds, a median of 130, and $Q_3 = 299$ (5 minutes); the mean is 826 seconds (about 13.75 minutes), but is significantly affected by extreme values. The irregularity of observation is due to interference (such as from the mud banks near the coast of the tidal basin) and the pinging rate of the shark VPS (VR2-W Positioning System) transmitters, which were between 40 and 80 seconds. The pinging rate measures how often a signal is transmitted, meaning that a shark will not be observed at time gaps of less than 40 seconds; indeed, the smallest observed time gap was 43 seconds.

Performing model building and inference on irregularly-observed data is problematic. It is true that analysis of anything other than completely continuously observed trajectories requires some abstraction from the true trajectory. For instance, the simplest way to guess the trajectory between a set of observed locations is to 'connect the dots' and assume the animal travels at straight lines between the locations, when in reality the movement is probably more sinuous like a spline. Nevertheless, the simplification of using Euclidean distance between locations often greatly aids model-building and computation. If we accept this simplification, however, creating a linear interpolation from irregularly-observed locations—especially if the degree of irregularity, i.e., the variance of time steps Δ_t, is large, as in our case—is problematic for several reasons. These issues are problematic even if we have regular observations but with long time gaps ('long' depends on the particular species and its typical range of movement), but are compounded with the case of the irregularly-

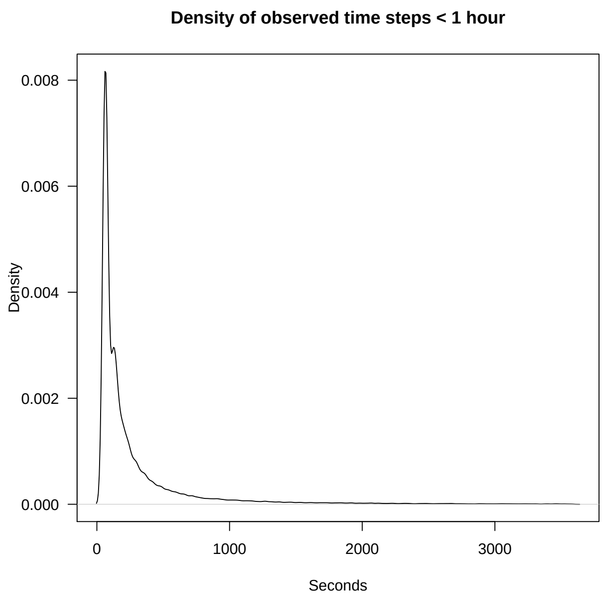


Figure 2.3: Empirical density of time gaps Δ_t (seconds), restricted to show $\Delta_t < 1$ hour.

observed time points.

The first problem reflects the fact that with increased time gaps, the amount of uncertainty involving modeling the actual trajectory of the animal between two locations is increased. The longer the time has elapsed, the more likely that the animal's true trajectory between observations is more sinuous and involves more back and forth motion, for instance, and thus is more likely to have departed from the straight line Euclidean simplification. One can set a reasonable upper bound on the time gaps where beyond this it is relatively useless to model the animal's trajectory between two points because the set of potential trajectories so large.

In sequential Bayesian models (see Section 4.1), which we will discuss in more detail, model parameters, such as for the distribution of turn angle and speed, are updated sequentially with each new point of data. The standard parameter updating formulas assume we are dealing with some random sample where each data point is equally weighted, or else observations weights must be incorporated; in update formulas, earlier-observed point have more influence in the overall parameter calculation because as more data is observed, we are more confident in the cumulative sample mean and thus weight new data points less and less, but this is a separate issue. If we choose to weight observations differently based on the time elapsed between them, formulating a weighting scheme can be difficult.

Secondly, irregular observations tend to violate many of the assumptions of common models. We will examine later the correlated random walk (CRW) model, a frequently-used model where the turn angles at each step are independent and symmetric around 0, so the animal is equally likely to go left or right. The variance parameter of the turn angle distribution controls the animal's directional persistence, meaning its tendency to continue in roughly the same direction over successive time points.

Accepting the Euclidean distance simplification, if the data are irregularly observed, we cannot treat the empirical distributions of speed ($\hat{v}_t$), calculated as the Euclidean distance between locations divided by the time gap, and of the turn angles ($\hat{\theta}_t$), calculated as the angular change at a location between its previous bearing direction ψ_{t-1} and its new direction ψ_t, assuming straight lines, as representing an approximation of reality. Since Euclidean distance is the lower bound on the potential distance actually traveled between two points, the longer the time gap Δ_t, the worse of an approximation of reality this straight line becomes, since the shark may have actually traveled a more sinuous, and thus longer, path between them. Thus, the actual average speed along the true trajectory is likely to be much higher than the Euclidean estimate $\hat{v}_t$. Similarly, trying to calculate turn angles at irregular or large time gaps is unreasonable. As noted by Gurarie ([19], p. 28), one cannot assume that the distributions of turn angles θ_t and speeds v_t are identically distributed when the gaps are irregular, but rather their distributions depend on the length of the gaps Δ_t. Thus, fitting a CRW to, or performing diagnostic tests to determine the appropriateness of a CRW model, using highly irregular observations, is invalid.

Thirdly, the irregular observations are especially problematic in our case when trying to model the shark's behavioral state (λ_t) at a point in time. A natural approach is to use a probability transition matrix to model how often the shark is likely to stay in the same behavior or change between them. Speed v_t is the primary way to differentiate between the slow foraging and fast transiting behaviors[1]. However, using the average speed $\hat{v}_t$ (assuming Euclidean distance) between locations to determine the behavior λ_t at that point is unreliable because at longer gaps the Euclidean distance becomes a worse approximation of the true distance traveled, and also, over longer gaps the shark is more likely to switch behaviors and its true (unobserved) speed is likely to vary. Thus, our approach, discussed later, will be to specify a constant and small time gap Δ_Υ at which we will model movement by interpolating the irregular timesteps t to constant-length ones c. This way we can model the relative frequencies of behaviors and their switching probabilities and perform inference on the relevant parameters.

Thus, to describe the distributions of the dataset variables to justify our CRW approach, we will not use the raw distributions of speed and turn angle, but rather the distributions from a version of the dataset interpolated to regular time intervals. As a note, we do not claim that the interpolated locations accurately represent the true unobserved locations, but rather that this approach is better than the raw data approach which weights irregularly-observed points as if they represent equal amounts of time, and assumes that the average speed over irregular intervals can describe the shark's behavior over the whole interval. Also, our modeling approach (discussed later) does not make use of these regular interpolations as inputs or for comparison, but rather for illustration.

To do exploratory data analysis, we will interpolate the observed locations using Bézier (pronounced 'beh-zee-ay') spline[2] curves. Like other splines, Bézier curves use a set of observations as control points to guide the curve

[1]We also use turn angles θ_t to differentiate the behaviors, but these distributions are centered around 0 and thus are harder to use for discrimination.

through a convex hull of the points. The spline will begin and end at the beginning and ending observation, and it will pass near but not through the intermediate control points. We will use this particular spline technique because they were demonstrated by Tremblay et al. ([51]) to be more effective than several other spline methods for interpolating animal trajectories. We use the implementation of Bézier splines in the `bezier` package by Olsen ([39]) for the `R` software language.

A Bézier spline is a piecewise polynomial of degree one less than the number of control points; for instance, two control points trivially give a line connecting them, and four control points give a cubic spline, which we use. Figure 2.4 illustrates interpolation of a quadratic (second-degree) polynomial with three control points. For the case of three control points, lines are drawn to connect the three points in order (in our case, the order in which they were observed). Consider a set of values $t \in [0, 1]$ (in the illustration, $t \in \{0.00,\ 0.25,\ 0.50, 0.75, 1.00\}$); points are marked along the line segments connecting the control points at the fractions of t. Then, line segments connecting each of the pairs of points at the same fractions are drawn; in the illustration, these are the blue, green, and red line segments. Then, assuming enough points t are used, the spline is a smooth curve along the hull of these line segments. Points at fractions t along the spline are found by marking points at the corresponding fraction t along these new line segments.

One key feature of the Bézier spline illustrated above is that selecting equally-spaced fractions t along the interval $[0, 1]$, as in the figure, does not give equally-spaced points along the spline, as long as the control points are not on a single line. Where the spline curvature is higher, such as at the 50% point in Figure 2.4, the t points are spaced closer together, and the distance between them is longer where the spline is less curved, such as near the first and last control points. The advantage of this is that if we take the t fractions to represent time, if these fractions are equally spaced, we can generate a set

[2]Kamermans' online book ([25]) provides an extensive background on the mathematics of Bézier curves.

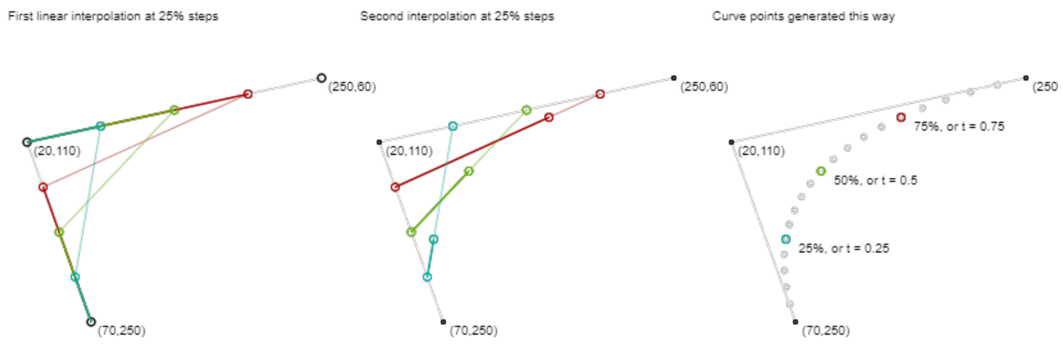


Figure 2.4: Illustration of a quadratic Bézier curve interpolation for three control points located at $(70, 250), (20, 100)$, and $(250, 60)$. Source: Kamermans ([25]).

of predicted locations that are not equally spaced along the spline, as if the shark is slowing down as it turns around the high-curvature sections of the spline (since the distance between these spline points is small but the time gaps in t are equal).

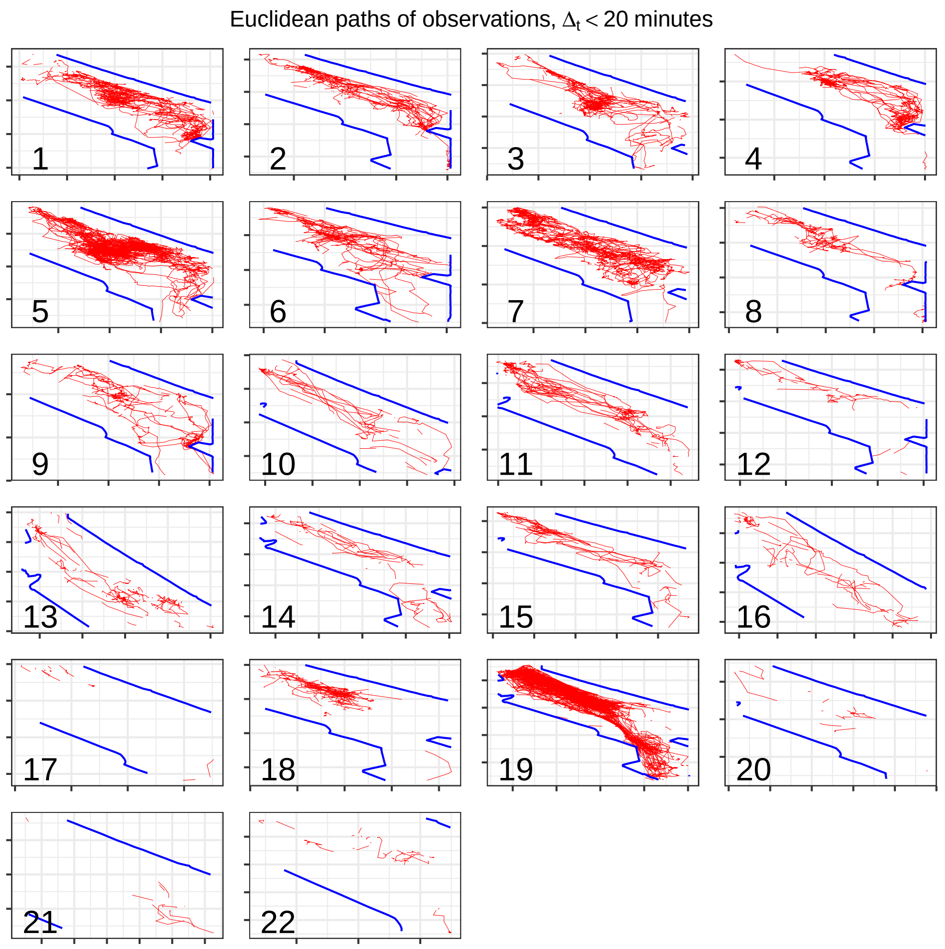
We choose to use Bézier splines instead of other available spline methods, such as B- or X-splines, which may more truthfully represent a curved path between observations that can be forced to pass through all of the control points. The reason is that Bézier splines, despite the fact that they do not pass through all control points, provide a natural way to generate locations along the spline that may be used to represent locations equally spaced in time but not in distance along the spline. Many software implementations of splines do not allow user control over returning locations along the spline, such as in number of points desired. Furthermore, the points along the spline that these methods generate are often simply those used to construct the spline, and there is no practical interpretation of them in terms of the time elapsed. Thus, they are an improvement from simple Euclidean distance in modeling speed, in that they lengthen the distance between observations, but we cannot use them to approximate changing speed of movement along the spline path.

As we have mentioned, we want to model the irregularly-observed shark data at regular time intervals, so to analyze some features of the observed data we will generate regularly-spaced time intervals on piecewise cubic Bézier splines. As an example, the points will be spaced at $\Delta_\Upsilon = 120$ seconds (2 minutes) along sequences of observations where each observation is spaced no more than $10\Delta_\Upsilon$ seconds (20 minutes) away from each other. We do not claim that these spline-interpolated points accurately represent the actual locations of the shark at constant intervals of $\Delta_\Upsilon = 120$, but rather that for our exploratory analysis these represent a better alternative. The Bézier spline interpolation results both in a nonlinear trajectory and varying speeds of travel between observations, which is more realistic than constant-speed linear Euclidean interpolation of the observations. Also, since the speed slows around sharp curves, this matches our model of foraging being represented by slower speeds and more curved movement. Especially since by definition, the Bézier spline does not go through the middle control point observations used in the piecewise spline, we do not use this interpolation as a ground truth either as direct input into our algorithm or as a metric to evaluate our results, but rather simply for exploratory analysis.

Figure 2.5 shows each shark's observations, with $\Delta_t < 20$ minutes interpolated with (naive) Euclidean linear interpolation and with Bézier spline interpolation. To differentiate regular steps from irregular steps, we will use the following notation:

- t will be used to index irregular steps $\mathbf{H}$. We consider a set of T (irregularly-spaced) observations at times $H_1 < H_2 < \cdots < H_T$. Here the time gap is $\Delta_t = H_{t+1} - H_t$.

- c will be used to index regular steps $\boldsymbol{\Upsilon}$, where the time gap is a (relatively small) constant Δ_Υ. The $C + 1$ constant-length timesteps are $\Upsilon_0 < \Upsilon_1 < \cdots < \Upsilon_C$, where $\Upsilon_0 = H_1 - \epsilon$ (a small number so that the first observation time H_1 falls in a regular interval that is open on the left) and $\Upsilon_c = H_1 + c\Delta_\Upsilon$, $c = 1, \ldots, C$, where $C = \underset{c}{\operatorname{argmax}}\{\Upsilon_c \leq H_T\}$ (i.e., enough regular steps to cover the set of irregular observations).



Based on the spline interpolation to regular steps, we can calculate the speed the shark would have traveled between these points, then take the natural log. This is done so that the resulting density plots can be modeled as a mixture of Gaussian distributions, as shown in Figure 2.6. As the figure shows, roughly half of the shark spline-interpolated trajectories have what can be modeled as a mixture of normals, with the division between the two components at $\ln(v_c) \approx -3$ or so. Thus, for instance, for our data exploration we can tentatively label behaviors $\lambda_c = 0$ (foraging) if $\ln(v_c) < -3$, otherwise $\lambda_c = 1$. Figure 2.7 shows the analogous log-speed densities when the observed points are interpolated to constant-length intervals of 2 minutes using the Euclidean distance (i.e., a Bézier spline with degree 1). The results are not too different

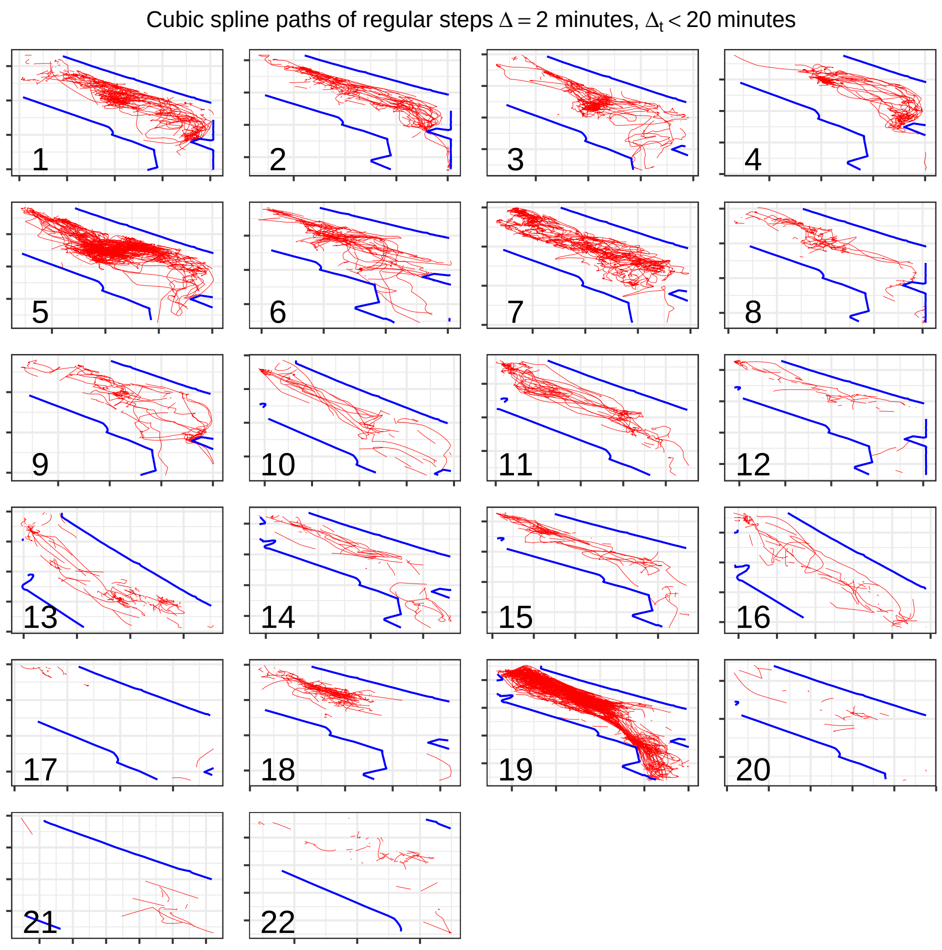


Figure 2.5: Actual shark observations in sequences with $\Delta_t < 20$ minutes between observations, connected by Euclidean interpolation (top) and cubic spline interpolation to regular steps $\Delta_\Upsilon = 2$ minutes.

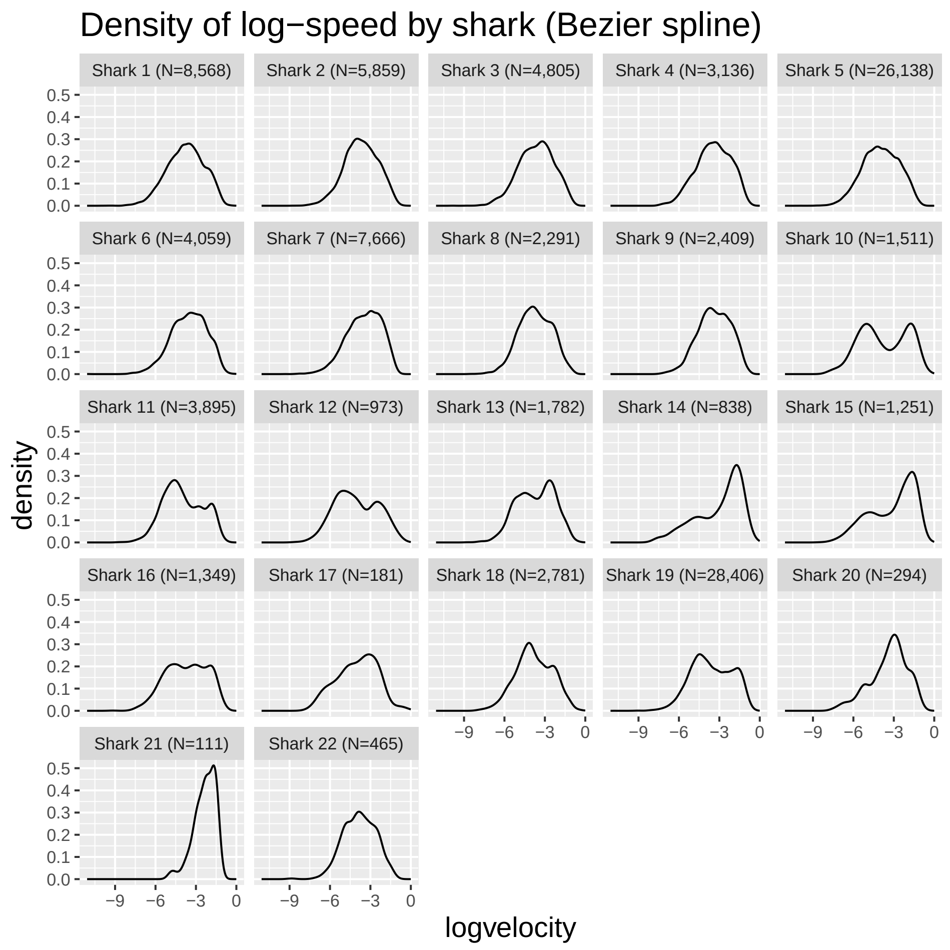


Figure 2.6: Density plot of log-speed $\ln(v_c)$ between Bézier spline-interpolated points.

in terms of the density estimates, but with Euclidean interpolation the speed is assumed to be constant between the observation, which we want to avoid. As mentioned before, the approach of modeling the animal's different behavior types as a mixture distribution with switching probabilities between the types has been used by other authors such as Morales et al. ([36]) and Jonsen et al. ([24]). Additional relevant theory about CRWs is reviewed in Section 3.2.

Figure 2.8 shows the joint density of the distributions of calculated log-speed ($\ln(v_c)$ and $\ln(v_t)$) and turn angle in radians (θ_c and θ_t) from the regular interval spline interpolated positions (at left) and the raw observations (right).

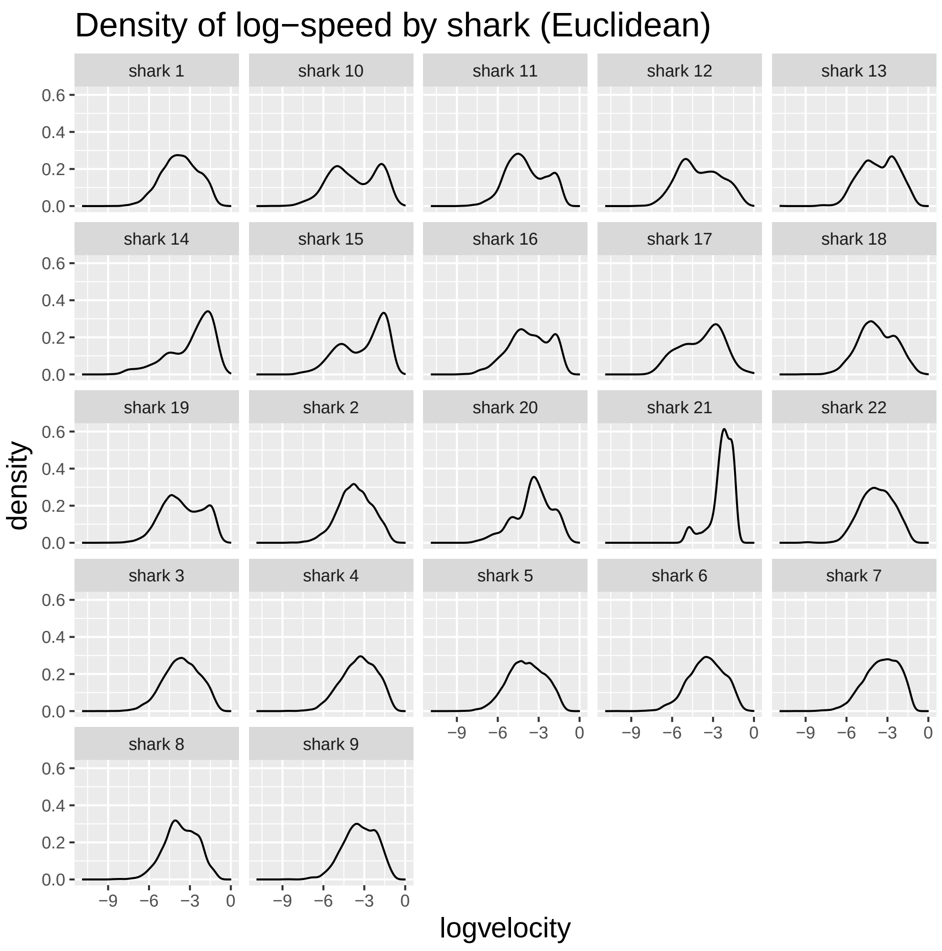


Figure 2.7: Density plot of log-speed $\ln(v_c)$ between Euclidean-interpolated points.

As mentioned earlier, examining these variable distributions from intervals of irregular time gaps is not very valid as a data exploration or modeling exercise. For the interpolated data (left), the turn angles are tightly concentrated around $\theta_c = 0$, due to the fact that the time gaps are small and the Bézier spline, by its nature, forces the trajectory to be smooth, and most of the turn angles are small, even if the speed is low around the peak of the curve. Our animal model posits that low speeds tend to represent foraging, that animals that forage will have larger turn angles, based on domain knowledge. We will maintain this model and keep in mind that the interpolation is only a guide.

For the irregular steps (right), which we acknowledge are not valid to use for the distributions of the speed and turn variables, we note that there is a concentrated area of observations with higher speeds and small turn angles, and a region with lower speeds and wide turn angles. These tend to arise when there are large time gaps.

2.5 Empirical resource distribution data

One of our primary questions is whether our model can infer from the shark movements that foraging behavior is more likely to occur in some regions than others. The details of the regional modeling will be discussed in Section 6. Shark behavior type is unobserved and we we infer it based on the movement patterns. To support this hypothesis, we will compare empirical data on the spatial distribution of the sharks' prey with our inference of regions where foraging is most likely. If, according to our model, foraging behavior tends to take place in regions where common species of shark prey can be found, this will lend support to our hypothesis.

In a second paper, Farrugia, Espinoza, and Lowe ([17]), the authors of the original study, collected samples of other fish and invertebrates in seine nets in the tidal basin over the same period, to measure the species' spatial distribution. By using counts of the number of specimens of each species collected in each seine net, we can calculate a spatial measure of relative abundance

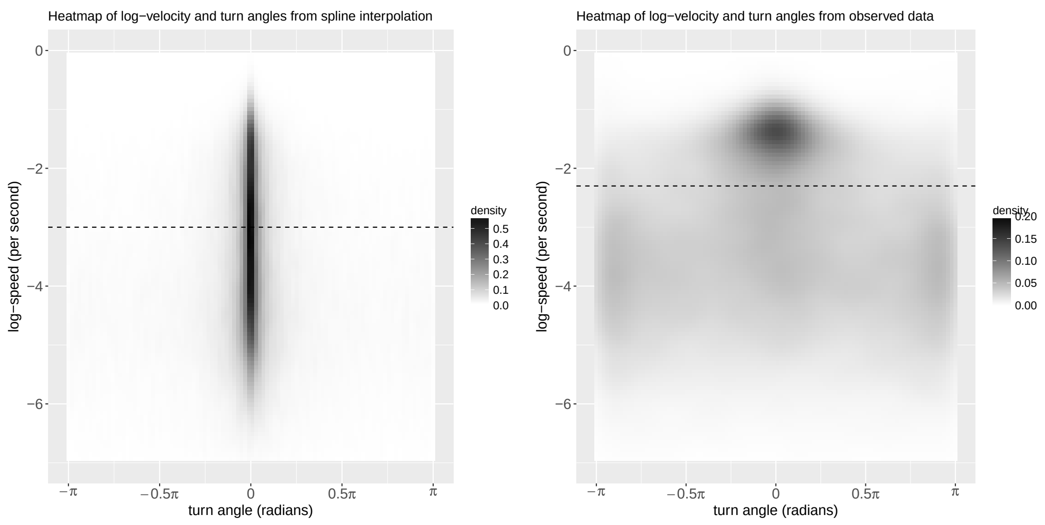


Figure 2.8: Joint distribution of log-speed and turn angle in radians for all observations, for the spline interpolated positions (left, $(\theta_c, \ln(v_c))$) and the raw observations (right, $(\theta_t, \ln(v_t))$). The raw observations are at irregular time gaps from each other, so examining the distributions of speeds and turn angles between them is not very valid. In each, a horizontal dashed line is drawn at the value of $\ln(v_c)$ or $\ln(v_t)$ that tentatively divides foraging from transiting behavior.

(i.e., spatial distribution of presence) of that species. The relative abundance measure adjusts the counts of specimens by the surface area of the seine nets at each location and the length of time each net was erected to estimate the number of specimens caught per unit of net area per unit time. Assuming that species are caught (by a fixed size net and over a fixed amount of time) in proportion to their actual abundance at a location, the relative abundance measure should estimate the spatial distribution of species locations.

The relative abundance was calculated for crabs, scallops, and shrimp, three common species of prey that our sharks eat. The measure is calculated separately for each type of prey because of their differing sizes, and it is difficult to combine species of different sizes into a composite measure of species distribution.[3] The relative abundances are shown in Figure 2.9 for each of these species. Note that in the images the density scale colors are normalized to each plot and thus cannot be compared across plots.

We wish to see how our model of foraging vs transiting behavior based on movement corresponds to the estimated spatial distribution of prey. Thus, we will use the spline interpolation of the observations to regular time intervals and the tentative labeling of intervals as corresponding to foraging or transiting behavior to estimate the spatial probability of foraging. Ideally, we should see, keeping in mind that the splines provide a tentative guess of the behavior, that sharks are more likely to forage in areas where the relative abundance of species is higher.

Figure 2.9 indicates that the sharks' prey tend to be found in the coast on the northeast side of the tidal basin (see the dark red regions in the crab and scallops plots) and in the southwest coast of the main tidal basin, above the narrower inlet to the ocean (see the shrimp and crab plots). As shown in

[3]There are techniques in ecology that combine the abundance counts of different-sized marine species into a composite measure of biomass (i.e., the volume of flesh) that can be used to estimate the spatial distribution of the species in terms of the physical mass of the species. For instance, ecologists use samples of species specimens to estimate the average physical dimensions—for instance, length, width, and weight—of a species, and then use simple geometric formulas to estimate the typical 'volume' a particular fish represents. See for instance Kimmerer et al. ([28]).

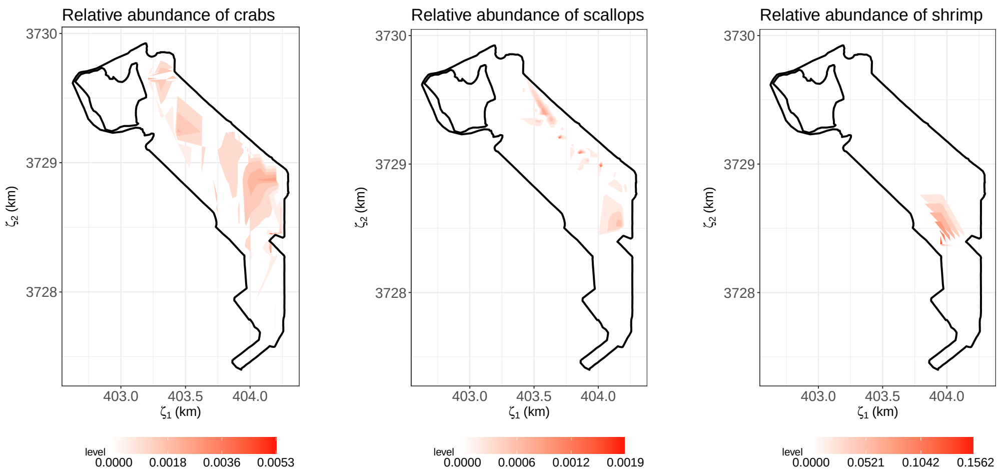


Figure 2.9: From left to right: relative abundance of crabs, scallops, and shrimp–three main food sources of our sharks–based on species samples caught in seine nets. Darker red indicates the given species is more likely to be found at that location. Red scales are relative abundance normalized to each particular species, and thus colors cannot be compared across plots.

Espinoza et al. ([16]), the areas along the diagonal coastal edges of the tidal basin correspond to mudflats, and the gray smooth-hound shark is believed to forage there because the mudflats tend to flood during incoming and high tides, which bring in prey. As Figure 2.10 shows, for both 2008 and 2009 (since the sampled sharks are observed for separate periods in these two years, as shown in Figure 2.1) our spline interpolation model shows that these same coastal regions have the highest spatial distribution of foraging, as approximated by the estimated speed.[4] That we see good correspondence between prey abundance and our behavioral model gives credence to this approximation.

[4]The spatial distribution of foraging was calculated by taking a sample of spline locations equally spaced by time, converting it to a marked (categorical label) Poisson point process, interpolating locations to a regularly spaced grid of coordinates, and estimating the spatial relative 'risk' of each behavior at each coordinate. See the `spatstat` package for `R` by Baddeley et al. ([3]).

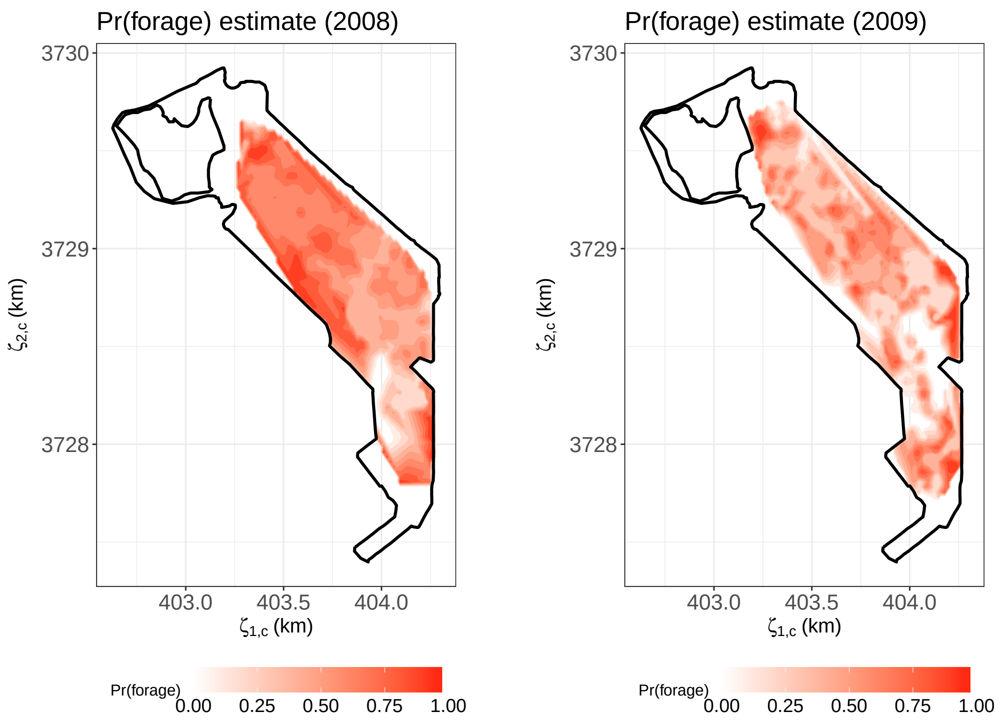


Figure 2.10: Estimated relative spatial probability of foraging based on random sample of 10,000 from 2-minute spline interpolation of each year's observations (out of approximately 50,000 per year). A sub-sample was used due to computational expense of estimating the relative spatial distribution, which is exponential in the number of points.

2.6 Modeling approach

Our basic approach is to model individual sharks' movements as a correlated random walk (CRW), which is commonly used in biology (see Codling et al., [12]) to model animal movements. In the following discussion, we will use the time subscripts t, which denote the unequally-spaced observations, even though ultimately we use constant-length intervals c, for consistency with literature. In a CRW, the turn angle (θ_t), meaning the change of direction, of movement, at a specific time t is random. As mentioned, the turn angle is the difference between consecutive bearing angles ψ_{t-1} and ψ_t. A small turn angle means the animal tends to maintain roughly the same direction over multiple steps. The CRW is in contrast to a simple random walk (considered unrealistic for animals), where the direction (ψ_t) itself is random and independent of the previous direction (ψ_{t-1}).

Furthermore, we can perform the same test on the binary sequence of our guesses of the shark behavior λ_t. As the table shows, the p-values for this test are essentially zero for all sharks, except for shark 21, which had the fewest observations (70). Thus, the sequence of inferred behaviors are not random, and the sharks tend to remain in the foraging or transiting states rather than immediately switching between states. This is sensible if λ_t can reasonably represent a behavioral mode and if sharks forage continuously in one area for a long time before searching for prey elsewhere.

In our SSM, we use probability transition matrices to model the sharks' switching between the foraging and transiting behaviors ($\lambda_t = 0, 1$). A major goal of our paper is to see if sharks tend to forage more often in certain regions of the tidal basin than others. We will partition the tidal basin into several roughly equally-sized regions; to test this hypothesis, each region will have its own transition probability. To demonstrate that this regional model would be appropriate, we do the runs test on the sequence of behaviors λ_t separately for each shark and each region, for several different partition configurations. These tests confirmed that each shark's behavior type in each region (for regions that they visited for enough observations) was not a random sequence. This supports the use of a transition probability matrix (i.e., with sequential correlation) for each region, rather than, for instance, randomly selecting the behavior in each region, even with different foraging probabilities for each region.

CHAPTER 3

STATE-SPACE MODELS (SSMs), KALMAN (KF), AND EXTENDED KALMAN FILTERS (EKFs)

3.1 State-space models

Using the CRW paradigm of animal movement, we will model the overall movement of the sharks with a discrete-time state-space model (SSM). In an SSM, we have observed variables $\mathbf{y}$ that are assumed to be measured with error; these are sequentially modeled based on unobserved ('state') variables $\mathbf{x}$, which may include hypothesized 'true' values of the observed $\mathbf{y}$. The sequential evolution of $\mathbf{x}$ and $\mathbf{y}$ is modeled based on specified densities $\ell(\cdot)$ and $m(\cdot)$, which may also depend on a vector of parameters $\boldsymbol{\theta}$.

$$\begin{array}{rlcl} \text{Prior on initial state:} & \mathbf{x}_0 & \sim & p(\mathbf{x}_0) \\ \text{State/dynamic equation:} & \mathbf{x}_t & \sim & \ell(\mathbf{x}_t \mid \mathbf{x}_{t-1},\ \boldsymbol{\theta}_t) \\ \text{Measurement equation:} & \mathbf{y}_t & \sim & m(\mathbf{y}_t \mid \mathbf{x}_t,\ \boldsymbol{\theta}_t) \end{array} \tag{3.1}$$

Each of the equations has its mean given by corresponding functions $\boldsymbol{\ell}_\mu(\cdot)$

and $\boldsymbol{m}_\mu(\cdot)$, so we have

$$\begin{aligned} \text{State/dynamic equation:} \quad \mathrm{E}(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \quad & \boldsymbol{\ell}_\mu(\mathbf{x}_{t-1}) \\ \text{Measurement equation:} \quad \mathrm{E}(\mathbf{y}_t \mid \mathbf{x}_t) = \quad & \boldsymbol{m}_\mu(\mathbf{x}_t) \end{aligned}$$

The purpose of an SSM is to extract the signal (error-free unobserved variables $\mathbf{x}_t$) from the noisy (measured with error) observations $\mathbf{y}_t$. In the simplest formulation, the exact distributions of the state and measurement equations densities $\ell(\cdot)$ and $m(\cdot)$ as well as their mean functions $\boldsymbol{\ell}_\mu(\cdot)$ and $\boldsymbol{m}_\mu(\cdot)$, are known, and thus no parameters need to be estimated. Usually, however, an SSM depends on a vector of parameters $\boldsymbol{\theta}_t$ whose values and distributions we want to sequentially estimate by the SSM. values. In our model, we are particularly interested in modeling the distributions of the movement parameters (speed v_t and turn angle θ_t) and the behavior type λ_t, since this is the focus of our ecological questions of interest.

3.2 Correlated random walk (CRW) theory

As mentioned earlier, a correlated random walk (CRW) is a model commonly used for animal movement, where the turn angle θ_t at each step is random, centered at 0, and uncorrelated with the previous turn θ_{t-1}. The animal's direction of movement (ψ_t) at a given time depends on the cumulative sequence of turns; at a given point, if the angle is small in magnitude ($\theta_t \approx 0$), the animal does not change its direction very much. As discussed in Section 2.4, we use a Bézier spline interpolation to generate estimated locations that are regularly-spaced in time from the irregularly-occurring observations, because CRWs cannot generally be formulated properly if the time steps are irregular. We also emphasize that the spline interpolation is a reasonable approximation of the ground truth unobserved locations, as long as the time gap Δ_Υ is small enough, and that we use this interpolation only to characterize the data and the dual-behavior model, and not as inputs or classification labels of the behaviors by which to judge our algorithm's performance.

Several statistical tests are employed in the modeling literature to test the appropriateness of the CRW assumption. One is the Wald-Wolfowitz runs test for randomness (see Zar, [55] p. 416–419; Brownlee, [7] p. 224–231). A key assumption of the CRW is that the turns $\{\theta_t\}$ themselves are serially uncorrelated, and thus so is the turn direction (left/right). In general, the runs test is used on a sequence of binary-valued observations (of which left/right is an example) to see if like values tend to be grouped consecutively in the sequence, forming 'runs' of like values. Consider two sequences of binary values A,B

1. **AAAAA**BBBBBBB**AAAA**B

2. **AA**BBB**AA**BB**A**BB**AA**B**AA**

Note that both sequences are of length 17 and have 9 A's and 8 B's. A run is an uninterrupted sequence of a given binary value. For instance, in the first sequence, the runs are AAAAA, BBBBBBB, AAAA, and B, which have lengths 5, 7, 5, and 1, respectively; here there is a tendency for the A's and B's to group. The second sequence has runs of lengths 2, 3, 2, 2, 1, 2, 2, 1, and 2, so the like values tend not to form runs of significant length. The runs test determines if the binary values tend to appear randomly in the sequence in proportion to their overall share of the sequence. Thus, the sequence need not be balanced in the proportions of the binary values, as long as, say, the less frequent values do not group together more than is likely by random chance.

In the runs test, the null hypothesis H_0 is serial randomness of the binary sequence, while the alternative is non-randomness. As mentioned by Zar ([55]), randomness may be violated either if (1) there are fewer observed runs than would occur at random (more grouping), a quality called 'contagious', or (2) there are more observed runs than would occur at random, meaning the binary values tend to be distributed uniformly, in proportion to their overall share. The contagion alternative, under which there are fewer than expected runs under randomness, seems to be a more significant violation to the CRW model, so we conduct a one-sided lower-tailed rather than two-sided hypothesis test. In this and the following tests, the test is done separately by shark. Also, because the regular steps were interpolated with a limit of a maximum observed time gap Δ_t, the tests must be done piecewise on each uninterrupted sequence of variables. For instance, if we have two sequences of turns separated by a large time gap, we will not jointly consider the end of the first sequence and beginning of the second when determining runs, since we assume the beginning of the second sequence is unrelated to where the first ended. Rather, runs are determined piecewise on each sequence.[1]

As noted before by Gurarie ([19]), these tests need to be done on data observed at constant-length time intervals, which is why we created the cubic spline-interpolated regular steps. The piecewise runs test on each shark's spline interpolated trajectory gives near-zero p-values for all sharks, rejecting the non-randomness hypothesis in favor of contagion or runs of grouped left and right turns. The test statistics are highly negative, confirming our choice to conduct the lower-tailed hypothesis tests.

This result is not very surprising, however, since the very assumption of the spline interpolation causes the sequence regular steps along the same spline curve to curve in the same direction and thus have turn angles that are either all left or right. Figure 3.1 shows a toy example of a cubic Bézier spline of equal time-intervals fitted to seven observed control points. As the illustration shows, the cubic spline is fit piecewise to each group of four points and passes through the first and fourth points only. Here, the regular time interval Δ_Υ is shorter than the typical time gap Δ_t between observations, so there tend to be multiple constant-time intervals along each piecewise spline. If it is significantly shorter, as tends to be the case with the observed shark data, the spline interpolations tend to feature unbroken sequences of left (L) and right (R) turns due to the smoothness of the spline curve. Since we are using the

[1]The runs test is implemented by Caeiro and Mateus ([9]) in the `randtests` package for the `R` language. The `runs.test` function needed to be modified to make it combine the piecewise results.

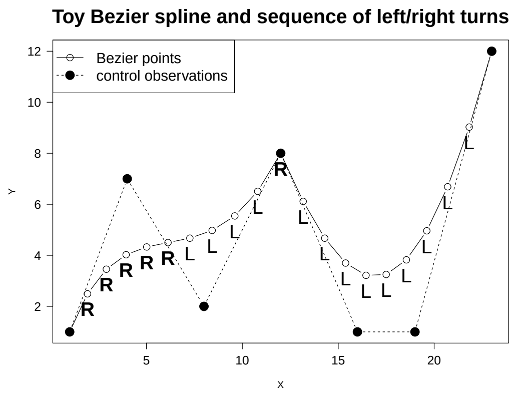


Figure 3.1: Toy Bézier spline interpolation with seven control points. “R” and “L” indicate if the turn is to the right or left at a given spline point. The smoothness of the spline causes longer sequences of left and right turns.

spline as an exploratory guide and not absolute ground truth, as we assume the unobserved regular step trajectory will not actually follow a smooth spline of the irregular observations, we do not take this test result as a rejection of the CRW model entirely.

Another aspect of the movement that statistical tests attempt to determine is whether a given trajectory exhibits directional persistence in movement (i.e., such as a CRW, where the animal tends to continue moving in the same general direction as long as the turn angles are small) or external bias or attraction towards a particular direction. Two common models that incorporate bias are the biased random walk (BRW), where an animal’s movement displays external influence in that the animal prefers to move towards a target, or the biased correlated random walk (BCRW), if there is bias in the direction but also some directional persistence or correlation in the bearing. Biased directional models tend to be more complicated because one generally has to determine what the direction of bias is and account for the possibility of the bias changing over time or being spatially dependent (see Codling, Plank, and Benhamou ([12]) for a thorough overview).

Several tests are proposed to detect directional bias. Among the most basic of these tests is the Δ_{MJ} statistic proposed by Marsh and Jones ([34]). Assuming constant-length time steps c, let

- $\Delta z_{1,c} = z_{1,c+1} - z_{1,c}$ and
- $\Delta z_{2,c} = z_{2,c+1} - z_{2,c}$

be the changes is the horizontal ($z_{1,\cdot}$) and vertical ($z_{2,\cdot}$) coordinate locations (here assuming $\boldsymbol{z}$ are the true, constant-interval measurements), respectively, from step c to $c+1$. The total (Euclidean) distance traveled is

$$l_c = \sqrt{\Delta z_{1,c}^2 + \Delta z_{2,c}^2}$$

The changes in the horizontal and vertical position can also be expressed as $\Delta z_{1,c} = l_c \cos(\psi_c)$ and $\Delta z_{2,c} = l_c \sin(\psi_c)$, in terms of the bearing angle $-\pi \le \psi_c \le \pi$.[2] The net displacement R_ψ, the total net distance traveled after C steps due to directional persistence of the animal, is thus

$$R_\psi = \sum_{c=0}^{C-1} \sqrt{(l_c \cos(\psi_c))^2 + (l_c \sin(\psi_c))^2}$$

A similar quantity, R_θ, captures the net displacement after C steps due to turning angles (there are $C-1$ of them between locations), and thus the effect on displacement due to external bias:

$$R_\theta = \sum_{c=1}^{C-1} \sqrt{(l_c \cos(\theta_c))^2 + (l_c \sin(\theta_c))^2}$$

Marsh and Jones' test involves the distribution of the net squared displacement $R_C^2 = R_\psi^2 - R_\theta^2$ and calculating its mean. On any given dataset, the statistic $\Delta_{\text{MJ}} = \overline{R_C^2}$ is the observed sample mean of R_C^2 (factoring out the distances ℓ_c,

[2]Marsh and Jones' original paper uses θ to stand for the direction, which we call ψ, and ζ to stand for the turn angle, which we call θ, so the discussion above is reformulated in our notation.

which are the same in each sum), namely

$$\Delta_{\mathrm{MJ}} = \frac{1}{C^2}\left(\left(\sum_{c=0}^{C-1}\cos\left(\psi_c\right)\right)^2 + \left(\sum_{c=0}^{C-1}\sin\left(\psi_c\right)\right)^2\right) + \frac{1}{(C-1)^2}\left(\left(\sum_{c=1}^{C-1}\cos\left(\theta_c\right)\right)^2 + \left(\sum_{c=1}^{C-1}\sin\left(\theta_c\right)\right)^2\right)$$

If $\Delta_{\mathrm{MJ}} > 0$, external bias, as evidenced from the turn angles, has a higher effect on the animal's displacement than does the animal's directional persistence, as evidenced by the bearing angles. If, on the other hand, $\Delta_{\mathrm{MJ}} < 0$, the dataset indicates that on average the directional persistence is the stronger effect, which indicates a CRW may be more appropriate than a biased random walk (BRW). Recall that Δ_{MJ} is simply the sample mean of a realization of a theoretic distribution of net squared displacement R_C^2, so the distributions are usually approximated by simulation from a model. As shown in in Table 3.1, the Δ_{MJ} statistic, calculated on the spline regular step approximation, is negative for all sharks, indicating a CRW is the more appropriate model, and hence we will not incorporate a directional bias term.

To illustrate the population distribution of Δ_{MJ}, we take repeated samples (without replacement) from the spline steps and calculate Δ_{MJ} on each subsample, and estimate the combined density of the sample statistics. This is shown in Figure 3.2, where we see the estimated densities are firmly over the negative values, not just that Δ_{MJ} calculated on the shark's entire trajectory are all negative, as shown in Table 3.1. This further supports the results of the Marsh-Jones statistic that the true shark trajectories are better modeled with a CRW than as a biased movement model.

The Δ_{MJ} statistic does have a few drawbacks. For instance, as Turchin ([53], p. 164) notes, Marsh and Jones' test only distinguishes between pure directional persistence (CRW) and pure external bias (BRW), but not bias with persistence (BCRW), which may be a valid model. One of the other tests he mentions, which do allow for the alternative hypothesis to specify a BCRW requires the horizontal axis to be aligned with the bias, whereas we

Shark num.	spline obs.	Δ_{MJ}	Shark num.	spline obs.	Δ_{MJ}
1	8,361	-0.582	12	907	-0.734
2	5,629	-0.691	13	1,672	-0.711
3	4,707	-0.561	14	781	-0.814
4	3,114	-0.476	15	1,181	-0.756
5	25,728	-0.521	16	1,271	-0.759
6	3,950	-0.735	17	168	-0.581
7	7,553	-0.498	18	2,649	-0.703
8	2,231	-0.621	19	27,629	-0.545
9	2,371	-0.593	20	266	-0.806
10	1,426	-0.712	21	100	-0.795
11	3,771	-0.606	22	420	-0.705

Table 3.1: Marsh and Jones' Δ_{MJ} statistic on the spline interpolated regular steps, with the number of spline observations and value of the Δ_{MJ} statistic. In each case, the statistic is negative, favoring a CRW over biased models.

may not know a priori the origin of the bias. Benhamou ([5]) mentions that the variances of the mean net squared displacements $\overline{R_C^2}$ in Marsh and Jones' test can be high if the number of trajectory steps is low, making it difficult to distinguish between the model types. He proposes an alternative test that does not restrict the bias direction but seems to require a cumbersome procedure of individually testing various potential bias directions over various time spans.

3.3 Review of animal telemetry literature

Improvements in telemetry technology have enabled gathering of movement data on animals of various size, even small fish and insects like butterflies, and thus development of models for the movement patterns. The CRW and other derivatives feature prominently in this literature. As McClintock et al. ([35]) noted in their 2012 paper, the development of theory for animal

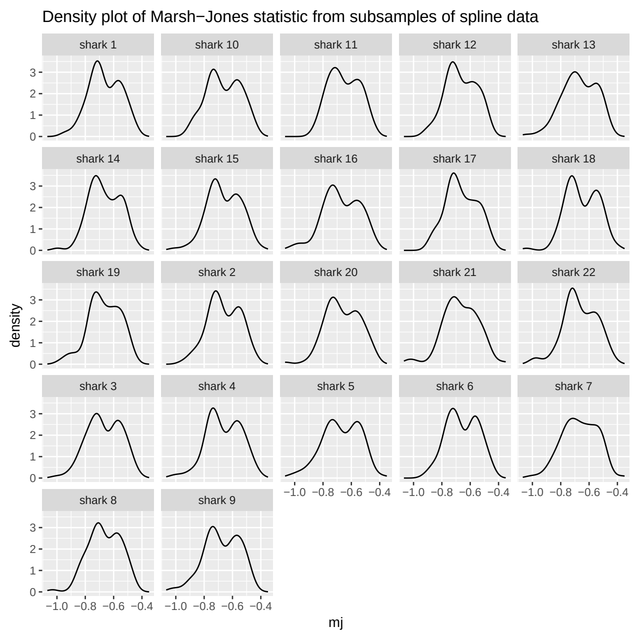


Figure 3.2: Density plot of Δ_{MJ} statistics from subsamples of the spline-interpolated data.

movement models has lagged behind the technological progress in development of telemetry devices. Particularly given the relatively recent technological ability to track diverse kinds of animals, many different approaches have been proposed and there is not yet firm scholarly consensus on the best ones, as there is in other biological applications. For instance, as they mention, models have been proposed in each combination of discrete or continuous time/space formulations.

Since we attempt to describe the different behavioral modes of the sharks, we focus on reviewing scholarly work on models with this feature. Skalski and Gilliam ([46]) developed an early movement model for animals with two behavior modes, though it was based on differential equations rather than state-space models. The use of SSMs and the Bayesian approach to modeling animal (here, marine turtles) movement trajectories specifically was advocated by Jonsen, Myers, and Flemming ([23]) to separately account for model error due to measurement error of positions and inherent variability in the movement process. Morales et al. ([36]) proposed a several versions of SSMs with behavioral switching probabilities based on the animal's habitat, where the trajectory was a mixture of two or more CRWs, each representing a behavior. This paper tracked four elk whose locations measured once a day were used to infer the animals' turn angles and speeds, which seems to be a weakness given the infrequency of the data.

Jonsen, Flemming, and Myers ([22]) also used a behavior-switching SSM to model marine movements of seals. They introduce the idea of predicting the values of the (irregularly) observed locations $\mathbf{y}$ by modeling the unobserved state locations $\mathbf{x}$ as occurring at fixed time intervals, and then interpolating to estimate $\mathbf{y}$. Many of the models mentioned here are fit using off-the-shelf programs like JAGS or WinBUGS, which uses Gibbs sampling to fit data with user-specified Bayesian priors. One weakness is that this tool appears to allow limited customization, unlike our models which are hand-coded.

Beyer, Morales, Murray, and Fortin ([6]) are one example of using a 'patch' model, where each region or 'patch' represents a different habitat, and behavioral probabilities differ in each type of habitat. For the habitat-specific behavior transition probabilities, they use a Poisson-like model where the probability of transitioning out of a state decreases explicitly the longer the animal has stayed in its current state; this tends to decrease the frequency of behavioral changes. An important aim in animal movement modeling is to quantify the animal's home range, or main area of habitat, to reflect the fact that animals do not tend to wander aimlessly.

Roever, Beyer, Chase, and van Aarde ([43]) likewise use a behavior-switching SSM with food sources distributed in different patches. Their model incorporates geographic covariates, such as distance to water or presence of vegetation, to inform their identification of the behavior. To spatially estimate the probability of foraging, they use a kernel density estimation based on the modeled locations. We would like to do so in a more statistical way with interpretable parameters.

An interesting approach to modeling the animal's home range is given in

Van Moorter et al. ([54]), who model the animal's home range by initializing patches (regions) that serve to provide utility (e.g., food) to the animal. When an animal spends time in a patch, it decreases the available utility (e.g., by eating the food); if it depletes the utility, it will move away from the patch, which in the meantime will replenish its utility (e.g., prey returns, food supply regenerates). On the other hand, animals tend to remember patches previously visited and will tend to return there; this is the main idea behind the concept of home range. The animal's movement between patches is balanced between being attracted to visited patches and being repelled by newly empty patches. This is a mechanistic model because these explicit instincts in the animal's behavior are modeled. We are not aiming to make such explicit biological assumptions about the sharks' behavior choices.

As mentioned, our approach will be to build an SSM to model the locations and behavior at regular steps $\{\Upsilon_c\}$ (of fixed length Δ_Υ seconds) using the irregular time observations $\{H_t\}$. This is done mainly for proper updating of movement parameters, which generally cannot be done well directly if the observations are irregular. Several other techniques besides interpolation are commonly used to modify data collected at irregular time gaps for use in building models. For instance, random walk models built off of regular intervals generally assume some serial statistical independence, such as that the distribution of the speeds at each time are statistically independent. If the data are collected non-uniformly in time, particularly if there are many observations within short time intervals of length Δ_Υ, the distributions may violate the assumption of serial non-correlation. The data may be sub-sampled to obtain a set of observations that are at roughly equal time intervals that are large enough so they can be assumed to be statistically independent. Alternatively, data may be aggregated to a lower time resolution of equal but longer time intervals, for instance by setting the speed at the interval as the average of the speeds of the observations within it.

Criticism of interpolation or discretization of irregular observations to regular time intervals is often focused on the fact that these techniques distort the true nature of the animal's movement, which occurs in continuous time. In our CRW model with regular time intervals, at the beginning of each interval, the animal 'chooses' its speed and turn angle, which dictate the movement path along a straight line for the duration of the interval. However, as Turchin remarks, this representation is problematic because "artificial 'moves' equated to steps do not correspond to actual behavioral events" ([53], p. 129). This means that since we are modeling a continuous path by discrete segments, then we should limit the degree to which we take the predicted discrete path as representing the animal's true behavior. For instance, if an animal's true trajectory approximately follows a straight line (i.e., with no turns) over several intervals, which we model piecewise as consisting of several straight lines with potential turns in between, when the animal's only real turning 'decision' was at the beginning of the straight movement, then we have created several artificial decisions that do not correspond to the true behavior. Depending on the time length of the animal's typical (if there is such a thing) time interval of step, our modeled regular time intervals may be too short, as in the previous example, or too long, in which case we have aggregated several short distinct behavioral decisions into one. Though we choose to use the CRW despite these shortcomings, this does motivate our choice to have a low variance on the distribution of turn angles to avoid aggressive zig-zag movement patterns, which is likely unrealistic.

Another criticism is that the observed data should be instead modeled as a discrete-time realization of the continuous movement process, rather than discretized to regular intervals; this avoids the need for techniques such as subsampling and aggregative. For instance, Johnson et al. ([21]) adapt the CRW model to a continuous-time[3] correlated random walk (CTCRW), using a model called an Ornstein–Uhlenbeck (OU) process. The process is a continuous-time autoregressive process on the locations, as done in Jonsen, Flemming, and Myers ([22]). Thus, the data that is being modeled are actually the changes in location coordinates (in our notation, $\Delta \boldsymbol{z}_t$) by modeling the horizontal and vertical axis velocities (i.e., rates of change $\Delta z_{1,\cdot}$ and $\Delta z_{2,\cdot}$ over time), rather than the actual location of the animal. Though this approach is common in many tracking applications, we believe that our model, which focuses on the animal's turning and angular speed, rather than the speed in each direction, is more appropriate for describing the animal's behavior process. For instance, the modeled speeds along each axis would not be invariant to a rotation of axes. Nevertheless, we acknowledge the potential advantages of a continuous-time model, and suggest it as a future research avenue.

McClintock et al. ([35]) present an attractive discrete-time model that extends Morales' two-behavior switching model by adding several features. Firstly, they adopt Jonsen's interpolation approach to model regular time intervals from irregular observations, and also allow for more than two behavior modes. Their model also allows for partial directional bias by specifying several coordinates of attraction for the animal; such points could represent foraging areas or other landmarks, for instance. When the animal is at a far distance from the attraction point, the animal travels at a roughly straight bearing angle to the point, but when it nears the attraction point, the turn angles are modeled to grow. This models the animal's tendency to travel to reach a foraging spot and then forage around in the surrounding area (at large turn angles).

One important feature is that they allow the locations of these attraction points to be learned algorithmically. The user pre-specifies how many attraction points there should be (usually only three or so), and initial coordinates of the attraction points are selected by a discrete uniform prior on the observed locations, since intuitively the animal should be observed visiting such attraction points sometime, if they exist. At each iteration of their algorithm,

[3]Durbin and Koopman ([15], p. 57–60) illustrate how the Kalman filter (discussed below) can be adapted to treat the irregularly-spaced observations as occurring in continuous time, allowing one to predict the state variable values ($\mathbf{x}(t)$, see Equation 3.2) at specified times t^* that were not observed. Essentially, the observation at t^* is treated as missing and then estimated by the Kalman filter and smoother by maximum likelihood (ML), conditional on the full set of observations. Presumably, if one wanted to use a particle filter approach, as we do, such estimation would only use the past observed values, rather than the future ones as well.

a Metropolis-Hastings step proposes to jitter the coordinates a bit (by selecting observed coordinates within a specific radius) and the algorithm randomly chooses to accept or reject the change. This is similar to Heikkinen and Arjas' ([20]) algorithm to iteratively estimate spatial density (mentioned later in Section 6). We consider this a potential improvement to our model in the future. McClintock et al. also address the enthusiasm of Johnson and others in the continuous-time Ornstein–Uhlenbeck processes, arguing that though these continuous models may be mathematically attractive, their parameters, which model instantaneous processes, are difficult to interpret biologically and may be "prohibitively technical for many non-statisticians."

3.4 Kalman filter (KF)

One of the simplest versions of an SSM is the Kalman filter (KF), where the state and measurement equations are linear functions of their inputs; these relations are allowed to change over time. The error covariances of these relations are usually Gaussian, making the densities $\ell(\cdot)$ and $m(\cdot)$ Gaussian, for which *exact* well-known recursive updates for the density parameters can thus be used. A Kalman filter setup is shown in Equation 3.2, where matrices $\mathbf{L}_t$ and $\mathbf{M}_t$, specifying linear relationships, can depend on time t[4].

$$\begin{array}{rrlclrl} \text{State/dynamic equation:} & \mathbf{x}_t = & \mathbf{L}_t\mathbf{x}_{t-1} & + & \mathbf{q}_t; & \mathbf{q}_t & \sim \mathcal{N}(\mathbf{0},\ \mathbf{Q}_t) \\ \text{Measurement equation:} & \mathbf{y}_t = & \mathbf{M}_t\mathbf{x}_t & + & \mathbf{r}_t; & \mathbf{r}_t & \sim \mathcal{N}(\mathbf{0},\ \mathbf{R}_t) \end{array} \tag{3.2}$$

[4]An important note here is that the parameters ($\mathbf{L}, \mathbf{M}$, etc.) in the KF's state and measurement equations are indexed to the same t. Some authors (for instance [45], p. 56) have the state equation indexed to $t-1$, for instance $\mathbf{x}_t = \mathbf{L}_{t-1}\mathbf{x}_{t-1} + \mathbf{q}_{t-1}$ rather than $\mathbf{L}_t$ and $\mathbf{q}_t$. Technically, the indexing is an arbitrary relabeling. However, in the model we will ultimately adapt (CDLM, see Equation 4.3), there is conditioning on a latent λ_t that needs to be marginalized over, and hence it is crucial to have the same index t for both the state and measurement equations. Thus, for consistency, we have used this form in all of our examples and have modified the Kalman filter update equations as given in [45] and other references, to match.

As a simple illustration of a Kalman filter, consider the case of a robot moving in one-dimensional space along the real line. At time t, its true (unobserved) location is $\zeta_t \in \mathbb{R}$; its sensor returns (with error) position measurement z_t. Assume, for simplicity, that the time difference between measurements is a constant value Δ_Υ. The robot moves from position ζ_t to ζ_{t+1} with random velocity $v_t \in \mathbb{R}$. Note that here v_t is velocity and that a negative velocity means the robot moves backwards, whereas in modeling sharks we use $v_t > 0$ to be the speed, since direction is accounted for by turn angles. Let

$\mathbf{x}_t = \begin{bmatrix}\zeta_t & v_t\end{bmatrix}^T$ be the unobserved true position and velocity at time t, and

$\mathbf{y}_t = \begin{bmatrix}z_{t+1}\end{bmatrix}$ be the robot's observed position at the next time $t+1$

A simple Kalman filter SSM to describe the relationships is given by

$$\begin{aligned}
\mathbf{x}_t = \begin{bmatrix}\zeta_t \\ v_t\end{bmatrix} &= \begin{bmatrix}1 & \Delta_\Upsilon \\ 0 & 1\end{bmatrix}\begin{bmatrix}\zeta_{t-1} \\ v_{t-1}\end{bmatrix} + \mathbf{q}_t;\ \ \mathbf{q}_t \sim \mathcal{N}_2(0,\Sigma). \\
\mathbf{y}_t = \begin{bmatrix}z_{t+1}\end{bmatrix} &= \begin{bmatrix}1 & \Delta_\Upsilon\end{bmatrix}\begin{bmatrix}\zeta_t \\ v_t\end{bmatrix} + \mathbf{r}_t;\ \ \mathbf{r}_t \sim \mathcal{N}_2(0,\sigma^2).
\end{aligned} \tag{3.3}$$

giving, the scalar equations

$$\begin{aligned}
\zeta_t &= \zeta_{t-1} + \Delta_\Upsilon v_{t-1} + \mathbf{q}_t[1] \\
v_t &= v_{t-1} + \mathbf{q}_t[2] \\
z_{t+1} &= \zeta_t + \Delta_\Upsilon v_t + \mathbf{r}_t
\end{aligned}$$

In this setup, the next true location $\zeta_t = \zeta_{t-1} + \Delta_\Upsilon v_{t-1}$, plus a random error $\mathbf{q}_t[1]$. The velocities at each time point are on average the same as the previous, plus random error ($v_t = v_{t-1} + \mathbf{q}_t[2]$).

3.5 Extended Kalman filter (EKF)

The Extended Kalman filter (EKF) is an approximation of the KF when the dynamic or measurement equations are nonlinear functions. The approximation uses a first-order Taylor expansion of the functions' Jacobians (matrices of first-order partial derivatives). The Jacobians themselves are thus matrices of functions of vector $\mathbf{x}$'s mean vector, $\boldsymbol{\mu}$, rather than a constant matrix (that may change over time). This linearizes the filter so the linear KF recursive formulas for posterior distribution updates can be used. Justification for the linear approximation, by using the multivariate Delta Method, is as follows:

Let $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$, and $\mathbf{f}(\cdot)$ be a vector-valued function with

Jacobian matrix $\mathbf{F}(\mathbf{x}) = \left[\mathbf{F}_{i,j}\right] = \left[\dfrac{\partial \mathbf{f}_i(\mathbf{x})}{\partial \mathbf{x}_j}\right]$, where $\mathbf{f}_{(}\cdot)i$ and $\mathbf{x}_j$ are, respectively, the i^{th} element of $\mathbf{f}(\cdot)$ and the j^{th} element of the vector $\mathbf{x}$. By the multivariate Delta Method, $\mathbf{f}(\boldsymbol{z}) \overset{d}{\rightsquigarrow} \mathcal{N}\left(\mathbf{f}(\boldsymbol{\mu}),\ \mathbf{F}(\boldsymbol{\mu})\,\Sigma\,\mathbf{F}(\boldsymbol{\mu})^T\right).$ (3.4)

The 1-D linear robot KF model as given above in Equation 3.3 has fixed values for all parameters (such as the normal errors) and hence cannot model unknown parameters. The following illustration will give a simplified example as to how the shark modeling works. Assume that for the robot, the only unknown parameter of interest is the mean velocity, which we call α, and that the velocities have the common distribution $v_t \sim \mathcal{N}(\alpha, 1)$, assuming for simplicity that the variance is known. By modeling the unknown true locations $\boldsymbol{\zeta}$ by knowledge of the observed locations $\boldsymbol{z}$, we can model the distribution of velocity, and hence learn α. To do this, we have to tweak the KF formulation in Equation 3.3 to have v_t depend on α rather than its previous value v_{t-1}. Thus, we need to reformulate the state equation for $\mathbf{x}_t \mid \mathbf{x}_{t-1}$ to be a function $\boldsymbol{\ell}_\mu(\cdot)$, and thus have density ℓ (see Equation 3.1) rather than matrix multiplication, so that it is an EKF with linear functions:

$$\begin{aligned}
\mathbf{x}_t &= \begin{bmatrix} \zeta_t \\ v_t \end{bmatrix} = \boldsymbol{\ell}_\mu(\mathbf{x}_{t-1}) + \mathbf{q}_t = \begin{bmatrix} \zeta_{t-1} + \Delta_\Upsilon v_{t-1} \\ \alpha \end{bmatrix} + \mathbf{q}_t \\
\mathbf{y}_t &= \begin{bmatrix} z_{t+1} \end{bmatrix} = \mathbf{m}_\mu(\mathbf{x}_t) + \mathbf{r}_t = \begin{bmatrix} \zeta_t + \Delta_\Upsilon v_t \end{bmatrix} + \mathbf{r}_t
\end{aligned} \tag{3.5}$$

We would thus set the covariance matrix Σ of the error $\mathbf{q}_t$ so that $\Sigma_{2,2} = 1$. The Jacobian matrices $\mathbf{L}(\cdot)$ and $\mathbf{M}(\cdot)$ of the robot EKF functions $\boldsymbol{\ell}_\mu(\cdot)$ and $\mathbf{m}_\mu(\cdot)$ are:

$$\mathbf{L}(\mathbf{x}_{t-1}) = \begin{bmatrix} 1 & \Delta_\Upsilon \\ 0 & 0 \end{bmatrix} \text{ and } \mathbf{M}(\mathbf{x}_{t-1}) = \begin{bmatrix} 1 & \Delta_\Upsilon \end{bmatrix}$$

The second row of $\mathbf{L}(\cdot)$ are both zero since these are the partial derivatives $\frac{\partial \alpha}{\partial \zeta_{t-1}}$ and $\frac{\partial \alpha}{\partial v_{t-1}}$, which are both zero since α is a constant. Note that in this particular case, the Jacobians end up being constant matrices (since Δ_Υ is a constant as well) rather than functions of the states $\mathbf{x}_t$.

CHAPTER 4

PARTICLE FILTERS (PFs) AND RESAMPLING

4.1 Introduction to Bayesian statistics

Before discussing filters in more mathematical detail, we will review the basic concept of Bayesian statistics. Generally, statistical modeling involves formulating a mathematical model for variables, which may be observed or not (latent), and often represent some practical quantity of interest, such as speed of ocean currents or the price of a share of stock over time, as depending on the values of underlying parameters. These mathematical models are in the form of statistical distributions.

As an example, let y be a univariate random variable modeled as having some distribution $\mathrm{y} \sim \ell(\mathrm{y} \mid \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is (possibly a vector) parameter. Let $\mathbf{y}_{1:n}$ be the set of observed values of y indexed $1, \ldots, n$. We will later take n, replaced by t, as representing time, and hence the values have a particular ordered sequence governed by time. In general, the observations may simply be a random unordered sample of the variable.

In a standard parametric statistical approach, we may assume we know the form of the distribution $\ell(\cdot)$ (for instance Gaussian, Poisson, etc.) and try to estimate the value of the parameter $\boldsymbol{\theta}$ given the observed data $\mathbf{y}_{1:n}$

accordingly, for instance by maximum likelihood (ML) techniques. In the Bayesian approach, we do not simply estimate the parameter values given the data *only*, but rather will assume we have some prior knowledge about the likely values of the parameter $\boldsymbol{\theta}$. This philosophy has the advantage of being more realistic in many scenarios where we may have external information about the value of something.

For instance, say y represented the height in feet of an adult male, and we had a random sample $\mathbf{y}_{1:n}$ of n adults' heights from some population. We may, for instance, assume that $\ell(\cdot)$ is the Gaussian distribution, so the population distribution is $\mathrm{y} \sim \mathcal{N}(\alpha, 1)$ (where here $\boldsymbol{\theta}$ is just $0 < \alpha$), and we seek to estimate α, the population mean height. A traditional ML approach would simply estimate α by the data, taking its sample mean $\bar{\mathrm{y}}$. In a Bayesian approach, we might first say that we have some idea from previous experience of what α might be. For instance, it is very unlikely that α is either 1 or 10 feet because these would be unrealistic given our knowledge of adult males in the world. Rather, we express our prior knowledge of likely values of α in a **prior distribution**. In general, for a parameter $\boldsymbol{\theta}$ (here, α), the prior takes the form

$$\boldsymbol{\theta} \sim \underbrace{p(\boldsymbol{\theta} \mid \boldsymbol{\Omega}_0)}_{\text{prior}}$$

where $\boldsymbol{\theta}$ is modeled as having its own distribution with hyper-parameters (i.e. parameters of a parameter's distribution) $\boldsymbol{\Omega}$, which may also be a vector. For instance, we may assign the prior $\alpha = \boldsymbol{\theta} \sim \mathcal{N}(\mu, \sigma^2)$, where $\boldsymbol{\Omega} = (\mu, \sigma^2)$. We may reasonably take $\mu = 5.5$ feet for instance, based on our real world experience. In general, assigning a prior will bias the estimated values of the parameter $\boldsymbol{\theta}$ toward values predicted by its prior, but the approach also allows one to specify non-informative priors which do not have this feature, but still provide other advantages within the Bayesian framework.

We now return to the time-indexed case, so we replace the index n by the time index t. Once one has a prior distribution on the parameter and observed data that are assumed to depend on the parameter, in the Bayesian approach we learn more about the value of $\boldsymbol{\theta}$ based on the sample (i.e., $d(\boldsymbol{\theta} \mid \mathbf{y}_{1:t})$, where $d(\cdot)$ denotes an arbitrary distribution) by learning the likely values of the hyper-parameters $\boldsymbol{\Omega}$ of $\boldsymbol{\theta}$'s prior distribution. This happens in two steps:

1. Calculate the **marginalized likelihood** $m(\cdot)$ of $\mathbf{y}_{1:t}$ (over $\boldsymbol{\theta}$):

$$\underset{\text{marginal}}{m(\mathbf{y}_{1:t} \mid \boldsymbol{\Omega}_0)} = \int \underset{\text{likelihood}}{\ell(\mathbf{y}_{1:t} \mid \boldsymbol{\theta})}\ \underset{\text{prior}}{p(\boldsymbol{\theta} \mid \boldsymbol{\Omega}_0)}\ \mathrm{d}\boldsymbol{\theta}$$

2. Update the **posterior distribution**, improving our knowledge of $\boldsymbol{\theta}$, combining the data $\mathbf{y}_{1:t}$ and prior parameters $\boldsymbol{\Omega}_0$ to update to new values $\boldsymbol{\Omega}_t$ of its parameters:

$$\underset{\text{posterior}}{p(\boldsymbol{\theta} \mid \boldsymbol{\Omega}_t)} = \frac{\ell(\mathbf{y}_{1:t} \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta} \mid \boldsymbol{\Omega}_0)}{m(\mathbf{y}_{1:t} \mid \boldsymbol{\Omega}_0)} = \frac{\textsf{likelihood} \times \textsf{prior}}{\textsf{marginal}}$$

3. Assuming we now encounter a new value y_{t+1} from the same population, we calculate the **predictive posterior** distribution, using the newly-constructed posterior as the prior on $\boldsymbol{\theta}$ to predict y_{t+1}:

$$\underset{\text{predictive posterior}}{d(\mathrm{y}_{t+1} \mid \boldsymbol{\Omega}_t)} = \int \underset{\text{likelihood}}{\ell(\mathrm{y}_{t+1} \mid \boldsymbol{\theta})}\ \underset{\text{prior}}{p(\boldsymbol{\theta} \mid \boldsymbol{\Omega}_t)}\, \mathrm{d}\boldsymbol{\theta}$$

Thus, the prior distribution of $\boldsymbol{\theta}$, through its hyper-parameters $\boldsymbol{\Omega}$, is continuously updated in the face of new data. This formulation is particularly attractive in cases called conjugate distributions, where the prior distribution p remains in the same form after being updated, since we may be interested in statistical inference on the values of posterior distribution to infer the likely values of $\boldsymbol{\theta}$. Eventually, the more data (t is bigger), the effect of the original prior values $p(\boldsymbol{\theta} \mid \boldsymbol{\Omega}_0)$ diminishes, and our knowledge of $\boldsymbol{\theta}$, in terms of its parameters $\boldsymbol{\Omega}_t$, is increasingly learned from the observed data $\mathbf{y}_{1:t}$.

4.2 Bayesian filtering

Although we have previously mentioned the concept of a filter, in the KF and EKF, we did not formally define it. A filter is a statistical method to recover a signal (i.e., the true values of some variable) from noisy measurements. Specifically, we aim to estimate the distribution of the unobserved, true $\mathbf{x}_t$ values from the noisy observed $\mathbf{y}_t$. This is done sequentially, at each time step, by alternating prediction and update operations. The state-space model (SSM) as formulated earlier in Section 3 described how a set of unobserved variables $\mathbf{x}_t$ evolve over time, and how the observations $\mathbf{y}_t$ might be a function of the underlying unobserved $\mathbf{x}_t$, but we did not demonstrate how statistical learning is achieved.

Using the Bayesian steps described above, a **Bayes filter** allows us to sequentially learn the unobserved signal values from successive noisy observations. Given the history of unobserved $\mathbf{x}$ and observed $\mathbf{y}$ up to $t-1$, we want to **predict** the distribution of the next $\mathbf{x}_t$:

$$p_{\text{predict}}(\mathbf{x}_t \mid \mathbf{y}_{1:(t-1)}) = \int \ell(\mathbf{x}_t \mid \mathbf{x}_{t-1})\, p_{\text{current}}(\mathbf{x}_{t-1} \mid \mathbf{y}_{1:(t-1)})\, \mathrm{d}\mathbf{x}_{t-1} \tag{4.1}$$

Now, given observation $\mathbf{y}_t$, we **update** the parameter estimates of $\mathbf{x}_t$. In our case, this amounts to finding the best movement parameters ($\mathbf{x}_t$) that predict the next observed location $\mathbf{y}_t = \boldsymbol{z}_{t+1}$.

$$p_{\text{current}}(\mathbf{x}_t \mid \mathbf{y}_{1:t}) \propto m_{\text{current}}(\mathbf{y}_t \mid \mathbf{x}_t)\, p_{\text{predict}}(\mathbf{x}_t \mid \mathbf{y}_{1:(t-1)})$$

We then continue to the next prediction step, estimating $p_{\text{predict}}(\mathbf{x}_{t+1} \mid \mathbf{y}_{1:t})$.

4.3 Introduction to particle filters (PFs)

Particle filters (PFs) are computational simulations of SSMs that have two general advantages over the simple KF and EKF approaches to SSMs. Firstly, they allow us to include a parameter vector $\boldsymbol{\theta}$ as part of the SSM formulation, which we can sequentially learn and perform Bayesian inference on. Secondly, they are more flexible because we are not limited to modeling Gaussian relations, as in the KF or EKF.

PFs use computational techniques to approximate the integration in the prediction and update steps of the Bayes filter (see Section 4.2) for an SSM when the densities are not necessarily Gaussian. This technique, called resampling, refers to replacing a random sample with a set of random draws with replacement from that sample. PFs simulate a SSM with N independent simulations of the process, each of which is called a 'particle'.

As an illustration, consider the following mixture Gaussian distribution, restricted to have support $20 \leq x \leq 29$:

$$m(x) = 0.6\mathcal{N}(x \mid \mu_1 = 22.5, \sigma_1 = 0.8) + 0.4\mathcal{N}(x \mid \mu_2 = 26, \sigma_2 = 1.15)$$

This bimodal distribution is shown as the black curve in Figure 4.2. Note that the density of m outside of the interval $[20, 29]$ is negligible, so we will ignore values of x outside this interval. Let $\{20, 21, 22, \ldots, 29\}$ represent a discretization of the distribution's domain, and assign each of $\mathbf{x}_0^{(n)}, n = 1, \ldots, N = 10$ to be one of these values, so $\mathbf{x}_0^{(1)} = 20, \ldots, \mathbf{x}_0^{(10)} = 29$. Each $\mathbf{x}_0^{(n)}$ is called a particle.

We wish to use this discretization to approximate a draw from the distribution m. Note that in this case we have specified a parametric form of the distribution, but this technique applies equally in more difficult cases where, for instance, we may not be able to directly draw from the desired density (such as through functions in `R` like `rnorm`), or where the distribution does not have a specific parametric form and may be known only through the densities (up to a constant factor) at particular discrete values in the domain. The fact that the densities can be known up to a constant factor means we do not need to compute the normalizing constant in the denominator when calculating the density, which is difficult or intractable in many applications. In this distribution, the density values of m at $\{\mathbf{x}_0^{(n)}\}_{n=1}^{10}$ are approximately {0.002, 0.052, 0.246, 0.251, 0.082, 0.097, 0.139, 0.095, 0.031, 0.005}, shown as the heights of the black dots in Figure 4.2. When normalized to have a sum of 1, we can denote these density values as representing weights $\{w_0^{(n)}\}_{n=1}^{10} \approx$ {0.002, 0.052, 0.247, 0.251, 0.082, 0.097, 0.139, 0.095, 0.031, 0.005}. In this case, the weight values are essentially unchanged after normalization, but this is not generally the case, particularly for higher N.

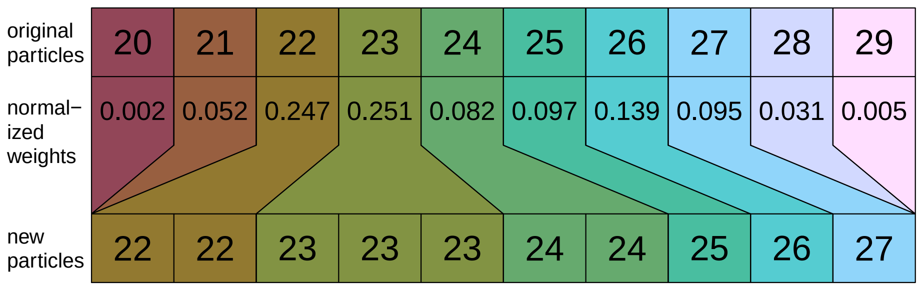


Figure 4.1: Discrete particle values $\{\mathbf{x}_0^{(n)}\}_{n=1}^{10} = \{20, 21, 22, \ldots, 29\}$ resampled with replacement by their weights from the density m. Weights do not add up to 1 due to rounding. The resampled set will approximate the density of the desired distribution m.

To approximate a random sample, say of size $N = 10$, from m, we use the original densitiy values, or equivalently, the normalized weights $\{w_0^{(n)}\}_{n=1}^{10}$, as weights when making N independent draws with replacement from the original set $\{\mathbf{x}_t^{(n)}\}_{n=1}^{N} = \{20, 21, 22, \ldots, 29\}$. Note that in this illustration, we have set the number of discretization points (i.e., particle values $\{\mathbf{x}_0^{(n)}\}$) $N = 10$ the same as the size of the desired sample. If all we want is a approximate random sample, then they do not have to be the same; in fact, we may want a larger sample size than the original points N, in order to better approximate the desired distribution. Here we set them the same because in a PF, the particles set $\{\mathbf{x}_t^{(n)}\}_{n=1}^{N}$ are sequentially replaced by an equal-size resampled draw $\{\tilde{\mathbf{x}}_t^{(n)}\}_{n=1}^{N}$.

Figure 4.1 shows one possible result of resampling with replacement from the original set $\{\mathbf{x}_0^{(n)}\}_{n=1}^{10} = \{20,\ 21,\ 22,\ \ldots,\ 29\}$. The resampled set is $\{\tilde{\mathbf{x}}_0^{(n)}\}_{n=1}^{10} = \{22,\ 22,\ 23,\ 23,\ 23,\ 24,\ 24,\ 25,\ 26,\ 27\}$, where the original values should appear roughly in proportion to their weights. For instance, low-weighted values 20 ($w_0^{(5)} = 0.002$) and 21 ($w_0^{(2)} = 0.052$) do not appear in the resampled set, but 22 ($w_0^{(3)} = 0.247$) and 23 ($w_0^{(4)} = 0.251$) appear two and three times each. Due to the inherent randomness in sampling, and in our case, the roughness of the discretization (we only have $N = 10$ particles) causes the resampling result to not be perfectly proportional in all cases. For instance, the value 24 ($w_0^{(5)} = 0.082$) was resampled twice, more than the higher-weighted value 26 ($w_0^{(7)} = 0.139$), which was resampled once; it is also resampled the same number of times as 22, whose weight (0.247) is more than three times its weight (0.082). Techniques that modify the resampling mechanism to reduce such variance in results are described in Section 4.7.

Figure 4.2 shows the density of the desired mixture Gaussian distribution m (see Section 4.3); the resampling weights $\{w_0^{(n)}\}_{n=1}^{10}$, scaled by the appropriate normalizing factor, lie along this curve. The red dashed line shows the density estimate of m based on only the resampled values $\{22, 22, 23, 23, 23, 24, 24, 25, 26, 27\}$. We see that this small sample roughly captures the bimodal aspect of m, but is unlikely to be very close to the true density due to the small size. The thin red line shows a density estimate based on resampling with replacement, but based on a finer discretization of the domain $\{\mathbf{x}_0^{(n)}\}$ to $N = 91$—rather than 10—points $\{20, 20.1, 20.2, \ldots, 28.9, 29\}$. In general, the discretization points need not be equally spaced. In fact, the sampling would improve if there was a higher concentration of discretization points $\mathbf{x}^{(n)}$ in areas of the domain with higher or more variable density to better capture the curvature.

Increasing the resolution of the domain discretization and the size of the sample will result in a more accurate approximation of the desired density, particularly if the density is multivariate. In our PF, this will be done by controlling N, the number of particles, but increasing N increases the total computation linearly because all calculations generally need to be repeated for each particle. For computational ease and timely simulations we will generally use $N = 100$ in our experiments.

By approximating a random sample from a desired density by resampling values by their weights (which are proportional to the density evaluated at that value), we can approximate an integration calculation that may be difficult to do in closed form. Say we have a desired moment $g(x)$, for instance the mean ($g(x) = x$) or variance ($g(x) = (x - \mathrm{E}(x))^2$), that we want to evaluate

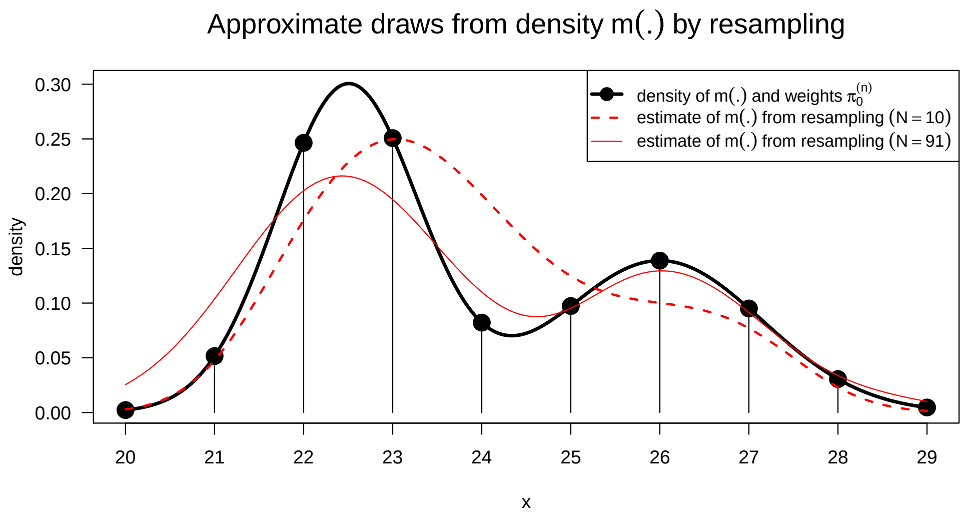


Figure 4.2: Comparison of the true density of the desired distribution (black) m with resamplings based on discretizations of the domain with $N = 10$ (red dashed line) and $N = 91$ (thin red solid line) of increasing resolution. The higher the resolution and resampling size, the closer the resampling approximates the desired density m.

for the distribution m. In closed-form integration, this calculation would be $\int g(x)\, m(x) \mathrm{d}x$. However, by using a discrete sum, we can approximate this as

$$\begin{aligned} \int \quad & g(x)\, m(x) \quad \mathrm{d}x \\ \approx \sum_{n=1}^{N} \quad & g\left(\tilde{\mathbf{x}}^{(n)}\right) \quad \frac{1}{N} \end{aligned}$$

where

- the sum is the discrete approximation of a continuous integral,
- using the particle values resampled by their weights $(\tilde{\mathbf{x}}^{(n)}, n = 1, \ldots, N)$ replaces the weighting of the function values $g(\cdot)$ at each of the density values $m(\cdot)$ for each value $x \in \text{domain}(m)$ in the product $g(x)m(x)$, and
- the fraction $\frac{1}{N}$, which has limit $\lim_{N\to\infty} \frac{1}{N} = 0$ as the number of particles N grows, takes the place of $\mathrm{d}x \approx 0$ which is the limit of the width of the 'rectangles' in Riemann summation. This term also rescales the sum by the number of particles N used.

We will see later that PFs perform particle learning by repeatedly updating particle estimates and resampling sets of particle samples to approximate their distribution given the observed values.

4.4 Particle filters and the Conditional Dynamic Linear Model (CDLM)

Particle filter formulations generally follow the basic SSM setup but differ in their resampling schemes and statistical assumptions as they try to achieve simulations with certain properties, such as specific independence structures (see Section 4.8). As shown earlier, the general form of a state-space model is shown in Equation 4.2, where unobserved states $\mathbf{x}_t$ evolve over time based on their previous value (state equation), and then the observations $\mathbf{y}_t$ are a function (measurement equation) of the unobserved values $\mathbf{x}_t$ at the same time t. For instance, the observed noisy variables $\mathbf{y}_t$ may have error-free corresponding variables be a subset of the states $\mathbf{x}_t$.

$$\begin{array}{rccl} \text{State/dynamic equation:} & \mathbf{x}_t & \sim & \ell(\mathbf{x}_t \mid \mathbf{x}_{t-1},\ \boldsymbol{\theta}_t) \\ \text{Measurement equation:} & \mathbf{y}_t & \sim & m(\mathbf{y}_t \mid \mathbf{x}_t,\ \boldsymbol{\theta}_t) \end{array} \tag{4.2}$$

Let $d(\cdot)$ denote an arbitrary density function. Particle filters are generally formulated to approximate the distribution $d(\mathbf{x}_T \mid \mathbf{y}_{1:T})$, known as the filtering density. That is, the distribution of the unknown states given the full set of observations $\mathbf{y}_t$ up to time T. This is achieved by sequential learning so that at time T, we have a set of N particle values $\{\mathbf{x}_T^{(n)}\}_{n=1}^N$, with final values present in the set roughly in proportion to their density values $d\left(\mathbf{x}_T^{(n)} \mid \mathbf{y}_{1:T}\right)$. At each step, the set of particles are usually resampled by weights $w_t^{(n)}$ calculated to achieve the desired integration.

The traditional form of the PF, also called the sequential importance resampling (SIR) algorithm (see [45], 124) **propagates** (draws sample values of $\mathbf{x}_t$ from their distributions and stores them in the sequence's 'history'), then resamples them according to their density value predicting the observations:

General outline of sequential importance sampling (SIR) PF algorithm:

```
1. For each particle $n = 1, \ldots, N$:
    (a) Initialize values $\mathbf{x}_0^{(n)} \sim p(\mathbf{x}_0)$ from the prior $p(\cdot)$.
2. End iteration over particle indices $n$.
3. For observations $t = 0, \ldots, T-1$:
    (a) For each particle $n = 1, \ldots, N$:
        i. Propagate state values $\mathbf{x}_{t+1}^{(n)} \sim \ell\left(\mathbf{x}_{t+1} \mid \mathbf{x}_t^{(n)},\ \mathbf{y}_{0:(t+1)}\right)$
```

```
      ii. Calculate resampling weight[1] $w_t^{(n)} \propto \frac{m\left(\mathbf{y}_{t+1} \mid \mathbf{x}_{t+1}^{(n)}\right)\ell\left(\mathbf{x}_{t+1}^{(n)} \mid \mathbf{x}_t^{(n)}\right)}{\ell\left(\mathbf{x}_{t+1}^{(n)} \mid \mathbf{x}_t^{(n)},\, \mathbf{y}_{1:T}\right)}$.
   (b) End iteration over particle indices $n$.
   (c) For each particle $n = 1, \ldots, N$:
      i. Resampling: Draw one index $j$ with replacement from the set $\{1, \ldots, N\}$, with probabilities $\{w_t^{(n)}\}_{n=1}^N$.
      ii. Set particle $\tilde{\mathbf{x}}_{t+1}^{(n)} = \mathbf{x}_{t+1}^{(j)}$.
   (d) End iteration over particle indices $n$.
   (e) Replace the set $\{\mathbf{x}_{t+1}^{(n)}\}_{n=1}^N$ with the resampled set $\{\tilde{\mathbf{x}}_{t+1}^{(n)}\}_{n=1}^N$.
4. End iteration over times $t$.
```

The important aspect of the SIR is that at each step, the next particle value $\mathbf{x}_{t+1}$ is propagated for each particle based on the state equation $\ell\left(\mathbf{x}_{t+1} \mid \mathbf{x}_t^{(n)}\right)$. Note that at this point, the set of propagated particles contains N unique values. Then, each propagated $\mathbf{x}_{t+1}^{(n)}$ has a weight $w_t^{(n)}$ reflecting the density of the measurement equation $m\left(\mathbf{y}_{t+1} \mid \mathbf{x}_{t+1}^{(n)}\right)$, that is, weighted by how likely that particle value at $t + 1$ 'predicts' the actual observed value $\mathbf{y}_{t+1}$. When the particles are resampled by these weights, the resulting set of values $\{\tilde{\mathbf{x}}_{t+1}^{(n)}\}_{n=1}^N$ have values present approximately in proportion to their density in the filtering density $(\mathbf{x}_{t+1} \mid \mathbf{y}_{0:(t+1)})$. Note that at this point, if any particles were resampled more than once, our set has fewer than N unique values. The resampling has thus conducted a discrete approximation of the continuous integration calculation in Equation 4.1. The approximation is used because such multivariate integration in closed form is often impossible except for the most simple cases of distributions, which are still complicated to calculate by hand.

[1]For simplicity, we assume particles are resampled at each step t. More generally, the resampling weights would be calculated by multiplying the factor shown above by the previous weight $w_{t-1}^{(n)}$, then calculating the effective size (see Section 4.6). In our illustration, the weights $w_{t-1}^{(n)}$ are each reset to 1 after they are used to resample, since now the particles are each equally weighted.

In addition to the fact that PFs can approximate filtering with non-Gaussian (or more generally, non-conjugate) densities by resampling particles, the resampling also PFs improve their predictions by acting as an ensemble. In the data mining and machine learning literature, ensembles, or collections of models, are often used in classification and prediction tasks, based on the recognition that combining multiple models' predictions often results in a better outcome than a single model's prediction, even if some of the models in the ensemble overall perform worse than the single model. For instance, the random forest algorithm is an ensemble method that creates a collection of decision trees, each of uses a subset of the variables and a subset of the observations to classify observations. The predicted class of a test observation is decided by majority vote of the trees in the forest; the multiplicity of trees is used so depending on the observation, some trees that have stronger accuracy can compensate for those that are weaker.

Similarly, in a PF, each of the particles to learn values of the parameters independently, meaning our PF consists of N individual models. The traditional Kalman filter is a *single* model, and it uses each observation to adjust the estimated density of the states $\mathbf{x}_t$; however, a single outlier, in which case the filter's prediction error is likely to be large, can impact the entire future performance of the filter by, for instance, making the values state covariance matrix too large. In a PF, if this happens to a given particle, that particle is less likely to be resampled in the future, so the effect of a single particle on the ensemble's overall predictive success or failure is small. In fact, particles that perform well are more likely to be resampled, hopefully improving, rather than worsening, the future performance.

The conditional dynamic linear model (CDLM), formulated by Carvalho et al. ([10]), is a particle filter formulation that introduces conditioning the SSM densities on a latent variable λ_t. For our case, λ_t represents the inferred behavioral class, allowing us to model shark movement separately for each of the foraging and transiting behaviors. The conditioning is done by modifying the generic KF formulation (Equation 3.2) by subscripting the linear relationship matrices $\mathbf{L}_t$ and $\mathbf{M}_t$ by λ_t, allowing them to depend both on the latent variable (behavior) and on time, as shown in Equation 4.3:

$$\begin{aligned} \text{State/dynamic equation:} \quad \mathbf{x}_t &\sim \mathcal{N}(\mathbf{L}_{\lambda_t}\mathbf{x}_{t-1},\ \mathbf{Q}_{\lambda_t} \mid \boldsymbol{\theta}_t) \\ \text{Measurement equation:} \quad \mathbf{y}_t &\sim \mathcal{N}(\mathbf{M}_{\lambda_t}\mathbf{x}_t,\ \mathbf{R}_{\lambda_t} \mid \boldsymbol{\theta}_t) \end{aligned} \tag{4.3}$$

CDLM in basic linear Kalman Filter form.

Here, as in the general SSM (Equation 4.2), $\boldsymbol{\theta}_t$ is an optional vector of latent, unobserved parameters. As discussed later, we will adapt this CDLM framework to be an EKF rather than a KF, in which case the linear relation matrices $\mathbf{L}_{\lambda_t}$ and $\mathbf{M}_{\lambda_t}$, as well as the noise covariance matrices are replaced by functions such $\boldsymbol{\ell}_\mu(\cdot)$ and $\boldsymbol{m}_\mu(\cdot)$ (as in Equation 3.5, the 1-D robot EKF). Rather than changing the functions themselves based on the value λ_t, the conditioning on λ_t will be done by having separate hyper-parameter values $\boldsymbol{\Omega}$ for the distributions of the parameter vector $\boldsymbol{\theta}_t$ for each value of λ_t.

For instance, in the 1-D robot example, consider a robot with two movement modes, fast ($\lambda_t = 0$) and slow ($\lambda_t = 1$), and velocity $v_t \sim \mathcal{N}(\alpha_t, 1)$; for simplicity, assume $\alpha_t > 0$ is high enough so that $\Pr(v_t < 0 \mid \alpha_t) \approx 0$. We can modify this to have $v_t \sim \mathcal{N}(\alpha_{\lambda_t}, 1)$ so that the velocity distribution depends, by definition, on the movement mode. For instance, we may have $\alpha_{\lambda_t=0} = 10$ miles per hour (mph) for the slow mode, and $\alpha_{\lambda_t=1} = 30$ mph, then use the respective value of α_{λ_t} when drawing random velocities v_t for each movement mode. Thus, our modification of the CDLM KF (Equation 4.3) to an EKF is the following:

$$\begin{aligned} \text{State/dynamic equation:} \quad \mathbf{x}_t &\sim \mathcal{N}(\boldsymbol{\ell}_\mu(\mathbf{x}_{t-1} \mid \boldsymbol{\theta}_{\lambda_t}),\ \mathbf{q}(\boldsymbol{\theta}_{\lambda_t})) \\ \text{Measurement equation:} \quad \mathbf{y}_t &\sim \mathcal{N}(\boldsymbol{m}_\mu(\mathbf{x}_t \mid \boldsymbol{\theta}_{\lambda_t}),\ \mathbf{r}(\boldsymbol{\theta}_{\lambda_t})) \end{aligned} \tag{4.4}$$

CDLM adapted to EKF form.

Note that in the above Equation 4.4, we have written the parameters conditioned on the behavior λ_t as $\boldsymbol{\theta}_{\lambda_t}$ for notational simplicity, when really we mean $\boldsymbol{\theta}_t \sim p(\cdot \mid \boldsymbol{\Omega}_{\lambda_t})$. That is, what depend on the latent λ_t are actually the hyper-parameters $\boldsymbol{\Omega}$ of the distributions p of the underlying parameters $\boldsymbol{\theta}$. For instance, for the robot, we may have a prior distribution on the velocity mean α_t such that $\alpha_t \sim \mathcal{N}(\boldsymbol{\Omega}_{\lambda_t})$, where $\boldsymbol{\Omega}_{\lambda_t} = (\mu_{\lambda_t}, \sigma_{\lambda_t})$. If α is the parameter (i.e., $\alpha_t = \boldsymbol{\theta}_t$) for the distribution of velocity v_t, then the distribution of v_t depends on λ_t through the hyper-parameters $\boldsymbol{\Omega}_{\lambda_t}$ of the distribution of velocity mean α_t. The goal of our particle filter will be to learn the optimal values of the hyper-parameters $\boldsymbol{\Omega}_{\lambda_t}$ for each behavior λ_t.

In addition to allowing conditioning on the latent λ_t, the CDLM addresses two main inefficiencies in the SIR formulation. These are:

- In the SIR, N unique values of $\mathbf{x}_{t+1}$ are propagated based on how they predict the observation $\mathbf{y}_{t+1}$ at the same time index. If resampled with replacement, these N values are replaced with a set of values, some of which are likely repeated. Even though they are resampled to approximate the filtering density, ideally we would want a sample of N unique values from this density. Instead, the CDLM uses the prediction density one step ahead to resample. Particles $\{\mathbf{x}_t^{(n)}\}_{n=1}^N$ are assigned a weight $w_t^{(n)}$ based on how well they predict $\mathbf{y}_{t+1}$, integrating over the intermediate $\mathbf{x}_{t+1}$; thus the resampling weights are different.
  Once $\{\mathbf{x}_t^{(n)}\}_{n=1}^N$ are replaced with the resampled set $\{\tilde{\mathbf{x}}_t^{(n)}\}_{n=1}^N$, and then stored, the values $\mathbf{x}_{t+1}^{(n)} \sim \ell\left(\mathbf{x}_{t+1} \mid \tilde{\mathbf{x}}_t^{(n)}\right)$ are propagated, thus giving N unique values. This modification reduces the problems of degeneracy and impoverishment (see Section 4.6) due to duplicated particles that reduce the diversity. It also improves efficiency by avoiding propagating values of $\mathbf{x}_{t+1}^{(n)}$ that may not be resampled and thus discarded.

- Like other PFs, the CDLM formulation will sequentially update the parameters of the state and measurement equation densities $\ell(\cdot)$ and $m(\cdot)$; in the linear multivariate Gaussian case, as in Equation 3.2, these are the matrices $\mathbf{L}_t$ and $\mathbf{M}_{t+1}$ and their associated error covariances. The CDLM formulation allows these parameters to be recursively updated using only the sufficient statistics (notation $\mathbf{S}_t^x$) of these parameters. Since PFs can potentially thousands or millions of particles or timesteps, it is important that the updates scale with time and that computer memory is not wasted, and thus the entire history of particle values does not need to stored, but only the sufficient statistics, which do not grow in dimension over time.

The general algorithm of the CDLM is illustrated below in Section 4.4. Here we illustrate the difference between it and the SIR in terms of the resampling scheme. The CDLM includes conditioning on the latent λ_t, which involves the transition probabilities between the behaviors λ_t. Here we omit this for simplicity. The main difference is that while the SIR propagates $\mathbf{x}_{t+1} \mid \mathbf{x}_t$ and then resamples the set $\{\mathbf{x}_{t+1}\}$, the CDLM first resamples the set $\{\mathbf{x}_t\}$, giving $\{\tilde{\mathbf{x}}_t\}$, and then propagates $\mathbf{x}_{t+1} \mid \tilde{\mathbf{x}}_t$ from these values.

General outline of conditional dynamic linear model (CDLM) algorithm:

---

1. For each particle $n = 1, \ldots, N$:
   (a) Initialize values $\mathbf{x}_0^{(n)} \sim p(\mathbf{x}_0)$ from the prior $p(\cdot)$.
2. End iteration over particle indices n.
3. For observations $t = 0, \ldots, T-1$:
   (a) For each particle $n = 1, \ldots, N$:
      i. Calculate parameters of marginal density $m_{\text{predict}}\left(\mathbf{y}_{t+1} \mid \mathbf{x}_t^{(n)}\right)$, which is proportional to $m(\mathbf{y}_{t+1} \mid \mathbf{x}_{t+1})\ell\left(\mathbf{x}_{t+1} \mid \mathbf{x}_t^{(n)}\right)$. Predict the observation $\mathbf{y}_{t+1}$ one step ahead of the current $\mathbf{x}_t^{(n)}$, marginalizing over the next potential intermediate $\mathbf{x}_{t+1}$.
      ii. Calculate resampling weight $w_t^{(n)} \propto m_{\text{predict}}\left(\mathbf{y} = \mathbf{y}_{t+1} \mid \mathbf{x}_t^{(n)}\right)$ by evaluating the predictive density at the observed value.

(b) End iteration over particle indices n.

(c) For each particle $n = 1, \ldots, N$:

i. Resampling: Draw one index j with replacement from the set $\{1, \ldots, N\}$, with probabilities $\{w_t^{(n)}\}_{n=1}^N$.

ii. Set particle $\tilde{\mathbf{x}}_t^{(n)} = \mathbf{x}_t^{(j)}$.

iii. Propagate $\mathbf{x}_{t+1}^{(n)} \sim \ell\left(\mathbf{x}_{t+1} \mid \tilde{\mathbf{x}}_t^{(n)}\right)$ using the resampled particle value. This generates a final sample of N unique values for $\mathbf{x}_{t+1}$.

(d) End iteration over particle indices n.

(e) Replace the set $\{\mathbf{x}_t^{(n)}\}_{n=1}^N$ with the resampled set $\{\tilde{\mathbf{x}}_t^{(n)}\}_{n=1}^N$.

4. End iteration over times t.

The basic idea of the CDLM is that at each time $t-1$, it tries to predict the next observed value $\mathbf{y}_t$ through the intermediate unobserved $\mathbf{x}_t$ conditional on each potential value of the latent λ_t. The predicted densities of $\mathbf{y}_t$ conditional on each potential value $\lambda_t = k$ are evaluated at the observed value of $\mathbf{y}_t$; whichever value of $\lambda_t = k$ gives higher density at the observed $\mathbf{y}_t$ is deemed the most likely for λ_t. Essentially we are trying to determine which behavior (given the parameter values we have learned) is the most likely to result in the shark ending up at the observed future location. Resampling particles by these conditional densities (weights), which for each particle n depends on the particle's latent parameter values $\boldsymbol{\theta}^{(n)}$, is how the particle filter should learn optimal values for the parameters.

The algorithm below expands on the general outline to add conditioning on λ_t, which can take any value $k \in \{1, \ldots, K\}$ for a positive integer K. Assume that the particle filter has N particles and learns from the $T + 1$ observed values $\mathbf{y}_{1:T}$:

General outline of the CDLM including latent variable:

```
For observations $t = 0, \dots, T-1$:

  1. For each particle $n = 1, \dots, N$:

     (a) For each potential value $k = 1, \dots, K$ of $\lambda_t$ (here, shark behavior):

         i. Determine the state-conditional densities of the observed value
            $\mathbf{y}_t$, marginalizing over the potential $\mathbf{x}_t \mid (\lambda_t = k)$:

            $m^{(n)}_{\text{predict}}\left(\mathbf{y}_t \mid \mathbf{x}^{(n)}_{t-1},\ \lambda_t = k,\ \lambda^{(n)}_{t-1},\ \boldsymbol{\theta}^{(n)}_{t-1}\right) \propto$
            $d\left(\mathbf{y}_t \mid \mathbf{x}^{(n)}_{t-1},\ \lambda_t = k, \lambda^{(n)}_{t-1},\ \boldsymbol{\theta}^{(n)}_{t-1}\right) \times \Pr\left(\lambda_t = k \mid \lambda^{(n)}_{t-1},\ \boldsymbol{\theta}^{(n)}_{t-1}\right)$

            The first component is the predicted density of the observation
            at the value of $\lambda_t = k$, and the second is the behavior transition
            probability into that $\lambda_t$.

        ii. Let $w^{(n)}_{t|k} = m^{(n)}_{\text{predict}}(\mathbf{y}_t \mid \dots, \lambda_t = k)$ be the above marginal
            density evaluated at the observed $\mathbf{y}_t$; these define behavior ($\lambda_t$)-
            specific weights for each particle. The overall weights for each
            particle are $w^{(n)}_t = \sum_{k=1}^{K} w^{(n)}_{t|k}$.

         End iteration over $k$.

     End iteration over $n$.

  2. Resample the set of particles $\mathbf{x}^{(n)}_{t-1}$ (and all their associated parameters)
     in proportion by the overall weights $\{w^{(n)}_t\}_{n=1}^{N}$. Let $\{\tilde{\mathbf{x}}^{(n)}_{t-1}\}_{n=1}^{N}$ represent
     the resampled set. Higher weights indicate the parameter values match
     the observed data better. Let the weights of the resampled particles now
     be $\tilde{w}^{(n)}_t$ and $\tilde{w}^{(n)}_{t|k}$.

  3. For each particle $n = 1, \dots, N$:
```

(a) Propagate values of λ_t by the switching probabilities (from the weights) of the resampled particles by the discrete distribution

$$\lambda_t^{(n)} \sim \Pr\left(\lambda_t = k \mid \tilde{\lambda}_{t-1}^{(n)}\right) = \tilde{w}_{t|k}^{(n)} / \tilde{w}_t^{(n)}$$

so the weights are normalized to represent probabilities.
If $K = 2$, this reduces to the bivariate case so $\lambda_t^{(n)} \sim \mathcal{Ber}\left(\tilde{w}_{t|1}^{(n)} / \tilde{w}_t^{(n)}\right)$

(b) Propagate states $\mathbf{x}_t^{(n)} \sim \mathcal{N}\left(\mathbf{L}_{\lambda_t}\tilde{\mathbf{x}}_{t-1}^{(n)},\ \tilde{\mathbf{Q}}_{\lambda_t} \mid \tilde{\boldsymbol{\theta}}_{t-1}\right)$ and update each particle n's sufficient statistics $\mathbf{S}_t^x$ for only the behavior $\lambda_t^{(n)}$ that is observed.

(c) Update the posterior distribution hyper-parameters of parameters $\tilde{\boldsymbol{\theta}}_t^{(n)} \mid \left(\mathbf{x}_t^{(n)}, \tilde{\boldsymbol{\theta}}_{t-1}^{(n)}\right)$ and draw a new value of $\boldsymbol{\theta}_t^{(n)}$ from them.

End iteration over n.

End iteration over t.

4.5 1-D robot CDLM illustration

In the CDLM setup, parameters are modeled by seeing which values best predict the observed $\mathbf{y}_t$ given unobserved $\mathbf{x}_{t-1}$ (note, not $\mathbf{x}_t$) by simulating the intermediate $\mathbf{x}_t$. In this case that means we have simulated $\mathbf{x}_{t-1}$ (starting location ζ_{t-1} and velocity v_{t-1} from there), and from this we can predict the next unobserved location ζ_t (first component of $\mathbf{x}_t$). Then we see what velocity v_t from ζ_t best predicts the observed location at the next time, $z_{t+1} = \mathbf{y}_t$. From the EKF linearization through the Jacobian (see 3.4), we can derive the following distributions:

$$\begin{aligned}
\mathbf{x}_{t-1} \mid \ \text{data} &\sim \mathcal{N}_2(\boldsymbol{\mu}_{t-1}, \Sigma_{t-1}), & \boldsymbol{\mu}_{t-1} &= \begin{bmatrix} \mathrm{E}(\zeta_{t-1}) & \alpha \end{bmatrix} \\
\mathbf{x}_t \mid \ \mathbf{x}_{t-1} &\sim \mathcal{N}_2(\boldsymbol{\ell}_\mu(\mathbf{x}_{t-1}), \quad \mathbf{P}_t), & \mathbf{P}_t &= \mathbf{L}(\mathbf{x}_{t-1})\Sigma_{t-1}\mathbf{L}(\mathbf{x}_{t-1})^T + \mathbf{q}_{t-1} \\
\mathbf{y}_t \mid \ \mathbf{x}_t &\sim \mathcal{N}_2(\boldsymbol{m}_\mu(\mathbf{x}_t), \quad \mathbf{D}_t), & \mathbf{D}_t &= \mathbf{M}(\mathbf{x}_t)\mathbf{P}_t\mathbf{M}(\mathbf{x}_t)^T + \mathbf{r}_t \\
\mathbf{y}_t \mid \ \mathbf{x}_{t-1} &\sim \mathcal{N}_2(\boldsymbol{m}_\mu(\boldsymbol{\ell}_\mu(\mathbf{x}_{t-1})), \quad \mathbf{S}_t), & \mathbf{S}_t &= \mathbf{M}(\boldsymbol{\ell}_\mu(\mathbf{x}_{t-1}))\mathbf{P}_t\mathbf{M}(\boldsymbol{\ell}_\mu(\mathbf{x}_{t-1}))^T + \mathbf{r}_t \\
\text{Denote } \mathbf{y}_t \mid \ \mathbf{x}_t \text{ as} & \quad \mathbf{y}_t \sim m_{\text{current}}(\cdot|\mathbf{x}_t, \alpha) \\
\text{and } \mathbf{y}_t \mid \ \mathbf{x}_{t-1} \text{ as} & \quad \mathbf{y}_t \sim m_{\text{predict}}(\cdot|\mathbf{x}_{t-1}, \alpha)
\end{aligned}$$

Here Σ_{t-1} represents the covariance of $\mathbf{x}_{t-1}$ conditioned on all the previous states $\mathbf{x}_{0:(t-1)}$ and observations $\mathbf{y}_{1:(t-1)}$. The function $\boldsymbol{\ell}_\mu(\mathbf{x}_{t-1})$ includes a new simulation of $v_t \sim \mathcal{N}(\alpha, 1)$ which we will use to update the value of α.

Since we assume that the standard deviation of v_t $(= 1)$ is known, we specify a prior on the values of the mean velocity α. Let $\boldsymbol{\Omega}_0 = (\mu_0, \sigma_0^2)$ denote the initial values of the prior hyper-parameters. Here, α_t (with a subscript) is a draw from the prior on α (which is a fixed number) at time t; thus our initial prior draw is $\alpha_0 \sim \mathcal{N}(\boldsymbol{\Omega}_0)$. Let N be the number of particles used in the simulation.

CDLM outline for 1-D robot:

Initialization of particle values: Let $\{\alpha_0^{(n)}\}_{n=1}^N$ be N iid draws of α from the prior $\mathcal{N}(\boldsymbol{\Omega}_0)$, and independently draw $v_0^{(n)} \sim \mathcal{N}(\alpha_0^{(n)}, 1), n = 1, \ldots, N$. Set all initial locations $\zeta_0^{(n)} = z_0$, the first observed value. Set $\mathbf{x}_0^{(n)} = \begin{bmatrix} \zeta_0^{(n)} & v_0^{(n)} \end{bmatrix}^T, n = 1, \ldots, N$.

Iterate over $t = 0, \ldots, T-1$, where $T+1$ is the number of observations:

1. We want to see which particle's parameters (particularly $\alpha_t^{(n)}$), were we to simulate $\mathbf{x}_{t+1}^{(n)} \mid \left(\mathbf{x}_t^{(n)}, \alpha_t^{(n)}\right)$ (i.e., the velocity $v_{t+1}^{(n)} \sim \mathcal{N}(\alpha_t^{(n)}, 1)$), would best predict $\mathbf{y}_{t+1}$ (the observed location z_{t+2}, two timesteps ahead).

2. For particle $n = 1, \ldots, N$:

(a) Marginalize over the potential $\mathbf{x}_{t+1}^{(n)}$ to get the desired resampling density for predicting the location z_{t+2}:

$$m_{\text{predict}}^{(n)}\left(\mathbf{y}_{t+1}\middle|\mathbf{x}_t^{(n)},\, \alpha_t^{(n)}\right) \propto$$
$$\int m_{\text{current}}^{(n)}\left(\mathbf{y}_{t+1}\middle|\mathbf{x}_{t+1}^{(n)},\, \alpha_t^{(n)}\right) \times \ell^{(n)}\left(\mathbf{x}_{t+1}^{(n)}\middle|\mathbf{x}_t^{(n)},\, \alpha_t^{(n)}\right) \mathrm{d}\mathbf{x}_{t+1}^{(n)}$$

(b) Assign a weight $w_{t+1}^{(n)}$ to each particle, where $w_{t+1}^{(n)} \propto m_{\text{predict}}^{(n)}(\mathbf{y}_{t+1})$, the density evaluated at the observed value. Particles whose parameters give higher likelihood to the observed $\mathbf{y}_1 = z_2$ get higher weights.

End iteration over n.

3. Create a new set of N particles by random resampling with replacement (RRWR) of the particle values $\{\mathbf{x}_t^{(n)}\}_{n=1}^N$, by their overall weights $\{w_{t+1}^{(n)}\}_{n=1}^N$. Let $\{\tilde{\mathbf{x}}_t^{(n)}\}_{n=1}^N$ represent the resampled set.

4. For particle $n = 1, \dots, N$:

(a) Propagate (draw the next value) of $\mathbf{x}_{t+1}^{(n)} \mid \left(\mathbf{x}_t^{(n)},\, \alpha_t^{(n)}\right)$.

(b) Given $\zeta_t^{(n)} \subset \mathbf{x}_t^{(n)}$, update the posterior distribution of $\alpha^{(n)} \mid \left(\mathbf{x}_{t+1}^{(n)},\, \alpha_t^{(n)}\right)$ by updating the hyper-parameters $\boldsymbol{\Omega}_{t+1}^{(n)} \mid \left(\mathbf{x}_{t+1}^{(n)}, \boldsymbol{\Omega}_t^{(n)}\right)$.

(c) Draw a new mean velocity $\alpha_{t+1}^{(n)} \sim \mathcal{N}\left(\boldsymbol{\Omega}_{t+1}^{(n)}\right)$ and $v_{t+1}^{(n)} \sim \mathcal{N}\left(\alpha_{t+1}^{(n)},\, 1\right)$ for $\mathbf{x}_{t+1}^{(n)}$.

End iteration over n.

End iteration over t.

As illustrated in Figure 4.2, resampling values x_i by their density $m(x_i)$ generates a discrete approximation of a random sample from the density. In step 3 we want to generate a sample of predictions ζ_1 of the observed location z_2. This is a draw from the density $m_{\text{predict}}^{(n)}$. By resampling original particles

$\{\mathbf{x}_0^{(n)}\}_{n=1}^N$ by the weights $w_1^{(n)} \propto m_{\text{predict}}^{(n)}\left(\mathbf{y}_1 \middle| \mathbf{x}_0^{(n)}, \alpha_0^{(n)}\right)$, and then drawing a value of $\mathbf{x}_1^{(n)}$ (which includes $\zeta_1^{(n)}$) from the density $\ell^{(n)}\left(\cdot \middle| \mathbf{x}_0^{(n)}, \alpha_0^{(n)}\right)$ from the resampled particle set, the resulting set of predictions $\{\zeta_1^{(n)}\}_{n=1}^N$ should be distributed around the observed z_2, with more values close to z_2.

Figure 4.3 illustrates the use of prediction likelihoods as resampling weights. Here, we have a toy example of three particles at time $t-1$ with simulated locations ζ_{t-1}; using the draw of the velocity α_{t-1} for each, the observed location $\mathbf{y}_t = z_{t+1}$ (the vertical dotted line) two steps ahead is predicted by marginalizing over the next true location ζ_t. Each particle predicts of the location z_{t+1} from the distribution m_{predict} (see 4.5). The confidence interval of the prediction is indicated by the left and right ends of of the gray rectangle, and the vertical line is the mean of the prediction, the best guess of its value. The confidence interval takes into account the errors from ζ_{t-1} to ζ_t and then to z_{t+1} (see 4.5). The confidence interval is symmetric since the errors are Gaussian.

The resampling weights z_t are each particle's density functions $m_{\text{predict}}^{(n)}$ evaluated at the observed $\mathbf{y}_t = z_{t+1}$. Of course, this measure of accuracy accounts for the density's mean and standard deviation. So, in Figure 4.3, particles 2 and 3 have the highest weights (0.4 and 0.5) because their predictions (end of the right arrow) are most likely near the observed value (dotted line); their confidence intervals have high density at the observed value. However, particle 3 has a higher weight because it has a smaller standard deviation of the error (width of the rectangle) and hence is more certain. Particle 1 has a low weight because its prediction is both far from the observed value and has low standard deviation, so it is confident in its inaccurate prediction.

This example illustrates the importance of appropriate prediction error in the PF in learning model parameters well. This is true of Bayesian inference in general, of course. The prior distribution we set on a parameter expresses our prior knowledge. If the distribution has too low standard deviation, it sets an inflexible prior with high confidence in our guess, which will fail to learn

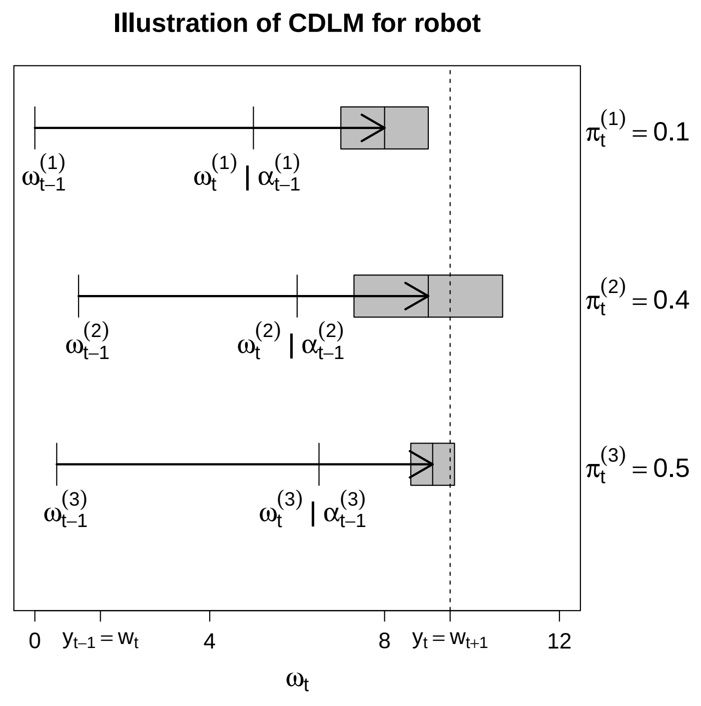


Figure 4.3: Toy illustration of 3 particles for a 1-D robot, predicting the next observed location $\mathbf{y}_t$ given simulated locations at ζ_{t-1}. Particle weights w_t reflect the particle's accuracy in predicting the next location. The width of the gray boxes represent 95% predictive intervals for the particle's density estimate of the observed location z_{t+1}.

from data values that are far from the distribution's center. In the case of the 1-D robot, this corresponds to the error distributions, and thus the confidence intervals of predictions being too narrow. In the extreme case, nearly all of the particle predictions have low density at the observed values, and hence have low weights $w_t^{(n)}$. In this case, the PF may fail to discriminate between decent and bad guesses of, say, the velocity, and thus cannot identify good particles. It will thus resample only a few, or resample all particles at roughly the same (low) weight.

Similarly, if the error distributions are too wide, giving wide confidence intervals, the PF cannot identify good particle guesses because they are all roughly bad (since the confidence intervals are too wide). Thus, a good PF, like a good prior distribution, should allow a reasonable amount of error. A too-small error means our estimates of the velocity distribution will have have too big a variance after updating.

4.6 Effective sample size

In a particle filter, resampling of particles by their weight $w_t^{(n)}$ serves the dual purpose of (1) removing particles with low weights (i.e., low predictive power) and replicating particles with higher weight, and thus also (2) weighting the set of particles so that they form a discrete approximation of the desired density. When particles are resampled, the set of parameter values they have learned over the process, as well as the particle history, are completely replaced. If a particle is not resampled, its parameter estimates are lost completely. Resampling is thus well-known to cause particle filters to be unstable for the following related reasons:

- **Degeneracy** is a situation when a few particles have very high weight and most have very low weight. Resampling particles in this situation causes

- **Impoverishment**, which is when the diversity of particle estimates for parameters is very low, meaning many particles have learned the same or similar values for a parameter. In the most extreme case, there is no diversity because all particles have been resampled with a single particle.

These problems illustrate the importance of particle diversity. On the one hand, while a particle filter should favor particles whose parameter estimates perform best. On the other hand, it is dangerous to completely discard particles with estimates that do not perform as well. The true values of the parameters may evolve over time; in general, even if the distribution is fixed, variable values, for instance, a shark's speed, may vary due to randomness. Because particle filters are a sequential learning algorithm, a particle whose current estimate of the mean speed, say, predicts the observed shark location well, may not perform as well in the future. This is strongly related to the fact that prior distributions on parameters should not be point masses but rather incorporate uncertainty (i.e., have variance). Thus, it is important to maintain some diversity of particle parameter values throughout the process so the filter does not get stuck trying to predict an observed value that all its particles believe is very unlikely.

One way to address this is to limit resampling. For instance, one may want to resample for the first time after a certain number of initial steps have passed, to let the particles evolve a bit. A commonly-used measure proposed by Liu ([33]) involves the effective sample size (N_{eff}), which measures particle degeneracy based on the set of weights. It is approximated by the following calculation $\widehat{N_{\text{eff}}}$:

$$N_{\text{eff}} \approx \widehat{N_{\text{eff}}} = \frac{1}{\sum_{n=1}^{N} \left(w_t^{(n)}\right)^2}, \quad \text{where } \sum_{n=1}^{N} w_t^{(n)} = 1$$

This formulation ensures that $1 \leq \widehat{N_{\text{eff}}} \leq N$. In the extreme case of absolute degeneracy, for one particle n, its weight $w_t^{(n)} = 1$ and the rest are 0, so $\widehat{N_{\text{eff}}} = 1$. On the other hand, if all weights are equal ($w_t^{(n)} = 1/N, \forall n$), then $\widehat{N_{\text{eff}}} = N$, the number of particles; this is the case of perfect diversity, since all particles are equally likely to be resampled. The effective sample size expresses the (weighted) diversity of sample values, since if $\widehat{N_{\text{eff}}}$ is low, then only a few unique particles will be resampled. Resampling every so often is necessary to prevent degeneracy and the effective sample size from falling to its minimum of 1, but resampling too often can hurt the filter's ability to learn parameter estimates effectively. To avoid resampling too soon when the particles and their weights have not begun to diversify enough, one may pick a threshold $1 < N_{\text{thresh}} < N$ and only resample if the degeneracy is bad enough, that is, if $\widehat{N_{\text{eff}}} < N_{\text{thresh}}$ (Doucet, [14]). One particular choice is setting $N_{\text{thresh}} = N/2$, suggested by Grisetti et al. ([18]).

4.7 Low-variance sampling

Traditionally, given a set of weights $\{w_t^{(n)}\}_{n=1}^N$ that are normalized so $\sum_{n=1}^N w_t^{(n)} = 1$, the indices of particles to be resampled are drawn by independent random sampling with replacement (RSWR) from the set $\{1, \ldots, N\}$ in proportion to their weight. RSWR can be termed 'roulette wheel sampling' it is analogous to a a gambler making N repeated rolls of a roulette wheel, each roll giving one number as a result, just as each draw from the particle indices is independent and results in one index. If we denote x_n as the number of times particle $\mathbf{x}_t^{(n)}$ is resampled in the set $\{\tilde{\mathbf{x}}_t^{(n)}\}_{n=1}^N$, these values $(x_1, \ldots, x_N)$ jointly have a multinomial distribution with number of trials N and probability vector $(w_t^{(1)}, \ldots, w_t^{(n)})$. Thus, with RSWR, particle $\mathbf{x}_t^{(n)}$ is resampled on average $\mathrm{E}(x_n) = Nw_t^{(n)}$ times with variance $Nw_t^{(n)}(1 - w_t^{(n)})$. The weakness here is that this variance can be high, depending on the value of the weight. Furthermore, particles with low weight are likely not to be resampled at all since it is not guaranteed they will be resampled in proportion to their weight, but some of these particles are essential to avoiding degeneracy.

Alternative methods of resampling, such as low-variance or universal stochastic resampling, developed by Baker([4]), ensure particles are resampled proportionately to their weight *each time*, not just on average. Imagine a roulette wheel now with N spaces, each with area proportionate to a weight $w_t^{(n)}$. Instead of N independent and identically distributed (iid) draws from the set $\{1, \dots, N\}$, the concept is similar to placing N equally spaced marks on the wheel above and spinning once, which guarantees that colors will be rolled (a mark falls in them) proportionately to their area on the wheel, except in the case of particles imagine that each of the N slots has area proportion to its weight $w_t^{(n)}$. This reduces the variance associated with resampling while preserving the randomness.

The basic algorithm for low-variance sampling can be formulated as follows:

- Consider N particles $\{\mathbf{x}^{(n)}\}_{n=1}^N$ with indices $n = 1, \dots, N$ with weights $\{w^{(n)}\}_{n=1}^N \geq 0$ normalized so that $\sum_{n=1}^N w^{(n)} = 1$. Note the weights do not need to be in a specific order.

- Let $\bar{w}_n = \sum_{i=1}^n w^{(n)}$ be the cumulative sum of the first n weights, so $\bar{w}_N = 1$, and define $\bar{w}_0 = 0$.

- We wish to draw a low-variance sample of size S; for particle resampling, we will have $S = N$. Let $s \in \{1, \dots, S\}$ index the resampled particles, and let $n(s) \in \{1, \dots, N\}$ be the index of the original N particles that is resampled in new particle index s. For instance, if $N = 10$ and we want a resampled set of size $S = 3$, then we may have $n(1) = 2, n(2) = 2, n(3) = 5$; this means the resampled set of size 3 consists of the values of the second (replicated twice) and fifth original N particles.

- Draw $u_1 \sim \mathcal{U}(0, 1/S)$. Now specify S partitions of the interval $[0, 1]$, where the first is $[0, u_1]$ and the others are of equal length $1/S$, corresponding to $(s/S, (s+1)/S]$. For, instance if we resample a set of size $S = 3$, say $u_1 \sim (0, 1/3)$ is drawn, then the other intervals are $(u_1,\ u_1 + 1/3]$ and $(u_1 + 1/3,\ u_1 + 2/3]$. In general, for $s = 1, \dots, S$, set $n(s) = \underset{j}{\operatorname{argmin}}\{j\colon u_1 + (s-1)/S \leq \bar{w}_j\}$, or equivalently $n(s) = \{j\colon \bar{w}_{j-1} < u_1 + (s-1)/S \leq \bar{w}_j\}$; note we may have $n(i) = n(j)$ for $i \neq j$, so some indices may be repeated.

- The low-variance resampled set is $\{\tilde{\mathbf{x}}^{(s)}\}_{s=1}^{S}$, where $\tilde{\mathbf{x}}^{(s)} = \mathbf{x}^{(n(s))}$.

An illustration of the difference between low-variance sampling and RSWR is shown in Figure 4.4. Visually, we can imagine creating N intervals partitioning $[0, 1]$, with interval n having width $w^{(n)}$. The assignment of particle indices above by assigning $n(s) = j$ if the s^{th} uniform draw falls in the j^{th} interval $(\bar{w}_{j-1}, \bar{w}_j]$ is logical because the probability of falling in this interval is proportional to the width of that interval and to the weight $w^{(j)}$. Thus, higher weighted particles are more likely to be resampled.

Resampling particles by independent RSWR is equivalent to making S (which we will take as N, the same as the number of particles) independent draws $\{u_1, \dots, u_S\} \sim \mathcal{U}(0, 1)$, and assigning $n(s) = j$ if u_s is in the j^{th} interval $(\bar{w}_{j-1}, \bar{w}_j]$. Because these are independent draws—in the metaphor of roulette, they are S repeated individual spins of the wheel—they will not necessarily be equally spaced along $(0, 1)$. This means that while the particles are *on average* resampled proportionately to their weight, a single instance of resampling will not necessarily contain the particles in proportion to their weights, meaning a single draw will not necessarily approximate the desired distribution well (see Figure 4.1). Low-variance sampling proceeds by a random draw $u_1 \sim \mathcal{U}(0, 1/S)$ (which provides the randomness) but thereafter $\{u_2, \dots, u_S\}$ are not each drawn uniformly $\mathcal{U}(0, 1)$ but rather equally spaced after u_1. The equal spacing guarantees resampling that is as close to proportional as possible and that the resulting number of times each particle n is likely to be resampled has low variance compared to N iid draws.

Figure 4.5 shows the proportions of the indices $\{n(s)\}_{s=1}^{S}$ corresponding to each of the original particle indices $\{1, \dots, N\}$ in Figure 4.4, from one draw each using low-variance sampling and RSWR. The horizontal lines indicate the normalized weights $w^{(n)}$ of each particle. We see that the proportions under low-variance sampling (black dots) are almost always closer to the true weights or proportions than are the proportions under RSWR (hollow dots). With a larger number N of particles, the low-variance proportions would be

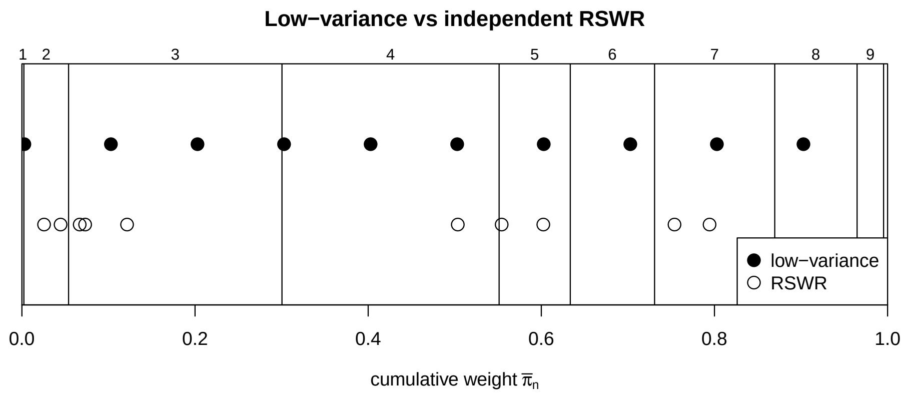


Figure 4.4: Resampling of the $N = 10$ particle values $\{20, 21, \dots, 29\}$ by their weights (see Figure 4.1). The low-variance resampling draws (black dots) are equally spaced, but the independent RSWR draws (hollow circles) are not, and hence do not guarantee proportional resampling on a given draw. For instance, particle $n = 2$ is over-sampled.

much closer to the weights.

4.8 Posterior parameter inference

In the end, each particle (simulation) in a PF learns its own posterior distribution of each parameter. Ultimately, we want to make posterior inference on the parameters (for instance, to approximate the mean velocity in each behavioral state) by combining estimates across the particles. Normally, Bayesian (or other) inference assumes that the sampled values are generated independently, but this is not strictly true for the PF. While it is true that at every step, each particle's simulations of say, the shark's velocity, are independent, the entire particle histories are not independent of each other. This is because, for instance, if at step t particle $\mathbf{x}_t^{(1)}$ is resampled to replace both particles 1 and 2 (i.e., $\tilde{\mathbf{x}}_t^{(1)} = \tilde{\mathbf{x}}_t^{(2)} = \mathbf{x}_t^{(1)}$), then those two new particles $n = 1$ and 2 have identical histories and parameters, so their posterior distributions at that point are not learned independently. At any given point t,

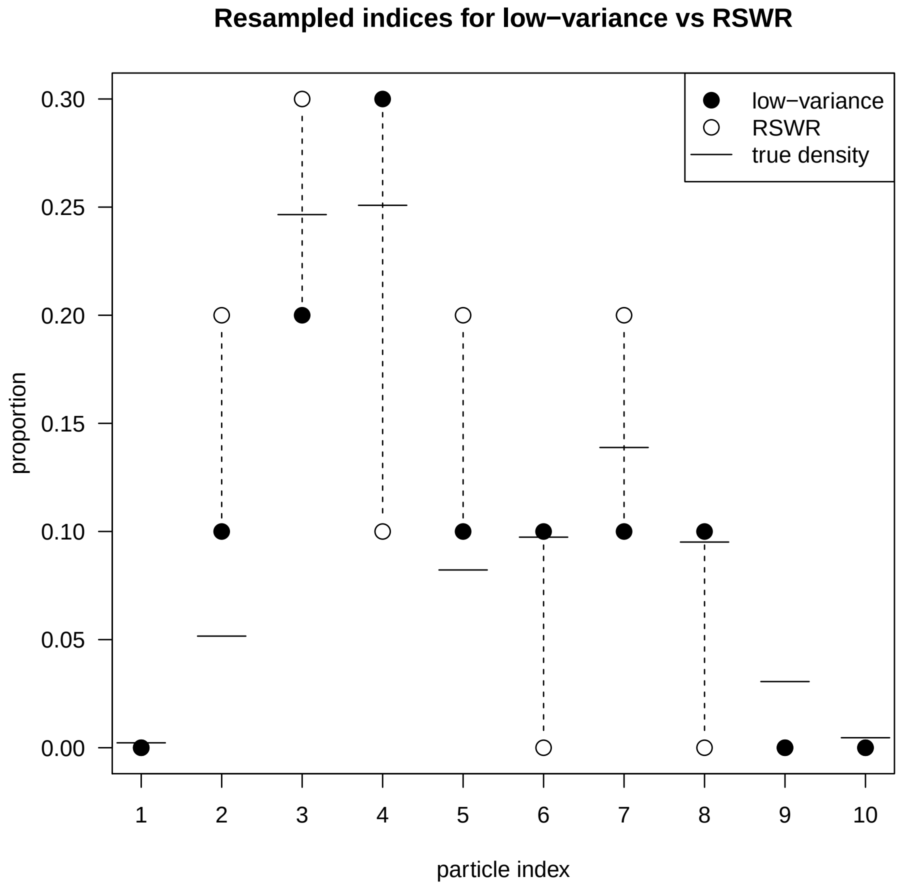


Figure 4.5: Proportions of the resampled indices corresponding to the original indices, under low-variance sampling and RSWR. The horizontal lines indicate the weights $w^{(n)}$, which are the proportions of the resampled set that each particle n should occupy, on average. The low-variance resampling makes a given resampling closer in index distribution than if RSWR were used, which is important since the resampling is supposed to represent a random draw from the desired distribution.

draws from the distributions are independent, however.

There have been some attempts to develop PF algorithms that generate conditionally independent samples in various ways (see Lin et al. [32]; Lamberti et al. [31]), but generally the practice is to treat the posterior distributions as independent, even though they are not. Crisan and Míguez ([13]) demonstrate that kernel density estimates of PF densities, treating them as independent, are generally reasonable asymptotically.

In PF inference, at the final time T each particle n's states $\mathbf{x}$ and parameters $\boldsymbol{\theta}$ have distribution $d\left(\left(\mathbf{x}_T^{(n)},\, \boldsymbol{\theta}^{(n)}\right) \mid \mathbf{y}_{1:T}\right)$. In order to do joint inference using all the particles, we need to approximate their joint distribution $d\left(\left\{\left(\mathbf{x}_T^{(n)},\, \boldsymbol{\theta}^{(n)}\right)\right\}_{n=1}^{N} \mid \mathbf{y}_{1:T}\right)$, which in the case of independence reduces to a product of the individual distributions. Nevertheless, it is customary to approximate the joint by a weighted sum of the individual particle distributions

$$\sum_{n=1}^{N} w_T^{(n)} \times d\left(\left(\mathbf{x}_T^{(n)},\, \boldsymbol{\theta}^{(n)}\right) \mid \mathbf{y}_{1:T}\right)$$

where the weights $w_T^{(n)}$ weight each of the n distributions to represent the desired joint density we want to approximate a draw from. In traditional SIR algorithms where propagation precedes resampling, we need to resample the particles $\mathbf{x}_T^{(n)}$ by the weights to approximate the desired joint density. In the case of the CDLM, the particles $\mathbf{x}_{T-1}^{(n)}$ were resampled by their weights $w_T^{(n)} \propto m_{\text{predict}}^{(n)}\left(\mathbf{y}_T \mid \mathbf{x}_{T-1}^{(n)}, \boldsymbol{\theta}_{T-1}^{(n)}\right)$ and then $\mathbf{x}_T^{(n)} \mid \left(\mathbf{x}_{T-1}^{(n)}, \boldsymbol{\theta}_{T-1}^{(n)}\right)$ are propagated. Thus the propagated $\{\mathbf{x}_T^{(n)}\}$, after parameters $\boldsymbol{\theta}_{T-1}$ are updated, should be equally weighted to represent the joint density, since they were resampled according to this density, and thus propagation should not change their weighting.

Therefore, to perform posterior inference we will simply draw samples of the desired parameter from all the particles at equal weight, and perform density estimation; this will also give us estimates of the posterior intervals. We will generate a sufficient number K (e.g., 100) independent draws from each posterior of each parameter $\theta \in \boldsymbol{\theta}$: $\theta_k \sim d(\cdot \mid \mathbf{x}_{0:T}^{(n)},\, \mathbf{y}_{1:T})$, $k = 1, \ldots, K$, and estimate a population density of θ by a density estimate on these draws combined. Ideally, due to the inherent randomness in Markov Chain Monte Carlo (MCMC) algorithms, the entire process of the PF should be repeated a number of times to account for possible bad estimates early in the simulation that affect the rest of it.

As a toy illustration, say our PF involves a variable with distribution $x \sim f(x \mid \mu)$ (not necessarily normal); for instance x could represent speed and μ its mean parameter. We wish to perform posterior inference on μ, based on the posterior distributions for this parameter learned by all the particles. As a further example, in Section 6.4 we propose a model for a parameter η to capture the behavioral effects of inter-shark influence, which is an input into another element of the particle filter. We have its priors and posteriors as $\eta \sim \mathcal{N}(\eta_0, \tau_0)$, and we wish to do posterior inference on η. Returning to the general case of a parameter μ, say the posteriors are the normal distribution $\mu \sim \mathcal{N}(\mu_*, \sigma_*)$.

Figure 4.6 shows the posterior densities of $N = 10$ such particles, with various values of μ_* and σ_*, but centered roughly around 0, which we would hope is close to the true value of μ. To obtain, say, an estimated 95% posterior interval of μ, we can draw 100 samples of $\mu^{(n)}$ from each particle's posterior distribution of μ, pool the samples (using the approximation of independence), and construct a 95% empirical posterior interval from the combined N samples. The pooled sample can also be used to construct a posterior density estimate of μ, by combining estimates across all particles. In this case, the estimated density is essentially an estimate of a mixture normal density with N components $\mathcal{N}\left(\mu_*^{(n)}, \sigma_*^{(n)}\right)$ that are equally weighted. This approach to posterior inference for parameters is flexible and can be used with any posterior distribution. Remarkably, based on our exposure to the particle filtering literature, researchers seem much more interested in the overall filter performance and accuracy of modeling the states $\mathbf{x}_t^{(n)}$ rather than doing inference on the underlying parameters $\boldsymbol{\theta}$.

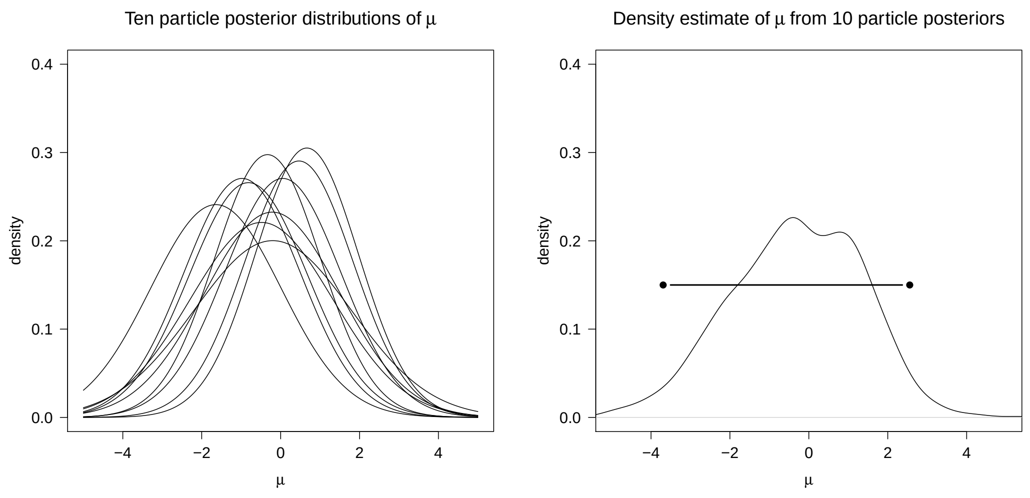


Figure 4.6: Toy illustration of posterior PF inference for parameters. Left: ten particles $n = 1, \ldots, N$, each with a different posterior distribution of the parameter $\mu \sim \mathcal{N}\left(\mu_*^{(n)}, \sigma_*^{(n)}\right)$. Right: density estimate of μ combining samples of size 100 from each of $N = 10$ particles, along with a 95% posterior interval based on the empirical samples.

4.9 Connections with robotics and tracking algorithms

In Section 4.5, we used the simple example of a one-dimensional robot with a location sensor in order to illustrate the basic mechanics of SSMs in general, and the CDLM particle filter in particular, which we will use to model the shark movement. One might wonder why the analogy from sharks to robots is appropriate at all. After all, sharks move on their own free will, whereas robots do not. However, the analogy is apt because particle filters and similar probabilistic models are widely used in robotics.

Sebastian Thrun, who founded the self-driving car division at Google, provides an overview of probabilistic algorithms for robotics in [48]. In summary, autonomous robots move by taking various readings of their surrounding by using sensors, such as camera readings, laser range scans, or odometer readings. Due to the inherent imperfections inherent in real life measurements, any sensor reading should be treated as having a certain degree of error associated with it. Similarly, the VPS transmitters on the sharks also provide measurements with error. In the SSM setup, $\mathbf{x}_t$ are unobserved true values (say, of location), for which $\mathbf{y}_t$ are the observed measurements taken with associated error $\mathbf{q}_t$, often modeled by a Gaussian distribution.

Autonomous robots are designed to perform a variety of tasks while moving around an area, particularly localization (determining their location on a map after a sequence of moves and sensor readings), mapping (exploring their environment and determining the map or layout of it, including the locations of obstacles and walls, doors, etc.), and navigation (planning a route between their location and a desired destination). Robots may need to be able to recover from a disturbance or localization failure, such as being suddenly moved to a new location, and then reorient themselves and find their way back; this is known as the ‘kidnapped’ robot problem.

As Thrun shows, due to both the error inherent in the sensors, and factors such as a dynamic environment (for instance a robot that is surrounded by crowds of people whose position may change, or doors that may open or shut), the best algorithms for solving these problems are probabilistic. This means, for instance, when performing localization, a robot calculates a probability distribution (2-D Gaussian confidence ellipses, for instance) around a prediction of its location; the robot quantifies the degree of uncertainty it has in its measurements, similar to how a prior distribution on a model parameter expresses our uncertainty of its value. The fact that a robot expresses uncertainty about its location or obstacles helps it perform better in navigation and recovery. For instance, if a robot is somewhat uncertain about obstacles nearby it may be more cautious in moving around them. In mapping, a robot needs to determine the location of obstacles, say by gridding the area and determining if the grid is occupied or not; if it can assign each grid a probability of occupation rather than a binary classification of occupied or empty, the map may be more accurate. Also, having some uncertainty lets the robot learn and modify its beliefs in the face of new sensor measurements.

The main difference between the robots and the sharks is that robot navigation models have to work in real time, whereas we are trying to model the shark data sequentially but after the fact. But if you take robot data that has already been observed ($\mathbf{y}_t$) and model the unobserved locations ($\mathbf{x}_t$) that generated these observations, that is what we are doing with the sharks. If we formulate a probabilistic filter model, each new observed location $\mathbf{y}_t$ can be incorporated probabilistically into our parameter learning. If a particle's parameters assign low density or probability to an observation, we can express lower confidence in that particle's values by resampling it at a lower weight.

Robots are a specific example of the use of the Kalman filter and its extensions to the problem of localization. More generally, the literature on **tracking** deals with the task of an observer—for instance, a SONAR technician on a submarine, or a surface-to-air missile defense system—estimating the location and velocity of a moving object (target). The object's motion must be estimated by sensors, such as rangefinders (which include RADAR and SONAR), which may be located at the observer's location or elsewhere; the measurements made by the sensors are assumed to have a certain degree of error. Tracking algorithms have been adapted to many situations, such as tracking a maneuvering target, meaning one that may re responsive or evasive to movements by the observer, such as an enemy submarine, or the non-maneuvering case; the observer's sensor may also be moving (e.g., a submarine), or stationary.

Naturally, estimating the velocity and location of the moving target involves estimating the direction of movement (bearing) and the distance from the sensor (range). Often, algorithms are made to address the situation where the sensor measures only one of these aspects (bearings- or range-only tracking). In a bearings-only algorithm, for instance, the range is unobserved and this error, as opposed to sensor measurement error, and can only be measured if the observer maneuvers (see [42] for more details). This illustrates another key difference between tracking and our setup: since we are not using sensor data in real-time, but rather are retroactively using location data from VPS transmitters, which should not be significantly affected by the range and bearing sensor issues, we can ignore some of these complications from the tracking literature.

CHAPTER 5

SHARK EKF SETUP

5.1 Specification of the state-space model

In our SSM, the unobserved state vector $\mathbf{x}_t$ models the unobserved true location coordinates $\boldsymbol{\zeta}_t = \begin{bmatrix}\zeta_{1,t} & \zeta_{2,t}\end{bmatrix}$, as well as the log-speed $(\ln(v_t)$ and bearing angle ψ_t (radians) characterizing the shark's movement at time t from $\boldsymbol{\zeta}_t$. The state $\mathbf{x}_t$ depends on the previous $\mathbf{x}_{t-1}$ by the density $\ell(\cdot)$—generally through a mean function $\boldsymbol{\ell}_\mu(\mathbf{x}_t)$—in the state equation as follows. The speeds v_t at each time t are independent of the previous speed v_{t-1} and depend only on the behavior λ_t at that point. The bearing angle (direction) ψ_t depends on the previous angle ψ_{t-1} through the turn angle θ_t. The coordinates $\boldsymbol{\zeta}_t$ depend on the previous coordinates, speed, and bearing at $t-1$ in a straightforward way.

Given the state $\mathbf{x}_t$, our SSM will predict $\mathbf{y}_t = \boldsymbol{z}_{t+1} = \begin{bmatrix}z_{1,t+1} & z_{2,t+1}\end{bmatrix}$, the observed coordinates at time $t+1$, through the measurement equation function $\boldsymbol{m}_\mu(\cdot)$ using the same method as in $\boldsymbol{\ell}_\mu(\cdot)$:

$$\mathbf{x}_t = \begin{bmatrix} \zeta_{1,t} \\ \zeta_{2,t} \\ \ln(v_t) \\ \psi_t \end{bmatrix} = \boldsymbol{\ell}_\mu \left(\begin{bmatrix} \zeta_{1,t-1} \\ \zeta_{2,t-1} \\ \ln(v_{t-1}) \\ \psi_{t-1} \end{bmatrix} \right) + \mathbf{q}_t =$$

$$\begin{bmatrix} \zeta_{1,t-1} + v_{t-1}\Delta_{t-1}\cos(\psi_{t-1}) \\ \zeta_{2,t-1} + v_{t-1}\Delta_{t-1}\sin(\psi_{t-1}) \\ \ln(v_t) \\ \psi_{t-1} + \theta_t \end{bmatrix} + \mathbf{q}_t$$

$$\mathbf{y}_t = \begin{bmatrix} z_{1,t+1} \\ z_{2,t+1} \end{bmatrix} = \boldsymbol{m}_\mu \left(\begin{bmatrix} \zeta_{1,t} \\ \zeta_{2,t} \\ \ln(v_t) \\ \psi_t \end{bmatrix} \right) + \mathbf{r}_t = \begin{bmatrix} \zeta_{1,t} + v_t\Delta_t\cos(\psi_t) \\ \zeta_{2,t} + v_t\Delta_t\sin(\psi_t) \end{bmatrix} + \mathbf{r}_t$$

$$\mathbf{q}_t \sim \mathcal{N}_4(\mathbf{0}, \mathbf{Q}_t), \quad \mathbf{r}_t \sim \mathcal{N}_2(\mathbf{0}, \mathbf{R}_t) \tag{5.1}$$

Here $\mathcal{N}_k$ denotes a k-variate normal distribution. In $\boldsymbol{\ell}_\mu(\cdot)$ and $\boldsymbol{m}_\mu(\cdot)$, technically the function relating location and log-speed $\ln(v_t)$ should be written as $\exp(\ln(v_t))$, for instance, since log-speed is actually the variable, but we have omitted this for clarity.

The Jacobian matrices $\mathbf{L}$ and $\mathbf{M}$ of the SSM equation functions $\boldsymbol{\ell}_\mu(\cdot)$ and $\boldsymbol{m}_\mu(\cdot)$ are:

$$\mathbf{L}(\mathbf{x}_{t-1}) = \begin{bmatrix} 1 & 0 & v_{t-1}\Delta_{t-1}\cos(\psi_{t-1}) & -v_{t-1}\Delta_{t-1}\sin(\psi_{t-1}) \\ 0 & 1 & v_{t-1}\Delta_{t-1}\sin(\psi_{t-1}) & v_{t-1}\Delta_{t-1}\cos(\psi_{t-1}) \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{M}(\mathbf{x}_t) = \begin{bmatrix} 1 & 0 & v_t\Delta_t\cos(\psi_t) & -v_t\Delta_t\sin(\psi_t) \\ 0 & 1 & v_t\Delta_t\sin(\psi_t) & v_t\Delta_t\cos(\psi_t) \end{bmatrix}$$

5.2 Wrapped normal distribution

In order to use the EKF formulation and its recursive update formulas, which are based on the Delta Method's first-order linear approximation, the underlying variables in our model must be formulated to have a multivariate normal distribution. As mentioned before, we model the log-speed $\ln(v_t)$ using a normal distribution and then convert to raw speed to predict future locations. The turn angle θ_t and bearing angle ψ_t, however, require slightly modified treatment because unlike the normal distribution which has support $\mathbb{R}^d$, these variables are restricted to the interval $[-\pi, \pi]$. This restriction requires the use of circular (wrapped) distributions, which are distributions d with domains $x \in [-\pi, \pi]$ and for which the statistics and densities are the same at the boundaries $\pm\pi$, since these are equivalent angles. Such distributions include the wrapped normal, wrapped Cauchy, and von Mises (which resembles the normal).

Like their respective unwrapped versions, the wrapped densities have a mean $\mu \in [-\pi, \pi]$ (technically, we can have $\mu \in \mathbb{R}$, but it is easier to assume μ is restricted to the function's domain) and a parameter $\sigma > 0$ or $\kappa > 0$ that controls the spead or concentration within the interval. Letting $\sigma \to \infty$ for the normal, for instance, causes it to converge to $\mathcal{U}(-\pi, \pi)$, also known as a circular uniform distribution. A uniform angular distribution would not generally be appropriate for animal models because in our models the turn angle tends to unimodal and be centered around a value (usually 0), which causes directional persistence in correlated random walks (CRWs, see Section 3.2). A circular unform distribution means any direction is equally likely—as is the case in a simple random walk, where bearings ψ_t are serially uncorrelated—which we dismissed as unrealistic for most animals.

For modeling the angle distribution, we use the wrapped normal (WN) because it fits naturally with the multivariate normal setup of the EKF. The probability density function of the WN distribution $\mathcal{WN}(\mu, \sigma)$ with domain $-\pi \leq x \leq \pi$, location parameter $\mu \in [-\pi, \pi]$ and concentration parameter $\sigma > 0$ is given by

$$d(x \mid \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \sum_{k=-\infty}^{\infty} \exp\left(-\frac{((x + 2k\pi) - \mu)^2}{2\sigma^2}\right) \tag{5.2}$$

Here, μ and σ are the mean and standard deviation parameters of the unwrapped distribution. Define the wrapping function $\Theta(x^*) = x$, where x is the result of adding or subtracting 2π to $x^* \in \mathbb{R}$ until $x^* \in [-\pi, \pi]$; if $x^* \in [-\pi, \pi]$ already, then $\Theta(x^*) = x^*$. A value $x \sim \mathcal{WN}(\mu, \sigma)$ can be drawn randomly from the wrapped distribution by first drawing $x^* \sim \mathcal{N}(\mu, \sigma)$ from the unwrapped distribution and then applying the wrapping function $\Theta(\cdot)$.

The density function above for the WN relects the fact that for an unwrapped angle $x^* = x + 2k\pi$, for an $x \in [-\pi, \pi]$ and any integer k, is angularly equivalent to the wrapped $x = \Theta(x^*)$, due to the 2π-periodicity of the unit circle in radians. Consider the mean parameter $\mu \in [-\pi, \pi]$; if $x \in [-\pi, \pi]$, then $x^* = x + 2k\pi$ for integers k of increasing magnitude becomes increasingly far from the wrapped value x, and thus $\exp\left(-(x^* - \mu)^2\right) \to 0$. Since the (unwrapped) normal distribution has unrestricted domain, angular values $x^* \notin [-\pi, \pi]$ in the unwrapped normal that are equivalent to a given $x \in [-\pi, \pi]$ are still possible but have increasingly small density. Hence, a complete representation of the WN density requires all integers k, but for practical purposes, the infinite sum in Equation 5.2 can be replaced with a sum over integers $|k| < K$ for $K > 0$ sufficiently large, assuming σ is not unreasonably large.

Figure 5.1 shows examples of WN densities $\mathcal{WN}(\mu, \sigma)$ for various values of μ and σ. For $\mathcal{WN}(\mu = 0, \sigma = 1)$ in this figure, the density is nearly the same as an unwrapped normal because most of the unwrapped normal domain falls in $[-\pi, \pi]$ anyhow. Once $\sigma = 2$, more values of the unwrapped $\mathcal{N}(\mu = 0, \sigma = 2)$ fall outside this interval, so when wrapped, these densities near the domain boundaries $\pm\pi$ increase relative to the unwrapped normal. Even though unwrapped normal draws can be wrapped to draw WN samples, this feature makes using the unwrapped normal *density function* itself a bad approximation for the WN density function when σ is large. As Kurz et al. ([29]) show, the Kullback-Leibler distance—which measures the dissimilarity between two probability density functions—between the wrapped and unwrapped normal densities with the same μ and σ, diverges when σ increases.

The divergence between the wrapped and unwrapped normal density functions motivates research into how to properly use the wrapped normal and other circular distributions in filtering applications, since they are so important in applications such as navigation and robotics (see Section 4.9). Kurz, Gilitschenski, and Hanebeck are three researchers who have released a series of papers on this topic. In [30], they review other approaches to filtering with SSM involving circular distributions; many of them use density approximations that may result in poor predictions, or they are limited because the SSMs they formulate are simplistic in that their state or measurement equations are identity relations (i.e., $\mathrm{E}(\mathbf{x}_{t+1}) = \mathrm{E}(\mathbf{x}_t)$). The main difficulty in these methods is that they often attempt to use Kalman filter update formulas, which are based on the normal distribution, which doesn't apply if the variable is wrapped normal. An example is in [49], which proposes a univariate SSM for an angle θ_t; the normal approximation in the KF is used for simplicity, but a correction using the the discrete approximation of the wrapped normal (Equation 5.2) is used for the Kalman Gain update (see Section A.2) of the state $\mathbf{x}_t$ covariance matrix so the KF recursive updates can be used. Unfortunately, the state

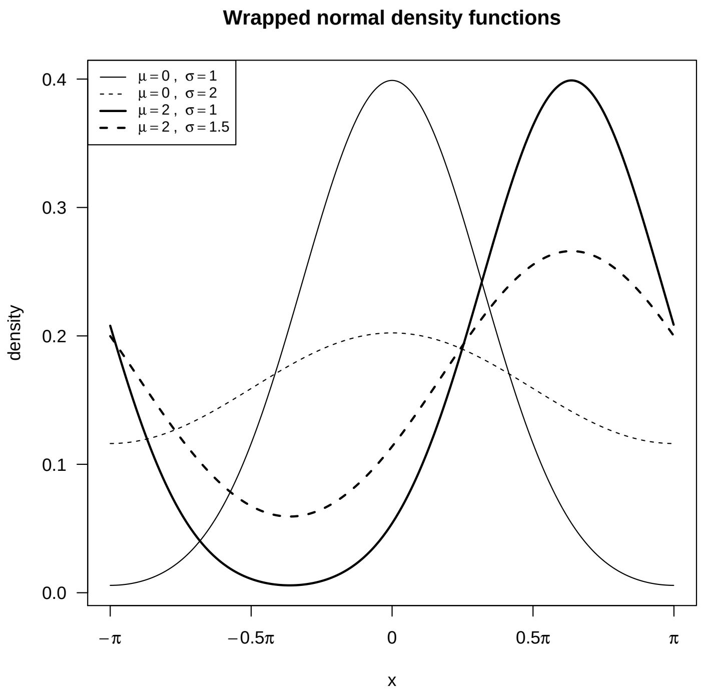


Figure 5.1: Wrapped normal distribution densities $\mathcal{WN}(\mu, \sigma)$ for various combinations of the parameters. Note that if the mean μ is close to the domain boundaries $\pm\pi$, or σ is large, the unwrapped normal $\mathcal{N}$ density becomes a bad approximation of the wrapped density $\mathcal{WN}$ due to the wrapping at the boundaries.

equation update for $\mathbf{x}_{t+1} \mid \mathbf{x}_t$ still uses the normal approximation for what is really a WN distribution.

Kurz et al. ([30] and [29]) propose a complex solution for filtering with SSM noise distributions and state/measurement equations involving circular distributions without resorting to the unwrapped normal approximation, which they say is a naive approach resulting in issues with mis-identification of the density due to the wrapping. The unwrapped normal distribution is closed under both multiplication and addition, meaning multiplying or adding two normal densities—operations that are required for the filtering equations—results in another normal density. In contrast, adding WN-distributed noise to a variable with a WN distribution does not result in another WN distribution, but if they are multiplied, a WN density results. The von Mises (VM) distribution, another circular distribution, has exactly the opposite pair of properties as the WN.

They thus propose an algorithm to convert the WN to von Mises and then back to update the WN parameters, thus bypassing the unwrapped normal approximation. In the case of an identity state relation $\mathbf{x}_t = \mathbf{x}_{t-1} + \mathbf{q}_t$ (where the noise $\mathbf{q}_t \sim \mathcal{WN}(\mu, \sigma)$), as [29] show, the resulting conditional density of $\mathbf{x}_t$ is $\mathcal{WN}(\mathbf{x}_{t-1} + \mu, \sigma)$, simply a location shift of the noise distribution, and thus no transformation to the von Mises is required. This is the case for our bearing angle ψ_t, which evolves by (linear) relation of WN-distributed turn angle noise. They show an example of a model of a robot arm where the state variable angle θ_t evolves by nonlinear trigonometric functions, in which case the von Mises transformation is required to model the resulting WN distribution of the new θ_t.

In our EKF setup (Equation 5.1), the state equation involves adding WN-distributed noise (the turn angle θ_t, which may have nonzero mean) to a variable with a WN distribution (the previous bearing ψ_t) to obtain the new bearing ψ_t, which should have a WN distribution as well since it is a circular variable. Therefore, we can estimate the WN distribution of the turn θ_t (and thus the bearing) directly without Kurz et al.'s transformation to von Mises, since this transformation is not needed when the state equation $\mathbf{x}_{t+1} \mid \mathbf{x}_t$ involves a linear transformation (such as simple addition) of the WN-distributed variable. We adapt the technique of Traa and Smaragdis ([49]), which estimates the WN distribution of their state variable θ_t by modifying the KF update equations to account for the wrapping aspect. For a sample $\{x_i^*\}_{i=1}^n \sim \mathcal{N}(\mu, \sigma)$, the wrapped version of the sample, $\{x_i\}_{i=1}^n = \{\Theta(x_i^*)\}_{i=1}^n \sim \mathcal{W}\mathcal{N}(\mu, \sigma)$, will not have the same sample variance and mean (sufficient statistics of the sample usually used to update the posteriors) if any of the original unwrapped sample $\{x_i^*\}_{i=1}^n$ fall outside of the interval $[-\pi, \pi]$. Thus the normal posterior update formulas cannot be used directly, as the summary statistics of the wrapped sample will not reflect the underlying spread σ of the wrapped distribution. However, care must be taken so that the standard deviation σ does not get over-estimated, leading to the distribution being mis-identified.

Let $\theta_t \sim \mathcal{W}\mathcal{N}(\mu, \sigma)$, so $\mathrm{E}(\theta_t) = \mu_t$, and the state equation be $\theta_t = \mathbf{L}\theta_{t-1} + \mathbf{q}_t$; for the EKF, we would have $\theta_t = \boldsymbol{\ell}_\mu(\theta_{t-1}) + \mathbf{q}_t$, replacing matrix multiplication with a function $\boldsymbol{\ell}_\mu$. Traa and Smaragdis' wrapped KF estimates the new mean as $\mathrm{E}(\theta_t \mid \mu_{t-1}) = \mu_t = \Theta(\mathbf{L}\mu_{t-1})$, applying the KF linear transformation to estimate the new mean, and then wrapping if necessary to ensure $\mu_t \in [-\pi, \pi]$; this has the additional benefit of helping protect the variance from being over-estimated. Their measurement equation is $\mathbf{y}_t = \mathbf{M}\theta_t + \mathbf{r}_t$. To estimate the density of the observation $\mathbf{y}_t$, we use the Kalman Gain (see Section A.2), which involves calculating the prediction error $\mathbf{y}_t - \mathbf{M}\mu_t$ (also called 'innovation'). Since the state and equation are both circular, the raw difference does not capture the true difference, which would be $\Theta(\mathbf{y}_t) - \Theta(\mathbf{M}\mu_t)$, but also does not account for the fact that any predictions $\hat{\mathbf{y}}_t = \mathbf{M}(\mu_t) \pm 2\pi$—if $\hat{\mathbf{y}}_t \notin [-\pi, \pi]$— when wrapped are equivalent to a $\hat{\mathbf{y}}_t = \mathbf{M}(\mu_t) \in [-\pi, \pi]$. Thus, the Kalman Gain formula needs to be adjusted to reflect the the posterior variance of the wrapped measurement in the case that the variance is large or the mean μ_{t-1} is near the boundary $\pm\pi$. To adjust for the wrapping in the Kalman Gain, they create weights $\eta_{t,k} = \mathcal{N}(\mathbf{y}_t + 2k\pi \mid \mathbf{M}\mu_t, \mathbf{r}_t)$—the densities of the prediction $\mathbf{y}_t$ shifted by several integer multiples of 2π on either side, normalized to have a sum of 1—which are then used to weight the differences $(\mathbf{y}_t + 2k\pi) - \mathbf{M}\mu_t$ in the Kalman Gain calculation.

In our EKF setup for parameter updates (see Section 5.4) we will place a normal prior $\mu_c^{(n)} \sim \mathcal{N}(\mu_c, V_c\sigma_c^2)$ on the mean of the turn angle $\theta_c^{(n)} \sim \mathcal{WN}\left(\mu_c^{(n)}, \sigma_c^{(n)}\right)$; the variance $\left(\sigma_c^{(n)}\right)^2$ will also have an inverse Gamma distribution (discussed later). We will update the posterior of the distribution of μ (which may be done after several steps rather than at each step) based on simulated particle values $\theta_c^{(n)}$. The update will proceed as follows:

- Calculate WN weights $\eta_{c,k} = \mathcal{N}\left(\theta_c^{(n)} + 2k\pi \mid \mu_c^{(n)}, \sigma_c^{(n)}\right)$, for $k = -K, \ldots, K$ for a sufficient integer K (2 or 3 is enough in general). Normalize the weights by dividing by the sum $\sum_{k=-K}^{K} \eta_{c,k}$.
- The updated posterior mean turn angle $\mu_c \mid \theta_c^{(n)}$ involves $\theta_c^{(n)}$. Rather than use the raw value $\theta_c^{(n)}$ in the calculation, we use a weighted adjustment $\hat{\theta}_c^{(n)} = \sum_{k=-K}^{K} \eta_{c,k}(\theta_c^{(n)} + 2k\pi)$.
- The posterior variance of the turn angle $\theta_c^{(n)}$ update involves its square $\left(\theta_c^{(n)}\right)^2$, so let the weighted approximation be $\left(\hat{\theta}_c^{(n)}\right)^2 = \sum_{k=-K}^{K} \eta_{c,k}(\theta_c^{(n)} + 2k\pi)^2$, by weighting the squared values rather than squaring the weighted adjustment $\hat{\theta}_c^{(n)}$.

To illustrate that this update formulation works successfully, we run the following simulation:

- Simulate $C = 100$ (unwrapped) sample values $\{\theta_i^*\}_{c=1}^{C} \sim \mathcal{N}(\mu, \sigma = 1.5)$, where $\mu = 0$ or -2, and apply the wrapping function to obtain $\{\theta_i\}_{c=1}^{C} = \{\Theta(\theta_i^*)\}_{c=1}^{C} \sim \mathcal{WN}(\mu, \sigma = 1.5)$. A reasonably-sized σ and nonzero mean μ ensures that the wrapping will affect the distribution of sample values θ compared to that of the unwrapped θ^*.
- Initialize priors $\sigma^2 \sim \mathcal{G}^{-1}(a_0 = 6,\ b_0 = 4)$ and $\mu \sim \mathcal{N}(\mu_0 = -2, V_0\sigma^2)$, where $\mu_0 = 0$ or -2 and $V_0 = 4$.

- Update the posteriors in blocks of 3 observations, since our PF should be able to update posteriors in blocks in addition to at each step c. We will
  - use the wrapped observed data $\{\theta_1, \theta_2, \theta_3\}$ etc. to update the μ and σ for the posterior of θ assuming (1) an unwrapped normal, and (2) a WN distribution with the weighting formulations above for updating the mean. And
  - use the un-wrapped version $\{\theta_1^*, \theta_2^*, \theta_3^*\}$ of the observed data, and update the posterior of θ assuming an unwrapped normal posterior. Note that this is an unrealistic scenario because when observing angular measurements they are wrapped, so we do not know for certainty where in the unwrapped normal distribution (i.e., $\theta \pm 2k\pi$) the wrapped value comes from. This is the challenge of working with wrapped distributions.

Figure 5.2 shows density estimates for the wrapped normal θ given the posterior parameters of μ and σ, estimated by either

1. assuming an (unwrapped) normal distribution for the observed wrapped θ and following the normal-inverse gamma updates (thin dashed line),
2. modeling the wrapped θ as a WN distribution and updating the posteriors according to the procedure above (thin solid line), or
3. using the unwrapped observed θ^*, which is unrealistic, and updating the posterior according to usual normal distributions, and then simulating from this posterior and wrapping (thick dashed line).

The thick black line shows the density estimate of the sampled θ. As we see, if θ has mean $\mu = 0$, the normal and WN approaches are both comparable, because most of the density $\theta \sim \mathcal{WN}(\mu, \sigma)$ is already in the interval $[-\pi, \pi]$. However, for $\mu = -2$, the wrapping of the θs makes the unwrapped normal estimate perform weakly, whereas the WN approach works very well; as we see,

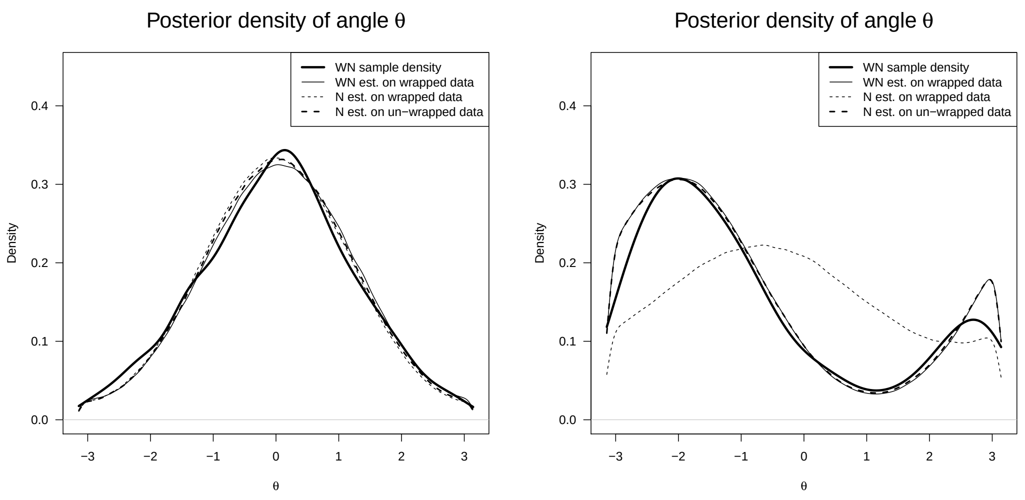


Figure 5.2: Density estimates of $\theta \sim \mathcal{WN}(\mu, \sigma)$ for $\mu = 0$ (left) and $\mu = -2$ (right), where the posteriors of μ and σ are estimated either assuming θ is unwrapped (N) or wrapped normal (WN). The WN model greatly outperforms the normal for a nonzero mean μ or if σ is relatively large.

the normal density in the right figure mostly fails to capture the bimodality of the density around π, which is caused by the wrapping and the nonzero mean. However, if $\mu = 0$ and σ was higher, the WN would still outperform the normal. However, it is important to for the prior distribution of μ to be close to the true value; for the shark turn angle it is reasonable to assume $\mu = 0$, but the model should be flexible. The success of the WN estimation on the wrapped observations vs the normal estimation on the unwrapped observations seems to also be sensitive to the value of the parameter V_0 in the NIG joint prior.

5.3 Shark EKF CDLM formulation

Following the setup of the CDLM, at each regular interval $c = 0, \ldots, C$ the shark's movement is conditioned on the behavior variable λ_c. We thus want to model $\mathbf{x}_c \mid (\mathbf{x}_{c-1}, \lambda_c)$ and $\mathbf{y}_t \mid (\mathbf{x}_c, \lambda_c)$ if there is an observation in the interval $(c, c+1]$, where the length of each time interval c is a constant Δ_Υ. At each c, the unobserved state variables $\mathbf{x}_c$ have the following conditional distribution, composed of a tri-variate normal distribution ($\mathcal{N}_3$) and a univariate wrapped normal ($\mathcal{WN}$, discussed in Section 5.2), which are independent.

$$\left[\begin{array}{c|c} \zeta_{1,c} & \\ \zeta_{2,c} & \lambda_c \\ \hline \ln(v_c) & \end{array}\right] \sim \mathcal{N}_3\left(\left[\begin{array}{c} \zeta_{1,c-1} + v_{c-1}\Delta_\Upsilon \cos(\psi_{c-1}) \\ \zeta_{2,c-1} + v_{c-1}\Delta_\Upsilon \sin(\psi_{c-1}) \\ \hline \alpha_{\lambda_c} \end{array}\right], \left[\begin{array}{cc|c} \multicolumn{2}{c|}{\Sigma_{\zeta_c}} & 0 \\ & & 0 \\ \hline 0 & 0 & \sigma^2_{\lambda_c} \end{array}\right]\right)$$

$$\left[\begin{array}{c|c} \psi_c & \lambda_c \end{array}\right] \sim \mathcal{WN}\left(\left[\psi_{c-1} + \theta_c \right], \left[\tau^2_{\lambda_c} \right]\right)$$

For the 1-D robot, Figure 4.3 visually illustrated how the CDLM (in this case, with one behavioral mode) learns parameter values. In that illustration, for each particle n, the location $\zeta^{(n)}_{t-1}$, was simulated. There, the mean velocity parameter α for velocity v_t was learned by simulating the next location $\zeta^{(n)}_t \mid \left(\zeta^{(n)}_{t-1}, v^{(n)}_{t-1}\right)$, evaluating the prediction density for the next observed location $z_{t+1} \mid \left(\zeta^{(n)}_t, v^{(n)}_t\right)$, given the measurement noises $\mathbf{r}^{(n)}_t$, where the new velocity $v^{(n)}_t$ was simulated using each particle's estimate of the mean velocity $\alpha^{(n)}$. The weights $w^{(n)}_t$ represented each particle's predictive density evaluated at the observed value z_{t+1}, where higher weight means the particle's parameters better predict what was observed. Ultimately the value of α (along with other potential parameters) is learned by resampling these particles by their weights.

Figure 5.3 shows a toy illustration of the two-behavior CDLM for a shark and one particle; multiple particles were omitted for clarity. For generality, this figure uses subscripts t for both observations $\mathbf{y}_t$ and states $\mathbf{x}_t$. In the 1-D robot we assumed the observations $\mathbf{y}_t$ are recorded at constant-length intervals Δ_Υ, which is not true in general. Later, in Section 7.1 we discuss the details of the interpolation model where observations $\mathbf{y}_t$ occurring at irregular times are used to simulate the trajectory at constant-length intervals $(\Upsilon_c, \Upsilon_{c+1}]$. Figure 7.3 is the corresponding image with interpolation for Figure 5.3 below.

The CDLM algorithm including the latent variable λ, here the behavior, is illustrated in Section 4.4. At time $t-1$, $\mathbf{x}_{t-1}$, defining movement

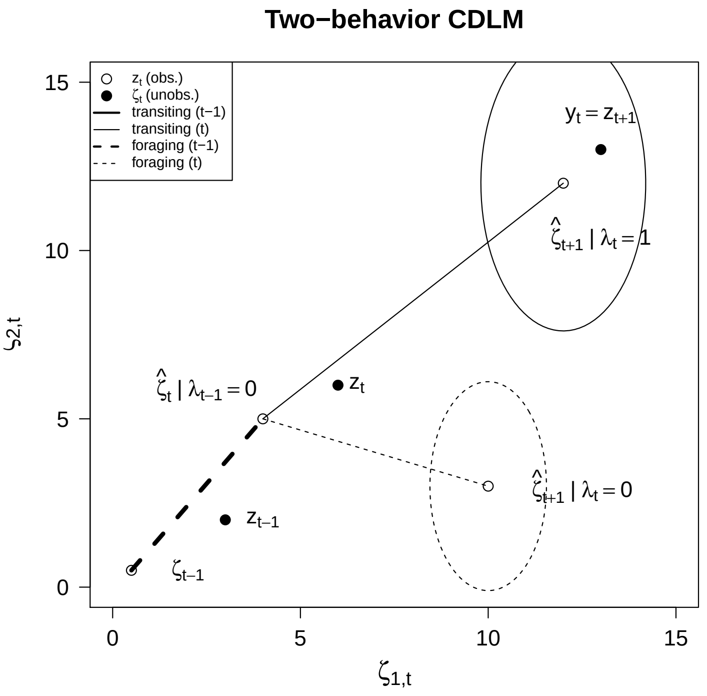


Figure 5.3: Toy illustration of one particle predicting the next observed location $\mathbf{y}_t = \boldsymbol{z}_{t+1}$ given simulated locations at ζ_{t-1}. Note the shark travels further when transiting, and at smaller turn angles, compared to foraging.

from location $\boldsymbol{\zeta}_{t-1}$, has already been simulated. The algorithm marginalizes over the intermediate $\boldsymbol{\zeta}_t \sim \ell(\cdot \mid \mathbf{x}_{t-1}, \boldsymbol{\theta}_{t-1})$, where the prediction estimate is $\hat{\boldsymbol{\zeta}}_t = \mathrm{E}(\boldsymbol{\ell}_\mu(\mathbf{x}_{t-1}))$. The algorithm then marginalizes over potential movement parameters, particularly speed and turn angle $(v_t,\ \theta_t) \mid (\lambda_t = k)$ for each potential behavior type k at time t. In the figure, these are shown by the pair of straight lines, since we assume the shark moves in a constant direction at each time t, forking out from location $\hat{\boldsymbol{\zeta}}_t$. The predictive density m_{predict_k} for each behavior, with error covariance shown by the error ellipses around each location $\hat{\boldsymbol{\zeta}}_{t+1} \mid (\lambda_t = k)$, is then evaluated at the observed value $\mathbf{y}_t = \boldsymbol{z}_{t+1}$. The behavior k with highest value $w_{t|k} \propto m_{\text{predict}_k}(\mathbf{y}_t) p_{\lambda_{t-1} \to k}$ is the most likely. Furthermore, particles n whose overall weight $w_t^{(n)} = \sum_k w_{t|k}^{(n)}$ have the highest overall likelihood and therefore the best overall predictive accuracy at that step t.

As an aside, we note that the PF and SSM tools are extremely general. We have chosen the movement model for sharks that focuses on the turn angles and speeds as the main features determining the behavior type. We have also chosen a two-behavior model for its simplicity, but also because we believe it adequately captures shark biology, but the setup can be extended to more behaviors. As an example of a different type of animal movement model, we turn to the continuous-time model in Johnson et al. ([21]), which they apply to movement of harbor seals. As they note, the CRW model is inappropriate for harbor seals because they are not in motion continuously, but rather they occasionally 'haul out' of the water onto land. Thus, we cannot describe the seal's behavior on land, during which it is stationary for periods of time, in terms of speeds and turn angles, as we could when it is in the water.

The CDLM setup can be tweaked to accommodate this setup. In our EKF specification of the SSM, we have a single state equation function $\boldsymbol{\ell}_\mu$ that governs the conditional distribution of $\mathbf{x}_c \mid \mathbf{x}_{c-1}$, in our case in terms of the turn angle and speed from the previous location $\boldsymbol{\zeta}_{c-1}$. The state equation function $\boldsymbol{\ell}_\mu$ is the same for both behaviors λ_c, but the behaviors different in terms of the parameters of the speed v_c and turn angle θ_c distributions. A potential difficulty is that the states $\mathbf{x}_c$ must contain the same set of state variables (speed, bearing, etc.) at each step c regardless of the behaviors λ_c. Assuming these states, or at least a subset of them, are meaningful for each behavior λ_c (for instance, haul out vs mobile for the seal) a different state equation function $\boldsymbol{\ell}_\mu$ and measurement equation function $\boldsymbol{m}_\mu$ that has a different *functional form* of $\mathbf{x}_{c-1}$ at each behavior λ_c, and not just different parameter values $\boldsymbol{\theta}_c$, can be specified, to define movement for each behavior. Then each observation $\mathbf{y}_t$ (for instance, an observed location $\boldsymbol{z}_{t+1}$, as in our setup) can be predicted from state $\mathbf{x}_c$ for each behavior λ_c using its separately-defined measurement equation $\boldsymbol{m}_\mu$.

5.4 Prior setup for parameter means and variances

The transformation of log-speed $\ln(v_c)$ to normality is to use the CDLM and thus EKF formulations, which depend on the variables having underlying normal distributions so that SSMs of them are normal.

We assign the means and variances of $\ln(v_c)$ independent normal-inverse-gamma (NIG) priors, again depending on the behavior λ_c. The turn angle θ_c is modeled by a wrapped normal (WN) distribution, which, because it has the same form as an unwrapped normal, can use the same prior distribution setup. The posterior update formulas are shown in Section A.2. As noted in Section 5.2, the update formulas are modified for the wrapped normal distribution of the turn angle θ_c.

$$\begin{aligned}
\ln(v_c) &\sim \mathcal{N}(\alpha_{\lambda_c},\ \sigma^2_{\lambda_c}) \\
p(\alpha_{\lambda_c},\ \sigma^2_{\lambda_c}) &\sim \mathcal{N}(\alpha_{\lambda_c} \mid \alpha_{c,\lambda_c},\quad \kappa_{1_{c,\lambda_c}}\sigma^2_{\lambda_c}) \quad\times\quad \mathcal{G}^{-1}(\sigma^2_{\lambda_c} \mid \ a_{c,\lambda_c},\ \ b_{c,\lambda_c}) \\
\theta_c &\sim \mathcal{W}\mathcal{N}(\beta_{\lambda_c},\ \tau^2_{\lambda_c}) \\
p(\beta_{\lambda_c},\ \tau^2_{\lambda_c}) &\sim \mathcal{W}\mathcal{N}(\beta_{\lambda_c} \mid \beta_{c,\lambda_c},\ \ \kappa_{2_{c,\lambda_c}}\tau^2_{\lambda_c}) \quad\times\quad \mathcal{G}^{-1}(\tau^2_{\lambda_c} \mid \ c_{c,\lambda_c},\ \ d_{c,\lambda_c})
\end{aligned}$$

Note: a variable X follows the inverse-gamma distribution $\mathcal{G}^{-1}(a,b)$ if $X = Y^{-1}$ and Y is gamma-distributed $\mathcal{G}(a,b)$. The inverse-gamma density is as follows:

$$f(x \mid a,b) = \frac{b^a}{\Gamma(a)} x^{-(a+1)} e^{(-b/x)}, \quad x,a,b > 0, \text{ with mean } \mathrm{E}(X) = \frac{b}{a-1}, \quad a > 1.$$

For all the prior distributions which depend on behavior λ_c, parameters are only updated for the prior for the λ_c simulated to have occurred at time t.

Equation 5.3 showed that the four components of $\mathbf{x}_t$ were modeled as two blocks

$$\boldsymbol{\zeta}_c = \begin{bmatrix}\zeta_{1,c} & \zeta_{2,c} & \ln(v_c)\end{bmatrix}^T \quad \text{and} \quad \begin{bmatrix}\psi_c\end{bmatrix}$$

the first of which is distributed tri-variate normal, with log-speed independent of the other two components, and the bearing ψ_c has a WN distribution. Focusing now on the unobserved spatial coordinates in the first block, we thus have

$$\begin{bmatrix}\zeta_{1,c} \\ \zeta_{2,c}\end{bmatrix} \sim \mathcal{N}_2\left(\begin{bmatrix}\zeta_{1,m-1} + v_{c-1}(\Delta_\Upsilon)\cos(\psi_{c-1}) \\ \zeta_{2,m-1} + v_{c-1}(\Delta_\Upsilon)\sin(\psi_{c-1})\end{bmatrix}, \Sigma_{\zeta_c}\right)$$

$$\Sigma_{\zeta_c} \sim \mathcal{W}_2^{-1}(\Lambda_{\zeta_c}, \eta_{\zeta_c})$$

This distribution, unlike that of the movement parameters $\ln(v_c)$ and ψ_c, is independent of the behavior λ_c. This means that knowing the coordinates $\boldsymbol{\zeta}_{c-1}$ and the movement from there in terms of the speed v_{c-1} and bearing ψ_{c-1}, the error of this prediction to the next true unobserved coordinates $\boldsymbol{\zeta}_c$ does not depend on the behavior λ_c (it may depend on the previous behavior λ_{c-1}, on which the movement parameters depend). This is sensible because the coordinates in either case are unobserved.

As opposed to the variances σ^2 and τ^2 of the log-speed and bearing, which are modeled independently, the elements of the covariance matrix Σ_{ζ_c} are not modeled individually, but rather the entire matrix has a prior distribution. This prior is called an inverse-Wishart distribution, a multivariate analogue of the inverse-Gamma distribution. A square $p \times p$ matrix $\mathbf{X}$ that follows the inverse-Wishart distribution $\mathcal{W}_p^{-1}(\boldsymbol{\Lambda}, \eta)$ is parametrized by a positive-definite square scale matrix $\mathbf{\Lambda}$ of dimension $p \times p$ and degrees of freedom $\eta > p - 1$. In our setup, the scale matrix $\mathbf{\Lambda}_{\zeta_c}$ of the prior is updated at each time step using the error sum of squares, regardless of the behavior λ_c, and the degrees of freedom ($\eta_{\zeta_c} = \eta_{\zeta_{c-1}} + 1$) reflect how many observations we have at c. One of the main functions of having this particle error between consecutive values $\boldsymbol{\zeta}_{c-1}$ and $\boldsymbol{\zeta}_c$, instead of assuming that the position $\boldsymbol{\zeta}_c$ is perfectly predicted by the values in the previous $\mathbf{x}_{c-1}$, is that in Section 7.1 we will be interpolating the shark movement to regularly-occurring time intervals, and thus simulating the shark movement over intervals without observed recordings. In absence of observations, the uncertainty of the predictions (expressed by $\mathbf{\Lambda}_{\zeta_c}$) increases over time.

Recall that $\mathbf{q}_c \sim \mathcal{N}_4(\mathbf{0}, \mathbf{Q}_c)$ is the error term for $\mathbf{x}_c | \mathbf{x}_{c-1}$. We partition $\mathbf{Q}_c$ as follows; note $\mathbf{Q}_c$ depends on λ_c only through σ^2 and τ^2, the variances of the log-speed and bearing angle variables:

The shark SSM models $\mathbf{y}_t = \begin{bmatrix} z_{1,t+1} & z_{2,t+1} \end{bmatrix}$, the observed location at the next time point $t+1$, based on the shark's unobserved movement $\mathbf{x}_c$ at time c, which depends on the behavior λ_c at that point. The prediction for the next location $\mathbf{y}_t$ depend on the behavior λ_c, since for instance, if foraging the shark is likely to go slower than if transiting. The prediction error will also depend on the behavior since for a given time gap Δ_c, if the shark is going faster, the range of possible locations is larger, and thus the error is likely to be larger with more variance. The covariance matrix $\mathbf{R}_c = \Sigma_{z_{t+1}}$ for the error term $\mathbf{r}_c$ of $\mathbf{y}_t \mid \mathbf{x}_c$ is also assigned an inverse-Wishart prior. Like $\ln(v_c)$ and ψ_c, it has a separate prior for each behavior λ_c.

$$\Sigma_{z_{t+1}} | \lambda_c \sim \mathcal{W}_2^{-1}(\mathbf{\Lambda}_{z_{t+1,\lambda_c}}, \eta_{z_{t+1,\lambda_c}})$$

The CDLM PF allows the parameters $\alpha_{\lambda_c}, \beta_{\lambda_c}$ (the log-speed and log-turn angle means), κ_{1,λ_c}, κ_{2,λ_c}, a_{c,λ_c}, b_{c,λ_c}, c_{c,λ_c}, d_{c,λ_c} (inverse-gamma parameters of their variances), $\mathbf{\Lambda}_{\zeta_c}$, η_{ζ_c}, $\mathbf{\Lambda}_{z_{t+1,\lambda_c}}$, $\eta_{z_{t+1,\lambda_c}}$ (inverse-Wishart covariance matrix parameters) to be updated recursively by tracking their sufficient statistics over time. Formulas are given in Section A.2.

5.5 Rejection sampling

Section 4.6 reviewed some of the general issues involving resampling in particle filters, such as degeneracy. Here, we will discuss some of the aspects of resampling and random sampling that are specifically relevant to our proposed EKF shark movement setup. Two pertinent aspects are how to deal with the geographic constraints of the tidal basin, and issues that arise when dealing with class imbalances in categorical variables, such as in a behavior switching model.

In the propagation step of the CDLM, $\mathbf{x}_c^{(n)}$ (which include the true unobserved coordinates $\boldsymbol{\zeta}_c$) are drawn from a multivariate normal distribution based on $\mathbf{x}_{c-1}^{(n)}$ (coordinates at $c-1$ and movement from it). Obviously, all the observed coordinates $\boldsymbol{z}$ are within the water boundaries of the tidal basin, and so must be all simulated coordinates $\boldsymbol{\zeta}$; this is known as a hard constraint. A soft constraint in optimization is when variables do not have to be within a set of constrained values but the algorithm will apply a penalty if they are not, to encourage prediction of values within the preferred constraint. For instance, LASSO regression prefers explanatory variable coefficients to be zero and only lets them be significant if there is sufficient evidence, accounting for the penalty, that they should be. When propagating, without any such constraints. we cannot ensure that the propagated $\mathbf{x}_c^{(n)}$ contains $\boldsymbol{\zeta}_c^{(n)} = \begin{bmatrix} \zeta_{1,c}^{(n)} & \zeta_{2,c}^{(n)} \end{bmatrix}$ that are within the boundary of the tidal basin. In particle filter applications, hard constraints are sometimes easily expressed in terms of inequalities. For instance, if $\mathbf{x}_c$ contains a simulated variable for a speed v_c, we need a restriction that $v_c > 0$ by definition. However, keeping the coordinates in the boundary cannot be expressed easily mathematically.

This hard restriction can be handled flexibly by sampling from a truncated distribution (see Challa and Bergman, [11]; Ristic, Arulampalam, and Gordon, [42]). Say that we require $\mathbf{x}_c \in \boldsymbol{\Psi}$, where, for instance $\boldsymbol{\Psi} = \mathbb{R}^+$ or a physical boundary. In our case, the hard restriction $\boldsymbol{\Psi}$ is such that the line connecting simulated $\boldsymbol{\zeta}_{c-1}$ and $\boldsymbol{\zeta}_c$ must fall in the boundary (to prevent the shark from crossing land between points), or for simplicity we may just require the point $\boldsymbol{\zeta}_c \in \boldsymbol{\Psi}$. In general, for a variable $\mathbf{x}_c$ with density $d(x)$, the truncated density $\widetilde{d}(x)$ where we must have $\mathbf{x}_c \in \boldsymbol{\Psi}$ is

$$\widetilde{d}(x) = \begin{cases} \dfrac{d(x)}{\int_{x \in \boldsymbol{\Psi}} d(x)\mathrm{d}x}, & x \in \boldsymbol{\Psi} \\ 0, & \text{otherwise} \end{cases}$$

That is, the truncated density function $\widetilde{d}(x)$ is just the original density $d(x)$ divided by the probability that the non-truncated $\mathbf{x}_c$ falls in the constraint $\boldsymbol{\Psi}$, for values $\mathbf{x}_c \in \boldsymbol{\Psi}$, otherwise it is 0.

To generate $\mathbf{x}_c \in \boldsymbol{\Psi}$, that is, a draw $\mathbf{x}_c \sim \widetilde{d}(x)$ from the truncated density, we do not have to deal directly with the truncated density itself and calculate the normalizing probability $\int_{x \in \boldsymbol{\Psi}} d(x)\mathrm{d}x$. Rather, all that is necessary is to draw $\mathbf{x}_c$ repeatedly from the un-truncated density $d(x)$, and select the first one satisfying the constraint. This general formulation eliminates the necessity to express complex hard constraints mathematically. Of course, if it takes many draws to generate a valid one, then perhaps the model specification is sub-optimal, or perhaps the shark is caught in a corner or narrow area, and most draws will result in a coordinate outside the boundary.

Let $\mathbf{x}_c^{(n)} = \begin{bmatrix} \zeta_{1,c} & \zeta_{2,c} & \ln(v_c) & \psi_c \end{bmatrix}^{(n)}$ be particle n's value of the unobserved state variables at time c, including the true location $\boldsymbol{\zeta}_c^{(n)}$. Let $d(\mathbf{x}_c^{(n)})$ denote the untruncated (in this case, multivariate normal $\mathcal{N}_4$) density of $\mathbf{x}_c^{(n)}$, using the notation above. The truncated density $\widetilde{d}(\cdot)$ requires only that the location be valid, that is $\boldsymbol{\zeta}_c^{(n)} \in \boldsymbol{\Psi}$, where $\boldsymbol{\Psi}$ denotes the water area of the tidal basin, and that the line connecting $\boldsymbol{\zeta}_c^{(n)}$ and $\boldsymbol{\zeta}_{c-1}^{(n)}$ fall entirely in the water. The truncation process draws from the original density $d(\cdot)$ until one result satisfies the truncation, up to maximum number of iterations; if the iteration limit is reached without a valid draw, the previous location $\boldsymbol{\zeta}_{c-1}^{(n)}$, which, recursively, must be valid because the particle filter begins at the initial observations, is returned. Occasionally, our particle filter would exceed the allowed rejection iterations, even when set at 5,000, for several particles. This is a problem because the returned value, of the previous location $\boldsymbol{\zeta}_{c-1}^{(n)}$, may be a bad prediction of the observation $\mathbf{y}_t = \boldsymbol{z}_{t+1}$, depending on how much the shark was simulated to have moved in the timestep.

One issue with sampling from the truncated density is that it may cause the algorithm to suffer if it has to draw too many rejection samples to get a valid draw in the boundary ($\boldsymbol{\Psi}$). The algorithm may get stuck in a corner, unable to predict a valid move. In the CDLM particle filter, particles receive weight $w^{(n)}$ proportional to the density they predict at the observation. However, a high density (a likely accurate prediction) may not be the only factor in determining which particle to select.

This is illustrated in Figure 5.4. Here, the shark is observed first at the black open circle, then at the black solid circle. The red star and triangle represent two particles' predictions of the location of the shark when it is observed at the solid black circle. The ellipses around each represent bivariate normal confidence intervals of, say 95%, around the prediction. Here, the red star particle generates a closer prediction to the observation than the red triangle particle does; hence, the red star would receive a higher resampling weight, since it predicts a higher density at the observation.

However, note that a large portion of the red star's confidence interval is located outside of the tidal basin boundary $\boldsymbol{\Psi}$, and hence represent invalid predictions for locations $\boldsymbol{\zeta}_c$. Thus, many more (perhaps half) of the random draws of the red star prediction, since we are truncating the draws to be inside the water, are invalid. The red triangle particle prediction, while less accurate, is more likely to result in a valid location draw. This can be an issue with particles whose mean predictions (hence the centers of the ellipses) are near the boundary; this problem is compounded if the position is in a narrow area or corner, where almost all draws will be invalid. A potential remedy is to re-weight particles by how likely their predictions are to be valid. For instance, for each particle n, make, say $k = 100$ draws from the prediction density, set $a^{(n)}$ to be the proportion of draws that result in a valid location (i.e., a

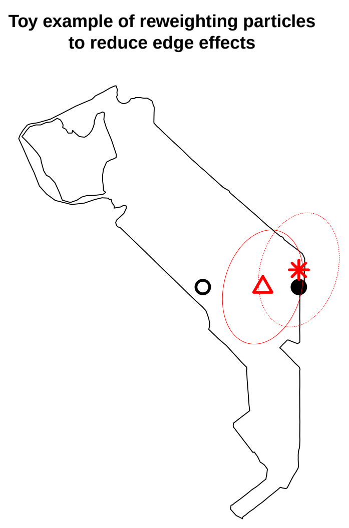


Figure 5.4: Animal is located at the black open circle, and is observed next at the right black dot. The red points represent the centers of two particles' location predictions, and the ellipses are their 95% confidence regions. The star is closer than the red triangle to the observation, and hence would receive a higher resampling weight since it has higher density at the observation. However randomly selecting a point from the star's particle is more likely to yield a draw outside the boundary or close to the edge, and hence may result in the trajectory getting 'stuck' in the corner.

rough Monte-Carlo estimate of the fraction of the density that is valid), then reweight the resampling weights $w_c^{(n)}$ by multiplying by $a^{(n)}$. This would add to the computational expense of the algorithm.

Another option to handle boundaries is to explicitly factor in obstacle avoidance to the movement model. One example is Tracey et al. ([50]), who model the animal movement angles with the von Mises distribution, a continuous distribution with support $[-\pi, \pi]$. This angle distribution has a parameter μ around which values cluster, and a concentration parameter κ (similar to precision in the normal distribution); $\kappa = 0$ makes it a uniform distribution on $[-\pi, \pi]$, and increasing $\kappa > 0$ makes the distribution more clustered around μ.

Their setup is illustrated in Figure 5.5. Given the animal location, an object (spatial feature) is located at an angle C_i radians relative to the animal, where 0 represents the positive x-axis ("east"); the angle C_i is the bearing angle the animal would travel at to reach the object. If the feature is a line or shape (e.g., the shore of the tidal basin) that is not at a single location, the point location is the closest point on the feature to the animal; if the feature is a line, the point would be given by drawing a line perpendicular to the feature from the animal. The angle A_i is the animal's change in movement angle in response to the object, and B_i is the resulting movement (bearing) angle. If the animal is attracted to the feature, $\mu = 0$ since the animal will head directly to it; if repelled, $\mu = \pi$ since the animal will turn 180 degrees in the opposite direction. The reactivity can also be modeled so the strength of the reaction increases the closer the animal is to the object. This model seems intuitive enough, but we prefer the truncated sampling approach to the problem because it fits naturally into our model and is a much more general model.

5.6 Resampling with rare behaviors

An additional aspect of resampling that is pertinent is the fact that the transiting behavior tends to be less frequent in the observed shark data, and also more difficult to model. Biologically, this is sensible since faster move-

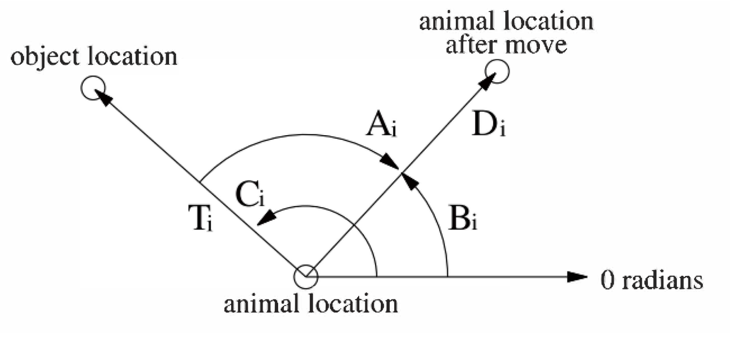


Figure 5.5: Diagram of animal movement in response to a landscape feature at angle C_i relative to 0. The feature (such as an obstacle or boundary) causes the animal to alter its trajectory by angle A_i, resulting in a bearing of angle B_i to avoid the feature. Source: Tracey et al. ([50]).

ment costs energy, and hence animals should move slowly most of the time to conserve energy. As shown in the CDLM algorithm (Section 4.4), the CDLM determines the overall particle weighs $w_c^{(n)}$ by conditioning on each behavior (in general, the latent variable λ) through the weights $w_{c|k}^{(n)}$. Each weight $w_{c|k}^{(n)}$ involves a calculation of the transition probability $\Pr(\lambda_c = k \mid \lambda_{c-1})$ into that behavior. After particles are resampled by their weights $w_c^{(n)} = \sum_{k=1}^{K} w_{c|k}^{(n)}$, the behaviors at time c are then propagated by a categorical distribution (in our bivariate case, Bernoulli) with the probabilities proportional to $\{w_{c|k}^{(n)}\}_{k=1}^{K}$. Thus, if transiting behavior is rare, meaning $\Pr(\lambda_c = 1 \mid \lambda_{c-1})$ is generally small, then particles' ability to model transiting is likely to suffer because their accuracy in this respect (measured by the likelihood) does not factor significantly into the calculation of the overall sampling weight $w_c^{(n)}$. The particles' ability to model transiting is not sufficiently rewarded to promote particles' with good predictions in this regard. Then, even if they are resampled, the transiting weights $w_{c|\lambda=1}^{(n)}$ are small, so transiting is in general less likely to be propagated as a behavior (this makes sense because it is rarer).

The problem in the case of transiting is compounded because transiting is characterized by faster movement. Therefore, over a fixed amount of time, there is more uncertainty in predicting location if the shark transits than if it forages because the shark is capable of traveling further due to its faster movement. The likelihood component $m_{\text{predict}_{k=1}}(\mathbf{y}_t \mid \lambda_c = 1, \dots)$ in the weight $w_{c|k=1}^{(n)}$ thus spreads the likelihood thinner, making the likelihood at the true location generally lower than if it were foraging, because here the shark moves a shorter distance and thus there is less uncertainty, and thus more concentration of the likelihood. The increased uncertainty and lower transition probabilities make $w_{c|k=1}^{(n)}$ tend to be lower.

In order to accurately model the sharks' behavior, we need to be relatively accurate in predicting behavior of both types. This becomes especially important in Section 6.4 where we use the relative concentrations of shark behavior types to quantify inter-shark behavioral influence; if the model inaccurately predicts behavior type, then the influence measure, as well as most other parameters, will not be accurate. Thus, to achieve better overall predictions, a particle that has successfully modeled transiting behavior, even though it is rarer, can be considered more valuable than a particle that has better success on the more common behavior.

One way to alleviate this issue is to preferentially weight the rarer behavior in resampling, but not prediction. Pitt and Shephard ([40]) recommend rescaling $w_{c|k}^{(n)}$ by $a_k = f(\mathbf{x}_{c-1}^{(n)}) > 0$, where $f(\cdot)$ is some appropriate function. In our case we will actually have $a_k = f(\mathbf{x}_c^{(n)})$. An easy choice is just to let $f(\cdot)$ be a constant function, with higher values for the rarer behavior, for instance $a_1 = 4$ for transiting and $a_0 = 1$ for foraging. The overall sampling weights are calculated as $w_c^{(n)} = \sum_k a_k w_{c|k}^{(n)}$, which overvalues the particle's ability to model the rarer state when resampling. This can be viewed as analogously to an example from data mining, where we evaluate the performance of a binary classifier on a dataset where the classes are unbalanced (one category being more common than the other). In these situations, a classifier model should not be evaluated by accuracy (i.e., the number of correct classifications) because it can achieve decent classification by simply assigning all observations to the more common class. Rather, measures such as the F_1 score or the ROC curve assess the performance by taking into account the accuracy conditional on each class. Thus, a particular particle may be more ‘valuable’ for the model if it has adequately learned movement parameters for the rarer behavior. For propagating, λ_c are sampled from the categorical (Bernoulli) distribution with the un-scaled weights $w_{c|k}^{(n)}$, since otherwise the model would imply transiting is much more common than it is.

CHAPTER 6

REGIONAL AND BEHAVIORAL INFERENCE

To analyze sharks' tendencies to have foraging as opposed to transiting behavior, we can estimate a probability transition matrix between these behaviors for each shark. For a given shark at time c, for $i, j \in \{0, 1\}$ (potential values of the behavior λ_c), define the behavior transition probability as

$$p_{c,i\to j} = \Pr(\lambda_{c+1} = j \mid \lambda_c = i)$$

and the probability transition matrix as

$$P_c = \begin{bmatrix} p_{c,0\to 0} & p_{c,0\to 1} \\ p_{c,1\to 0} & p_{c,1\to 1} \end{bmatrix}$$

At time c, the distribution $d(\lambda_{c+1} \mid \lambda_c)$ for the behavior is thus Bernoulli (in general, categorical), where $\lambda_c = 0$ is foraging and 1 is transiting:

$$\Pr(\lambda_{c+1} \mid \lambda_c = i) \sim (p_{c,i\to 0})^{1-\lambda_{c+1}} (p_{c,i\to 1})^{\lambda_{c+1}} = \prod_{j\in\{0,1\}} (p_{c,i\to j})^{x_{c+1,j}} \tag{6.1}$$

where $x_{c+1,i} = \mathrm{I}(\lambda_{c+1} = i)$.

There are two general ways we can consider of modeling these transition probabilities. One is to assume they are Markovian, namely that the probability of the next behavior depends only on the current behavior λ_c. The other is to assume the perhaps more realistic scenario that the probability of the next behavior λ_{c+1} depends on the length of time spent in the current behavior λ_c.

To test the Markovian hypothesis, we can run a test[1] formulated by Anderson and Goodman ([1]) on the contingency table of transitions between the behaviors from the spline-interpolated regular steps. The null hypothesis of the test is that the behavior transition probabilities for each shark within each region are Markovian, so a low p-value indicates that the transition probabilities are not constant within regions; if so, perhaps they change based on how long the shark has been in the current behavior. We will keep in mind that this is an exploratory analysis and we are not necessarily taking the behavior results from the spline as ground truth, which we don't have.

Table 6.1 shows the results of the test by shark and region, applied piecewise over all the sequences of moves within a region. At least according to the spline model, many of the regions have low p-values, rejecting the Markovian hypothesis, but the results are mixed. Therefore, we may want to consider a more flexible model for the transition probabilities for the actual shark data that is not Markovian. This possibility will be explored in Section 6.3.

6.1 Markovian behavioral transitions

In the case of the Markovian transition probabilities, the prior for the likelihood $d(\lambda_c)$ depends on the previous cumulative counts of transitions. Let $h \in \mathbb{Z}$ be a positive integer. For two time steps c and $c+h$, $h \in \mathbb{Z}$, we can define the cumulative between-times transition counts as

$$n_{c,c+h,i\to j} = \sum_{k=c}^{c+h-1} \mathrm{I}(\lambda_k = i \,\&\, \lambda_{k+1} = j)$$

[1]The runs test is implemented by Spedicato ([47]) in the `markovchain` package for the R language. The `verifyMarkovProperty` function needed to be modified to make it combine the piecewise results.

Shark	1	2	3	4	5	6	7	8
1	–	1.00 (21)	0.08 (1,107)	0.00 (2,803)	0.01 (1,425)	0.98 (208)	0.00 (2,832)	0.06 (169)
2	0.99 (107)	1.00 (68)	0.00 (1,130)	0.02 (1,022)	0.00 (1,427)	0.99 (27)	0.22 (1,977)	0.82 (101)
3	–	0.99 (76)	1.00 (296)	0.03 (2,431)	0.79 (1,002)	1.00 (32)	0.01 (894)	0.99 (74)
4	1.00 (15)	1.00 (55)	0.00 (1,054)	0.00 (1,313)	0.39 (583)	1.00 (5)	0.08 (111)	–
5	–	1.00 (35)	0.17 (483)	0.00 (16,193)	0.00 (5,645)	0.33 (655)	0.00 (2,686)	1.00 (441)
6	1.00 (24)	1.00 (71)	1.00 (780)	0.06 (1,430)	0.19 (1,045)	–	0.88 (709)	–
7	–	1.00 (16)	0.04 (708)	0.00 (2,283)	0.03 (1,970)	0.00 (570)	0.00 (2,070)	1.00 (49)
8	0.22 (57)	0.81 (69)	0.87 (324)	0.37 (775)	0.04 (209)	0.81 (215)	0.12 (638)	–
9	–	0.97 (52)	0.25 (729)	0.98 (561)	0.02 (541)	1.00 (90)	1.00 (435)	–
10	–	1.00 (5)	0.91 (103)	0.54 (636)	0.89 (435)	0.21 (474)	0.85 (348)	0.00 (1,894)
11	–	–	0.14 (76)	1.00 (152)	1.00 (180)	1.00 (39)	1.00 (148)	0.00 (916)
12	1.00 (6)	1.00 (19)	–	0.92 (102)	1.00 (183)	1.00 (13)	1.00 (118)	0.00 (532)
13	0.38 (59)	1.00 (54)	1.00 (83)	0.72 (145)	0.80 (149)	0.60 (304)	0.68 (179)	0.30 (278)
14	1.00 (26)	1.00 (106)	1.00 (30)	1.00 (118)	1.00 (90)	0.15 (117)	0.95 (123)	0.04 (228)
15	–	–	–	1.00 (381)	0.05 (484)	0.06 (141)	1.00 (96)	0.47 (680)
16	–	1.00 (12)	1.00 (5)	–	–	1.00 (18)	1.00 (30)	0.64 (116)
17	–	–	0.93 (42)	0.72 (118)	0.07 (288)	0.99 (135)	0.02 (166)	0.01 (600)
18	–	–	–	0.03 (1,203)	0.11 (432)	1.00 (30)	0.00 (1,107)	–
19	0.00 (1,019)	0.00 (1,212)	0.00 (1,838)	0.00 (8,326)	0.00 (832)	0.00 (1,370)	0.00 (7,226)	0.00 (6,583)
20	–	–	–	0.94 (116)	0.37 (63)	1.00 (13)	0.20 (56)	1.00 (42)
21	–	–	–	1.00 (55)	1.00 (36)	–	1.00 (14)	1.00 (6)
22	–	–	0.89 (54)	0.71 (210)	1.00 (47)	0.84 (54)	0.97 (100)	–

Table 6.1: Results of Markov test applied to guesses of behaviors λ_c from the spline interpolated regular steps from the observations (see Figure 2.5). A low p-value means the Markovian assumption of behavior change probabilities is rejected.

and the associated cumulative between-times transition counts matrix as

$$N_{c,c+h} = \begin{bmatrix} n_{c,c+h,0\to 0} & n_{c,c+h,0\to 1} \\ \\ n_{c,c+h\to 0} & n_{c,c+h,1\to 1} \end{bmatrix}$$

Note that the sum of elements in matrix $N_{c,c+h}$ is $(c+h)-c=h$, the number of steps elapsed. The cumulative transition counts matrix $N_{1,c-1}$ defines the prior on the transition probability matrix P_c for $\Pr(\lambda_{c+1} \mid \lambda_c)$. Each row of of the matrix P_c, consisting of pairs of complementary probabilities, is an independent likelihood, for which a Dirichlet distribution is defined by the corresponding row of $N_{1,c}$. A K-dimensional Dirichlet distribution is a multivariate extension of the beta distribution to K-way categorical probabilities:

$$\begin{bmatrix} p_1, \ldots, p_K \end{bmatrix} \sim \mathcal{Dir}_K(\alpha_1, \ldots, \alpha_K) \text{ with joint density } f(p_1, \ldots, p_K) \propto \prod_{k=1}^{K} p_k^{\alpha_k - 1}$$

$$0 < \alpha_k, \quad 0 < p_k < 1, \; k = 1, \ldots, K \text{ and } \sum_{k=1}^{K} p_k = 1$$

In this setup, the Dirichlet distribution is a conjugate prior, where the prior parameters are updated by the observed counts for each category. For instance, in the bivariate case ($K = 2$), as we have, each pair of transition probabilities from $\lambda_c = i$ (row $(i+1)$ of P_c), which must sum to 1, can be modeled independently with a bivariate Dirichlet distribution:

$$\begin{bmatrix} p_{c,0\to 0} & p_{c,0\to 1} \end{bmatrix} \sim \mathcal{Dir}_2\left(\alpha_{c,0\to 0}, \; \alpha_{c,0\to 1}\right)$$

$$\begin{bmatrix} p_{c,1\to 0} & p_{c,1\to 1} \end{bmatrix} \sim \mathcal{Dir}_2\left(\alpha_{c,1\to 0}, \; \alpha_{c,1\to 1}\right)$$

Letting $x_{c+1,i} = \mathrm{I}(\lambda_{c+1} = i), i \in \{0, 1\}$ as before, if the prior is applied and updated sequentially, the prior setup of transitions from behavior $\lambda_c = i$, for instance, is simply

$$\Pr(\lambda_{c+1} = i \mid \lambda_c = i, \boldsymbol{\alpha}) \propto (p_{c,i\to i})^{x_{c+1,i}} (p_{c,i\to j})^{x_{c+1,j}} \times (p_{c,i\to i})^{\alpha_{c,i\to i}-1} (p_{c,i\to j})^{\alpha_{c,i\to j}-1}$$

The posterior updates are of the αs are simply $\alpha_{c+1,i\to j} = \alpha_{c,i\to j} + x_{c+1,j}$, $j \in \{0,1\}$. Since only one of $x_{c+1,i}$ and $x_{c+1,j}$ are nonzero, depending on the behavior λ_{c+1}, the transition parameter of the behavior that occurred is just increments by 1. More generally, if updates are not performed at each step, but rather on a collection of h steps, for a positive integer h, we have the following update formula, depending on the cumulative number of each transition observed in that interval, which are the elements of $N_{c,c+h}$:

$$\alpha_{c+h,i\to j} = \alpha_{c,i\to j} + n_{c,c+h,i\to j}, \quad i,j \in \{0,1\}$$

For the Dirichlet, if $\left[p_1,\ldots,p_K\right] \sim \mathcal{Dir}_K(\alpha_1,\ldots,\alpha_K)$ then $\mathrm{E}(p_k \mid \boldsymbol{\alpha}) = \frac{\alpha_k}{\sum_{j=1}^{K}\alpha_j}$. Specifically, in the bivariate case, $\mathrm{E}(p_{c,i\to j} \mid \boldsymbol{\alpha}) = \frac{\alpha_{c,i\to j}}{\sum_{k\in\{0,1\}}\alpha_{c,i\to k}}$. It is thus reasonable to assign prior values for the Dirichlet α hyper-parameters reflecting the relative proportions of transitions we expect, and scaling them by a constant to affect amount of variance or uncertainty we want to assign to the prior.

6.2 Regional partitions and foraging probability

To see if behavioral tendency is region-dependent under the Markovian assumption—one of the major biological questions we attempt to answer—we can model these transitions on a regional basis. Regions can be defined by partitioning the tidal basin into, say $R = 10$ regions of roughly equal size, based on a Voronoi partition map.[2] Figure 6.1 shows the spatial density of the spline interpolated locations and the Voronoi partition of ten regions that we will use in modeling the observed locations.

We define $r_c = r, r \in \{1,\ldots,R\}$ if the shark begin its movement at time c to $c+1$ in region r. We will therefore model region-specific transition probability transition matrices $P_c^{(r)}$ and cumulative counts matrices $N_{c,c+h}^{(r)}$, where each of their elements $p_{c,i\to j}^{(r)}$ and $n_{c,c+h,i\to j}^{(r)}$ are as before with the additional condition

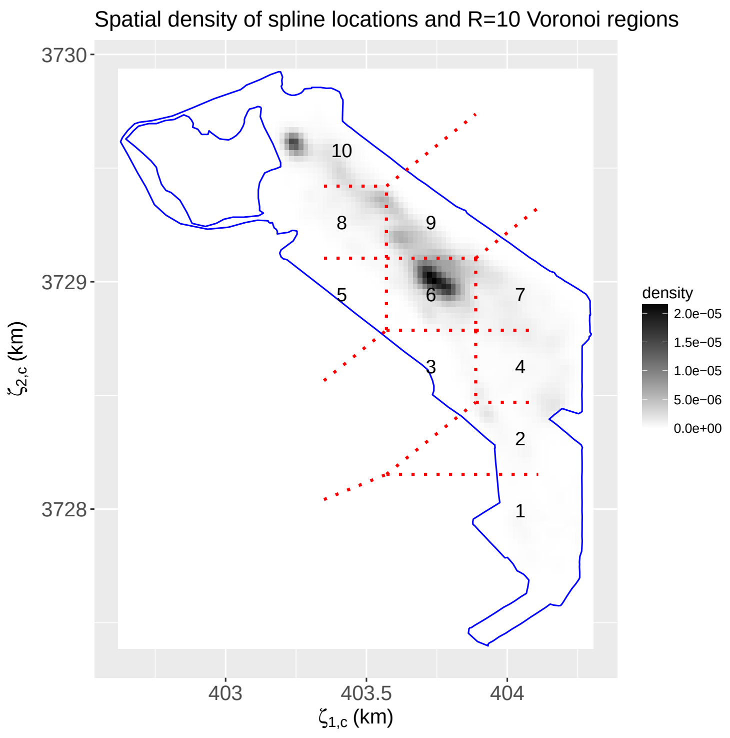


Figure 6.1: Spatial density of constant-length interval spline locations from the observed shark locations. The Voronoi partition of the tidal basin into ten regions, for each of which we estimate behavior transition probabilities in our simulations, are shown as well.

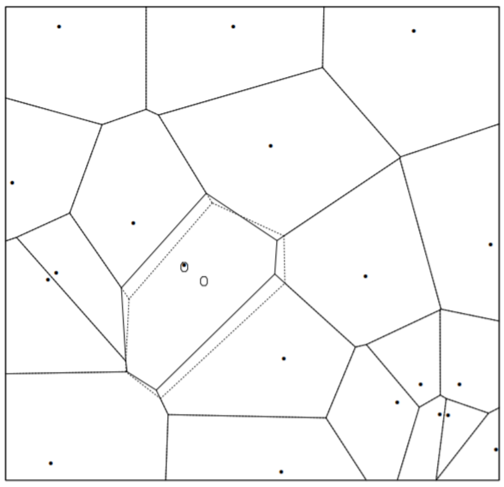


Figure 6.2: Spatial density estimated by Voronoi tiles with a specified number of centroids. Source: Byers and Raftery ([8]).

$r_c = r$. Thus, for instance $n^{(r)}_{c,c+h,i \to j}$ refers to the number of transitions from $\lambda_k = i$ to $\lambda_{k+1} = j$ between times $k = c$ and $k = c + h$, where the shark was in region r at time k.

Additionally, we can define $n^{(r)}_{c,c+h}$ as the sum of elements in $N^{(r)}_{c,c+h}$, meaning the number of times in this interval that she shark begin movement in region r. Note we must have $\sum_{r=1}^{R} n^{(r)}_{c,c+h} = h$ since h transitions have occurred between c and $c + h$ across all R regions, since each time involves a transition of either staying in the same behavior or changing.

[2]A Voronoi partition works by selecting R points (centroids) in a space and defining a region for any location in the area by assigning it to the centroid it is closest to; the regional boundaries are thus defined automatically. Centroids can be selected independently at random, in which case regions are likely to vary in size, or evenly spaced, in which case the regions will be of about equal size, depending on the placement of the first centroid.
A more complicated option (not discussed here) is to iteratively learn the regional definition for best separation of regional foraging. Examples of this are Byers and Raftery ([8], see Figure 6.2) or Heikkinen and Arjas ([20]), which estimate the spatial intensity (e.g., Poisson) of a set of points by a discrete nonparametric approximation or Voronoi tesselation. They define a partition of the space into tiles, each with a single intensity value; the algorithm iteratively combines and splits tiles of an initial partition to achieve a partition that optimally approximates the continuous intensity function. This method is similar to iterative clustering techniques where clusters are combined and divided to determine the optimal cluster number. Although it seems difficult to incorporate this method into the particle filter itself, it could be a good way to pre-define the regions. For instance, rather

If $r_c = r$, $\lambda_c = i$, and $\lambda_{c+1} = j$, then $n^{(r)}_{c,c+1,i\to j} = 1$, all others remain the same. The Dirichlet posteriors are updated recursively by transition counts, so $\alpha^{(r)}_{c+1,i\to j} = \alpha^{(r)}_{c,i\to j} + 1$, and the pairwise probabilities for $p^{(r)}_{c+1,i\to j}$ and $p^{(r)}_{c+1,i\to i}$ are re-drawn. Note that this Bayesian model for $P^{(r)}_c$ contains the assumption that transition probabilities are independent of the amount of time spent in the current state λ_c. A relaxation of this assumption will be discussed.

In the CDLM, at time $c-1$, region $r_{c-1} = r$ and behavior λ_{c-1} are known, and we want to simulate the next behavior λ_c probabilistically. The CDLM is modified so λ_c is determined by considering the most likely behavior given λ_{c-1} and region r, so $p^{(r)}_{c-1,(\lambda_{c-1})\to k} = \Pr(\lambda_c = k \mid \lambda_{c-1} \,\&\, r_{c-1} = r)$. The fact that these probabilities can differ between regions allows us to learn the most likely next future behavior, given the current region r of the shark.

Ultimately, we want to measure the posterior probability a shark will forage in region r, in addition to just the transition probabilities; denote this probability as $p^{(r)}_{c,0}$ (since $\lambda_c = 0$ denotes foraging). A simple way to do this is to use use theory from Markovian stochastic processes for a system with probability transition matrix P; this estimates the limiting probabilities of ending up in each of the states assuming the system runs indefinitely, given initial probabilities for each state. Given a regional transition probability matrix $P^{(r)}_c$, we can find the left eigenvector $\boldsymbol{\pi}^{(r)}$, the stationary probability of the first state (foraging) $p^{(r)}_{c,0}$ is estimated by normalizing $\boldsymbol{\pi}^{(r)}$ to have sum 1, and then taking the first element.

As discussed in Section 4.8, posterior inference on parameters can be done by density estimation on pooled samples of draws from the posterior distributions of parameters across particles n. Foraging probabilities can be estimated by drawing, say $K = 100$ regional transition matrices $P^{(r)}_c$ from each particle's posterior Dirichlet parameters, calculating the foraging probabilty $p^{(r,n)}_{c,0}$ for each from the eigenvector, and performing density estimation on the pooled

than partitioning the space into equal-sized regions, one may want to define more regions in the areas where sharks tend to spend more time (at least according to the observed data). These would then be the regional inputs into the particle filter definition.

$\{\{p_{c,0}^{(r,n,k)}\}_{k=1}^{K}\}_{n=1}^{N}$.

6.3 Time-dependent (non-Markovian) transition probabilities

Another way to model the transitions is to assume that the longer a shark has been in the current behavior, the more likely it is switch behaviors; this violates the Markovian assumption that the transition probabilities are time-independent. An example of this is in Beyer, et al. ([6]) where the staying probabilities are $p_{c,i\to i} = e^{-\kappa t}$, where κ is a constant and t is the time since the last state transition, so $p_{c,i\to i}$ is decreasing in t. Their formulation of the switching probabilities is not Bayesian, however, since the parameter κ does not appear to be learned from the data. We propose an similar formulation which uses the Beta distribution to model the probability $p_{i\to j}, i \neq j$ of changing behaviors. However, in our synthetic and actual data simulations, we will use the Markovian model for simplicity.

We use the reformulation of the likelihood for choosing next behavior in Equation 6.1. Here, the shark's behavior is $\lambda_{c-1} = i$ and the behavior $\lambda_c = j$ is being drawn. The likelihood is

$$\Pr(\lambda_{c+1} \mid \lambda_c = i) = (p_{c,i\to i})^{x_{c+1,i}} (p_{c,i\to j})^{x_{c+1,j}}$$

where $x_{c+1,j} = \mathrm{I}(\lambda_{c+1} = j), j \in \{0, 1\}$. The likelihood is thus rewritten in terms of the probability of changing to a different behavor $j \neq i$, and thus $x_{c+1,j} = 1$ if the behavior changes. Let $t = \underset{h}{\operatorname{argmax}}\{h \geq 1\colon \lambda_{c-h-1} = \cdots = \lambda_c = i\}$. It represents the number of consecutive regular time steps at c the shark has had the behavior λ_c; $t = 1$ if the shark had just changed into its current behavior, that is, if $\lambda_{c-1} \neq \lambda_c$.

We can thus formulate a bivariate Dirichlet prior

$$(p_{c,i\to i},\ p_{c,i\to j}) \sim \mathcal{Dir}_2(\alpha_{i\to i}/t^b,\ \alpha_{i\to j}t^b)$$

where $b > 0$ is a constant power. The α parameters are updated based on which transition occurs, as follows:

- If the animal stays in its current behavior ($\lambda_{c+1} = \lambda_c = i$), then the $t^{'} = t+1$, since the animal has increased its time in the current behavior by 1 step. Thus, $x_{c+1,i} = 1$ and $x_{c+1,j} = 0$.
- If the animal changes behavior ($\lambda_{c+1} \neq \lambda_c$), then $t^{'} = 1$ since the animal begins a new sequence of steps in the new behavior $j \neq i$. Thus, $x_{c+1,i} = 0$ and $x_{c+1,j} = 1$.

The posterior updates of the α parameters are thus

$$\begin{cases} \alpha^{'}_{i\to i} = (t+1)^b(\alpha_{i\to i}/t^b + 1) & \text{if } \lambda_{c+1} = \lambda_c = i \\ \alpha^{'}_{i\to j} = \alpha_{i\to j}t^b + 1 & \text{if } \lambda_{c+1} = j,\ \lambda_c = i,\ i \neq j \end{cases}$$

We recommend setting b to be relatively small, such as $1/4$ or so, because a higher power b would cause imbalance between the Dirichlet parameter updates. For instance, the α parameter for changing behavior, $\alpha_{i\to j}$, would grow too fast if it were multiplied by t^b, where b is too high; setting it at a root less than 1 keeps t^b closer to 1. Increasing one of the Dirichlet α parameters would cause the posterior to become imbalanced and over-predict one of the choices, either leaving or staying, based on past observations. We note also that the choice of initial α parameter values in the time-dependent setup would differ from the choice in the Markovian setup, where the αs should be proportional to the overall transition probabilities. Additionally, they may be affected by the choice of b.

6.4 Inter-shark behavioral influence

One question of biological interest is whether sharks are influenced in their behavior by nearby sharks, over and above the region-specific foraging/transiting effects that we are modeling as well. Unfortunately, due to the data collection methods, the 22 sharks' observation spans are separated across two years (see Figure 2.1), with 9 and 13 in each of the two years, with no more than five or six sharks having overlapping observation spans at any given point. Thus, we can only try to model inter-shark behavioral influence of at most five or six sharks at a time, and it is difficult to assume that such few sharks represent some population-level dynamics of the sharks in the tidal basin. Nevertheless, we will try to use the few sharks we have. The authors' surveys of the tidal basin's species ([17]) indicate 422 unique gray smooth-hound sharks were caught over the study's duration, so there is a population of at least several hundred unique sharks that use the tidal basin. However, they estimate that at any given time, the tidal basin may have 30–100 or so sharks in it.

We will try to measure the association between a shark's behavior and the behavior of other sharks simulated in a circular spatial neighborhood $N(\,\cdot\,)$ of its location and over a specified temporal window, thus forming a spatial-temporal neighborhood. This association is over and above the region-specific effect. Like all other parameters in the model, these association parameters will be learned by iterative resampling and updating. The animal CRW telemetry literature has models to incorporate congregation or attraction effects[3], such as the tendency of some animals to group together when moving, but we are not aware of this particular application featuring prominently in the literature.

This approach is inspired by target-tracking literature for tracking multiple moving objects (targets) that may interact with each other. The target tracking literature in general deals with the problem of modeling the locations of moving targets but also with the problem of maintaining identification throughout observation. Tracking algorithms may also have to use computer vision techniques to read the targets' positions from a video or sensor (for instance, tracking members of a basketball team as they move around the court in real time). Other considerations in target tracking involve detecting when

[3]See, for example, Turchin ([53]) p. 109–126, p. 161–175.

a target has left the area under consideration. In our case, these aspects are irrelevant since we are modeling the sharks' movements retroactively—not in real time—and we will not assume that, for instance, we may have failed to detect a shark. Although the sharks can leave the basin, we will assume they do not, since we do not have observations outside the basin, out of range of the receivers.

For instance, Khan, Balch, and Dellaert ([27]) describe an experiment where 20 ants were tracked in a small rectangular arena. Their algorithm jointly tracks the moving ants and has to maintain identification; for instance, when a model for ant 1 is learned, it should continue to identify which ant is which, even though ants may crowd together. In the first setting, the arena was closed, and in another it was open so the ants could leave; this adds the complication of identifying when an ant leaves and adjusting the number of targets estimated as a result.

When estimating the joint likelihood of the ants' predicted locations, the algorithm wants to penalize predictions of ant locations in very close proximity to other ants—i.e., within another ant's circular "region of influence"—since multiple ants rarely occupy the same location as one another simultaneously. They add a pairwise interaction function $\psi(\cdot,\cdot)$ (not to be confused with our ψ denoting the bearing angle) between the simultaneous locations $\boldsymbol{\zeta}_s$ and $\boldsymbol{\zeta}_{s'}$ of two ants s and s', as long as they are within a certain spatial neighborhood of each other. The function ψ captures the influence of two nearby ants on each other; in their application it is used to penalize predicting two ants in coinciding locations, but we will adapt this idea to model the behavioral influence between neighboring sharks.

Let the ordered pair (c, s) denote the constant interval-shark combination. Because we only simulate the sharks' unobserved movement at intervals c around which they have observations $\mathbf{y}_t$, for a given shark s, the pair (c, s) may not be defined for all intervals $c = 0, \ldots, C$; likewise, for a given interval c, not all sharks s are simulated then. Now, assuming the pair (c, s) is defined (the shark is simulated then),

- let $\boldsymbol{\zeta}_{c,s}$ denote the simulated coordinates of our particle filter for (c, s), and
- let $\lambda_{c,s} \in \{0, 1\}$ denote the shark's behavior at c

To define the spatial-temporal neighborhood $N(\,\cdot\,)$ under consideration, let $\delta_{\zeta} > 0$ be the diameter of the temporal neighborhood. Our neighborhood $N(\,\cdot\,)$ will consist of all previous observations (c', s') of other sharks $(s' \neq s)$ within a distance of δ_{ζ} of $\boldsymbol{\zeta}_{c,s}$—the shark's location at timestep c—and less than δ_{Υ} seconds before the time Υ_c. In the following definition, we note that all variables include an implicit superscript of $n = 1, \ldots, N$ for each particle, which we omit for clarity. Since each particle n is considered an independent simulation from the others, the neighborhoods of (c, s) are only calculated by considering pairs (c', s') from the same particle history n. The neighborhoods are defined as

$$\begin{aligned} N\big((c,s)\big) = \big\{(c', s') \colon & (s' \neq s) \;\& && \text{other sharks} \\ & ((\Upsilon_c - \delta_{\Upsilon}) \leq \Upsilon_{c'} < \Upsilon_c) \;\& && \leq \delta_{\Upsilon} \text{ sec. before} \\ & (\|\boldsymbol{\zeta}_{c,s} - \boldsymbol{\zeta}_{c',s'}\| < \delta_{\zeta})\big\} && < \delta_{\zeta} \text{ meters away} \end{aligned}$$

where $\|\cdot\|$ indicates Euclidean distance. One may also want to only consider neighborhood where the number of neighbors is above a threshold, say 10 observations. This would prevent small neighborhoods of only a few observations having an outsize effect in the estimation of the spatial attraction parameters. The spatial radius δ_{ζ} of the neighborhood should not be too big—or else other sharks far away that should not influence the shark would be included–or too small—in which case few neighborhoods above the minimum size would be detected. Because the coordinates $\boldsymbol{\zeta}_{c',s'}$ in the particle filter are propagated by random draws from the prediction density (in our case multivariate normal), there is some variance when propagating the coordinates. Hence, the spatial radius should be big enough to account for this variance but not too small that it only picks up noise due to the randomness of propagation.

After having specified the determination of neighborhoods, we will now outline our proposed Bayesian framework of modeling the degree of behavioral influence. Our formulation is based on the intuition that if a greater proportion of a shark's neighboring sharks are foraging, that shark may be more likely to forage as well, which is why we also set a minimum number of neighbors to consider. Let $\pi_{c,s,k}$, $0 \leq \pi_{c,s,k} \leq 1$ represent the proportion of simulated observations (c', s') in the neighborhood $N\big((c,s)\big)$ having behavior $\lambda_{c',s'} = k$, namely

$$\pi_{c,s,k} = \frac{\displaystyle\sum_{(c',s') \in N((c,s))} \mathrm{I}(\lambda_{c',s'} = k)}{|N\big((c,s)\big)|}$$

where $|N(\cdot)|$ here denotes the number of neighbors. By definition, $\sum_k \pi_{c,s,k} = 1$. The proportion $\pi_{c,s,k}$ denotes the relative influence we may expect the neighbors to have on the likelihood that the shark s will have behavior k ($\lambda_{c,s} = k$) at time Υ_c, if the spatial-temporal clustering hypothesis assumption is valid.

Figure 6.3 shows a toy illustration of spatial-temporal neighborhoods. Here, observation (c, s) under consideration, is shown as a red "X". Fifty other sharks' ($s' \neq s$) previous ($c' < c$) observations, $\boldsymbol{\zeta}_{c',s'}$ are plotted as round points. Of these, some (solid) are foraging ($\lambda_{c',s'} = 0$) and the others (hollow) are transiting ($\lambda_{c',s'} = 1$). Assuming that all points plotted are within the temporal neighborhood δ_Υ, the spatial-temporal neighborhood $N\big((c,s)\big)$ with radius $\delta_{\boldsymbol{\zeta}} = 1.5$ is drawn as a circle around the red "X", $\boldsymbol{\zeta}_{c,s}$. The neighborhood contains $|N\big((c,s)\big)| = 14$ neighboring observations, of which 11 are solid (foraging). As we note later, we are only concerned with the proportion of neighbors that are foraging, namely $\pi_{c,s,0} = 11/14 \approx 0.79$, in this case. The higher this proportion, the more likely it seems that (c, s) will represent a foraging observation as well. Specifically, we believe that the foraging proportion for neighbors may be higher than the overall foraging proportion in the region, which in this toy example is $p_0^{(r)} = 23/50 = 0.46$.

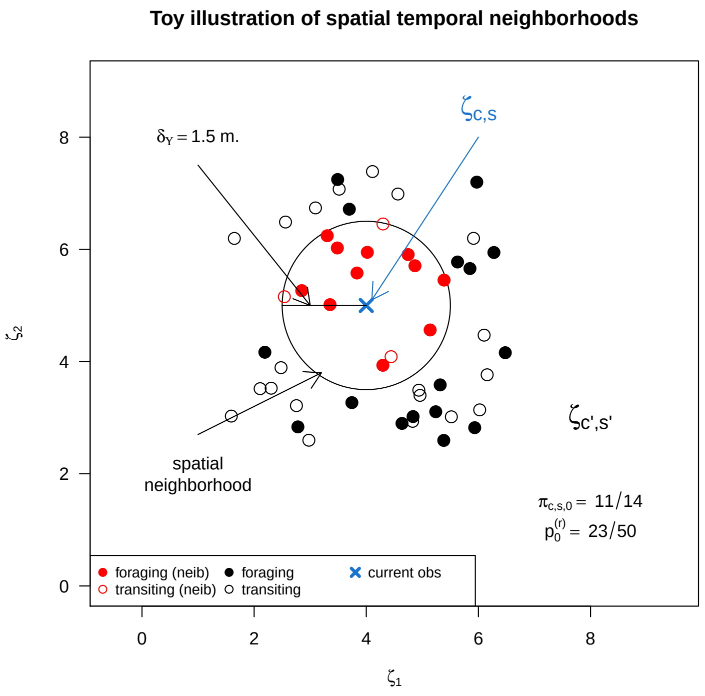


Figure 6.3: Toy illustration of spatial-temporal neighbors. The red "X" denotes the location $\boldsymbol{\zeta}_{c,s}$ of the observation (c, s) being considered. Fifty other sharks' observations $(\boldsymbol{\zeta}_{c',s'})$, the circular points, are plotted as well. We expect the proportion of neighbors foraging ($\pi_{c,s,0} = 11/14$ to be larger than the foraging proportion of all sharks, in this case the regional foraging probability $p_0^{(r)}$ which is empirically 23/50 in this case.

We will demonstrate whether the data seem to support the potential of this kind of neighborhood influence. Figure 6.4 uses the regular interval spline-interpolated trajectories to illustrate correlation between neighboring sharks's behaviors and the shark's current behavior. For each foraging observation ($\lambda_{c,s} = 0$) the spatial-temporal neighborhoods $N\big((c,s)\big)$ were determined; the neighborhoods, with a minimum size of 5 neighbors, included all other sharks' regular observations occuring less than $\delta_{\Upsilon} = 3,600$ seconds (1 hour) before time Υ_c. To show the effect of increasing the spatial neighborhood radius, the neighborhoods $N((c,s))$ were determined at spatial radii of $\delta_{\boldsymbol{\zeta}} = 25, 50, 75, 100, 150, 200, 300, 400$ meters, and the proportion of foraging observations $\pi_{c,s,k=0}$ within the neighborhood was calculated. Since we believe there may be a positive association between the proportion of neighboring sharks' observations that are foraging and whether the shark at (c,s) was foraging, we expect to see the proportion of foraging observations fall at large spatial radii, where the spatial influence is weak or nonexistent. At very small radii, the relationship will probably be indeterminate.

Figure 6.4 plots the distributions of the foraging proportions $\pi_{c,s,k=0}$ for each shark s, conditional on the shark foraging ($\lambda_{c,s} = 0$), across the range of spatial radii $\delta_{\boldsymbol{\zeta}}$. For each of the 22 sharks, the number of spline observations is listed as well. We see that for the several sharks (17, 20, 21, 22) that have few spline observations, the boxplot pattern looks haphazard. Keeping in mind that the spline model is only an approximation of the underlying speeds and behavioral modes of the shark, we generally see the pattern that at small spatial radii such as 25 or 50 meters, including other sharks that are very close to the given one, when the shark is foraging the proportion of neighbors that are foraging is high. The proportions tend to fall when the spatial radius increases. Note also that in our spline approximation, about 65% of regular steps overall are designated foraging, so these proportions $\pi_{c,s,k=0}$ do not fall very low overall. This makes intuitive sense biologically, because fast movement (i.e., transiting) expends significant energy, we expect sharks to spend the majority of their time foraging or moving slowly, and only transiting

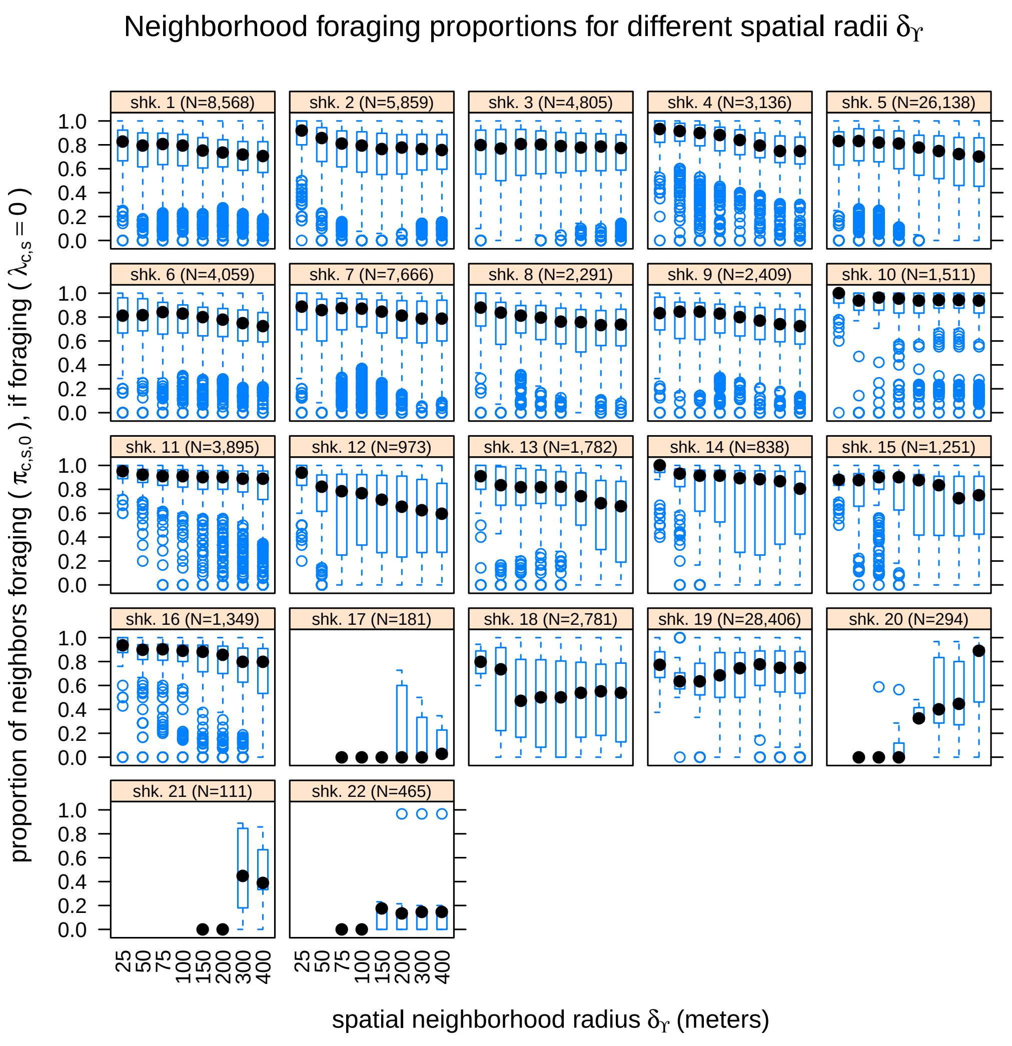


Figure 6.4: Boxplots of $\pi_{c,s,k=0}$ (proportions of spatial-temporal neighbors that are foraging) across various spatial radii, conditional on the shark foraging ($\lambda_{c,s} = 0$), for the spline-interpolated trajectories. The proportions of neighbors foraging tends to decline as the spatial radius δ_ζ increases, lending support to the potential relationship between a shark's neighbors' behaviors and its own.

occasionally and for short periods of time as they move from region to region. This is true of most animals, including humans, not only marine ones. For instance, cheetahs obviously conserve their energy for very fast movement when it is needed, such as hunting prey or fleeing.

We now outline the details of the influence parameter model. In the following, the particle (n) and shark indices (s) are dropped for clarity. In the setup, a parameter η_k, $k \in \{0, 1\}$ captures the strength of the spatial influence of neighbors in $N\big((c, s)\big)$ with behavior $\lambda_{c',s'} = k$ on whether the shark s will have that behavior, that is, on the Bernoulli probability $\Pr(\lambda_{c,s} = k)$. The parameter η_k is learned and modeled through the particle filter process.

For this application, we will parametrize the normal distribution in terms of its precision (τ), the inverse of the variance σ^2, because the form of the posterior is more attractive. Thus $\sigma = 1/\sqrt{\tau}$. The normal density can be thus written as

$$f(x \mid \mu,\, \tau) = \sqrt{\frac{\tau}{2\pi}}\exp\left(-\frac{\tau(x-\mu)^2}{2}\right)$$

The log-normal distribution is also parametrized in terms of its precision, which we denote ϵ:

$$\mathcal{Lognormal}(x \mid \mu,\, \epsilon) = \frac{\sqrt{\epsilon}}{x\sqrt{2\pi}}\exp\left(-\frac{\epsilon(\ln(x)-\mu)^2}{2}\right)$$

Using this parametrization, for the spatial-temporal influence, we set a normal prior on η_k

$$\eta_k \sim \mathcal{N}(\omega_k,\, \tau_k)$$

The interaction multiplier effect (denoted ρ_k) of the neighbors depends both on the strength of this influence (η_k) and the proportion of the neighbors with the behavior ($\pi_{c,s,k}$). The likelihood will be modeled with a log-normal distribution as

$$\rho_{c,s,k} \sim \mathcal{Lognormal}(\pi_{c,s,k}\eta_k,\, \epsilon_k)$$

where ϵ_k is the precision of the log-normal distribution, which is assumed to be known. Since $\rho_k > 0$ since it is log-normally distributed, ρ_k will act as a scaling factor in the CDLM to adjust the conditional likelihood of each particle n's transition probability $\Pr\left(\lambda_{c,s}^{(n)} = k \mid \lambda_{c-1,s}^{(n)},\ \boldsymbol{\theta}_{c-1,s}^{(n)}\right)$. In our case, with only two behaviors, we need only to model η_k and ρ_k for one (foraging, $k = 0$), and keep $\rho_{c,s,k=1}$ constant for transiting. Thus $\rho_{c,s,k=0}$ adjusts the probability of foraging vs transiting, where transiting acts as a baseline category, similar to how in linear regression the number of dummy variables for a categorical variable is one less than the number of levels.

For a log-normal distribution of a variable X with mean μ is e^{μ}, so if $\mu = 0$ the median is 1, and if $\mu > 0$, the median is above 1, so $\Pr(X > 1 \mid \mu > 0) > 0.5$. In our case, the median of the scaling factor $\rho_{c,s,k}$ is $\exp(\pi_{c,s,k}\eta_k)$. Thus $\eta_k = 0$ exactly means the proportion of neighbors with behavior $\lambda_{c',s'} = k$, i.e. $\pi_{c,s,k}$ does not influence that shark's probability $\Pr(\lambda_{c,s} = k)$. The more positive η_k is, which is more likely if its prior mean $\omega_k > 0$, the stronger is the positive association between the neighborhood proportion $\pi_{c,s,k}$ and $\Pr(\lambda_{c,s} = k)$. A value $\eta_k > 0$ indicates behavior k has spatial-temporal attractiveness, or clustering; the reverse, $\eta_k < 0$, doesn't tend to correspond to a stable model since like behaviors will 'repel' each other, and we wouldn't expect it in a realistic biological model. Nevertheless, it seems appropriate to set the prior means $\omega_k = 0$ (no effect) for η_k so we do not bias our model to learn influence if it is nonexistent. The posterior of η_k, which is only updated if a neighborhood $N\big((c,s)^{(n)}\big)$ is defined, is

$$\eta_k \mid (\rho_{c,s,k},\ \pi_{c,s,k}) \sim \mathcal{N}\left(\frac{\epsilon_k \pi_{c,s,k} \ln(\rho_{c,s,k}) + \tau_k \omega_k}{\epsilon_k \pi_{c,s,k}^2 + \tau_k}\ ,\quad \epsilon_k \pi_{c,s,k}^2 + \tau_k\right)$$

where again the precision is used to parametrize the normal distribution. The interaction effect η_k can be tested for significance by seeing if the posterior interval of each η_k is strictly positive, and by density estimates or posterior intervals of the interactions ρ_k at various values of the neighborhood proportions $0 \leq \pi_{c,s,k} \leq 1$.

More generally, if the posterior of η_k at step Υ_c is to be updated for the past h steps, rather than for a single time step Υ_c as above, we have the initial value drawn from the prior

$$\eta_k \sim \mathcal{N}(\omega_k,\ \tau_k)$$

Say for steps $c-h, c-h+1, \ldots, c$, let $A \subseteq \{c-h, c-h+1, \ldots, c\}$ be the set of steps for which a neighborhood $N(\,\cdot\,)$ is defined (i.e. there were enough other sharks within a given spatial-temporal radius and above the minimum number required). The interaction effect is only updated for steps where neighborhoods are defined. The posterior is thus

$$\eta_k \mid \{(\rho_{i,s,k},\ \pi_{i,s,k})\}_{i\in A} \sim \mathcal{N}\left(\frac{\epsilon_k \left(\sum_{i\in A} \pi_{i,s,k} \ln\left(\rho_{i,s,k}\right) \right) + \tau_k \omega_k}{\epsilon_k \left(\sum_{i\in A} \pi_{i,s,k}^2 \right) + \tau_k},\quad \epsilon_k \left(\sum_{i\in A} \pi_{i,s,k}^2 \right) + \tau_k \right)$$

To demonstrate that the interaction parameters η_k are learning a meaningful interaction between the sharks, we can run the model on simulated irregular data interpolated from regularly-observed data (see Section 7.1) and try to model the unobserved underlying regular observations. It is reasonable to assume that the distributions of time gaps between irregular observations for the actual shark data are independent across sharks. We can generate similar shark data under this assumption by independently generating each shark's regular-step trajectory and interpolating the trajectory to a different draw of time gaps from the log-normal (to imitate the observed data), then combining the datasets. Whenever sharks' synthetic trajectories are generated independently, there should be no meaningful interaction, and our posterior intervals of η_k should include 0.

We can also select several synthetic sharks to have interaction potentials with certain values of η_k to see if our algorithm captures this effect and learns a posterior interval of η_k including the specified intensity. In the synthetic generation, at each step c, we determine the spatial-temporal neighborhood

$N\big((c,s)\big)$ for shark s and determine the neighbor proportions $\pi_{c,s,k}$, which then determines the interaction weight $\rho_{c,s,k}$ to adjust the shark's behavior transition probabilities and simulate their behavior $\lambda_{c,s}$. The key requirement for success here, of course, is that our algorithm can relatively successfully identify the true behaviors in the underlying irregularly-observed data.

CHAPTER 7

INTERPOLATION AND SMOOTHING

7.1 Interpolation CDLM particle filter

The primary feature of the raw shark data that makes modeling and addressing our biological questions difficult is the fact that the observations are irregular in time. For instance, based on the shark's speed at a given time, we can tentatively classify the behavior λ_t (based on the speed, $\lambda_t = 0$ if $v_t < 0.1$) in the raw dataset and tabulate these values to see if the sharks are more likely to transit or forage. However, since observations are at irregular intervals, as shown in Figure 7.1, because the speed v_t in the raw data is average speed between observations, assuming the distance traveled is the Euclidean distance between them. Thus, the average speed v_t actually represents the combination of perhaps a large number of (unobserved) intermediate steps of equal size.

Furthermore, when the time gap Δ_t is larger, there is more uncertainty as to the shark's actual behavior between observations. At a larger step size, the Euclidean distance becomes less applicable because the shark can turn and change directions; thus, if a lot of time has passed between two locations that are near to each other, it will appear if the shark is traveling slowly because the true distance can be much larger than the 'connect-the-dots' Euclidean distance. Thus, a direct tabulation on behaviors from the raw data as if they represented equally weighted observations would not be accurate. Furthermore, we would like to assess not only the relative frequency of transiting vs foraging overall but also the frequency within different regions. From a modeling perspective, the irregular observations are also problematic for the PF because the Bayesian updates of parameter distributions treat each new observation as equally weighted, though in fact they are not, because they represent different underlying spans of time.

Jonsen, Flemming, and Myers ([22]) take the approach of interpolating their irregularly-recorded observations $\mathbf{y}_t$ to regularly spaced unobserved $\mathbf{x}_t$, then model these. Our approach is guided by this, as well as data assimilation techniques. Assume regularly-spaced timesteps $\Upsilon_0, \Upsilon_1, \ldots$ and observations $\mathbf{y}_t$ that occur at these times but not necessarily at each–that is, there are missing observations but no interpolation. Reich and Cotter ([41], chapter 6) show that the particle filter operates as usual except at any time Υ_c where there are no observations, states $\mathbf{x}_k$ are simply propagated forward, keeping the resampling weights $w_c^{(n)}$ constant. When an observation occurs, the weights are updated and the algorithm may choose to resample or not (for instance, based on the effective size criterion). The assumption that observations occur only at the regular timesteps is used for notational simplicity (authors' communication) but can be extended to the case where they occur in between the regular times $\boldsymbol{\Upsilon}$.

For our interpolation, we specify a desired regular time gap Δ_Υ (say, 120 seconds), and hence the regular timesteps $\boldsymbol{\Upsilon}$ at which we simulate values of the process. At each regular timestep, if a shark is observed, then the algorithm uses that observation; otherwise, the algorithm simulates the behavior in that step using the existing information. Section 6.4 introduced the spatial-temporal neighborhood ($N(\cdot)$) structure for modeling inter-shark behavioral influence by the parameter η_0, at intervals c. Because this joint shark filter involves both interpolation to evenly-spaced time intervals $\boldsymbol{\Upsilon}_c$, we must introduce notation to deal with multiple sharks. The notation is as follows:

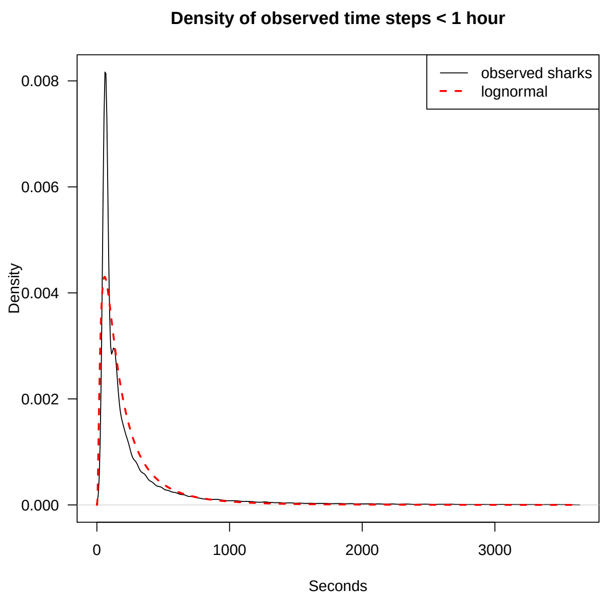


Figure 7.1: Empirical density of the observed distribution of time gaps Δ_t (black) in seconds (restricted to be less than 3,600 seconds, which is above the empirical Q_{98}) between observations for all sharks. This distribution seems reasonably approximated by a log-normal distribution, shown in black. The raw Δ_t have $Q_1 = 68$ seconds, median is 130, and $Q_3 = 299$ (5 minutes); the mean is 826 seconds (about 13.75 minutes), but is significantly affected by extreme values. The variability of the time gaps necessitates accounting for this in modeling the movement parameters.

Note that at small time gaps, the empirical density is nearly twice that of the log-normal approxiation. This is a bit of an artefact, since the minimum observed Δ_t is 43 seconds, rather than a hypothetical value near zero for the log-normal. The transmission code delays (a technical specification known as the pinging rate) of the shark VPS transmitters were between 40 and 80 seconds, meaning that a shark will not be observed at time gaps of less than 40 seconds.

- As before, we have T total observations $\mathbf{y}_t$, $t = 1, \ldots, T$, observed at clock times $H_1 \leq H_2 \leq \cdots \leq H_T$. These observations are pooled across sharks, so observations will alternate between sharks.

- Each observation corresponds to one shark $s = 1, \ldots, S$. The times H_t are treated as continuous, and so we assume they are all unique (so the inequalities are generally strict), but we allow them to be equal, as long as each observation $\mathbf{y}_t$ is uniquely identified with a time H_t to preserve consistency of notation.

- The observations are interpolated to $C+1$ constant-interval times $\Upsilon_0 < \Upsilon_1 < \cdots < \Upsilon_C$ indexed $c = 0, 1, \ldots, C$, where $H_{T-1} < \Upsilon_C \leq H_T$ and the desired constant difference between them is a small Δ_Υ seconds. These are generated as $\Upsilon_0 = H_1 - \epsilon$, where ϵ is a small number so that the first observation $\mathbf{y}_1$ falls in an interval that is open on the left. Thereafter, let $\Upsilon_c = H_1 + c\Delta_\Upsilon$, $c = 1, \ldots, C$, where $C = \underset{c}{\text{argmax}}\{\Upsilon_c \leq H_T\}$.

- We may choose to only interpolate to intervals $\boldsymbol{\Upsilon}_c$ between two consecutive observations at H_t and H_{t+1} if they differ by fewer than c_{thresh} intervals. For instance, we may set $c_{\text{thresh}} = 10$, so we need $H_{t+1} \leq H_t + c_{\text{thresh}}\Delta_\Upsilon$, or $H_t \leq \Upsilon_c < \cdots < \Upsilon_{c+c_{\text{thresh}}-1} < H_{t+1} < \Upsilon_{c+c_{\text{thresh}}}$ to interpolate to times $\Upsilon_{c:(c+c_{\text{thresh}}-1)}$. If you want time-unrestricted interpolation, set $c_{\text{thresh}} = \infty$.

- Because we generally want to model more than one shark ($S > 1$), we need to uniquely associate each observation $\mathbf{y}_t$ and time H_t with a specific shark s. Let $s_t \in \{1, \ldots, S\}$, $t = 1, \ldots, T$ denote the shark s that observation $\mathbf{y}_t$ corresponds to. Figure 7.2 shows a toy illustration of this notation. Consider row labeled 3 for shark $s = 3$. The first observation in this row (at 3:00) is the second ($t = 2$) overall, so $s_{t=2} = 3$. Further, let the notation $h(s, j)$ mean the hour of observation of shark s's j^{th} observation overall. Therefore, we can also say $h(s = 3, j = 1) = H_2$

since this observation at $t = 2$ is shark 3 first (#1) observation overall, and its time observation is $H_2 = 2{:}00$.

- We need to identify observations within a given interval c of times $(\Upsilon_c, \Upsilon_{c+1}]$. Let the notation $t(s, c, j)$ refer to the index t corresponding to shark s's j^{th} observation occurring in time interval $(\Upsilon_c, \Upsilon_{c+1}]$, if such as observation exists. Note j only indexes observations within the interval, not overall.

- This notation $t(s, c, j)$ can be used to express sets. The expression $t(s, c, \cdot) = \{t\colon (\Upsilon_c < H_t \leq \Upsilon_{c+1})\ \&\ (s_t = s)\}$ is the set of indices t corresponding to shark s in interval c. For instance, $t(s = 3, c = 3, \cdot)$ means the indices t of shark 3's observation in the interval $c = 3$, between times $\Upsilon_3 = 8{:}00$ and $\Upsilon_4 = 10{:}00$; there are two such observations, corresponding to shark 3's third and fourth observations, labeled $h(3, 3)$ and $h(3, 4)$ by their clock times. Assuming the observations in the image are the only ones for the time interval 2:00 to 12:00 (assume sharks $s = 4, \ldots, 21$ are observed for the first time after 12:00), these observations correspond to the seventh and ninth overall, in chronological order. Therefore the set $t(3, 3, \cdot) = \{7, 9\}$; note they are not consecutive, because the eighth observation overall belongs to shark 1, at $h(1, 2) = 9{:}00$. Let $|\cdot|$ denote set cardinality; thus $|t(3, 3, \cdot)| = 2$ because there two such observations. To refer to each of their indices, we can say $t(3, 3, 1) = 7$ and $t(3, 3, 2) = 9$. We can also iterate over all of a shark's observations in an interval, for instance iterating over $j = 1, 2, \ldots, |t(s, c, \cdot)|$, where $t(s, c, j)$ refers to the time index of the j^{th} such observation.

At each regular timestep Υ_c, the joint particle filter iterates over each shark $s = 1, \ldots, S$ and does one of three things:

1. If shark s's first observation—or, the first after a long span without observations (i.e, the last one was more than c_{thresh} constant intervals away)—occurs in this regular interval (for instance $\Upsilon_c < h(s, 1) \leq \Upsilon_{c+1}$,

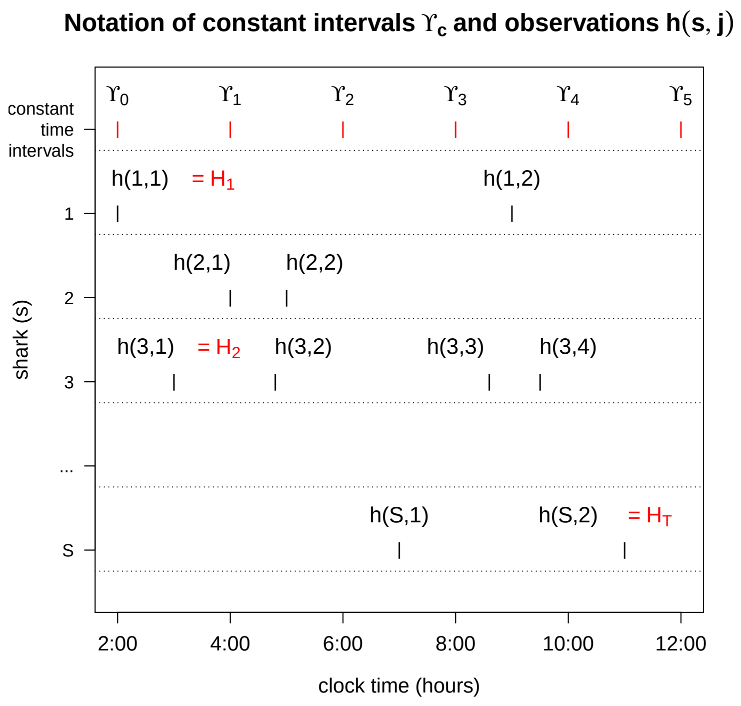


Figure 7.2: Toy illustration of observation times $H_1, \ldots, H_T$ and constant intervals $\Upsilon_0, \ldots, \Upsilon_C$ for a total of $S = 22$ sharks, covering the range of clock times 2:00 to 12:00. Here, the constant interval length is $\Delta_\Upsilon = \Upsilon_{c+1} - \Upsilon_c = 2$ hours. As shown, the observations $\mathbf{y}_t$ alternate between sharks. The first observation $\mathbf{y}_1$ at time $H_1 =$ 2:00 belongs to shark 1. Therefore $s_{t=1} = 1$ since it belongs to shark 1, and also this is the clock time of shark $s = 1$'s first (#1) observation, so $h(1,1) = H_1$. The next observation $t = 2$ is the first for shark 3, so $s_{t=2} = 3$ and $h(3,1) = H_2$.

where $h(s,1) = H_t$, if it is the first observation overall), the recorded observation $\mathbf{y}_t$ is taken as given, and its position $\boldsymbol{z}_{t+1} = \mathbf{y}_t$ and bearing angle ψ_t are used to interpolate the shark's position to the end of the regular period.

2. If shark s has no recorded observations (i.e., the set $t(s,c,\cdot) = \emptyset$) but has been observed before within c_{thresh} intervals ($\exists c'\colon\ (c - c_{\text{thresh}} \leq c' < c)\ \&\ (t(s,c',\cdot) \neq \emptyset)$), its behavior λ_c and movement are simulated using existing parameter values ($\boldsymbol{\theta}_{c-1}$, etc.). These movement variables are stored and may or may not be used to update the distributions. This generates a set of equally-spaced simulated observations $\{\mathbf{x}_c\}$ for the entire span of time we would ideally have recordings for the shark.

3. If shark s has recorded observations but not for the first time (within c_{thresh} intervals before), then we try to infer the unobserved behavior λ_c at that time by simulating movement for each potential behavior (CDLM) and seeing which one best predicts the observed location. Past simulated values of the movement are used to update the parameters at this point, if they were not done before.

Outline of interpolation CDLM:

- Let $H_1 <, \ldots, < H_T$ be the (irregularly) observed times for all sharks under consideration, with observations $\mathbf{y}_1, \ldots, \mathbf{y}_T$. Refer to Section 7.1 for details on notation.

- At constant-interval Υ_c, let $\mathbf{x}_c = \begin{bmatrix} \zeta_{1,c} & \zeta_{2,c} & \ln(v_c) & \psi_c \end{bmatrix}^T$, where $\boldsymbol{\zeta}_c = \begin{bmatrix} \zeta_{1,c} & \zeta_{2,c} \end{bmatrix}$ are the unobserved coordinates, and r_c be the region that $\boldsymbol{\zeta}_c$ are in.

- At each regular step c, the iteration is done separately for each shark s.

For regular step $c = 0, \dots, C-1$:

1. For shark $s = 1, \dots, S$:

   (a) Let $t(s, c, \cdot)$ be the ordered set of indices of the observed timesteps $H_1, \dots, H_T$ falling in the constant-length timestep interval $(\Upsilon_c, \Upsilon_{c+1}]$ at which shark s was observed. Let $|t(s, c, \cdot)|$ denote the number of shark s's observations in the interval, where it is $= 0$ if $t(s, c, \cdot) = \emptyset$, in which case there are no such observations.

   (b) If $t(s, c, \cdot) \neq \emptyset$ (**there are observations**):

      i. Let the set values be $t(s, c, j)$, using the third index, with corresponding observations $\mathbf{y}_{t(s,c,j)}$ at times of observation $H_{t(s,c,j)}$, $j = 1, \dots, |t(s, c, \cdot)|$.

      ii. Calculate the fractions of the regular interval at which each observation falls: let time differences be

      $$\Delta_{s,c,j} = \begin{cases} H_{t(s,c,1)} - \Upsilon_c & j = 1 \\ H_{t(s,c,j+1)} - H_{t(s,c,j)} & j = 2, \dots, |t(s, c, \cdot)| \end{cases}$$

      be the times in seconds between each observation in the interval.

      iii. **At the first time the shark is observed:** If $t(s, c, \cdot)$ is non-empty for the first time, meaning $t(s, c, 1)$ is the first observation of that shark overall or the first in a long gap of time (more than c_{thresh}), we will take these observations as fixed and model movement beginning at the next interval $c + 1$. Let

         A. $t^* = t(s, c, |t(s, c, \cdot)|)$ be the index of the last observation in the interval, and

         B. $\Delta_{t^*} = \Upsilon_{c+1} - H_{t^*}$ be the time from the observation to the end of the interval.

         C. Set $\lambda_c = \lambda_{t^*}$ and $\psi_c = \psi_{t^*}$, the observed behavior and bearing; simulate the speed $v_c \mid \lambda_c$.

D. Simulate the next regular location $\boldsymbol{\zeta}_{c+1}$ by going Δ_{t^*} seconds from the observed location $\mathbf{y}_{t^*}$ with the simulated speed and bearing. To simulate the location $\boldsymbol{\zeta}_c$ at the beginning of the interval, go backwards $\Delta_\Upsilon - \Delta_{t^*}$ seconds from the observation $\boldsymbol{z}_{t^*}$, reversing the bearing $\psi_c - \pi$.

iv. **If there are observations, but not for the first time** ($|t(s,c,\cdot)| > 0$): we want to generate predictions at each of the observed times $H_{t(s,c,j)}$, $j = 1, \ldots, |t(s,c,\cdot)|$ and see which state $\lambda_c \in \{0,1\}$ best matches all of the movement in the regular interval overall.

CDLM simulation: Iterate over $k \in \{0,1\}$ (potential values of λ_c):

A. Simulate movement variables, values of $(\ln(v_c), \theta_c)^{(n)} \mid (\lambda_c^{(n)} = k)$ for $\mathbf{x}_c$.

B. Estimate the **joint probability** that observations $\{\mathbf{y}_t\}_{t \in t(s,c,\cdot)}$ occurred given behavior $\lambda_c = k$ and the simulated movement $\mathbf{x}_{c-1}^{(n)}$. Define[1] the interpolation density $m^{(n)}_{\text{interp}_{t(s,c,j)|k}}(\cdot)$ of each observation $\mathbf{y}_t$, $t \in t(s,c,\cdot)$ for each particle n as

$$\begin{cases} d(\mathbf{y}_{t(s,c,1)} \mid \Delta_{s,c,1},\, \mathbf{x}_{c-1}^{(n)},\, \lambda_c = k), & j = 1 \\ d(\mathbf{y}_{t(s,c,j)} \mid \Delta_{s,c,j},\, \mathbf{x}_{c-1}^{(n)},\, \lambda_c = k,\, \hat{\mathbf{y}}_{t(s,c,j-1)}), & j = 2, \ldots, |t(s,c,\cdot)| \end{cases}$$

For multiple observations $j \geq 2$, the density conditions on the mean value of the previous predicted value $\hat{\mathbf{y}}_{t(s,c,j-1)}$.

The joint probability of the observations, given each potential behavior, is

$$m^{(n)}_{\text{interp}_{t(s,c,\cdot)|k}}(\mathbf{y}_{t(s,c,\cdot)}) = \prod_{j=1}^{|t(s,c,\cdot)|} m^{(n)}_{\text{interp}_{t(s,c,j)|k}}(\mathbf{y}_{t(s,c,j)})$$

That is, predict the first observed location $\hat{\mathbf{y}}_{t(s,c,1)}$ at $\Delta_{s,c,1}$

seconds from the previous simulated true location $\zeta_c^{(n)}$ following a straight line at angle $\psi_c^{(n)}$ and speed $v_c^{(n)}$. If $|t(s,c,\cdot)| > 1$ (there are more observations for the same shark), continue along this line to predict $\hat{\mathbf{y}}_{t(s,c,2)}$ at a further $\Delta_{s,c,2}$ seconds from $\hat{\mathbf{y}}_{t(s,c,1)}$, etc.

These intermediate $\hat{\mathbf{y}}_{t(s,c,j)}$ are only used for this probability calculation and are not kept as part of the simulated trajectory of timesteps $\{\Upsilon_c\}$.

C. The particle resampling weights $w_{c|k}^{(n)}$ for each behavior $\lambda_c = k$ are the above density (evaluated at the observed values), scaled by the transition probabilities:

$$w_{c|k}^{(n)} = m_{\text{interp}_{t(s,c,\cdot)|k}}^{(n)}(\mathbf{y}_{t(s,c,\cdot)} \mid \ldots, \lambda_c = k) \times \Pr\left(\lambda_c = k \mid \lambda_{c-1}^{(n)}\right)$$

v. End iteration over behavior type k.

vi. For particles $n = 1, \ldots, N$, set the overall weights as $w_c^{(n)} = \sum_{k=0,1} w_{c|k}^{(n)}$.

vii. Resample particles $\{\mathbf{x}_{c-1}^{(n)}\}_{n=1}^N$ by $\{w_c^{(n)}\}_{n=1}^N$. As before, let $\{\tilde{\mathbf{x}}_{c-1}^{(n)}\}_{n=1}^N$, $\{\tilde{w}_c^{(n)}\}_{n=1}^N$, and $\{\tilde{w}_{c|k}^{(n)}\}_{n=1}^N$ represent the resampled set of the N particles at $c-1$ and the weights.

viii. For particles $n = 1, \ldots, N$, propagate behaviors (assuming binary) $\lambda_c^{(n)} \sim \mathcal{B}er\left(p = \tilde{w}_{c|1}^{(n)} / \tilde{w}_c^{(n)}\right)$.

ix. For the propagated behavior $\lambda_c^{(n)}$, propagate $\mathbf{x}_c^{(n)} \mid \left(\tilde{\mathbf{x}}_c^{(n)}, \lambda_c^{(n)}\right)$.

x. Let $c^* = \underset{c'<c}{\text{argmax}}(c' \colon t(s,c',\cdot) \neq \emptyset)$ be the most recent regular step that an observation was made for the shark. Update the sufficient statistics and other parameters, given the values of $\mathbf{x}^{(n)}$ simulated over steps $c^*+1, \ldots, c$.

(c) **If there are no observations in this interval** c ($t(s,c,\cdot) = \emptyset$ but shark s previously had observations within c_{thresh} intervals): we simulate $\mathbf{x}_c \mid \mathbf{x}_{c-1}$ using existing parameter values.[2]Iterate over particles:

For $n = 1, \ldots, N$:

i. Let $r_{c-1}^{(n)}$ be the region of the previous coordinates $\boldsymbol{\zeta}_{c-1}^{(n)}$. Draw the behavior $\lambda_c^{(n)} \sim \Pr\left(\lambda_c^{(n)} \mid \lambda_{c-1}^{(n)},\, r_{c-1}^{(n)}\right)$.

ii. Simulate movement variables, values of $(\ln(v_c), \theta_c)^{(n)} \mid (\lambda_c^{(n)})$. Do not resample particles (since there are no observations to calculate weights). However, one may choose to update the distributions of parameters at each step or not.

(d) End iteration over particles n.

2. End iteration over shark s.

End iteration over constant time step c.

Figure 7.3 illustrates the interpolation density $m_{\text{interp}_{t:(t+1)|k}}^{(n)}$ for an interval $(\Upsilon_c, \Upsilon_{c+1}]$ with two observations $(\mathbf{y}_t, \mathbf{y}_{t+1}) = (\mathbf{z}_{t+1}, \mathbf{z}_{t+1})$, observed at times (H_t, H_{t+1}). The corresponding figure without interpolation is shown in Figure 5.3. Between successive simulated locations $\{\boldsymbol{\zeta}_c\}$ of short time interval Δ_Υ, the shark is modeled to travel in a straight line. Since the intervals are

[2] If there are no observations in a regular interval c, the algorithm should simulate movement given the current parameters, and update parameters based on the simulated movement, either at each step or when the next observation is encountered. For the sake of reducing the simulation or invention of data when the time gap Δ_t between successive observations $\mathbf{y}_t$ and $\mathbf{y}_{t+1}$ is big, we may, for instance, only simulate movement at regular intervals between $\mathbf{y}_t$ and $\mathbf{y}_{t+1}$ if Δ_t is below some threshold, say, less than 10 or 15 constant-length intervals of length Δ_Υ.

[2] The formulas for the interpolation densities $m_{\text{interp}_{t|k}}^{(n)}\left(\mathbf{y}_t \mid \mathbf{x}_{c-1}^{(n)}, \lambda_c^{(n)} = k\right)$ are as follows (see 4.5):

- **State equation** (does not depend on behavior $\lambda_c^{(n)} = k$):

$$\mathbf{x}_c^{(n)} \mid (\mathbf{x}_{c-1}^{(n)}) \sim \mathcal{N}_4\left(\ell\left(\mathbf{x}_{c-1}^{(n)}\right), \mathbf{P}_c^{(n)}\right), \quad \mathbf{P}_c^{(n)} = \mathbf{L}\left(\mathbf{x}_{c-1}^{(n)}\right)\Sigma_{c-1}^{(n)}\mathbf{L}\left(\mathbf{x}_{c-1}^{(n)}\right)^T + \mathbf{q}_c^{(n)}$$

where $\Sigma_{c-1}^{(n)}$ is the covariance of $\mathbf{x}_{c-1}^{(n)}$ given all previous particles and observations, and $\mathbf{q}_c^{(n)}$ is the error covariance of the state equation $\ell(\cdot)$ for Δ_Υ seconds.

- **Measurement equation:**

relatively short, this should be a decent approximation of the true movement. The straight lines between locations are either dashed for foraging ($\lambda_c = 0$) or solid for transiting ($\lambda_c = 1$).

At time Υ_{c-1}, movement from the foraging behavior ($\lambda_{c-1} = 0$) has been simulated from $\boldsymbol{\zeta}_{c-1}$ resulting in a prediction of $\hat{\boldsymbol{\zeta}}_c$. From $\boldsymbol{\zeta}_{c-1}$, the algorithm marginalizes over movement at Υ_c to best predict the two observations $(\mathbf{z}_{t+1}, \mathbf{z}_{t+1})$ shown by black solid dots. Given each potential behavior λ_c, the algorithm predicts the location at the times of observation H_t and H_{t+1}, that is, $(\hat{\mathbf{y}}_t,\ \hat{\mathbf{y}}_{t+1}) \mid (\lambda_c = k)$. These predicted locations are shown as hollow circles along each potential straight line path. Since the shark is assumed to travel at a constant speed $v_c \mid \lambda_c$ in the interval, the predictions $\hat{\mathbf{y}}$ of the observed locations are located at the same proportions of distance along the lines as the time gaps $\Delta_{s,c,j}$ between the observations. For instance, say the first observation has $\Delta_{s,c,1} = H_t - \Upsilon_c = 0.5\Delta_\Upsilon$, occurring halfway into the time interval. The second observation is recorded at $H_{t+1} - H_t = 0.25\Delta_\Upsilon$ seconds later, a further quarter into the interval after H_t. Thus, the predictions $\hat{\mathbf{y}}$ are located at fractions 0.5 and 0.75 (=0.5+0.25) of the distance along the predicted straight line movement for each behavior.

Likewise the prediction error depends on these time gaps $\Delta_{s,c,j}$. The prediction errors $\mathbf{q}_c \sim \mathcal{N}_2(\mathbf{0},\ \mathbf{Q}_c \mid \lambda_c)$ and $\mathbf{r}_c \sim \mathcal{N}_2(\mathbf{0},\ \mathbf{R}_c \mid \lambda_c)$ apply over a full

---

If $j = 1$:
$$d\left(\mathbf{y}_{t(s,c,1)} \mid \mathbf{x}_{c-1}^{(n)}, \lambda_c^{(n)} = k\right) \sim \mathcal{N}_2\left(\boldsymbol{m}(\ell(\mathbf{x}_{c-1}^{(n)})),\ \mathbf{S}_{s,c,1}^{(n)} \mid \Delta_{s,c,1}\right),$$
$$\mathbf{S}_{s,c,1}^{(n)} = \mathbf{M}\left(\ell\left(\mathbf{x}_{c-1}^{(n)}\right)\right)\mathbf{P}_{c-1}^{(n)}\mathbf{M}\left(\ell\left(\mathbf{x}_{c-1}^{(n)}\right)\right)^T + \Delta_{s,c,1}^2\mathbf{r}_c^{(n)}$$
If $j > 1$: let $\ell_{s,c,j-1}^{(n)} = \left[\hat{\mathbf{y}}_{t(s,c,j-1)},\quad \ln(v_c) \mid k,\quad (\psi_{c-1}^{(n)} + \theta_c) \mid k\right]^T$
be the previous observation $(j-1)$'s predicted location, the simulated log-speed, and bearing for behavior $\lambda_c = k$
$$d\left(\mathbf{y}_{t(s,c,j)} \mid \mathbf{x}_{c-1}^{(n)}, \lambda_c^{(n)} = k\right) \sim \mathcal{N}_2\left(\boldsymbol{m}(\ell_{s,c,j-1}^{(n)}),\ \mathbf{S}_{s,c,j}^{(n)} \mid \Delta_{s,c,j}\right),$$
$$\mathbf{S}_{s,c,j}^{(n)} = \mathbf{M}\left(\ell_{s,c,j-1}^{(n)}\right)\mathbf{P}_{c-1}^{(n)}\mathbf{M}\left(\ell_{s,c,j-1}^{(n)}\right)^T + \Delta_{t(s,c,j)}^2\mathbf{r}_c^{(n)}$$

For $j > 1$, we continue along the same angle ψ_c with speed v_c simulated at regular step c and predict each observed location $\mathbf{y}_{t(s,c,j)}$ in turn, conditional on the expected value of the previous location $\hat{\mathbf{y}}_{t(s,c,j-1)}$.

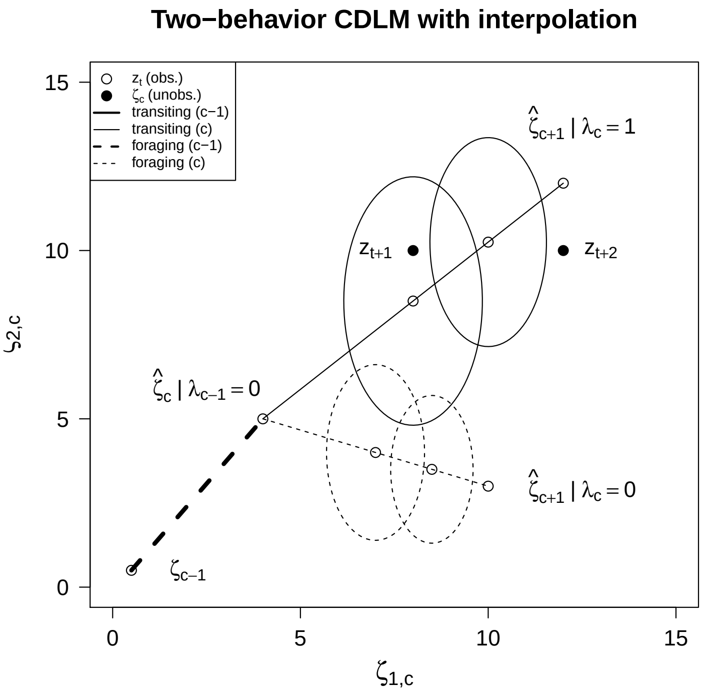


Figure 7.3: Interpolation predictive density $m_{\mathrm{interp}_{t|k}}$ illustrated by predicting two observations $\mathbf{z}_{(t+1):(t+2)}$ in the interval $(\Upsilon_c, \Upsilon_{c+1}]$ for each potential behavior λ_c.

time interval of Δ_Υ. Given movement from $\boldsymbol{\zeta}_{c-1}$, the prediction error of $\hat{\boldsymbol{\zeta}}_c$ is $\mathbf{q}_c$. Now, marginalizing from Υ_c to H_t, the time of the first observation, the additional prediction error should have covariance $\sqrt{\Delta_{s,c,1}}\mathbf{r}_c \mid \lambda_c$. Between each further observation in the interval, the prediction is conditioned on the previous prediction $\hat{\mathbf{y}}_t$. In Figure 7.3, this is shown by the error second prediction ellipse (if each represents, say, a 95% confidence ellipse) along each line being smaller than the first, since the second time gap $\Delta_{s,c,2}$ is smaller than the first, $\Delta_{s,c,1}$. As detailed in the algorithm, the interpolation weight $m_{\text{interp}_{t|k}}$ is calculated as the prediction interpolation density for $\hat{\mathbf{y}}_t$ at the observed value of $\mathbf{y}_t$ at H_t, and the joint weights $w_{c|k}$, if there are multiple observations, as the product across t in the interval, accounting for any transition probabilities $p_{c-1,\lambda_{c-1}\to k}$.

Because we do not know the true values of unobserved constant-interval locations $\boldsymbol{\zeta}_c$ for the observed data, we may want to test the performance of the algorithm on simulated data where we know the ground truth. The key difference between the interpolation and the regular CDLM is that the observed data, occurring at irregular intervals, are viewed as obscuring the 'true' underlying process with regular observations. Say we want to model the shark's irregularly-observed movement as occurring at a series of regular steps with constant time difference Δ_Υ. We can first generate a movement trajectory (with transition probabilities and other parameters) at equally-spaced times $\{\Upsilon_c\}$ with step Δ_Υ and behavior states $\{\lambda_c\}$, from which we can calculate the (true) density estimates of speed, turn angles, transition probabilities, and other parameters of interest. We then generate a series of irregular timesteps $\{H_t\}$ from an appropriate distribution, such as gamma or log-normal, to mimic the distribution of timesteps in the observed data (see Figure 7.1). Then let our observed simulated data be the movement at regular steps $\{\Upsilon_c\}$ interpolated to the irregular steps $\{H_t\}$. Importantly, we take the irregularly observed behaviors $\{\lambda_t\}$ as simply the values $\{\lambda_c\}$ of the regular intervals in which each irregular observation falls. This linear interpolation is outlined in 7.1.

Regular trajectory simulation and linear interpolation to irregular observations

---

1. Fix regular step length Δ_Υ and set maximum regular interval index C. Let regular times $\Upsilon_c = \Upsilon_0 + c\Delta_\Upsilon, \forall c = 0, \dots, C$.
2. Set initial position $\boldsymbol{\zeta}_0$ and behavior λ_0; if 2-D, set initial bearing ψ_0. Draw initial speed $v_0 \mid \lambda_0$.
3. Iterate over $c = 1, \dots, C-1$:
   (a) Set location $\boldsymbol{\zeta}_c = \boldsymbol{\ell}_\mu(\boldsymbol{\zeta}_{c-1}, v_{c-1}, \dots)$.
   (b) If multiple behaviors ($K > 1$), draw behavior $\lambda_c \mid \lambda_{c-1}$ (if not, just set $\lambda_c = 0$).
   (c) Draw speed $v_c \mid \lambda_c$, and if 2-D, turn angle $\theta_c \mid \lambda_c$.
4. End iteration over steps c. Now create 'observations' interpolated at non-constant time intervals:
5. Draw irregular step lengths $\{\Delta_t\}_{t=1}^T$ (for instance, from a log-normal distribution) such that $\sum_{t=1}^{T-1} \Delta_t < C\Delta_\Upsilon \leq \sum_{t=1}^T \Delta_t$, i.e. you have enough to cover the span of regular observations.
6. Optional: set final time gap $\Delta_{T-1} = \Upsilon_C - H_{T-1}$ so that the final observation $\mathbf{y}_t$ is at time $H_T = \Upsilon_C$ (usually $H_1 = \Upsilon_0 = 0$ for convenience).
7. Set irregular times $H_1 = \Upsilon_0$ and $H_t = H_{t-1} + \Delta_{t-1}, \forall t = 2, \dots, T-1$.
8. Iterate over irregular indices $t = 1, \dots, T$:
   (a) Let the regular interval observation H_t falls in, be indexed $c : \Upsilon_c < H_t \leq \Upsilon_{c+1}$.
   (b) Set behavior $\lambda_t = \lambda_c$ and speed $v_t = v_c$, the values observed in the regular interval (there may be more than one observation in each interval c). If 2-D, set bearing $\psi_t = \psi_c$.

    (c) Set location $\boldsymbol{\zeta}_t = \boldsymbol{\zeta}_c + (H_t - \Upsilon_c)v_c$ if 1-D (or use cosine and sine of bearing ψ_c if 2-D).

9. End iteration over non-constant timesteps t.

In the interpolation PF, particles are only potentially resampled in a regular interval in which irregular observations $\mathbf{y}_t$ fall. Between these, the states $\mathbf{x}_c$ (the shark's movement) are simulated using the existing movement parameters; we have the option to update parameters at each regular step c or update based on past values only at steps c where there are observations. Resampling weights $w_c^{(n)}$ depend, by definition, only on how movement at Υ_c (through the parameters learned up to then) match the observations in the interval $(\Upsilon_c, \Upsilon_{c+1}]$. Particularly if there is a long time gap between observations, it is possible that a particle's movement may have a high weight by chance. For instance, the particle may predict large zig-zags of movement, as opposed to a straighter trajectory, that happens to end up where the shark was observed. Therefore, we may want to resample particles more often than if we modeled the observed trajectory directly at the observed irregular time points without interpolation. We may, for instance, want to use the effective sample size threshold $N_{\text{thresh}} = N$, and thus we will resample at each regular Υ_c with observations.

The interpolation to irregular times can affect the observed densities of speeds and turn angles, and importantly, the distribution of observed behavior types and the transition probabilities between them. Based on experiments with the above, the densities of the simulated speeds and turn angles do not differ substantially between the irregular and regular observations in a way that would prevent learning the distributions of the regular observations from the irregular observations. Here, we assume that the gaps between observations are random and not related to the behavior or region the sharks are observed in. In reality there may some spatial association, such as if sharks spend time in muddy areas which block reception from the acoustic receivers and lengthen the time between observations.

7.2 Classification accuracy of behaviors

When comparing the irregularly-observed data and our model of the regular interval data, of particular interest is the degree to of agreement between which the observed behaviors $\{\lambda_t\}_{t=1}^T$ and the predicted behaviors $\{\lambda_c^{(n)}\}_{c=0}^C$ from the particle filter at the regular intervals $\{\Upsilon_c\}_{c=0}^C$. If we have simulated regular observations to begin with, then $\{\lambda_c\}_{c=0}^C$ (without the particle superscript) are true values of the behavior at the regular intervals, which are unobserved in the case of real data. On the actual shark data, the observed behaviors are determined by the classification boundary $\lambda_t = 0$ (foraging) if $v_t < 0.1$ m/s, as shown in Figure 2.8; although for the shark data this classification is to guide our modeling and they are not assumed to be absolute truth, it is still informative to measure the agreement.

Consider a set of irregular timesteps $H_1, \ldots, H_T$ and regular timesteps $\Upsilon_0, \ldots, \Upsilon_c$. Recall that the notation $t(s, c, \cdot)$ is the set of observation indices t whose clock times H_t fall in $(\Upsilon_c, \Upsilon_{c+1}]$, since we assume classification accuracy is calculated separately for each shark s. We can denote the overall behavior agreement between the behaviors of irregular observations λ_t and our particle filter $\lambda_c^{(n)}$ at the regular timesteps as $A\left(\{\lambda_t\}_{t=1}^T, \{\lambda_c^{(n)}\}_{c=0}^C\right)$ as

$$A\left(\{\lambda_t\}_{t=1}^T, \{\lambda_c^{(n)}\}_{c=0}^C\right) = \left(\sum_{c=0}^{C-1} \sum_{t\in t(s,c,\cdot)} \sum_{n=1}^{N} \mathrm{I}\left(\lambda_c^{(n)} = \lambda_t\right)\right) / (NT)$$

where $0 \leq A \leq 1$, with 1 indicates perfect agreement. We can also define behavior-specific agreement measures, for $k \in \{0, 1\}$:

$$A\left(\{\lambda_t\}_{t=1}^T, \{\lambda_c^{(n)}\}_{c=0}^C \mid k\right) = \left(\sum_{c=0}^{C-1} \sum_{t\in t(s,c,\cdot)} \sum_{n=1}^{N} \mathrm{I}\left(\lambda_c^{(n)} = \lambda_t = k\right)\right) / (N \times \sum_{t=0}^{T} \mathrm{I}(\lambda_t = k))$$

To measure the accuracy of the PF against the unknown true regular trajectory, assuming that both have the same constant timestep Δ_Υ, the accuracy formula is

$$A\left(\{\lambda_c\}_{c=0}^{C},\ \{\lambda_c^{(n)}\}_{c=0}^{C}\right) = \left(\sum_{c=0}^{C}\quad\sum_{n=1}^{N}\mathrm{I}\left(\lambda_c^{(n)} = \lambda_c\right)\right)/(N \times C)$$

To see an example of how behavior type classification accuracy changes with the degree of variation in time step size, we generated a simple one-dimensional (see Section 4.5 for the technical details of a PF with only one behavior type) trajectory with two behavior types: slow ($\lambda = 0$) and fast ($\lambda = 1$). For simplicity, here the distributions of velocity of the two types were separable, with $(v_t \mid \lambda_t = 0) \sim \mathcal{N}(2, 0.4)$ and $(v_t \mid \lambda_t = 1) \sim \mathcal{N}(9, 0.1)$. This represents the true unobserved trajectory.

As with the shark data, we imagine our observed sensor readings of the robot will occur at irregular times. When the irregularity (i.e., the variance of the observed time step lengths) increases, we expect the accuracy of behavior classification, and of location prediction in general, to decrease because there are now longer stretches of time without observations. We use the log-normal distribution $\Delta_t \sim \mathcal{Lognormal}(\mu = \ln(120), \sigma)$ to generate a sequence of observed irregular timesteps. For a log-normal distribution, its median is e^{μ} so all of our sequences were drawn from a distribution with median of 120, the same as the simulated true trajectory. To change the degree of irregularity, we use the true trajectory of regular-timestep locations ($\sigma =$'0' theoretically), and let $\sigma = 0.25,\ 0.50,\ 0.75,\ \ldots,\ 1.75,\ 2.00$ in turn; these give mean step lengths of $\mathrm{E}(\Delta_t) \approx 120,\ 124,\ 136,\ 159,\ 198,\ 262,\ 370,\ 555,\ 887$, respectively ($\mathrm{E}(\Delta_t) = \exp(\mu + \sigma^2/2)$). Figure 7.4 shows log-normal density functions for several choices of σ, when $\mu = \ln(120)$. We then generated an 'observed' dataset linearly interpolated (see 7.1) from the true regular steps using each sequence of irregular steps Δ_t. For each, the PF tried to recover a 100-timestep trajectory $\Upsilon_0, \ldots \Upsilon_{99}$ with $\Delta_\Upsilon = 120$ to match the unknown true trajectory.

Figure 7.5 shows the accuracy of behavior classification, both overall and behavior-specific, for the regular trajectory recovered from each of our irregularly observed datasets, which were interpolated from the same true regular

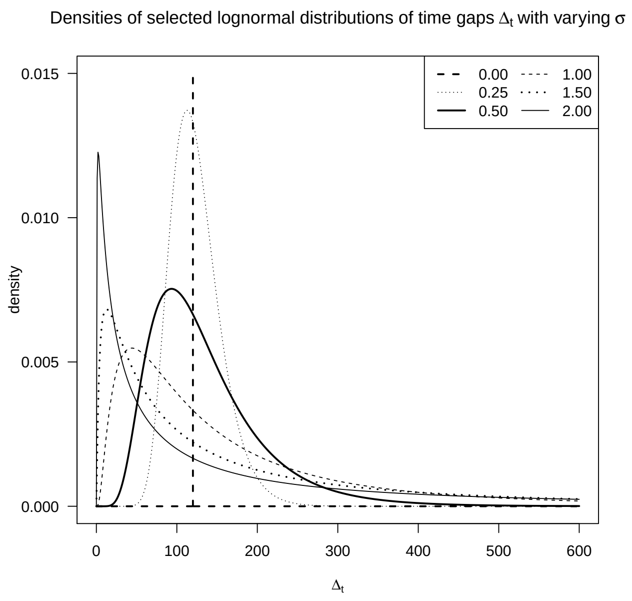


Figure 7.4: Log-normal distributions of time gaps $\Delta_t \sim \mathcal{L}ognormal\,(\mu, \sigma)$. For each, $\mu = \ln(120)$ so that the median is 120 seconds, but σ is allowed to vary.

trajectory. The left panel shows the accuracy of the PF regular trajectories compared with the irregular observations, and the right panel is the accuracy compared with the true regular behaviors, which for real data are of course unobserved. We see that keeping the median step length at 120, increasing the standard deviation of observed step length, and thus increasing irregularity of observations, decreases the accuracy both relative to the observed data and the true regular trajectory. Later, in Figure 8.5 we show results of accuracy on more complicated 2-D shark data.

7.3 Trajectory smoothing

The term 'filtering' refers to estimating the joint distribution of the states $\mathbf{x}$ and parameters θ conditional on all the observations $\mathbf{y}_{1:T}$, that is, the joint filtering distribution $d(\mathbf{x}_{0:T}, \boldsymbol{\theta} \mid \mathbf{y}_{1:T})$. This is achieved by the particle simulations: at the end of the process at time T, the particles are weighted so

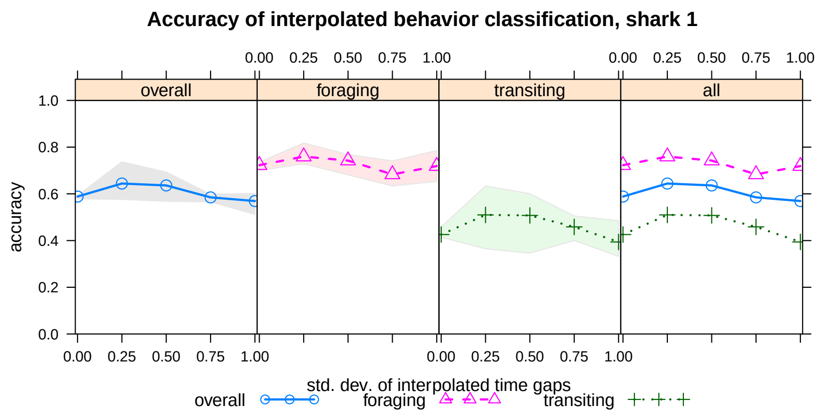


Figure 7.5: Example of behavior type classification accuracy measures on simulated one-dimensional trajectory. The PF estimates the locations and behaviors of a regular 300-step trajectory with $\Delta_\Upsilon = 120$ seconds. The classification accuracy of the PF in modeling the regular step behaviors from the irregular interpolations is calculated, compared with the (usually unknown) true regular trajectory. Five different random interpolations with the same time gap distributions were calculated at each level of irregularity (value of Δ_t standard deviation σ). The plotted line shows the mean accuracy across the five, with the vertical spread of the colored regions showing the minimum and maximum accuracy at each level.

that jointly their parameters $\boldsymbol{\theta}^{(n)}$ and state histories $\mathbf{x}_{0:T}^{(n)}$ approximate the filtering distribution. Once we have sequentially learned optimal values of the parameters $\boldsymbol{\theta}$, sometimes we want to estimate the joint **smoothing distribution**—that is, just $d(\mathbf{x}_{0:T} \mid \mathbf{y}_{1:T})$, the joint of the states conditional on the observations, marginalized over the parameters $\boldsymbol{\theta}$. For instance, since the state history $\mathbf{x}_{0:T}$ includes simulated values of the true unknown trajectory $\boldsymbol{\zeta}_{0:T}$, using the final optimal parameter values $(\boldsymbol{\theta}_T \mid \mathbf{y}_{1:T})$ estimated on all the observations, we may be able to improve the accuracy of our trajectory estimation. Note that we refer to a trajectory here because that is the context of our algorithm, but in general the states $\mathbf{x}$ can refer to any unobserved variables, not just in the context of tracking a moving object or animal. We use the subscript C in the following because the final parameters $\boldsymbol{\theta}$ and unobserved states $\mathbf{x}$ are assumed to be at regular intervals c while the observations are at irregular intervals t.

Carvalho et al. ([10]) present an algorithm for backwards smoothing that is an extension for when the parameters $\boldsymbol{\theta}$ are not fixed. The algorithm begins with the final simulated values of the states $\{\underline{\mathbf{x}}_T^{(n)}\}_{n=1}^N$ and proceeds backwards, resampling particles at $t-1$ according to how well their parameters match the subsequent particles at time t. We perform the smoothing algorithm on the states interpolated to the regular intervals $\{\Upsilon_c\}_{c=0}^C$, keeping the simulated behaviors λ fixed. At time c, let $\{\underline{\mathbf{x}}_c^{(n)}\}_{n=1}^N$ represent the smoothed set of the states. Note that the smoothing algorithm does not need to conduct interpolation calculations since it does not involve the observations $\mathbf{y}_{1:T}$ but just uses the interpolated output $\mathbf{x}_{0:C}$ from the interpolated PF.

Outline of smoothing for interpolated CDLM:

Set J as the number of smoothing iterations per time index. A common choice is $J = N$, where N is the number of particles. For C steps, this makes the complexity of the smoothing algorithm $O(CNJ)$, whereas filtering is only $O(CN)$.

However, if we do not draw values of the parameters $\boldsymbol{\theta}^{(n)}$ anew for all particles for each of the J iterations, as this is unnecessary, we reduce the complexity to just $O(CN)$. Note: all states $\mathbf{x}_c$ and parameters for $c = 0, \dots, C$ have been learned through a forward filtering algorithm.

```
1. Initialize by iterating over smoothing indices $j = 1, \dots, J$:
    (a) Draw one index $n_j$ uniformly from the indices $n = 1, \dots, N$.
    (b) Set $\underline{\mathbf{x}}_c^{(j)} = \mathbf{x}_c^{(n_j)}$ and $\underline{\lambda}_c^{(j)} = \lambda_c^{(n_j)}$.
2. End iteration over $j$.
3. For regular intervals $c = (C-1), \dots, 0$:
    (a) Iterate over all the particles. For $n = 1, \dots, N$:
        i. Let $\boldsymbol{\Omega}_{c|\lambda_c}^{(n)}$ be the hyper-parameters for parameters $\boldsymbol{\theta}^{(n)}$ for the
           given behavior $\lambda_c^{(n)}$ learned up to the final timestep $C$. Draw
           values of parameters $\boldsymbol{\theta}^{(n)} \sim p(\boldsymbol{\Omega}_{c|\lambda_c}^{(n)})$.
        ii. Iterate over possible values of $\lambda$. For $k = 1, \dots, K$:
            A. Draw transition probability values of $p_{\lambda_c \to k}^{(n)}$ from the appro-
               priate Dirichlet distribution.
        iii. End iteration over behaviors $k$.
    (b) End iteration over particles $n$.
    (c) Iterate over smoothing indices. For $j = 1, \dots, J$:
        i. Iterate over all particles. For $n = 1, \dots, N$:
            A. Let $w_{j,c}^{(n)} = \ell\left(\underline{\mathbf{x}}_{c+1}^{(j)} \mid \mathbf{x}_c^{(n)},\ \lambda_c^{(n)},\ \boldsymbol{\theta}^{(n)}\right) \times p_{\lambda_c \to \underline{\lambda}_{c+1}^{(j)}}^{(n)}$.
        ii. End iteration over particles $n$.
        iii. Draw one index $n_j$ from the indices $n = 1, \dots, N$ by the weights
             $\{w_{j,c}^{(n)}\}_{n=1}^{N}$.
        iv. Set $\underline{\mathbf{x}}_c^{(j)} = \mathbf{x}_c^{(n_j)}$ and $\underline{\lambda}_c^{(j)} = \lambda_c^{(n_j)}$.
    (d) End iteration over iterations $j$.
4. End iteration over timesteps $c$.
```

An additional variant is to not keep the latent variables (behaviors) $\lambda_c^{(n)}$ fixed as they were in filtering. Here we simply pick the smoothed behaviors at step c based on which best predicts the fixed behavior at $c+1$, similarly to the filtering CDLM.

Outline of smoothing for interpolated CDLM, allowing behavior λ to change:

```
1. Initialize by iterating over smoothing indices $j = 1, \ldots, J$:
   (a) Draw one index $n_j$ uniformly from the indices $n = 1, \ldots, N$.
   (b) Set $\underline{\mathbf{x}}_c^{(j)} = \mathbf{x}_c^{(n_j)}$ and $\underline{\lambda}_c^{(j)} = \lambda_c^{(n_j)}$.
2. End iteration over $j$.
3. For regular intervals $c = (C-1), \ldots, 0$:
   (a) Iterate over all the particles. For $n = 1, \ldots, N$:
       i. Iterate over possible values $k$ of $\lambda$ (in our case, $k \in \{0, 1\}$):
          A. Let $\boldsymbol{\Omega}_{c|\lambda_c=k}^{(n)}$ be the hyper-parameters for parameters $\boldsymbol{\theta}^{(n)}$ for
             behavior $k$ learned up to the final timestep $C$. Draw values
             of parameters $\boldsymbol{\theta}^{(n)} \sim p(\boldsymbol{\Omega}_{c|\lambda_c=k}^{(n)})$.
          B. For each potential value $k'$, draw transition probability val-
             ues of $p_{k \to k'}$ from the appropriate Dirichlet distribution.
       ii. End iteration over behaviors $k$.
   (b) End iteration over particles $n$.
   (c) Iterate over smoothing indices. For $j = 1, \ldots, J$:
       i. Iterate over particles. For $n = 1, \ldots, N$:
          A. For each behavior $k$, set behavior-conditional weights
             $w_{j,c|k}^{(n)} = \ell\left(\underline{\mathbf{x}}_{c+1}^{(j)} \mid \mathbf{x}_c^{(n)},\ \lambda_c^{(n)} = k,\ \boldsymbol{\theta}^{(n)}\right) \times p_{k \to \underline{\lambda}_{c+1}^{(j)}}^{(n)}$.
          B. Set overall particle weight $w_{j,c}^{(n)} = \sum_k w_{j,c|k}^{(n)}$.
       ii. End iteration over particles $n$.
       iii. Draw one index $n_j$ from the indices $n = 1, \ldots, N$ by the weights
            $\{w_{j,c}^{(n)}\}_{n=1}^{N}$.
       iv. Set $\underline{\mathbf{x}}_c^{(j)} = \mathbf{x}_c^{(n_j)}$.
       v. Draw behavior $\underline{\lambda}_c^{(j)} \sim \mathcal{Ber}\left(w_{j,c|0}^{(n_j)}/w_{j,c}^{(n_j)}\right)$ instead of automatically setting it to be $\lambda_c^{(n_j)}$.
   (d) End iteration over indices $j$.
4. End iteration over timesteps $c$.
```

The idea of smoothing is that the distributions of the smoothed states $\underline{\mathbf{x}}_c$ should be "smoother" in some way—since the full information at time C is used—but not necessarily more accurate, in terms of their distance to the observations, than the unsmoothed $\mathbf{x}_c$ after filtering. If the states $\underline{\mathbf{x}}_c$ include a location variable $\underline{\boldsymbol{\zeta}}_c$, then the resulting trajectory should be visually smoother than the unsmoothed $\boldsymbol{\zeta}_c$. We illustrate this by generating a one-dimensional ('robot') trajectory (see Section 4.5) with 50 steps and regular time gap $\Delta_\Upsilon = 120$, and only one behavior type, since this is harder to visualize when the overall distribution of speeds depends on more than one behavior. Here, the true velocity was distributed $v_c \sim \mathcal{N}(\alpha = 6, \sigma^2 = 1.6)$ and allowed to be negative, meaning backwards motion. In this illustration, we assume the constant-length steps c are observed without interpolation.

In Figure 7.6, we compare the results of the 1-D EKF with 100 particles (N) on the regular observations, before (unsmoothed) and after smoothing, in predicting the robot's true regular step location 'X' ($\boldsymbol{\zeta}_c$, here just the univariate ζ_c). The top left panel shows that the variance at each timestep of location predictions across particles is generally lower for the smoothed compared to the unsmoothed ones. Increased variance of location prediction implies the unsmoothed particles have a higher variance in estimation of the distribution of velocity, and hence will generate more jagged trajectories of movement. The top right panel, however, shows that the accuracy of estimating the true locations, as measured by root mean square error[3] (RMSE), is essentially the same between the smoothed and unsmoothed particles; the smoothed ones sometimes have a lower RMSE. This is not always the case, as shown by examples

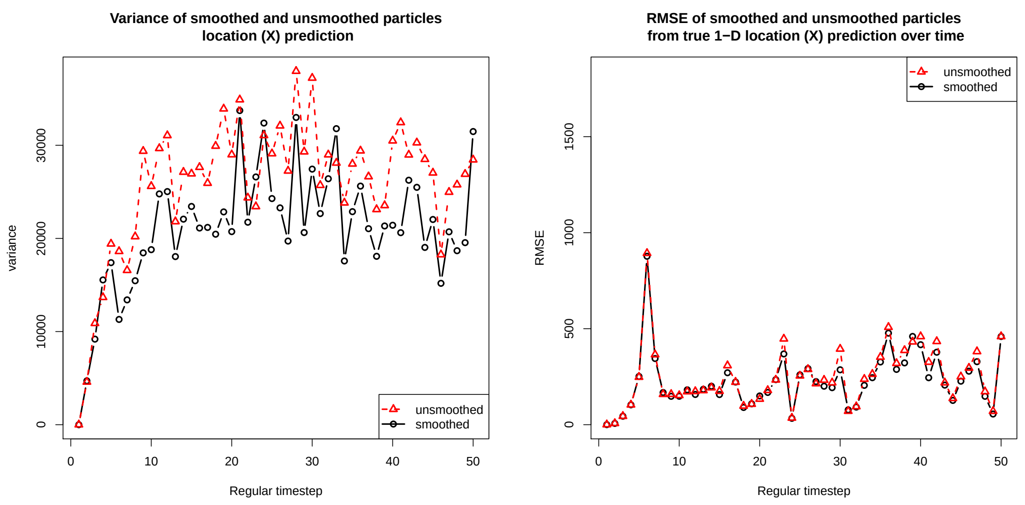


Figure 7.6: Results of 1-D ‘robot’ EKF with one behavior on simulated regular steps, with the particle locations (‘X’) predicted before (unsmoothed) and after smoothing (smoothed). Comparison of variance of locations (left) and root mean squared error from the true locations (right) at each time step.

in [45], where the smoothing achieves both lower variance and RMSE. The RMSE remains relatively stable over time because the particles are resampled often; otherwise, errors would accumulate and particles with less successful predictions would not be weeded out.

Recall that the smoothing calculations do not involve the measurements $\mathbf{y}_t$ but rather just the state equation $\ell(\mathbf{x}_c \mid \mathbf{x}_{c-1})$. In the particle filtering algorithm, the posterior density (to get the weights) was only evaluated at $\mathbf{y}_t$, rather than at $\mathbf{x}_c$, but in smoothing, the resampling weights $w_{j,c}^{(n)}$ are evaluated at $\ell\left(\mathbf{x}_{c+1}^{(j)} \mid \mathbf{x}_c^{(n)}\right)$ (we use regular indices c because we assume regular interpolation). Thus, the density evaluation must take into account the fact that the bearing ψ_c is distributed wrapped normal rather than unwrapped, as outlined in Section 5.2. Since the bearing is modeled to be independent

[3]For true location ζ_t and estimate $\hat{\zeta}_t^{(n)}$ from a particle n from a total of N particles, the RMSE at time t is defined as $\sqrt{(1/N)\sum_{n=1}^{N}\left(\zeta_t - \hat{\zeta}_t^{(n)}\right)^2}$, a sample estimate of $\sqrt{\mathrm{E}\left((\zeta_t - \hat{\zeta}_t)^2\right)}$.

of the other state variables $\mathbf{x}_c$, we can multiply the densities to obtain the joint density. Denote $\mathring{\mathbf{x}}_c = \begin{bmatrix} \zeta_{1,c} & \zeta_{1,c} & \ln(v_c) \end{bmatrix}^T$, namely the state variables $\mathbf{x}_c$ excluding the bearing ψ_c. Thus, for the shark EKF, the likelihood calculation in the smoothing algorithm (Section 7.3) for the weights $w_{j,c}^{(n)}$ would need to be

$$\ell\left(\underline{\mathbf{x}}_{c+1}^{(j)} \mid \mathbf{x}_c^{(n)}\right) = \mathcal{N}_3\left(\underline{\mathring{\mathbf{x}}}_{c+1}^{(j)} \mid \mathbf{x}_c^{(n)},\ \lambda_c^{(n)} = k,\ \boldsymbol{\theta}^{(n)}\right) \times \mathcal{WN}\left(\underline{\psi}_{c+1}^{(j)} \mid \psi_c^{(n)},\ \lambda_c^{(n)} = k,\ \boldsymbol{\theta}^{(n)}\right)$$

CHAPTER 8

SYNTHETIC SHARK DATA EKF SIMULATIONS

8.1 Simulation setup

To assess our interpolation CDLM algorithm's ability to model the parameters of the irregularly-observed shark trajectories, we can follow the approach introduced in Section 7.1. In our interpolation approach we try to model the observed data as being an irregularly-observed interpolation of regularly-occurring measurements. The constant-length step trajectory is considered the underlying true but unobserved trajectory. This is sensible because certain parameters should be updated only based on observations that are equally time-weighted. For instance, in the error covariance matrices, the degrees of freedom are updated by a constant each time, reflecting the number of observations, so either the data have to be regular or we have to scale the update by the amount of time that has passed.

Similarly, the transition probabilities are updated based on simulated counts of transitions $n_{c,c+h,i\to j}$ between steps c and $c+h$. If a transition between observations of different behavior, say, two minutes apart is counted the same as if they, say, were ten minutes apart, this is not sensible. The transition between these two observations does not represent the 'true' set of transitions in the ten minutes. Defining an interpolation to regular intervals of small equal length Δ_Υ essentially defines Δ_Υ as the smallest and common unit of time under consideration. The definition of 'small' will depend on the context, but for our purposes we say it is on the order of, say, 30 seconds to 3 minutes.

A two-dimensional simulation is an approximation of what the observed shark trajectories look like; it simulates a random walk in 2-D space constrained by the borders of the tidal basin. A starting point $\boldsymbol{\zeta}_0$ is selected at random in the boundary of the tidal basin, and a random initial bearing angle ψ_0 and behavior type λ_0 is chosen. The behavior type λ_c at each step is chosen by a predetermined probability transition matrix $P_0^{(r)}$, possibly one with more than one region. The turn angle θ_c and speed v_c are drawn from their distributions conditional on λ_c. If the resulting next coordinate $\boldsymbol{\zeta}_{c+1}$ is outside of the boundary, the speed and turn are simulated again until the next point is inside. A more complicated method would be to include a repelling effect from the boundaries so the shark 'chooses', say, a turn angle that will cause it to turn it away from the boundary if it was headed towards it (see discussion in Section 5.5).

A reasonable setup (that is similar to the observed shark data) is the following. The resulting simulated trajectories are shown in Figure 8.1.

- Constant-length step $\Delta_\Upsilon = 120$ seconds.
- Two behaviors $\lambda_c \in \{0, 1\}$ (foraging and transiting).
- Two roughly equal regions ($R = 2$).
- Maximum $C = 200$ regular steps for each shark.
- Distribution of log-speed:
  - Foraging: $\ln(v_c) \mid (\lambda_c = 0) \sim \mathcal{N}(\alpha_c = -4,\ \sigma_c^2 = 1)$
  - Transiting: $\ln(v_c) \mid (\lambda_c = 1) \sim \mathcal{N}(\alpha_c = -1.6,\ \sigma_c^2 = 0.25)$

  These correspond to average speeds v_c of approximately 0.018 and 0.202 m/s.

- Distribution of turn angle θ_c:
  - Foraging: $\theta_c \mid (\lambda_c = 0) \sim \mathcal{N}(\beta_c = 0,\ \tau_c^2 = 1)$
  - Transiting: $\theta_c \mid (\lambda_c = 1) \sim \mathcal{N}(\beta_c = 0,\ \tau_c^2 = 0.25)$
- $R = 2$ regional behavior probability transition matrices of

$$P^{(1)} = \begin{bmatrix} 4/5 & 1/5 \\ 1/3 & 2/3 \end{bmatrix} \text{ and } P^{(2)} = \begin{bmatrix} 5/6 & 1/6 \\ 1/2 & 1/2 \end{bmatrix}$$

- Behavior interaction effects:
  - Sharks 1 and 2 are simulated independently without any interaction parameters.
  - Sharks 3 and 4 are simulated to have behavior influenced by neighboring observations of sharks 1 and 2 and each other. The neighborhood is defined to be all observations occuring within $\delta_\Upsilon = 30$ minutes before and within a spatial radius of $\delta_\zeta = 150$ meters. For sharks 3 and 4, $\eta_{k=0}$ was 2 and 1, respectively. With $\pi_{c,s,0}$ being the proportion of spatial-temporal neighboring other shark observations that are foraging, the interaction effect is $\rho_{c,k=0} \sim \mathcal{L}ognormal(\pi_{c,s,0}\eta_{k=0},\ \tau_{k=0})$, with $\tau_{k=0} = 0.75$. Thus, shark 3 was simulated to be more strongly influenced by neighbors than was shark 4.

As before, the simulated 'true' trajectories are then interpolated to irregular steps (see Section 7.1) by generating $\Delta_t \sim \mathcal{L}ognormal(\mu = \ln(120),\ \sigma)$, where $\sigma =$'0' (i.e., no interpolation), 0.25, 0.5, 0.75, 1, 1.25, 1.5, 1.75, and 2, in turn; we then run a joint PF on the interpolated irregular steps. The process is repeated five times, so that at each level of σ (step size variability) we have five different random samples of step size from the same distribution, to account somewhat for potential variation in step sizes.

- Variance (σ^2) of log-speed $\ln(v_c)$:

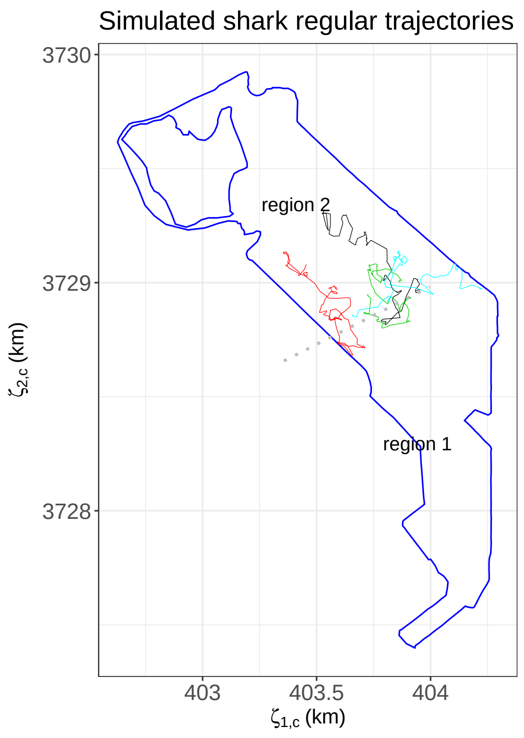


Figure 8.1: Example of four shark trajectories (colors) simulated in two regions, shown by the dotted lines. Two of the sharks (1 and 2, black and red lines) are simulated independently; sharks 3 and 4 (blue and green) are simulated independently but take into account behavioral influence from the others.

  - Foraging: $\sigma_c^2 \mid (\lambda_c = 0) \sim \mathcal{G}^{-1}(a_0 = 8,\ b_0 = 5)$
  - Transiting: $\sigma_c^2 \mid (\lambda_c = 1) \sim \mathcal{G}^{-1}(a_0 = 8,\ b_0 = 3)$

- Variance (τ^2) of turn angle θ_c:
  - Foraging: $\tau_c^2 \mid (\lambda_c = 0) \sim \mathcal{G}^{-1}(a_0 = 8,\ b_0 = 4)$
  - Transiting: $\tau_c^2 \mid (\lambda_c = 1) \sim \mathcal{G}^{-1}(a_0 = 8,\ b_0 = 1)$

- Transformed speed $\ln(v_c)$:
  - Foraging: $\ln(v_c) \mid (\lambda_c = 0) \sim \mathcal{N}(\alpha_c = -4, \kappa_{0,0} \times \sigma^2)$
  - Transiting: $\ln(v_c) \mid (\lambda_c = 1) \sim \mathcal{N}(\alpha_c = -1.3, \kappa_{1,0} \times \sigma^2)$

- Turn angle θ_c:
  - Foraging: $\theta_c \mid (\lambda_c = 0) \sim \mathcal{W}\mathcal{N}(\beta_c = 0, \kappa_{3,0} \times \tau^2)$
  - Transiting: $\theta_c \mid (\lambda_c = 1) \sim \mathcal{W}\mathcal{N}(\beta_c = 0, \kappa_{4,0} \times \tau^2)$

- NIG degrees of freedom (above) $\kappa = 20$.

- Probability transition probabilities, for regions $r = 1, \ldots, R$:
  - From foraging: $(p_{0\to0},\ p_{0\to1}) \sim \mathcal{Dir}_2(\alpha_{0\to0} = 8,\ \alpha_{0\to1} = 2)$
  - From transiting: $(p_{1\to0},\ p_{1\to1}) \sim \mathcal{Dir}_2(\alpha_{1\to0} = 2,\ \alpha_{1\to1} = 4)$

  The parameters are the same for each region to see if we can learn the differences between the regions. This corresponds to an average transition probability matrix $P = \begin{bmatrix} 4/5 & 1/5 \\ 1/3 & 2/3 \end{bmatrix}$; the variances depend on the absolute, not relative, magnitude of the Dirichlet parameters $\boldsymbol{\alpha}$.

- Location error covariance matrices: Let $\mathbf{\Lambda} = \begin{bmatrix} 1.00 & -0.30 \\ -0.30 & 1.25 \end{bmatrix}$.
  - Error covariance between states for $\boldsymbol{\zeta}_c \mid \mathbf{x}_{c-1}$ (not dependent on previous behavior λ_{c-1}):

  * $\Sigma_{\zeta_0} \sim \mathcal{W}_2^{-1}(\mathbf{\Lambda}_{\zeta_0},\ \eta_{\zeta_0})$ where degrees of freedom $\eta_{\zeta_0} = 20$ and scale matrix $\mathbf{\Lambda}_{\zeta_0} = 0.5\eta_{\zeta_0}\mathbf{\Lambda} = \begin{bmatrix} 10.0 & -3.0 \\ -3.0 & 12.5 \end{bmatrix}$.

- Error covariance between for observations $\mathbf{y}_c \mid (\mathbf{x}_c, \lambda_c)$:
  $\Sigma_{z_0} \mid \lambda_c \sim \mathcal{W}_2^{-1}(\mathbf{\Lambda}_{z_0,\lambda_0},\ \eta_{z_0,\lambda_0})$, where
  * Degrees of freedom $\eta_{z_0,\lambda_0=0} = \eta_{z_0,\lambda_0=1} = 20$
  * Scale matrices for foraging $\mathbf{\Lambda}_{z_0,\lambda_0=0} = 5\eta_{z_0,\lambda_0=0}\mathbf{\Lambda} = \begin{bmatrix} 100 & -30 \\ -30 & 125 \end{bmatrix}$ and transiting $\mathbf{\Lambda}_{z_0,\lambda_0=1} = 15\eta_{z_0,\lambda_0=1}\mathbf{\Lambda} = \begin{bmatrix} 300 & -90 \\ -90 & 375 \end{bmatrix}$.

• Maximum intervals to simulate without observations: $c_{\text{thresh}} = 10$.

• Interaction parameters (see Section 6.4):

  - The neighborhood influence structure used to simulate the data was assumed to be known. Neighborhoods $N(\cdot)$ are defined as having a spatial radius $\delta_{\boldsymbol{\zeta}}$ of 150 meters and a time radius δ_{Υ} of 30 minutes prior to the current time Υ_c.
  - For a given fraction π_λ of a shark's neighbors having foraging behavior, the spatial interaction effect is $\rho_{c,s,0} \sim \mathcal{Lognormal}(\pi_{c,s,0}\eta_0,\ \epsilon_0)$, where,
  - Precision (assumed known) $\epsilon_0 = 2$, and
  - Spatial interaction strength $\eta_\lambda \sim \mathcal{N}(\omega_{\lambda,0} = 0,\ \tau_{\lambda,0})$. We hope our PF will learn the mean values η_0 of the parameter η used to generate the synthetic data.

• Behavior rescaling weights (see Section 5.6): $a_0 = 1, a_1 = 2$.

The positional covariance matrix parameters are chosen roughly to represent the variance in coordinates along each axis; for instance, the range along the vertical axis ζ_2 is larger than the horizontal ζ_1. For an inverse Wishart distribution $\mathbf{X} \sim \mathcal{W}_p^{-1}(\mathbf{\Lambda}, \eta)$, we have $\mathrm{E}(\mathbf{X}) = \frac{1}{\eta - p - 1}\mathbf{\Lambda}$. Therefore, for instance, the prior on the prediction error when foraging of $\mathbf{y_c} \mid (\mathbf{x}_c, \lambda_c = 0)$ has expected value $\approx \begin{bmatrix} 5.9 & -1.8 \\ -1.8 & 7.4 \end{bmatrix}$. The prediction error when transiting is larger than when foraging, because in transiting the sharks travel faster, and thus the prediction uncertainty is higher. Furthermore, the covariance error should represent the prediction uncertainty for the chosen regular step length Δ_Υ; therefore, if we increase Δ_Υ, the error scale matrix components should increase in magnitude. As mentioned in Section 5.6, the rescaling weights a_0, a_1 may be necessary for better estimation of transiting behavior parameters. Naturally, we hope that we can model this behavior well enough by choosing appropriate prior distributions, such as allowing more positional prediction error for transiting, without introducing this somewhat makeshift adjustment.

The stepwise position error (Σ_{ζ_c}) in the absence of observations should not be too small because if the we simulate several regular intervals in a row without observations, we should accumulate uncertainty. Nevertheless, we need to put a limit on the amount of uncertainty we wish to deal with. The parameter $c_{\text{thresh}} = 10$ is the maximum number of regular intervals to simulate in a row without an observation. Say for instance $\Upsilon_c < H_t \leq \Upsilon_{c+1} < \Upsilon_{c+h} < H_{t+1} \leq \Upsilon_{c+h+1}$, where $h > c_{\text{thresh}}$, meaning the next observation $\mathbf{y}_{t+1}$ after $\mathbf{y}_t$ is more than c_{thresh} regular intervals away, then we simulate movement for regular steps $\Upsilon_{c+1}, \ldots, \Upsilon_{c+c_{\text{thresh}}}$, and then jump to Υ_{c+h}. If we do not have observations for a long enough period ($H_{t+1} - H_t$ is too big), we should just say we don't have a good idea of where the shark is to try to predict.

For the spatial interaction parameters, the priors are the same for both behaviors, because we do not wish to bias the results too much with our assumptions. The prior distribution on η_λ has mean 0 and precision τ_λ so that the initial assumption, unless the data teach us otherwise, is that there is close to no interaction.

8.2 Simulation results

The following results are from PF simulations based on synthetic shark trajectories generated at regular time gaps. The 'observed' irregular time gaps are generated from a log-normal distribution; the trajectories are interpolated to the irregular times, from which we try to recover the behavior at the unobserved regular intervals. The amount of irregularity in the observed time gaps is varied by changing the parameter σ of the log-normal distribution. In each case, the observed irregular steps cover 100 regular steps of length $\Delta_\Upsilon = 120$ seconds. On each irregularly-interpolated dataset, five repetitions of the PF are done to account for possible variability in the results.

To first illustrate the effect simulating time step lengths has on interpolation success, in Figure 8.2 we show the first 50 regular steps of length 120 seconds (thin lines). For comparison, we simulate step lengths from $\Delta_t \sim \mathcal{Lognormal}(\mu = \ln(120),\ \sigma)$, where $\sigma = 0.25, 0.75, 1.50, 2.00$, and then plot the times (i.e., the cumulative sums of the step lengths) at which these observations would occur, in dotted red lines. Two things are apparent:

1. Because increasing σ increases the variance of the step sizes, since log-normal is a right-skewed distribution, the mean step size $\mathrm{E}(\Delta_t)$ increases as well, as we have seen. Thus, the total number of irregular steps T needed to cover a fixed amount of time $C\Delta_\Upsilon$—a fixed number of regular steps C—decreases. This means that the total number of observations T that we have to estimate the trajectory with decreases with increasing σ. This is illustrated in more detail in Figure 8.3 for the simulated shark data used.

2. When σ, and thus irregularity, increases, the fewer observations that we *do* have tend to occur at less evenly-spaced time H_t relative to the true time Υ_c. This means, as we see in the third and fourth rows, that we may end up with multiple observations in a single regular interval $(\Upsilon_c, \Upsilon_{c+1}]$, as well as long stretches of time between consecutive observed times H_t

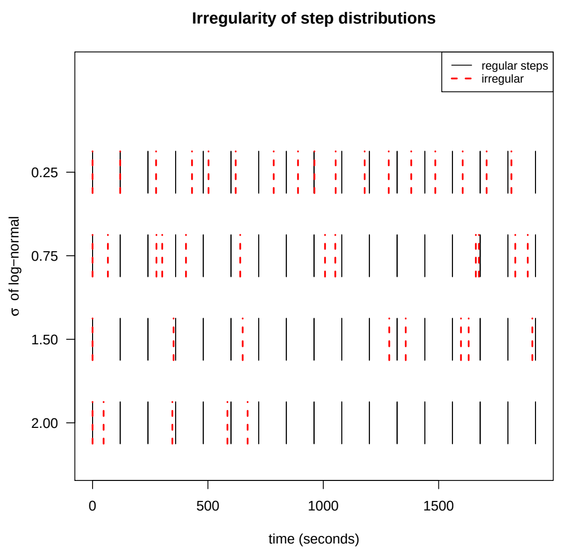


Figure 8.2: Illustration of effect of simulating time steps from a log-normal distribution with $\mu = \ln(120)$ and the σ parameter of the log-normal increasing.

and H_{t+1}. The more unique constant intervals $c$our observations fall in, the more complete information we have about the shark's trajectory. Both of these aspects make interpolating back to the true regular steps more difficult when our observations are irregular.

Figure 8.4 shows, for shark 3 (green in Figure 8.1), simulated trajectories for one of the five repetitions at values of $\sigma = 0$ (meaning the true data), $0.75, 1.25$, and 2.00 for the step size distribution. In each panel, the red lines show the observed trajectory resulting from interpolating the true unknown trajectory (the upper left panel) at step sizes with the different values of σ. As mentioned, increasing irregularity according to this distribution means that the number of observations decreases; because of this, the distortion of the shape of the observed trajectory (since it is linearly interpolated) from the true one increases. In each case, we simulate particle locations at $C = 100$ regularly-spaced time intervals based on the observed interpolated data. In each panel, a 2-D density plot of all of the particle locations is shown under

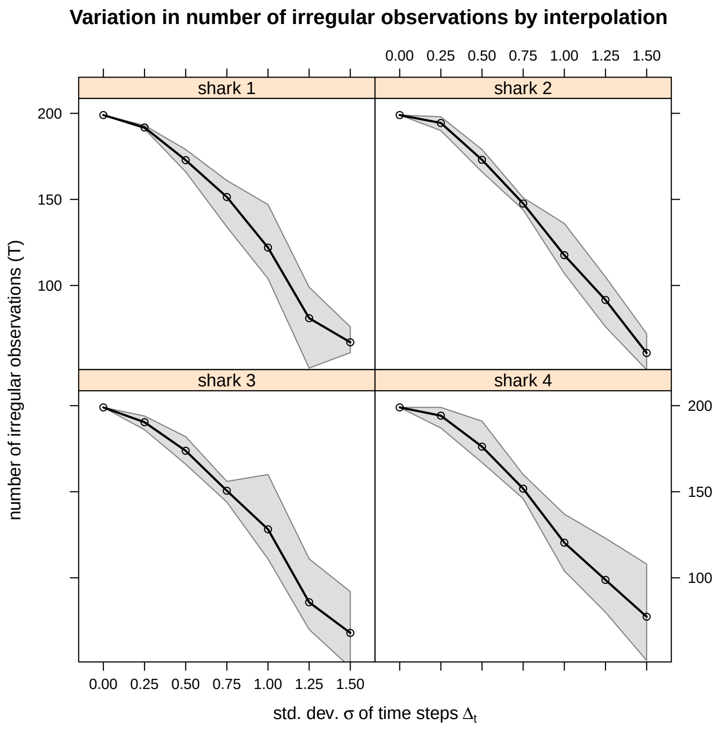


Figure 8.3: Mean and range of number of observations T in interpolated datasets, across five repetitions at each level of irregularity. The mean number of observations decreases and the variance increases with σ, the standard deviation of the log-normal distribution of Δ_t, the length of time between 'observed' times H_t.

the red trajectory. The darker the areas around the trajectory, the more concentrated the particle predictions there. This figure shows us that even with few observations, our PF can recover the observed trajectory well enough. As long as the observed trajectory $\boldsymbol{z}_t$ does not differ too much from the true trajectory $\boldsymbol{\zeta}_c$ in overall shape, the particles will reasonably approximate the true trajectory as well.

To demonstrate that we can recover the true regular step behavior types from the irregular observations, we calculate the classification accuracy (foraging or transiting) of the $C = 200$ regular steps simulated in the PF compared to both the behaviors at the T observed irregular steps (left column) and the C unobserved regular steps (right column) in Figure 8.5. The classification accuracy is calculated at the different levels of irregularity of step size (value of σ in the log-normal distribution of Δ_t).

8.3 Behavior classification accuracy

Since we are interested ultimately in inferring the relative prevalence of behavior types (foraging or transiting) for the actual shark data, we will now examine the success of the interpolation in modeling the behavior type. For each of the four synthetic sharks, the true constant-interval trajectory $\boldsymbol{\zeta}_{0:C}$ at times $\Upsilon_{0:C}$, which was simulated using behaviors λ_c at each interval. Interpolation to synthetic irregularly-observed trajectories $\boldsymbol{z}_{1:T}$ at times $H_{1:T}$, based on simulated time step lengths Δ_t, was conducted, where the unobserved behaviors λ_t were taken to be $\lambda_t = \lambda_{c'}$, where $c' := c\colon \Upsilon_c < H_t \leq \Upsilon_{c+1}$. That is, the behavior λ_t at observed time H_t is taken to be the behavior λ_c of the constant-length interval $(\Upsilon_c, \Upsilon_{c+1}]$ that H_t falls in. Note, there may be more than one observation H_t in each interval c. We wish to see if we can predict which (unobserved) behavior that shark had at each H_t, given only the observed locations $\boldsymbol{z}_t$ and the observed time gaps Δ_t between them.

We use the three accuracy measures discussed in Section 7.2. The measure $A(\cdot)$ is the overall classification accuracy measure of λ_t, while $A(\cdot \mid 0)$ and

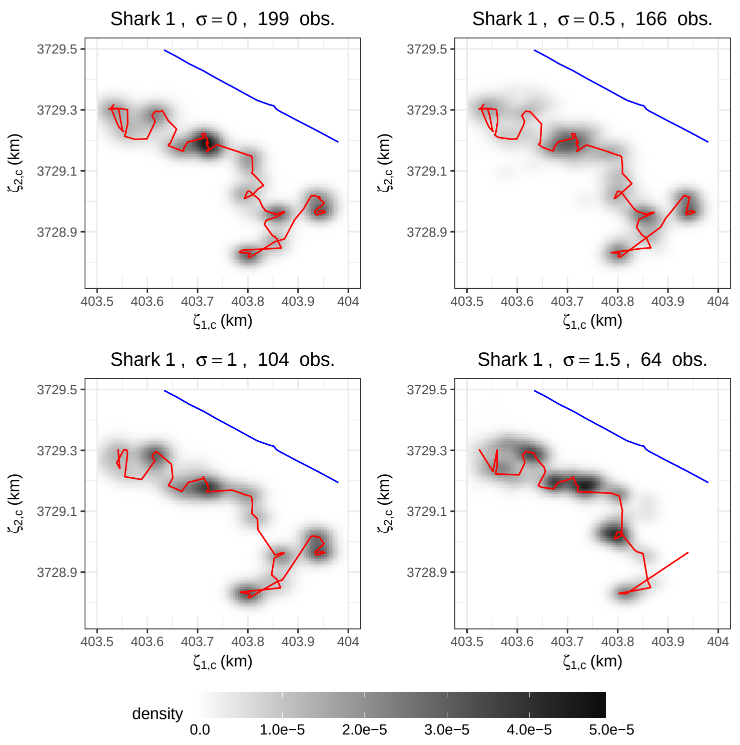
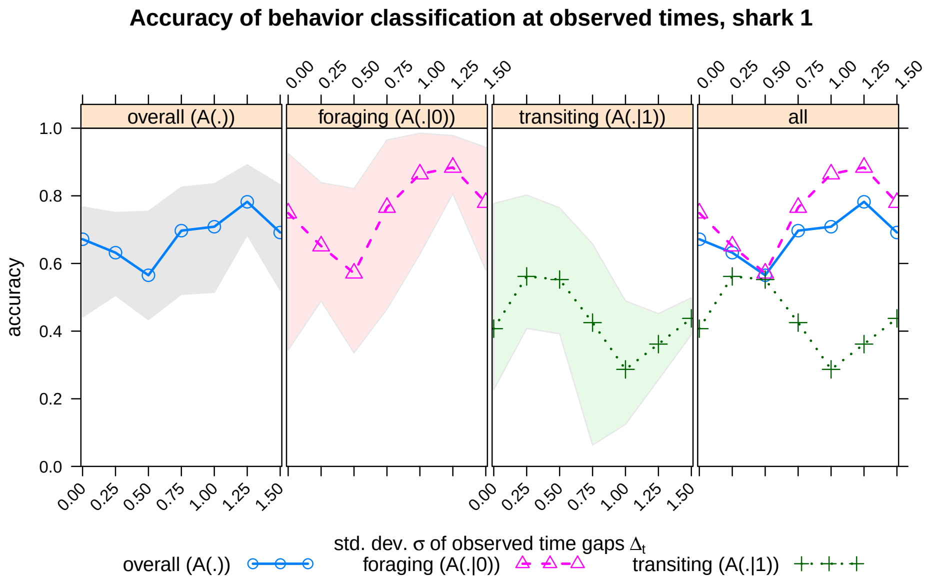


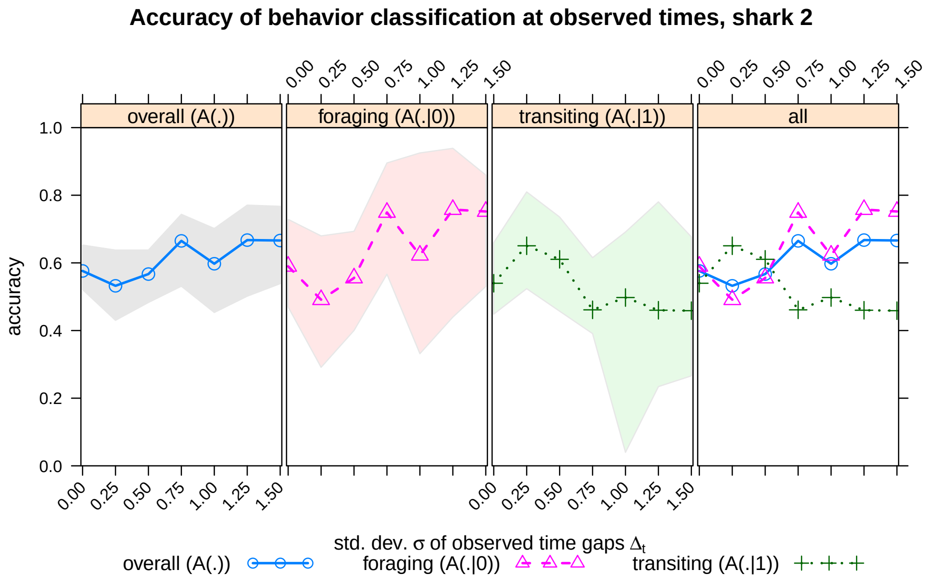
Figure 8.4: Spatial density (gray) of particle simulations of true locations $\boldsymbol{\zeta}_c$ for one instance of interpolating a true unobserved synthetic trajectory at regular steps c to observed locations (red lines), with non-constant time gaps Δ_t simulated with a log-normal distribution with various values of σ. The upper left panel shows $\sigma = 0$, namely no interpolation, so the observed data is the true trajectory, and the particles are simulated to the same steps c as the true one. With higher σ, the time gaps Δ_t are more variable, so the resulting observed trajectory $\boldsymbol{z}_t$ tends become a worse representation of the unobserved trajectory $\boldsymbol{\zeta}_c$.

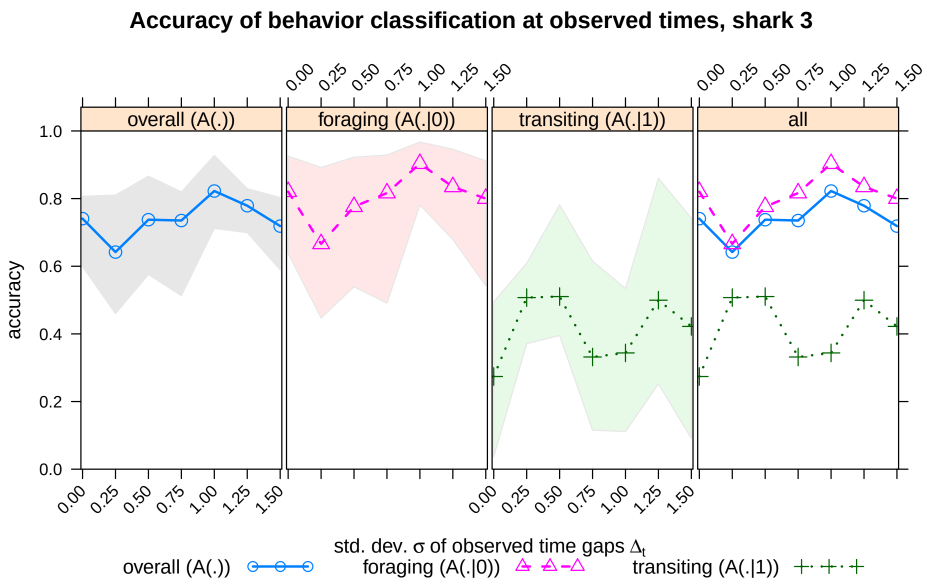
$A(\cdot \mid 1)$ are the classification accuracy measures conditional on λ_t being either foraging (0) or transiting (1). Figure 8.5 shows the accuracy measures for each shark at each level σ of irregularity of the observed time gaps Δ_t, across each of the five repetitions of the PF. The dashed line shows the mean accuracy, and the shaded areas show the minimum and maximum accuracies across the five repetitions.

The results for all four sharks are similar. Interestingly, the mean accuracies A do not display a clear trend as the irregularity σ increases, though the variability of each accuracy performance tends to increase. This is probably because with higher standard deviation σ, the placement over the time range $[\Upsilon_0, \Upsilon_C]$ of the observed times $H_{1:T}$ becomes more irregular and non-uniform, and therefore at a given higher level of σ, the observed trajectories $\boldsymbol{z}_{1:T}$ in each repetition (each new draw of time gaps $\{\Delta_t\}$) will tend to look different from each other. In contrast, when σ is low, the interpolate observed trajectories will look more similar, and thus classification performance on them is likely more stable. Furthermore, higher σ means the number of observations T is smaller on average, since the average time gap Δ_t is longer, and classification accuracy on a smaller sample is likely to be more variable than on a larger sample of observations when σ is smaller. Note that the classification accuracy for transiting, $A(\cdot \mid 1)$, tends to be smaller than for foraging. This may be because transiting is a rarer behavior and is less likely to be predicted, even with the weight rescaling factors a_1.

Figure 8.3 shows how the number of observations in the interpolated datasets decrease with the level of irregularity in step size. Even though the log-normal distributions of Δ_t are chosen to keep the median step size the same, because the log-normal distribution is right-skewed, the mean step size increases exponentially as we increase the standard deviation. Since the interpolated datasets all are based on a true dataset of the same time span, a large step size between observations, which is more likely with increased irregularity, means a larger portion of the true data is unobserved. Larger mean step sizes means both that we tend to have both fewer observations and longer spans of times with-


Accuracy of behavior classification at observed times, shark 1
overall (A(.))
foraging (A(.|0))
transiting (A(.|1))
all
accuracy
0.0
0.2
0.4
0.6
0.8
1.0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
std. dev. σ of observed time gaps Δ_t
overall (A(.))
foraging (A(.|0))
transiting (A(.|1))


Accuracy of behavior classification at observed times, shark 2
overall (A(.))
foraging (A(.|0))
transiting (A(.|1))
all
accuracy
0.0
0.2
0.4
0.6
0.8
1.0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
std. dev. σ of observed time gaps Δ_t
overall (A(.))
foraging (A(.|0))
transiting (A(.|1))



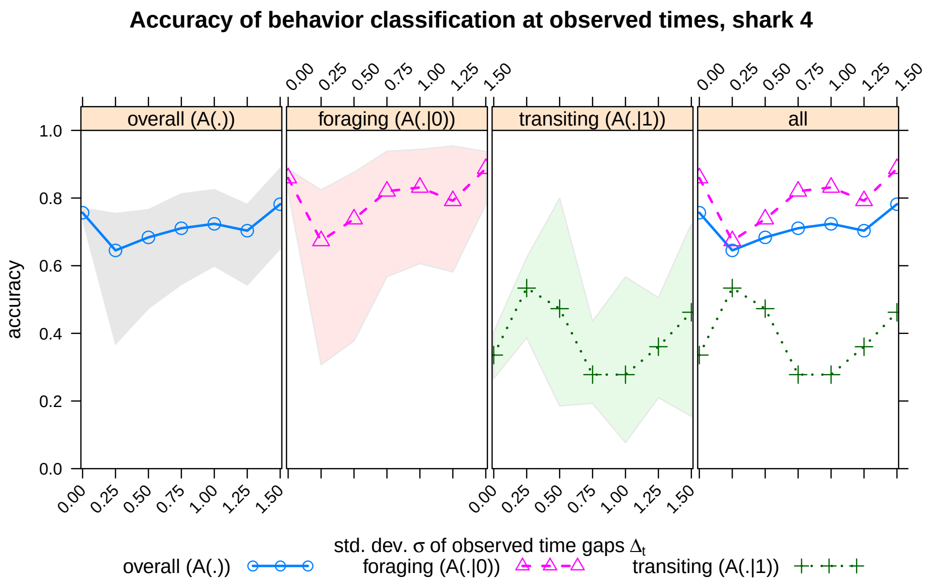
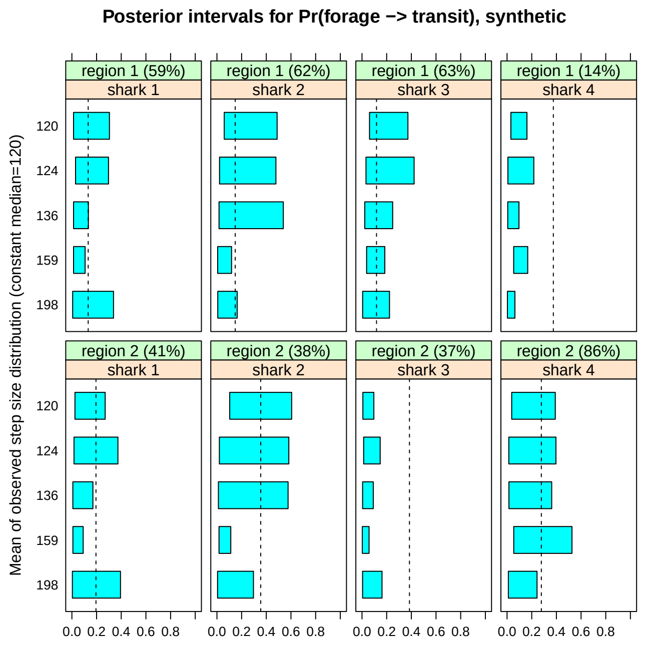


Figure 8.5: Accuracy of behavior classification (λ_t) at observed locations $\boldsymbol{z}_t$ of synthetic shark trajectories, based on simulated interpolation to constant-length intervals Υ_c and modeled behavior λ_c.

out observations, which makes modeling the missing steps more difficult. The regular observations chosen to interpolate based on are effectively chosen from the set of 100 steps by the log-normal random sample, and with smaller samples we expect higher variance for, say, estimation of sample statistics (e.g., sample mean). Thus, even though the accuracies are generally stable even at irregular interpolation, presumably with fewer observations, the classification accuracy presumably depends on which observations are chosen. With low irregularity, the distortion of the true data in the observations is lower, since most of the observations will be used in the interpolation.

The transition probabilities, which we model by Dirichlet distributions, are a key movement parameter we wish to recover. Our model's posterior intervals for the transition probabilities, combined across each of the five repetitions, are shown in Figure 8.6. In this figure, within each cell, the vertical dashed lines show the true empirical transition probabilities (fraction of steps begun in each region where the shark left behavior i, out of the total number of steps in that region in behavior i) for comparison with the intervals. If the vertical line is close to the side of the box, it means the true foraging probability is likely either 0 or 1, which generally means the shark entered the region only a few times in the true data. Within each cell, the posterior intervals are represented by the blue rectangles, stacked in order of increasing irregularity of the simulated time step Δ_t distribution. The highest interval in each cell is at an irregularity of $\sigma = 0$, namely the particle filter learned on the true data. If a vertical line is missing, it means the shark did not enter that region in the true data. In this case, a posterior interval may still be estimated if a shark was simulated to enter the region at least once by at least one particle simulation; disagreements in this aspect are not unusual since the PF trajectories do not perfectly follow the observed ones.

Based on a linear combination of the transition probabilities, we can estimate the density of the overall foraging probability in each region, which is a ecological description of the shark's overall behavior. These posterior interval estimates of the foraging probabilities, which are not a direct input into our



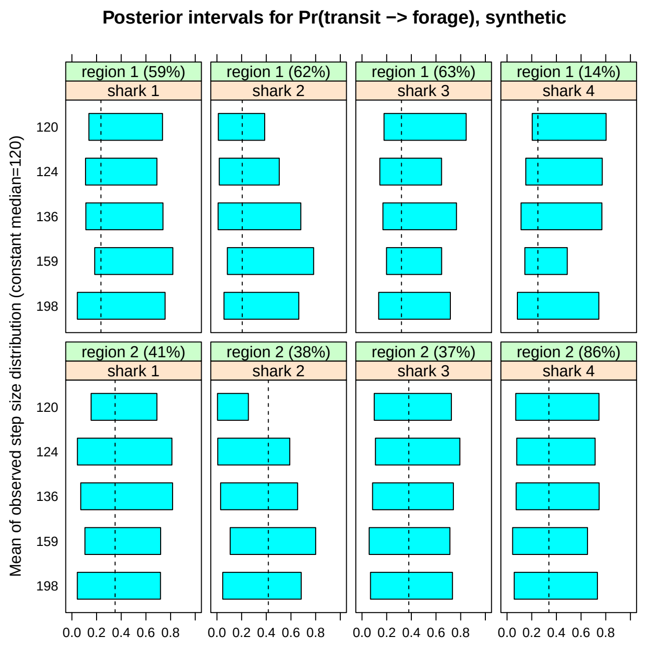


Figure 8.6: 95% posterior intervals for switching probabilities between behaviors, based on irregular interpolation of regular steps to irregular observations. Vertical dashed lines indicate empirical probabilities from the true data, if the shark entered that region.

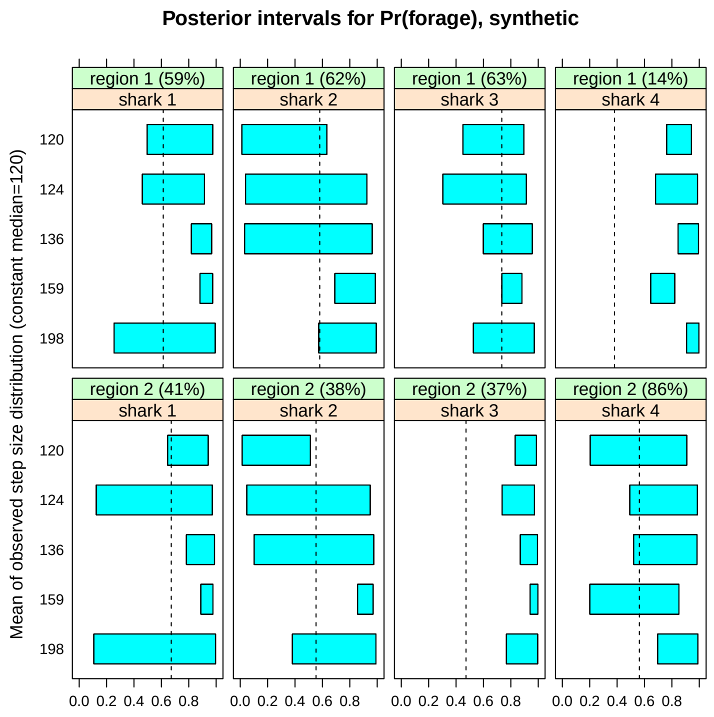


Figure 8.7: 95% posterior intervals of foraging probabilities from simulated data for each shark and region combination, at various levels of irregularity. The labels indicate the percentage of true regular steps that the shark spent in each region. The vertical dashed line in each panel shows the true foraging probability (percentage of regular steps spent foraging) $p_0^{(r)}$ in each region $r \in \{1, 2\}$.

simulation of synthetic trajectories, are shown in Figure 8.7.

8.4 Inter-shark behavioral influence

In Section 6.4, we proposed a spatial-temporal measure of behavioral association between sharks. The behavior association is modeled separately for each shark and behavior but does not depend on the regions. The interaction setup measures whether a shark is more likely to forage or transit depending on the relative frequency of other sharks' behaviors in a specified Euclidean

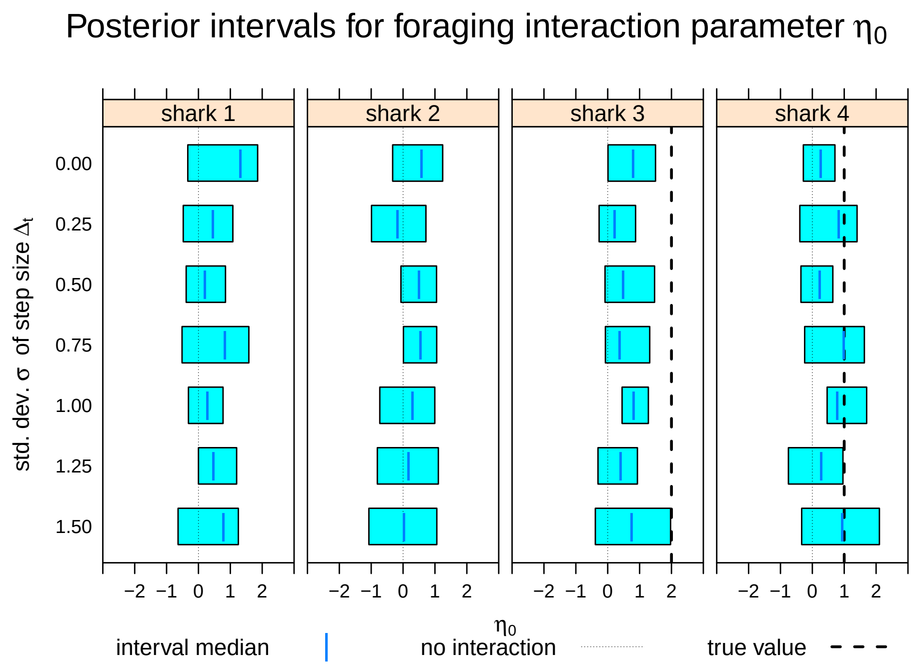


Figure 8.8: 95% posterior intervals for behavior interaction parameter η_0, at each level of irregularity of step size, with irregularity again increasing in descending order.

spatial and temporal neighborhood $N(\cdot)$. We need only model one parameter, η_0, for the influence of foraging, where we set the prior mean $\eta_{k=0,0} = 0$. A mean value $\eta_0 > 0$ means that foraging behavior $\lambda_c = 0$ tends to spatially cluster, while a negative value means foraging sharks tend to repel each other. The proportion $\pi_{c,s,0}$ is the fraction of the shark's neighbors (for a neighborhood above a minimum number of members) that are foraging. The parameters η_0 and $\pi_{k=0}$ are then used in a log-normal distribution to draw an intensity multiplier $\rho_{k=0} \sim \mathcal{Lognormal}(\eta_{k=0}\pi_{k=0},\ \tau_{k=0})$, which is used to weight the behavior-conditional particle weights $w_{c|0}^{(n)}$ for resampling. As with all parameters, we hope the resampling weights will help our algorithm learn the correct values by favoring particles whose parameters best predict the observations.

Figure 8.8 shows the 95% posterior intervals of foraging interaction intensity η_0, at each level of step size Δ_t irregularity. If there is no interaction effect for foraging behavior, then $\eta_{k=0} \approx 0$, and thus the mean parameter of the multiplier $\rho_{k=0}$, $\mu = \eta_{k=0}\pi_{k=0} = 0$, for any fraction $0 \leq \pi_{k=0} \leq 1$. This means that, for instance, having a higher proportion of neighbors be foraging does not impact the shark's foraging probability. Thus, to test for significant interactions in this way, we need $\eta_{k=0}$ to not be significantly different from 0, meaning the posterior interval does not contain 0. Generally if there is significant interaction, then $\eta_{k=0} > 0$ since we expect foraging behavior to cluster rather than repel each other.

Sharks 1 and 2 (black and red in Figure 8.1) were generated independently and then the datasets combined, so we should not learn any significant interaction effects. Sharks 3 and 4 were simulated to have positive foraging interaction effects, namely that they are more likely to forage if other nearby sharks (including 1 and 2) foraging. Indeed the results of Figure 8.8 generally confirm these results. First, note that none of the intervals for any sharks are significantly to the left of $\eta = 0$, which would indicate repelling rather than congregation of foraging.

CHAPTER 9

OBSERVED SHARK DATA SIMULATION RESULTS

9.1 Parameter setup

The results in Section 8 indicated that our interpolation EKF PF algorithm can decently recover movement and behavioral parameters of an irregularly-observed trajectory based on a true, unobserved, regular-step CRW. The model is able to classify behavior type at the points we do observe, with reasonable average accuracy across levels of irregularity of time step length. We assume that the actual shark data can similarly be modeled as irregularly-timed observations from an underlying unobserved CRW at regular time steps. To assess the accuracy of our model, we will use the values from the spline interpolation to the same constant step interval Δ_{Υ}—for behavior transition and foraging probabilities—as comparison. However, we will use these only as a guide without assuming them to be the true values, as we have with the synthetic trajectories in Figure 8.6 and others.

The subset of the raw data we will use covers sharks 12, 13, 14, 15, and 16 for the period 3:40 AM to 5:40 AM on May 2, 2009, a span of 50 hours, slightly over two days. We estimate a total of $C = 2,000$ steps of constant length $\Delta_{\Upsilon} = 90$ seconds to cover this period. Figure 9.1 shows, using the subset

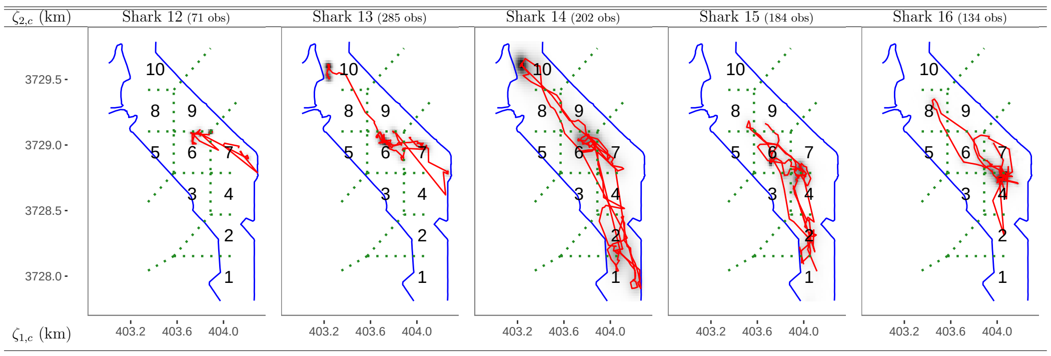


Figure 9.1: Actual trajectories of sharks 12–16 used for simulation. The red lines show Euclidean paths of the observations $\boldsymbol{z}_t$ for each shark. The gray and black areas show the spatial densities of the constant-interval locations $\{\{\boldsymbol{\zeta}_c^{(n)}\}_{c=1}^{C}\}_{n=1}^{N}$ collected across the $N = 200$ particles and for 10 repetitions of the simulation. Each panel title indicates the number of observed locations T are used as inputs for the particle filter.

of the actual data, the five trajectories using Euclidean paths connecting the observed locations $\boldsymbol{z}_t$ for each shark. As we can see, the five sharks covered most of the tidal basin area over the subset of time examined; good regional coverage is essential to have more complete analysis of the regional behavior patterns.

The following prior parameters were used for all sharks and regions, based on experiments tuning the parameters:

- Variance (σ^2) of log-speed $\ln(v_c)$:
  - Foraging: $\sigma_c^2 \mid (\lambda_c = 0) \sim \mathcal{G}^{-1}(a_0 = 8,\ b_0 = 2)$
  - Transiting: $\sigma_c^2 \mid (\lambda_c = 1) \sim \mathcal{G}^{-1}(a_0 = 8,\ b_0 = 1)$
- Variance (τ^2) of turn angle θ_c:
  - Foraging: $\tau_c^2 \mid (\lambda_c = 0) \sim \mathcal{G}^{-1}(a_0 = 10,\ b_0 = 0.75)$
  - Transiting: $\tau_c^2 \mid (\lambda_c = 1) \sim \mathcal{G}^{-1}(a_0 = 10,\ b_0 = 0.25)$

- Transformed speed $\ln(v_c)$:
  - Foraging: $\ln(v_c) \mid (\lambda_c = 0) \sim \mathcal{N}(\alpha_c = -4.25, \kappa_{0,0} \times \sigma^2)$
  - Transiting: $\ln(v_c) \mid (\lambda_c = 1) \sim \mathcal{N}(\alpha_c = -2, \kappa_{1,0} \times \sigma^2)$
- Turn angle θ_c:
  - Foraging: $\theta_c \mid (\lambda_c = 0) \sim \mathcal{WN}(\beta_c = 0, \kappa_{3,0} \times \tau^2)$
  - Transiting: $\theta_c \mid (\lambda_c = 1) \sim \mathcal{WN}(\beta_c = 0, \kappa_{4,0} \times \tau^2)$
- NIG degrees of freedom (above) $\kappa = 20$.
- Probability transition probabilities, for regions $r = 1, \ldots, R = 10$:
  - From foraging: $(p_{0\to0},\ p_{0\to1}) \sim \mathcal{Dir}(\alpha_{0\to0} = 25,\ \alpha_{0\to1} = 3)$
  - From transiting: $(p_{1\to0},\ p_{1\to1}) \sim \mathcal{Dir}(\alpha_{1\to0} = 2,\ \alpha_{1\to1} = 7)$

  The parameters are the same for each region to see if we can learn the differences between the regions. This corresponds to a average transition probabilities of $P \approx \begin{bmatrix} 0.89 & 0.11 \\ 0.22 & 0.78 \end{bmatrix}$; the variance depends on the absolute, not relative, magnitude of the parameters.
- Location error covariance matrices: Let $\mathbf{\Lambda} = \begin{bmatrix} 1 & -0.3 \\ -0.3 & 1.25 \end{bmatrix}$.
  - Error covariance between states for $\boldsymbol{\zeta}_c \mid \mathbf{x}_{c-1}$ (not dependent on previous behavior λ_{c-1}):
    * $\Sigma_{\zeta_0} \sim \mathcal{W}_2^{-1}(\mathbf{\Lambda}_{\zeta_0},\ \eta_{\zeta_0})$ where degrees of freedom $\eta_{\zeta_0} = 20$ and scale matrix $\mathbf{\Lambda}_{\zeta_0} = 0.5\eta_{\zeta_0}\mathbf{\Lambda} = \begin{bmatrix} 10 & -3 \\ -3 & 12.5 \end{bmatrix}$.
  - Error covariance between for observations $\mathbf{y}_c \mid (\mathbf{x}_c, \lambda_c)$: $\Sigma_{z_0} \mid \lambda_c \sim \mathcal{W}_2^{-1}(\mathbf{\Lambda}_{z_0,\lambda_0},\ \eta_{z_0,\lambda_0})$, where
    * Degrees of freedom $\eta_{z_0,\lambda_0=0} = \eta_{z_0,\lambda_0=1} = 20$

  * Scale matrices for foraging $\mathbf{\Lambda}_{z_0,\lambda_0=0} = 5\eta_{z_0,\lambda_0=0}\mathbf{\Lambda} = \begin{bmatrix} 100 & -30 \\ -30 & 125 \end{bmatrix}$ and transiting $\mathbf{\Lambda}_{z_0,\lambda_0=1} = 15\eta_{z_0,\lambda_0=1}\mathbf{\Lambda} = \begin{bmatrix} 300 & -90 \\ -90 & 375 \end{bmatrix}$.

- Maximum intervals to simulate without observations: $c_{\text{thresh}} = 10$.
- Interaction parameters (see Section 6.4):
  - For a given behavior $\lambda \in \{0,1\}$ and fraction π_λ of a shark's neighbors having behavior λ, the spatial interaction effect is $\rho_\lambda \sim \mathcal{Lognormal}(\pi_\lambda\eta_\lambda, \tau_\lambda)$, where, for both $\lambda \in \{0,1\}$,
  - Precision (assumed known) $\tau_\lambda = 2$, and
  - Spatial interaction strength $\eta_\lambda \sim \mathcal{N}(\eta_{\lambda,0} = 0,\ \tau_{\lambda,0})$
- Behavior rescaling weights (see Section 5.6): $a_0 = 1, a_1 = 2.5$.

9.2 Discussion of results

We will use the spline interpolation as a basis for evaluating the accuracy of the joint particle filter, since we do not know the underlying regular-step trajectories underlying the observed data. For the raw data, we perform ten repetitions of the joint PF with the same prior parameters. For each parameter of interest, the distributions estimated are combined across the N particles and ten repetitions for each shark.

First, we we will examine the distributions of the log-velocities $\ln(v_c)$ and turn angle θ_c, the main movement variables of interest, conditional on each behavior λ_c. These are estimated by making 100 draws from each particle n's distribution, and pooling the results to estimate the densities. As Figure 9.2 shows, the density of the log-velocities are generally well-separated, and thus identifiable, with foraging having lower speed than transiting. However, the turn angle (θ_c) distributions for the two behaviors are difficult to separate because they are both centered around 0 (following the CRW model assumption) but the variances were hypothesized as differing. Even if the true variances of turn angles for the two behaviors differed, it is difficult to accurately model the two distributions because they overlap significantly. This is probably true even when running the EKF algorithm on synthetic data where the true distributions are generated as having different variances.

Figure 9.3 shows 95% posterior intervals for the stationary foraging probabilities (see Section 6.2) $p_0^{(r)}$ by for each region r. If a shark was simulated at any time in the 10 repetitions to enter a region, a posterior interval is estimated; therefore, for instance, shark 12 has intervals estimated for only regions 6,7, and 9, since it did not travel very far in this time window. The black dots show the empirical foraging probability for that region based on the spline-interpolated data subset, not the full raw data; a point is only plotted if the spline interpolation has the shark entering the region at all. For each shark, the widths of the rectangles are the posterior interval, and the rectangles' heights as proportion of the total vertical space in the regional box are the percentages the shark's simulated movements at constant-length intervals c spent in that region. One main reason for interpolating to constant-length time intervals is to be able to calculate rates and probabilities (e.g., foraging, regional occupancy), since simulated variables are now equally-weighted in terms of time. Thus, a shark that is simulated to have spent all of its time in interval 1 would have a rectangle take up 100% of the vertical space for that regional box, with all other regions empty. The total rectangle heights should thus be equal, totaling 100%, for each shark.

The rectangle heights are scaled by the proportion of time the shark spent in each region because more time means that region's estimate of foraging probability or any other quantity is likely to be more reliable. With total number of time intervals $C = 2,000$, the actual number of constant time steps c simulated for each shark can be less than C because c_{thresh}, the maximum number of intervals c allowed between pairs of consecutive observed times H_t and H_{t+1} in order to simulate movement at times Υ_c between them, may be exceeded. The problem lessens as C, the number of time intervals to sim-

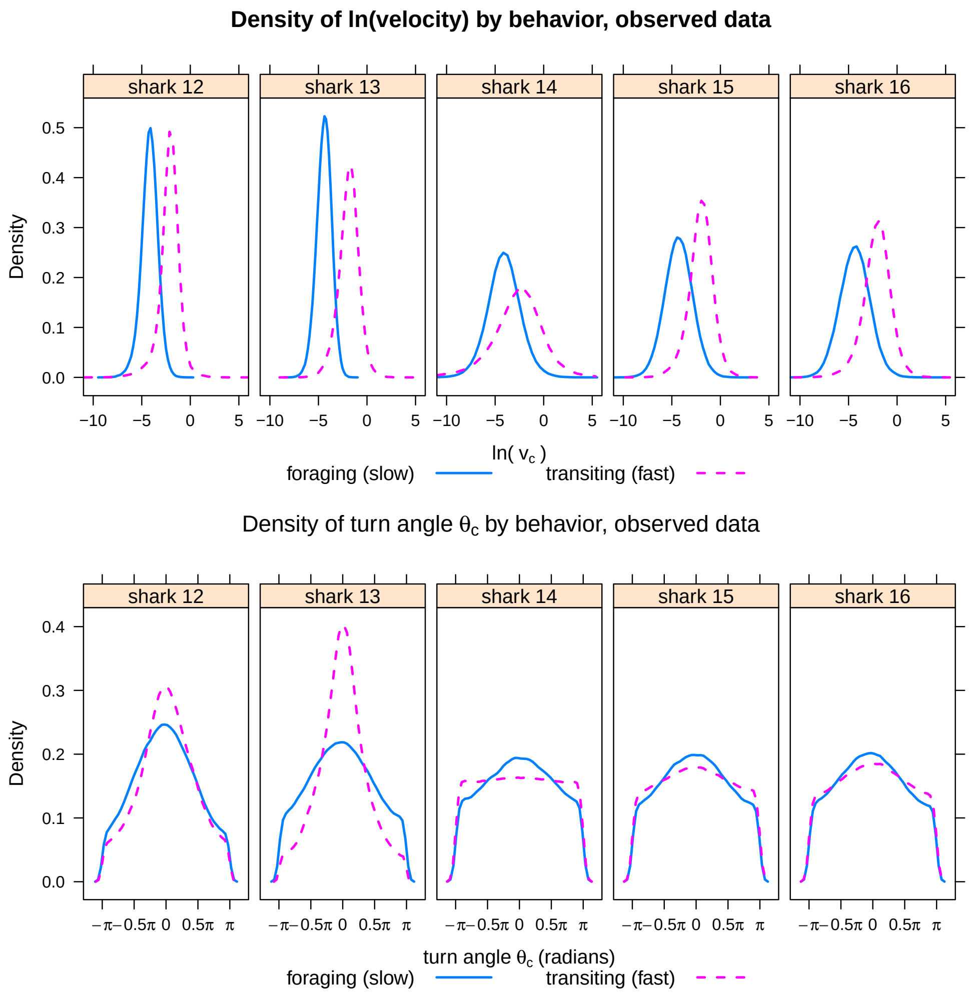


Figure 9.2: Estimation of densities of log-velocity $\ln(v_c)$ and turn angle $\theta_c \in [-\pi, \pi]$ for each shark, for inferred foraging and transiting behavior, based on the subset of the observed data.

ulate, increases, but a region that the shark spent, say, 1% of its time in, only represents at most 20 constant intervals c of movement if $C = 2,000$. Twenty simulated movements is not enough to reliably estimate probabilities for foraging or transitions between behaviors.

The black dots in each region represent the regional foraging probabilities estimated by the Bézier spline. As discussed, we do not necessarily believe these represent true values, but are just an approximation. A black dot is shown if the spline estimates ever have a shark entering that region. These are probably not reliable estimates for comparison if the shark spent only a few constant intervals c in that region, in which case they often fall outside of our posterior intervals. If there is an interval estimated but no black dot, it means the shark was simulated (probably only rarely) as entering a region that it did not enter in the spline interpolation. Note that the posterior intervals are skewed to the right, so overall foraging is more likely than transiting at any point. This matches the biological intuition that animals should spend the vast majority of their time (i.e., steps) moving slowly or foraging, to conserve their energy.

Figure 9.4 shows 95% posterior intervals for the regional transition probabilities $\Pr(\text{forage} \to \text{transit} \mid r) = p^{(r)}_{0\to 1}$ and $\Pr(\text{transit} \to \text{forage} \mid r) = p^{(r)}_{1\to 0}$, with black dots showing the empirical transition probabilities estimated from the spline interpolation. Biologically, we assume that sharks tend to extended longer periods of time foraging, punctuated by occasional movements (transiting) between foraging areas, and hence the per-step probability of leaving foraging behavior should be small. The corresponding probability of leaving the transiting behavior should be larger, since the shark can spend only a short amount of time transiting, or moving more quickly. This intuition is confirmed in the the figure, where the posterior intervals for $\Pr(\text{forage} \to \text{transit} \mid r)$ tend to be narrow and close to 0, while for $\Pr(\text{transit} \to \text{forage} \mid r)$, they tend to be wider.

As a word of caution against relying too strongly on the estimates from the spline interpolation, we note that the for $\Pr(\text{transit} \to \text{forage} \mid r)$, the second

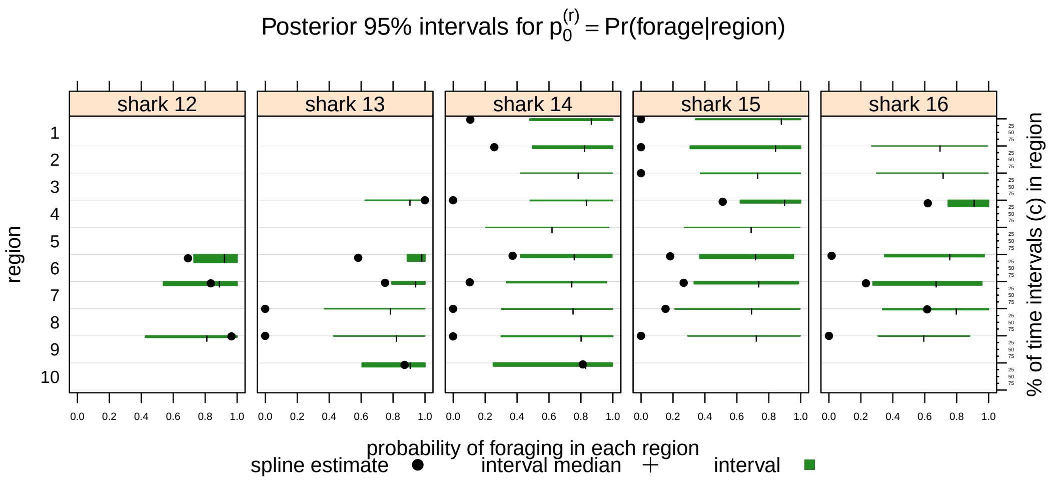


Figure 9.3: 95% posterior intervals for the foraging probability for each shark for each region, with vertical line segments indicating the median probability estimate. The black dots indicate the empirical foraging probability for the data subset interpolated to regular steps of $\Delta_\Upsilon = 90$ seconds. The columns of percentages show the percentage of regular steps that the shark spent in each region, in the data subset. For regions where this 'empirical regional share' was low, analysis should be limited.

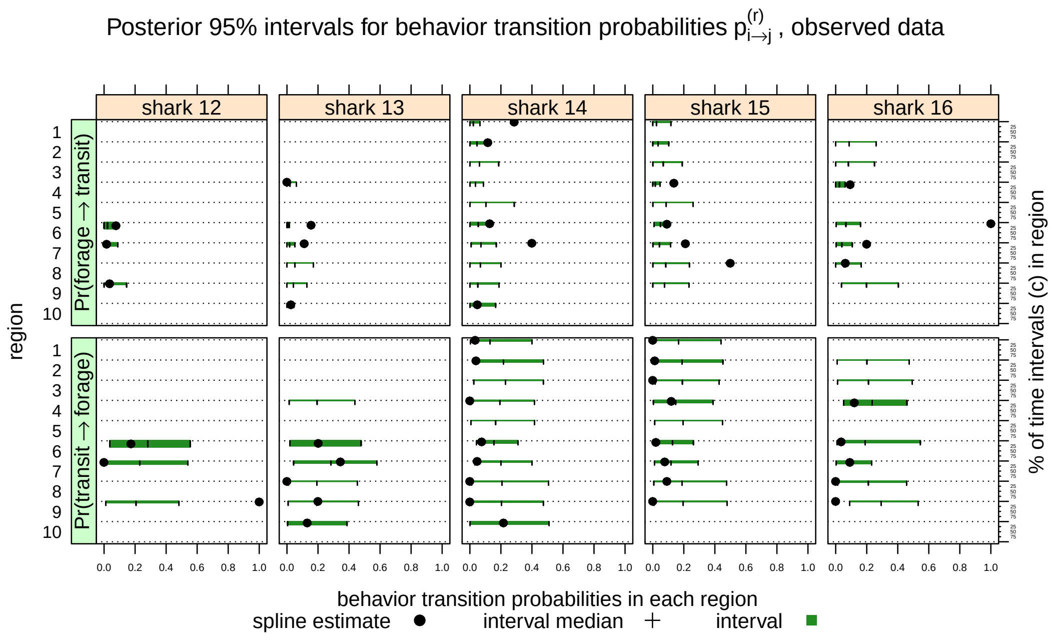


Figure 9.4: 95% posterior interval estimates of transition probabilities between behaviors in each region, with vertical line segments indicating the median probability estimate. The black dots show empirical estimates of these probabilities from the spline interpolation; however, as discussed, these estimates may result as an artifact of the spline algorithm rather than indicating something about the true biological behavior.

plot, many of the empirical estimates indicated by the black dots are also close to zero, while the biological intution is that these probabilities should be higher. As Figure 3.1 noted, in the case when the constant time step length Δ_Υ of the Bézier spline interpolation is somewhat smaller than the time intervals between the observations used as control points, the spline results in longer sequences of left or right turns. Similarly, if we use the average speed over these regular intervals, calculated as the ratio of the distance to the time interval Δ_Υ, as an indicator of the behavior type, as in Figure 2.6, depending on the sinuosity of the estimated spline, the spline may give longer unbroken sequences of each behavior. These sequences have more to do with the spline algorithm than with the actual biological behavior. Thus, it may make more sense to use the splines to estimate the overall regional relative prevalence of foraging vs transiting behaviors, as in the foraging probability (Figure 9.3), rather than the transition probabilities between the behaviors. The probabilities $\Pr(\text{transit} \to \text{forage} \mid r)$ in particular may not be estimated well by the spline paradigm.

Figure 9.5 shows the 95% posterior estimates of η_0, the mean parameter of the normal distribution for η, the intensity of the interaction, which when mutliplied by $\pi_{c,s,0}$ (the fraction of neighbors that are foraging), serves as the mean parameter of the log-normal distribution of ρ_0, the multiplier for the interaction effect. As discussed in Section 6.4, if $\eta > 0$, which is more likely when its mean $\eta_0 > 0$, a higher proportion of foraging neighbors ($\pi_{c,s,0}$ closer to 1) is associated with the shark being more likely to forage itself; this indicates an effect on the shark's behavior above and beyond the regional parameters. If there is a neighbor effect on behavior, the 95% posterior intervals of η_0 should exclude zero, and we expect them to be positive rather than negative. Based on the data subset, none of the intervals are firmly negative, and two (for sharks 12 and 13) are positive. This provides at least some evidence that there may be some inter-shark behavioral influence, at least of the form that we have posited. An experiment on more data would be needed for further analysis.

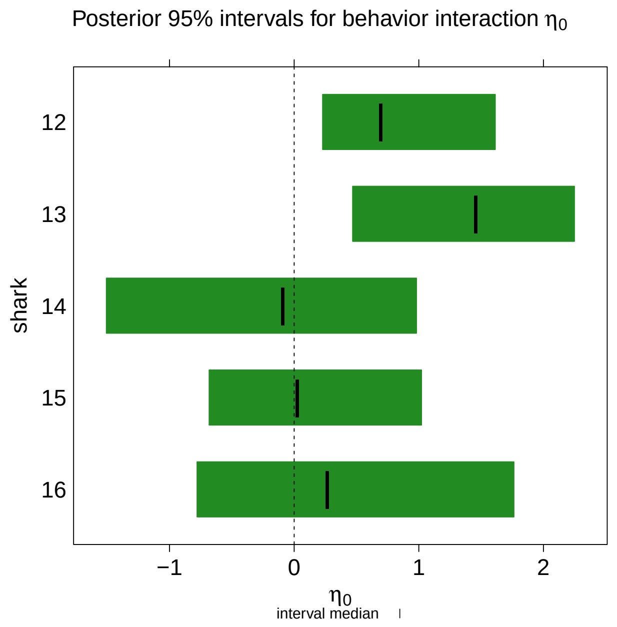


Figure 9.5: 95% posterior interval estimates of η_0, the mean parameter of the behavior interaction parameter, with vertical line segments indicating the median estimate; if $\eta_0 > 0$, more foraging neighbors is associated with the shark being more likely to forage.

CHAPTER 10

FUTURE AREAS OF RESEARCH

In the course of learning about animal telemetry models and particle filtering, we considered several potential extensions or modifications of our model based on ideas in the literature. Unfortunately, due to time constraints, we were not able to explore all these extensions in detail. Several that we have mentioned include

- Considering a time-dependent model for behavioral transitions (see Section 6.3), where the animal is more likely to leave the current behavioral model the longer it has remained in that mode, uninterrupted. As mentioned, such as model may be more realistic than the Markovian assumption, because it may better reflect the biological model of animals foraging in an area until its resources are depleted, then leaving (i.e., changing to transiting); such a model is considered in Van Moorter et al. ([54]).
- Considering a continuous-time model, such as the Ornstein-Uhlenbeck model (see Johnson [21]) rather than discrete-time, in which we interpolate the irregular observations to a sequence of regular-length steps of a short time span Δ_Υ.

- Learning regional behavioral parameters, such as transition probabilities, where the region partition is learned sequentially and is not fixed, as in McClintock et al. ([35]).

- Studying in more depth the effect particle smoothing has on filter parameter estimates.

- Using a Bayesian model fit measure such as the Bayes Factor (see Kass and Raftery [26] for a summary of Jeffreys' work) to validate the model fit to each shark. The Bayes Factor is a measure that compares two competing models—for instance a simpler and more complex model, or competing sets of model parameter values—for a dataset by evaluating the data likelihood conditional on each of the models. For instance, let $\boldsymbol{\theta}^{(i)}$ be the movement parameters learned from the observed trajectory $\mathbf{y}^{(i)}$ of shark i, and let ℓ denote the likelihood of the data. To validate our particle filter results, we may want to verify that for various pairs of sharks $(i, j), i \neq j$, that $\ell\left(\mathbf{y}^{(i)} \mid \boldsymbol{\theta}^{(i)}\right) > \ell\left(\mathbf{y}^{(i)} \mid \boldsymbol{\theta}^{(j)}\right)$ and vice versa. The ratio of these likelihoods, when combined with optional priors on the parameters $\boldsymbol{\theta}^{(i)}$ vs $\boldsymbol{\theta}^{(j)}$, constitutes the Bayes Factor. That is, that the parameters learned using shark i's trajectory provide a better model for its trajectory than do those learned from shark j's trajectory; in other words, that the parameters actually represent something unique to that shark, or at least that they provide the best fit.
  Carvalho et al. ([10], 6–7) describes use of the Bayes Factor to monitor the performance of the PF while it is running, but what we would like is a method to compare the fit of pairs of sets of parameter values after the PF has run. This concern, at least how we have described it, does not appear to figure prominently in the literature.

- Modifying the shark EKF setup to use a circular distribution for turn angle, such as the von Mises distribution, rather than a transformation to normality. As noted earlier, transformations to normality to allow us to use the CDLM and EKF formulations, which rely on the Delta Method.

- Using a different formulation such as the Unscented Kalman filter (UKF, see Särkkä [45], page 81–92), which can model particle distributions directly without specification of the nonlinear function as in the EKF. The UKF can outperform the EKF on nonlinear problems because it models the second moment of the nonlinear function, as opposed to the EKF, which is a first-order approximation.

- The choice to model the sharks' velocity with the natural log transformation was motivated by the spline interplolation (Figure 2.6), which showed the log-velocities as being roughly normally-distributed and usually bi-modal, indicating two potential behaviors with different speeds. When we simulate $\ln(v_c) \sim \mathcal{N}(\cdot)$ and then apply the exponential transformation to obtain the speeds v_c to estimate the next location, the resulting speeds have a log-normal distribution and thus are skewed right. Say the time gap to the next observation is $\Delta_\Upsilon = 500$, and the shark begins from locations centered at $\boldsymbol{\mu}_\zeta = \begin{bmatrix} 70 & 50 \end{bmatrix}$. Say we have $N = 10,000$ particles, and that the observed locations are $\boldsymbol{\zeta}_c^{(n)} \sim \mathcal{N}_2(\boldsymbol{\mu}_\zeta, \Sigma_c)$ with uncertainty covariance $\Sigma_c = \begin{bmatrix} 4 & -1 \\ -1 & 3 \end{bmatrix}$. Let the bearing be $\psi_c \sim \mathcal{WN}(2, 0.5)$ (based on the standard deviation of the turn θ_c) and the log-speed $\ln(v_c) \sim \mathcal{N}(-2, 0.5)$. Based on these movements, the prediction of the next location is $\hat{\boldsymbol{\zeta}}_{c+1}^{(n)} = \boldsymbol{\zeta}_c^{(n)} + v_c^{(n)} \Delta_\Upsilon \begin{bmatrix} \cos(\psi_c^{(n)}) & \sin(\psi_c^{(n)}) \end{bmatrix}$. According to the EKF formulation, the predictions (like all variables) should have the multivariate normal distribution

$$\boldsymbol{\zeta}_{c+1}^{(n)} \sim \mathcal{N}_2\left(\hat{\boldsymbol{\zeta}}_{c+1}^{(n)}, \mathbf{F}_\mathbf{x}\left(\mathbf{x}_c^{(n)}\right) \Sigma_c \mathbf{F}_\mathbf{x}\left(\mathbf{x}_c^{(n)}\right)^T + \mathbf{Q}_{c+1}\right)$$

However, if we plot the empirical distribution of the locations $\{\boldsymbol{\zeta}_{c+1}^{(n)}\}_{n=1}^N$ (Figure 10.1), we see the locations don't have an elliptical shape as would be typical of a multivariate normal, but rather are skewed in the direc-

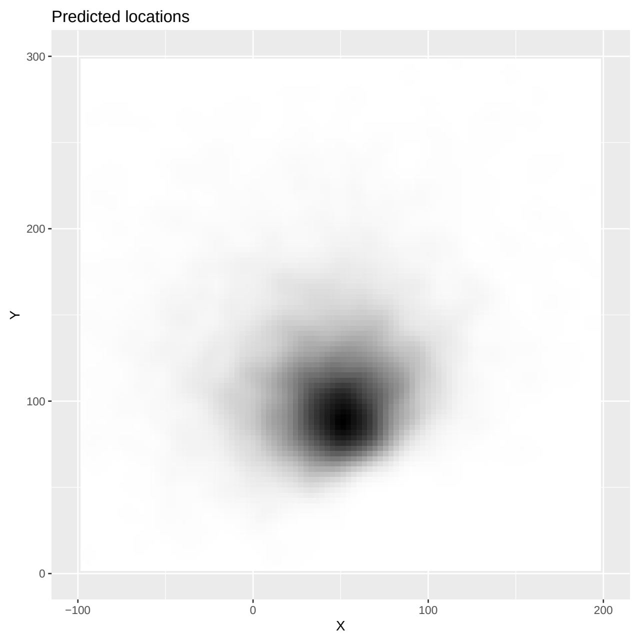


Figure 10.1: Empirical distribution of toy predicted locations.

tion of the bearing $\hat{\psi}_c = 2$, due to the log-normal distribution of the speeds. Assuming the choice to model the log-speed (as opposed to the raw speeds) by a normal distribution, the multivariate skew-normal distribution (introduced by [2], see Figure 10.2 for an example of a contour plot) may be more accurate for modeling locations. This distribution has been used for Kalman filtering (see, for instance [38]), but its distribution is significantly more complex than the multivariate normal.

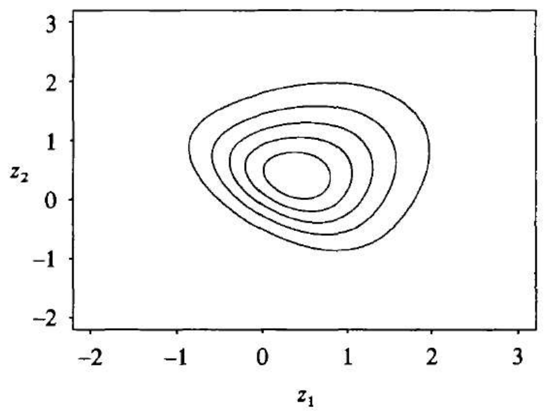


Figure 10.2: Example of 2-D contour plot of a multivariate skew normal distribution (source: [2]).

REFERENCES

[1] Anderson, T. W. and Leo A. Goodman. Statistical inference about Markov chains. *The Annals of Mathematical Statistics*, 1957.

[2] Azzalini, Adelchi and A. Dalla Valle. The multivariate skew-normal distribution. *Biometrika*, 1996.

[3] Baddeley, Adrian, Rubak, Ege, and Rolf Turner. Spatial point patterns: Methodology and applications with R. `http://www.crcpress.com/Spatial-Point-Patterns-Methodology-and-Applications-with-R/Baddeley-Rubak-Turner/9781482210200`, 2015. R package version 1.53-2.

[4] Baker, James Edward. Reducing bias and inefficiency in the selection algorithm. *Proceedings of the Second International Conference on Genetic Algorithms on Genetic Algorithms and Their Application*, 1987.

[5] Benhamou, Simon. Detecting an orientation component in animal paths when the preferred direction is individual-dependent. *Ecology*, 2006.

[6] Beyer, Hawthorne L., Morales, Juan M., Murray, Dennis, and Fortin, Marie-Josée. The effectiveness of Bayesian state-space models for estimating behavioural states from movement paths. *Methods in Ecology and Evolution*, 2013.

[7] Brownlee, K. A. *Statistical theory and methodology in science and engineering.* Wiley, 1965.

[8] Byers, Simon D. and Adrian E. Raftery. Bayesian estimation and segmentation of spatial point processes using Voronoi tilings. Technical report, University of Washington, 1997.

[9] Caeiro, Frederico and Ayana Mateus. randtests: Testing randomness in R. `https://CRAN.R-project.org/package=randtests`, 2014. R package version 1.0.

[10] Carvalho, Carlos M., Johannes, Michael S., Lopes, Hedibert F., and Nicholas G. Polson. Particle learning and smoothing. *Statistical Science*, 2010.

[11] Challa, S. and Bergman, N. Target tracking incorporating flight envelope information. *Proceedings of the third International Conference on Information Fusion*, 2, 2000.

[12] Codling, Edward A., Plank, Michael J., and Simon Benhamou. Random walk models in biology. *Journal of the Royal Society Interface*, 2008.

[13] Crisan, Dan and Joaquín Míguez. Particle-kernel estimation of the filter density in state-space models. *Bernoulli*, 2014.

[14] Doucet, Arnaud. On sequential simulation-based methods for Bayesian filtering. Technical report, University of Cambridge, 1998.

[15] Durbin, James and Siem Jan Koopman. *Time series analysis by state space methods.* Oxford University Press, 2001.

[16] Espinoza, Mario, Farrugia, Thomas J., and Christopher G. Lowe. Habitat use, movements and site fidelity of the gray smooth-hound shark in a newly restored Southern California estuary. *Journal of Experimental Marine Biology and Ecology*, 2011.

[17] Farrugia, Thomas J., Espinoza, Mario, and Christopher G. Lowe. The fish community of a newly restored Southern California estuary: ecological

perspective 3 years after restoration. *Environmental Biology of Fishes*, 2014.

[18] Grisetti, Giorgio, Stachniss, Cyrill, and Wolfram Burgard. Improving grid-based SLAM with Rao-Blackwellized particle filters by adaptive proposals and selective resampling. *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, 2005.

[19] Gurarie, Eliezer. *Models and analysis of animal movements: from individual tracks to mass dispersal.* PhD thesis, University of Washington, 2008.

[20] Heikkinen, Juha and Elja Arjas. Non-parametric Bayesian estimation of a spatial Poisson intensity. *Scandinavian Journal of Statistics*, 1998.

[21] Johnson, Devin D., London, Joshua M., Lea, Mary-Anne, and John W. Durban. Continuous-time correlated random walk model for animal telemetry data. *Ecology*, 2008.

[22] Jonsen, Ian D., Flemming, Joanna Mills, and Ransom A. Myers. Robust state-space modeling of animal movement data. *Ecology*, 2005.

[23] Jonsen, Ian D., Myers, Ransom A., and Joanna Mills Flemming. Meta-analysis of animal movement using state-space models. *Ecology*, 2003.

[24] Jonsen, Ian D., Myers, Ransom A., and Michael C. James. Identifying leatherback turtle foraging behaviour from satellite telemtry using a switching state-space model. *Marine Ecology Progress Series*, 2007.

[25] Kamermans, Mike. A primer on Bézier curves. `https://pomax.github.io/bezierinfo/#control`, 2011.

[26] Kass, Robert E. and Adrian E. Raftery. Bayes factors. *JASA*, 1995.

[27] Khan, Zia, Balch, Tucker, and Frank Dellaert. MCMC-based particle filtering for tracking a variable number of interacting targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2005.

[28] Kimmerer, Wim, Avent, Sean R., and Steven M. Bollens. Variability in length-weight relationships used to estimate biomass of estuarine fish from survey data. *Transactions of the American Fisheries Society*, 2005.

[29] Kurz, Gerhard, Gilitschenski, Igor, and Uwe D. Hanebeck. Recursive nonlinear filtering for angular data based on circular distributions. American Control Conference, 2013.

[30] Kurz, Gerhard, Gilitschenski, Igor, and Uwe D. Hanebeck. Recursive Bayesian filtering in circular state spaces. *IEEE Aerospace and Electronic Systems Magazine*, 2016.

[31] Lamberti, Roland, Peletin, Yohan, Desbouvries, François, and François Septier. Independent resampling sequential Monte Carlo algorithms. *arXiv*, 2016.

[32] Lin, Ming T., Zhang, Junni L., Cheng, Qiansheng, and Rong Chen. Independent particle filters. *JASA*, 2005.

[33] Liu, Jun S. Metropolized independent sampling with comparisons to rejection sampling and importance sampling. *Statistics and Computing*, 1996.

[34] Marsh, L.M. and R.E. Jones. The form and consequences of random walk movement models. *Journal of Theoretical Biology*, 1988.

[35] McClintock, Brett T., King, Ruth, Thomas, Len, Matthiopoulos, Jason, McConnell, Bernie J., and Juan M. Morales. A general discrete-time modeling framework for animal movement using multistate random walks. *Ecological Monographs*, 2012.

[36] Morales, Juan Manuel, Haydon, Daniel T., Frair, Jacqui, Holsinger, Kent E., and John M. Fryxell. Extracting more out of relocation data: building movement models as mixtures of random walks. *Ecology*, 2004.

[37] Murphy, Kevin P. Conjugate Bayesian analysis of the Gaussian distribution. `https://www.cs.ubc.ca/~murphyk/Papers/bayesGauss.pdf`, 2007.

[38] Naveau, Philippe, Genton, Marc G., and Xilin Shen. A skewed Kalman filter. *Journal of Multivariate Analysis*, 2005.

[39] Olsen, Aaron. bezier: Bézier Curve and Spline Toolkit. `https://CRAN.R-project.org/package=bezier`, 2014. R package version 1.1.

[40] Pitt, Michael K. and Neil Shephard. Filtering via simulation: auxiliary particle filters. *JASA*, 1999.

[41] Reich, Sebastian and Colin Cotter. *Probabilistic forecasting and Bayesian data assimilation.* Cambridge University Press, 2015.

[42] Ristic, Branko, Arulampalam, Sanjeev, and Neil Gordon. *Beyond the Kalman Filter: Particle filters for tracking applications.* Artech House, 2004.

[43] Roever, C.L., Beyer, H.L., Chase, M.J., and R.J. van Aarde. The pitfalls of ignoring behaviour when quantifying habitat selection. *Diversity and Distributions*, 2013.

[44] Särkkä, Simo. Particle filtering, other approximations, and continuous-time models, 2011.

[45] Särkkä, Simo. *Bayesian filtering and smoothing.* Cambridge University Press, 2013.

[46] Skalski, Garrick T. and James F. Gilliam. A diffusion-based theory of organism dispersal in heterogeneous populations. *The American Naturalist*, 161(3), 2003.

[47] Spedicato, Giorgio Alfredo. markovchain: Discrete Time Markov Chains with R. `https://journal.r-project.org/archive/2017/RJ-2017-036/index.html`, 2017. R package version 0.6.9.7.

[48] Thrun, Sebastian. Probabilistic algorithms in robotics. *AI Magazine*, 21(4), 2000.

[49] Traa, Johannes and Paris Smaragdis. A wrapped Kalman filter for azimuthal speaker tracking. *IEEE Signal Processing Letters*, 20(12), 2013.

[50] Tracey, J.A., Zhu, J., and K. Crooks. A set of nonlinear regression models for animal movement in response to a single landscape feature. *Journal of Agricultural, Biological, and Environmental Statistics*, 10(1), 2005.

[51] Tremblay, Yann and others. Interpolation of animal tracking data in a fluid environment. *The Jounral of Experimental Biology*, 209(1), 2006.

[52] Tukey, John W. *Exploratory data analysis.* Addison-Wesley Publishing Company, 1977.

[53] Turchin, Peter. *Quantitative analysis of movement: measuring and modeling population redistribution in animals and plants.* Sinauer Associates, Inc., 1998.

[54] Van Moorter, Bram, Visscher, Darcy, Benhamou, Simon, Börger, Luca, Boyce, Mark S., and Jean-Michel Gaillard. Memory keeps you at home: a mechanistic model for home-range emergence. *Oikos*, 2009.

[55] Zar, Jerrold H. *Biostatistical Analysis.* Prentice-Hall, 1984.

APPENDIX A

Appendix

A.1 `R` package

In addition to the results in this thesis, we will be publishing a statistical software package for `R`, called `animalEKF` that can be publicly downloaded. This package will have the following features:

- Implementation of the 1-D robot EKF and 2-D shark EKF movement model.
- Implementation of functions to simulate synthetic 1-D and 2-D movement data at regular and irregular intervals.
- Interactive `shiny` visualizations to show
  - Visualization of the original raw shark data
  - How 1-D robot EKF works with both one and two-behavior movement
  - How 2-D robot EKF (analogous to the sharks) works with two-behavior movement

Here, we briefly review the animations available in the package.

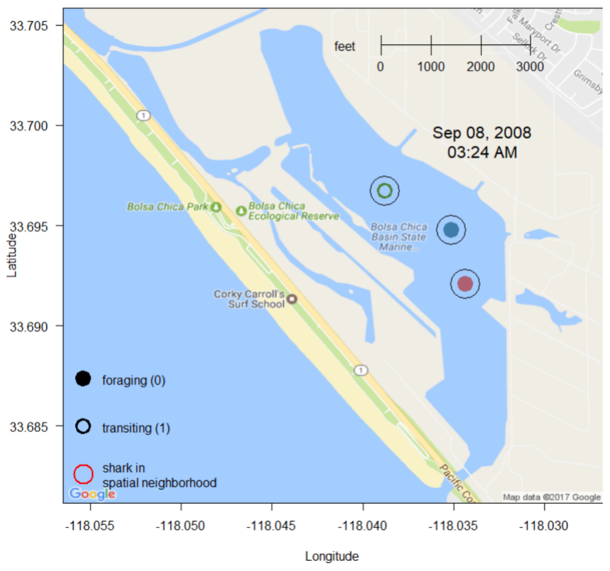


Figure A.1: Screenshot of observed data visualization from function `shark_vis_longlat`.

The function `shark_vis_longlat` shows a sped-up time-lapse animation of the observed shark data, interpolated to constant-length intervals with the Bézier spline (see Section 2.4). Users can select a subset of the observations to view; the default values are set to an 'interesting' portion of the data where there are several sharks observed concurrently. There are options to display a circular spatial neighborhood of a chosen radius around each shark (to mimic the inter-shark influence parameter discussed in Section 6.4), where the outer circle flashes red whenever a shark enters another's spatial neighborhood. A screenshot of this function is shown in Figure A.1.

The function `cdlm_robot` which implements a particle of the 1-D robot with one behavior (see Section 4.5) which aims to estimate the mean α of velocity $v_t \sim \mathcal{N}(\alpha, \sigma)$. The user can choose the following parameters:

- Unknown true velocity mean α and known true variance σ^2 used to simulate a 1-D path
- maximum number of observed steps (T)

- constant time gap Δ_Υ between observations t
- number of particles N
- prior variance on velocity mean
- covariance error matrix $\mathbf{Q}_t$ of $\mathbf{x}_t$ and measurement error variance $\mathbf{R}_t$
- covariance matrix $\mathbf{P}_t$ of the elements of $\mathbf{x}_t$
- option to color features (weights, densities, etc.) separately by particle, or to color each particle gray except for the best ones (by weight) which are in red.

Figure A.2 shows a screenshot of the function graphs, which display the diagnostic results of the particle filter in real time. The graphs show, from left to right and top to bottom,

1. Given each particle n's initial value of $\alpha^{(n)}$ drawn from the common prior, we show the normal density function for velocity $v_t \sim \mathcal{N}(\alpha^{(n)}, \sigma)$ (various colors), compared to the true unknown distribution centered at mean α (black). Eventually the particle distributions should converge to the true distribution around α.

2. At time t, each particle's prediction of the observation $\mathbf{y}_{t+1} = z_{t+2}$, shown by the black dots. The width of each rectangle is the 95% confidence interval of $\hat{\mathbf{y}}_{t+1} \mid \mathbf{x}_t$. Since the only unknown parameter is α, each particle has the same measurement error variance, so the intervals are all the same width.

3. Resampling weights $\{w_t^{(n)}\}_{n=1}^N$, shown by the heights of the sticks.

4. Resampled predictions $\{\tilde{\mathbf{y}}_{t+1}^{(n)}\}$. The resampling is shown in slow-motion to illustrate the resampling process. Each weight in the graph to the left flash as they are resampled, and predictions appear.

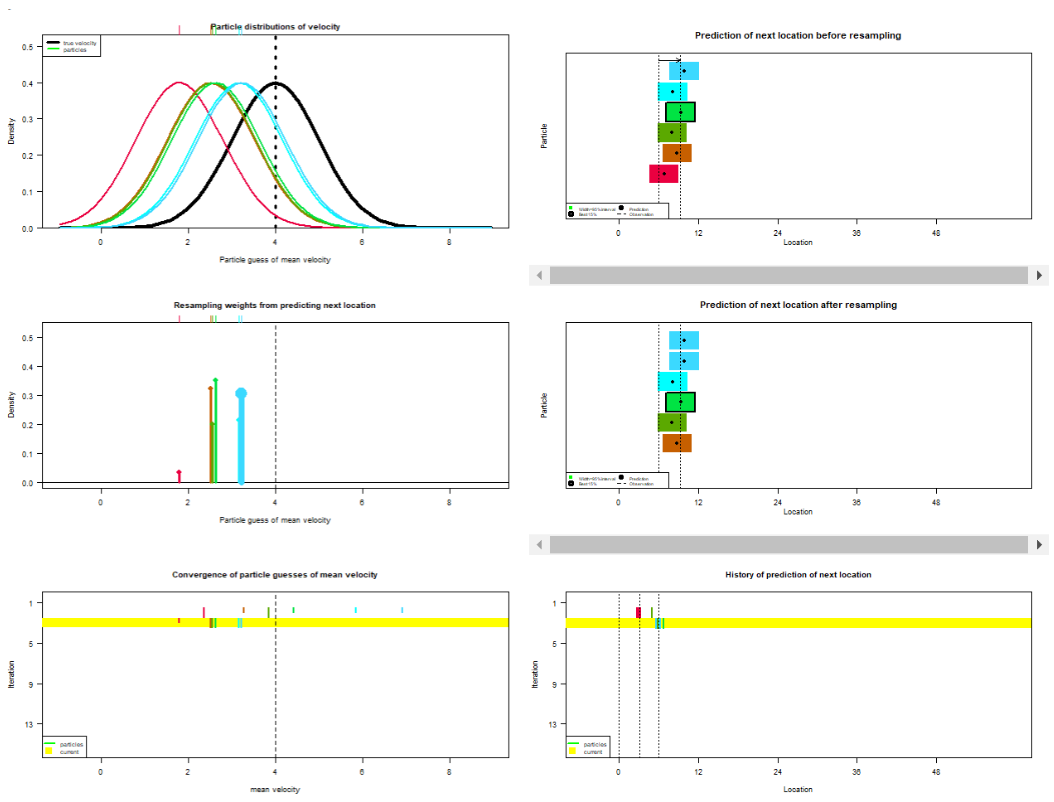

Particle distributions of velocity
true velocity
particles
Density
Particle guess of mean velocity
Prediction of next location before resampling
Particle
Location
Resampling weights from predicting next location
Prediction of next location after resampling
Convergence of particle guesses of mean velocity
Iteration
mean velocity
particles
current
History of prediction of next location

Figure A.2: Screenshot of 1-D robot PF for a single behavior from function `cdlm_robot`.

5. History of convergence (ideally) of values of $\alpha_t^{(n)}$ over time to the true value of mean velocity α.

6. History of distribution of simulated locations $\{\zeta_t^{(n)}\}$ around the observed values $z_{1:T}$.

The function `cdlm_robot_twostate` implements the 1-D robot CDLM with two movement modes (see Section 4.4). The same parameters for the single-mode function `cdlm_robot` are available, in addition to

- additional mean parameter for the second mode $\alpha_t \mid (\lambda_t = 2)$ and prior on this mean

- true constant transition probabilities $p_{1\to 2}$ and $p_{2\to 1}$

- option to have transition probabilities known or estimated. If estimated, the user provides a Dirichlet prior distribution on them.

Both behavioral modes' velocity distributions are assumed to have the same variance for simplicity. The graphs in Figure A.3 are

1. Initial distributions of $v_t \sim \mathcal{N}(\alpha^{(n)}, \sigma \mid \lambda_t)$ for each behavior type. Here, type 1 is slow and type 2 is fast.

2. Predictions and confidence intervals of $\hat{\mathbf{y}}_{t+1} \mid \lambda_t$. The second mode is shown by rectangles of the same color with black crosshatches.

3. Overall particle resampling weights $\{w_t^{(n)}\}$ (top) and behavior-conditional weights $\{w_{t|k}^{(n)}\}$ (bottom).

4. Resampled predictions $\hat{\mathbf{y}}_{t+1}$ after particles are resampled and then behaviors λ_t are propagated by the conditional weights. As before, the resampling is shown in slow-motion.

5. History of convergence (ideally) of values of $\alpha_t^{(n)} \mid \lambda_t$ over time to the true value of mean velocity α for each behavior.

6. History of distribution of simulated locations $\{\zeta_t^{(n)}\}$ around the observed values $z_{1:T}$.

7. If transition probabilities are unknown, the estimated marginal Beta distributions of each pair $p_{1\to2}$ and $p_{2\to1}$, which ideally should increase in density around the true values, shown by the vertical lines.

8. History of the proportion of particles predicting the true behavior λ_t at each step t, shown by the heights of the bar. The bars are shaded according to the true, unobserved behavior type. Ideally all of the bars should be high, reflecting consistently high classification accuracy.

The function `cdlm_robot_twostate_2D` implements a 2-D CDLM with two behavioral modes, similar to the shark model but without most of the complicating factors of turn angle modeling, regions, and interactions. Similar to

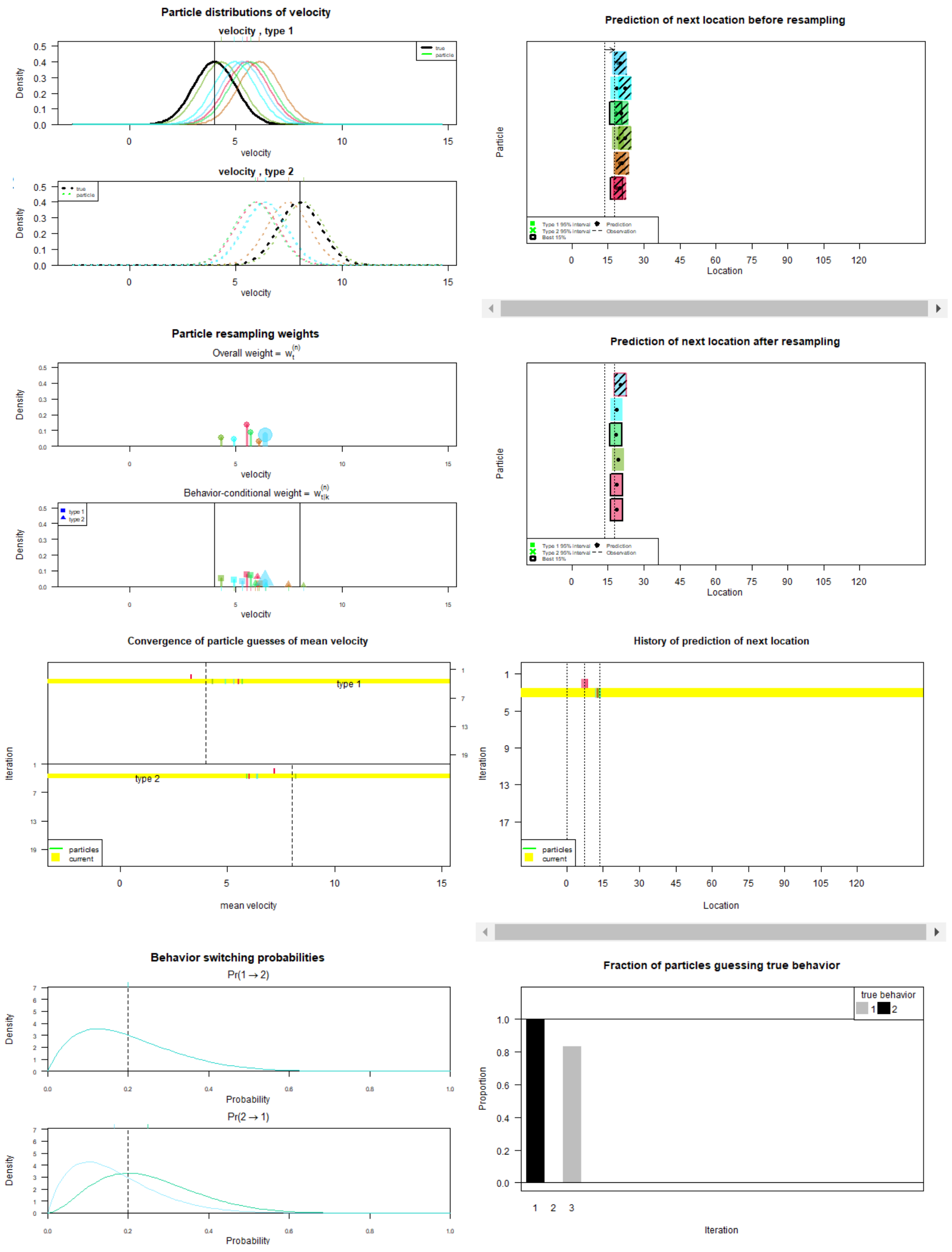

Particle distributions of velocity
velocity , type 1
velocity , type 2
Density
velocity
true
particle
Prediction of next location before resampling
Prediction of next location after resampling
Particle
Location
Type 1 95% interval
Type 2 95% interval
Best 15%
Prediction
Observation
Particle resampling weights
Overall weight = $w_t^{(n)}$
Behavior-conditional weight = $w_{t|k}^{(n)}$
type 1
type 2
Convergence of particle guesses of mean velocity
mean velocity
Iteration
particles
current
History of prediction of next location
Behavior switching probabilities
Pr(1 → 2)
Pr(2 → 1)
Probability
Fraction of particles guessing true behavior
true behavior
Proportion

Figure A.3: Screenshot of 1-D robot PF for two movement modes from function `cdlm_robot_twostate`.

the 1-D two behavior CDLM, here there are two behaviors and the goal is to model the mean of the log-speed $\ln(v_t)$ for each behavior λ_t. To avoid the compilations of the wrapped normal distribution, we assume the turn angle distribution is known. The log-speed distributions also have a common known variance, as before.

The same parameters for the two-mode function `cdlm_robot_twostate_2D` are available, in addition to

- known variance τ^2 of the turn angle distribution, which is centered at $\beta = 0$.
- confidence level (default 50%) of the posterior ellipse around predictions.

The graphs in Figure A.4 show output for the function `cdlm_robot_twostate_2D`. Many of them are the same as the 1-D version.

1. Initial distributions of log-speed $\ln(v_t) \sim \mathcal{N}(\alpha^{(n)}, \sigma \mid \lambda_t)$ for each behavior type. Here, type 1 is slow and type 2 is fast.
2. Predictions and confidence ellipses of $\hat{\mathbf{y}}_{t+1} \mid \lambda_t$. The second mode is shown by ellipses with dashed borders.
3. Overall particle resampling weights $\{w_t^{(n)}\}$ (top) and behavior-conditional weights $\{w_{t|k}^{(n)}\}$ (bottom).
4. Resampled predictions $\hat{\mathbf{y}}_{t+1}$ after particles are resampled and then behaviors λ_t are propagated by the conditional weights. As before, the resampling is shown in slow-motion.
5. History of convergence (ideally) of values of $\alpha_t^{(n)} \mid \lambda_t$ over time to the true value of mean velocity α for each behavior.
6. History of simulated locations $\{\boldsymbol{\zeta}_t^{(n)}\}$, shown by black 2-D density. The observed trajectory is shown as a red path on top. Ideally the density should cluster around the observations.

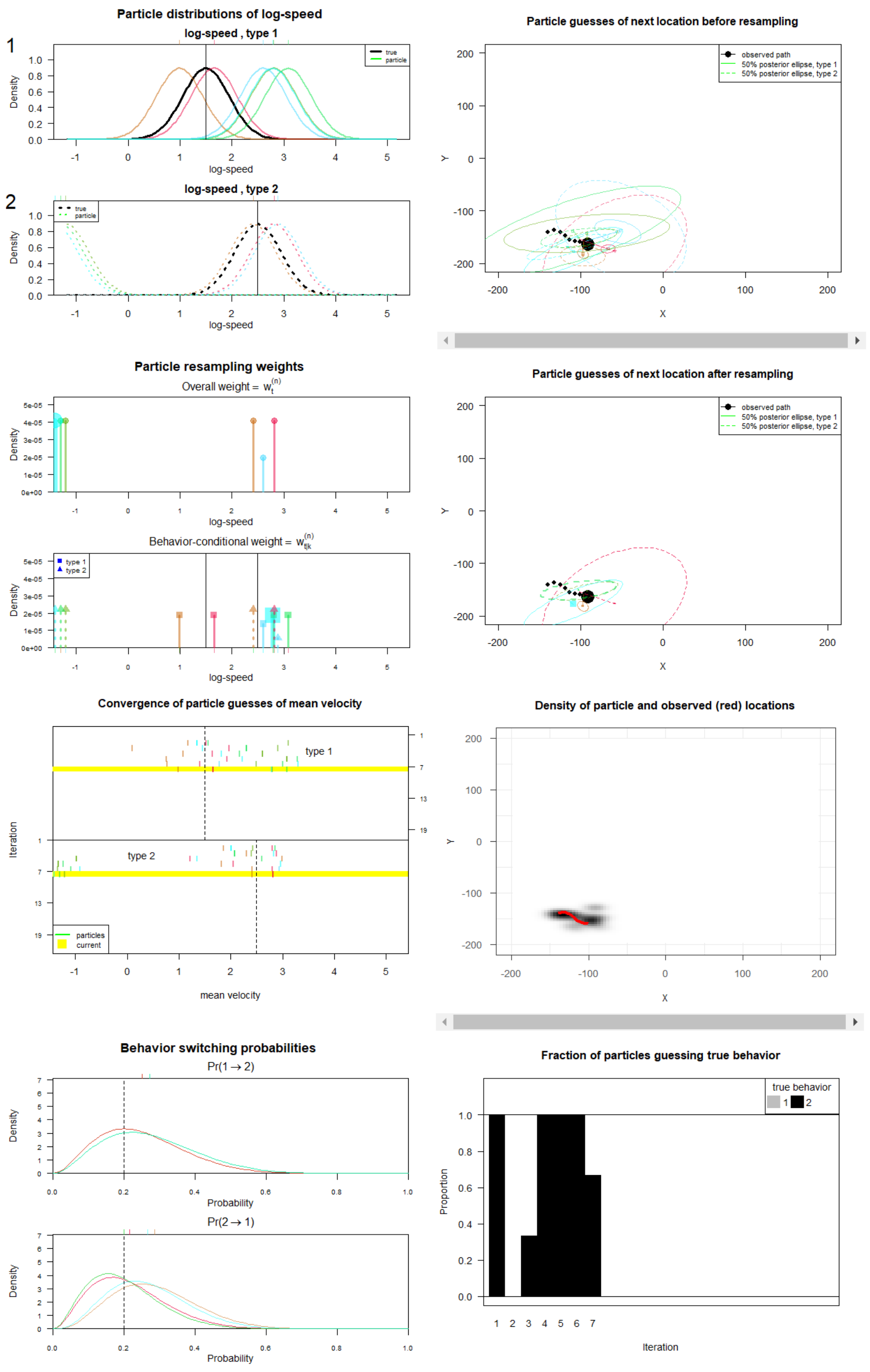

Particle distributions of log-speed
log-speed , type 1
log-speed , type 2
Density
log-speed
true
particle
Particle guesses of next location before resampling
Particle guesses of next location after resampling
observed path
50% posterior ellipse, type 1
50% posterior ellipse, type 2
Particle resampling weights
Overall weight = $w_t^{(n)}$
Behavior-conditional weight = $w_{t|k}^{(n)}$
type 1
type 2
Convergence of particle guesses of mean velocity
Iteration
mean velocity
particles
current
Density of particle and observed (red) locations
Behavior switching probabilities
Pr(1 → 2)
Pr(2 → 1)
Probability
Fraction of particles guessing true behavior
true behavior
Proportion
Iteration
X
Y

Figure A.4: Screenshot of 2-D robot PF for two movement modes from function `cdlm_robot_twostate_2D`.

7. Beta distributions of each pair of transition probabilities $p_{1\to 2}$ and $p_{2\to 1}$, if unknown.

8. Particle classification accuracy of the behavior λ_t.

A.2 Shark EKF parameter updates

Kalman filter recursions

Using the linear relations and Gaussian noise in the KF, equations for recursive updates of the mean vector and covariance matrix, and the posterior of $\mathbf{y}_t$ at each step can be derived. Due to the Markovian property, the dependence is actually only on the previous observation.

Adapted from Särkkä ([45], page 57). Here the superscript $^-$ indicates the parameter value (e.g., the vector $\mathbf{m}_t$) before being updated to reflect the most recent observation $\mathbf{y}_t$.

$$\mathbf{x}_t \mid \mathbf{y}_{1:(t-1)} \sim N(\mathbf{m}_t^-, \mathbf{P}_t^-); \qquad \mathbf{y}_t \mid \mathbf{y}_{1:(t-1)} \sim \mathcal{N}(\mathbf{M}_t\mathbf{m}_t^-, \mathbf{S}_t)$$

$$\mathbf{x}_t \mid \mathbf{y}_{1:t} \sim \mathcal{N}(\mathbf{m}_t, \mathbf{P}_t)$$

$$\mathbf{m}_t^- = \mathbf{L}_t\mathbf{m}_{t-1}$$

$$\mathbf{P}_t^- = \mathbf{L}_t\mathbf{P}_{t-1}\mathbf{L}_t^T + \mathbf{Q}_t$$

$$\mathbf{m}_t = \mathbf{m}_t^- + \mathbf{P}_t^-\mathbf{M}_t^T(\mathbf{y}_t - \mathbf{M}_t\mathbf{m}_t^-) \qquad \text{(mean update given prediction error to } \mathbf{y}_t\text{)}$$

$$\mathbf{S}_t = \mathbf{M}_t\mathbf{P}_t^-\mathbf{M}_t^T + \mathbf{R}_t \qquad \text{(innovation covariance matrix)}$$

$$\mathbf{K}_t = \mathbf{P}_t^-\mathbf{M}_t^T\mathbf{S}_t^{-1} \qquad \text{(Kalman gain matrix)}$$

$$\mathbf{P}_t = \mathbf{P}_t^- - \mathbf{K}_t\mathbf{S}_t\mathbf{K}_t^T$$

State ($\mathbf{x}_t$) parameter updates

See reference by Murphy ([37]) for posterior inference on the normal distribution.

For behavior (λ) $k \in \{0,1\}$

$$\begin{aligned}\ln(v_c):\ p(\alpha_k,\sigma_k^2) &\sim \mathcal{N}(\alpha_k \mid \alpha_{0,k}, \kappa_{1_{0,k}}\sigma_k^2) &\times&\ \mathcal{G}^{-1}(\sigma_k^2 \mid a_{0,k}, b_{0,k})\\ \theta_c:\ p(\beta_k,\tau_k^2) &\sim \mathcal{N}(\beta_k \mid \beta_{0,k}, \kappa_{2_{0,k}}\tau_k^2) &\times&\ \mathcal{G}^{-1}(\tau_k^2 \mid c_{0,k}, d_{0,k})\end{aligned}$$

The turn angle θ_c really has a wrapped normal distribution. However, the normal distribution can be used. We draw the next bearing $\psi_c \sim \mathcal{N}(\psi_{c-1} + \beta_k, \tau_k^2)$. The resulting angle $\theta_c = \psi_c - \psi_{c-1}$ (before wrapping) is used to update the parameters of the wrapped normal distribution the same way as the normal is.

Perform the following recursive updates for each particle at iteration c. Updates are only performed to parameter for which λ_c is observed:

$$\begin{aligned}\kappa_{1_{c,\lambda_c}} &= (\kappa_{1_{c-1,\lambda_c}}^{-1} + 1)^{-1}\\ \kappa_{2_{c,\lambda_c}} &= (\kappa_{2_{c-1,\lambda_c}}^{-1} + 1)^{-1}\\ \alpha_{c,\lambda_c} &= (\kappa_{1_{c-1,\lambda_c}}\alpha_{c-1,\lambda_c} + \gamma_c)\kappa_{1_{c,\lambda_c}}\\ \beta_{c,\lambda_c} &= (\kappa_{2_{c-1,\lambda_c}}\beta_{c-1,\lambda_c} + \phi_c)\kappa_{2_{c,\lambda_c}}\\ a_{c,\lambda_c} &= a_{c-1,\lambda_c} + 0.5\\ c_{c,\lambda_c} &= c_{c-1,\lambda_c} + 0.5\\ b_{c,\lambda_c} &= b_{c-1,\lambda_c} + 0.5(\alpha_{c-1,\lambda_c}^2\kappa_{1_{c-1,\lambda_c}}^{-1} + \gamma_c - \alpha_{c,\lambda_c}^2\kappa_{1_{c,\lambda_c}}^{-1})\\ d_{c,\lambda_c} &= d_{c-1,\lambda_c} + 0.5(\beta_{c-1,\lambda_c}^2\kappa_{2_{c-1,\lambda_c}}^{-1} + \phi_c - \beta_{c,\lambda_c}^2\kappa_{2_{c,\lambda_c}}^{-1})\end{aligned}$$

Inverse-Wishart Distribution Parameter updates

Here, Δ_Υ is the constant-length time interval length.

$\begin{bmatrix}\zeta_{1,c}\\ \zeta_{2,c}\end{bmatrix}$ is bivariate normal with covariance matrix Σ_{ζ_c}

The inverse-Wishart distribution is updated by summing squared errors:

$\Sigma_{\zeta_c} \sim \mathcal{W}_2^{-1}(\mathbf{\Lambda}_{\zeta_c}, \eta_{\zeta_c})$

$$A_c = \begin{bmatrix}\zeta_{1,c} - (\zeta_{1,c-1} + v_{c-1}\Delta_{c-1}\cos(\psi_{c-1}))\\ \zeta_{2,c} - (\zeta_{2,c-1} + v_{c-1}\Delta_{c-1}\sin(\psi_{c-1}))\end{bmatrix}$$

$\mathbf{\Lambda}_{\zeta_c} = \mathbf{\Lambda}_{\zeta_{c-1}} + A_cA_c^T; \qquad \eta_{\zeta_c} = \eta_{\zeta_{c-1}} + 1$

Here, state $\mathbf{x}_c$ occurs at constant interval time Υ_c, where $\Upsilon_c < H_t \leq \Upsilon_{c+1}$ and H_t is the observation time of observation $\mathbf{y}_t$. The time gap is $\Delta_t = (H_t - \Upsilon_c) < \Delta_\Upsilon$.

$$\mathbf{y}_t = \begin{bmatrix} z_{1,t+1} \\ z_{2,t+1} \end{bmatrix} \text{ is bivariate normal with covariance matrix } \Sigma_{z_{t+1}} \mid \lambda_c$$

$$\Sigma_{z_{t+1}} \mid \lambda_c \sim \mathcal{W}_2^{-1}(\mathbf{\Lambda}_{z_{t,\lambda_c}}, \eta_{z_{t,\lambda_c}})$$

$$B_t = \begin{bmatrix} z_{1,t+1} - (\zeta_{1,c} + v_c \Delta_t \cos(\psi_c)) \\ z_{2,t+1} - (\zeta_{2,c} + v_c \Delta_t \sin(\psi_c)) \end{bmatrix}$$

$$\mathbf{\Lambda}_{z_{t,\lambda_c}} = \mathbf{\Lambda}_{z_{t-1,\lambda_c}} + B_t B_t^T; \qquad \eta_{z_{t,\lambda_c}} = \eta_{z_{t-1,\lambda_c}} + 1$$