# Changes in Trajectories of Foraging Agents Under Spatial Learning

Camille Mirmiran

Thesis submitted to the Faculty of Science in partial fulfillment of the requirements
for the degree of
Master of Science Mathematics and Statistics[1]

Department of Mathematics and Statistics
Faculty of Science
University of Ottawa



[1]The M.Sc. program is a joint program with Carleton University, administered by the Ottawa-Carleton Institute of Mathematics and Statistics

# Abstract

The goal of this thesis is to identify differences and consistencies in the trajectories taken by foraging agents before and after they have learned the location of a target. The challenge is that these agents do not go directly towards the target after learning and keep a certain amount of randomness in their paths. We use different versions of discrete curvature and head angle as tools in this analysis. We also build models of foraging agents using stochastic processes with data supported parameters.

# Résumé

L'objectif de cette thèse est d'identifier des différences et invariances de trajectoires d'agents cherchant de la nourriture avant et après avoir appris l'emplacement de cette nourriture. La difficulté est que ces agents ne se dirigent pas directement vers la nourriture après avoir appris où elle se situe et préservent des mouvements aléatoires dans leurs trajets. Nous utilisons des versions différentes de courbure discrète et d'angle de tête comme outils dans cette analyse. Nous construisons aussi des modèles d'agents cherchant de la nourriture en utilisant des processus stochastiques avec des paramètres dérivés de données réels.

# Dedications

To my wife Samira.

# Acknowledgement

I would like to thank Drs. Maia Fraser and Leonard Maler for their guidance, patience, and support throughout this research. I would also like to thank my colleagues in the department of molecular medicine and the department of mathematics and statistics who have helped me along the way.

Merci à Armand, Adam, Roxane, et mes parents.

# Contents

# List of Figures

# List of Tables

# Introduction

Animal behaviour has been traditionally studied in spatially restrained animals such as mice in mazes [DR06, SHT+12]. Although this approach is useful for motor-sensory studies of the brain, it is not as useful in the context of modern questions in behavioural neuroscience. In contrast, the study of animal foraging in open space instead of restricted to mazes has become an actively researched topic, particularly when looking at its relationship to the brain and behaviour [JLM16, BEY+20, OD71]. It is known that the study of animal foraging generates highly complex data and requires data analysis techniques that are simply not available through off-the-shelf data analysis packages [Jua22].

In foraging, vision is typically the dominant sense guiding the animal to its target and so is the dominant sense for spatial learning. When vision is not an option, such as in the case of blind rats, spatial learning can occur but the sense(s) used have not been clearly identified [SCTBP98]. Weakly electric fish can learn spatial relations in the dark using their electrosense [JLM14, EWM21]. That is, weakly electric fish can use electrolocation to situate themselves in an environment under poor visibility conditions [ARS99, BLL07, CCM05, CMM20]. After learning, these fish can use path integration [MW88] to find landmarks and food [JLM16, WHGJ+18, EWM21]. Although the time and distance of the trajectories taken by these fish has been studied and used to demonstrate learning [JLM16], the evolution of the trajectories themselves has not been studied. Particularly, differences in trajectories before and after learning in features other than time and distance remain unexplored. This analysis aims to fill that gap by exploring which aspects change and which remain constant during learning in freely foraging agents. In particular, some features that can be recorded are directly related to brain activity and as such have immediate interest. In our case, the agents are Gymnotus sp. weakly electric fish foraging in an open water tank, and one such feature is the Electric Organ Discharge (EOD). Initially, the analysis is done through the study of planar curvature of the trajectories, with attempts to construct a discrete equivalent for planar curvature given non-continuous data. There is dense literature on discrete curvature estimation [CMT01][NR17], the conclusion of which being that there are many definitions and some are more appropriate than others for various problems [MSR07]. In this analysis, we use various methods of computing discrete curvature in order to find the most appropriate definition for our

purposes. Once a working definition of discrete curvature has been found, we use this to generate simulations based on this definition. That is, we use the definition found to simulate artificial and virtual agents' trajectories which we then compare to real foraging trajectories.

# Chapter 1

# The Data

## 1.1 Smoothing and Interpolation

In this section, we describe two interpolation methods and two smoothing methods. Particularly, we describe linear interpolation and cubic spline interpolation as well as moving average smoothing and binomial filter smoothing.

Consider a mapping, $\gamma$, from a finite subset of $\mathbb{R}$ to the plane. In interpolation, we have limited observations of $(t, \gamma(t))$ pairs and we wish to estimate $\gamma(t)$ for some $t$ "between" the already observed ones. Note that, by contrast, extrapolation (not covered here) estimates $\gamma(t)$ for some $t$ "beyond" the already observed data points (usually for the purposes of prediction). Smoothing, on the other hand, is a simplifying approximation for $\gamma$. Smoothing is only influenced by observed values of $\gamma$ but the resulting function need not necessarily pass through all these points, in fact, it may pass through none of the points and still be considered a smoothing. Notice that interpolation doesn't revise already existing points, but smoothing does. In the case of interpolation, the function obtained must pass through all given points, only the inter-points segments are newly generated.

Taking a different perspective, we can view $\gamma : [a, b] \to \mathbb{R}^2$ as two independent time series: The x-coordinate time series, $x(t)$, and the y-coordinate time series, $y(t)$. Typically, in applied problems (and in our project), the sampling of points is equidistant in the domain of $x(t)$ and $y(t)$. That is, the sampling of points is equally spaced in time.

### 1.1.1 Linear Interpolation

Given two points in the plane $(x_1, y_1)$ and $(x_2, y_2)$, we can connect the two points continuously by a unique linear segment starting at one point and ending at the other. Once we have the linear function given from points $(x_1, y_1)$ and $(x_2, y_2)$, we

may evaluate that function at a set number of equidistant points. In doing so, we add a finer (artificial) granularity to the original two data points.

Given a sequence of multiple points $(x_1, y_1), \ldots, (x_n, y_n)$, we can connect each consecutive point by linear interpolation to obtain a piecewise linear function. This is called the linear interpolation of points $(x_1, y_1), \ldots, (x_n, y_n)$.

Once we have the piecewise linear function given by the linear interpolation of points $(x_1, y_1), \ldots, (x_n, y_n)$, we may evaluate that function at a set number of equidistant points between each two points. In doing so, we add a finer (artificial) granularity to the original data points. Usually, the set number of points used for the evaluation is the same between all sequences of two points.

### 1.1.2 Cubic Spline Interpolation

Given a series of points $(x_1, y_1), ..., (x_n, y_n)$, a spline is a piecewise polynomial whose segments meet at points called knots. In the case of a cubic spline, the polynomial segments have degree 3 and the first and second derivatives are continuous while the third derivative is not. They are thus part of the functional set $C^3$[CW78]. The first and second derivatives of the whole function also exist, i.e., one-sided derivatives match at the knots. The whole function is thus part of the functional set $C^2$. In the statistical software R, cubic splines may be obtained from data using the function *splinefun*.

As with linear interpolation, we may evaluate the resulting spline function at a constant interval that is shorter than the interval between the observed values of $\gamma$ to create a finer granularity for our data, as an estimate of $\gamma$. Usually, the set number of points used for the evaluation is the same between all sequences of two points.

### 1.1.3 Moving Average Smoothing

A moving average is a smoothing technique that replaces time series data points by the average of the data points within a given span. The span is usually an odd number so that the average may be computed equally on the left and right of the data points. We may describe moving average smoothing more mathematically as follows [BD02].

Consider the time series $x(t)$ and let q be a nonnegative integer, then the two-sided moving average smoothing of $x(t)$ is given by the following time series.

$$w(t) = (2q+1)^{-1} \sum_{j=-q}^{q} x(t-j)$$

As abbreviation for this moving average smoothing we will write MA(2q+1). In the statistical software R, a moving average may be applied by using the filter function and setting the coefficients to $(1, \ldots, 1)/n$. For example, MA(5) can be applied in the statistical software R by setting the filter coefficients to $(1, 1, 1, 1, 1)/n$.

### 1.1.4 Binomial Filter Smoothing

A binomial filter is a smoothing technique that replaces times series data points by a specific weighted average of the points within a given span [MM83]. It is very similar to the moving average smoothing with the exception that the coefficients are different. The weighted average corresponds to the binomial coefficients. For example, for a span of 5, the coefficients of the weighted average would be $(1, 4, 6, 4, 1)/16$. The division by the sum of the coefficients allows for a sum of the weights to be one. The binomial coefficients allow for more focus on the current point and dissipating focus in the adjacent points. In the statistical software R, a binomial filter may be applied by using the function filter and inputting the binomial coefficients.

## 1.2 Unprocessed Data

Previous research done in our labs [JLM16] yielded data consisting of the following form: Time series of x and y coordinates for three fish (genus Gymnotus, same but unknown species, sex unknown, ages unknown, sizes ranging from 13.9 cm to 17.9 cm, obtained from Big Al Aquarium Services, Ottawa, ON, Canada) engaging in open space foraging in a square aquarium of size 1.8 m×1.8 m×0.3 m with a circular fence of radius 0.8 m and water of depth 10±1 cm (the shallow water allowed the recording to be essentially in two dimensions). The food source was always placed in the same position at the near center of the aquarium relative to the entry points of the fish (coordinates (x,y)=(0, -0.1), (-0.1,0) and (0, 0.1) for fish A, B and C, respectively). The fish foraged within the circular fence and they are essentially blind. Jun et al used four fish in their study, but a hard disk crash damaged the data for one fish so that it could not be used; spatial learning in this fish was entirely comparable to the other fish and so our smaller sample is still characteristic of electrosensory spatial learning in this species.

The video recording started from the moment the fish's entire body entered the circular fence and lasted until it came to less than 1 cm from the food source. The frame rate of the recording was originally 15 frames $s^{-1}$ but was synchronized with the recording of the fish's electric organ discharge rate (the focus of the previous experiments in our lab) through linear (spline) interpolation. The resulting frame data was then 100 samples $s^{-1}$.

Pulse-type weakly electric fish such as those studied in this analysis actively electrosense their environment by generating electric organ discharge (EOD) pulses. The rates of the pulses range from 20 to 150 Hz [CCM05]. In our data, we have the EOD rate. Because it was synchronized with the positional data, the EOD rate data also has a frame rate of 100 samples $s^{-1}$.

The data also included the head orientation. It was computed as the velocity vector of the head which was identified reliably and automatically through a Matlab

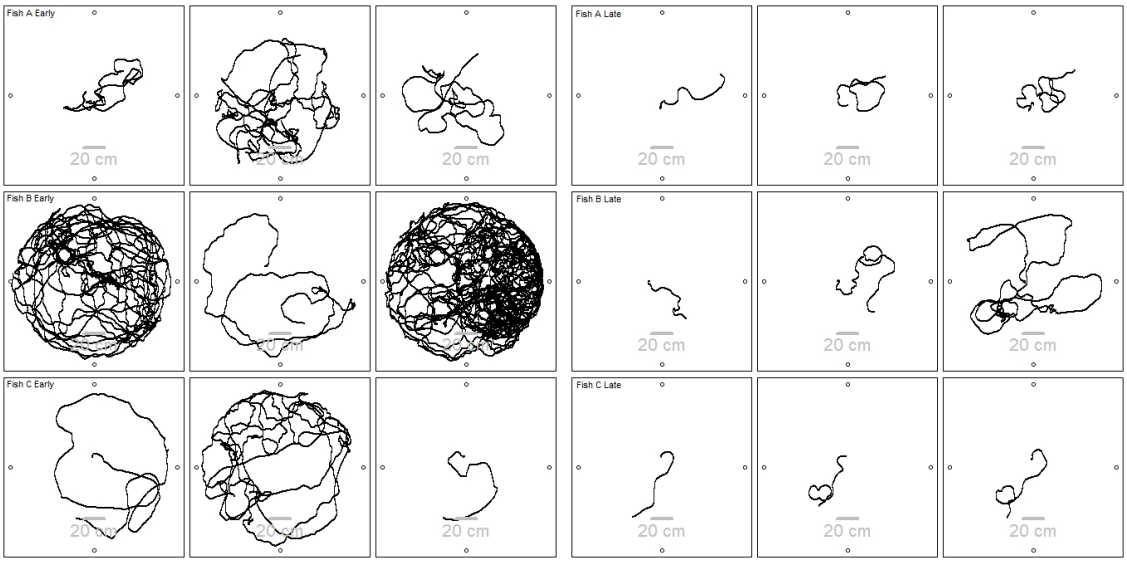


Figure 1.1: **Examples of trajectories for the gymnotiform fish.** Three examples of each fish and learning status (Early vs. Late). Note that the early fish trajectories vary in length (and thus duration) quite a bit. Nonetheless, they are consitently longer than the late learning trajectories. This is true in general, but exceptions do occur. Note also that most of the time, the fish travels with low curvature (mainly in a straight line) and has few sharp turns. Fish A starts at the top right corner. Fish B starts at the bottom right corner. Fish C starts at the bottom left corner.

script from the videos [JLM14]. The head direction was also previously synchronized and thus already linearly interpolated.

In Jun et al (2015), a definition was made for late learning fish (otherwise known as learnt fish). It was defined as any fish trajectory after and including the $7^{\text{th}}$ session of trials (Each session had a maximum of 4 trials per fish). This was because the 7th session of the trials corresponded to the first point of the plateau in learning with respect to distance travelled and time spent foraging, i.e., the point where the fish no longer improved its time spent to find the food source [JLM16] . The early learning fish was defined as any trajectory within the first 2 sessions of trials. This is because a sharp drop in time taken to find the food source was observed after the $2^{\text{nd}}$ session. We reuse the definitions of early learning fish and late learning fish in this analysis.

## 1.3 Data Processing

Because linear interpolation was used, and our focus was on the curvature/turning angle of the trajectories, we smoothed each fish trajectory using a moving average of

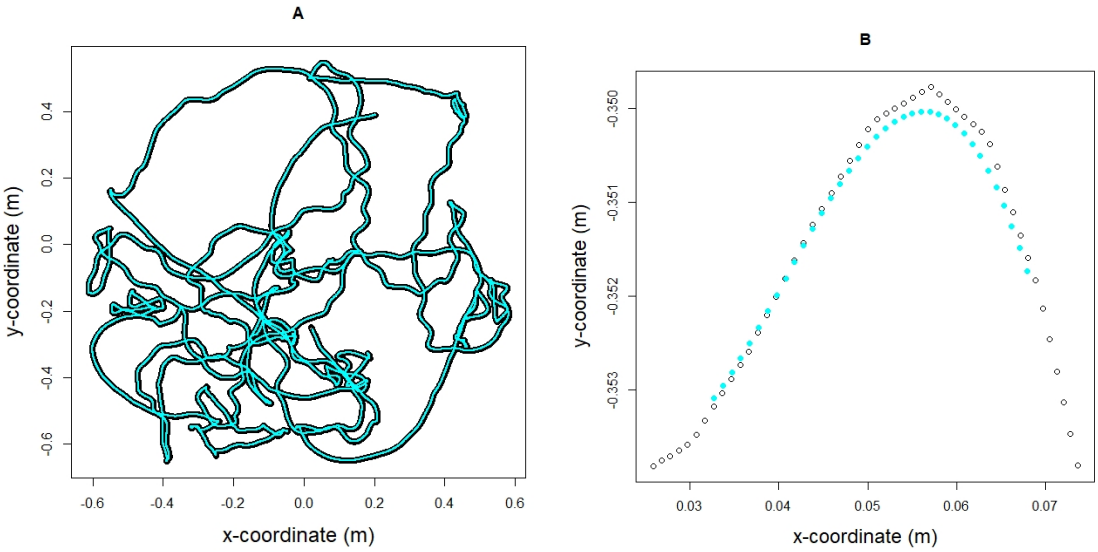


Figure 1.2: **Example fish trajectory (in black) along with its smoothed-out trajectory (in cyan).** The original unsynchronized coordinate data is not shown in this figure. The computation of curvature and all other analysis will be done on the smoothed-out data. **A** – Example of a fish trajectory after syncing and linear interpolation (in black) and the smoothed-out trajectory using MA15 (in cyan) derived from that data. Note that the modifications (cyan) are not different from the original synchronized data. **B** - Sub-trajectory of the fish trajectory shown in A (effectively zoomed-in) showing the synchronized and linearly interpolated data (in black) along with the smoothed data using MA15 (in cyan) derived from that data. Note that the first 7 and last 7 points of the MA15 data are undefined by nature of the moving average computation.

size 15 (MA15). This way, the trajectories were not piecewise linear, removing the strong bias towards 0 curvature/turning angle while minimally modifying the trajectories (Figure 1.2). Other spans for the moving average were used at various points in our research but MA15 allowed for the minimal modification of the trajectories at the human eye level of granularity. That is, allowing only modifications that would not be perceived by the naked eye, we took the highest span possible. Moving averages of span greater than 15 jeopardized the naked eye restriction. We also considered using binomial filter smoothing but upon obtaining similar results, we opted for the simpler moving average.

Since we did not need to compute the curvature of the head orientation one-dimensional time series, we did not apply the MA15 smoothing for the head orientation data. Likewise, we did not apply the MA15 smoothing to EOD rate.

However, linear interpolation affected the head orientation in a different way. The transition from zero to the head orientation of value 360, and the inverse (360 back to 0) had additional points that were added by the linear interpolation. Figure 1.3 shows the added points for a sample from a fish trajectory.

In Figure 1.3, we see two scenarios that are both nearly vertical line when looking at head orientation over time but are visibly different when looking at the actual trajectory. One of the scenarios is spurious and is an artifact of the way we handled arithmetic modulo 360, the other represents true head orientation changes. We thus had to distinguish between the two scenarios and correct the first one. We had to replace the interpolated values between a transition from 0 to 360 or 360 to 0 by the value 0 (alternatively 360). That is, we had to remove the mishandling of arithmetic modulo 360 by linear interpolation that replaced the previously discontinuous function by an approximative continuity. To do so, we took the difference in angle between time n and n+1 and checked if it was greater than 45 or less than -45. The value 45 was obtained by dividing 360 by 8, the number of points within the interpolation in Figure 1.3. However, with a manual check over all trajectories, we found that removing the linear interpolation would require a threshold change in head angle of 7 and -7. Thus, the threshold was set to 7 and -7. That is, should any change in head angle be greater than 7 or less than -7, the head orientation values for the steps n, n+1, and n-1, would be replaced by 0. The reason we included n-1 and n+1 is because the last and first change in head angle typically do not reach the threshold but are still part of the interpolated points (see Figure 1.3 in red, as an example). This correction allowed the removal of all linearly interpolated points between 0 and 360, points that would otherwise overstate the presence of sharp turns – a point of focus in our analysis. All the while, this threshold did not (with only a handful of exceptions) remove sharp turns.

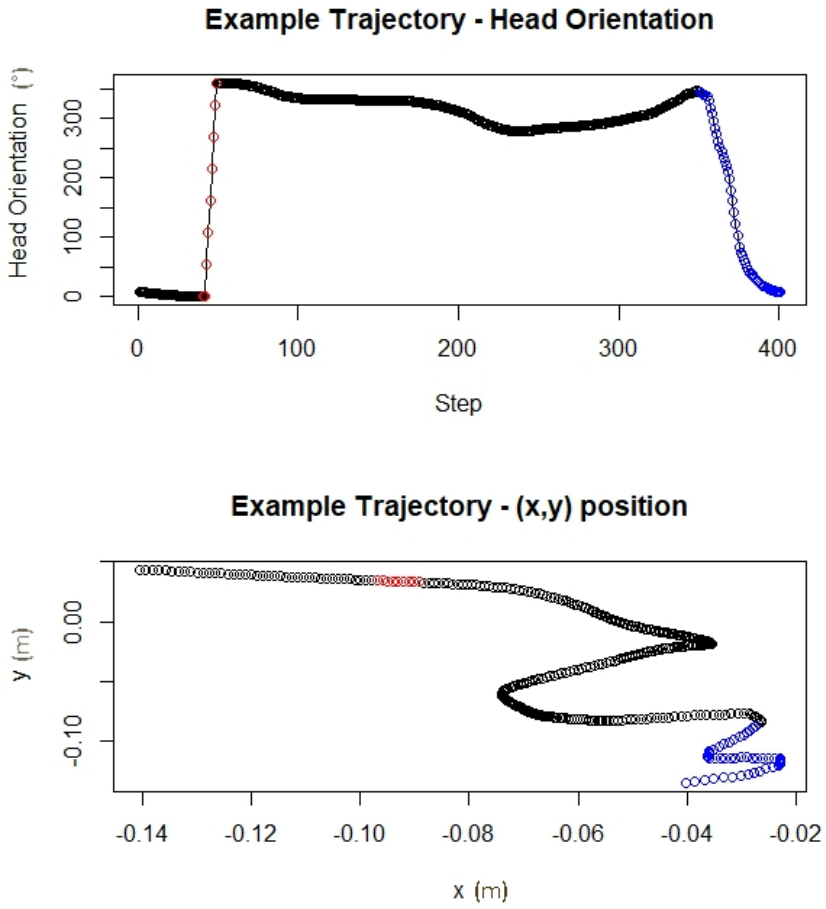


Figure 1.3: **Example of linear interpolation in head orientation. Top** – The original head orientation data for a sub-trajectory. **Bottom** – That same sub-trajectory plotted in the cartesian plane. Note the lack of change in head orientation during the red portion (one of the scenarios) of the sub-trajectory in the cartesian plane plot but an indication of large change in head orientation in the head orientation over time plot. In contrast, note the heavy change in head orientation in the blue portion (the other scenario) of the sub-trajectory in the cartesian plane and the corresponding actual changes in the head orientation over time plot.

# Chapter 2

# Curvature

## 2.1 Definition and Distribution of Curvature

There are multiple ways of defining and computing the planar curvature of a plane curve. All definitions and methods give the same result for the curvature at a point p. By design of these definitions, a straight line is said to have zero curvature. In this section, we will explore these different methods and how useful they are in an applied setting, particularly using the statistical software R and with goal to derive an accurate distribution of curvature for our fish trajectories (which are parameterized planar curves).

### 2.1.1 Parameterized Definition of Curvature

The most common way to define the curvature of a planar curve $\gamma$ is to first give it a unit-speed parameterization $t \to \gamma(t)$ s.t. $||\dot{\gamma}(t)|| = 1$, then the curvature is defined to be the magnitude of the second derivative $||\ddot{\gamma}(t)||$ at each point $\gamma(t)$ [Pre10]. Given a parameterization $t \to (x(t), y(t)) = \gamma(t)$ that is not unit-speed but is regular (first derivative is nowhere 0), we may use the following formula to compute the curvature ($\kappa$). See [Pre10] for the derivation.

$$\kappa = \frac{\dot{x}\ddot{y} - \dot{y}\ddot{x}}{(\dot{x}^2 + \dot{y}^2)^{\frac{3}{2}}} \tag{2.1.1}$$

The dots represent a derivative in terms of time $t$. This formulation is useful when a unit-speed parameterization is complicated or inexpressible.

For our purposes, these definitions are not directly applicable. This is because our data is discrete, not continuous. Since we need a continuous function to compute the curvature at each point of the function, we need a way to extend our discrete points into a continuous function that is moreover, differentiable. The simplest way to obtain a continuous function from discrete points is to use spline interpolation.

Particularly, 1st degree (or linear) spline interpolation is the simplest of all spline interpolations. However, the resulting function would be piecewise linear and thus would have a constant curvature of zero. For our purposes (computing the distribution of curvature), linear interpolation won't do. We then turn to cubic spline interpolation so that our function is closer (in the sense of having derivatives) to a smooth function and is not linear (curvature zero everywhere).

In [CL07], strong arguments are made for the use of cubic splines as opposed to splines of higher degree polynomials. We thus use cubic splines for turning our discrete data into differentiable "data".

R's cubic spline interpolation function *splinefun* returns a black-box input-output function that is the cubic spline interpolation of input data points which are used as knots. The R function *splinefun* also has a parameter "deriv" which returns the 1st, 2nd or 3rd derivatives once again in the form of a black-box input-output function. This is presumably the closest we can get to an analytical equation from a set of discrete points. Using R's *splinefun* function, we apply equation 2.1.1 to the unit circle (radius 1) and see if the output is the expected constant value of 1 (Figure 2.1).

For any of our purposes, in Figure 2.1, the overstatement of the curvature by such a small amount is negligible. However, we will still perform further tests to see if the computation of curvature goes as expected. Should this error turn out to be important, it will be found in other tests.

We thus turned to circles of different radii. We increased the granularity (how often we plot points) from 0.25 to 0.01 and computed the curvature for each circle. Since each circle's curvature is constant, we extracted the most common value as an output and deemed it to be the computed curvature of the circle of this radius. For input, we used the radius. Since the inverse of the radius of the circle equals its curvature at every point in the circle, we can directly test our computation of curvature method (through *splinefun* and equation 2.1.1) by inputting a radius and outputting the constant curvature. The expected result is the function $1/x$. The experimental results are given in Figure 2.2.

The results in Figure 2.2 retrospectively make sense. Our granularity is 0.01. When the radius gets close or smaller than the granularity, the circular shape is lost in the *splinefun* interpolations. This is confirmed in Figure 2.3, where we show the distortions get much worse if the granularity is too coarse and the radius too small. We conclude that when applied to the fish data, very sharp (high curvature) will be distorted. Since these sharp turns are important to our analysis, we looked for an alternative approach to curvature.

## 2.2 Unit-Speed Reparameterization

Suspecting the distortions of the curvature were originating from the non-unit speed parameterization formula for curvature, we turned back to the more standard defini-

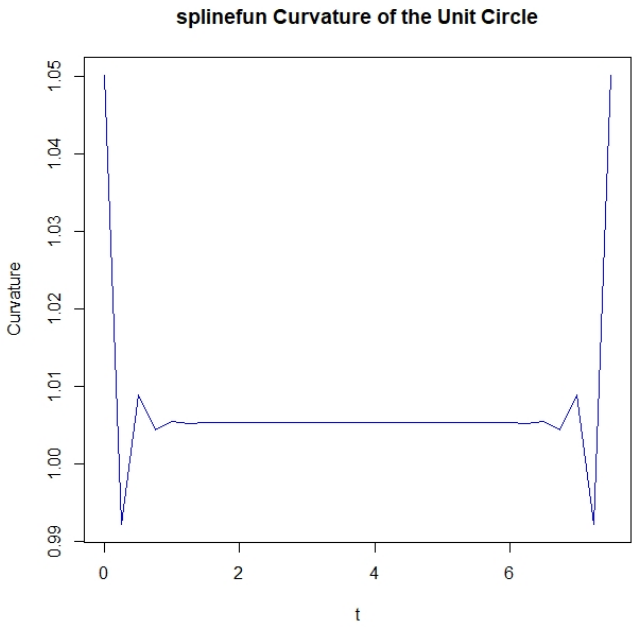


Figure 2.1: **Testing the non-unit speed curvature formula on the unit circle.** The curvature of the unit circle as computed with R's *splinefun* function without the assumption of unit-speed parameterization (equation 2.1.1). Note the accuracy of the result but also the slight overstatement of curvature (1.005 instead of 1). The input ranges from 0 to 7.5, creating points every 0.25 units (For a total of 30 points). The expected function is the constant (unit) function.

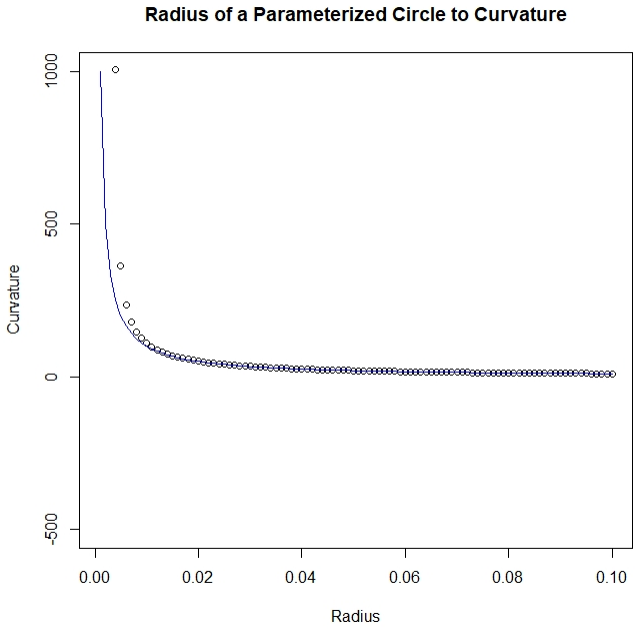


Figure 2.2: **Testing the computation of curvature for circles of different radii.** The radius of the input circle on the x-axis and the output curvature computed using Equation 2.1.1 and *splinefun*. In blue, the function 1/x, the expected result from radius to curvature. In black, the output given by the computation of curvature. Note the gradually increasing overstatement of curvature as we approach smaller and smaller radii.

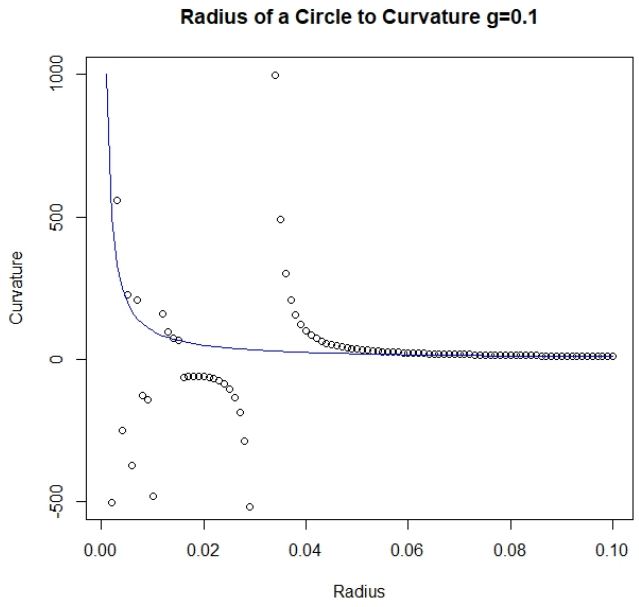


Figure 2.3: **Distortions of output curvature with small radii.** Distortions when granularity (g) is close or higher than the radius of the circle for which curvature is to be computed. In blue, the function 1/x, the expected result from radius to curvature. In black, the output given by the computation of curvature with granularity of the input circle at 0.1.

tion that depends on unit-speed reparameterization. That is, we aimed to reparametrize our given data into unit-speed and use the magnitude of the second derivative as curvature, i.e., a discrete equivalent of $||\ddot{\gamma}(t)||$ for the unit-speed planar curve $\gamma(t)$. We also used a different definition of derivative instead of the cubic spline definition. These definitions are given in equations 2.2.1 - 2.2.3.

$$\dot{\gamma}_i := \gamma_{i+1} - \gamma_i = (x_{i+1} - x_i, y_{i+1} - y_i) \tag{2.2.1}$$

$$\ddot{\gamma}_i := \dot{\gamma}_{i+1} - \dot{\gamma}_i = (\dot{x}_{i+1} - \dot{x}_i, \dot{y}_{i+1} - \dot{y}_i) \tag{2.2.2}$$

$$\kappa_i := ||\ddot{\gamma}_i|| \times sgn(-(\ddot{x}_i\dot{y}_i - \ddot{y}_i\dot{x}_i)) = \sqrt{\ddot{y}_i^2 + \ddot{x}_i^2} \times sgn(-(\ddot{x}_i\dot{y}_i - \ddot{y}_i\dot{x}_i)) \tag{2.2.3}$$

Here, $\kappa_i$ is the curvature at step $i$. The $sgn$ function takes the sign of the expression within the brackets, which is taken from the non-unit speed formula for curvature. That is, we took the sign (and only the sign) from a portion of the expression of non-unit speed curvature in order to compute the signed curvature instead of the absolute curvature.

To use the unit-speed definition of curvature, we initially thought of resampling and taking steps of unit distance. This way, between each point, there would be a distance of 1. However, our data ranges from -0.8 to 0.8 on both the x and y axis, so after 1 step, the resampling would be done and would return a single point. To remedy this, we thought of scaling up the data and only then resample. The question became "How much should we scale up our data?". Before we answer this question, let us look at the reparameterization algorithm (in pseudocode).

$n$ = total length of trajectory
$\tilde{\psi}_1 = \psi_1$
$j = 1$
$i = 2$
while$(i < n)$
    if$(||\tilde{\psi}_j - \psi_i|| < 1)$:
        $i++$
    else
        $\tilde{\psi}_{j+1} = \psi_i$
        $j++$
        $i++$
return $\tilde{\psi}_1, ..., \tilde{\psi}_j$

Here, $\psi_i$ is the $\text{i}^{\text{th}}$ point of the original data. What is effectively happening when this code is applied to our data is the addition of an already existing point as soon as the threshold of 1 unit of distance from the previous point is passed. This means that all consecutive points have a distance of at least 1, but potentially more. How

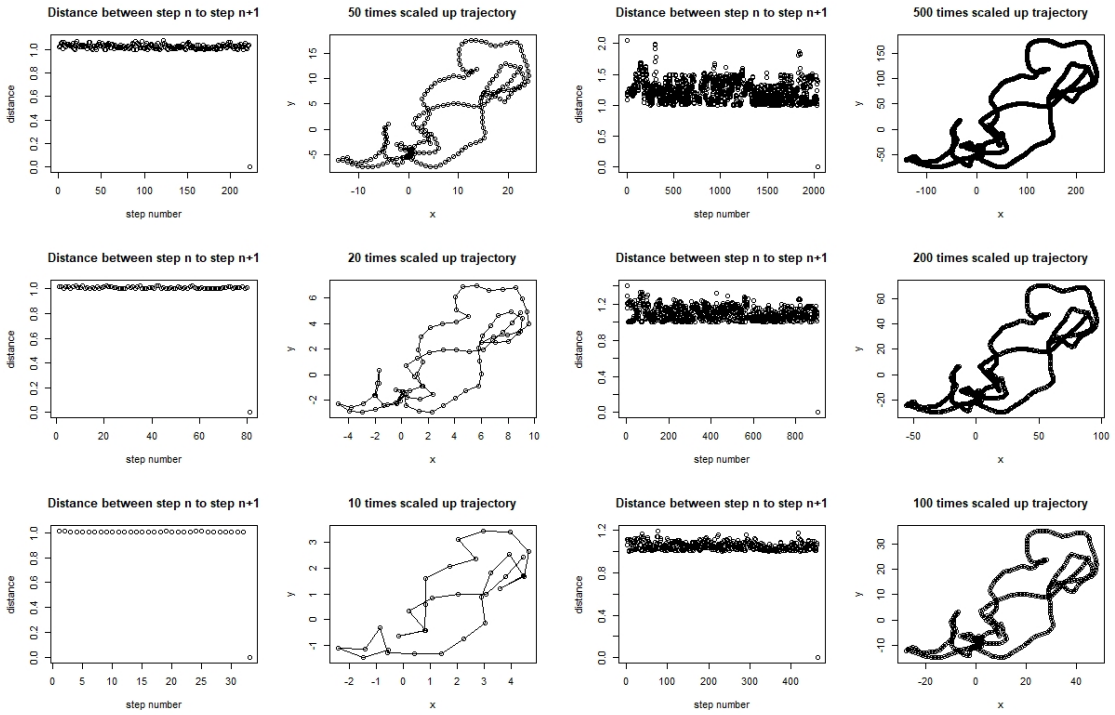


Figure 2.4: **Various resizing scales and their effect on the unit speed trajectory and distance noise.** All vertical units are in meters. The horizontal units for the trajectories are in meters as well. Notice that the smaller the scale used to resize the trajectory, the coarser the trajectory is made to be, and thus the closer to unit-speed the resulting parameterization. On the other hand, large scales for resizing yield very fine granularity of the data but gives a parameterization that is not close to unit-speed.

much is overshot by the algorithm depends on the granularity of the data (The average distance between consecutive points in the original data was $< 0.001$) and thus on the scaling factor. In Figure 2.4, we show the distances between each consecutive point along with the resulting resampling for a few different scale factors. The important conclusion from that figure is that the coarser the data, the closer it is to unit speed. Inversely, the finer the data, the more noise and overshoot there is to the distance between each point (imperfect unit-speed reparameterization).

When balancing solely the overshot (noise) and granularity of the data, one might be inclined to select a scale factor between 20 and 50. We continued our exploration using a scale factor of 27. This yielded noticeable differences in the distribution of curvature between early and late learning fish, particularly in the variance and kurtosis of these distributions (Table 2.1). However, as it turns out, the scale factor has a big impact on the distribution of curvature, and when correcting distortions

caused by the definition of curvature as in equations 2.2.1 - 2.2.3 (by applying a transformation of $asin(x/2)$), these differences became dependent on the scale factor (Figure 2.5).

| | Variance | Kurtosis |
|---|---|---|
| Fish A Early | 0.35 | 5.70 |
| Fish A Late | 0.29 | 3.31 |
| Fish B Early | 0.35 | 4.47 |
| Fish B Late | 0.31 | 3.77 |
| Fish C Early | 0.25 | 6.57 |
| Fish C Late | 0.22 | 5.01 |

Table 2.1: **Variance and Kurtosis of the distribution of (discrete) curvature at scale factor 27.** Note that both the variance and kurtosis are consistently higher for early learning fish than for late learning fish.

Figure 2.5 indicates that the scale factor plays an important role in determining differences between early and late learning fish. In this case, it determines if the kurtosis of early and late learning fish are different. At this point, we could either continue our analysis by choosing a scale factor (which would most likely be 27) or take a different approach. Because the scale factor plays such an important role in the distribution of curvature and we had to choose a scale factor, we felt we could not really choose a scale factor without imposing some bias in our findings. We thus opted for a different method of computation of curvature for discrete data. That is, despite some findings, our definition of curvature could likely still be improved.

## 2.3 Osculating Circle Definition of Curvature

The traditional definition of curvature is given by osculating circle. In this case, the curvature at point p is defined as the inverse of the radius of the circle passing through the point p and two additional points infinitesimally close to p. This definition is independent of parameterization. Such a circle is called an osculating circle. Once again, for our purposes, this definition is not applicable as we have discrete data. We considered instead an analogous definition of curvature where we replace "infinitesimally close to p" by "closest to p in the data". That is to say, the two additional points close to p would be the points just before and just after p. In the case of our fish data, this would be the points 0.01 seconds after and before the point for which we wish to compute curvature.

In the case of three discrete points, the unique circle passing by the three points is called the circumcircle. We can compute the radius of the circumcircle by using the Sine Law. More precisely, for a curve $\gamma$, at a point $\gamma_i$, we may compute the curvature

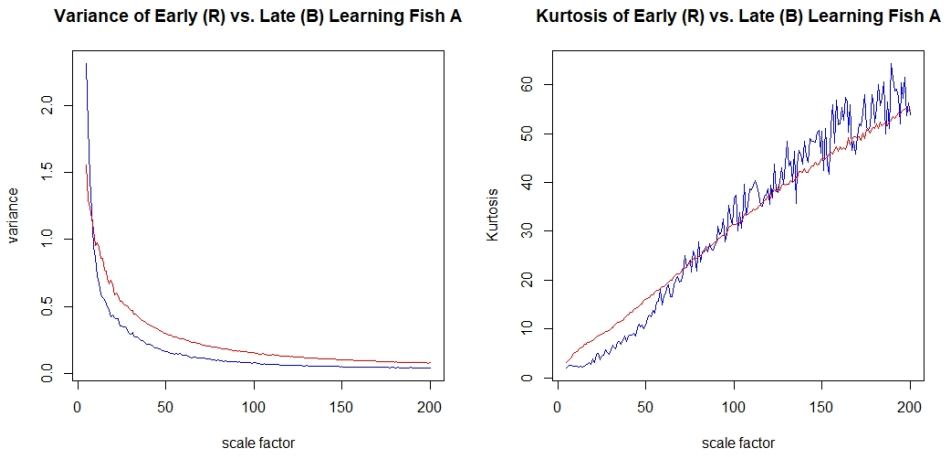


Figure 2.5: **Variance and Kurtosis of corrected discrete curvature over scale factor for fish A.** Note that apart from very small scale factors (where the data is under 100 points), the variance (Left) is consistently lower for late learning fish (Blue) than it is for early learning fish (Red). This is not the case for kurtosis (Right), where the late learning fish distribution has lower kurtosis only from 0 to 60, after which, the kurtosis of both early and late learning fish are relatively the same. Note that the kurtosis of late learning fish gets noisier as the scale factor increases. This is because a large scale factor means that most of the curvature will be near zero. Only a few points will have high curvature. It is these points that will determine the kurtosis. The larger the scale factor, the fewer high curvature points we will find, and thus, the fewer points that are truly contributing to the kurtosis. Fewer contributing points results in noisier data.

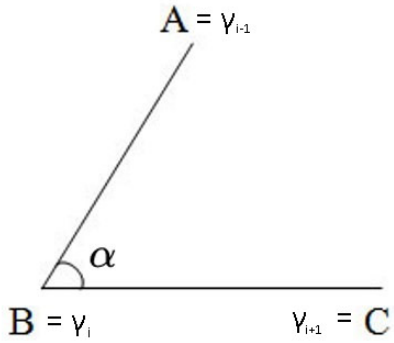


Figure 2.6: **Notation for equation 2.3.2**

(the inverse of the radius of the circumcircle) using equation 2.3.1, a straightforward manipulation of the Sine Law.

$$\frac{2 \times sin(\alpha)}{||\gamma_{i+1} - \gamma_{i-1}||} = \frac{1}{R} \tag{2.3.1}$$

where $\alpha$ is the angle between the previous point ($\gamma_{i-1}$) and the next point ($\gamma_{i+1}$) and $R$ is the radius of the circumcircle. Simplifying our notation through Figure 2.6, we can compute the angle $\alpha$ using equation 2.3.2.

$$cos(\alpha) = \frac{AB \cdot BC}{||AB||||BC||} \tag{2.3.2}$$

Equation 2.3.2 works well to find the absolute curvature at a point. However, we are more interested in the signed curvature, since our goal is to derive a (two-sided) distribution of curvature. A useful function to use in this scenario is the $atan2$ function (two argument arctangent function), which returns the angle between the given (x,y) coordinates and the x-axis, ranging from $-\pi$ to $\pi$. In our case, the formula is applied to compute the angle between the points before a certain point p (or $\gamma_i$) and after. That is, we want to compute $\alpha$. Knowing $cos(\alpha)$ and $sin(\alpha)$ allows us to determine $\alpha$ through equation 2.3.3. But $cos(\alpha)$ is given in 2.3.2 and $sin(\alpha)$ is obtained by replacing the inner product by the determinant in 2.3.2. So $\alpha$ can be computed using equation 2.3.3.

$$\alpha = atan2(AB \cdot BC, det(AB, BC)) \tag{2.3.3}$$

We can now compute the signed angle at any given point on the discrete curve, and in turn, use that signed angle to compute the radius of the circumcircle. The circumcircle acts as the analogue to the osculating circle and as a result, we can compute the signed discrete curvature of any point on the discrete curve (this without a scale factor like in the case of unit-speed parameterization seen in the previous

section). Before we compute the distribution of curvature using the circumcircle, let us justify its use.

The following test (with results displayed in Figure 2.7) is essentially the same as the test used in Figure 2.3. That is, we produced circles of radii ranging from -0.01 to 0.01 excluding the circle of radius 0. We used the unit-speed parameterization of each circle to produce points every 0.01 units travelled. This way, the circle of circumference 5 would have 500 points, but the circle of circumference 0.1 would only get 10 points, and its shape could be confused with a polygon instead of a circle. After producing the circles, we computed their curvature using two methods: the osculating circle method, and the parameterized method based on cubic splines. Because the curvature of circles is constant (In our case, this was verified manually for a few circles of varying radii), we could take any one value from the output of the computed curvature for each circle to determine the whole output curvature for that circle. Figure 2.7 shows on the x-axis the input radius of the circle and on the y-axis the output curvature for both the osculating circle method and the parameterized method based on cubic splines (in different colours). We also show the expected curvature based on the input radius. This is precisely the inverse of the input radius. The aim of the test is to show the different distortions both methods experience when the radius of the circle is smaller than the granularity of the input data. In this case in particular, it is when the radius of the input circle is smaller than 0.01.

From Figure 2.7, we can conclude that the distortions of the parametrized method based on cubic splines is significant. The errors are large, reaching up to 10000, and this will distort our distribution, exaggerating the tail and kurtosis.

From that same figure, we conclude that the osculating circle method also has distortions. However, they are absolutely correct (they are correct up to their sign). When producing a distribution that is assumed to be symmetric the errors in signage for negative values will appear as positive and the errors in signage of positives values will appear as negative. If our data has the same amount of both negative curvature and positive curvature over all values, our distribution for the method of osculating circle will be accurate. In our fish data, this assumption states that there is no bias towards the right or the left in terms of turning angle. We accept this assumption, as a top view of the trajectories indicate that this is the case, and thus can use the method of osculating circles to compute a distribution of curvature accurately.

Before the computation of the distribution of curvature based on the osculating circle method, let us illustrate through Figure 2.8 how such different results appear between the osculating circle method and the parameterized method based on cubic splines.

Through Figure 2.8, we can see how the parameterized method to compute curvature based on cubic splines returns larger curvature values than the method using osculating circles (through their discrete analogue: circumcircles). Neither of these methods is wrong, and neither is right, as we are using the data we have to produce

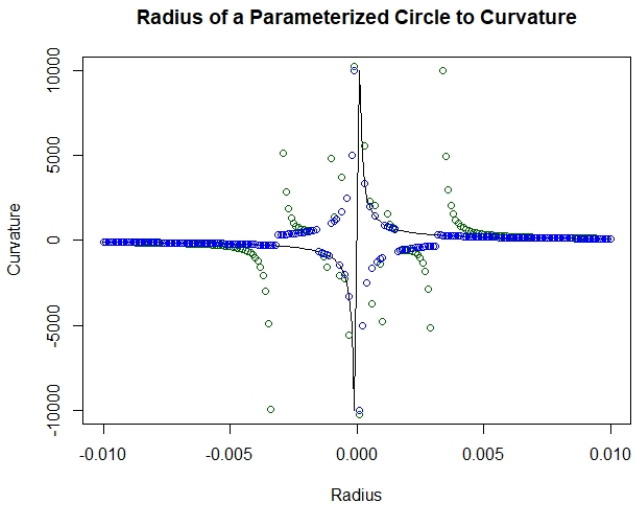


Figure 2.7: **Distortions of the osculating circle method of computing curvature (blue) and of the parameterized method based on cubic splines (green).** In black, the expected curvature value for each input radius. The granularity of the circles is 0.01, meaning every 0.01 units travelled along the circumference, we generate a point for the circle. Small circles have less points, and big circles have more points. This causes distortions when the radius is between -0.01 and 0.01. These distortions are illustrated here. Note that the osculating circle method gives distortions that are only absolutely wrong. That is, the output curvature is occasionally of the wrong sign when the radius gets close to 0. For the parametrized methods using cubic splines, the distortions are not only the wrong sign occasionally but are also off by up to 10000. This would exaggerate the tails of the distribution of curvature and overstate the kurtosis. Note also that the distortions for both methods have symmetry in the sense that there is no bias towards any side. Thus, when computing a distribution with data that is assumed to be symmetric, the sign of these distortions should balance out and cancel each other out to produce a distribution that is wrong only with absolutely incorrect values.

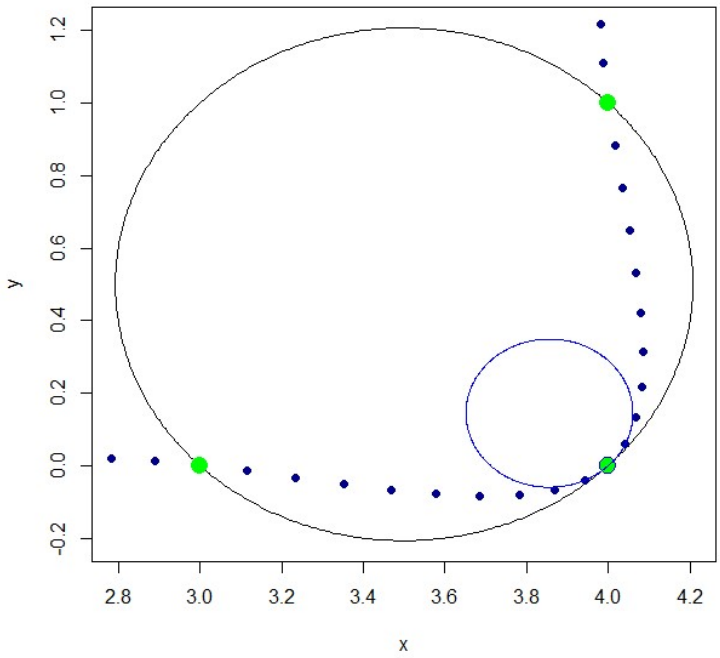


Figure 2.8: **Differences between curvature from the osculating circle method and the parameterized method based on cubic splines.** In light green, the original points of the data (example). In black, the osculating circle computed based on the osculating circle method. Its radius is large and thus returns a low curvature. In dark blue, the cubic spline interpolation generated by the light green points. They are cubic polynomials segments joined together at the light green points and satisfying some continuity properties. In light blue, the osculating circle based on the cubic spline interpolation used in the parametrized method of computing curvature using cubic splines. Note that the light blue circle passes through the middle light green points and the two adjacent points. The light blue circle has smaller radius and thus larger curvature at that point. Despite starting with the same data points (light green), the curvatures returned from the two methods are different. Because there are few data points, there is no right answer based solely on this figure.

a value we need. However, with our goal of computing an accurate distribution of curvature in mind, we opt for the osculating method of computing curvature. Figure 2.9 shows the distributions of curvature for each fish and learning status using the method of computing curvature based on osculating circles through their analogue of circumcircles. Figure 2.10 does the same but zooms in to exclude the extreme values and get a better idea of what the distribution looks like.

Based on Figure 2.10, there may be differences in variance and kurtosis between early and late learning fish. Because there are extreme points that were omitted in Figure 2.10, we expect inaccurate values for both variance and kurtosis (as they are sensitive to outliers). We thus turn to the non-parametric analogues of variance and kurtosis: interquartile range (IQR) and % outliers. In this case, the outliers are defined as 1.5 times away from the $1^{st}$ or $3^{rd}$ quartile [BL11]. We plot the non-parametric statistics in Figure 2.11, with the x-axis being the IQR and the y-axis being the percentage of outliers. Note that a different outlier threshold is used for each fish and learning status as the threshold depends on the data for that fish and learning status.

The results in Figure 2.11 are not very strong. Indeed, we cannot make any strong conclusions about the differences between early and late learning fish using the percentage of outliers in the distribution of curvature, much less so with the IQR of those distributions. The weak conclusion we can make is that the percentage of outliers in the distribution of curvature for early fish is on average higher than the percentage of outliers in the distribution of curvature for late learning fish. This is because both the weighted average (weights by length of trajectories, identified in Figure 2.11 as the circle) and the non-weighted average (identified in Figure 2.11 by the cross) are higher for early learning than for late learning in all fish (A, B, and C).

Therefore, although we have an accurate distribution of curvature using the circumcircle as an analogue to the osculating circle, we essentially failed to find differences between early and late learning using this method.

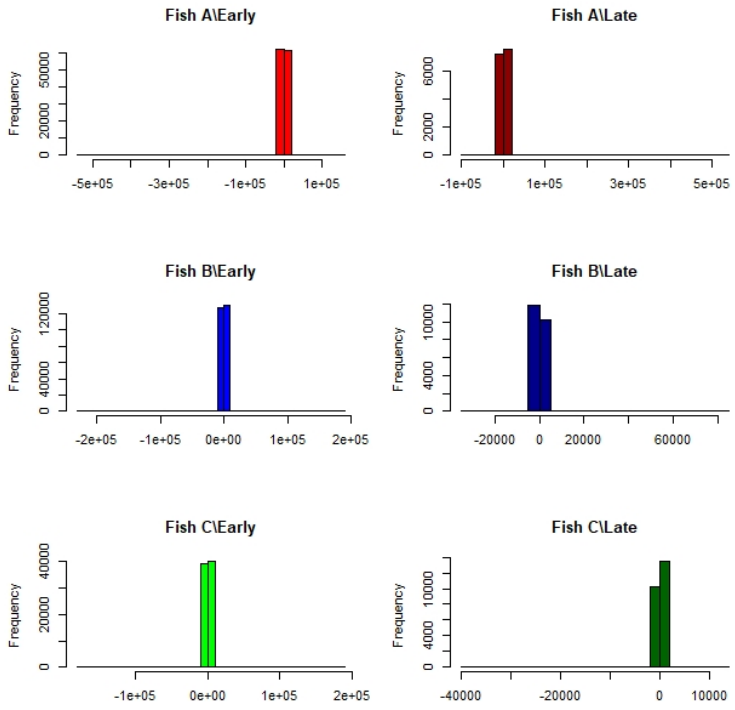


Figure 2.9: **Distributions of curvature using the method of computing curvature based on osculating circles through their analogue of circumcircles.** Note that the x-axis range is automatically generated by the plotting software (R) and indicates that there are values that reach up to that range. For example, Early Fish A has a minimum value of about $-5 \times 10^5$ and a maximum value of about $1 \times 10^5$. Although the distributions seem to be asymmetric, this is in fact not necessarily true. Indeed, the extreme points are rare and effectively determine the appearance of the distribution, so any small difference in minimum and maximum will return an asymmetric look. No real conclusions can be made from these histograms apart from the long tail of the underlying distribution.

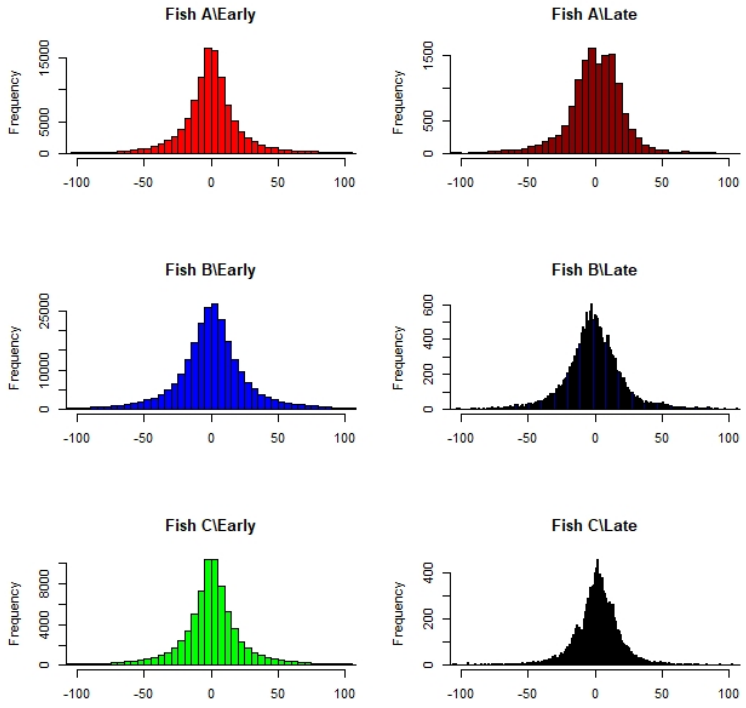


Figure 2.10: **Distributions of curvature using the method of computing curvature based on osculating circles through their analogue of circumcircles, zoomed in to -100 and 100.** Note that the distributions appear symmetric when the extreme points (seen in Figure 2.9) are removed. We observe a potentially larger variance in late learning fish distributions, particularly fish A. Note that the early learning fish have more data because their trajectories last longer (particularly in the case of fish B), what is important is the shape of the distributions, not necessarily the y-axis which counts frequency.

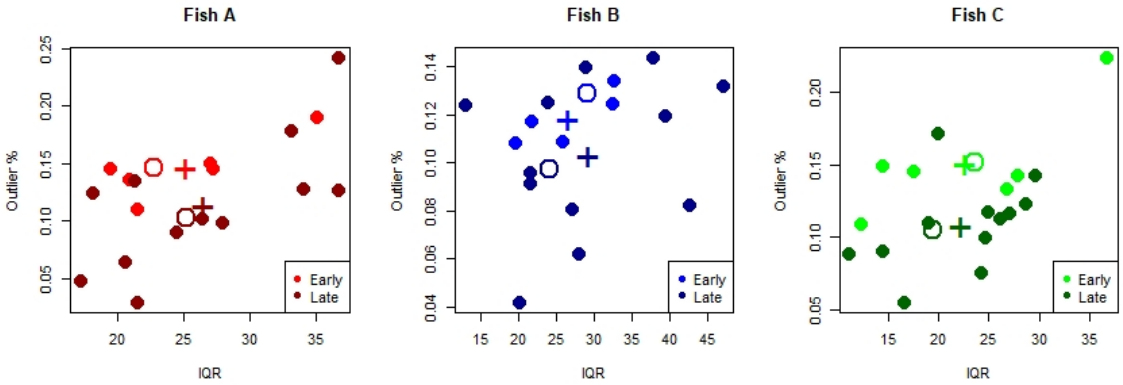


Figure 2.11: **IQR and percent outliers of the distribution of curvature as computed by the osculating circles method.** Each point is a trajectory whose IQR and % outliers in change in head angle has been computed and makes the x and y coordinates, respectively, in the plot. The plus sign is the centre of mass (average) of all the points of the same colour. The circle is the IQR and % outliers of all trajectories of one fish and learning status (i.e., one colour) when all the trajectories have been amalgamated. It can also be thought of as a weighted centre of mass with the weights being proportional to the length of each trajectory. Note that there is somewhat of a consistency in the separation of percent outliers but not in IQR.

# Chapter 3

# Simulations

## 3.1 Stochastic Processes and Statistical Distributions

In this section, we give definitions for the mathematical and statistical tools we will use in this chapter. Particularly, we define stochastic processes and a few important statistical distributions for our purposes.

### 3.1.1 Stochastic Processes

A stochastic process is a family of random variables $Y_0, Y_1, Y_2, \ldots, Y_n$. In this analysis, we will mainly focus on the case where $Y_i = \sum_{j=0}^{i} X_j$ and $X_j$ are i.i.d random variables. For example, we may consider a Poisson process, where the distribution of the $X_j$ is Poisson. The Poisson process has applications in numerous fields such as biology [ODA88] and economy [AB92]. It is commonly applied as it makes the least number the assumptions (from models where average time between events is known) and so is the simplest to start with in modeling.

### 3.1.2 Some Statistical Distributions

*Log-Normal and Normal Distribution*
The Gaussian (normal) distribution needs no introduction. The Log-Normal distribution is related to the Gaussian distribution in the sense that if $X \sim N(\mu, \sigma^2)$ then $exp(X) \sim lognormal(\mu, \sigma^2)$. It has previously been used in biological models, particularly pertaining to neuroscience [BM14], where it has been an accurate description of various activity related to the brain.
*Pareto Distribution*
The Pareto distribution is a long-tail distribution described by the density function in equation 3.1.1.

$$p(x) = \frac{\alpha x_m^\alpha}{x^{\alpha+1}} \tag{3.1.1}$$

where $\alpha$ is called the scale parameter and $x_m$ is the location parameter (minimum value). It was introduced by Vilfredo Pareto as a power law distribution [Par73] and has been applied initially to the field of economics [Fam63], and later to describe many other phenomena [PH05].

*Lomax Distribution*

The Lomax distribution, also known as Pareto type II, is a two-parameter long-tail distribution with the density function in equation 3.1.2.

$$p(x) = \frac{\alpha\lambda^\alpha}{(x+\lambda)^{\alpha+1}} \tag{3.1.2}$$

where $\alpha$ is called the shape parameter and $\lambda$ is the scale parameter.

It is effectively a Pareto (Type I) distribution that has been translated to allow for the minimal value (location parameter in Pareto Type I) to be zero.

One of the properties of the Pareto and Lomax distributions is self-similarity, which means that when plotted in a log-log plot, data following a pareto distribution fits in a straight line. Although it can be biased and inaccurate [GMY04], the slope of this straight line can be used to estimate the shape parameter.

Alternatively, maximum likelihood estimation (MLE) can be used to estimate the shape parameter [GMY04]. The maximum likelihood estimator for the shape parameter of the Lomax distribution leads to an expression without a closed form, that can essentially only be solved numerically using a computer.

*Cauchy Distribution*

The Cauchy distribution is a long-tail distribution described by the density function in equation 3.1.3.

$$p(x) = \frac{1}{\pi\gamma(1+(\frac{x-x_0}{\gamma})^2)} \tag{3.1.3}$$

where $\gamma$ is called the scale parameter and $x_0$ is the location parameter. This distribution is famous for its undefined mean and variance [Blo58, JKB95]. The scale parameter of the Cauchy distribution can be estimated by computing half of the interquartile range (IQR) as an alternative to MLE. Alternatively, the mean of the absolute values can be used if the location parameter is zero [Kra05].

*Laplace Distribution*

The Laplace distribution (sometimes known as the two-sided exponential) is a long-tail distribution described by the density function in equation 3.1.4.

$$p(x) = \frac{1}{2\alpha} exp(-\frac{|x-\mu|}{\alpha}) \tag{3.1.4}$$

where $\alpha$ is called the scale parameter and $\mu$ is the location parameter.

The scale parameter of the Laplace distribution can be estimated by the MLE. The MLE for the scale parameter of the Laplace distribution is given in equation 3.1.5 for $x_1, \ldots, x_N$, a given sample.

$$\hat{\alpha} = \frac{1}{N} \sum_{i=1}^{N} |x_i - \mu| \tag{3.1.5}$$

*Normal-Inverse Gaussian Distribution*

The Normal-Inverse Gaussian is a four-parameter distribution that is defined through a combination of the Gaussian (normal) distribution and the Inverse-Gaussian distribution [BN77]. Because it has four parameters, it can fit many different types of datasets with high accuracy. However, the interpretation and justification of its use is difficult.

## 3.2 Simulations of Early Learning

In our analysis, simulations of trajectories that aim to replicate the real fish trajectories using the distribution of curvature have been somewhat successful. The idea is to generate a stochastic process (see section 3.1.1) for curvature and use the fundamental theorem of planar curves [Pre10] to construct a two-dimensional curve based on the randomly generated curvature. However, once again, this theorem applies for continuous functions. We derived an analogous version of the theorem using sums instead of integrals. It is given in equations 3.2.1 and 3.2.2 for a collection of given $k_i$.

$$\psi(t) = \sum_{i=0}^{t} k_i \tag{3.2.1}$$

$$\gamma(s) = (\sum_{i=0}^{s} cos\psi(t), \sum_{i=0}^{s} sin\psi(t)) \tag{3.2.2}$$

Equipped with our discrete analogue to the fundamental thereom of planar curves, we identify the best theoretical, well formulated distribution that fits our curvature data. As is typical in determining an underlying distribution for a problem with real data, we aim to balance simplicity of the theoretical distribution with goodness of fit to the data. In this section, we explore various distributions that could represent our data well and produce simulations based on these distributions. We then give a visual interpretation of goodness of fit based on the resemblance of the simulations to the real data.

The theoretical distributions we have considered are the following parametric statistical distributions (described in section 3.1.2): Pareto Type I, Pareto Type

II (Lomax), Normal-Inverse Gaussian, Log-normal, Cauchy, Laplace, Gaussian. Of this list (which is composed of the most common long-tail distributions and the Gaussian distribution), there are a few statistical distributions that do not align with our observations. For example, we have observed points of zero curvature, thus our statistical distribution must accept zero values. This is not the case for the Pareto Type I distribution. Indeed, the Pareto Type I requires as a parameter the minimum value of the distribution. This parameter is required to be strictly greater than zero. An arbitrary choice just above zero would significantly change the distribution. As a result, we can already rule out the Pareto Type I distribution. In this particular case, we have added an alternative. We have added the similar Pareto Type II (also known as the Lomax distribution) which is in principle the same as Pareto Type I but is designed to allow a minimum value parameter of zero. This is done by horizontal translation of the Pareto Type I. As another example, we know the data to be long-tailed. That is, the histograms produced, as well as the computation of kurtosis indicate a leptokurtic underlying distribution. Thus, leptokurtic (higher kurtosis than Gaussian) distributions are acceptable but not platykurtic (lower kurtosis than Gaussian) or mesokurtic (Gaussian kurtosis) distributions. This then right away eliminates the Gaussian from our choices of appropriate models. The last comment to make on this list of distributions is regarding the log-normal distribution. The log-normal distribution has support starting at (and excluding) zero. This would rule it out just like the Pareto Type I distribution, however, it is still kept and fitted to the data. This is the case for two reasons. The first reason is that the parameter choice does not change the tail as much as the Pareto Type I, meaning that we can choose a parameter that would yield a log-normal with minimum value close to zero. The second reason is due to its simplicity and applicability.

*Log Normal-Inverse Gaussian and log Normal fits*

We start off this data fitting analysis with the most complex statistical distribution in our list: The Normal-Inverse Gaussian distribution. It is complex because of the number of parameters that need to be fitted. It has four parameters. Using the R statistical software's *nigFit* function from the *GeneralizedHyperbolic* package, we obtained the four parameters that best fit our data. However, these parameters were impractical, with very small values or very large ones. To get parameters that are more easily useable, we took the logarithm of our data. That is, we excluded all curvature values of zero and computed the distribution of the log of the absolute value of each curvature data point. As a result, the distribution fit would no longer be a Normal-Inverse Gaussian distribution but instead a log Normal Inverse Gaussian distribution. Since we have already taken the log of the data, we also took the opportunity to fit the Gaussian distribution to it. Thus, we fitted both a log Normal-Inverse Gaussian distribution and a log-normal distribution. This is done in Figure 3.1.

*Laplace, Cauchy, and Lomax fits*

The parameters for the Laplace distribution were computed using Maximum Likeli-

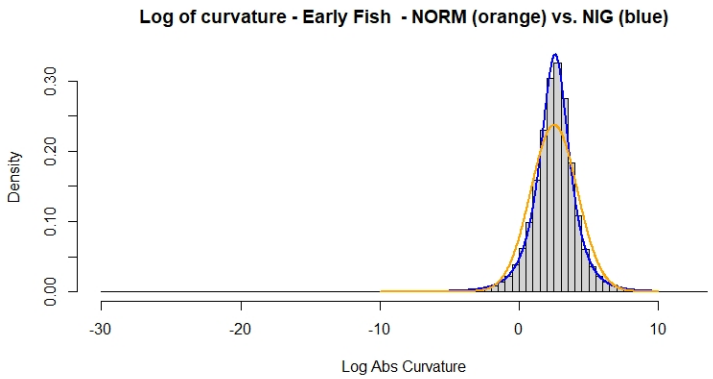


Figure 3.1: **The fitting of a Normal-Inverse Gaussian distribution (blue) and a Gaussian distribution (orange) to the logarithm of the absolute value of curvature for all (amalgamated) early fish.** Note that points of exactly zero curvature were removed to allow the application of the logarithm. Note the near perfect fit of the Normal-Inverse Gaussian distribution (blue). This is not surprising as the distribution has four parameters. The parameters obtained by the $nigFit$ R function were tail heaviness=0.54, location=2.48, scale=1.46, asymmetry=-0.029. The Gaussian fit (orange) is relatively not as good, but adequate given that only two parameters are used. The parameters obtained by computing the mean and the standard deviation were mean=2.53, sd=1.68.

hood Estimation (MLE). The MLE is a technique to estimate the parameters as to maximize the probability of getting the observed data (See [HHTZ18]). Because we assume the location parameter to be zero, the estimation of the scale parameter is the mean absolute deviation. This can also be called the absolute average of the data. The obtained scale parameter was 88.06. Note that this fit was done on the original data, not on the logarithm of the data. For the Laplace distribution as well as all distributions fitted on the original (non-logarithmic) data, we omit the histograms as they give no information on goodness of fit (being composed of a single bar and an otherwise empty histogram).

The Cauchy distribution has two parameters, one of which is a location parameter which we can simply set to zero. We thus only need to estimate the scale parameter of the distribution. As the MLE involves finding the root a very high degree polynomial, we instead opted for the more straight-forward methods of half the IQR, and (since the location parameter is zero) the median of the absolute value of curvature (See section 3.1.2). The estimates return 13.1560 and 13.1534 for half IQR and median of the absolute value, respectively. We thus use the estimated parameter value of 13.15 for the Cauchy distribution.

The Lomax distribution has two parameters, but neither of these is a location parameter that we can simply set to zero. Using the MLE equates to solving an equation without an exact closed form [GFG13]. It can be solved numerically through the use of the R function *flomax* from the package *Renext*. Doing so produces a shape parameter of 1.45, and a scale parameter of 21.31.

*Simulation Framework*

Care was taken to reproduce the aquarium in a virtual setting, so that the simulations would look more realistic and be more comparable visually. The main components of the simulation framework are the threshold to detect the food source (the target), the walls, and the step size. The threshold to detect food is relatively straight forward, we can base it on the data. It was previously reported [JLM14] that the fish detected the food when it was within 4 cm. Thus, we used this value as our threshold. Once the virtual agent reaches within 0.04 units of the target location (set to the origin instead of (0,-0.1) for simplicity) we end the simulation.

As for the walls, we had to make sure the virtual agent did not go beyond the wall limits. In order to do so, we established that if the agent was 0.75 or more units away from the origin, we undid the last movement and generated a new value for the curvature at that point. Very often, the new generated value would result in another "wall hit", i.e., a position 0.75 units away from the origin. However, after enough attempts, the curvature would be generated so that the virtual agent would face away from the wall and find itself able to continue its trajectory.

The step size was initially set to be 0.001, a rounding of the average step size in the real data (0.000 961 m). However, at this granularity, all simulations were overly dense and effectively looked like Gaussian noise (Figure 3.2). As a result, we freed

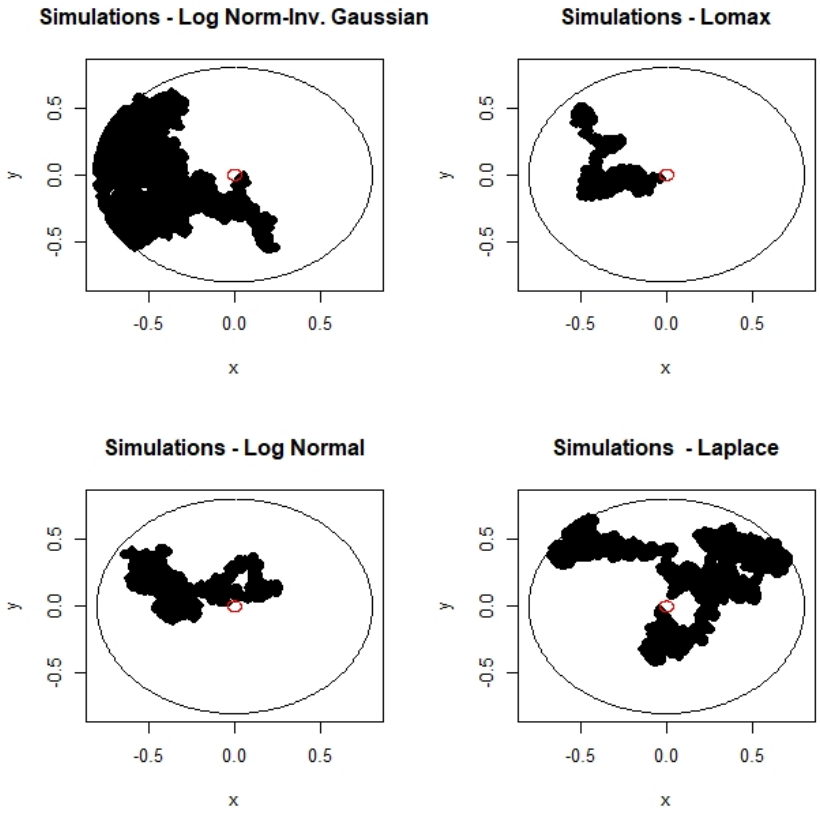


Figure 3.2: **Various simulations with data supported step size of 0.001.** Note the density and lack of resemblance with the original trajectories. Seed was set to 123 for each simulation.

the restriction to rely on the data for the step size and instead aimed to maximize the resemblance to the real trajectories visually. This led to much larger value of 0.05.

Other less important details were implemented such as the starting location which was set to be (0.43, 0.43) or any equivalent position $(\pm 0.43, \pm 0.43)$ and the initial curvature which was set to 0. Finally, a limit of 1 million steps was added to ensure the simulation would run in a set amount of time.

*Early Fish Simulations*

Figure 3.3 shows four simulations at step size 0.05 for the Log Normal distribution. Figure 3.4 shows four simulations for the log Normal-Inverse Gaussian distribution. Figure 3.5 shows four simulations for the Laplace distribution. Figure 3.6 shows four simulations for the Cauchy distribution. Figure 3.7 shows four simulations for the Lomax distribution.

Based on these examples (Figures 3.3 to 3.7), we can say that the best visual fit for the real fish trajectories comes from the simulations based on the curvature as

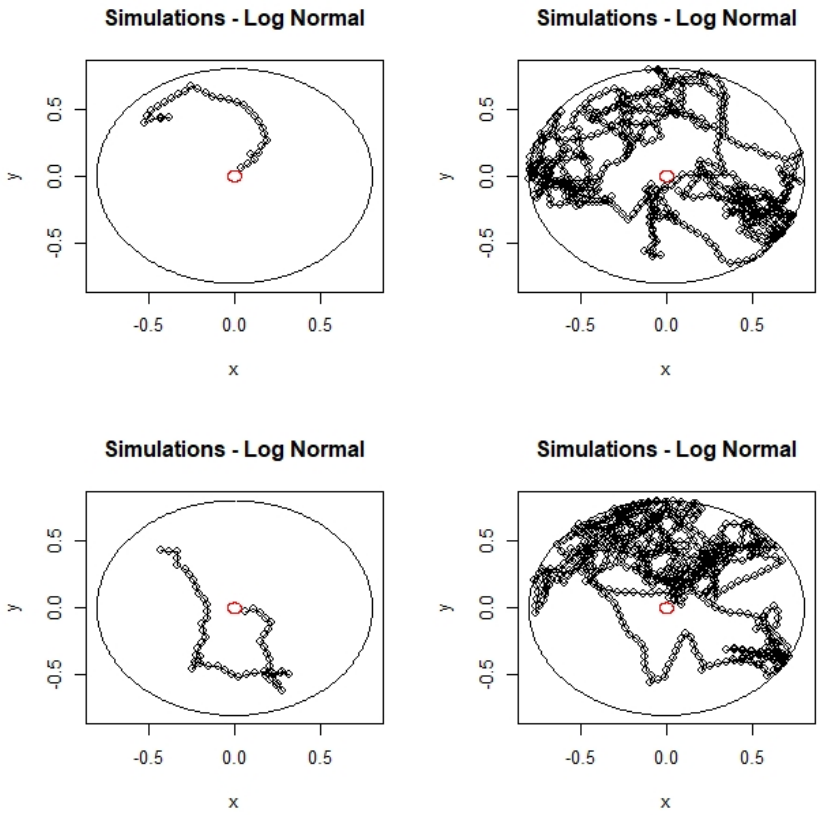


Figure 3.3: **Simulations based on curvature generated by a lognormal distribution with parameters derived from the data and step size of 0.05.** The random seed used here was 123. Note the much-improved resemblance to real trajectories when using the 0.05 step size instead of the 0.001 step size (Figure 3.2). The log normal distribution being one-sided, we used a 50-50 generator to determine the sign of the value at each point, effectively making the distribution symmetric and two-sided.

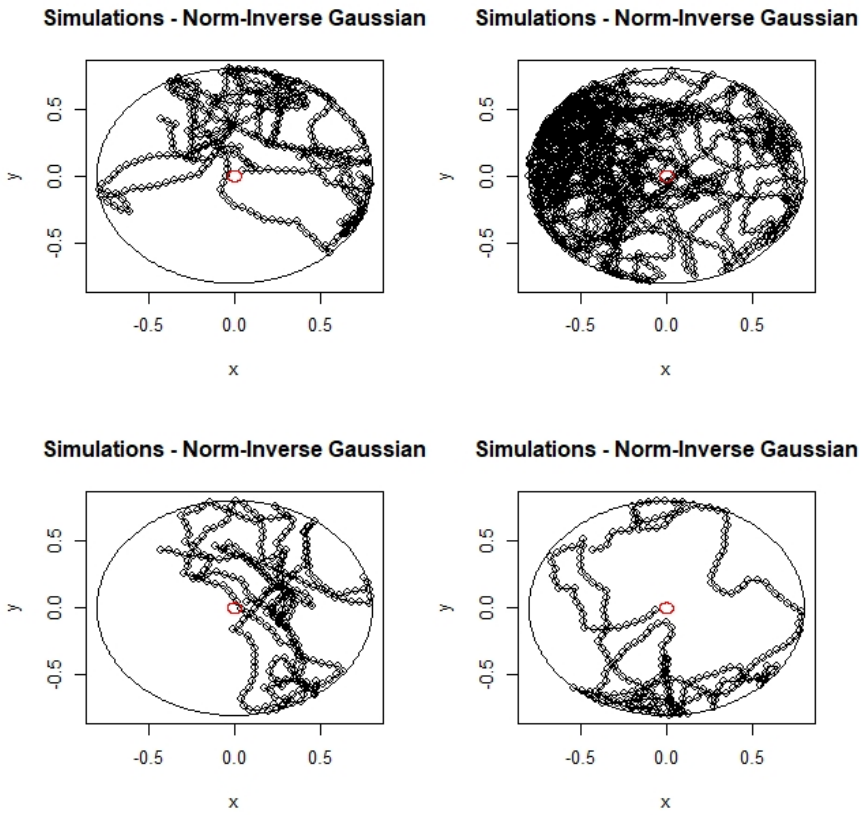


Figure 3.4: **Simulations based on curvature generated by a log Normal-Inverse Gaussian distribution with parameters derived from the data and step size of 0.05.** The random seed used here was 123. The log normal-inverse Gaussian distribution being one-sided, we used a 50-50 generator to determine the sign of the value at each point, effectively making the distribution symmetric and two-sided. In this case, the trajectories are a bit sharp, with the duration of zero curvature lines a bit longer than the regularly observed lines in the real data.

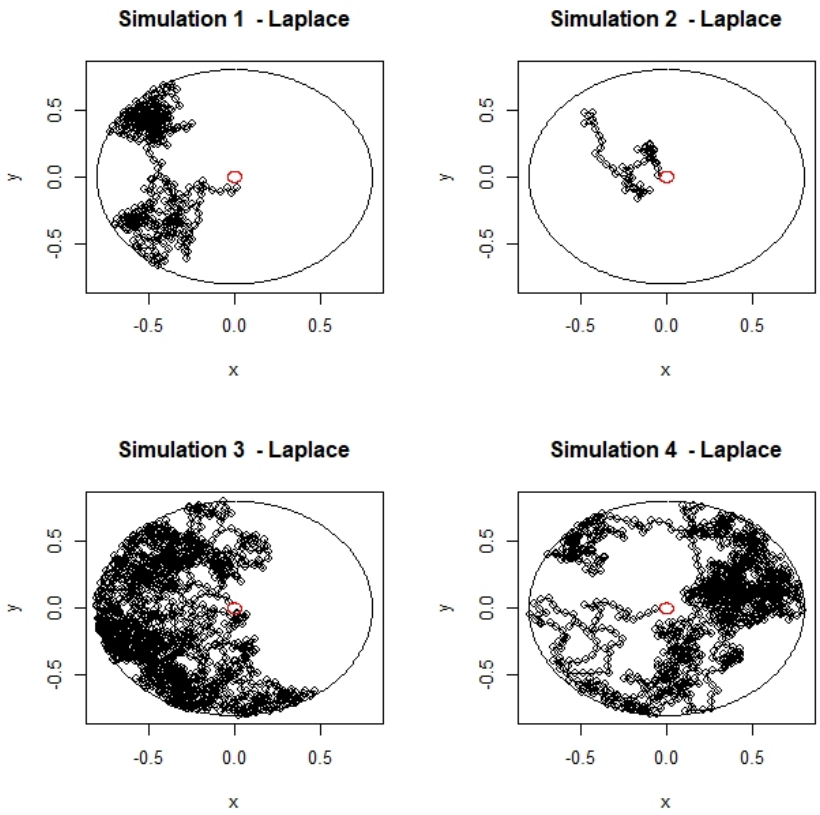


Figure 3.5: **Simulations based on curvature generated by a Laplace distribution with parameters derived from the data and step size of 0.05.** The random seed used here was 123. In this case, the trajectories are a bit too noisy, with the duration of zero curvature lines much shorter than the regularly observed lines in the real data.

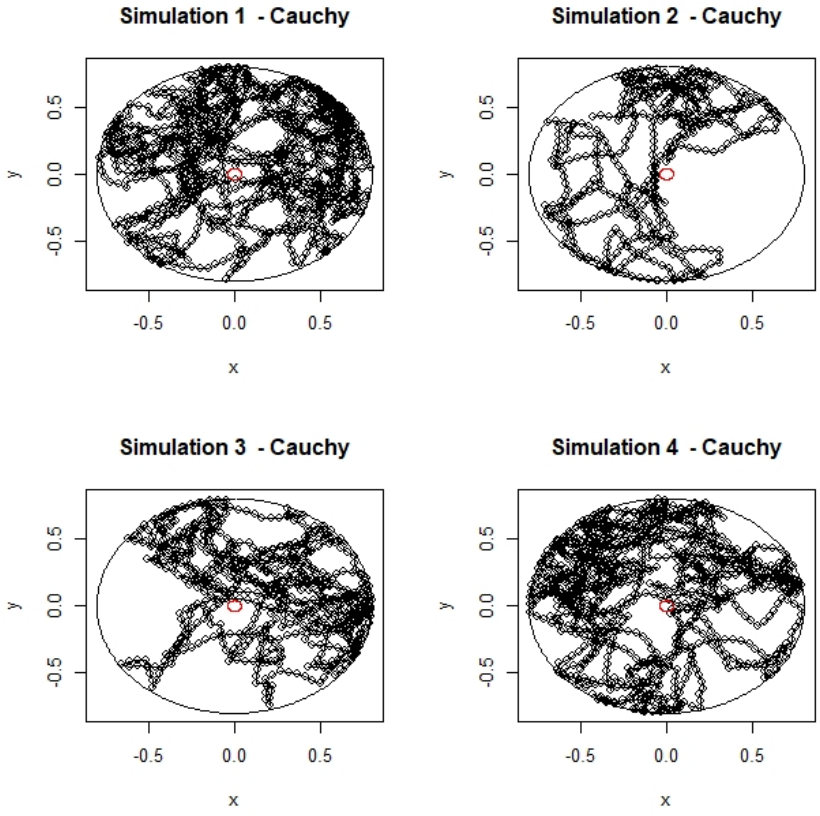


Figure 3.6: **Simulations based on curvature generated by a Cauchy distribution with parameters derived from the data and step size of 0.05.** The random seed used here was 123. In this case, the trajectories are a bit sharp, with the duration of zero curvature lines a bit longer than the regularly observed lines in the real data. They are even less smooth than the log Normal-Inverse Gaussian distribution given in Figure 3.4.

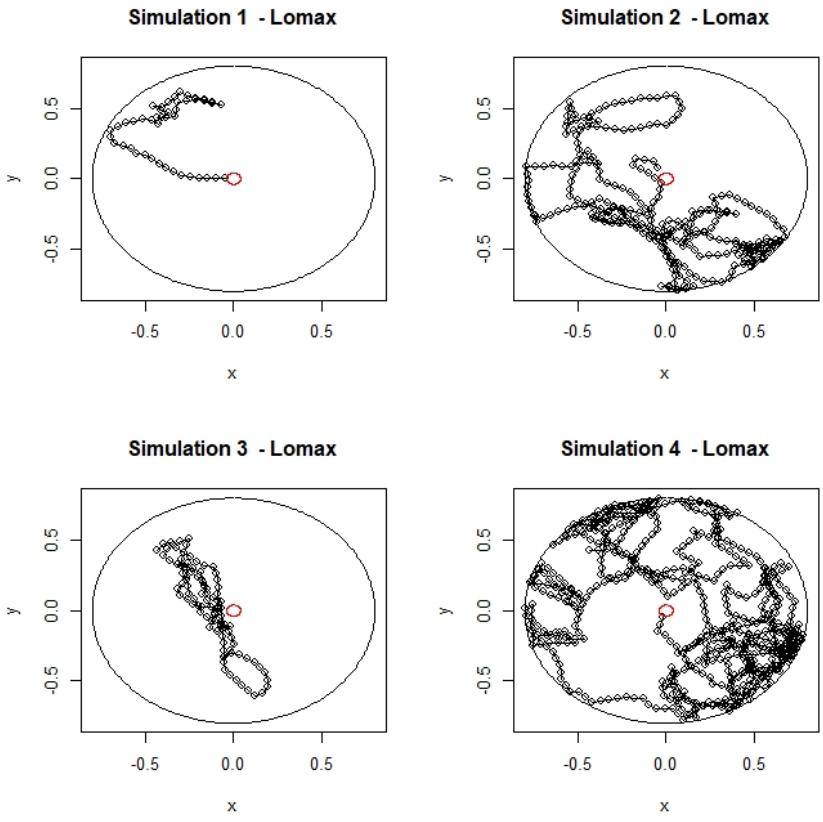


Figure 3.7: **Simulations based on curvature generated by a Lomax distribution with parameters derived from the data and step size of 0.05.** The random seed used here was 123. The Lomax distribution being one-sided, we used a 50-50 generator to determine the sign of the value at each point, effectively making the distribution symmetric and two-sided. There is a generally good visual resemblances with the trajectories of the real fish data.

generated by the Lomax distribution. The simulations are neither too sharp, nor too noisy. They are somewhat smooth, resembling the real fish trajectories. A comment can be made on its smoothness, however. The real trajectories are smoother (in the sense of potential derivability) than the simulations. The autocorrelation function (ACF) of the real fish data reveals that this is indeed the case (Figure 3.8). This autocorrelation can be thought of a certain type of inertia, with the fish having a certain continuity (or smoothness) to its curvature.

As a way to artificially improve the ACF of the simulations (which are flat, since they are generated randomly) and simultaneously smooth the simulated trajectories, we used a moving average with span 5 (MA5). The obtained ACF computed based on MA5 produced a lag 1 and 2 correlation similar to that of the real data. Note that we also considered using binomial filter smoothing, but since we obtained similar results as the simpler moving average, we opted for the moving average. The span of 5 was used because it yielded the right autocorrelation function. A larger span would create autocorrelation at lags of values too high (lag 4 or 5, for example) and a shorter span (3) would not return any autocorrelation at lag 2. Our final simulation design for early learning was thus a Lomax distribution smoothed out by a MA5. A few examples of this are illustrated in Figure 3.9.

## 3.3 Simulations of Late Learning

In this section, we briefly talk about efforts towards producing simulations of late learning fish. The main idea was to introduce a bias towards the location of the food. The bias would range from 0 to 1, with 0 being equivalent to an early learning simulation, and 1 being a straight trajectory towards the food. We note that this model is already unrealistic since real late learning fish trajectories are not straight paths towards the food and instead persist in their randomness even after learning has ceased to improve in terms of duration of search [JLM16]. The model is given by the equations 3.3.1 and 3.3.2.

$$\omega(i) = (1-\lambda)\psi(i-1) + \lambda(\phi(i-1)) \tag{3.3.1}$$

$$(x(i), y(i)) = (s \times cos(\omega(i)) + x(i-1), s \times sin(\omega(i)) + y(i-1),) \tag{3.3.2}$$

where $\psi(i) = \psi(i-1) + X_i$, with $X_i \sim Lomax$. $\lambda$ is the inputted bias (ranging from 0 to 1), and $\phi$(i) is the optimal angle from the current direction to the target (food). For equation 3.3.2, $s$ is the step size (0.05).

An example of how the bias impacts the trajectories is given in Figure 3.10. The model is not great. It is too noisy and it is not realistic. Effectively, the two portions of equation 3.3.1 are fighting each other in the sense that the bias pulls the agent

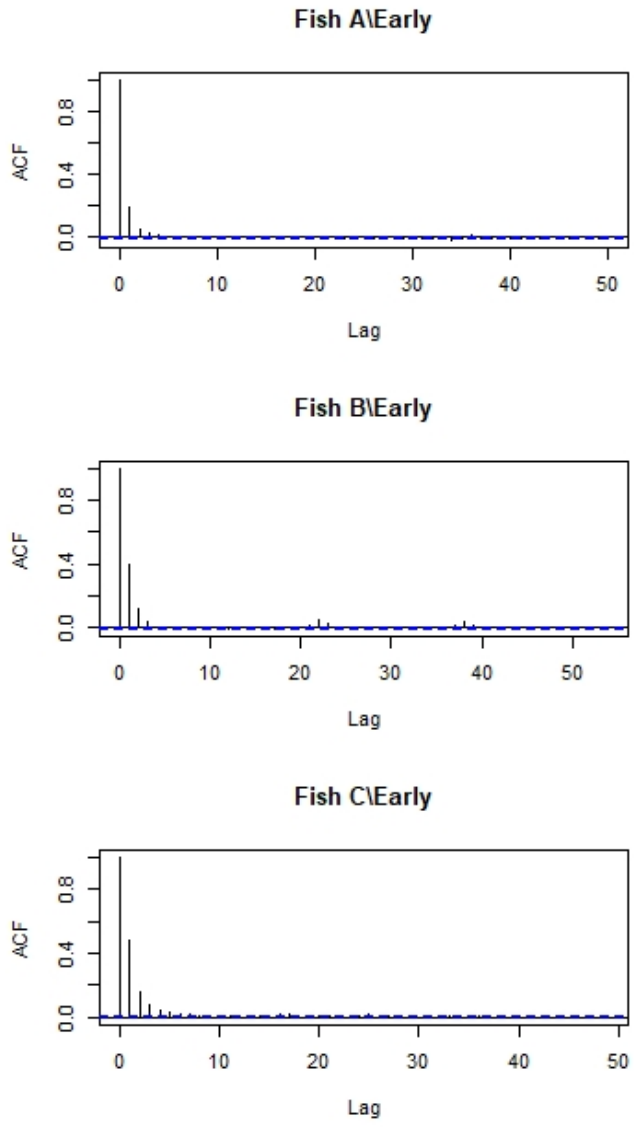


Figure 3.8: **Autocorrelation function of the osculating curvature of the early fish.** Note that there is some correlation for lag 1 and 2 in all early fish.

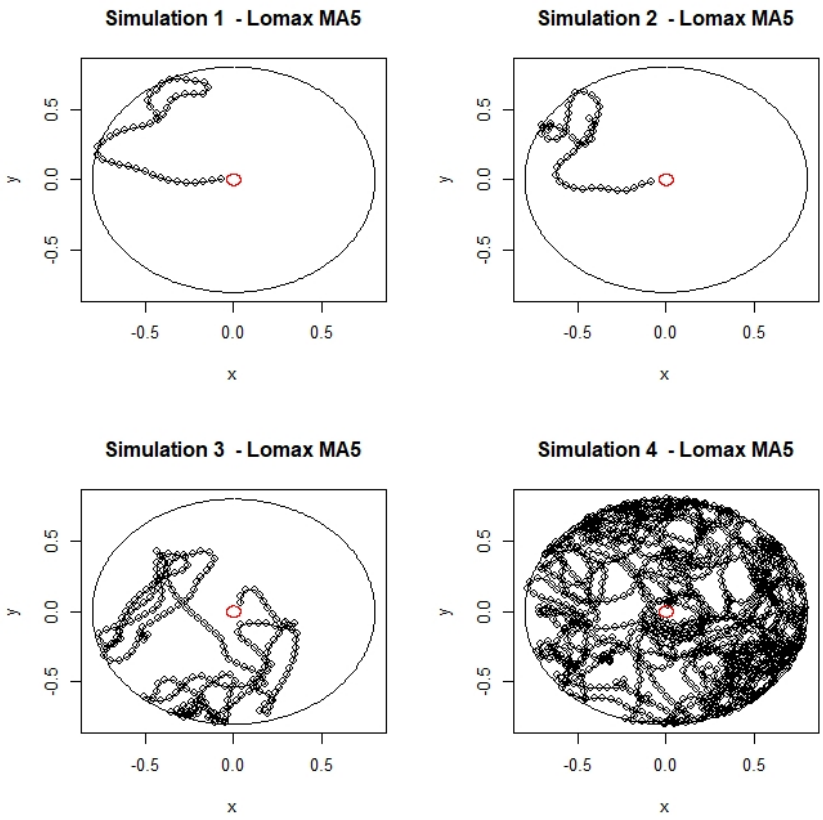


Figure 3.9: **Simulations of fish trajectories with curvature based on a Lomax distribution smoothed with moving average 5.** The step sizes are 0.05. The seed used was 123. Note the smoothness of the resulting trajectories and the resemblance to real trajectory data.

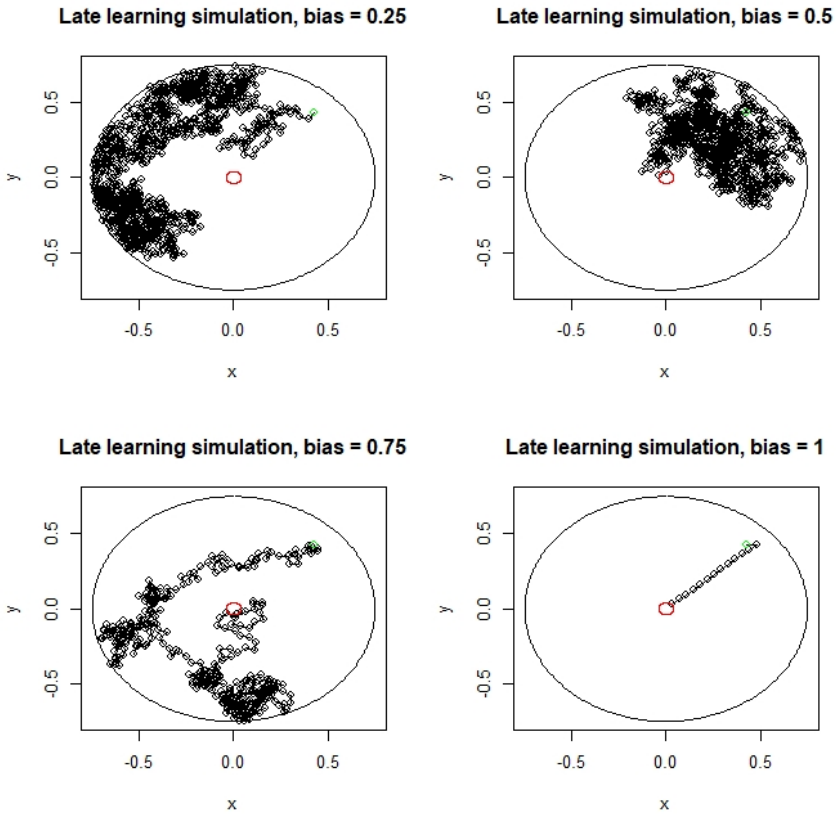


Figure 3.10: **Late learning simulations with varying bias.** Step size is 0.05. No smoothing has been applied. Seed was set to 123. Note the direct path to the food when the bias is 1, but also the unrealistically noisy movement of the biased trajectories.

towards the food more like gravity than like a neurological bias. For example, in Figure 3.10 at bias 0.75, we can see occasional "twitching" towards the food in what would otherwise be a relatively smooth simulated trajectory.

In conclusion, despite the realistic early learning simulations based on stochastic processes, the late learning model is inaccurate in multiple ways. Most importantly, it does not properly simulate the learnt fish trajectories, thus we know that the underlying neural basis for learning cannot be modelled by equations 3.3.1 and 3.3.2. As a more promising approach, future works can focus on models that modify the generating distribution over time according to the findings in Chapter 4. In accordance with these findings, the distribution would be generating turning angles instead of curvature.

# Chapter 4

# Head Angle

The head angle (also referred to as head orientation) is a value ranging from 0 to 360 identifying the orientation of the head of the fish, where 0 and 360 indicates the fish is heading perfectly to the right. It increases counter-clockwise and must be interpreted modulo 360, therefore, a value of 90 indicates that the fish is going up (or north), and a value of 270 indicates the fish is going down (or south). Keeping in mind that the water the fish is swimming in is very shallow, we do not consider the depth the fish is swimming at and thus have no data on the angle with respect to depth.

## 4.1 Change of Head Angle

The change of head angle is defined as the angle difference between the head angle at step n and the head angle at step $n+1$. For example, if step 11 of the fish trajectory has for head angle 275 and the head angle at step 10 (0.01 seconds prior, in our case) was 270, then the change in head angle is 5 degrees. The computed change in head angle varies from -180 to 180, with the two extremes being the same. In this section, we cover the differences between early and late learning fish in change in head angle. We determined the distribution of change of head angles for early and late learning; outliers of this distribution were taken as being 1.5 times the IQR away from the 1st or 3rd quartile [BL11]. These outliers were defined as "sharp turns" and we then calculated the time intervals between the sharp turns.

### 4.1.1 Effects of Spatial Learning on Change in Head Angle

The first step in our analysis was to derive distributions of the change in head angle for each fish and learning status (Early vs. Late learning). This way we could spot any obvious differences and direct our analysis in terms of what the next steps would be. The histograms of change in head angle for each fish and learning status are given in Figure 4.1.

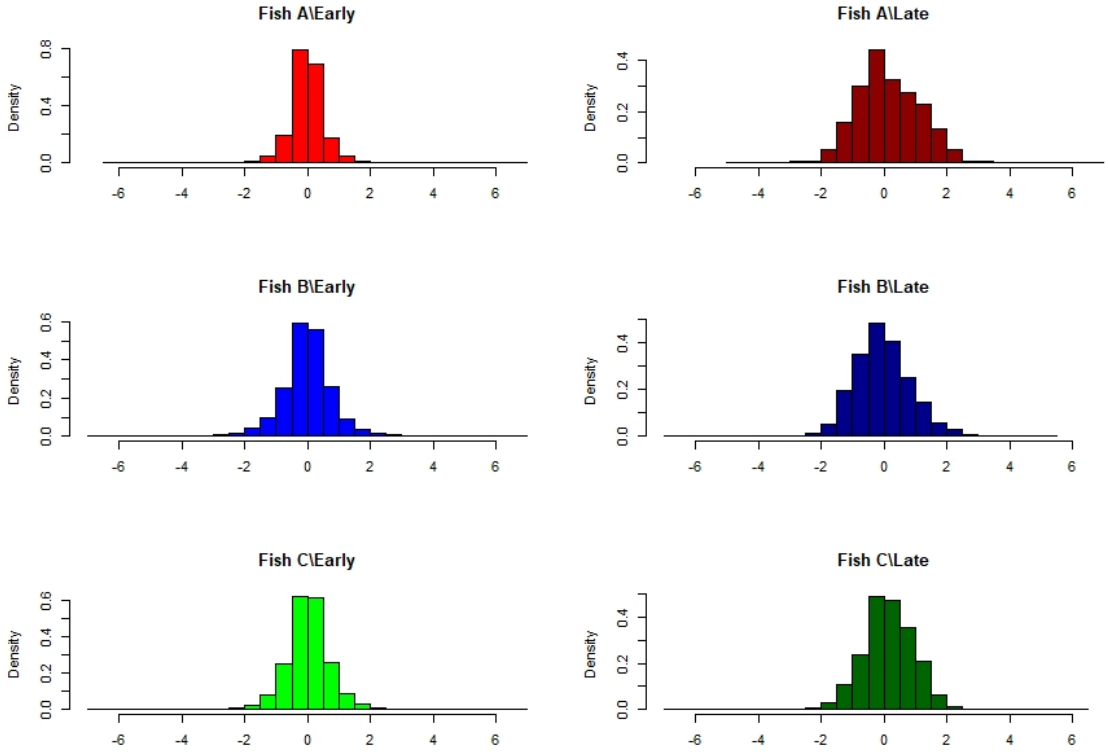


Figure 4.1: **Histograms of change in head angle for each fish and learning status.** Each fish and learning status had its trajectories amalgamated and a histogram was produced from this amalgamation. The shape of the distributions on the left (Early) differs from the shape of the distributions on the right (Late). On the x-axis, the change in head angle (rarely going over 3 degrees).

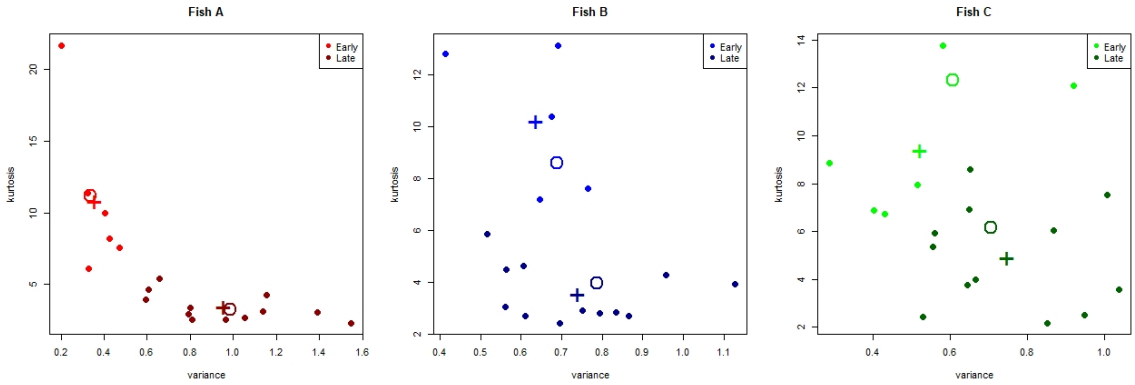


Figure 4.2: **Variance and kurtosis of the distributions of change in head angle for all trajectories.** Each point is a trajectory whose variance and kurtosis in change in head angle has been computed and makes the x and y coordinates, respectively, in the plot. The plus sign is the centre of mass (average) of all the points of the same colour. The circle is the variance and kurtosis of all trajectories of one fish and learning status (i.e., one colour) when all the trajectories have been amalgamated. It can also be thought of as a weighted centre of mass with the weights being proportional to the length of each trajectory. Note the separation between early and late learning fish for all three fish, some more clear than others.

We looked for statistics that would capture the differences in the histograms in Figure 4.1. Since the mean seems to be the same for all fish and learning status (it is 0), we considered the variance and kurtosis. In order to compare each group, we computed the variance and kurtosis for all trajectories individually and plotted them into 3 (one for each fish) 2-dimensional figures (Figure 4.2) with one axis as the variance and the other as kurtosis. We could then see if clusters were formed. The clusters would indicate consistent differences between late and early learning fish.

Although there is somewhat of a separation between the two learning statuses, Figure 4.2 does not illustrate clear clusters. This indicates that the variance and kurtosis may not be the exact statistics we need to find differences between early and late learning. We opted, then, for a non-parametric approach. We instead used the IQR and the percentage of the trajectory data that was considered an outlier with respect to the standard outlier definition (1.5 times away from the 1st or 3rd quartile), henceforth referred to as percent outliers (% outliers). The idea here was to replace variance by its non-parametric analogue, the IQR, and kurtosis by its non-parametric analogue, the % outlier. The reason why kurtosis was replaced by a statistic based on the outlier is simply because kurtosis is heavily influenced by outliers. Indeed, it is much more influenced by outliers than variance, being a quartic of the difference

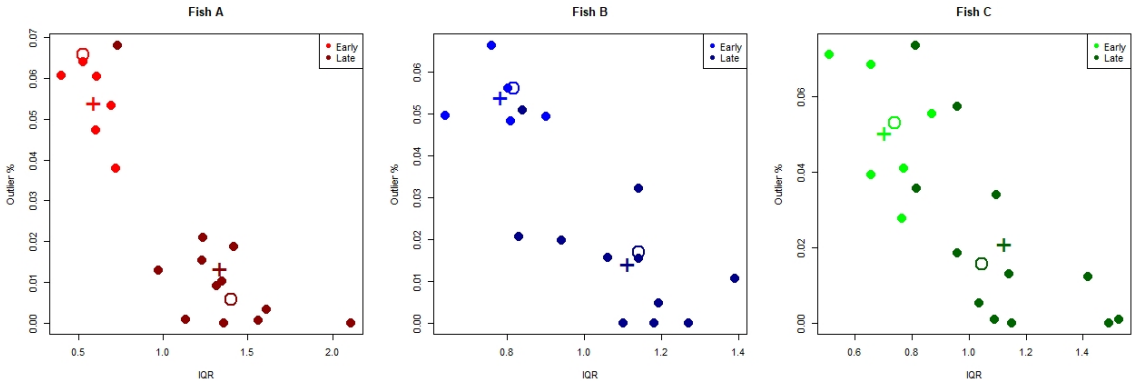


Figure 4.3: **IQR and % outliers of the distributions of change in head angle for all trajectories.** Each point is a trajectory whose IQR and % outliers in change in head angle has been computed and makes the x and y coordinates, respectively, in the plot. The plus sign is the centre of mass (average) of all the points of the same colour. The circle is the IQR and % outliers of all trajectories of one fish and learning status (i.e., one colour) when all the trajectories have been amalgamated. It can also be thought of as a weighted centre of mass with the weights being proportional to the length of each trajectory. Note the nearly perfect separation between early and late fish A and B, and an acceptable separation for fish C.

between the mean and each point instead of the less sensitive quadratic difference. Generally, kurtosis is a statistic describing tailness [Wes14], which is composed of outliers in the case of long-tailed distribution. Figure 4.3 shows the same plot as Figure 4.2 with variance and kurtosis replaced by IQR and % outliers, respectively.

The clusters are clearer when using IQR and % outliers than when using variance and kurtosis. We visibly see that early learning trajectories nearly consistently have lower IQR and higher % outliers. We then thought of using boxplots, which encompass the IQR and outliers within them, to continue our comparison. The boxplots are given in Figure 4.4.

Figure 4.4 confirms that the median remains relatively unchanged at zero. The IQR is consistently higher in late fish when compared to early learning fish. The number of outliers is much greater for early learning fish as compared to late learning fish. One interpretation is that the fish make far more sharp turns during early versus late learning. In other words, after learning, the fish makes relatively more sustained smooth changes in their head angle. We note that the size of the datasets between fish and between learning status vary a lot and so a more appropriate statistic is percentage outliers instead of number of outliers, which is shown in Figure 4.3. Numerically, the percent outlier is 6.58%, 5.61%, and 5.29% for early learning fish

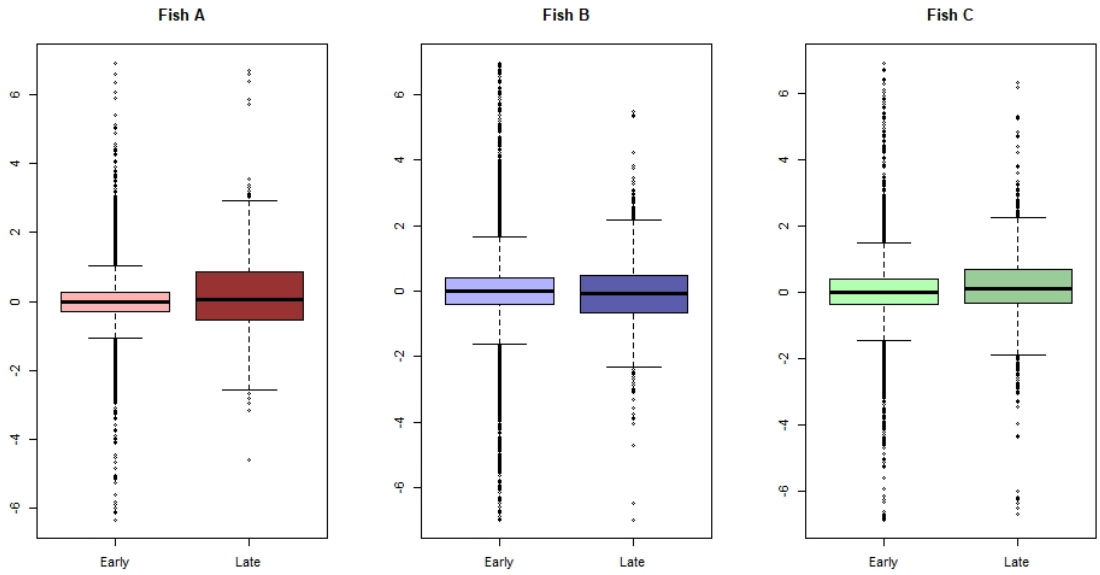


Figure 4.4: **Boxplots of change in head angle for the amalgamated trajectories of each fish and learning status.** Comparing separately each fish (A, B, and C) for their early learning change in head angle versus their late learning change in head angle. For the change in head angle, we look at the angle between where the fish was facing 0.01 second ago, and where it is facing now. Note that the interquartile range is consistently larger for late learning fish than for early learning fish. Furthermore, the number of outliers in early learning fish is very much higher than the number of outliers for late learning fish. The outlier thresholds are indicated by bars. This is consistent with the findings in Figure 4.3.

A, B, and C, respectively, and 0.57%, 1.69%, 1.56% for late learning fish A, B and C, respectively. Thus, when considering the number of outliers and the percent of outliers, we get the same results: outliers, i.e., very sharp turns, are more present in early learning than in late learning. We may interpret these to give them meaning in terms of swimming motion and behavioural state of the early fish as compared to the late fish. For both early and late learning fish, the median is nearly 0, indicating that the fish turns equally left and right. The IQR is larger for the late learning fish, meaning the centre of mass of the distribution (the bulk central 50%) is more widely distributed after learning has occurred (Figure 4.4). However, there are more outliers (i.e., sharp head angle turns) before the learning has occurred (Figure 4.4). We interpret this as a more relaxed swimming behaviour for the late learning fish, allowing more turns but reducing the number of very sharp turns.

### 4.1.2 Intervals Between Sharp Turns

We have previously defined sharp turns as outliers in the change in head angle. We may consider the occurrence of these sharp turns in the early learning fish to study the foraging behaviour of these fish before they have undergone spatial learning. The first question we asked was "Are the intervals between these sharp turns random?". As is ordinary, we computed a histogram to potentially identify the distribution of time intervals between sharp turns. This histogram is given in Figure 4.5 for fish A, B, and C, all in early learning phase.

We considered the exponential distribution because it models the interval between two random events. Since we have a good fit for the exponential distribution (Figure 4.5), we have evidence to say that the time interval between sharp turn is random. That is, the foraging fish initiates a sharp turn randomly in terms of time. This was further supported by the lack of lag 1 serial correlation (autocorrelation) for fish A and C, and a weak lag 1 serial correlation for fish B (0.19).

Similar results were found when distance replaced time. Figure 4.6 shows an exponential fit to the distance interval between sharp turns. It shows that the exponential distribution is once again a good fit. The serial correlation at lag 1 of fish A is statistically not significant while fish B and C have a very weak positive serial correlation (0.14 both). Thus, from both perspectives (time and distance) the fish seem to initiate sharp turns randomly.

The lag 1 serial correlation of speed was computed and found to be 0.65, 0.79, and 0.76 for fish A, B, and C, respectively. This indicates that the foraging fish swam at fairly constant speeds and explains the similarity in the results for time and distance intervals between sharp turns. We also considered whether the sharp turns were made around the landmarks simply to avoid them when detected. As it turns out, there were not more sharp turns near the landmarks (<4cm away) than there were away from landmarks, in open water. For early learning and across all three

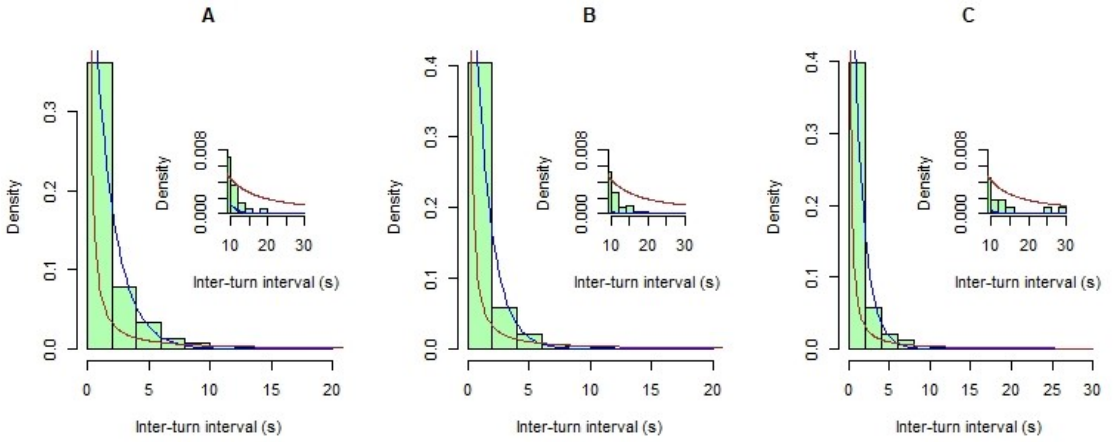


Figure 4.5: **Distribution of time intervals between sharp turns in early learning fish.** Each histogram has a power law fit (Pareto with minimum value 0.01 and shape parameter given by MLE, in brown) and an exponential fit (parameter given by MLE, in blue). The exponential parameter was 0.60, 0.77, and 0.74, and the power law parameter was 1.25, 1.25, and 1.27 for fish A, B and C, respectively. Note that the fit of the exponential is good ($\ell_2$ distance 0.04, 0.05, and 0.05, for fish A, B, and C, respectively), indicating the model distribution may be accurate, which in turn could indicate that the process is random. The fit of the power law distribution is acceptable ($\ell_2$ distance 0.29, 0.33, and 0.32, for fish A, B, and C, respectively). Each histogram has an inset that displays the tail of the distribution. **A** - Early fish A. **B** - Early fish B. **C** - Early fish C.

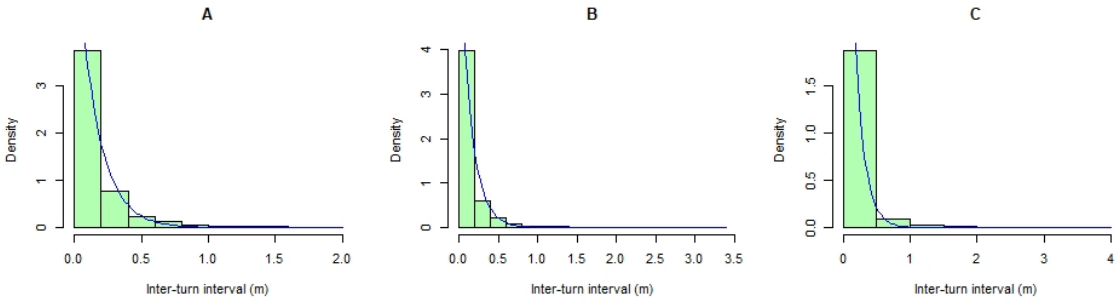


Figure 4.6: **Distribution of distance intervals between sharp turns in early learning fish.** Each histogram has an exponential fit (in blue, parameter given by MLE). The exponential parameters were 6.56, 6.88, and 7.01 for fish A, B, and C, respectively. Note once again, the goodness of fit of the exponential distribution to the early learning fish data.

fish, we found 160 instances of sharp turns near landmarks and 1612 instances of sharp turns in open water. We emphasize that, in the latter case, there is no sensory input of any sort that might have evoked the turn. This suggests that the sharp turns are randomly generated by internal neural dynamics. We conclude that fish foraging may be well modelled by stochastic processes, particularly a Poisson process because the inter-sharp turns intervals were best fit with an exponential distribution, which is equivalent to being a Poisson process.

In terms of features closely linked to neural activity, we compared the EOD rate during sharp turns to the EOD rate when the foraging fish was not making sharp turns. Here, again, sharp turns are defined as any change in head angle greater than 1.5 times the IQR (i.e., a statistical outlier). In the sharp turn period, we included the 150 frames (1.5 seconds) before and after the sharp turn. In the case where there is another sharp turn within that time frame, the two periods are merged. The 1.5 seconds comes from previous research [JLM14] and was found there to be the time difference between an EOD increase (first) and a spontaneous movement (second). Anything that is not defined as a sharp turn period is smooth motion. We found no relationship between EOD rate and change in direction. During sharp turns, the mean EOD rate was 74.4, 73.9, 72.5 pulse/s for fish A, B, C, respectively. The instances not surrounding a sharp turn (smooth motion) had a mean EOD rate of 74.3, 72.7, 72.6 pulse/s for fish A, B, C respectively. When only looking at open water swimming (>4 cm away from the landmarks), EOD rates during sharp turns had means 74.2, 72.8, 72.3 for fish A, B, C, respectively. During smooth motion, the average EOD rate was 74.0, 72.9, 72.8, for fish A, B, C respectively. It was previously reported [JLM14] that there are large increases in EOD rate ($\sim$ 8 Hz) before and during the spontaneous initiation of movement. In contrast, there was clearly no difference in EOD rate between smooth swimming and sharp turns.

## 4.2 Error in Head Angle

Since the location of the food source is constant relative to the entry point of the fish and recorded in our data (coordinates (0, -0.1) for Fish A, (-0.1, 0) for Fish B, and (0, 0.1) for Fish C), we may compute the direct route to the food based on the x, y coordinates of the fish at any time. We may then compute the angle between the fish's current direction and the direct route to the food. We called this the error in head angle, because it represents the degree at which the fish is going away from the food. We designed this value to range from 180 to -180 degrees with 0 indicating that the fish is going straight towards the food source.

### 4.2.1 $\ell_2$ Distance From One Distribution to Another

A measure of distance between two histograms is given by the $\ell_2$ distance. If the two histograms have the same intervals on the x-axis, we simply add up the square of the differences in density in each interval and square root the total. Thus, a large $\ell_2$ distance indicates that the two histograms are very different while a small $\ell_2$ distance indicates that the two histograms are nearly the same. Note that the $\ell_2$ distance depends on a plenitude of data. Indeed, sparse data will have an ill-defined histogram that may not represent the underlying distribution. As a result, the $\ell_2$ distance is more likely to increase due to sparsity and misrepresent a difference between the underlying distributions. The ideal scenario is to have a plenitude of data so that the histograms closely match their underlying distribution.

The $\ell_2$ distance can also be used to measure the difference between a theoretical distribution and a histogram. This is done by evaluating the distribution at the middle value of each histogram interval and using that in the sum of square differences. Particularly, we may define the $\ell_2$ distance from the uniform distribution of a given histogram. This is simply done by taking the maximum and minimum values of the histogram and defining the uniform distribution on that interval, then the inverse of the length of that interval will constitute the values on which the original histogram will be compared.

### 4.2.2 Distribution of Error in Head Angle

As a first step in our study of the error in head angle, we plotted the distribution of error in head angle for each fish and learning status. This time, instead of amalgamating all of the trajectories, we computed the histograms of each trajectory and took an average density for each interval. This way, bias towards the longer trajectories were removed. The histograms are given in Figure 4.7.

The average histograms in Figure 4.7 suggest some consistent differences between early and late learning fish in terms of error in head angle. The bias towards the center of the Late learning trajectories indicates that the fish are foraging slightly differently after learning. We theorized, then, that the late learning fish browsed its environment then eventually decided to go straight to the food. This would be different from the early learning fish which browsed its environment until it randomly found the food source. To test this, we split each trajectory into two halves. That is, we separated the first 50 % and the last 50 % of each trajectory after computing the error in head angle and plotted the average histograms. If our theory was correct, the first halves of the late learning fish would be essentially uniform and the second half would be much more unimodal and centered at zero (indicating the fish was going towards the food most of the time). The average histograms of error in head angle for the second halves of the late trajectories and the second halves of the early trajectories (for comparison) are superimposed in Figure 4.8.

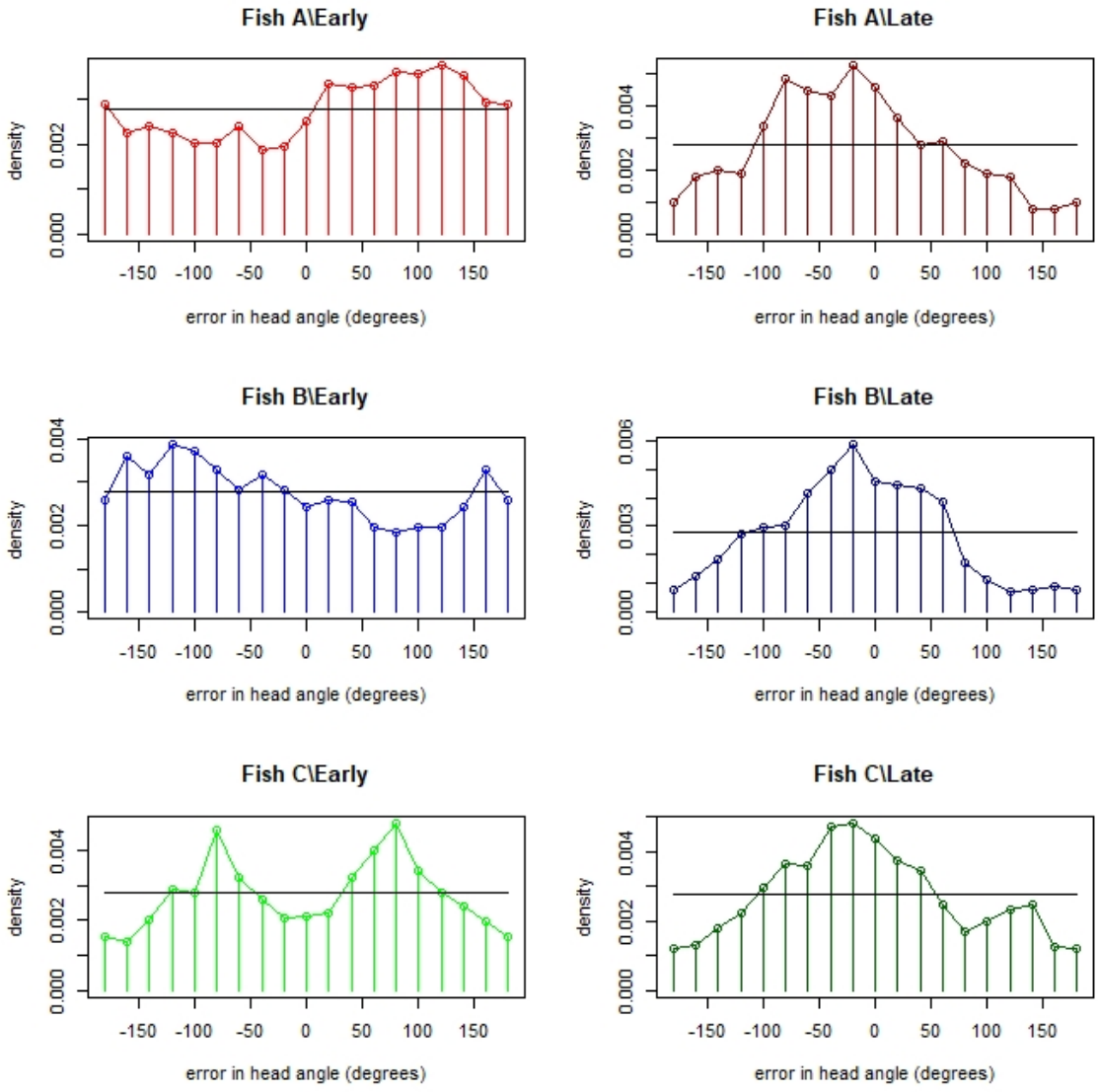


Figure 4.7: **Average histogram of error in head angle.** We computed the histograms of each trajectory and took an average density for each interval. For example, in Fish A Early, histograms were computed for all trajectories and their density were then averaged to produce the density in the red plot (Top Left) in this figure. Note that there are some differences between Early and Late learning fish in the sense that the Late fish is slightly biased towards the food. The horizontal black line indicates the uniform distribution over the interval -180 to 180.

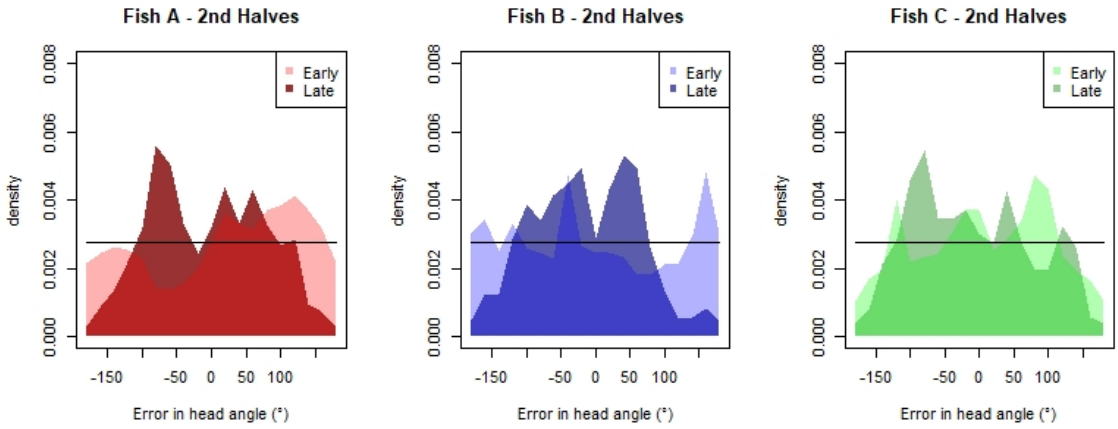


Figure 4.8: **Average histograms of error in head angle for the second halves of trajectories.** For each fish and learning status, the distribution is nearly uniform, indicating that the fish is foraging similarly during the second half of the trajectories whether it be in early or late learning. The theoretical uniform distribution is the black line in each panel.

Our theory was wrong, the late learning fish is not going straight towards the food during the second half of the trajectory. In fact, based on the error in head angle, the second halves of the trajectories are hardly differentiable. They are both nearly uniform, i.e., the fish move randomly with respect to the food source. This was unexpected. Nonetheless, we plotted the first halves of the trajectories in a similar manner as Figure 4.8. This is given in Figure 4.9.

Figure 4.9 is also unexpected, but reveals a difference between early and late learning fish. The first half of their trajectories are different with respect to the food source. The late learning fish generally heads towards the food source in the first half of its trajectory (This is the case for all three fish). However, in the second half of the trajectory, it generally no longer has a significant bias towards the food and behaves indistinctly from the early learning fish with respect to error in head angle. That is, it moves randomly with respect to the food source. This is a big finding and it helped us redirect our analysis towards a strong conclusion.

The next step in our analysis was to split the distributions into finer-grain to see the transformation of the distribution from one that is peaked at zero to a uniform one. Instead of splitting the trajectories into halves, we considered "slices" of time that are both shorter and better defined. We wanted to fix a time length instead of having a percentage of a trajectory. On one side, the shorter the time length of slices, the clearer our analysis would be. On the other, the shorter the time length of slices, the more our histogram deviated from its underlying distribution. After some trial and error to see how short we could have our slices, we settled on the value of

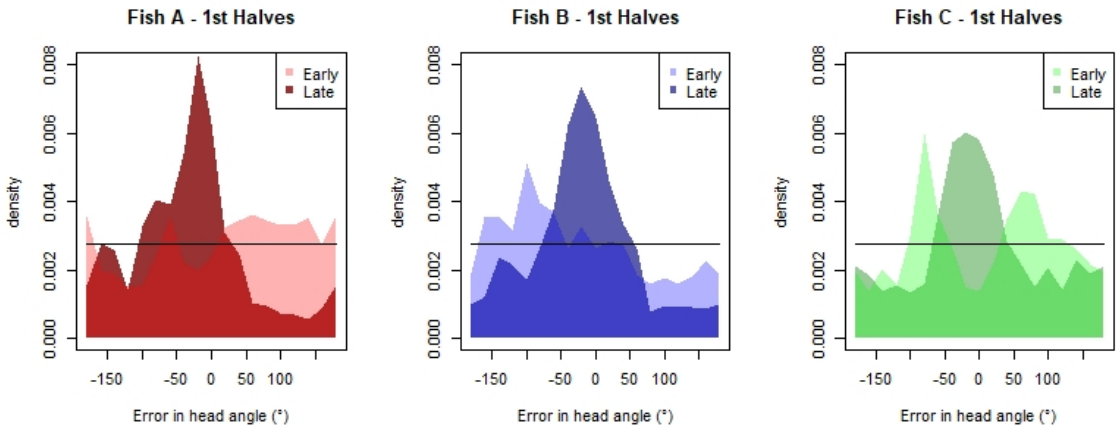


Figure 4.9: **Average histograms of error in head angle for the first halves of trajectories.** Unlike in Figure 4.8, the distributions are not similar. Indeed, the late learning distributions are unimodal with peaks near zero, very different from the uniform distribution (plotted as a black line in each panel) while the early learning distributions are nearly uniform, much like the second halves plotted in Figure 4.8. This indicates that the late learning fish behaves differently from the early learning fish in the first half of its trajectories.

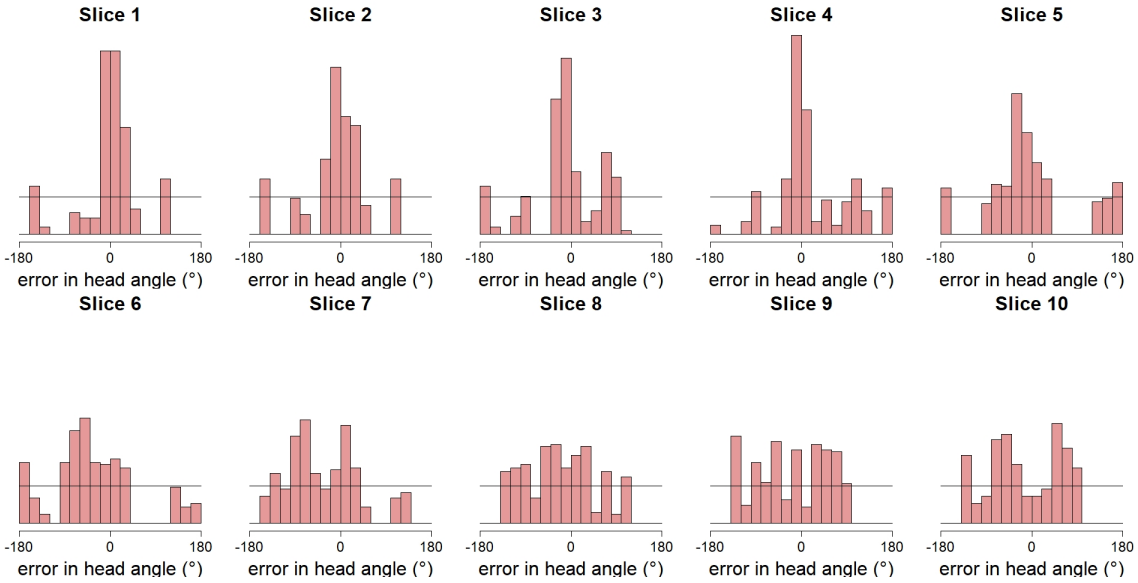


Figure 4.10: **Evolution of the error in head angle distribution over slices (over time) for late learning fish A.** Each histogram is composed of 360 data points (30 per trajectory with 12 trajectories in late learning). Each slice lasts 0.3 seconds. All late learning slice 1 data points were put together to make the histogram labelled "Slice 1". All late learning slice 2 data points were put together to makes the histogram labelled "Slice 2", etc. All trajectories were incomplete (hadn't reached the food) within the time span of the 10 slices. Note the transition from peaked at zero at the first few slices to a nearly uniform distribution at the last few slices. The black line indicates the theoretical uniform distribution.

0.3 seconds per slice. This is 30 frames of data per slice per trajectory. Given that our late learning fish each have 12 trajectories, this is a histogram of 360 data points per slice for the late learning fish. Our focus was at that time on late learning fish, as they were the only ones with a difference in their first and second halves of error in head angle distributions. We will cover the evolution of the distribution of early learning later in this section. As an example of evolution of the distribution of late fish, Figure 4.10 shows the first 10 slices (each of 0.3 seconds and composed of 360 data points) of late learning fish A.

From Figure 4.10, we can conclude that for the late learning fish A, the distribution of error in head angle transitions from a unimodal peaked at zero one to a nearly uniform one. This occurs within the first 3 seconds (10 slices). Similar results were obtained for fish B and C. This is shown in Figure 4.11. We can further illustrate this change in distribution over slices by using the $\ell_2$ distance from the uniform distribution. Figure 4.12 shows both the change of $\ell_2$ distance over slice (time) for

late learning and early learning.

Through figures 4.10 to 4.12, we have shown that the distribution of late learning fish transitions from a peaked distribution centered at zero to a uniform one. Meanwhile, the early learning fish has a constantly uniform distribution that does not change over time. This is further confirmed when looking at each trajectory split by slice number in Figure 4.13.

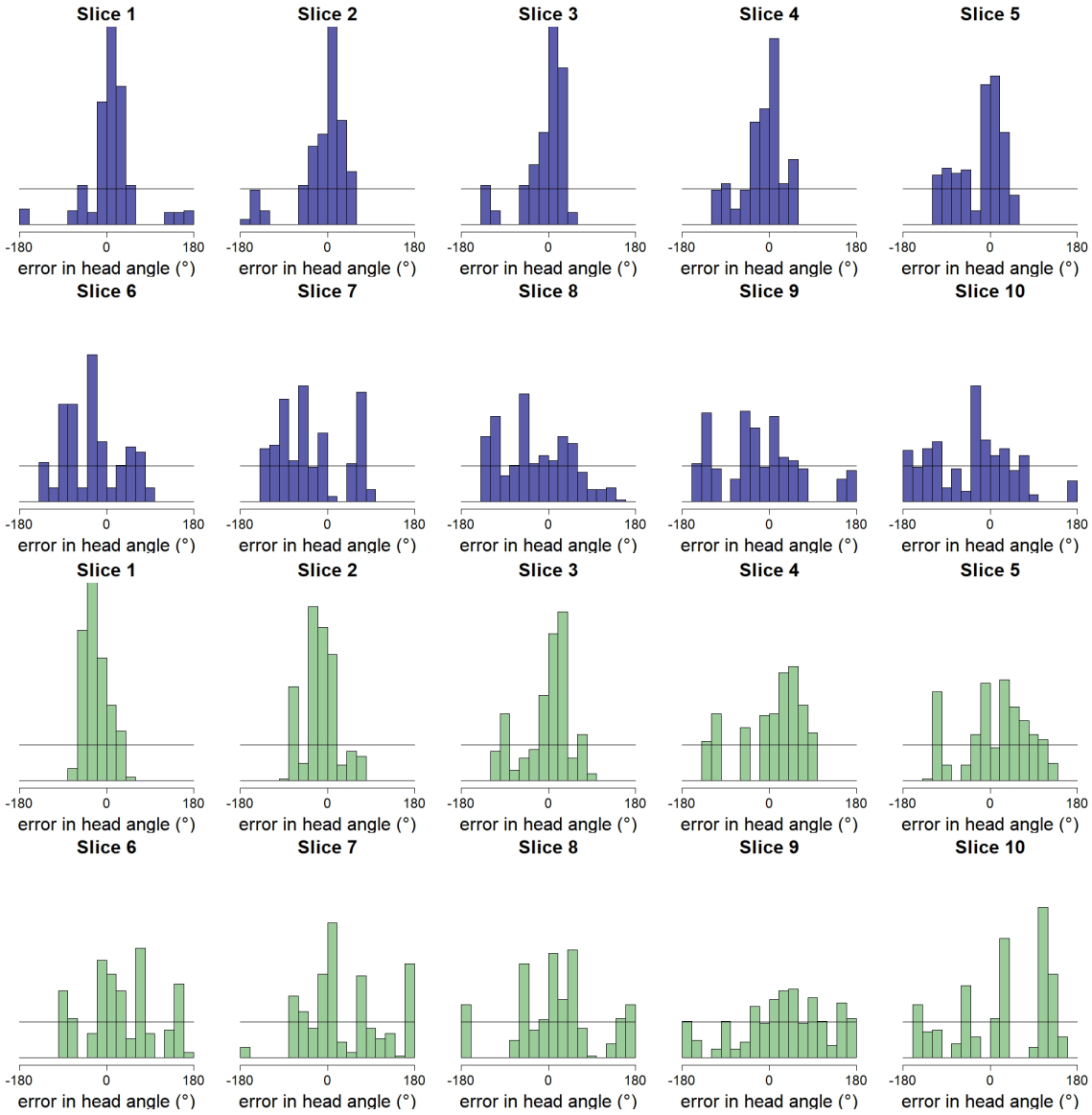


Figure 4.11: **Evolution of the error in head angle distribution over slices (over time) for late learning fish B (blue) and C (green).** Each histogram is composed of 360 data points (30 per trajectory with 12 trajectories in late learning). Each slice lasts 0.3 seconds. All late learning slice 1 data points were put together to make the histogram labelled "Slice 1". All late learning slice 2 data points were put together to makes the histogram labelled "Slice 2", etc. All trajectories were incomplete (hadn't reached the food) within the time span of the 10 slices. Note the transition from peaked at zero at the first few slices to a nearly uniform distribution at the last few slices. The black line indicates the theoretical uniform distribution.

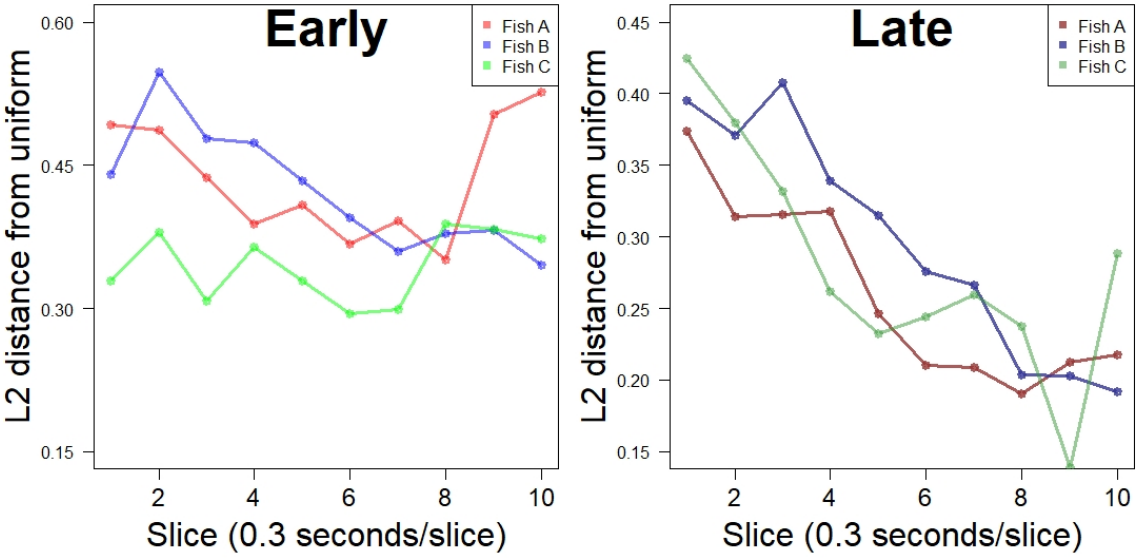


Figure 4.12: $\ell_2$ **distance from the uniform distribution over slice.** Each histogram in Figures 4.10 and 4.11 had its $\ell_2$ distance from the uniform distribution computed. The $\ell_2$ distance from the uniform distribution was then plotted as a single point for that slice. Note that in the case of late learning, the $\ell_2$ distance from the uniform distribution decreases over the slice number (over time). This indicates that the distribution is getting closer and closer to the uniform distribution and that the first few slices were very different from the uniform distribution. In contrast, the early learning fish is generally non-decreasing and flat. It does not get closer to the uniform distribution over the slice number. It is important to note that the early learning fish data points have fewer data points in their histograms because there were fewer early trajectories, by definition of early learning (6 trajectories for Fish A and C, 5 for fish B). Thus, the histogram is naturally further from the underlying distribution (presumably a uniform distribution) and the $\ell_2$ distance from the uniform distribution increases as a result. The late learning data (12 trajectories for all fish) has twice more data in each point, making it closer to the underlying distribution. That is to say, the y-axis values of early and late learning in this figure are not comparable. The important conclusion is that one is a decreasing function (late) while the other is not (early).

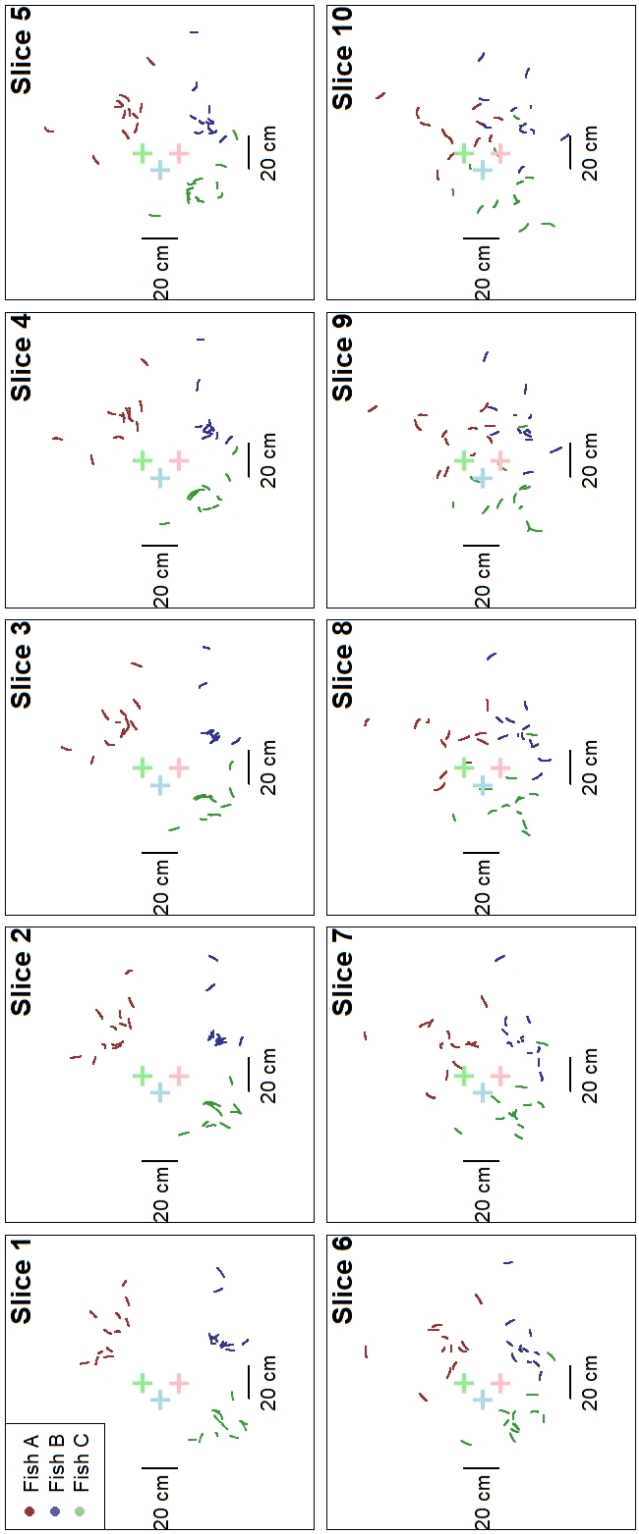


Figure 4.13: **Sub-trajectories per slice (each slice is 0.3 seconds) for late learning fish.** Note that the fish initially go straight towards the food and gradually diverge elsewhere to a near random direction. The food location is represented by a + sign in light red, light blue, and light green for Fish A, Fish B, and Fish C respectively.

### 4.2.3 Interpretation of Results

We have demonstrated that Gymnotus sp. displays learning objectives that are not limited to time taken to find the food source or distance travelled. We have shown that its method of foraging changes once it is in a late learning phase, particularly through its error in head angle (Figures 4.8-4.13) and its change in head angle (Figures 4.2-4.4). This, all the while not displaying stereotypically straight trajectory towards the food source (Figures 4.13).

We first address the consistently non-optimal trajectories the fish take once they know where the food is. Whereas a non-direct route to the food source in early learning is expected and its shape can be explained by energy-constrained proportional betting [CMM20], a non-direct route for late learning is unexpected [MW88]. We provide two interpretations. The first is of imperfect path integration. The fish use path integration to locate the food in the dark [MBG$^+$96, EWM21]. However, this path integration is imperfect and results in only an approximate localization of the food source, after which, the fish begin foraging around the approximate location of the food source. This is supported by the distribution of error in head angle transitioning from unimodal and centered to nearly uniform (Figures 4.8-4.13). The second interpretation is of exploration vs. exploitation [SB18]. Despite the knowledge of the approximate food location, the fish maintain a small natural tendency to explore the environment in case there are new food sources. This contrasts (and is in balance with) the exploitation portion of the behaviour, which incites the fish to "exploit" their knowledge of the food location and go towards it. The balance of the two is only observed in late learning fish, as early learning fish have no knowledge to exploit.

Next, we address the change of foraging style between the early and late learning fish through the statistics of its change in head angle. For both early and late learning fish, the median is nearly 0, indicating that the fish turns equally left and right. The IQR is larger for the late learning fish, meaning the centre of mass of the distribution (the bulk central 50%) is more widely distributed after learning has occurred (Figures 4.3-4.4). However, there are more outliers (i.e., sharp head angle turns) before the learning has occurred (Figure 4.4). We interpret this as a more relaxed swimming behaviour for the late learning fish, allowing more turns but reducing the number of very sharp turns.

We have also demonstrated that the sharp turns observed during early learning foraging follow an exponential distribution with no serial correlation at lag 1 (Figures 4.5-4.6). Furthermore, we have shown that sharp turns occur in open water without any sensory input to evoke it. This indicates that the process may be entirely randomly generated though the fish's neural dynamics. These renewal processes were previously observed for the initiation of swimming [JLM14] with associated EOD. The lack of correlation between EOD and change in direction suggests that there may be different mechanisms for initiating spontaneous movement and changing direction in the fish brain.

# Conclusion

Through this analysis, we have demonstrated that there are features that differ between the early learning fish and the late learning fish. That is, we have found differences in foraging behaviour of agents before and after learning. Namely, we have found that the distributions of head angle of the late learning fish converge towards uniform distributions, which is not the case for the early learning fish. We have also found that the fish takes proportionally fewer sharp-angled turns after learning. We have not, however, linked any of the features found to the neuronal activity of the fish (EOD) and thus not found any differentiation in neuronal activity between early and late learning fish.

We have shown that the early fish foraging trajectories can be well simulated using a Lomax distribution as the underlying generating distribution for the curvature of a two-dimensional curve. This was done by using a discrete analogue to the fundamental theorem of planar curves and some smoothing to simulate physical or neuronal inertia. However, no such model was successfully made for late learning agents in this analysis.

In conclusion, we have successfully reduced the gap in the literature of spatial learning by exploring the learning differences and learning constants in freely foraging fish with a focus on their trajectories. These observations are likely to apply to other foraging agents such as mice or other animals, perhaps even humans foraging by foot or in a vehicle.

# Bibliography


[AB92] Philippe Aghion and Patrick Bolton. Distribution and growth in models of imperfect capital markets. *European Economic Review*, 36(2-3):603–611, 1992.

[ARS99] Christopher Assad, Brian Rasnow, and Philip K Stoddard. Electric organ discharges and electric images during electrolocation. *Journal of Experimental Biology*, 202(10):1185–1193, 1999.

[BD02] Peter J Brockwell and Richard A Davis. *Introduction to time series and forecasting*. Springer, 2002.

[BEY+20] Praneet C Bala, Benjamin R Eisenreich, Seng Bum Michael Yoo, Benjamin Y Hayden, Hyun Soo Park, and Jan Zimmermann. Automated markerless pose estimation in freely moving macaques with openmonkeystudio. *Nature communications*, 11(1):1–12, 2020.

[BL11] Raluca Balan and Gilles Lamothe. *Expect the Unexpected: A First Course in Biostatistics*. World Scientific, 2011.

[BLL07] David Babineau, John E Lewis, and André Longtin. Spatial acuity and prey detection in weakly electric fish. *PLoS Computational Biology*, 3(3):e38, 2007.

[Blo58] Gunnar Blom. *Statistical estimates and transformed beta-variables*. PhD thesis, Almqvist & Wiksell, 1958.

[BM14] György Buzsáki and Kenji Mizuseki. The log-dynamic brain: how skewed distributions affect network operations. *Nature Reviews Neuroscience*, 15(4):264–278, 2014.

[BN77] Ole Barndorff-Nielsen. Exponentially decreasing distributions for the logarithm of particle size. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 353(1674):401–419, 1977.

[CCM05] Angel Ariel Caputi, Bruce A Carlson, and Omar Macadar. Electric organs and their control. In *Electroreception*, pages 410–451. Springer, 2005.

[CL07] Jaerin Cho and Boris Levit. Optimal design in regression and spline smoothing. 2007.

[CMM20] Chen Chen, Todd D Murphey, and Malcolm A MacIver. Tuning movement for sensing in an uncertain world. *Elife*, 9:e52371, 2020.

[CMT01] David Coeurjolly, Serge Miguet, and Laure Tougne. Discrete curvature based on osculating circle estimation. In *International Workshop on Visual Form*, pages 303–312. Springer, 2001.

[CW78] Peter Craven and Grace Wahba. Smoothing noisy data with spline functions. *Numerische mathematik*, 31(4):377–403, 1978.

[DR06] Robert MJ Deacon and J Nicholas P Rawlins. T-maze alternation in the rodent. *Nature protocols*, 1(1):7–12, 2006.

[EWM21] Jacob Engelmann, Avner Wallach, and Leonard Maler. Linking active sensing and spatial learning in weakly electric fish. *Current opinion in neurobiology*, 71:1–10, 2021.

[Fam63] Eugene F Fama. Mandelbrot and the stable paretian hypothesis. *The journal of business*, 36(4):420–429, 1963.

[GFG13] David E Giles, Hui Feng, and Ryan T Godwin. On the bias of the maximum likelihood estimator for the two-parameter lomax distribution. *Communications in Statistics-Theory and Methods*, 42(11):1934–1950, 2013.

[GMY04] Michel L Goldstein, Steven A Morris, and Gary G Yen. Problems with fitting to the power-law distribution. *The European Physical Journal B-Condensed Matter and Complex Systems*, 41(2):255–258, 2004.

[HHTZ18] R. Hogg, R.V. Hogg, E.A. Tanis, and D.L. Zimmerman. *Probability and Statistical Inference*. Pearson, 2018.

[JKB95] Norman L Johnson, Samuel Kotz, and Narayanaswamy Balakrishnan. *Continuous univariate distributions, volume 2*, volume 289. John wiley & sons, 1995.

[JLM14] James J Jun, André Longtin, and Leonard Maler. Enhanced sensory sampling precedes self-initiated locomotion in an electric fish. *Journal of Experimental Biology*, 217(20):3615–3628, 2014.

[JLM16] James J Jun, André Longtin, and Leonard Maler. Active sensing associated with spatial learning reveals memory-based attention in an electric fish. *Journal of Neurophysiology*, 115(5):2577–2592, 2016.

[Jua22] Ashley L Juavinett. The next generation of neuroscientists needs to learn how to code, and we need new ways to teach them. *Neuron*, 2022.

[Kra05] OY Kravchuk. Rank test of location optimal for hyperbolic secant distribution. *Communications in Statistics-Theory and Methods*, 34(7):1617–1630, 2005.

[MBG+96] Bruce L McNaughton, Carol A Barnes, Jason L Gerrard, Katalin Gothard, Min W Jung, James J Knierim, H Kudrimoti, Y Qin, WE Skaggs, M Suster, et al. Deciphering the hippocampal polyglot: the hippocampus as a path integration system. *The Journal of experimental biology*, 199(1):173–185, 1996.

[MM83] P Marchand and L Marmet. Binomial smoothing filter: A way to avoid some pitfalls of least-squares polynomial smoothing. *Review of scientific instruments*, 54(8):1034–1041, 1983.

[MSR07] Evgeni Magid, Octavian Soldea, and Ehud Rivlin. A comparison of gaussian and mean curvature estimation methods on triangular meshes of range image data. *Computer Vision and Image Understanding*, 107(3):139–159, 2007.

[MW88] Martin Müller and Rüdiger Wehner. Path integration in desert ants, cataglyphis fortis. *Proceedings of the National Academy of Sciences*, 85(14):5287–5290, 1988.

[NR17] Laurent Najman and Pascal Romon. *Modern approaches to discrete curvature*, volume 2184. Springer, 2017.

[OD71] John O'Keefe and Jonathan Dostrovsky. The hippocampus as a spatial map: Preliminary evidence from unit activity in the freely-moving rat. *Brain research*, 1971.

[ODA88] Hans G Othmer, Steven R Dunbar, and Wolfgang Alt. Models of dispersal in biological systems. *Journal of mathematical biology*, 26(3):263–298, 1988.

[Par73] Vilfredo Pareto. 1897. *Travaux de Sciences Sociales*, pages 319–357, 1973.

[PH05] Elizabeth Purdom and Susan P Holmes. Error distribution for gene expression data. *Statistical applications in genetics and molecular biology*, 4(1), 2005.

[Pre10] Andrew N Pressley. *Elementary differential geometry*. Springer Science & Business Media, 2010.

[SB18] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[SCTBP98] Etienne Save, Arnaud Cressant, Catherine Thinus-Blanc, and Bruno Poucet. Spatial firing of hippocampal place cells in blind rats. *Journal of Neuroscience*, 18(5):1818–1826, 1998.

[SHT+12] Hirotaka Shoji, Hideo Hagihara, Keizo Takao, Satoko Hattori, and Tsuyoshi Miyakawa. T-maze forced alternation and left-right discrimination tasks for assessing working and reference memory in mice. *JoVE (Journal of Visualized Experiments)*, (60):e3300, 2012.

[Wes14] Peter H Westfall. Kurtosis as peakedness, 1905–2014. rip. *The American Statistician*, 68(3):191–195, 2014.

[WHGJ+18] Avner Wallach, Erik Harvey-Girard, James Jaeyoon Jun, Andre Longtin, and Len Maler. A time-stamp mechanism may provide temporal information necessary for egocentric to allocentric spatial transformations. *Elife*, 7:e36769, 2018.