THE UNIVERSITY
of EDINBURGH

Behavioural motifs of larval *Drosophila melanogaster* and *Caenorhabditis elegans*

Balázs Szigeti

Doctor of Philosophy
Institute of Perception, Action and Behaviour
School of Informatics
University of Edinburgh
2017

Abstract

I present a novel method for the unsupervised discovery of behavioural motifs in larval *Drosophila melanogaster* and *Caenorhabditis elegans*. Most current approaches to behavioural annotation suffer from the requirement of training data. As a result, automated programs carry the same observational biases as the humans who have annotated the data. The key novel element of my work is that it does not require training data; rather, behavioural motifs are discovered from the data itself. The method is based on an eigenshape representation of posture. Hence, my approach is called the eigenshape annotator (ESA).

First, I examine the annotation consistency for a specific behaviour, the Omega turn of *C. elegans*, and find significant inconsistency in both expert annotation and the various Omega turn detection algorithms. This finding highlights the need for unbiased tools to study behaviour.

A behavioural motif is defined as a particular sequence of postures that recurs frequently. In ESA, posture is represented by an eigenshape time series, and motifs are discovered in this representation. To find motifs, the time series is segmented, and the resulting segments are then clustered. The result is a set of self-similar time series segments, i.e. motifs. The advantage of this novel framework over the popular sliding windows approaches is twofold. First, it does not rely on the 'closest neighbours' definition of motifs, by which every motif has exactly two instances. Second, it does not require the assumption of exactly equal length for motifs of the same class.

Behavioural motifs discovered using the segmentation-clustering framework are used as the basis of the ESA annotator. ESA is fully probabilistic, therefore avoiding rigid threshold values and allowing classification uncertainty to be quantified. I apply eigenshape annotation to both larval *Drosophila* and *C. elegans*, and produce a close match to hand annotation of behavioural states. However, many behavioural events cannot be unambiguously classified. By comparing the results to eigenshape annotation of an artificial agent's behaviour, I argue that the ambiguity is due to greater continuity between behavioural states than is generally assumed for these organisms.

Acknowledgements

I would like to express my gratitude for my supervisors, Barbara Webb and Matthieu Louis, for their support and help throughout the project. I am also in debt to my family and many friends, however, I would like to avoid giving an inevitably incomplete list, so thank you everyone who helped me throughout the Edinburgh years. You all know who you are!

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Balázs Szigeti*)

Table of Contents

1 Introduction **1**

1.1 Motivation 1

1.1.1 Studying behaviour 1

1.2 Literature review 4

1.2.1 Behavioural annotation 4

1.2.2 Machine learning techniques 12

1.3 Summary and thesis purpose 16

2 Inconsistencies in *C. elegans* behavioural annotation **19**

2.1 Introduction 20

2.2 Methods 21

2.2.1 Behavioural data 21

2.2.2 Consistency of Omega turn detection algorithms 22

2.2.3 Community survey of Omega turns 23

2.3 Results 27

2.3.1 Consistency of Omega detection algorithms 27

2.3.2 Consistency of expert annotation 28

2.4 Discussion 30

3 Eigenmaggots **33**

3.1 Introduction 33

3.2 Methods 36

3.2.1 Behavioural movies 36

3.2.2 Dropped frames 36

3.2.3 Skeletonising 36

3.2.4 Skeleton normalisation 38

3.2.5 Dimensionality reduction 44

3.3 Results . . . 47
3.3.1 PCA and NNMF eigenmaggots . . . 47
3.3.2 Eigenshape coefficient time series . . . 50
3.3.3 Interpreting eigenmaggots . . . 51
3.4 Discussion . . . 54

4 Motif discovery **57**
4.1 Introduction . . . 57
4.2 Motif discovery with the MK algorithm . . . 60
4.2.1 The MK algorithm . . . 60
4.2.2 The MK dictionary . . . 62
4.2.3 Problems with the MK dictionary approach . . . 63
4.2.4 MK motifs as references . . . 64
4.2.5 Problems with the MK reference approach . . . 67
4.3 The impact of similarity measures . . . 67
4.4 An alternative method: Segmentation-clustering . . . 69
4.4.1 Overview of segmentation-clustering . . . 69
4.4.2 Testing SC on synthetic data . . . 72
4.5 Discussion . . . 74

5 Segmentation **77**
5.1 Introduction . . . 77
5.2 Methods . . . 79
5.2.1 Dynamics change segmentation . . . 79
5.2.2 MK segmentation . . . 82
5.2.3 Quantitative comparison . . . 83
5.3 Results . . . 85
5.4 Discussion . . . 87

6 Clustering **89**
6.1 Introduction . . . 89
6.2 Methods . . . 91
6.2.1 Alignment of ECTS subsequences . . . 91
6.2.2 Model selection . . . 93
6.2.3 Non-classical multidimensional scaling . . . 95
6.2.4 K-means clustering . . . 96

6.2.5 DBSCAN . . . 97
6.2.6 Mixture of regressions models . . . 98
6.3 Results . . . 102
6.3.1 Number of clusters . . . 102
6.3.2 Comparison to ground truth . . . 105
6.4 Discussion . . . 108

7 Continuity of behavioural states **111**
7.1 Introduction . . . 111
7.2 Methods . . . 113
7.2.1 Data collection . . . 115
7.2.2 Constructing eigenmaggots . . . 115
7.2.3 Eigenshape coefficient time series . . . 116
7.2.4 Dropped frames . . . 116
7.2.5 Segmentation . . . 116
7.2.6 Curve alignment and clustering . . . 119
7.2.7 Comparison of behavioural annotations . . . 120
7.2.8 Visualisation, density cross sections and feature histograms . . . 121
7.3 Results . . . 123
7.3.1 Eigenshapes . . . 123
7.3.2 Motifs for *Drosophila* larva . . . 125
7.3.3 Motifs for *C. elegans* . . . 126
7.3.4 Do the larva and the worm exhibit discrete behaviours? . . . 128
7.4 Discussion . . . 135

8 Conclusions and future work **139**
8.1 Introduction . . . 139
8.2 Overview and novel aspects . . . 139
8.3 Limitations and possible extensions . . . 140
8.4 Conceptual considerations . . . 142
8.4.1 Discrete behavioural states versus a spectrum of behaviour . . . 142
8.4.2 Tools for studying the behavioural spectrum . . . 143
8.4.3 The problem with ground truth comparisons . . . 144

A Quantitative evaluation **145**

B Supplementary figures **147**

C Supplementary tables **151**

D Video captions **153**

Bibliography **155**

List of Figures

2.1 Visual explanation of the tightness score 26
2.2 Outcomes of the Omega turn community survey 29

3.1 Overview of eigenworm construction 35
3.2 Examples of successful and failed maggot skeletonisations 38
3.3 Illustrating the discrete angle distribution of skeleton points 41
3.4 Number of skeleton points vs. skeletonisation error 41
3.5 Schematic explanation of the head tracking algorithm 43
3.6 Comparison of NNMF and PCA derived eigenmaggots 49
3.7 Comparing the locomotion of *Drosophila* larva and *C. elegans* 53

4.1 Schematic representation of the MK-reference motif discovery 66
4.2 Euclidean distance vs. dynamic time warping 68
4.3 Schematic overview of the Segmentation-Clustering methodology . . 71

5.1 The dynamic change segmentation algorithm 82
5.2 Evaluation of the segmentation methods 84

6.1 Alignment of the behavioural windows in the time domain 92
6.2 Overfitting 92
6.3 Explaining DBSCAN 97
6.4 Sample of results using the DBSCAN algorithm 104
6.5 Density heat maps of the multidimensional scaling mappings 104
6.6 Multidimensional scaling maps of larval *Drosophila* and *C. elegans* behaviour 107

7.1 The DCS segmentation algorithm 118
7.2 Summary of eigenmaggots 124
7.3 The structure of behavioural motifs for larval *Drosophila* 127

7.4 The structure of behavioural motifs for *C. elegans* 129
7.5 Multidimensional scaling visualisation of a virtual agent's behavioural state space 131
7.6 Continuity among behavioural states 133

B.1 Density cross sections of aggregated ECTS curves 148
B.2 Histograms of larval *Drosophila* action features 149
B.3 Histograms of *C. elegans* action features 150

List of Tables

2.1 Consistency of Omega turn detection algorithms. 27

3.1 Fraction of the original data's variance recovered given the dimensionality of the PCA/NNMF representation 48

4.1 Summary statistics of the SC annotation of synthetic data 73

4.2 Summary statistics of the SC annotation of synthetic data with extra noise 73

5.1 Summary of the performance of the MK dictionary/DCS segmentations on *C. elegans* data 86

6.1 Strength of evidence for model selection by Bayes factors 94

6.3 Statistical summary of *C. elegans* behavioural annotation using *k*-means/spline regression clustering 106

6.2 Statistical summary of the larval *Drosophila* behavioural annotation using *k*-means/spline regression clustering 106

7.1 Statistics of the annotation of larval *Drosophila* behaviour 126

7.2 Statistics of the annotation of *C. elegans* behaviour 128

C.1 Annotation counts of the SC algorithm on sythetic data 151

C.2 Annotation counts of the SC algorithm using *K*-means/spline regression clustering on larval*Drosophila* data 151

C.3 Annotation counts of the SC algorithm using *K*-means/spline regression clustering on *C. elegans* data 152

C.4 Comparing the annotation counts of the SC and JAABA algorithm on larval *Drosophila* behaviour 152

C.5 Comparing the annotation counts of the SC and CBD algorithm on larval *C. elegans* behaviour 152

Chapter 1

Introduction

1.1 Motivation

1.1.1 Studying behaviour

In the past decades, laboratories have been busy characterising the various genes, proteins, cells and neural circuits involved in the generation of behaviour. During this molecular era of biology, the functions of many such components have been elucidated, but much of the toolkit for analysing behaviour has remained the same. The recent development of tools for manipulating neural activity, such as optogenetics, has made the need to quantify the behavioural consequences of these manipulations crucial. After all, behavioural control is the ultimate function of neural information processing [Gomez-Marin et al., 2014], and therefore, in order to understand the nervous system, sophisticated analysis of behaviour is required.

Recently, tracking assays have became popular in both the larval *Drosophila* and the *C. elegans* research community. In this paradigm, the animal is free to roam within an area while a camera mounted on a motorised stage follows it [Martin, 2004,Gomez-Marin et al., 2011]. In the arena, the animal can be subjected to a range of different stimulus conditions. The resulting video recordings can be combined with machine vision techniques to form high throughput behavioural assays.

One advantage of tracking assays is that they allow a more in-depth description of behaviour compared to traditional assays. For example, consider the preference index that is a frequently used quantitative summary of chemotaxis. The preference index

is a quantitative measure of animal's preference for odour A over odour B (where B is often the 'no odour' condition). In typical assays, two odour sources are placed at opposite ends of a Petri dish and a number of larvae (or other animals) are placed in the middle. After a certain time, the number of larvae that end up on each side of the dish are counted, and their quotient is the preference index of the chemicals. Preference index is a quantitative measure of chemotaxis, but it completely ignores what route the larva took, how many times they stopped during locomotion, how they crawled there, how they interacted with each other during this time, etc. Depending on the nature of the query, the preference index might be sufficient to answer a scientific question, but it does not reveal all aspects of chemotaxis. During tracking experiments, the entire behaviour is recorded. Therefore, at least in principle, a more detailed description of behaviour can be obtained.

Currently, the analysis of tracking assays can be loosely divided into two overlapping groups of methods: either the analysis is focused on extracting behavioural features (such as average locomotion speed) to detect phenotypic differences ([Yemini et al., 2013, Brown et al., 2013, Vogelstein et al., 2014]) or the videos are annotated with a fixed set of behavioural states [Gomez-Marin et al., 2011, Gomez-Marin and Louis, 2014, Berman et al., 2014]. This thesis is concerned with the latter, behavioural state annotation approach.

Behavioural states are the recurring, stereotypical actions of an animal. The ethogram, or behavioural dictionary, is the collection of such states. Traditionally, behavioural states have been annotated manually, but this is no longer feasible given the ever-increasing size of behavioural datasets. As a consequence, automated high-throughput behavioural annotators have been developed. An example of the available tools is the Janelia Automatic Animal Behaviour Annotator (JAABA) [Kabra et al., 2013]. JAABA requires initial hand annotation of a subset of the data. Then, the software uses machine learning algorithms to detect the same patterns in the non-annotated data. Other researchers have developed classifiers that extract specific parameters from behavioural data and then register a state if a certain parameter (or parameter set) exceeds a user defined threshold [Ohyama et al., 2013, Gomez-Marin et al., 2011, Salvador et al., 2014, Yemini et al., 2013].

Both of these approaches - supervised learning and the threshold-based methods - share a common problem. Specifically, these methods require a predefined set of behaviours, where the behaviours are defined by the experimenter. In practice, the set of behavioural states is often derived from experimental tradition rather than rigorous mathematical analysis. This practice is troubling because humans are biased observers. It should be noted that the annotators that rely on user-defined states carry the same potential observational biases as human annotation. Furthermore, the lack of a computational definition of behavioural states makes it difficult to compare results across publications. In response to these problems, the primary concern of this thesis was to develop a method that detects behavioural states in an unbiased manner.

1.2 Literature review

The idea of behavioural states has been around for a long time and there are numerous papers that conceptualise behaviour as a set of discrete states. The first subsection (1.2.1) covers studies that applied some form of computer assisted annotation of behavioural states or related ideas to study either larval *Drosophila* or *C. elegans*. The technical objective of this thesis is the development of unsupervised motif discovery from the posture time series of these animals. Hence, in the second subsection (1.2.2), the machine learning literature regarding the problem of motif discovery is reviewed.

1.2.1 Behavioural annotation

An ethogram is a catalogue of featured behaviours or actions of an animal. The aim of this thesis is to achieve an unbiased construction of the ethogram. Hence, in this literature review, the focus is on past attempts to construct the ethograms for *C. elegans* and *Drosophila*. Studies that did not construct an ethogram, but that represent a significant step towards computer assisted tracking and behavioural analysis, are also presented.

[Green et al., 1983] is one of the first studies that aimed to mathematically analyse the ethogram of *Drosophila* larva. For the experiments, larvae were placed on agarose contaminated with various concentrations of alcohol. The study characterized how two species of Drosophila (*Drosophila melanogaster* and *Drosophila simulans*) modulate their behaviour as a function of the concentration of alcohol in the agarose. The larva's behaviour was annotated manually, with six distinct behavioural states: locomotion, turning, retreating, rearing, bending and burrowing. These elemental behaviours were based on observation of the larvae in the lab. That is, they were based on the human perception of what the recurring elements of larval behaviour are. Such a determination therefore has a subjective element, and much of the current thesis revolves around how subjectivity can be reduced when constructing the ethogram. Based on the annotated behavioural data in the study, a computer algorithm was used to construct the matrix of transition frequencies, or the probability of transition from one given state another. The departure from what was expected by chance for each transition was used to derive a z-score. The analysis demonstrated both inter- and intra-species variations in behaviour and it was shown that larvae modulate their food intake as a function of alcohol concentration.

Similar approaches have been used in more recent works. For example, [Godoy-Herrera and Connolly, 2007] compared the organisation of foraging in natural populations of four *Drosophila* species (*Drosophila simulans*, *Drosophila mauritania*, *Drosophila gaucha* and *Drosophila pavani*) and their crosses. To assess behaviour, larva were placed on agarose, where they were observed for 2 mins. To characterise the behaviour, the authors used a Markov model with four bases states (feeding, locomotion, bending and turning). Similarly, as in the [Green et al., 1983] study, these states were justified by observation rather than being deduced from the data. To record when each behaviour happened, a human observer annotated the data. The resulting frequencies and transitions between behaviours were tested using the χ^2 test, with the null hypothesis being that associations between behaviours occur randomly in the sequence. The studys findings were that the transition between behaviours was random for two species (*Drosophila simulans*, *Drosophila mauritania*), the other two showed statistically significant dependencies.

Despite the continued use of manual approaches, computer aided analysis is becoming increasingly more common. [Gomez-Marin et al., 2011] combined tracking *Drosophila* larvae in odour gradients with computer assisted behavioural annotation to study chemotaxis. The paper characterizes behaviour as a discrete-time, first-order Markov model. The states correspond to head casts and turns, with both of these behaviours further differentiated based on whether the head cast/turn is towards the higher or lower concentration in the odour gradient. A threshold of head angle, or the angle between the axes of the head and the body, defines head casts. Note that in this case, behaviours were detected by a computer rather than by a human observer, as was done in the previous studies. The paper demonstrates that the sensory stimulus is integrated during runs and negative gradients control the timing of turns, while high amplitude head casts determine turn direction. The authors argue that larvae actively sample their environments and hence, the larva's navigation strategy lies somewhere between biased random walk and stereo olfaction.

[Ohyama et al., 2013] were interested in characterising the response of the larvae to various noxious stimuli. It should be noted that this is a more constrained problem than describing the unconstrained roaming of the animal, since the stimulus defines a narrow time window in which the behaviour is to be analysed. The following behaviours were assessed in the study: crawling, rolling, head cast and hunching. To

detect these, the authors devised a Schmitt-trigger algorithm. In short, there is a 'key function' parameter associated with each behaviour. For example, speed is the key parameter used to recognise crawling. The algorithm requires two thresholds - thr_{lower} and thr_{upper} - in each key parameter. Whenever the key parameter increases above thr_{upper}, the associated behaviour begins, and it ends when the time series returns to below thr_{lower}. To calibrate the system's threshold values, the authors hand annotated a subset of their data and searched for the parameters until the false detection rate was below 5% and false negative rates were $<15\%$. Using this behavioural detection method, the authors characterise how larvae attempt to escape various noxious stimuli.

An example of a study that did not pre-specify the number of states was [Vogelstein et al., 2014]. This study aimed to create a neuronal-behavioural map, that is, a mapping between neuronal activity and specific behaviours. The authors optogenetically activated 1,054 neuronal lines, and the resulting behaviours were clustered using multi-scale unsupervised structure learning. Similarly, as in [Ohyama et al., 2013], the authors were interested in characterising behaviour within a predefined time window. Neurons were activated optogenetically for 30s and the animal was then observed for a further 10s. The behaviour during this 40s time window was analysed for each strain. For each experiment, that is, for each 40s time window of larval behaviour, the larvae were tracked and eight features (area, head turn, direction of motion, width, forward/backward crawling bias, length, speed, sideways speed) were calculated. This eight-dimensional time series was fed into an iterative de-noising tree algorithm, which was a variant of hierarchical clustering. Important to note is that this study did not use explicit thresholds to detect behaviours; however, the iterative de-noising tree algorithm contained 6 user-defined parameters. The algorithm recognised 29 distinct behavioural responses, which were typically a combination of more elementary behaviours. For example, some of the behavioural responses were labelled post-hoc by a human as 'right-left-avoid,' 'wiggle-escape' and 'turn-slow-crawl.' The main achievement of this paper was the demonstration of how specific behaviours map onto the activation of specific neuron populations.

A similar approach that focused more on comparing parameters than on detecting states was taken by [Aleman-Meza et al., 2015]. The paper presented an automated tracking system coupled with a software package for analysing the locomotion patterns of larval *Drosophila* strains. The tracker kept the animal in the center of the field

of view during the experiments, while the software recorded 20 locomotion parameters (such as body length, stride distance, run duration, etc.). Using this data, the authors examined strains that covered more distance during a standard 5-min assay. Strains that went further did so by increasing their striding speed but not striding time. Correlation analysis revealed that the increased speed was due to both faster and longer strides. The authors also demonstrated that a number of locomotion parameters are significantly different for larva in their second and third instars. Overall, this paper presents an integrated platform for recording video tracking assays and analysing the videos using a feature extraction methodology.

The description of *C. elegans* behaviour started with qualitative, observational studies. The first study of this kind was done by [Croll, 1975], and it described behavioural patterns. This work was also the first in which Omega turns were explicitly mentioned. The development of automated behavioural analysis took a different trajectory in worm research due the family of *unc* ('uncoordinated') mutants. These worms have some defects in their locomotor behaviour and the quantitative description of such differences was a primary goal of early behavioural studies.

[Baek et al., 2002] combined tracking assay data with machine learning to quantitatively describe the *unc C. elegans* phenotypes. For every video frame collected during tracking assays, 94 features were measured. Most of the features were morphological (e.g., min, max, and average of best-fit ellipse's major axis, number of times the worm coiled, etc.) or described the worm's locomotion (e.g. min, max, average distance travelled in 10 frames, number of reversals, etc.). These feature time series were then fed into a classification and regression tree (CART) algorithm. CART is a supervised learning method that uses binary decisions to iteratively separate the data into nodes (in this example, the nodes should correspond to worm phenotypes). The study concluded that 7 features are sufficient to distinguish the *unc* worm phenotypes.

[Geng et al., 2003] attempted to find phenotype clusters in the *C. elegans* locomotion and morphology feature space. The study included the wild type N2 worm and seven well-characterised loss-of-function mutants. First, the authors used principal component analysis to demonstrate that the size of feature space could be significantly reduced. Principe component analysis (PCA) revealed that the top 43 features (of the 253 measured during the tracking assay) capture $>94\%$ of the total variance. Note

that in this case the PCA was applied to morphological features, not to the posture of the animal as in the case of [Stephens et al., 2008]. This finding indicates that a few carefully selected features might be sufficient to distinguish the behavioural phenotypes. To select the features with the best phenotype discriminatory power, a nearest neighbour clustering with a 10-fold cross validation was used. The chosen 39 features were then used as the input to k-means clustering with the Euclidean distance metric. Gap statistics and distortion curves were used to find that 6 was the optimal number of clusters. Among 10,000 random starts to best fit clustering with k=6, the algorithm grouped together worms with calcium channel deficiency and strains with nicotine receptor mutations. For the other strains, the algorithm correctly grouped the worms with the same genetic background into the same cluster. The study demonstrates that the reduced feature set is capable of discriminating phenotypes such as $unc-2$ and $unc-36$, which are difficult to distinguish, even for experienced human observers.

[Huang et al., 2006] presented the first machine vision algorithm that could resolve *C. elegans* postures while the animal was intersecting itself (coils). In many such events, the morphological skeleton is not a valid characterisation of posture. The author's algorithm relies on measuring the average length and width (L and W) of the worm in the non-coiled frames and then using this information to resolve coils. In short, touching parts are identified by looking at where the worm is significantly wider then W. The touching segment is 'cut through' from the exterior to interior boundary. To the resulting image, a standard skeletonising algorithm can be applied to recover the biologically meaningful skeleton. Using this skeletonising algorithm, a number of features were measured during foraging and the phenotypes were clustered using the CART algorithm. CART distinguished weak and strong coiler phenotypes. Furthermore, the paper reported a strong ventral bias of Omega turns for the wild type worms that disappeared for certain *unc* strains.

The cornerstone of the present thesis is the concept of eigenshapes. In the study of animal behaviour, the eigenshapes of *C. elegans*, the eigenworms, were first introduced by [Stephens et al., 2008]. The authors used data from a tracking assay, i.e. the top-view videos of worms as they freely roamed in an environment. The posture of worms was approximated by their midline and principal component analysis (PCA) was used to reduce the dimensionality of the midlines. PCA revealed that four eigenworms accounted for 92% of the variance in postures. This means that four numbers can de-

scribe any actual worm posture with high precision. Figure 3.1 shows the eigenworms and an example of posture reconstruction. See Chapter 3 for a detailed overview of eigenshapes, as well as the development of the corresponding representation of larval *Drosophila*, the eigenmaggots.

[Brown et al., 2013] used the eigenshape approach to create a behavioural fingerprint of mutant strains, and showed that the fingerprints cluster around genes with similar biological function. In the paper, it was first established that wild-type derived eigenworms could be used to efficiently represent the postures of mutant worms. That is, the same four-dimensional vector space can be used to represent postures of the wild-type and mutant strains. Next, the authors searched for motifs in the eigenworm time series for all worms (wild type and mutant strains). To detect motifs, a simple sliding window methodology by [Mueen et al., 2009], the MK algorithm, was used. By definition, this method identifies motifs as pairs of similar subsequences with a fixed length. I found this definition inadequate, however, and this thesis was largely inspired by my dissatisfaction with the algorithm (for a detailed criticism, see Chapter 4). All motifs detected by the MK algorithm (either in the behaviour of wild type or mutant worms) were put into a motif database, the 'behavioural dictionary.' This dictionary was pruned to 700 motifs using a minimum-redundancy, maximum-relevance criterion. Each worm's motifs were matched to the closest element in the pruned dictionary. The average distance between each of the worms motifs and the closest matching dictionary element were used as a measure of phenotypic dissimilarity. This phenotypic dissimilarity distance matrix was fed into an affinity propagation clustering algorithm to find clusters of strains with similar behavioural fingerprint. The algorithm detected four broad groups that were loosely associated with monoamine synthesis, neuropeptide production, G-protein coupled receptors and the uncoordinated (*unc*) mutant strains. It was also shown how this network of phenotypic associations could be used to generate hypotheses about the biological function of mutations.

Like many of the other papers in this literature review, [Yemini et al., 2013] was concerned with the automated phenotyping of *C. elegans* mutants. However this paper used much more data, significantly boosting the statistical power of the study, along with having a wider feature set then any other study. Furthermore, all of the videos and the corresponding feature files (containing all of the calculated features) are publicly available. The database contained information about 305 different strains, including 76

mutants that previously had not been associated with a distinct phenotype. Much like other work on worm phenotyping, morphology and locomotion features were used. Every feature of every strain was tested against the wild type N2 worm using the Wilcoxon rank-sum test. The authors report that at least one of the features of every strain is significantly different from the wild type.

Although [Berman et al., 2014] looked at adult *Drosophila*, the paper is still worth discussing, as it also used machine learning to search for behavioural states. The paper first established the 'eigenflies,' or the eigenshapes that represent adult *Drosophila*. To construct the eigenflies the authors follow an analogous pipline as presented in Chapter 3. Due to having a large number of joints and a complicated body shape, the fly's posture can be characterised as a superposition of 50 eigenshapes (in contrast, to characterise the posture of *C. elegans*, four numbers are sufficient). Therefore, the adult fly's posture is represented by a 50-dimensional time series. The authors used Morlet wavelet analysis for each component of the postural time series, resulting in 50 spectograms. To identify similar patterns in the spectrograms, a spatial embedding method called t-distributed stochastic neighbourhood embedding (t-SNE) was used. t-SNE turns the spectograms into a 2-dimensional map, where each point corresponds to a wavelet vector and proximity between points represents how closely the vectors are related. Densely populated areas of the t-SNE map indicate frequent wavelet components, i.e. frequently repeated posture sequences. To turn the map into a set of discrete behavioural states, a watershed algorithm is used. The drawback of this method is the large number of freely adjustable parameters. The t-SNE algorithm alone has 22 adjustable parameters and the paper does not present a sensitivity analysis. Similarly, the watershed algorithm is dependent on its parameter settings. Therefore, the behavioural map (as provided by the t-SNE algorithm) can be discretized in many different ways depending on the user-defined parameters for the watershed algorithm.

Machine learning was applied by [Kabra et al., 2013] in order to train behavioural state detection from human annotation. The paper introduced the interactive JAABA software package, a framework for training behavioural detectors. First, the user provides training examples, based on which a behavioural classifier is trained. JAABA includes a number of visualisation features that allows the user to quickly evaluate the classifier. Frames where the classifier is mistaken can be tagged by the user, and based on this information the classifier is updated. Therefore, the classifier is trained

in an iterative manner until its performance is satisfactory. The underlying machine learning algorithm takes trajectory information and videos as input. From the input data, a number of species-specific features are calculated. Most of the features are calculated 'per-frame' (e.g. the midbody bend for larval *Drosophila*), but a number of window-based features (e.g. the mean of midbody bend) are calculated as well to provide JAABA with a temporal context. The set of feature time series are then fed into the GentleBoost learning algorithm, which combines many weak rules, where each rule is a threshold on a single feature. The algorithm tries to find the thresholds that maximise the distance between the examples and the decision boundary. To demonstrate the feasibility of JAABA, the authors trained behavioural classifiers for mice and both the adult and larval *Drosophila*. In all three cases, JAABA was able to train classifiers that were comparable to expert annotation. Furthermore, to demonstrate the tool's robustness, it was shown that JAABA could reliably distinguish populations of animals in different conditions (age and starvation groups) and six wild strains of adult *Drosophila*.

1.2.2 Machine learning techniques

So far the literature review has been focused on approaches to behavioural annotation in specific species. When viewed as an abstract problem, the key issue is the identification of motifs. Hence, this section of the literature review focuses on the different motif-finding algorithms in the machine learning literature. Note that the overwhelming majority of motif discovery methods are based on discrete univariate data. The aim of this thesis is to discover motifs in postural data represented as a continuous multivariate time series. Therefore, only a small fraction of the methods explored in the literature are relevant to this problem.

[Mueen et al., 2009] designed an MK motif-finding tool based on the calculation of the distance between subsequences, where the size of the subsequences is a user-defined parameter. I have experimented extensively with this tool and find it unsatisfactory (see Chapter 4 for details). In short, its main problem is that the authors define a motif as the pair of closest neighbours, that is, the pair of subsequences that have the smallest distance among all possible pairs of subsequences. However, intuitively, a motif is not a subsequence that happens twice, but rather a subsequence that is frequently repeated throughout the time series. The algorithm is 'exact' in the sense that it is guaranteed to find the closest neighbours. The MK algorithm uses a pair of techniques 'early abandoning' and 'linear projection ordering' to speed up the search compared to brute force search, but yields the same end result. With these two tricks, the algorithm achieves a 100-fold speed increase. The dramatically increased speed allowed the authors to find motifs within large datasets, where the time taken would have been prohibitive for a number of other algorithms.

The variable length motif discovery algorithm (VLMD) presented in [Nunthanid et al., 2011] uses the MK algorithm as a subroutine, while attempting to overcome two problems associated with the MK. These two problems are the fixed window size and the closest neighbour definition of motifs. VLMD applies MK to the time series multiple times, with each run having a different window size. It starts with the smallest scale and then each consecutive run uses a larger window size. If a newly discovered motif overlaps with an already discovered motif, they are added to the same motif group. Otherwise, a new group is initiated. Once VLMD finishes the scan of the time series at every length scale, a representative example is selected for each motif group.

The exemplar is the motif with the least sum of within-group distances. VLMD looks promising on paper, but for practical applications, a more sophisticated motif-merging strategy is required.

[Mohammad et al., 2012] proposed the 'greedy stem extension' (G-SteX) algorithm to detect motifs without any user-defined length limit. The algorithm uses constraints on the location of the motifs. Specifically, it locates motif stems around the change points of the time series (change points are where the time series changes in terms of some statistical characteristics). In Chapter 5, I will use a similar approach to segment the eigenshape coefficient time series. G-SteX first applies the robust singular spectrum transform algorithm ([Mohammad and Nishida, 2009]) to find the change points. Motif stems are defined by a short window around the change points. These motif stems are then extended in the time domain until a stopping criterion is met. Multiple stopping criteria were proposed by citemohammad2009robust, but were derived from the distribution of pair-wise distances among the motif stems. Motifs are grown by locally optimal decisions that might result in a globally suboptimal solution (i.e. the greedy algorithm). Through experiments on synthetic data and motion patterns extracted from virtual reality experiments, the authors demonstrate the viability of this method.

[Minnen et al., 2007] conceptualized motif searching as finding high density areas in the space of all possible time series subsequences. This definition of motifs is appealing because it captures the notion of both 'similarity' and 'frequent occurrence.' To identify high density regions, the k nearest non-overlapping neighbours are found for each subsequence and then the local density is estimated by the distance to the k^{th} neighbour. In this method, the subsequence length is a fixed, user-defined parameter. A high-density subsequence is defined as having a higher local density then any of its neighbours. These high-density regions are an over-complete set of motifs seeds. To define the motifs, every motif seed is modelled as a hidden Markov model (HMM). HMMs are optimally fitted to the time series segments by a generalised Viterbi algorithm (adapted from speech recognition studies) and to avoid over-fitting, a mixture-learning framework is adopted. The algorithm has conceptual appeal, but is unfortunately difficult to test, as it has no publicly available implementation.

[Oates, 2002] proposed the PERUSE algorithm for the discovery of motifs in continuous time series data. The algorithm assumes that the entire time series is a concatenation of a set of predefined motifs (often called the 'dense motif' assumption). Each motif is associated with a generating process and a certain distribution (some patterns might occur more frequently then others). The algorithm generates samples by sliding a window across the time series (the window size is user defined) and a maximum probability mapping criterion is used to find candidate motifs among the sample subsequences. These candidates are then passed to an expectation-maximisation algorithm to recover both the generating processes and the identity of each motif.

Symbolic aggregate approximation (SAX) is a method for discretizing time series [Lin et al., 2007]. The algorithm is included here because some of the presented motif-finding algorithms start by using the SAX method to discretize the time series. SAX starts with two user-defined parameters: the window size and the alphabet size. Alphabet size is the number of unique discrete symbols used, and window size controls how many time series points are aggregated into a single symbol. SAX first normalizes the time series ($\mu = 0$, $\sigma = 1$) and then the average of subsequences is taken with the user-defined window length (this window is slid across the time series and the average is taken for each window). Each segment is assigned a symbol based on Gaussian breakpoints, such that each symbol has an equal probability of occurrence in the symbolic time series.

[McGovern et al., 2011] presented an algorithm for locating motifs within multidimensional time series, with applications in weather forecasting. The strategy is to first identify motifs in each individual dimension using maximum probability of detection and minimum false alarm ratio criteria, and then to extend the motifs across the dimensions using the same criteria. First, each dimension of the time series is converted into a symbolic time series using SAX. Next, a trie is built for each dimension of the data. Trie is a tree-like data structure where each leaf corresponds to a unique SAX word, i.e. aggregates of SAX symbols with length equal to a user-defined window size. Once the trie structure is built, the leaves/words are pruned using the maximum probability of detection and minimum false alarm ratio criteria. The threshold values for the criteria are determined by the user and they influence how many words are left in the SAX 'dictionary' after pruning. Once the SAX dictionary is ready for each dimension, the words are then grown across the dimensions of the data using the same maximum

probability of detection and minimum false alarm ratio criteria. The algorithm has been found to be useful in narrowing the domain of weather forecasting, but there is no evidence of its generalizability to other applications.

[Li and Lin, 2010] used time series discretization and the greedy grammar inference to find approximate variable-length motifs. Their algorithm first applies the SAX discretization to convert the time series into symbols. Next, the SAX symbols are aggregated to form SAX words of user-defined length. Consecutive SAX words are eliminated from the stream ('numerosity reduction') and then a greedy grammar induction algorithm, the Sequitor [Nevill-Manning and Witten, 1997], is applied to the SAX words. Briefly, repeated patterns of SAX words are merged and replaced by a new symbol to compress the symbol series into a hierarchical rule structure. The rules, substituting one symbol for a set of other symbols, are the discrete equivalent of motifs. Note that the grammar is learned locally, and therefore may not be minimal. The advantage of this approach is its fast speed and automatic organisation of the data into a hierarchical structure.

Finally, throughout this thesis, hand annotated data is compared with the annotation achieved by various automated tools. The framework for comparing two sets of annotations (one of which is the reference while the other is the one being evaluated) is adopted from [Powers, 2011]. To summarise annotation accuracy, the *Precision* (also known as positive predictive value) and *Sensitivity* (also known as recall and true positive rate) are reported. *Precision* is the ratio of true positive events to all events tagged by the annotator, while *sensitivity* is the proportion of true positives to all reference events. These two measures are commonly combined into a single summary number, the $F-score$. These evaluation metrics are frequently used throughout the thesis. See *Appendix A* for a more through mathematical description of this evaluation framework.

1.3 Summary and thesis purpose

As has been shown in the literature review, currently most behavioural studies use supervised methods to identify behavioural states. These approaches suffer from the problem that they learn the same observational biases that are inherent to the human-produced training datasets. Chapter 2 provides direct evidence that this can lead to strikingly inconsistent classification between different individuals and different algorithms.

One way to overcome this problem is to use unsupervised methods to discover behavioural states directly from the data. Technically behavioural states are postural motifs. In Chapter 3, an efficient description of posture based on principal component analysis of midline shapes is extended from *C. elegans* to *Drosophila* larva, creating an eigenshape coefficient time series on which unsupervised motif discovery tools could be used.

The current standard to find continuous multidimensional motifs typical involves some version of the sliding window approaches. It is argued in Chapter 4 the underlying assumption of this methodology is not appropriate to study behaviour. After evaluating the shortcoming of the sliding window approaches, in the same chapter, an alternative motif discovery approach, the segmentation-clustering framework is outlined.

As the name suggests segmentation-clustering is a two step process. In Chapter 5 and 6 the technical details of the segmentation and the clustering steps are presented in detail. These chapters are focused on the technical details of the algorithms and justify why these particular methods have been used. Furthermore it is demonstrated that the behavioural annotation for *C. elegans* and larval *Drosophila* are competitive with the best supervised approaches such as JAABA.

The main scientific finding of this thesis is that many behavioural states of both *C. elegans* to *Drosophila* larva can not be unambiguously classified. Hence behaviour should be conceptualised as a ‘spectrum of behaviours’, rather than a ‘set of discrete states’. The arguments for the behavioural spectrum are presented in Chapter 7.

In Chapter 8 the conclusions and final thoughts are presented along with possibilities how the current work could be extended

Chapter 2

Inconsistencies in *C. elegans* behavioural annotation

Note

At the time of submission, this manuscript is under review for publication in the journal *Frontiers of Behavioural Neuroscience*. It was the last peer-reviewed paper produced during this thesis project, but it is being presented early on, as it demonstrates the need for unsupervised methods in behavioural studies.

My co-authors on this paper were Tom Stone and Barbara Webb. The paper includes the following author's contribution statement:

BS conceived of the study, developed the code, analysed the data and wrote the article. TS developed the web implementation of the Omega event selection algorithm and maintained the survey's website. BW supervised the project and helped write the manuscript.

Abstract

High quality behavioural annotation is a key tool for linking genes to behaviour, yet relatively little attention has been paid to checking the consistency between various automated methods and expert judgment. In this paper we investigate the consistency of annotation for the 'Omega' turn of *C. elegans*, a frequently used behavioural assay for this animal. First, the output of four Omega detection algorithms are examined for the

same data set, and are shown to have relatively low consistency, with F-scores around 0.5. The consistency of expert hand annotation is then analysed based on an online survey combining two methods: participant judgement of a fixed set of predetermined clips; and an adaptive psychophysical procedure used to estimate individual's threshold for Omega turn detection. The survey revealed a substantial lack of consistency in decisions and thresholds. Such inconsistency makes cross-publication comparison difficult and raises issues of reproducibility.

2.1 Introduction

Traditionally, behavioural annotation has been done manually, with the known weakness of inherent variability, as well as its labour intensive nature. In the current era of big data biology, there is an increasing tendency for behavioural annotation to be automated [Kabra et al., 2013, Gomez-Marin et al., 2014]. Automated methods can obviously scale to significantly larger data sets, but they are also supposed to improve consistency by removing human judgement from the process. However, to the best of our knowledge, little effort has been made thus far to determine the actual consistency between different automated methods. Furthermore, these algorithms are typically validated relative to a human-produced 'ground truth' dataset [Szigeti et al., 2015,Laurent et al., 2015, Yemini et al., 2013, Huang et al., 2006, Salvador et al., 2014]. Such an evaluation process raises the possibility that algorithms are trained to learn the same observational biases - and variance - that are inherent to human annotation. Given that different research groups often use different annotation methods, a lack of consistency in their output could make comparison of published results from these groups difficult.

In this paper we specifically address the consistency of behavioural annotation for the nematode worm *Caenorhabditis elegans* (*C. Elegans*), focussing on a particular worm behaviour, the Omega turn. Omega turns occur during reorientations, with the animal adopting a shape resembling the Greek letter Ω (see Figure 2.1**A** for a representative example). This behaviour was chosen because it is often treated as a discrete, well-defined element of worm behaviour [Croll, 1975, Pierce-Shimomura et al., 1999, Yemini et al., 2013, Salvador et al., 2014, Albrecht and Bargmann, 2011].

Our Omega turn consistency check has two components. First, we examine the consistency of four Omega detection algorithms from the literature [Laurent et al., 2015, Yemini et al., 2013, Huang et al., 2006, Salvador et al., 2014]. Second, we present the results of an on-line survey, in which experts were invited to score Omega turns. The survey itself had two underlying components. In the first, participants scored a set of predetermined clips, and in the second, we employed an adaptive psychophysical method to identify individual's threshold for Omega turns.

The results show that both expert annotation and algorithms are surprisingly inconsistent. Thus, greater effort may be needed to ensure that annotation methods can provide a reliable basis for studies that include behavioural assays.

2.2 Methods

2.2.1 Behavioural data

This study used data from the *C. elegans* behavioural database (CBD) [Yemini et al., 2013], which consists of worm videos and corresponding feature files that contain a number of pre-calculated feature time series (such as speed, eccentricity, eigenworm coefficients, etc.). We examined 776 experiments, all performed with hermaphrodite N2 worms. In the experiments, worms were placed on a plate covered with a bacterial layer and behaviour was recorded after a 30-minute habituation period. Each video was approximately 15 minutes long, so a total of 194 hours of worm behaviour were analysed.

During Omega turns, the worm can contact itself, producing an intersecting shape in the videos, and for these frames it is difficult to extract a biologically meaningful skeleton [Huang et al., 2006, Broekmans et al., 2016]. As a consequence, these 'coiling' frames were not processed in the CBD and the features for the corresponding frames were not calculated. If the gap was smaller than 20 consecutive frames (0.6 sec), linear interpolation was used to gain a proxy for the features. This interpolation method is not reliable for longer gaps, however, and so Omega events with longer gaps were discarded.

2.2.2 Consistency of Omega turn detection algorithms

Algorithms

Four algorithms have been taken from the literature to examine their consistency with one another. The algorithms are from the Zentracker package [Laurent et al., 2015], the *C. elegans* behavioural database (CBD) [Yemini et al., 2013], a computer vision based study for detecting Omega events [Huang et al., 2006], and a recent publication looking at search behaviour [Salvador et al., 2014]. Common to all these methods is that they detect Omega turns when a feature (or a combination of features) exceeds a user-defined threshold. For example, [Yemini et al., 2013] uses the midbody bend as the defining property of Omega turns. Note that this is not an exhaustive list of Omega turn detection algorithms. These particular algorithms were chosen because the code used for the original publication was readily available.

Consistency quantification

To summarise annotation consistency, we report the precision (positive predictive value) and sensitivity (also known as recall and true positive rate) [Powers, 2011]. Precision is the ratio of true positive events to all events tagged by the annotator, while sensitivity is the proportion of true positives to all reference events. Mathematically they are expressed as follows:

$$Precision = \frac{TP}{TP+FP}, \qquad Sensitivity = \frac{TP}{TP+FN}, \tag{2.1}$$

where TP, FP and FN are true positive, false positive and false negative respectively. A TP is counted if at least 50% of the frames overlap. Furthermore, the precision and sensitivity are combined into a single number summary, the F-score, which is defined as:

$$F = \frac{2(Precision \times Sensitivity)}{Precision + Sensitivity}. \tag{2.2}$$

Threshold tuning

In Table 1, the results of the consistency check are presented with the original threshold (taken from the respective publications) for each method. The consistency measures are also presented, with the thresholds tuned for optimal match to demonstrate that the observed inconsistency cannot be eliminated by parameter adjustments.

To find the optimal match, 25 runs with different thresholds were created for each algorithm. For each run, the difference in the threshold was increased or decreased by 2.5% of the initial value. Therefore, a range of 70%-130% of the initial threshold values were scanned. Lower percentages indicate a more permissive definition, but some scales needed to be inverted. For example, [Laurent et al., 2015]'s method uses an upper bound on 'eccentricity' and a lower bound on 'solidity.' Therefore, to make the run associated with 70% more permissive, the eccentricity scale had to be inverted.

2.2.3 Community survey of Omega turns

Survey structure

To compare the consistency of expert Omega turn detection, an online survey was developed [1]. After a brief registration, the participants were shown 40 short (2-5s) clips of Omega events and were asked to indicate, using a button press, whether each was an Omega turn. Participants were also asked to rate their confidence in detecting Omega turns on a scale of 1-5 (with 5 being very confident).

In the survey we wanted to include ambiguous, wide amplitude turns that one may or may not consider an Omega turn. Therefore, to select events for the survey, we ran [Huang et al., 2006]'s Omega detection algorithm on the videos, but with the threshold reduced to 75% of its original value. Using this criterion, 1,526 Omega-like events were detected.

Of the 40 clips in the survey, 20 were predetermined videos that were scored by everyone. The remaining 20 were determined by an adaptive threshold finding procedure, by which the next clip shown depended on previous answers. Specifically, the truncated staircase method was used [Treutwein, 1995] to estimate thresholds (see be-

[1] The survey can be found at *http://groups.inf.ed.ac.uk/worms/index.html*

low). To conceal this structure and reduce order effects, these two components (the predetermined set and the threshold finding) were mixed together such that each predetermined clip was followed by a clip used to detect the threshold. The participants were not told in advance of these two underlying components to eliminate possible cognitive biases.

To gather survey responses, we emailed 47 experts (PIs identified from publications on C. elegans behaviour), inviting them and their laboratory members to participate. The survey was also advertised through the social media presence of the OpenWorm project.

Selection of predetermined clips

To select the 20 predetermined clips, the eigenshape annotator (ESA) was used [Szigeti et al., 2015]. In brief, ESA is an unsupervised behavioural annotator that produces a probabilistic annotation. Events were selected that were labelled as Omega turns but had a high entropy ($0.75H_{max} \leq$H), i.e. high classification uncertainty. In total, 158 events met this criteria, and 20 were selected randomly (*Supplementary video* SV1 shows the predetermined clips).

Adaptive threshold finding

To deploy an adaptive threshold finding technique, it was necessary to have a single metric by which Omega turns could be ranked. We developed a 'tightness' metric score based on the Omega turn detection algorithms in the literature. Most Omega turn detection algorithms recognise such events when a certain feature exceeds a user-defined threshold. Features that are commonly associated with Omega turns are solidity, midbody angle, head-tail distance and midbody bend. For a visual explanation of each of these features, see Figure 2.1.

For each Omega event, the peak amplitudes of these features were measured. Across all events, the z-score was calculated for each feature peak and the tightness score of each event was the mean z-score across the four features. This procedure ranks the Omega-like events from wide amplitude turns to the sharper, more 'characteristic' Omega turns. It is not claimed that the tightness score captures every variation of Omega-like events. However, the score quantifies the sharpness of coils, which is

a key feature of turning behaviours. For a demonstration of the ranking results, see *Supplementary video* SV2.

To estimate each expert's threshold (measured using the tightness score) for detecting Omegas, the truncated staircase method was used [Treutwein, 1995]. The equation for selecting the next clip was

$$T_{n+1} = T_n - \delta(2R_n - 1) + z, \tag{2.3}$$

where δ is a fixed step size (in tightness score), T_n is the tightness of the clip shown in the n^{th} step, R_n is the n^{th} response ($R_n = 1$ if the answer is yes and $R_n = 0$ if the answer is no) and z is a small random variation to avoid repetitions. In this process, the sequence of clips has either an increasing or decreasing T_n until a switch in the subject's response (from yes to no, or no to yes) for successive clips occurs. In this case, the step direction is reversed and again the stimulus strength (T_n) monotonically increases or decreases until the next switch in response. To estimate the threshold, the average T_n at the points where the subject switched responses was taken.

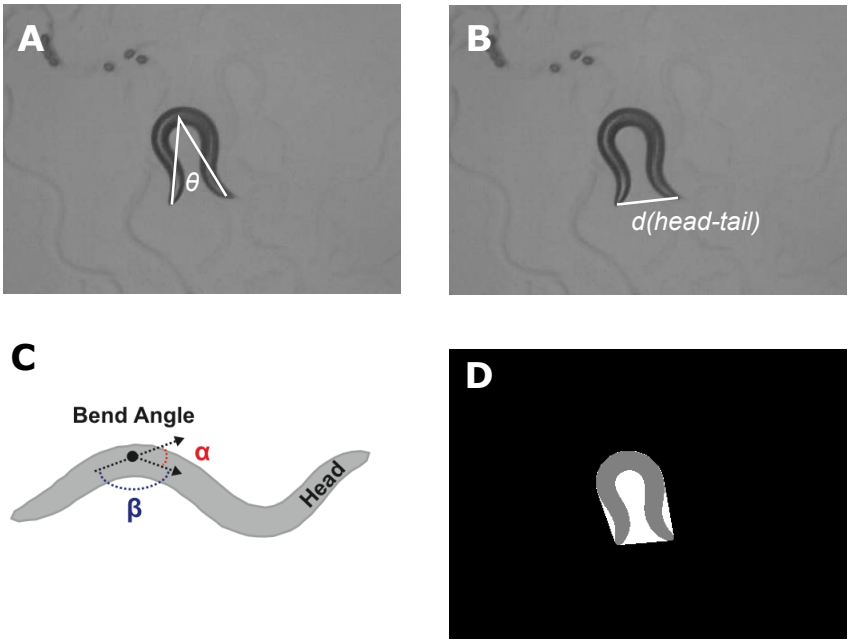


Figure 2.1: Visual explanation of the features used to construct the tightness score. Panel **A** shows the midbody angle θ, which is the angle between the head-middle and middle-tail vectors. Note that $\pi - \theta$ is the angle of reorientation of the event [Huang et al., 2006]. Panel **B** shows the head-tail distance. **C** illustrates worm bending that is measured using the supplementary angles to the bends formed along the skeleton. The bend angle (α) is the difference between tangent angles at each point; or, alternatively phrased, the supplementary angle (α) with respect to the angle formed by any three consecutive points (β). To detect Omega turns, the midbody bend is calculated; this is the mean supplementary angle along the middle 1/3 of the worm's body (image and caption are taken from [Yemini et al., 2013]). Finally, Panel **D** introduces solidity, a measure of the overall concavity. It is defined as the ratio of the image (the worm's body in grey) and the area of the convex hull (shown in white).

2.3 Results

2.3.1 Consistency of Omega detection algorithms

The consistency of four Omega turn detection algorithms was quantified. In Table 2.1, the precision, sensitivity and F-score of the methods are presented relative to one another. These measures showed little consistency, with an average F-score of 0.3. The same statistics are also presented after tuning the parameters of two methods for optimal match in outputs (for details of the tuning procedure, see Section *Threshold tuning*). Note that even in this case, the F-score frequently stayed below 0.5, indicating poor consistency in classification.

The algorithm developed by [Laurent et al., 2015] produced the worst match to the other algorithms. This is due to the method only picking out the sharpest of Omega turns, and hence identifying far fewer events than the other methods. It is not argued that any of the methods assessed is better or worse than the others. Rather, the point is that results can differ significantly depending on which method a particular analysis uses.

	Huang 2006	Yemini 2013	Salvador 2014	Laurent 2015
Huang 2006	1/1/1	0.40/0.46/0.43 (0.64/0.65/0.65)	0.28/0.15/0.20 (0.52/0.38/0.43)	0.13/0.67/0.22 (0.79/0.69/0.74)
Yemini 2013	0.46/0.4/0.42 (0.66/0.67/0.67)	1/1/1	0.45/0.22/0.29 (0.66/0.43/0.51)	0.05/0.21/0.08 (0.92/0.69/0.79)
Salvador 2014	0.15/0.28/0.20 (0.48/0.52/0.43)	0.26/0.5/0.34 (0.47/0.71/0.56)	1/1/1	0.12/0.1/0.11 (0.62/0.83/0.77)
Laurent 2015	0.68/0.13/0.22 (0.64/0.79/0.74)	0.22/0.05/0.1 (0.7/0.93/0.8)	0.62/0.1/0.13 (0.83/0.72/0.77)	1/1/1

Table 2.1: Consistency of Omega turn detection algorithms. Each column represents the reference algorithm against which the algorithm in a given row is evaluated. In each cell, the $Precision/Sensitivity/F-score$ are reported for the given combination of reference and evaluated algorithm. For a description of these measures, see Section *Consistency quantification*. The numbers in parentheses in each cell report the same statistics with thresholds tuned for optimal match. See Section *Threshold tuning* for further details.

2.3.2 Consistency of expert annotation

Overall, 27 survey responses were collected during the period 2016 May 30 - June 14. For the results presented here, we discarded the responses of participants whose self-reported confidence in detecting Omega turns was below 4. Thus, only expert annotation is analysed (19 participants in total).

As described in the *Methods*, the survey had two components: a set of predetermined clips and an adaptive threshold finding procedure. Figure 2.2**A** shows the distribution of answers for the predetermined clips, which had been selected for high classification uncertainty according to an unsupervised behavioural annotator (see *Methods*). None of these clips received unanimous consensus, and only 6 were judged the same by more than 75% (at least 15 out of 19) of the experts. Almost half of the clips produced a split of 12:7 or worse.

The estimated decision thresholds for each expert and the corresponding 95% confidence intervals are shown in Figure 2.2**B**. The size of the confidence interval reflects the number of samples used to estimate the threshold, which depended on the number of switch points from yes to no (and vice versa) for each subject in the sequence of 20 presentations (see *Adaptive threshold finding*). The size of confidence intervals also serve as an indicator of the subject's internal consistency, as it reveals whether the staircase quickly converged to oscillate around a specific value. It is clear that the estimated thresholds were spread widely, with no region where the majority clustered, or where all confidence intervals overlapped.

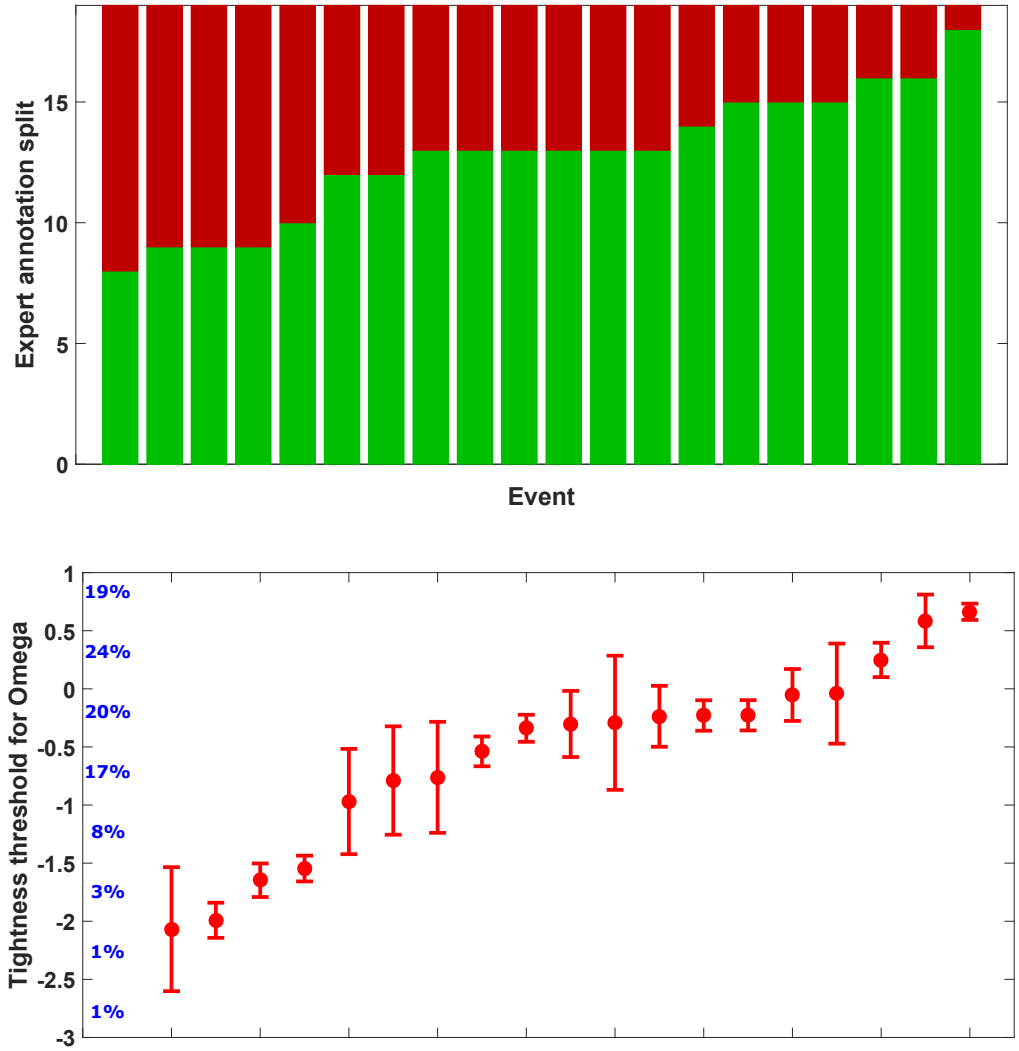


Figure 2.2: Outcomes of the Omega turn community survey. The data were filtered to exclude non-expert annotations (see the *Community response* for details). The top panel shows the results for the set of predetermined clips. The bars show how the expert annotation was split for each of the 20 clips (green: 'yes, it was an Omega'; red: 'not an Omega'). Bars have been arranged such that the proportion of 'yes' answers increases. Out of the 20 events 14 were split at 6-14 or more even, indicating that for the majority of events, there was no expert consensus. The bottom panel shows the results of the threshold detection. Each data point is an expert's estimated tightness threshold for detecting Omegas, with the corresponding 95% confidence intervals. The blue numbers next to the y-axis indicate the percentage of the data that falls within a certain z-score range (e.g. 19% of the events had a tightness score between 0.5 and 1). Note that the distribution is not normal; it is more concentrated at low z-scores. Also, no clear consensus for Omegas can be found in any region.

2.4 Discussion

In this paper, we have shown that both the automated and expert annotations of *C. elegans* Omega turns are surprisingly divergent. The implications of this for worm research are discussed in the current section. Then, some general comments regarding supervised behavioural analysis are presented. Finally, we speculate as to whether the observed annotation inconsistency is a more general feature of behavioural studies.

Characterising *C. elegans* behaviour often involves an estimate of Omega turn probability [Szigeti et al., 2015, Laurent et al., 2015, Yemini et al., 2013, Huang et al., 2006, Salvador et al., 2014, Pierce-Shimomura et al., 1999]. It is important to check whether the algorithms used to detect Omega turns are consistent. Otherwise, it would be difficult to make cross-publication comparison of results. It was found that the four Omega turn detection algorithms we tested produce surprisingly divergent annotations, even after their respective parameters have been adjusted for optimal match.

One way to overcome the inconsistency problem would be for the research community to adopt the same platform for behavioural analysis. There is a range of publicly available packages [Szigeti et al., 2015,Laurent et al., 2015, Yemini et al., 2013]. However, each comes with its own strengths and weaknesses, and thus it is difficult to see the entire community adopting any one of these methods. A potential solution would be an open-source software package that is developed and maintained not by a single laboratory, but rather by the entire research community. In this way, each lab would have ownership and the cross-talk between laboratories could lead to a deeper appreciation of the limitations of each analysis technique.

A potential source of the observed inconsistency is that the Omega turn is not a distinct behaviour, but rather a part of a spectrum of turning behaviours. We have previously argued for this possibility based on the high proportion of uncertain classifications of behavioural events [Szigeti et al., 2015]. Others have also provided support for this hypothesis based on the geometry of locomotion states [Gallagher et al., 2013] and the continuous neuronal representation of motor sequences [Kato et al., 2015].

A major limitation of both our earlier paper and the current publication is that events in which the worm intersected itself for an extended period could not be anal-

ysed (see *Methods*). A new method that can resolve coiling postures was recently developed by [Broekmans et al., 2016]. Their analysis of eigenworm amplitudes found a multi-modal distribution that could be used as a data-driven definition of Omega turns. Furthermore, the study reports that 'beyond' Omega turns, there exists another sharper turning behaviour, the Delta turn.

However, it should be noted that the experimental conditions in [Broekmans et al., 2016] were not identical to ours. In the CBD data (used here), the worms are browsing in food, while in the other study, the worms were analysed off-food. The 1^{st} and 3^{rd} eigenworms switch positions (sorted by eigenvalues) in these two conditions, indicating that the behaviour is altered (when off-food, the first two eigenworms are associated with locomotion and 3^{rd} is associated with turns; when on-food, the 1^{st} eigenworm corresponds to the turning postures) [Stephens et al., 2008, Yemini et al., 2013]. Therefore, the results may or may not generalise to other experimental conditions.

Our analysis of expert annotation has general implications for supervised approaches to behavioural analysis. The common element in these methods is that they take an investigator-labelled dataset and apply an algorithm that learns to reproduce the expert annotation [Kabra et al., 2013]. As a consequence, supervised methods can be only as consistent as their training data. Therefore, prior to using supervised methods, we would urge investigators to first examine the variability of expert opinions. Furthermore, we note that unsupervised methods are often evaluated against a human-produced 'ground truth' dataset. This evaluation process imposes subjective factors and hence leads to similar problems as with the supervised methods. The validation of unsupervised methods is a complex issue that raises many philosophical questions [Todd et al., 2016, Jain et al., 1999].

Although we have only analysed one specific behaviour of a single model organism, the observed inconsistencies in behavioural annotations (both expert and automated) seem likely to be more widespread. For example, there exists an analogous uncertainty about how to define the behavioural states of larval *Drosophila melanogaster* [Gomez-Marin and Louis, 2013, Kane et al., 2013, Gomez-Marin et al., 2011, Green et al., 1983, Szigeti et al., 2015]. Different publications use different ways of defining the behavioural states, most likely due to the difficulty of finding an unambiguous

characterisation. As a result, similar inconsistencies in the various analysis techniques should be a cause for concern as far as the reproducibility of maggot research. We hope that with our analysis we will have inspired investigators to look more carefully at the issue of consistency for other model organisms and behaviours.

Chapter 3

Eigenmaggots

3.1 Introduction

Behaviour may be thought of as a sequence of postures, although posture does not capture every form of behaviour. For example, animals often communicate through chemical signals, a form of behaviour in which posture is not involved. In this work, however, I am only concerned with the externally visible, physical actions of animals. Therefore, for the scope of this study, behaviour is approximated as the sequence of postures.

To represent larval *Drosophila* postures, I follow the eigenworm method, originally developed by [Stephens et al., 2008] to describe the postures of the nematode worm *C. elegans*. *C. elegans* has a flexible body without any joints; hence, its body shape space is potentially very high dimensional. However, using the eigenworm technique, all worm postures can be effectively described by only four numbers [Stephens et al., 2008]. Technically, eigenworms are the bases of a vector space that spans posture space. In a vector space, any point can be reached by adding up the bases; analogously, every actual worm posture can be described as a superposition of eigenworms. This can be written as:

$$posture(t) = \sum_{i=1}^{n} \alpha_i(t) eigenworm_i, \tag{3.1}$$

where $\alpha_i(t)$ is the coefficient associated with the i^{th} eigenworm at time t. Figure 3.1 provides a visual overview of how eigenworms are constructed and an example of posture reconstruction. Eigenworms provide a compact representation of posture,

and hence have potential use in behavioural studies. Specifically, behaviour (change in posture over time) is described by the time evolution of eigenshape coefficients, i.e. the time series of $\alpha_i(t)$s. Broadly, the strategy of this thesis is to take data from tracking assays, extract the time series of $\alpha_i(t)$s and then analyse them using statistical methods.

Eigenworms are an elegant representation of worm postures, but it is an open question as to whether the method can be generalised to larval *Drosophila*. The larvae are significantly more complex then *C. elegans*, as reflected by both their size and number of neurons: the worm is about 1 mm in length and has 302 neurons [Varshney et al., 2011], while larvae are approximately 5 mm, with an estimated 10,000 neurons [Shen et al., 2011]. Nonetheless, the two organisms share the same body architecture. Hence, at least in principle, the eigenworm method should be applicable to *Drosophila* larvae.

This chapter will demonstrate that the eigenworm method is indeed generalisable to larval *Drosophila*. First, the derivation of eigenmaggots (the larval equivalents of eigenworms) is described step by step. As an alternative to principal component analysis, non-negative matrix factorisation is considered and the resulting two sets of eigenmaggots are compared.

Note that eigenworms and eigenmaggots refer to the specific set of basis shapes for larval *Drosophila* and *C. elegans*, respectively. However eigenworms and eigenmaggots are sometimes collectively called eigenshapes when neither specific set is in mind.

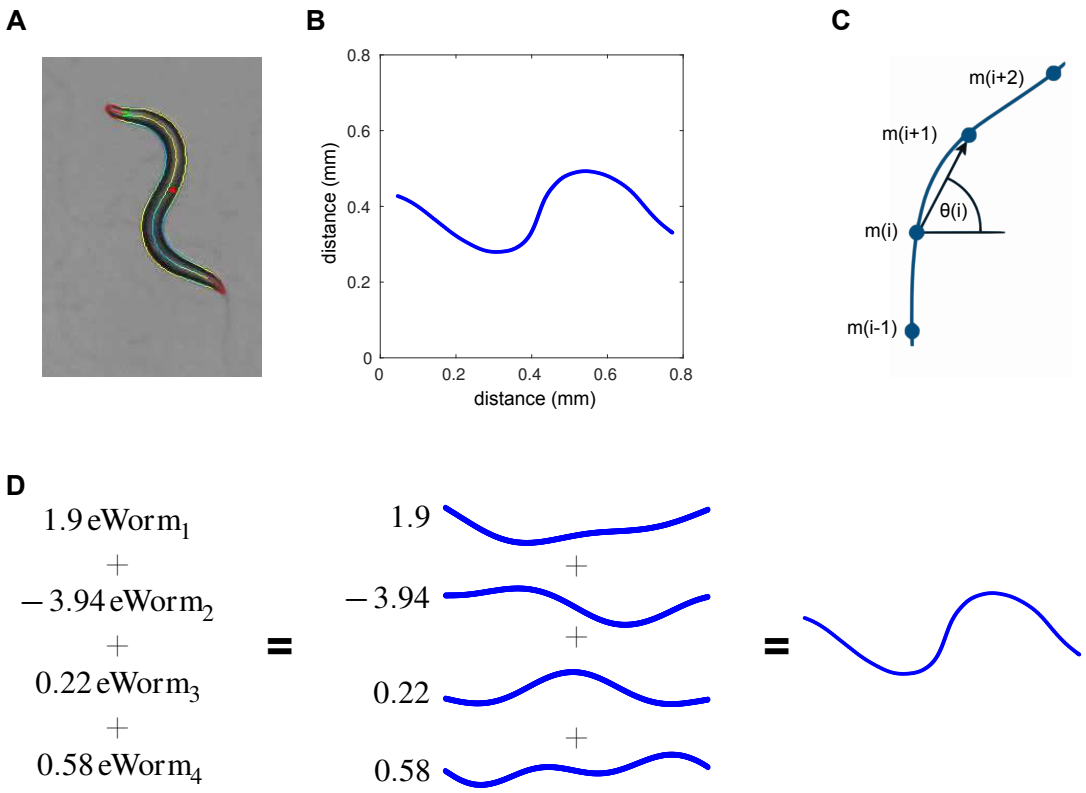


Figure 3.1: Overview of eigenworm construction. Panel **A** shows a frame from the *C. elegans* behavioural database, with the worm's contour and midline highlighted. Thresholding is used to separate the animal from the background, and the resulting binary images are then skeletonised. This skeleton is used as a proxy for the animal's posture. **B** shows the corresponding midline. The skeleton has been rotated to remove the worm's overall rotation relative to the plate. **C** zooms in on the midline, showing how a set of θ_i angles provide a piecewise linear approximation of the midline curvature. These angular data form a vector for each frame, or a matrix for a movie. The matrix's principal components are the eigenworms. **D** shows an example of a posture reconstruction. The blue shapes in the middle column are the eigenworms, which can be added together with different weights to reconstruct any actual worm posture.

3.2 Methods

3.2.1 Behavioural movies

Behavioural experiments on the larvae were conducted in Barcelona in Matthieu Louis's laboratory. Canton-S flies were maintained on conventional cornmeal-agar molasses medium at 22°C and kept under a 12-hour dark-light cycle. For the behavioural experiments, larvae were placed on 3 % agarose and were allowed to freely forage. Across 33 individuals, 14 hours of video were recorded at 30 fps. The tracking and data acquisition hardware used for the experiments are described in detail by [Schulze et al., 2015]. Briefly, each larva moving over a fixed stage was imaged using a camera (Basler A622f) positioned on top. The camera was mounted on a moving stage to follow the animal. The software for image capture and stage control was written in C using the OpenCV libraries.

3.2.2 Dropped frames

When the larva curls up, standard machine vision techniques fail to extract the biologically meaningful skeleton [Huang et al., 2006], leading gaps in the posture time series. In our experiments, if a gap was short (<0.5s), the posture parameters were linearly interpolated. After the interpolation, 4.2% of the *Drosophila* frames were missing. For an example of failed skeletonisation due to the maggot curling, see 3.2.

3.2.3 Skeletonising

In this section, the term 'midline' denotes the imaginary line along the anterior-posterior axis of the animal, running halfway between the lateral sides of the animal. The 'skeleton' is a set of points on a 2D plane, or on an image, and these points are the discrete representation of the underlying continuous midline.

To obtain the skeletons, every movie frame is first slightly blurred using a Gaussian filter ($r = 5, \sigma = 1$). The blurring is necessary to make the larva's body stand out from the background and to even out the otherwise variable lighting conditions. The blurred frames are thresholded, yielding a binary image that separates the background and the object, i.e. the larva's body. The standard MATLAB implementation of the Otsu algorithm [Otsu, 1975] is used to derive the threshold separating the background

and the object. In brief, the algorithm works by assuming a bimodal pixel intensity distribution and searches for the threshold value that minimizes the intra-class variance of pixel intensity values, or equivalently, the threshold that maximizes inter-class variance of pixel intensity values.

Contamination on the plate or uneven agarose surface can sometimes lead to the detection of false objects. To eliminate this source of error, only the largest connected object is kept, while all other foreground pixels are added to the background.

The binary images of the frames are skeletonised using standard MATLAB morphology function. The algorithm works by successively removing the outer layer of pixels from the object until a single line, the skeleton, remains (recall that there is only a single object on every binary frame corresponding to the body of the larva). The resulting skeletons neighboring pixels are connected either by their edges or corners. Since each square pixel has 4 edges and 4 corners, such a line of pixels is called 8-connected. To check whether the skeletonisation was successful, the following tests are performed:

1. a frame is rejected if it does not contain exactly two endpoints, i.e. points that have only one neighbour. If a frame has more then two endpoints, the skeleton is branching, which is non-physical given the body structure of maggots. If the skeleton has no endpoints, the skeleton is a circle, which is also non-physical

2. a frame is rejected if the length of the skeleton is greater than 4σ away from the mean length of all skeletons. If the skeleton is suspiciously long or short, the binarisation algorithm has most likely failed to separate the larva from the background

Figure 3.2 shows examples of successful and failed skeletonisation. The skeleton was obtained successfully for 96.4% of the frames. The main source of error was the frames where the animal was turning sharply; these events are called ‘doughnut turns’ (for an example, see Figure 3.2B). Other than doughnut turns, there were rare single-frame errors where parts of the agarose plate were confused with the larva’s body.

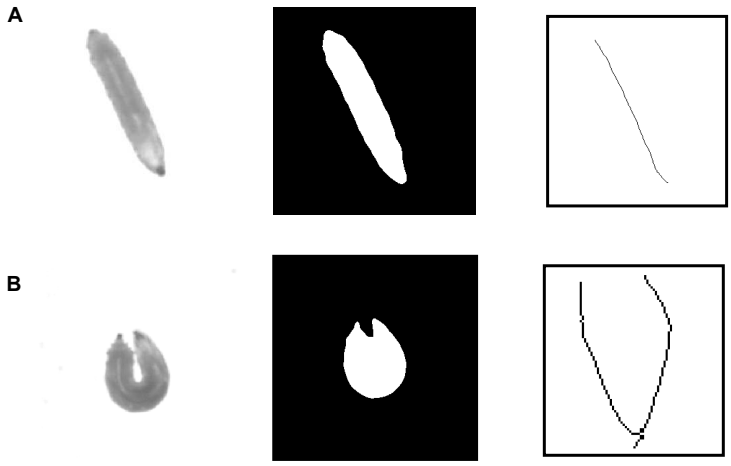


Figure 3.2: Examples of successful and failed maggot skeletonisations. Panel **A** shows an image of a larva, the binary image of the larva, and the resulting skeleton going from left to right. The same are shown in **B** for a maggot during doughnut turn. In this case, the skeletonisation has produced multiple branch-points and hence has failed. Note that for Panel **B**, the skeleton on the right has been zoomed into for better visibility.

3.2.4 Skeleton normalisation

To mathematically analyse postures, a numerical representation of the skeletons is needed. Consider the vector

$$\vec{\theta} = [\theta_1, \theta_2, ..., \theta_n], \tag{3.2}$$

where θ_1 is the angle between the 1^{st} and the 2^{nd} pixel of the skeleton (relative to the x-axis), θ_2 is the angle between the 2^{nd} and the 3^{rd} element of the skeleton, and so on (see Figure 3.1C for an illustration). $\vec{\theta}$ can be used to provide a piecewise linear approximation of the midline. However, before the angles θ_i can be measured, the skeletons must be normalised. These normalisation procedures are described below along with the problems they aim to solve.

Redistribution of skeleton points

To be useful for further analysis, the dimensionality of $\vec{\theta}$ should be constant. Lets call this constant n. Furthermore, in order for $\vec{\theta}$ to provide a piecewise linear approximation of the midline, points of the skeleton should be equidistant from each other. These two requirements (constant dimensionality of $\vec{\theta}$ and equidistance of skeleton points) can both be fulfilled by redistributing points along the skeleton.

To redistribute the points, a spline is first fitted to the set of pixels making up the initial skeleton. This spline is treated as the midline. A new skeleton is constructed by lying n points equidistant from each other along the midline. Successive points are found by 'drawing circles' from the previous skeleton point (starting from the head). A new skeleton point is placed where the midline and circle cross. The radii of the circles are initially calculated as follows:

$$r = \frac{arclength(midline)}{n}. \tag{3.3}$$

Given the Euclidean distance between the last skeleton point and the tail of the midline, the algorithm either increases or decreases by r and then a skeleton is again constructed. These two steps (fitting the skeleton points, adjusting r) are repeated until the final skeleton point and the end of the midline are within one pixel distance from each other. Typically, the algorithm converges after 2-3 iterations.

The only input parameter for the algorithm is n, the number of points that make up the normalised skeleton. Ideally, n should be set such that all information about the curvature of the midline is preserved, but also so that redundant information is not introduced by oversampling. To find the ideal number of points, the 'elbow method' was used. See Figure 3.4C, where n is plotted against the error of skeletonisation, and the error metric is the area between the midline and the piecewise linear skeleton constructed from $\vec{\theta}$. The figure suggests that not much information is gained by increasing n beyond ≈ 70. Therefore $n = 71$, resulting in 70 skeleton segments, was used for all further analysis.

The redistribution of skeleton points also solves the problem of 'discrete angles'. By definition, skeleton points are 8-connected, meaning that each pixel has 8 possible neighbors (neighbor pixels are connected either by their edges or corners). As a consequence there are only 8 possible values for θ_i: $0, \pm\pi/4, \pm\pi/2, \pm 3\pi/4, \pi$. Based on the redistribution of skeleton points, the skeleton is resampled, with the discrete angle distribution replaced by a continuous one. Note that the maggot's body in the video is made up of approximately 130 skeleton points, so by redistributing 71 skeleton points, the final skeleton is down-sampled. For a visual illustration, see Figure 3.3.

Note that while constructing the eigenmaggots, it is implicitly assumed that the larva has the same length in every frame. However, this is not true, as the larva is continuously stretching and contracting during its peristaltic locomotion. The eigenmaggots do not capture this variation. If needed, $length(t)$ could be reintroduced as an additional parameter of the posture description, but in this thesis, the variation in length is ignored.

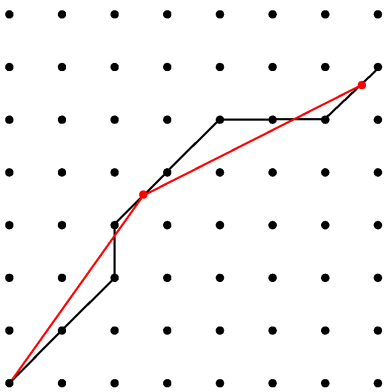


Figure 3.3: Illustrating the discrete angle distribution of skeleton points. The black dots are the set of possible skeleton points and the black line is the original skeleton. Note that there are only 8 possible connections from any skeleton point. Therefore there are only 8 possible values for θ_i, resulting in a discrete angle distribution. The skeleton is resampled (red dots and red line), replacing the discrete with a continuous distribution for the possible θ_i values.

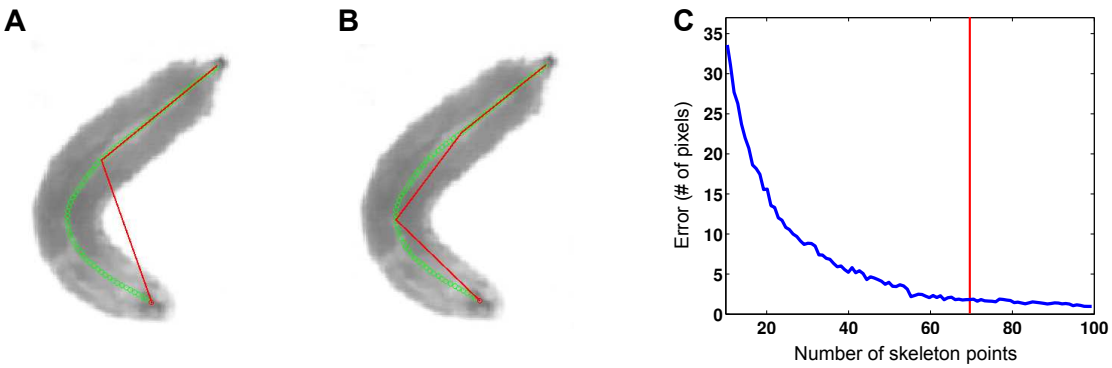


Figure 3.4: Number of skeleton points vs. skeletonisation error. In Panels **A** and **B**, the green circles show the pixels making up the skeleton. The red lines show the resulting skeleton if 2 or 3 points (Panels **A** and **B**, respectively) are used to construct it. To evaluate the accuracy of the skeletons, the area between the midline and the skeleton is used as an error metric (the area between the set of green dots and the red line). Panel **C** shows the error against the number of skeleton points. The error shown is the average pixel area per frame. Initially, the error drops quickly, but the improvement then diminishes. The vertical red line crosses the x-axis at 71, the number of skeleton points used for further analysis.

Ordering of skeleton points

The next problem is ensuring that components of $\vec{\theta}$ always represent the angle between the same points of the skeleton, i.e. the 1^{st} component of $\vec{\theta}$ should always be the angle between the 1^{st} (head) and the 2^{nd} points of the skeleton, the 2^{nd} component of $\vec{\theta}$ should always be the angle between the 2^{nd} and the 3^{rd} elements of the skeleton, and so on.

A semi-automated head-tracking algorithm has been developed to guarantee that the ordering of the skeleton pixels is consistent. In the first frame the endpoint corresponding to the head is annotated by the user. For all successive frames, the Euclidean distance between the endpoints of the current skeleton and the skeleton of the previous frame is calculated. If the distance between the 1^{st} element of the previous skeleton and the last element of the current skeleton is smaller than the distance between the 1^{st} element of the previous skeleton and the 1^{st} element of the current skeleton, the ordering of points (on the current skeleton) is flipped. This algorithm works robustly because of the high recording speed of the behavioural videos (30fps), meaning that the head and the tail do not move much between consecutive frames.

This algorithm works well if the head identity is never lost, but in the videos there were consecutive frames in which the skeleton could not be obtained and hence the head identity was lost. Specifically this happened during 'doughnut turns', see section 3.2.3. For gaps smaller than 7 consecutive dropped frames ($\approx 0.25s$), the Euclidean distance to the last skeleton is used (where last skeleton means the skeleton of the last frame in which a skeleton could be obtained). For larger gaps, the algorithm is reset by asking the user to again annotate which end is the head. In practice, gaps larger than 7 frames happen exclusively during doughnut turns (see Figure 3.2 for an illustration of such an event).

In summary, the algorithm requires the user to annotate the head in the first frame and then again in every frame that is preceded by a doughnut turn. For a visual overview of the head tracking algorithm, see Figure 3.5.

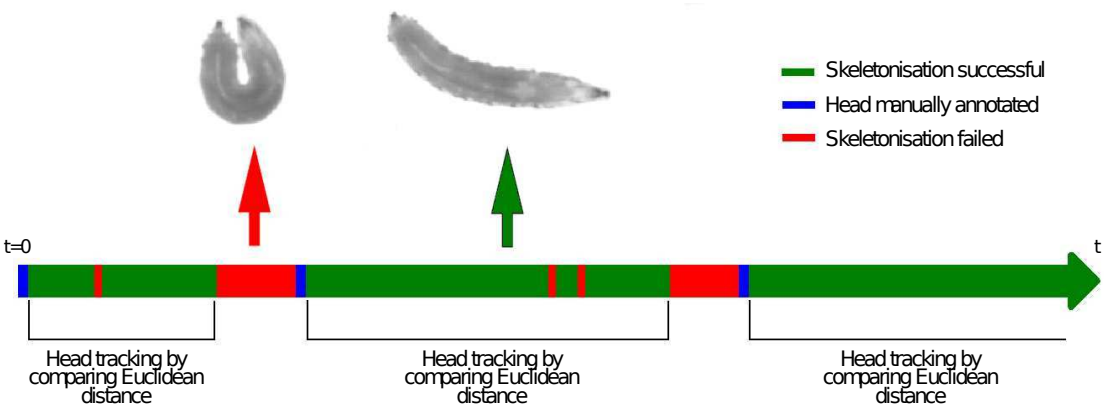


Figure 3.5: Schematic explanation of the head tracking algorithm. In the first frame of a behavioural video, the head is annotated by the user. In successive frames, the head is identified by comparing the Euclidean distance between the endpoint of the current skeleton and that of the skeleton of the previous frame (or the last successful frame in the case of short gaps). This works robustly until the algorithm encounters a doughnut turn, which causes a long gap. If that happens, the user is asked to again provide manual annotation of the head.

Orientation on plate

A maggot in a particular posture can be facing any direction. Thus, postures may look different but are fundamentally the same if they are just rotated versions of the same underlying posture. Therefore, the animal's orientation relative to the plate should be removed from the skeletons. To achieve this goal, all skeletons are rotated using the following steps:

1. Translate every point with $-(x_{71}, y_{71})$, where x_{71} and y_{71} are the coordinates of the tail. This translation places the tail at the centre of the coordinate system.

2. Measure the angle between the x-axis and the 1^{st} element of the skeleton (the head). This angle is called β.

3. Apply the rotation matrix $R(-\beta)$ to every point of the skeleton.

These steps position the head and the tail on the x-axis. With the tail in the center and head towards the positive half of the x-axis, all skeletons are right-facing.

Note that the rotation described here is not necessary if one uses the relative angle between consecutive skeleton points. The codebase I have used for this thesis measures angles relative to the x-axis, hence this normalization process was required. Note that these schemes to measure angles are a matter of convention (between what lines the angles are measured), the results they yield are identical.

Furthermore, alternative strategies can be used to remove the animal's orientation. For example, [Stephens et al., 2008] and [Yemini et al., 2013] normalise the skeletons to give a mean angle of 0. These normalisation schemes result in a different orientation of the reconstructed skeletons, but the corresponding eigenmaggot coefficients, the α_is in Eq. 3.1, are the same. Therefore, for practical purposes, these normalisation schemes are equivalent to each other.

3.2.5 Dimensionality reduction

So far, $\vec{\theta}$ has been constructed for each frame of a behavioural movie, where $\vec{\theta}$ is a 70-dimensional vector with each element representing an angle between consecutive elements along the skeleton. Hence, $\vec{\theta}$ encodes a piecewise linear approximation of the midline, which is used as a proxy for the animal's posture.

It is easy to see that the elements of $\vec{\theta}$ are not independent. As with the worm, the larva's body lacks any fixed joints and therefore neighbouring elements of the skeleton are highly correlated. This observation suggests that the effective dimensionality of the midline is potentially much less than 70. To exploit these correlations, the eigenworm method applies principal component analysis (PCA) to the matrix. For the case of *C. elegans*, PCA revealed that only four eigenworms account for $> 95\%$ of the original data's variance. In practice, this means that the coefficients of the first four eigenworms ($\alpha_1, ..., \alpha_4$ in Eq. 3.1), provide a near complete representation of worm postures.

Before proceeding further, it should be noted that PCA is applied to the matrix Θ, which is an aggregate of the $\vec{\theta}$ vectors. Specifically, the t^{th} row of Θ is $\vec{\theta}^t$, which is the $\vec{\theta}$ corresponding to the t^{th} frame of the movie:

$$\Theta = \begin{bmatrix} \vec{\theta}^1 \\ \vec{\theta}^2 \\ \vdots \\ \vec{\theta}^t \end{bmatrix} = \begin{bmatrix} \theta_1^1 & \theta_2^1 & \cdots & \theta_n^1 \\ \theta_1^2 & \theta_2^2 & \cdots & \theta_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ \theta_1^t & \theta_2^t & \cdots & \theta_n^t \end{bmatrix}. \tag{3.4}$$

Note that in equation 3.4 above, the upper index is an index of time, while the bottom is the index of the skeleton point ordering (see 3.2). In summary, Θ is a matrix that represents how the angles among consecutive skeleton points evolve over time.

This section describes how PCA is applied to the larval Θ to check whether a similar reduction in dimensionality occurs as does for the worm. Along with PCA, an alternative dimension reducing technique, non-negative matrix factorisation, is considered. Both of these techniques are standard practices in the literature. Hence, only brief overviews of the methods are provided here. For details, see the corresponding references.

Principal component analysis

PCA is a rotation of the basis states of the data space (i.e. the vector space in which the observations, that is Θ, are described). The principal components are the resulting set of basis vectors after the rotation. Note that the basis states are orthogonal, and therefore the data in the rotated space is linearly uncorrelated. This is the geometric interpretation of PCA, but in practice the principal components are found by solving the eigenvalue equation of the data covariance matrix, as follows:

$$cov(\Theta)\mathbf{x} = \lambda \mathbf{x}, \tag{3.5}$$

where λ and $\mathbf{x}$ are the eigenvalues and eigenvectors associated with Θ. The eigenvectors, $\mathbf{x}$, are the principal components or the basis of the rotated space, i.e. the eigenmaggots. The principal component associated with the highest eigenvalue accounts for the maximum amount of possible variance in the data, the component with second highest eigenvalue aligns with the maximum of the remaining variance, etc. This property means that, by definition, PCA accounts for the greatest amount of variance given a fixed dimension of the representation [Jolliffe, 2002].

Non-negative matrix factorisation

Non-negative matrix factorisation (NNMF), also known as non-negative matrix approximation, is a group of related algorithms in multivariate analysis [Lee and Seung, 1999]. Generally, the goal is to factorise a given non-negative matrix Θ into non-negative factors $\mathbf{W}$ and $\mathbf{H}$. This can be written as

$$\Theta \approx \mathbf{WH}, \tag{3.6}$$

where the rows of $\mathbf{H}$ can be interpreted as the eigenmaggots, and the elements of $\mathbf{W}$ are the weights or the coefficients analogous to the α_is in Eq. 7.1. Note that eq. 3.6 requires elements of Θ to be all positive, because all elements of $\mathbf{W}$ and $\mathbf{H}$ must be positive as well. This is guaranteed as the skeleton angles are not measured on the $-\pi \leq \theta_i < \pi$ interval, rather they are defined on $0 \leq \theta_i < 2\pi$ going around the unit circle.

Equation 3.6 cannot be solved analytically for all cases. Hence, some iterative approximation method is typically used. We used the MATLAB implementation of NNMF that attempts to minimize the root-mean-squared residual between Θ and $\mathbf{WH}$ [Berry et al., 2007]. The non-negativity constraint can be advantageous in many situations. For example, unlike PCA, NNMF yields physically plausible larval postures as bases.

3.3 Results

3.3.1 PCA and NNMF eigenmaggots

Either PCA or NNMF can be applied to Θ. Both methods result in a set of basis states and associated coefficients, which together allow the reconstruction of the sequence of postures encoded by Θ. In this section, these two families of eigenmaggots are compared.

See Figure 3.6 for the eigenmaggots derived using PCA and NNMF. With either method, only four eigenmaggots account for $> 90\%$ of the original data's variance. Therefore, just like for *C. elegans*, larval body shapes are also low dimensional. See Table 3.1 for the exact percentage of variance recovered using both methods for a given dimensionality of representation. Both methods are efficient (in terms of the original data's variance recovered for a given dimensionality of representation), but the PCA and NNMF derived eigenmaggots look different from one another. Notably, using PCA, the 3^{rd} and following eigenmaggots are not physical, i.e. the larva never actually adopt these shapes in reality. In contrast, the NNMF derived eigenmaggots do not suffer from the same problem due to the non-negativity constraint.

If the eigenmaggots were analysed in isolation, the physicality of the NNMF shapes would be a strong argument for using the NNMF derived eigenmaggots. However, eigenmaggots are always combined in the subsequent analysis. By definition, the combination of eigenmaggots is physical, as they correspond to the shapes observed in the videos.

Dimensionality	PCA	NNMF
1	0.728	0.668
2	0.89	0.78
3	0.93	0.858
4	0.954	0.91
5	0.964	0.923
6	0.972	0.941

Table 3.1: Fraction of the original data's variance recovered given the dimensionality of the PCA/NNMF representation.

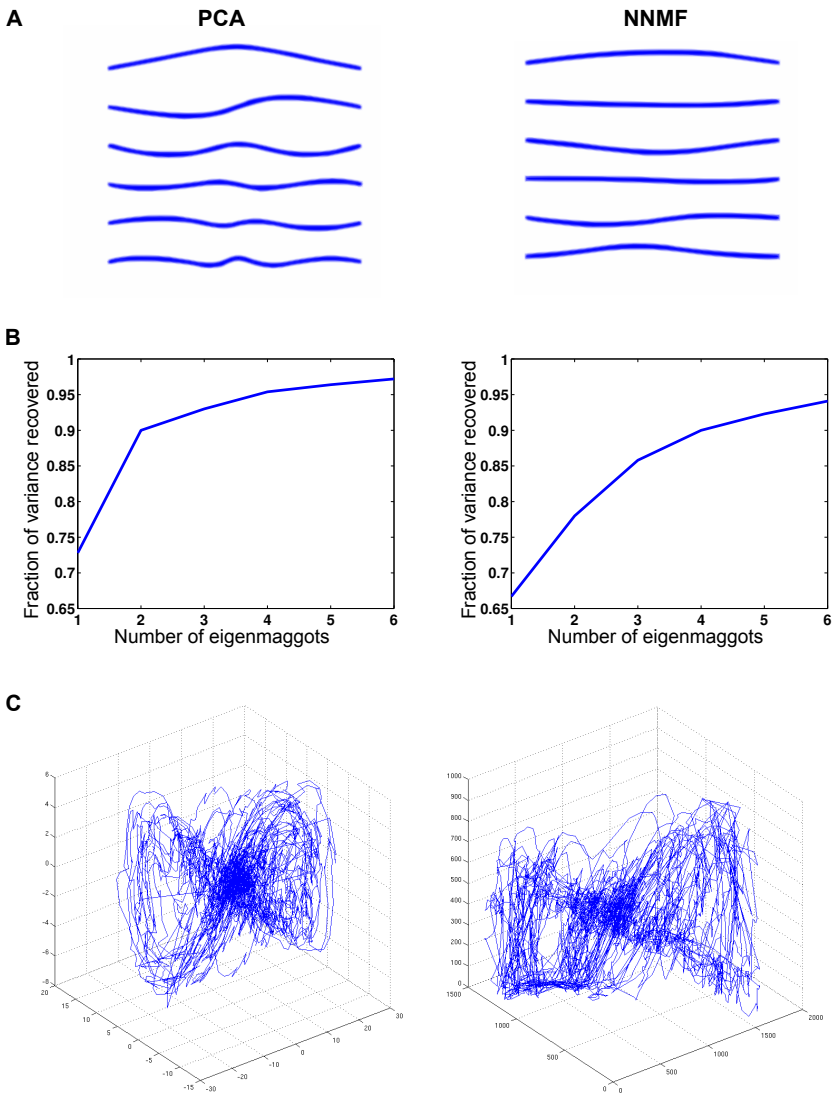


Figure 3.6: Comparison of NNMF and PCA derived eigenmaggots. The left side of each panel shows the graph of PCA derived eigenmaggots, while the right side shows their NNMF counterparts. **A** shows the first six eigenmaggots, with the most significant one on top. Notice that while the PCA eigenmaggots are non-physical (i.e. the maggot never adopts any of these shapes), the NNMF shapes are plausible. **B** shows how much of the original data's variance is accounted for by a given number of eigenmaggots. The PCA curve rises more steeply and quickly plateaus ≈95%. See Table 3.1 for the exact percentages. **C** shows the trajectory corresponding to a 15-minute long movie plotted in a 3D phase space. Each axis corresponds to one of the first three eigenmaggots, and the eigenmaggot coefficients ($\alpha_i(t)$s in Eq. 3.1) are plotted. In both representations, the trajectory corresponds to a 'butterfly' shape, where the wings correspond to wider amplitude turns.

We expected the behaviour to be left/right symmetric, corresponding to the positive/negative symmetry in the eigenmaggot coefficients. According to this property, two postures are left/right symmetric if the eigenmaggot coefficients are equal in magnitude but have opposite signs. This symmetry is lost for the NNMF representation because the coefficient matrix **W** can hold only non-negative elements. The loss of symmetry introduces some artefacts to the NNMF representation. For example, consider Figure 3.6C, which shows the trajectory across NNMF eigenmaggot space. The bottom of the trajectory is clearly 'clipped', and the postures that would be represented by negative values are represented by lower-ranked NNMF eigenmaggots (where 'rank' refers to the variance accounted for by a given dimension).

Given that the PCA has a better compression rate, and since it has already been used for the analysis of *C. elegans*, the PCA-derived eigenmaggots are used for the rest of the thesis.

There are numerous non-linear dimensionality-reducing techniques that could have additionally been tested. The reason I did not do a more through survey is twofold. First, PCA eigenmaggots already allow the construction of a low dimensional representation of the larval postures, which was the goal in the first place. Second, a recent review of dimensionality-reducing techniques stated that *'From the results obtained, we may conclude that non-linear techniques for dimensionality reduction are, despite their large variance, often not capable of outperforming traditional linear techniques such as PCA.'* [van der Maaten et al., 2009]. Therefore, it is unlikely that any method would provide a significant advantage over PCA.

3.3.2 Eigenshape coefficient time series

Irrespective of the specific basis set, the following equation can be used to reconstruct the posture in any frame:

$$posture(t) = \sum_{i=1}^{n} \alpha_i(t) eigenshape_i, \tag{3.7}$$

where $\alpha_i(t)$ is the coefficient associated with the i^{th} eigenworm/eigenmaggot at time t. Figure 3.1 shows an example of a posture reconstruction for *C. elegans*. Eigenshapes provide a compact representation of posture and hence clearly have potential for use in behavioural annotation. Specifically, behaviour (change in posture over time)

is represented by the time evolution of eigenshape coefficients, i.e. the time series of $\alpha_i(t)$s. This time series is referred to as the eigenshape coefficient time series (ECTS) and forms the basis of our method.

For the rest of this work, for both larval and worm analysis, the coefficients of the three most significant eigenshapes are included in the ECTS, that is, $ECTS(t) = [\alpha_1(t), \alpha_2(t), \alpha_3(t)]$. After principal component analysis, inspection of the eigenvalues reveals that for both organisms, three coefficients account for approximately 90% of the posture variance [Jolliffe, 2002], thus providing a sufficiently accurate description of posture. At the same time, a three-dimensional ECTS is small enough to avoid 'the curse of dimensionality' that could lead to difficulties during the subsequent analysis [Verleysen and François, 2005] (see Chapter 6).

3.3.3 Interpreting eigenmaggots

For *C. elegans*, each of the first 3 eigenworms (see Figure 3.1D) can be loosely associated with a behaviour. The third eigenworm is a bent shape; high values of this component indicate turning behaviours, while an oscillation between the first two eigenworms is a signature of locomotion. Does a similar mapping between eigenmaggots and behaviour exist for larval *Drosophila*? The first eigenmaggot is a bent shape and is thus associated with turning, but we do not find eigenmaggots associated with crawling. This difference can be understood by looking at the difference between the worm and the larva locomotion (see Figure 3.7).

C. elegans propels itself by moving its body in a near sinusoidal wave perpendicular to the direction of motion [Boyle, 2009]. This sinusoidal motion corresponds to an oscillation between the first two eigenworms. If body shapes during locomotion are considered, then the joint probability distribution of the first two eigenworm coefficients shows a ring structure. Locomotion corresponds to eigenworm coefficients going around on this ring, as shown in Figure 3.7B&C.

Drosophila larvae, on the other hand, crawl around utilizing peristaltic contraction waves consisting of periodic strides [Heckscher et al., 2012]. Each stride has two phases. First, abdominal segments remain planted on the agar while a wave of extension propagates from the posterior towards the anterior of the body. During this phase,

the head and tail translocate, pushing forward the centroid of the larva. This phase is followed by a second step in which the head and the tail remain fixed on the agar while the abdominal sections are pulled in the direction of the crawl. Thus, while the phases of locomotion are best distinguished by the position of abdominal sections, their location is undetectable using top-view videos (see Figure 3.7A). Therefore, unlike the worm, none of the eigenmaggots reflect the phases of locomotion.

The eigenmaggots do not capture locomotion, but it is easy to label frames in which the larva are translocating by calculating $tailSpeed$, which is the discrete time derivative of the distance covered by the tail. Care has already been taken to establish which end of the skeleton is the tail, and so obtaining $tailSpeed$ is a straightforward matter. The tail only moves during the first phase of the stride; hence, $tailSpeed$ can also be used to dissect the phases of locomotion.

In the literature, the phases of larval locomotion are often identified from the skeleton length. However larvae are constantly bending, swirling and turning, all of which have an effect on the skeleton length. For this reason, I have found $tailSpeed$ to be a more reliable signature of the stride phase.

It is noted here that the ordering of the eigenshapes is indicative of the eigenshapes's relative importance for reconstructing postures. This effect is because the ordering of the eigenworms corresponds to the magnitude of the associated eigenvalues, which in turn are proportional to the variance accounted for by the given eigenvector. For example, when the worm is off-food, the 1^{st} and 3^{rd} eigenworms swap positions (compared to the on-food case from Figure 3.1), indicating that the worm is spending more time turning while on-food. Therefore one should be careful to construct a new set of eigenshapes whenever behaviour is analysed in a new experimental setting.

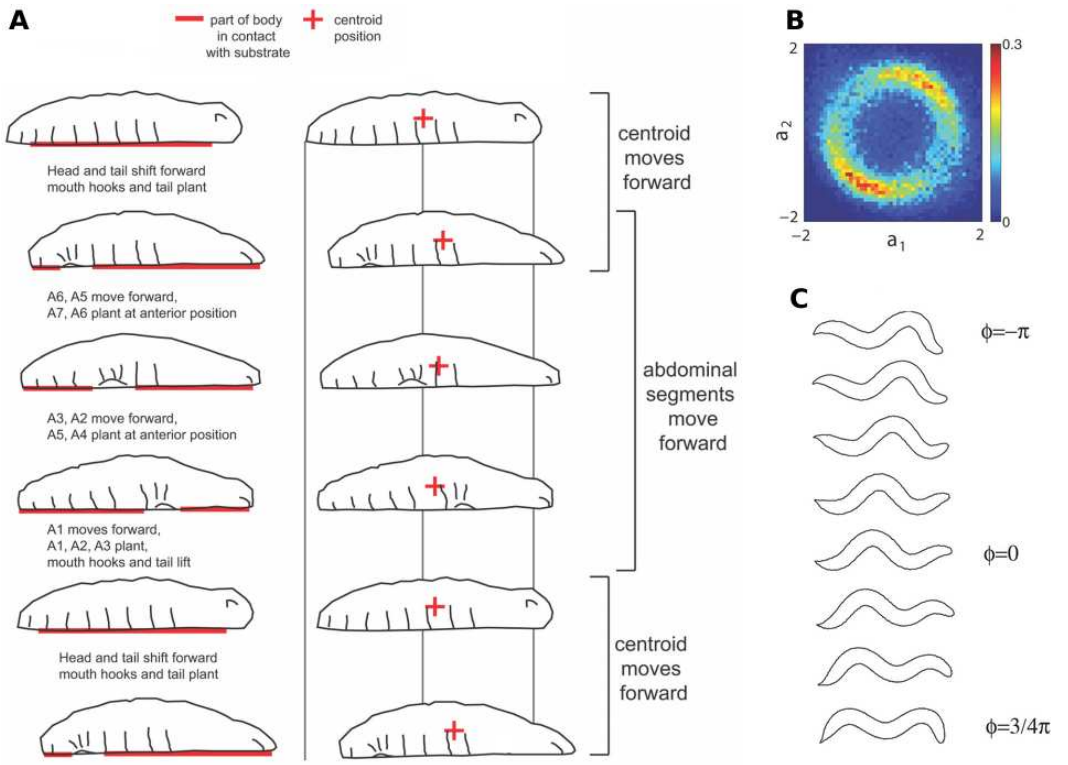


Figure 3.7: Comparing the locomotion of *Drosophila* larva and *C. elegans*. **A** shows the different phases of a stride during larval locomotion. A1, A2, etc. refer to the abdominal segments of the organism. Panel **B** show the joint probability density of the first two eigenworms's amplitude during locomotion of *C. elegans*. There is a clear ring structure, indicating that locomotion is an oscillating superposition among these modes. **C** shows the body of the worm during various phases of locomotive oscillation (**A** is taken from [Heckscher et al., 2012], while **B** and **C** are taken from [Stephens et al., 2008]).

3.4 Discussion

The central theme of this thesis is analysing the behaviour of larval *Drosophila* and *C. elegans* as the continuous change of postures. Traditionally, the posture of these animals has been described by a set of morphological features such as length, width, etc. (for concrete examples, see the papers discussed in the literature review, Section 1.2.1). Note that these features are perceptually important to us, but they may or may not be relevant to the animal's actual behaviour. For example, there are natural variations in the width of animals, meaning that differences might not be behaviourally significant. A further problem is that this 'feature list' is typically a high dimensional description that makes analysis difficult.

[Stephens et al., 2008] introduced eigenworms, a highly efficient scheme for representing *C. elegans* postures. The method approximates the worm's posture using its midline, employing principal component analysis to reduce the dimensionality of the midline. Recall that any actual worm posture can be reproduced as a superposition of eigenworms:

$$posture(t) = \sum_{i=1}^{n} \alpha_i(t) eigenworm_i, \tag{3.8}$$

where $\alpha_i(t)$ is the coefficient associated with the i^{th} eigenworm at time t. Note that eigenworms are the basis of the space where the postures are described, and the eigenmaggot coefficients(α_i) denote how much of each eigenmaggot contributes to the overall shape in a given frame. Examination of the eigenvalues reveals that four eigenworms capture 94% of the variance in midlines. Hence, eigenworms provide a low dimensional, but nearly complete, description of *C. elegans* posture. In other words, eigenworms allow *C. elegans* postures to be represented as low dimensional vectors, and behaviour (change in postures over time) to be described as a trajectory across this low dimensional vector space.

The question this chapter has aimed to answer is whether the eigenworm method can be generalised for use with larval *Drosophila*. Using the same pipeline as [Stephens et al., 2008], it was shown that only four eigenmaggots account for $> 95\%$ of the variance in the animal's posture. Hence, the method does allow the construction of a low dimensional representation of larval postures. The advantage of constructing the eigenmaggots is that we can now describe the behaviour of both *Drosophila* and *C.*

elegans using the same conceptual language. It is hoped that, based on this general representation, a general behavioural analysis tool can be built.

Possible points of criticism are that the second-ranked and lower eigenmaggots are non-physical (i.e. the maggots never adopt those shapes), and that the eigenmaggots cannot be related to specific behaviours. While it is true that some of the eigenmaggots are non-physical, the eigenmaggots never appear in isolation in the analysis that follows. Eigenmaggots are always combined, and the combinations are, by definition, physical. Similarly it is expected that eigenmaggots themselves are not necessarily related to behaviours. Eigenmaggot composition only captures single postures, while behaviour is dynamic. Therefore, to capture behaviour, the time series of eigenmaggot coefficients should be analysed.

After establishing the eigenmaggots as a valid numerical representation of posture, behaviour can be conceptualised as a multidimensional time series. Specifically, behaviour (change in posture over time) is represented by the time evolution of eigenshape coefficients, i.e. the time series of $\alpha_i(t)$s in 3.8. This time series is referred to as the eigenshape coefficient time series (ECTS) and forms the basis of my method. Next, I turn to the problem of identifying recurring patterns, i.e. motifs, in the ECTS. ECTS motifs represent recurring sequences of postures that can potentially be identified as behavioural states [Green et al., 1983].

Chapter 4

Motif discovery

4.1 Introduction

The aim of this thesis is to develop a method for the unsupervised discovery of behavioural states, where a behavioural state is understood as a frequently repeated behaviour. It was argued in 3.1 that behaviour can be approximated by the change in posture over time. Hence, *frequently repeated posture sequences* are analogous to *frequently repeated behaviours*. The eigenshape coefficients represent posture and therefore, behavioural states can be represented by frequently repeated subsequences in the ECTS.

In the data mining literature frequently repeated subsequences are also known as motifs [Fu, 2011]. Therefore the technical objective of this thesis is the unsupervised motif discovery in the ECTS. Due to the abundance of genetic sequencing data there is a great variety of motif discovery algorithms for univariate discrete data. Unfortunately significantly less attention has been paid to motif discovery in multivariate continuous data such as the ECTS. There are only a handful of tools available and most of the proposed methods are a variation of the 'sliding windows' approach. For an overview of the proposed methods see Section 1.2.2 in the Literature review.

There are several variations of the sliding window approach (e.g. [Chiu et al., 2003, Lonardi and Patel, 2002, Mueen et al., 2009, McGovern et al., 2011]), but for a more focused discussion, I will focus on a variation of the Mueen-Keogh (MK) algorithm [Mueen et al., 2009]. Here, the codebase shared by [Brown et al., 2013] is used, which is a minimal extension of the MK method. This extended MK is referred

to as the 'MK dictionary' approach and it is presented in 4.2.2. At several points in this thesis, the performance of the MK dictionary algorithm is compared with other approaches.

The reason for comparing the MK dictionary approach against the alternatives is twofold. First, this code is publicly available, so the results are easy to reproduce. Secondly, [Brown et al., 2013] also aimed to discover motifs in the eigenworm coefficient time series. Hence, it is a rational choice for a baseline performance against which our method can be measured ([Brown et al., 2013] is discussed in detail in 1.2.1).

Note that the discussion focuses on the MK algorithm, but the main conceptual criticisms presented in 4.2.3 apply generally to other sliding window methods as well.

In 4.2.4, a second extension of the MK algorithm, called 'MK reference,' is discussed. This motif discovery technique uses MK as a subroutine and relies on the collection of all subsequences within a threshold distance of the average of the closest neighbours. While this approach is not used in the later parts of this thesis, it is briefly discussed here, as it demonstrates that overcoming the conceptual problems of the MK algorithm is not a straightforward task.

The reader should note that, somewhat confusingly, three similar terms (*MK algorithm*, *MK dictionary*, *MK reference*) are used in this chapter to refer to three separate algorithms. These methods are discussed and criticised in 4.2.1, 4.2.2 and 4.2.4, respectively. Briefly, the *MK dictionary* and *MK reference* are both motif discovery pipelines that use the *MK algorithm* as a subroutine.

Before presenting my alternative to MK-based approaches in 4.3, the impact of the choice of distance measure is discussed. Note that unless otherwise stated, 'distance' in this chapter refers to Euclidean distance. It is argued that the use of the dynamic time warping distance measure could theoretically improve the measure of similarity, but the choice of distance metric cannot help overcome the conceptual shortcomings of the MK framework.

In 4.4, a novel alternative motif discovery approach is proposed that avoids the pitfalls associated with the MK. This method is based on a two-step process: first the time series is segmented into windows, and then the subsequences are clustered. This method is thus referred to as segmentation-clustering, or SC in short. The viability of the SC approach is demonstrated by discovering motifs within synthetic data. This test provides a proof of principle before applying the method to the more challenging real-life data.

4.2 Motif discovery with the MK algorithm

4.2.1 The MK algorithm

The brute force version of the sliding windows motif-finding algorithm takes two subsequences (or windows) with a fixed length w, slides both windows over the entire time series, and measures the distance between every possible pair of subsequences. The pair of subsequences with the smallest distance is treated as a motif (see the pseudocode below).

The brute force algorithm has a computational cost of $O(m^2)$, where m is the number of possible subsequences for a given length of time series and window size ($m \approx length(timeSeries)/w$, where w is the window size). This cost is prohibitive, and hence most of the literature is focused on finding approximate motifs (e.g. [Mohammad et al., 2012, Oates, 2002]). The MK algorithm is an exception, as it is guaranteed to find the same solution as the brute force method at a significantly faster speed. Therefore, to understand the motifs found by MK, it is sufficient to understand the output of the brute force algorithm.

To speed up the search process, MK uses two techniques. The first is known as early abandoning. If the partial distance between two subsequences is already larger than the best distance so far (that is, the smallest distance between subsequences calculated so far), then the distance computation is abandoned. The second technique is based on the observation that if data points are projected onto a random axis, the distance between adjacent pairs along the linear projection is a lower bound of the true distance among them. This can be used to rule out many candidate subsequences without having to compute the full Euclidean distance.

Before moving on, the problem of trivial matches must be discussed. Consider two subsequences offset by a single time point:

$$z_t = [y_{t-(w/2)}, \ldots, y_{t+(w/2)}], \tag{4.1}$$

$$z_{t+1} = [y_{(t+1)-(w/2)}, \ldots, y_{(t+1)+(w/2)}]. \tag{4.2}$$

It is likely that no other subsequence is closer to z_t than z_{t+1}, given that they are offset by a single time point (the other trivial candidate for closest subsequence is z_{t-1}). Hence, z_t and z_{t+1} are called a trivial match. To avoid trivial matches, MK requires a

minimum distance of w between two candidate subsequences in order to be classified as a closest neighbour. Note that this means that there are no overlapping elements; each point in the time series can belong to only one motif.

Brute force sliding window motif discovery

Function $[i,j]$=BRUTEFORCEMOTIFSEARCH(ts,w,thr)

Input: ts - time series
w - window size
thr - maximum distance among closest neighbours

Output: $[i,j]$ - starting locations of closest pair subsequences

---

for t = 1 to length(ts)-w **do**
  $ss(t) = ts(t:(t+w))$ ▷ Collect all subsequences

best-so-far = Inf
while best-so-far $<$ thr OR best-so-far = Inf **do**
  for i = 1 to m **do**
    for j = i+1 to m **do** ▷ Check all subsequences
      if d(ss_i,ss_j) $<$ best-so-far **then** ▷ Check distance
        best-so-far = d(ss_i,ss_j) ▷ Overwrite smallest distance

---

4.2.2 The MK dictionary

[Brown et al., 2013] uses a minimal extension of the MK algorithm to find motifs in the eigenworm coefficient time series. The MK are iteratively applied at different user-defined length scales. Initially, the MK algorithm is called using the window size w_1 that returns the pair of subsequences with the shortest distance. The MK is then called again to find the pair with the second shortest distance and so on until the closest neighbours do not exceed the user-defined threshold. This procedure is repeated for every length scale. In this way, a set of closest neighbours pairs is collected for each length scale w_i. Finally, a 'pruning' parameter called δ is introduced. Once all motifs are collected for all length scales, only the closest $\delta\%$ of the pairs is kept. For an overview of the algorithm, see pseudocode below.

Function $[motif_k]$=MKMOTIFDICTIONARY(ts,$[w_1, w_2, ..., w_{max}]$,thr,δ)

Input: ts - time series

$[w_1, w_2, ..., w_{max}]$ - vector of window sizes

thr - maximum distance among closest neighbours

Output: $motif_k$ - catalogue of motifs; at each k a separate set of subsequences is stored as examples of the same motif

---

for k = 1 to length(w) **do** ▷ Loop through all scales

  while d(ss_i,ss_j) $<$ thr OR [ss_i,ss_j]=empty **do**

    [ss_i,ss_j]=MK(ts,w_k,thr) ▷ Find MK motif subsequences

    if d(ss_i,ss_j) $<$ thr **then** ▷ Check distance

      $motif_k = motif_k + [i,j]$ ▷ Add subsequences to motif entry k

$motif_k$ = PRUNING($motif_k, \delta$)

4.2.3 Problems with the MK dictionary approach

For the results below, the MK dictionary code was applied to 3 hours of maggot behaviour. For an overview of the experimental conditions, see 7.2. Note that the MK dictionary code was run on the dataset, but the criticism presented here equally applies to the output of the plain MK algorithm.

Closest neighbours are not motifs

Motifs are interesting because they are conserved structures in the data and hence could potentially provide insight into the underlying generative processes. However, sliding window methods do not identify conserved structures, but rather find the pair of closest neighbours, that is, the two subsequences that are closest to one another. To redefine motifs as closest neighbours is methodologically convenient, but it contradicts how motifs are intuitively understood and makes the results difficult to interpret.

Using the MK dictionary led to the identification of 2223 'motifs' for wild type and mutant *C. elegans* ([Brown et al., 2013]). This is a suspiciously large number, since it is three orders of magnitude greater than usual estimates of the dimensionality of worm behaviour [Albrecht and Bargmann, 2011, Salvador et al., 2014]. This discrepancy is accounted for by the closest neighbours definition of motifs. The algorithm finds 2223 closest neighbours, but many of these are examples of the same behaviour.

Similarly, applying MK dictionary to the eigenmaggot coefficient time series yields 287 motifs, of which at least 100 are head cast behaviours (based on visual inspection). All of these closest pairs should form a single motif corresponding to head cast! The problem is that the MK dictionary discovers motifs as pairs and hence it does not recognise when two pairs are examples of the same motif.

In summary, the sliding window methods find the closest neighbours, but to identify closest neighbours as motifs is misleading. The goal is to find frequently repeated patterns of behaviour, not to find pairs of behaviours (which are the output of the MK dictionary algorithm).

Motif lengths are not variable

Common to the sliding window methods is the use of a fixed window size (w); the algorithm can only find closest neighbours that are of the exact same lengths. Given the variability of biological systems, it is not surprising that behavioural elements do not have a well-defined time scale. Indeed, 'turning manoeuvres' (a behavioural state of larvae that is defined in 7.3.2) have a variable length scale, with $\mu = 0.83s$ and $\sigma = 0.27s$. The MK dictionary is therefore unable to discover all examples of this behaviour due to its fixed window size.

Bias in starting position of motifs

A prevalent problem with longer motifs is that distance from the reference can be distributed unequally over the subsequence. In many cases, this can lead to false detection of the motifs starting position. Consider a motif that is 300 frames long. Although the last 280 frames of a subsequence are a good match to the reference, the first 20 frames may be almost anything. Because there is little deviation in the last 280 frames overall, however, the 300 frames are under the threshold, irrespective of the content of the first 20 frames. In these situations, the subsequence is almost certainly a good instance of the motif, but the MK algorithm is triggered on the first frame for which the distance between two subsequences is smaller then the best found so far. Hence, the starting position is misidentified.

4.2.4 MK motifs as references

In this section, an extension of the MK algorithm that aims to solve the problems raised above is presented. The algorithm is based on my observations of how the MK algorithm behaves and on conversations with the author of [Brown et al., 2013].

The MK reference algorithm works as follows. Take TS, w and thr as the input; these are the time series, the initial motif length and the threshold value, respectively. The MK function is called to find the pair of most similar subsequences with length w. The closest neighbours are averaged to form a reference motif. Then, the algorithm searches for all subsequences (with length w) that are closer than thr to the reference motif. Both the closest neighbours and all subsequences within the thr distance are treated as a single motif. The algorithm continues until the closest neighbour subsequences are more than thr apart (see the pseudo-code below).

This algorithm relies on calling MK and using the resulting closest neighbours to form a reference motif. This approach is therefore called the 'MK-reference' algorithm.

Function [$motif_k$]=MKREFERENCEMOTIFSEARCH(ts,w,thr)

Input: ts - time series
w - window size
thr - threshold

Output: $motif_k$ - catalogue of motifs; at each k a separate set of subsequences is stored that represent the same motif

---

for t = 1 to length(ts)-w **do**
  $ss(t) = ts(t:(t+w))$ ▷ Collect all subsequences

[i,j] = BRUTEFORCEMOTIFSEARCH(ts,w) ▷ Find closest neighbours
$minDist$ = d(ss_i,ss_j) ▷ Get minimum distance
$refMotif$ = mean(ss_i,ss_j) ▷ Define reference motif

while $minDist < thr$ **do** ▷ Check distance
  $motif_k = motif_k + refMotif$ ▷ Start motif entry k
  for j = 1 to length(ss) **do**
    if d($refMotif$,ss_j) $< thr$ **then**
      $motif_k = motif_k + ss_j$ ▷ Add ss$_j$ to motif entry k

  $k = k+1$ ▷ Increase motif index
  $ts = ts$ - $motif_k$ ▷ Exclude motifs from ts
  [i,j] = BRUTEFORCEMOTIFSEARCH(ts,w) ▷ Find closest neighbours
  $minDist$ = d(ss_i,ss_j)
  $refMotif$ = mean(ss_i,ss_j)

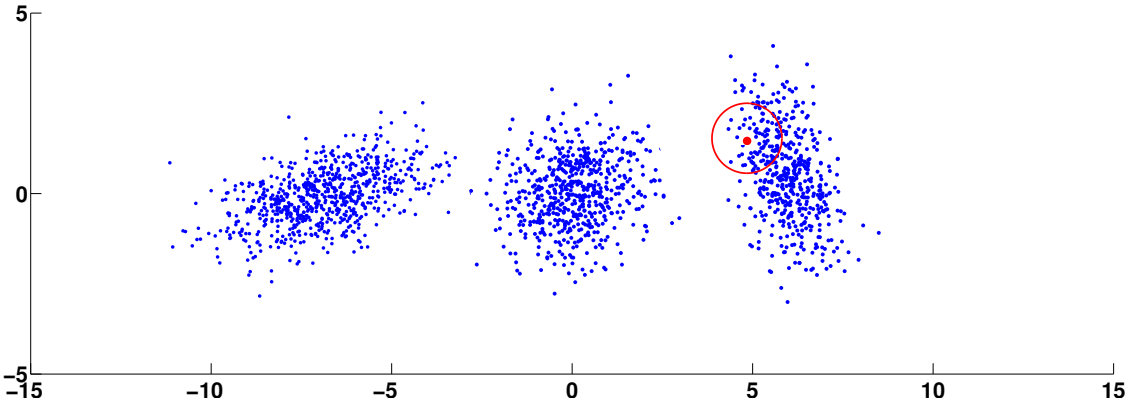


Figure 4.1: Schematic representation of the MK-reference motif discovery. Each point corresponds to a time series subsequence and the distance between points correlates with the distance between subsequences. In this figure, there are three well-separated clusters corresponding to three motifs. First, the MK algorithm is called to find the closest neighbours. Note that the closest neighbours can generally be located anywhere. These are averaged to form the reference motif, marked with the red dot. Next, every subsequence within thr of the reference motif is collected; these are all subsequences within a circle of radius thr (red circle). Note that in this picture, thr is too small, and hence the red circle does not capture the whole 'blob.' However if thr were increased, it could include elements of another cluster (corresponding to the middle blob). This highlights the difficulty and importance of selecting the right thr value.

The algorithm was implemented in MATLAB and tested on maggot behaviour. While the output of the algorithm is no longer just pairs of subsequences, the results remain unsatisfactory. Eighteen motifs have been identified, a significant reduction compared to the plain MK algorithm, but many of the motifs are still seemingly the same behaviour. The problem is that the reference motifs provided by MK are not necessarily representative examples.

4.2.5 Problems with the MK reference approach

Consider the space of subsequences, where each point represents a possible subsequence of w length. In the MK-reference algorithm, a reference motif is constructed and then all subsequences are found within a distance of thr. Geometrically, this means that a hypersphere is drawn around the reference motif.

The space of subsequences is generally a high dimensional space, but a 2D analogy is shown in Figure 4.1, the caption of which explains how the MK-reference works. When looking at the figure, note that the MK-reference algorithm can capture motifs if the following two conditions are met:

1. closest neighbours are always near the centre of clusters; and
2. clusters have a spherical shape in the data space.

However there is no guarantee that even one of these two conditions will be generally met; as a consequence, the reference motifs identified by MK could be misleading. If the reference motifs are not representative, the entire MK reference pipeline will be unreliable.

4.3 The impact of similarity measures

Empirical investigations have revealed that Euclidean distance is competitive with other computationally more expensive measures in many domains [Ding et al., 2008]. This is the primary justification for [Brown et al., 2013]s use of Euclidean distance to discover motifs, and why Euclidean distance has been used in this chapter thus far. However, it should be noted that, in principle, any other distance metric can be used.

I have extensively experimented with replacing the Euclidean distance in the MK approaches with the dynamic time warping (DTW) distance measure. DTW is a method that calculates an optimal 'warping' between two time series segments. The time series points are mapped non-linearly in the time dimension such that the distance is minimised, where 'distance' refers to the sum of the Euclidean distances along the warping path. For a visual comparison with Euclidean distance, see Figure 4.2. Unlike Euclidean distance, there is no one-to-one correspondence between points. Instead, one point can be mapped to many points. This mapping flexibility makes DTW a more

reliable measure when the time series segments may be off-phase or shifted.

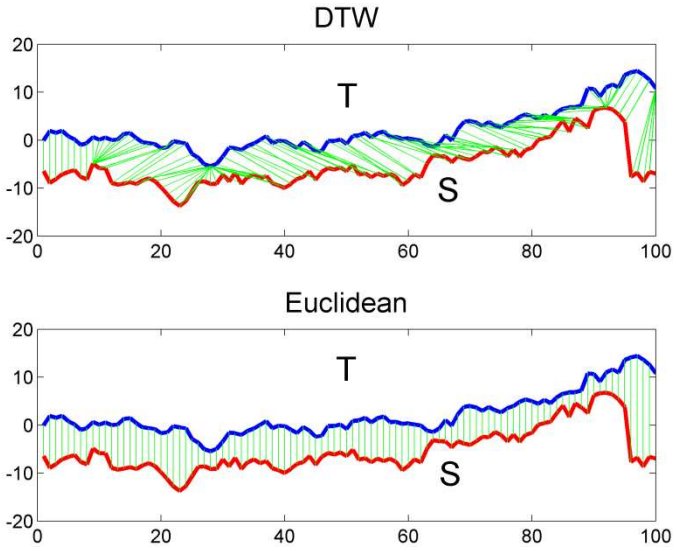


Figure 4.2: In both panels the green lines represent the mapping of points between time series S and T. The Euclidean and DTW distances are the sums of the length of the green lines in the corresponding panels. Due to its non-linear mapping, DTW is resistant to small shifts in the time series; hence, in theory, it represents an upgrade over Euclidean distance. Note that if S and T are identical, the two distance metrics will have the same value (figure adopted from [Cassisi et al., 2012]).

Substituting the DTW distance metric into the MK dictionary algorithm and applying it to the *Drosophila* eigenshape time series yielded 273 motifs. This number is still extremely large, and many of the different motifs still correspond to the same behaviour. This finding highlights that the problem with the MK dictionary approach is conceptual and cannot be fixed by using an alternative distance metric. Regardless of the distance metric used, the MK approaches rely on the closest neighbours definition; hence, the motif discovery will always result in a large number of overlapping motifs that do not capture the intuitive meaning of motifs (see 4.2.3).

It is noted here that the notion of distance in the alternative motif discovery framework that will be presented below is based on splines. Time series segments can be presented as splines (piecewise polynomial functions), and as a consequence, the distance is not measured between the time series points, but rather between the coefficients representing the time series.

4.4 An alternative method: Segmentation-clustering

It was argued in the previous section that sliding window methods are inadequate for the discovery of behavioural motifs, where a behavioural motif is understood as a frequently repeated behaviour. This section presents a novel motif discovery method, the segmentation-clustering (SC) algorithm, which avoids the pitfalls of the sliding windows methodology.

The key insight gained from the segmentation-clustering method is that motif discovery can be decomposed into two separate tasks. First, the subsequences of potential interest are found, and then the subsequences are clustered. The result is a set of clusters, each of which contains a set of frequently repeated subsequences. In other words, each cluster contains the instances of a motif.

Time series segmentation and clustering are both problems that have been explored in depth in machine learning. In SC, these steps are independent, meaning that the subsequences produced by any segmentation algorithm can be used as input into any clustering method. This flexibility is both a strength and a weakness of the proposed method. Therefore, SC is not proposed as a fixed algorithm, but rather as a framework that, when combined with domain knowledge, can be used to identify time series motifs.

The purpose of this section is to provide a high-level overview and proof of principle for SC. Therefore, it is merely stated here what segmentation and clustering methods were used. For the justification of why these particular methods were chosen, see the chapters 5 and 6, where alternative segmentation and clustering methods are explored.

4.4.1 Overview of segmentation-clustering

The intuition behind the segmentation algorithm is that boundaries between windows can be placed where the dynamics of a time series change. These points are identified by the local maxima and minima of the time series. A ‘behavioural window’ is defined as a local maximum bounded by two local minimums. These behavioural windows are passed on to the clustering algorithm as input.

The extracted subsequences are aligned in the time domain and clustered using a spline regression model [Gaffney and Smyth, 2004, Gaffney, 2004]. This method is analogous to Gaussian mixture models, but clusters are parameterised by splines instead of Gaussians. The spline parameters are learned using an expectation-maximisation algorithm. Bayesian Information Criteria (BIC) [Schwarz et al., 1978, Fraley and Raftery, 1998, Konishi and Kitagawa, 2008] are used to identify the optimal number of clusters. See Figure 4.3 below for a visual overview of the method.

Model based clustering has the advantage of producing a membership probability for each element (i.e. the probability that a given behavioural window belongs to a given cluster), rather than just a rigid cluster assignment. Therefore, this methodology allows classification uncertainty to be quantified. To measure the classification uncertainty, we used Shannon entropy [Shannon, 2001], which is defined as follows:

$$H = -\sum_i p_i log_2 p_i, \tag{4.3}$$

where p_i is the probability that a given action belongs to a cluster i. Note that the most uncertain situation is when the probability is equally distributed among the clusters. Correspondingly, H is at its maximum when all $p_i = 1/i_{max}$ (i_{max} is the number of clusters).

Note that like other studies discussed in the *Literature review*, SC implicitly accepts that it makes sense to discretise behaviour into elementary units, i.e. behavioural states. While this fundamental assumption is shared with the MK-based methods, there are crucial differences in terms of what is recognised as a motif. Specifically,

1. Possible motif lengths are defined by the data rather than by user input
2. SC can recognise instances of the same motif, even if they are not the same length
3. Most importantly, SC does not rely on the closest neighbours definition of motifs

These are the key novel features of my method. In 8, I further reflect on this list and evaluate the results as well as the shortcomings of SC after it has been applied to the eigenshape coefficient time series.

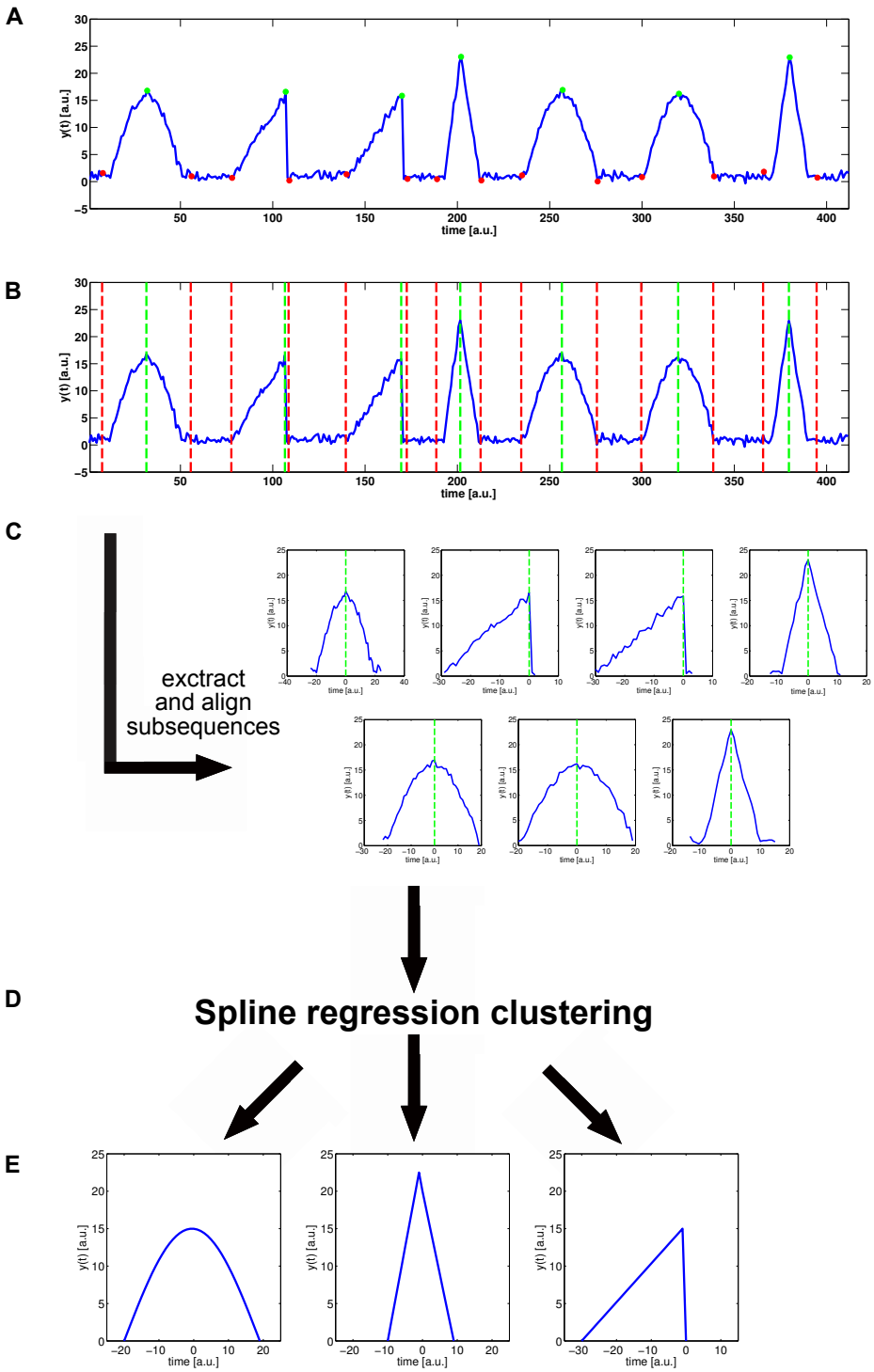


Figure 4.3: Schematic overview of the Segmentation-Clustering methodology using synthetic data (for details of data generation, see 4.4.2). First, the time series is segmented to yield the set of behavioural windows to be clustered. In Panel **A**, the red and green dots represent local maxima and minima of the time series. A behavioural window is defined as a local maximum bounded by two local minimums (one before and one after the local maximum), e.g. the green dashed line bounded by two dashed red lines in Panel **B**. The behavioural windows are aligned by their local maxima (green dashed lines) and fed into the spline regression algorithm. For this synthetic time series, the SC algorithm recovered the three motifs that were used to generate the time series.

4.4.2 Testing SC on synthetic data

Generating the synthetic time series

To demonstrate the feasibility of the method, the segmentation-clustering method was tested using synthetic data. A time series was generated that contained three motifs: triangles, right-angle triangles and sinusoids. In between these elements, the time series had a constant value of 0. The motifs were placed a random distance from one another; the distance was chosen from a normal distribution, with $\mu = 30$ and $\sigma = 10$ (both measured in frames). Additionally, noise was added to each frame, noise being modelled as a normal random variable with $\mu = 0$ and $\sigma = 1$. See Figure 4.3A for a sample of the time series. A synthetic time series of 100000 frames was generated in this way to test the SC algorithm.

Note that in this section, only a one-dimensional time series is analysed. In the following chapters, the method is generalised to multidimensional time series.

Results using the SC algorithm

For the current tests, the spline regression model was set up such that each spline had 3 knot-points and each polynomial had an order of 3. This meant that each subsequence contained within a behavioural window was modelled as two smoothly-joined cubic functions.

Using the SC pipeline, Bayesian information criteria indicated the presence of three motifs in the data (the issue of model selection and how the Bayesian information criteria were used is discussed in 6.2.2). A part of the time series was hand annotated, against which the annotation provided by SC was tested. Table 4.1 below summarizes the results. See A for a description how the matches are counted and for the interpretation of the terms *precision*, *recall* and *F-score* used in Table 4.1.

The maxima/minima-finding algorithm is controlled by a single master parameter. If this parameter is adjusted by $\pm 25\%$, then the F-score has a minimum at 0.92, which still indicates very high performance. Therefore, fine-tuning of the parameter is not required.

The algorithm is furthermore robust to noise if the random variable has a $\sigma < 4$. Note that each motif has a peak amplitude of 15. For this higher noise case, a new value for the master parameter of the maxima/minima-finding algorithm is selected. If $\sigma > 4$, random variations in the signal make it difficult to reliably segment the time series, resulting in a deteriorated classification. For classification accuracy at the $\sigma = 4$ noise level, see Table 4.2.

In conclusion, the SC motif discovery algorithm worked robustly on the synthetic data. However, it should be pointed out that in this synthetic time series, the instances of the motifs were highly stereotypical, and thus the near perfect performance is not indicative of the performance expected with real-life data.

	Triangles			R.A. triangles			Sinusoidal			All patterns		
	Pre.	Sen.	F	Pre.	Sen.	F	Pre.	Sen.	F	Pre.	Sen.	F
SC	0.99	0.96	0.98	0.99	0.93	0.96	0.99	0.93	0.96	0.99	0.94	0.96

Table 4.1: Summary statistics of the SC annotation of synthetic data. Precision, sensitivity and F-score values were derived from Table C.1.

	Triangles			R.A. triangles			Sinusoidal			All patterns		
	Pre.	Sen.	F	Pre.	Sen.	F	Pre.	Sen.	F	Pre.	Sen.	F
SC	0.76	0.66	0.71	0.73	0.69	0.71	0.71	0.66	0.69	0.72	0.68	0.69

Table 4.2: Summary statistics of the SC annotation of synthetic data for the high noise case ($\sigma = 4$)

4.5 Discussion

Behavioural motif and behavioural state are used interchangeably in this thesis; both refer to a frequently repeated sequence of postures. Frequently repeating subsequences in a time series are known as motifs. Therefore, to identify the behavioural states, we need to discover motifs in the eigenshape coefficient time series.

One of the most popular methods for multidimensional motif finding, due its simplicity and speed, is the MK algorithm. In 4.2.2, the minimal extension of the algorithm, as used by [Brown et al., 2013], is presented. This is the MK dictionary approach. In this thesis, the performance of the MK dictionary is treated as the baseline against which to measure our method. This is justified, as [Brown et al., 2013] also aimed to discover eigenshape coefficient time series motifs.

The MK dictionary algorithm was tested on maggot ECTS. Importantly, the MK algorithm is based on finding the closest neighbours, or the pair of equal-length subsequences that are closest among all possible pairs, and treating them as motifs. This definition is methodologically convenient, but contradicts the intuitively understanding of motifs. A motif is not a pair of closest neighbours, but a subsequence that repeats over and over again.

This poor definition of motifs explains why we found 287 larval behavioural states using the method, clearly an overestimate based on the literature (see Section 1.2.1 in the literature review). Our key observation was that many of the 287 behavioural motifs were different instances of the same underlying motif. Exemplars of the same motif are counted as separate motifs, leading to overestimation of the number of motifs in the time series.

At the early stages of this project, I experimented with how MK motifs could be merged to avoid overestimating the number of behavioural states. This line of research resulted in the development of the MK-reference approach (4.2.4), which was based on my own observations and extensive discussions with the author of [Brown et al., 2013]. Nonetheless, this extension also fails to capture motifs, as the method does not reliably detect the reference subsequences.

Due to the lack of available tools, a novel motif discovery methodology, the segmentation-clustering (SC) approach, was developed. It relies on a two-step process: first, subsequences of interest are extracted from a time series, and this set of subsequences is then used as the input for a clustering algorithm. The resulting clusters of subsequences correspond to frequently repeated subsequences, i.e. behavioural motifs.

To the best of my knowledge, the combination of segmentation and clustering represents a novel approach to motif finding. In SC, the segmentation and clustering steps are independent, meaning that the subsequences produced by any segmentation algorithm can be used as an input to any clustering method. This flexibility is both a strength and a weakness of the proposed method. Therefore, SC is not proposed as a fixed algorithm, but rather as a framework that, when combined with domain knowledge, can be used to discover time series motifs.

The key advantage of the SC methodology over the MK-based approaches is that it does not rely on the closest neighbours definition of motifs. As a consequence, a motif is not required by definition to have only two exemplars. Furthermore, the pipeline can deal with motifs of unequal lengths even if they are examples of the same motif.

In the next two chapters, both the segmentation and clustering steps of our proposed algorithm are evaluated in detail. In both cases, several algorithms are tested against one another to justify our choice of segmentation algorithm and the use of spline regression clustering.

Chapter 5

Segmentation

5.1 Introduction

Segmentation-clustering (SC) is a two-step process in which the time series is first segmented into 'behavioural windows' and then these windows are clustered. This chapter takes a closer look at the segmentation step and aims to justify the use of dynamic change segmentation, a custom algorithm developed for this thesis.

The goal of segmentation is to produce behavioural windows that can be clustered later on. Bear in mind that we are segmenting the three-dimensional ECTS, and thus each segment contains three time series subsequences. Also note that throughout this chapter, the terms 'segments' and 'behavioural windows' are used interchangeably, both referring to ECTS subsequences.

Dynamic change segmentation (DCS) is a novel method that was briefly introduced earlier (see 4.4). In this chapter, the details of the algorithm are presented. DCS is inspired by a family of related segmentation algorithms called 'change point methods' [Liu et al., 2013, Cho and Fryzlewicz, 2015, Chu and Wong, 1999, Oliver et al., 1998, Fu, 2011]. The main idea behind these algorithms is that boundaries between behavioural windows can be placed at 'change points,' or points at which the dynamics of the time series are changing. The various algorithms differ in terms of how they define change points and how the change points are used to define the behavioural windows.

The application of change point segmentation methods is a novel technique for studying the behaviour of *C. elegans* and *Drosophila*. Change point segmentation methods have not been applied previously because in most studies, behaviours are detected by user-defined thresholds that also define the time series segments. It is important to point out that just like threshold methods, change point segmentation also implicitly assumes that it makes sense to discretise behaviour into elementary units, i.e. behavioural states. The crucial difference in our approach is that behaviours are not identified by an arbitrary user-defined threshold, but rather are discovered from the data.

The performance of the DCS is evaluated against the segmentation provided by the MK dictionary algorithm in Section 5.3. In this section, the motif identity of the time series segments provided by the MK dictionary approach is discarded and MK is treated as a segmentation algorithm. Expert annotation of the worm behaviour is taken as ground truth and the number of overlapping behavioural windows are counted to quantify the goodness of segmentation. While the goal is not to perfectly reproduce the human annotation, but human annotation can be used to guide the selection of segmentation method. Sensitivity analysis of both the DCS and MK segmentation methods is also provided to demonstrate that DCS does not require fine-tuning of its parameters.

5.2 Methods

5.2.1 Dynamics change segmentation

Consider a time series

$$y(t) = [\beta(1), \beta(2), ..., \beta(t)], \tag{5.1}$$

where each $\beta(t)$ element is an n-dimensional vector such that

$$\beta(t) = [\alpha_1(t), \alpha_2(t), ..., \alpha_n(t)], \tag{5.2}$$

i.e. $y(t)$ is an n-dimensional time series. To summarise, the multidimensional dynamics $y(t)$ are combined into a one-dimensional time series by taking the weighted average of the components. Mathematically, this is written as

$$z(t) = \frac{\sum_{i=1}^{i=n} \lambda_i \alpha_i(t)}{\sum_{i=1}^{i=n} \lambda_i}, \tag{5.3}$$

where the $\alpha_i(t)$s are defined in Eq. 5.2 and λ_i is the weight associated with the i^{th} component.

In the context of postural motifs, $y(t)$ is the eigenshape coefficient time series (the time series formed by the $\alpha_i(t)$s in 7.1), while $z(t)$ is referred to as the '*bodyScore*'. The *bodyScore* time series represents the overall dynamics of the multidimensional ECTS and hence can guide the search for change points.

To construct the *bodyScore*, the weights (λ_i's in Eq. 5.3) are set by the eigenvalues associated with each eigenshape. Eigenvalues correlate with how much of the original data's variance is accounted for by a given eigenshape. Therefore, this weighting scheme reflects how important each dimension is to the resulting posture (for additional details about the relationship between eigenvalues and variance, see 3.3.2).

As mentioned in the *Introduction* of this chapter, DCS is a change point segmentation method. Such algorithms are based on the idea that the boundaries between behavioural windows can be placed where the dynamics of the time series are changing. In the literature, fluctuations in many statistical features of time series (e.g. mean, standard deviation, correlation, etc.) have been used to define change points [Liu et al., 2013, Cho and Fryzlewicz, 2015, Chu and Wong, 1999, Oliver et al., 1998, Fu, 2011]. For example, a change point can be identified whenever there is a greater than threshold increase in the mean value of the signal (measured over a user-defined window size).

For this thesis, a simple and intuitive definition of change points was used: a change point is defined as a local maximum or local minimum in the *bodyScore* time series.

To find the local maxima and minima, we used a MATLAB code from the MATLAB community resources [peakFinder 2014]. The code requires a single parameter κ, which adjusts how much higher/lower a point must be compared to its local environment to be considered a local maximum/minimum.

With the change points identified, a rule defining the behavioural windows is needed. For DCS, a behavioural window is defined as a local maximum surrounded by two local minimums (one local minimum preceding and one local minimum following a local maximum). This rule was formulated based on a trial-and-error basis, where the evaluation criterion was comparison to the ground truth data. The DCS algorithm is summarised in Figure 5.1 and the pseudocode is shown below.

One point of criticism for the MK dictionary algorithm is that it uses windows with a predefined size (see 4.2.3). Note that for DCS, there is no restriction in the distance between the local minimums bounding the action. Therefore, unlike with the MK dictionary, the window lengths are determined by the data and not by a user-defined input.

Note that the DCS segmentation leaves gaps in the time series, so that not every frame is a member of a behavioural window. For example, in Figure 5.1, there is a period between the first two turns shaded in grey that is not a part of a segment. In this case, there are two consecutive local minimums without an intervening maximum, and hence no segment is formed. Similarly, two consecutive maximums without an intervening minimum would not be part of any segment. The gaps are discarded from further analysis and thus will not be associated with any behaviour by the eigenshape annotation pipeline.

For *Drosophila*, 82% of the time series was included in the DCS segments, and the gaps had $\mu = 10$ frames and $\sigma = 8$. Similarly, for *C. elegans*, 91% of the time series was covered and the discarded segments had $\mu = 16$ frames and $\sigma = 5$. For both organisms, the majority of the gaps were short in duration ($\leq$1/3s) and after visual inspection, were found to not correspond to any obvious behaviour.

Function [$segment_k$]=DYNAMICSCHANGESEGMENTATION(ts)

Input: ts - time series

Output: $segment_k$ - stores the first and last frame of the k^{th} segment

---

i = FINDLOCALMAXIMA(ts) ▷ Collect all local maxima in i

j = FINDLOCALMINIMA(ts) ▷ Collect all local minima in j

for k = 1 to i_{max} **do**

$start$ = FINDCLOSESTLOCALMINIMUM+(ts,i,j)

$finish$ = FINDCLOSESTLOCALMINIMUM-(ts,i,j)

$segment_k$ = [$start$;$finish$]

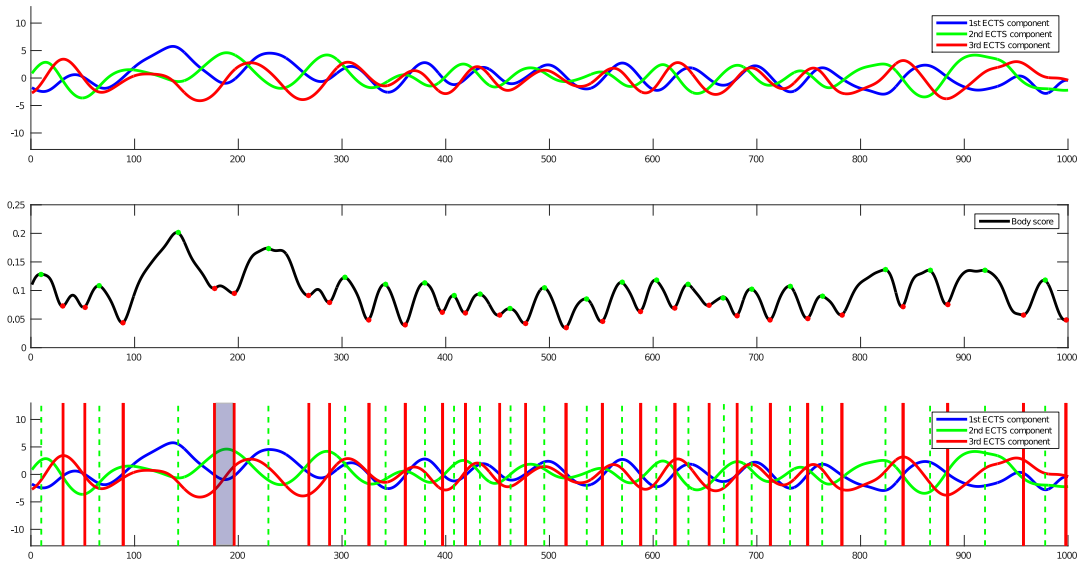


Figure 5.1: The dynamic change segmentation algorithm. Panel **A** shows the three-dimensional *C. elegans* ECTS. The dynamics of this multidimensional time series are summarised by the $bodyScore$ time series, which is shown in Panel **B** (see Eq. 5.3 for the relationship between ECTS and $bodyScore$). The local maxima and minima are denoted by the green and red points on the curve, respectively. A behavioural window is defined as a local maximum enclosed by two local minimums (one preceding and one following the local maximum). Panel **C** shows how the DCS segmentation projects onto the ECTS. In this panel the behavioural windows are the time series segments where a green dashed line (local maximum) is bounded by red lines (local minimums). Note that there are gaps in the DCS segmentation in Panel **C**. For example, there is no behavioural window in the area shaded grey, as it corresponds to local minimums without an intervening maximums.

5.2.2 MK segmentation

In Chapter 4, the MK dictionary motif-finding methodology is presented in detail. MK dictionary is a motif-finding algorithm, but it can be treated as a segmentation algorithm by ignoring the motif identity of the closest subsequences found by the algorithm (i.e. only using the information about the location of the MK dictionary motifs). At this step, the content and annotation of the behavioural windows are ignored and only their location in the time domain is evaluated.

As discussed earlier, there are many ways in which MK can be used as a subroutine of a motif-finding pipeline. For tractability, I have used the same code as was used in [Brown et al., 2013]. As this publication had a very similar goal to that of this thesis, its segmentation of the code was treated as the baseline against which to measure our method.

The MK dictionary algorithm, as used by [Brown et al., 2013], requires two parameters. These are:

1. *motifLengths* is a vector and the algorithm looks for motifs with lengths that are elements of the vector. The default setting is $motifLengths = [10, 20, 30, ..., 150]$, meaning that the algorithm finds motifs that are 10, 20, 30, etc. frames long (the videos are recorded at 30fps).

2. δ is the fraction of motifs kept. After the algorithm finishes looking for motifs of a given length, only the closest δ percentage of them is kept.

5.2.3 Quantitative comparison

The segmentation was validated by comparing it to the annotation of the *C. elegans* behavioural database (CBD). To produce Table 5.1 below, two videos from the CBD database were analysed. At this stage, only the segmentation was evaluated; hence, the behavioural labels (i.e. what is the worm was doing in the behavioural window) were ignored. To quantify the goodness of segmentation, the number of DCS/MK dictionary segments overlapping with the annotation windows of the CBD were counted. The more the windows overlapped, the better the agreement between the result of a segmentation method and the CBD annotation.

CBD is an automated behavioural annotation that is not based on general principles, but rather was built on domain knowledge. Hence, in an indirect way, CBD encodes the expert knowledge about the behavioural windows. Note that if we were to require our algorithm to produce exactly the same behavioural windows as the CBD, our segmentation would carry the same possible biases as the expert-produced CBD annotation. Therefore, the CBD segmentation should not be treated as absolute truth. Nonetheless, the number of overlapping windows can be used to guide the evaluation of the segmentation methods. For a more general discussion of the validity of ground

truth, see Chapter 8.

The ECTS components closely resemble sinusoidals during locomotion. The CBD annotation treats the whole sinusoidal segment as a single locomotive action, while the DCS algorithm resolves these oscillations into $\pi/2$ segments. See Figure 5.2 for a visual explanation of this difference. Note that all of the short step segments (as produced by the DCS) could be matched to the same CBD annotation window if they (individually) met the overlap criteria (for details of how the matches were counted, see Section A in the *Supplementary materials*).

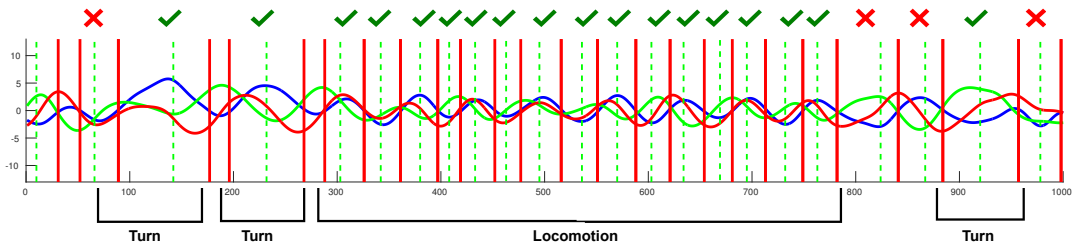


Figure 5.2: Counting the number of matching behavioural windows between the CBD and the DCS segmentation. The brackets at the bottom show the behavioural annotation of the CBD. This time series contains a long episode of locomotion and a few low-amplitude turns. Vertical lines denote the behavioural windows found by the DCS segmentation: a behavioural window is a green dashed line (local maximum) bounded by red lines (local minimums). At the top of the figure, the green check marks and the red crosses show whether a given DCS segment was counted as a match or not. As described in the text, consecutive locomotive steps were matched to the locomotion.

5.3 Results

The DCS method was applied to the ECTS for both larval *Drosophila* and *C. elegans*. The resulting DCS segments were 'subjectively valid,' in the sense that the video clips corresponding to the segments were largely turns and episodes of locomotion, as would be intuitively expected. Note that at this point, only the segmentation of the worm ECTS has been evaluated quantitatively.

To summarise the performance of the segmentation, the *Precision*, *Sensitivity* and *F-score* statistics are presented in Table 5.1. For the description of these statistics and how the matches were counted, see Section A in the *Supplementary materials*.

As discussed earlier, DCS has an input parameter denoted by κ that sets how much higher/lower a point must be from its local environment to be a local maximum/minimum. Similarly, the MK dictionary algorithm requires a pruning parameter known as the δ parameter. To assess the sensitivity of the segmentation methods with respect to their parameters, Table 5.1 shows how the statistical measures changed.

	TP	FP	FN	Sensitivity	Precision	F-Score
DCS $\kappa = 0.85\kappa_{def}$	179	48	8	0.81	0.95	0.86
DCS $\kappa = 0.9\kappa_{def}$	179	47	8	0.8	0.96	0.87
DCS $\kappa = 0.95\kappa_{def}$	179	45	8	0.8	0.96	0.87
DCS $\kappa = \kappa_{def}$	179	45	8	0.8	0.96	0.87
DCS $\kappa = 1.05\kappa_{def}$	179	45	8	0.8	0.96	0.87
DCS $\kappa = 1.1\kappa_{def}$	178	44	9	0.79	0.96	0.87
DCS $\kappa = 1.15\kappa_{def}$	177	43	10	0.79	0.96	0.86
MK $\delta = 0.85\delta_{def}$	115	20	72	0.85	0.62	0.7
MK $\delta = 0.9\delta_{def}$	119	27	68	0.82	0.64	0.7
MK $\delta = 0.95\delta_{def}$	121	34	66	0.78	0.65	0.71
MK $\delta = \delta_{def}$	125	42	62	0.75	0.67	0.71
MK $\delta = 1.05\delta_{def}$	130	48	57	0.73	0.69	0.71
MK $\delta = 1.1\delta_{def}$	133	53	54	0.71	0.71	0.71
MK $\delta = 1.15\delta_{def}$	135	59	52	0.7	0.72	0.70

Table 5.1: Counting the number of behavioural windows produced by the DCS or MK dictionary algorithm that overlapped with the segmentation of the CBD annotation. For details on the statistical measures, see 5.2.3 and [Powers, 2011]

.

5.4 Discussion

Segmentation is the problem of dividing a time series into meaningful 'segments' or 'behavioural windows.' The goal of segmentation is to produce behavioural windows that contain subsequences that are the most critical for understanding the structure of the time series. There is no general recipe for what constitute meaningful behavioural windows in a time series; hence, domain knowledge is required to evaluate any segmentation scheme.

In this chapter, the behavioural windows produced by the DCS and the MK dictionary algorithms are evaluated against the annotation of the *C. elegans* behavioural database. Importantly, the behaviour labels are discarded at this stage, as we are only interested in whether the segments selected by the algorithms are the same as the ones selected by the CBD annotation.

DCS is a custom algorithm that is based on change point segmentation. The concept is that behavioural windows can be separated at points where the dynamics of the time series change. Specifically for DCS, the change points are the local maxima/minima in the weighted sum of the ECTS components. A behavioural window is defined as a local maximum bounded by local minimums.

MK dictionary is a motif-finding algorithm, but in this chapter it is used as a segmentation method: the 'motif identities' of the subsequences are discarded and only the positions of the segments are kept.

Comparison of the algorithms found that the F-Score values of DCS were higher and more stable with respect to the changing parameters (see Table 5.1). The most frequent error of DCS was false positives, meaning that the DCS algorithm selected subsequences that were not annotated by the CBD. This is in contrast with the MK dictionary algorithm, where false negatives were dominant, i.e. the MK dictionary missed behavioural windows. This might not be surprising, as most of the motifs found by the MK dictionary were discarded (see 4.2.2). However, when the number of motifs discarded was decreased (as determined by the δ parameter), the number of false positives rose sharply and the F-score stagnated around 0.7.

While DCS was found to be superior to MK dictionary, a number of other segmentation algorithms could have been tested as well. It should be kept in mind, however, that the goal of the thesis is to develop an unsupervised pipeline to extract behavioural motifs. CBD annotation is based on human intuition; hence, if a segmentation algorithm were to produce the same behavioural windows as the CBD, it would likely also carry the same biases. Therefore, the goal is not to produce a perfect match with the CBD annotation, but rather to show that there is a general agreement.

Note that at this point, no quantitative evaluation has been done for the segmentation of *Drosophila* larva ECTS. The segmentation of larval ECTS is validated retrospectively after the segments are clustered. It is shown in Section 7.3 that the resulting behavioural annotation is a good match to expert annotation.

Applying DCS to ECTS results in a collection of behavioural windows, where each window consists of a three-dimensional time series (corresponding to the first three components of ECTS). Next, various clustering methods are applied to evaluate how many typical patterns can be found in the collection of behavioural windows.

Chapter 6

Clustering

6.1 Introduction

Segmentation-clustering (SC) is a two-step process in which the time series is first segmented into behavioural windows and then the windows are clustered. As discussed in the previous chapter, the segmentation step implicitly assumes that it makes sense to discretize behaviour. This assumption is carried forward by the clustering step, where the goal is to find groups of similar ECTS segments. As will be shown, not every algorithm tested was able to abstract meaningful clusters from the datasets.

The purpose of clustering is to group together data objects that are more similar to each other than to objects in other clusters. In the context of finding posture motifs, the 'data objects' are the ECTS segments found by the DCS method. Clustering is a classic problem in machine learning, and as a consequence, there are hundreds of methods available. To gain an idea of which method would work best for this particular dataset, three clustering methods were evaluated: k-means, DBSCAN and regression clustering. These methods were chosen because they represent different archetypes of clustering methods.

The conceptual simplicity and ease of implementation of K-means clustering makes it the most used clustering algorithm. It is based on iteratively assigning data objects to the closest cluster centroid and then repositioning the cluster centroids to minimise an error function. DBSCAN is based on the idea that clusters can be found as high-density areas in the data object space. Loosely speaking, a cluster is found if there are more than a threshold number of objects in the neighbourhood of a point. Regression

clustering reproduces the data as a sum of component functions. The most well-known example of the component function is Gaussian, but splines are used here instead, as they naturally parameterise ECTS subsequences.

These three clustering methods were applied to the set of larval *Drosophila* and *C. elegans* ECTS subsequences and the resulting behavioural annotations were evaluated against expert annotation of the same dataset.

6.2 Methods

6.2.1 Alignment of ECTS subsequences

To improve the consistency of spline fitting, the ECTS subsequences were aligned in the time domain. The need for alignment arises because there is uncertainty as to when a given action has exactly started and finished. Therefore, if no alignment is performed, the corresponding features of actions may appear at different times.

The subsequences were aligned by their peak amplitudes in the *bodyScore* time series, as shown in Figure 6.1. The frame of the highest *bodyScore* is less ambiguous regarding when a given action began/ended than the exact frame. Therefore, the frame of the highest *bodyScore* is a rational choice for defining a reference point in time to which the actions are aligned. All further data analysis was done on the time-aligned subsequences.

As discussed earlier, the dynamic time warping (DTW) distance measure is able to deal with misaligned sequences (see the discussion in 4.3). The use of DTW could eliminate the need for alignment in all the clustering methods, except for the spline regression framework. In that case, the distance among ECTS segments is not directly measured on the curves, but rather on the coefficients that describe the curves. DTW allows many-to-one point mapping that is not appropriate for measuring distance between the coefficients. As discussed in 6.2.6, Euclidean distance between the curve coefficients is used to quantify similarity for the spline regression framework.

For the other methods tested here (k-means and DBSCAN clustering), the use of DTW distance or Euclidean distance for the aligned sequences yielded practically identical results. This equivalence was due to the very strong correlation between the DTW distance and the Euclidean distance measured for the aligned sequences ($R^2 = 0.94$). Note that the correlation is much weaker between the DTW distance and Euclidean distance measured for the non-aligned sequences ($R^2 = 0.62$).

Given the correspondence of the two distance measures, and the fact that most packages use Euclidean distance by default for simplicity, Euclidean distance was used for the aligned sequences.

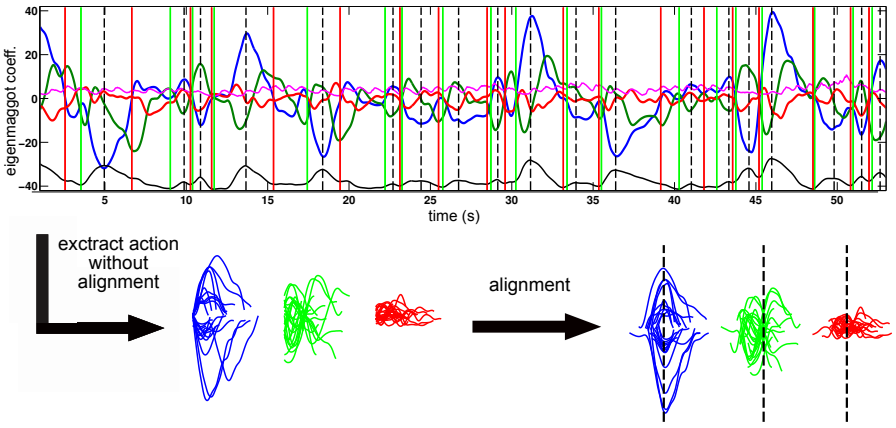


Figure 6.1: Alignment of the behavioural windows in the time domain. The figure shows a window of *Drosophila* behaviour, with the ECTS components and the body score at the bottom. For each action, the frame with the highest $bodyScore$ value, represented by a dashed black line, was used as a reference. Actions are shifted in time such that their point of highest $bodyScore$ coincides. The frame with the highest body score defines the most curved posture during actions. Therefore, it provides a rational measure for defining the midpoint of actions (in time).

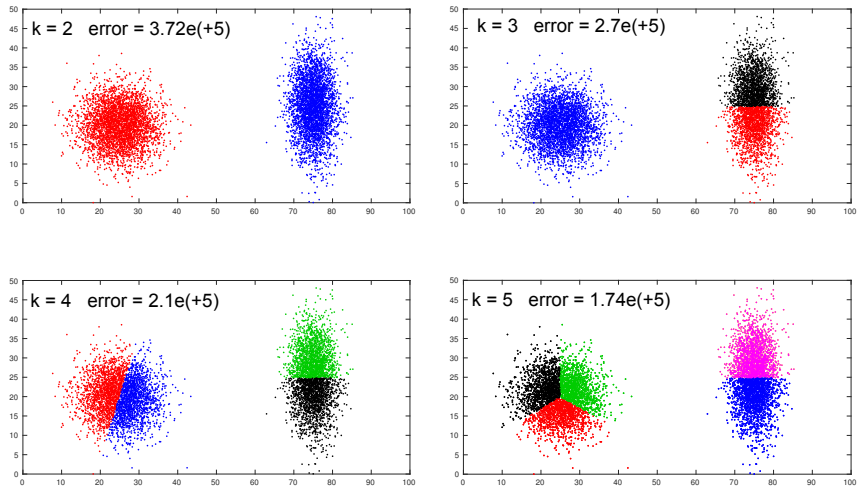


Figure 6.2: Illustration of over-fitting using two-dimensional data and k-means clustering. Each box shows the result of clustering with a different value of k; points are coloured according to cluster identity. Notice that the error, defined as the sum of the distances between cluster centroids and the data objects, decreases with increasing k. However, the global structure of the data is best explained by k=2, despite other solutions having lower error.

6.2.2 Model selection

The problem of model selection arises because the greater degrees of freedom a clustering method has, the 'better' the results are. Here, 'better' means that the error metric that the algorithm is trying to minimise has a smaller value. However, a smaller error does not necessarily mean the model possesses better explanatory power.

Consider the application of k-means clustering to a set of two-dimensional data objects that have a global structure, as shown in Figure 6.2. From the arrangement it can be seen that the set has two distinct clusters. The error function that k-means is trying to minimise is the sum of the distance between cluster centroids and the members of the cluster (see Section 6.2.4 for more details). By increasing k, the error decreases, although the additional clusters are not justified by the structure of the data. The error decreases because there are more parameters that the algorithm can fit to the data. In the extreme case in which k is set as equal to the number of data points, the error function can potentially be zero.

A number of techniques have been developed to avoid over-fitting and to find the k that best describes the data. In this thesis, two of them are used: the Dunn index and the Bayesian information criteria (BIC). Both are internal criteria, meaning that they can be calculated without reference to any external (known to be true) cluster assignment. An important difference to note is that BIC requires a probabilistic clustering, such as regression clustering, while the Dunn index is applicable to methods like k-means, which only outputs the cluster label for each object.

Bayesian information criteria

Bayesian Information Criteria (BIC) [Schwarz et al., 1978, Fraley and Raftery, 1998, Konishi and Kitagawa, 2008] are defined as:

$$BIC = 2ln(\mathscr{L}_{model}) - k * ln(n), \tag{6.1}$$

where $\mathscr{L}_{model}$ is the likelihood of the model, k is the number of free parameters and n is the number of observations. The first term reflects the goodness-of-fit of the model, and the second is a penalty term for the number of free parameters. [Schwarz et al., 1978] first introduced BIC, providing an argument based on Bayesian statistics to justify the combination of the goodness-of-fit and the penalty terms.

In practice, BIC is applied to models of various k and the ΔBIC indicates the strength of evidence for a given model. The higher the ΔBIC is, the stronger the evidence for the given model, where the ΔBIC is measured against models with both higher and lower k. While no hard ΔBIC thresholds have been defined, the general consensus regarding the strength of evidence is summarised in table 6.1.

ΔBIC	Evidence
$0 < \Delta\text{BIC} \leq 2$	Not substantial evidence
$2 < \Delta\text{BIC} \leq 6$	Weak evidence
$6 < \Delta\text{BIC} \leq 10$	Considerable evidence
$\Delta\text{BIC} > 10$	Strong evidence

Table 6.1: Summary of the strength of evidence for model selection by Bayes factors. Table is adopted from [Kass and Raftery, 1995].

Dunn index

The Dunn index is defined as the ratio between the average inter- and intra-cluster distances. A larger Dunn index indicates better clustering, as it means that the average distance within clusters is smaller than the distance between cluster centroids. If C_i is the i^{th} cluster centroid, x_j is the j^{th} data object, n_i is the number of data objects in the i^{th} cluster, N is the total number of data objects and k is the number of clusters, the average inter-cluster distance, ρ, and the average intra-cluster distance, δ, can be defined as follows:

$$\rho = \frac{1}{\sum_{k=1}^{k-1} k} \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} d(C_i, C_j), \tag{6.2}$$

$$\delta = \frac{1}{N} \sum_{i=1}^{k} \sum_{j=1}^{n_i} d(C_i, x_j), \tag{6.3}$$

which leads to the formal definition of the Dunn index as

$$DI = \frac{\rho}{\delta}. \tag{6.4}$$

Note that there are multiple ways to calculate both the inter- and intra-cluster distances. For example, δ is sometimes calculated as the average pairwise distance within a cluster. Hence, there are variations in how different studies have defined the Dunn index.

6.2.3 Non-classical multidimensional scaling

Non-classical multidimensional scaling (MDS) is a data visualisation technique. It takes a distance matrix as an input and it projects the elements onto a two dimensional plane such that the loss function *stress* is minimised. *Stress* is defined as

$$Stress = \sum_{i \neq j}^{n} (D_{(i,j)} - \mid x_i - x_j \mid)^2, \tag{6.5}$$

where $D_{(i,j)}$ is the Euclidean distance matrix of the mapped points and x_i is the coordinate of the i^{th} point of the original data. Standard MATLAB implementation of the algorithm was used for the figures presented in this thesis.

For all the MDS plots in this thesis, the algorithm was initiated 100 times with random boundary conditions and the solution with lowest *stress* was kept.

R^2 is the correlation between the DTW distances measured for ECTS subsequences and the Euclidean distances between the corresponding points in the MDS maps. For the *Drosophila* maps $R^2 = 0.84$, while for the *C. elegans* maps $R^2 = 0.75$. While these figures do not perfectly reproduce the distance matrix, they are reasonably close, providing an intuitive understanding of the data structure. Note that in all of the MDS figures, there is a left/right symmetry corresponding to the left/right symmetry in the animal's behaviour (for example, turns to the left or right).

6.2.4 K-means clustering

K-means clustering (also known as Lloyd's algorithm) is a simple and intuitive clustering method. Initially, k data objects are randomly selected as cluster centroids. Afterwards, the algorithm proceeds by alternating between two steps:

1. **Data object assignment**: Each data object x_i is assigned to the closest cluster, that is, the cluster for which

$$\arg\min_j D(x_i, \mu_j), \tag{6.6}$$

   where $j = 1, 2, ..., k$ is the index of the clusters, μ_j is the centroid of the j^{th} cluster and $D(a,b)$ is a general distance function.

2. **Cluster centroid update**: The position of each cluster centroid is updated such that the within-cluster sum of squares (often abbreviated as WCSS) is minimised:

$$error = \sum_i D(x_i, \mu_k), \tag{6.7}$$

   also known as the error function.

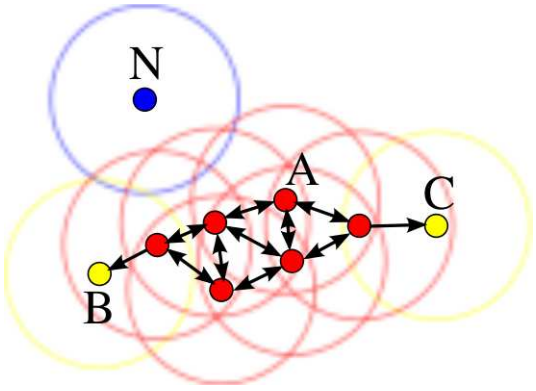
The algorithm is guaranteed to converge to a local minimum that may or may not be the global minimum. To avoid getting stuck at a local minimum, the algorithm was run 100 times with random boundary conditions for all experiments presented. The solution with the lowest error, as defined by Eq. 6.7, was kept.

The main drawback of the k-means algorithm is that it is best suited to discovering globular clusters of roughly equal size. Furthermore, a poor estimation of k can lead to misleading results. To mitigate this issue, the Dunn index was used to find the optimal number of clusters. Note that BIC is not applicable to the output of the k-means algorithm, as it gives hard cluster labels instead of probabilistic cluster assignments. For details on the k-means algorithm, see [Hartigan and Wong, 1979].

6.2.5 DBSCAN

DBSCAN stands for density based spatial clustering of applications with noise. This method conceptualizes clusters as high-density volumes in the space of the data objects. DBSCAN requires two input parameters: the ε and *minPoints*. Based on these parameters, the algorithm divides all points into three classes:

- A **core point** is defined as a point that has more than *minPoints* points in its ε neighbourhood
- Point q is a **reachable point** from p if there is a chain of points $p_1, ..., p_n$ that satisfies the following criteria:
  1. $p_1 = p$
  2. $p_n = q$
  3. every p_{i+1} is in the ε neighbourhood p_i
- **Outliers** are points that are not reachable from any other point.



e

Figure 6.3: In this diagram, minPts = 3. Point A and the other red points are core points, because at least three points surround them in an ε radius. Because they are all reachable from each other, they form a single cluster. Points B and C are not core points, but are reachable from A (via other core points) and thus belong to the cluster as well. Point N is a noise point that is neither a core point nor density reachable

DBSCAN defines a cluster as a core point and all of the points reachable from it. Note that k here is derived from the data, rather than being a free parameter. This property is one of its advantage compared to the k-means algorithm. Furthermore, DBSCAN can detect clusters of various sizes and shapes, and also has a natural way of dealing with noisy outliers in the dataset. Its disadvantage is that both parameters, *minPoints* and ε, heavily influence the clustering results and thus it can be difficult to fix them. For a more detailed description of DBSCAN, see [Ester et al., 1996].

6.2.6 Mixture of regressions models

In this section a mixture of regression models clustering framework is presented. It was originally developed by Scott Gaffney to cluster trajectory data and this discussion closely follows his papers [Gaffney and Smyth, 2004, Gaffney, 2004]. ECTS segments are trajectories across the eigenshape space, therefore trajectory clustering is highly relevant. The original author's MATLAB implementation of the algorithm has been used in this thesis [1].

Finite mixture models

Finite mixture models represent data as a linear combination of component functions. It is a form of semi-parametric density estimation, where the final probability density function (PDF) is a convex combination of the component PDFs. Let $\mathbf{y}_i$ stand for the i^{th} observation of the vector of interest. Using this notation the PDF can be written as:

$$P(\mathbf{y}_i \mid \theta_k) = \alpha_k f_k(\mathbf{y}_i \mid \theta_k), \tag{6.8}$$

where θ_k is the set of parameters associated with the k^{th} component and α_k is the weight of the k^{th} component. Note that θ_k is potentially multidimensional, for example in the case of Gaussian component functions $\theta_k = \{\mu_k, \sigma_k\}$.

Mixture models can be used to identify clusters in the data. An underlying assumption is that each observation was generated by one of the component functions and each of the components are treated as a cluster. Note that initially we do not know which observation was generated by which component, often denoted by z_i, and this latent variable will be called the 'cluster identity of observations'.

[1] The code can be found at http://www.ics.uci.edu/ sgaffney/software/CCT/ (last accessed on 2017 Jan 17)

To find the components/clusters, observations are treated as a random sample of the underlying PDF and this data is used to estimate the missing parameters (θ_k) associated with the components. To estimate the parameters the maximum likelihood principle is involved. Assume that the observations are independently and identically distributed. Furthermore let Y denote the whole set of observations, that is $Y = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$, and let $\Theta = \{\theta_1, \ldots, \theta_{k_{max}}\}$. Then the log-likelihood can be written as

$$\mathscr{L}(\Theta \mid Y) = logP(Y \mid \Theta) \tag{6.9}$$

$$= \sum_i logP(\mathbf{y}_i \mid \Theta) \tag{6.10}$$

$$= \sum_{i,k} logP(\mathbf{y}_i \mid \theta_k). \tag{6.11}$$

By maximising $\mathscr{L}$, the Θ is found that best fit the observations. The problem is that not only the component parameters are unknown, but so is z_i, the cluster identity of observations. To estimate Θ we would need to know z_i and vice versa. As a result, in general, there is no easily obtainable analytic solution to eq.6.11.

To obtain an approximate numerical solution, the expectation-maximization (EM) algorithm is widely used [Krishnan and McLachlan, 1997]. Briefly, EM is an iterative root-finding procedure that is used when a statistical model involves both latent variables and a set of unknown parameters. In the current context the latent variables are the cluster identity of observations. The intuition behind the EM algorithm is that one can first randomly select the latent variables and use this to estimate the parameters, then use the estimated parameters to find a better allocation of the latent variables. These two steps are alternating until the log-likelihood stabilizes. It can be shown under some fairly general conditions the EM is guaranteed to converge [McLachlan and Krishnan, 1997].

Once the EM algorithm has converged, Bayes rule can be applied to eq. 6.8 to estimate the membership probability of observations, i.e.: the probability that a given observation was generated by a given which component. To complete the clustering often each observation is assigned to the cluster with the highest membership probability (i.e. ‘hard cluster label’). However it is not necessary to collapse the probabilities, one advantage of the model based clustering procedure is that the membership probabilities provide a way to quantify our uncertainty by calculating the classification entropy. I will take advantage of this later in chapter 7.

Similar to the k-means algorithm discussed earlier, model based clustering also requires a user-defined number of clusters. To make this choice data driven, the Bayesian information criteria was used (see Section6.2.2) through this thesis.

Mixture of regression models

In this subsection the finite mixtures models are extended to a *mixture of regression models* framework. Consider now that not only $\mathbf{y}_i$, but also x_i is observed and there is a cluster specific regression relationship between these two variables:

$$\mathbf{y}_i = g_k(x_i) + \varepsilon_k, \tag{6.12}$$

where $g_k(x_i)$ is a cluster specific deterministic function and ε_k is a non-zero Gaussian noise term with a standard deviation of σ_k. In our case x_i corresponds to time, but it could be any independent variable. Note that the noise term, ε_k, could be generalized to incorporate a dependence on x_i, but for the purposes of this discussion it is assumed that noise is independent of x_i.

To incorporate the new observed variable and the regression relationship, the PDF and the model likelihood needs to be updated as:

$$P(\mathbf{y}_i \mid x_i, \Theta) = \sum_k \alpha_k f_k(\mathbf{y}_i \mid x_i, \theta_k), \tag{6.13}$$

$$\mathscr{L}(\Theta \mid Y) = \sum_{i,k} logP(\mathbf{y}_i \mid x_i, \theta_k). \tag{6.14}$$

The important point is that now the conditional mixture density and the $\mathscr{L}$ is defined on a set of regressions or trajectories (where the trajectory is defined by eq. 6.12) rather than on a fixed length vector as in the previous section. An important advantage is that the regression relationship naturally handles variable length curves and random measurement intervals.

The regression and the error model together result in the cluster specific conditional PDFs:

$$P(\mathbf{y}_i \mid x_i, \theta_k) = f_k(\mathbf{y}_i \mid x_i, \theta_k), \tag{6.15}$$

$$= \mathcal{N}(g_k(x_i), \sigma_k), \tag{6.16}$$

where $\mathcal{N}(g_k(x_i), \sigma_k)$ is a Gaussian with mean $g_k(x_i)$ and standard deviation σ_k. This is an important observation, because it means that the EM algorithm needs to be minimally modified compared to the case of standard Gaussian mixture models. For a technical derivation of the EM algorithm for mixtures of regressions models see [Gaffney, 2004].

In this section mixture models have been extended to a mixture of regressions models. In the final clustering of ECTS subsequences, a mixture of spline regression models have been used, because the ECTS subsequences are naturally represented by splines. To incorporate spline regressions requires a minimal extension of the current framework and is presented next.

Extension to spline regressions

Splines are piecwise polynomial curves that are connected at 'knot points' and satisfy certain smoothness criteria [De Boor et al., 1978]. To present spline theory is beyond the scope of this thesis, for practical purposes it should suffice that splines represent curves as an expansion over a set of basis functions. In practice, splines are often implemented as B-splines, because the resulting basis matrices are computationally convenient block-diagonal matrices. Let $\mathbf{B}_i$ define the set of basis functions:

$$\mathbf{B}_i = \{B_{1p}(x_i), B_{2p}(x_i), \ldots, B_{mp}(x_i)\}, \tag{6.17}$$

where the particular basis matrices ($B_{np}(x_i)$) are dependent on the knot points selected and the order of the polynomials used. Note that $\mathbf{B}_i$ is already evaluated at x_i. Hence, using the spline bases, a curve can be described as

$$\mathbf{y}_i = \mathbf{B}_i \mathbf{c}, \tag{6.18}$$

where $\mathbf{c}$ is a vector of spline coefficients. Using this notation the regression relationship between $\mathbf{y}_i$ and x_i can be written as

$$\mathbf{y}_i = \mathbf{B}_i \beta_k + \varepsilon_k, \tag{6.19}$$

where β_k is the cluster defining spline coefficients and ε_k is a cluster specific Gaussian noise term. Similarly as in the previous section, the form of the regression and the Gaussian noise term (eq. 6.19) result in a conditional PDF for $\mathbf{y}_i$ as:

$$P(\mathbf{y}_i \mid x_i, \theta_k) = f_k(\mathbf{y}_i \mid x_i, \theta_k) \tag{6.20}$$

$$= \mathcal{N}(\mathbf{y}_i \mid \mathbf{B}_i\beta_k, \sigma_k). \tag{6.21}$$

Furthermore, the model likelihood can be obtained by substituting eq.6.21 to eq.6.11 and yields the following form:

$$\mathcal{L}(\Theta \mid Y) = \sum_i logP(\mathbf{y}_i \mid x_i, \Theta) \tag{6.22}$$

$$= \sum_{i,k} log(\alpha_k P(\mathbf{y}_i \mid x_i, \theta_k) \tag{6.23}$$

$$= \sum_{i,k} log(\alpha_k \mathcal{N}(\mathbf{y}_i \mid \mathbf{B}_i\beta_k, \sigma_k)). \tag{6.24}$$

This completes the overview of spline regression mixture models. More details and the corresponding EM algorithm can be found in the original publication associated with the methodology [Gaffney and Smyth, 2004, Gaffney, 2004].

When clustering ECTS subsequences, splines had three knot points, meaning that each ECTS subsequence in each dimension was represented by two polynomial curves. Each polynomial had an order of three, and therefore each ECTS subsequence was represented by 18 spline coefficients (3*2*3 corresponding to (number of ECTS dimensions)*(number of polynomials in each dimension)*(order of each polynomial)). This choice of spline parameters is justified in that increasing either the number of knot points or the dimensionality of the each polynomial did not yield any noticeable improvement ($\leq 6\%$) in the residual sum of squares metric (measured between the original curve and the curve reconstructed by the spline parameters).

Throughout this thesis when the mixture of regressions methodology is used, the EM algorithm was initiated 500 times with random boundary conditions and the solution with the highest likelihood was kept.

6.3 Results

6.3.1 Number of clusters

As discussed earlier, both *k*-means and regression clustering require the user to define *k*, the number of clusters in the dataset. To avoid bias, *k* was set using the Dunn index in the case of *k*-means and BIC in the case of regression clustering (both selection criteria are discussed in 6.2.2).

For *k*-means clustering, the Dunn index indicated that a single cluster best describes both the larval *Drosophila* and *C. elegans* data. It is meaningless to cluster a dataset with k=1; hence, during the comparison to ground truth described in 6.3.2, *k* was set to 2 and 3 for *Drosophila* and *C. elegans*, respectively. These numbers were chosen because they have been indicated by BIC as suitable during regression clustering.

In the case of regression clustering, BIC indicated the presence of 2 and 3 'weak clusters' for larval *Drosophila* and *C. elegans*, respectively. 'Weak clusters' mean that there are many segments that cannot be unambiguously assigned to either cluster. Hence, it will be argued in the next chapter that behaviour should be considered as a spectrum rather than as a set of well-defined states.

For both organisms, the number of behavioural motifs map well onto behavioural classification schemes in the literature. The description of the behavioural motifs for both organisms are given in 7.3.

In the case of DBSCAN, the algorithm sets *k* automatically; however, the user-defined parameters of ε and $minPoints$ influence the number of clusters found. Both parameters were varied on a large scale. However, in all cases, only 1 or 2 clusters were found. The second cluster was always formed by a few scattered points near the edge of the MDS maps. Hence, effectively, for all parameter settings, k=1. An example of a $k = 2$ solution for both organisms is shown in Figure 6.4. The lack of well-defined clusters is due to the density of points falling uniformly going outward from the centre (see Figure 6.5).

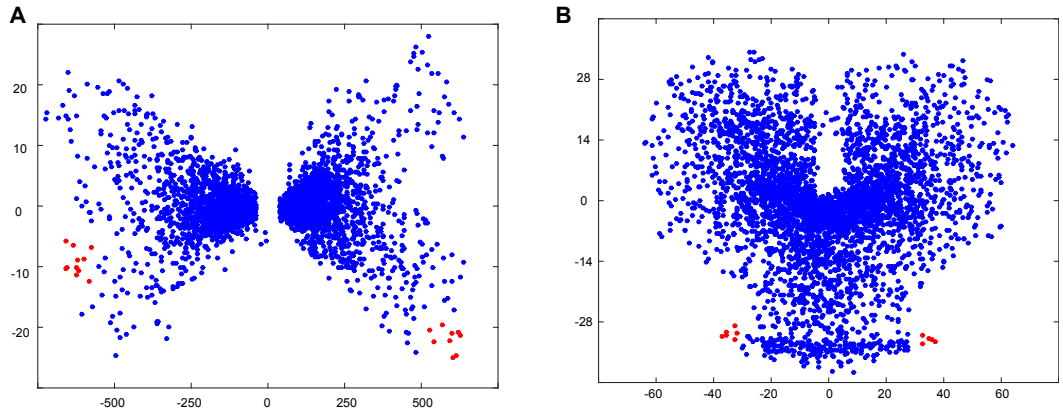


Figure 6.4: Sample of results using the DBSCAN algorithm. For most parameter settings, the method identified a single cluster for both organisms, but for low values of $minPoints$, some $k = 2$ solutions were found. The second cluster in all cases consisted of a few points. Hence, effectively, every solution had $k = 1$. The DBSCAN algorithm discards outsider points, and thus these figures appear to be trimmed compared to the other MDS maps.

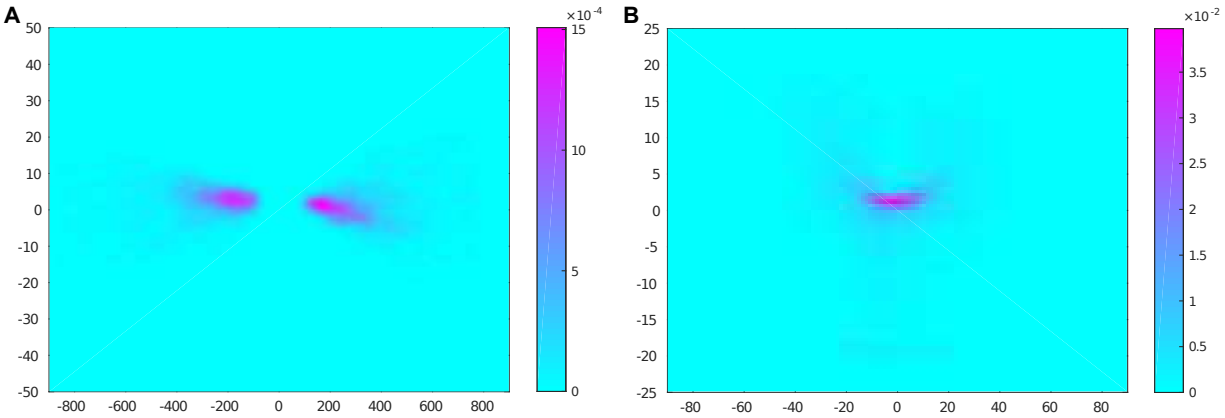


Figure 6.5: Density heat maps of the multidimensional scaling figures for the maggot in Panel **A** and the worm in Panel **B**. In both cases, the density drops uniformly from the centre of the figure, indicating that the ECTS subsequences continuously change (best viewed on computer screen).

6.3.2 Comparison to ground truth

To evaluate how each clustering method performed, the resulting behavioural annotation was compared to expert annotation. The results of the DBSCAN algorithm were not evaluated, as it produced $k = 1$. For the k-means algorithm, the Dunn index was discarded and k was set to 2 and 3 for *Drosophila* and *C. elegans*, respectively.

For the regression and k-means algorithms, the MDS maps with various k values are presented in Figure 6.6. These maps can help in visualising the global state space and understanding where the algorithms place the decision boundaries.

To summarise the performance, the *Precision*, *Sensitivity* and *F-score* statistics are presented below in tables 6.2 and 6.2 for the maggot and worm, respectively. For a description of these statistics, see Section A. The corresponding *true positive*, *false positive* and *false negative* counts from which the statistics are derived can be found in the *Supplementary tables* (Section C).

	Locomotion			Turn			Dwelling			All behaviours		
	Pre.	Sen.	F	Pre.	Sen.	F	Pre.	Sen.	F	Pre.	Sen.	F
K-means	0.58	0.68	0.63	0.58	0.86	0.69	0.54	0.58	0.56	0.57	0.73	0.64
Regression	0.83	0.93	0.9	0.67	1	0.8	0.73	0.83	0.77	0.74	0.95	0.82

Table 6.3: Statistical summary of *C. elegans* behavioural annotation using k-means/spline regression clustering. For details on the comparison procedure, see A in the *Supplementary materials*. *Precision*, *sensitivity* and *F-score* values were derived from supplementary table C.5.

	Run Cast			Stop Cast			Turn			All behaviours		
	Pre.	Sen.	F	Pre.	Sen.	F	Pre.	Sen.	F	Pre.	Sen.	F
K-means	0.51	0.73	0.6	0.72	0.75	0.73	0.55	0.39	0.46	0.55	0.64	0.59
Regression	0.64	0.91	0.75	0.74	0.75	0.75	0.7	0.51	0.59	0.67	0.77	0.72

Table 6.2: Statistical summary of the larval *Drosophila* behavioural annotation using k-means/spline regression clustering. For details on the comparison procedure, see A in the Supplementary materials. *Precision*, *sensitivity* and *F-score* values were derived from supplementary table C.2.

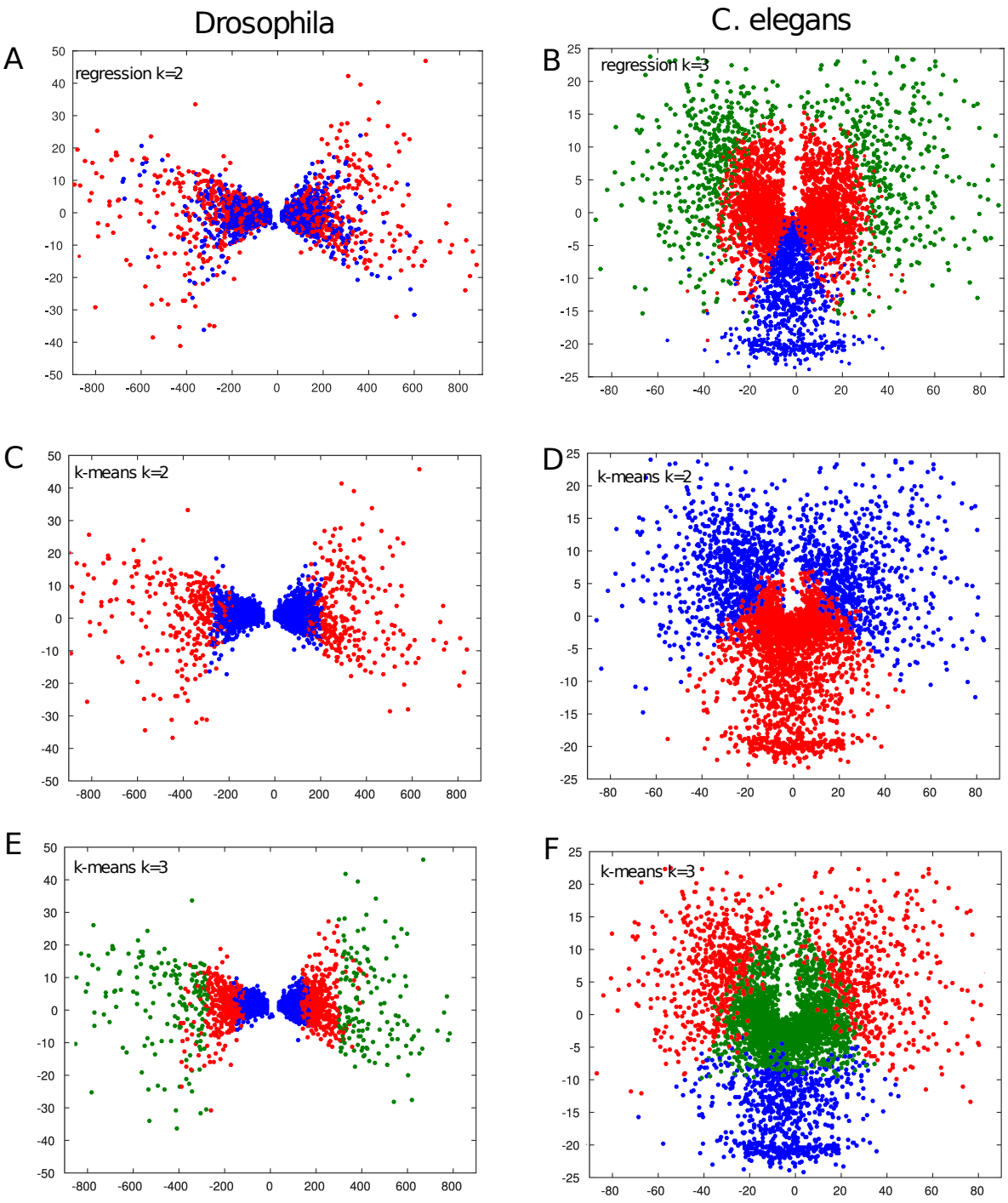


Figure 6.6: Multidimensional scaling maps of larval *Drosophila* and *C. elegans* behaviours (left and right columns, respectively). For technical details see 6.2. Panels **A** and **B** show the results of regression clustering, while Panels **C**-**F** show the results of the k-means clustering for various k values. Each point corresponds to a behavioural window, as produced by the DCS segmentation. The cluster label for each point is colour coded, although it should be noted that the colours are arbitrary and not consistent across figures.

6.4 Discussion

The purpose of clustering is to group together data objects that are more similar to one another than objects in other clusters. In the current case, the 'data objects' are the ECTS subsequences produced by DCS segmentation. Segments in the same cluster correspond to similar sequences of postures, and therefore each cluster can be interpreted as a behavioural motif or behavioural state.

Three clustering methods have been evaluated here: k-means, DBSCAN and regression clustering. These three methods were chosen because they represent three different conceptual approaches. The cluster membership of the segments have been interpreted and the resulting annotation has been evaluated against expert annotation for the behaviour of both larval *Drosophila* and *C. elegans* (except for the DBSCAN approach, through which no meaningful clusters were found).

Regression clustering had the best performance by a large margin for both organisms. I believe this was because the EM algorithm's probabilistic approach can better handle a uniformly populated data space. By using the membership probabilities, instead of hard cluster labels, regression clustering takes into account the uncertainty of the cluster identity of the observations. An inherent assumption of clustering is that there is no continuous transition between element clusters, but rather that there is some distinguishing feature separating the two clusters. This assumption is not true for the set of *Drosophila* and *C. elegans* ECTS subsequences. In the next chapter, I further argue for the continuity of the behavioural states, but Figure 6.5 already hints that there is gradual transformation among the clusters, leading to the lack of well-defined states.

The continuous density of data objects is the reason why the DBSCAN algorithm did not yield meaningful results. The method relies on identifying density fluctuations in the data space. However, in this case, the density of objects changed smoothly, and hence, the algorithm failed to identify separate clusters.

The key difference between the EM (the learning method for regression clustering) and the k-means is that EM estimates the probability of each object belonging to each cluster. This yields a membership probability distribution for each data object:

$$M_i = [p_i^1, p_i^2, p_i^3], \tag{6.25}$$

where M_i is the membership probability distribution associated with the i^{th} segment, and p_i^n is the probability that object i belongs to cluster n (M_i in Eq. 6.25 is set up using $k = 3$, but it could be generalised to any k). This probabilistic cluster assignment contrasts the k-means algorithm, in which each object is assigned to a single cluster. That is, in Eq. 6.25, for one cluster $p_i^n = 1$, while for the other two, $p_i^n = 0$. The probabilistic M_i of the EM carries information about how strongly an object belongs to a cluster. Hence, it conveys more information about the global structure of the data than the simple cluster assignment of the k-means algorithm. I believe this is the key reason why regression clustering performed better than the other clustering methods.

It should be noted here that the MDS visualisation maps reproduce imperfectly the relationships in the distance matrix. However, the clustering methods and their comparison to the expert annotation are not dependent on the MDS maps.

So far, the SC motif discovery pipeline has been established, the use of the DCS method has been justified, and evidence has been provided that regression clustering is appropriate for the current problem. In the next chapter, the biological interpretation of the resulting clusters is given and the issue of the separability of behavioural states is closely examined.

Chapter 7

Continuity of behavioural states

Note

This chapter is the main publication associated with my work [Szigeti et al., 2015]. Therefore it contains information that already appeared earlier in the thesis, especially with respect to the *Methods* used. However it adds novelty as Section 7.3 carries further in the biological interpretation of the results and arguments are provided for why the behavioural states should not be considered as discrete entities, but rather as a continuous spectrum of behaviours.

My coauthors on this paper were Ajinkya Deogade and Barbara Webb. The paper includes the following author's contribution statement:

BS designed the study, implemented the code and wrote the manuscript. AD collected and hand annotated the larval data and helped to interpret the results. BW supervised the work and helped to write the manuscript.

7.1 Introduction

Automated analysis of behaviour is of increasing importance to biology and neuroscience. Behavioural control is the ultimate function of neural processing [Gomez-Marin et al., 2014]. The recent expansion of tools for manipulating neural activity, such as optogenetics, has made it crucial to be able to screen rapidly and automatically for the behavioural consequences of these manipulations. Standardisation of quantitative behavioural assays and reproducibility of analyses are thus key to progress in

understanding neural circuits.

Traditional manual annotation of behavioural data is not feasible for large datasets. As a consequence, automated high-throughput behavioural annotators have been developed. An example is the Janelia Automatic Animal Behaviour Annotator (JAABA) [Kabra et al., 2013]. JAABA first requires hand annotation of a subset of the data and then the software uses machine learning algorithms to find the same patterns in the unannotated data. Other researchers have developed classifiers that extract specific parameters from behavioural data and then register a state if a certain parameter (or parameter set) exceeds a user defined threshold [Ohyama et al., 2013, Gomez-Marin et al., 2011, Salvador et al., 2014, Yemini et al., 2013]. Note that for these classifiers both the set of possible behaviours and the description of those behaviours are encoded by the user. In contrast, our goal is to discover patterns in behaviour without reference to any user defined thresholds or examples.

Posture is the main observable component of behaviour, and the behavioural annotators mentioned above mainly use postural information as input to classify behavioural states. In this context, Stephens et al. introduced eigenworms [Stephens et al., 2008], using principal component analysis to produce a low dimensional representation of *C. elegans* midline shapes. For the unrestricted free behaviour of *C. elegans*, four eigenworms account for 92% of the animal's posture variance. This means that four numbers can describe any actual worm posture with high precision. Mathematically, postures are described by a superposition of eigenworms, i.e.:

$$posture(t) = \sum_{i=1}^{n} \alpha_i(t) eigenworm_i, \tag{7.1}$$

where $\alpha_i(t)$ is the coefficient associated with the i^{th} eigenworm at time t. Figure 3.1 shows the eigenworms and an example of posture reconstruction. Eigenshapes provide a compact representation of posture and hence clearly have potential use in behavioural annotation. Specifically, behaviour (change in posture over time) is represented by the time evolution of eigenshape coefficients, i.e., the time series of $\alpha_i(t)$s. This time series is referred to as the eigenshape coefficient time series (ECTS) and forms the basis of our method.

The technical aim of this paper is the unsupervised discovery of frequently repeated

ECTS subsequences. In the data mining literature frequently repeated subsequences are also known as motifs [Fu, 2011]. ECTS motifs correspond to frequently repeated sequences of posture, which can be viewed as behavioural states or actions [Green et al., 1983, Berman et al., 2014]. Previous attempts to extract ECTS motifs using a simple 'sliding window' motif discovery approach [Brown et al., 2013] suffer from two major problems. Firstly, the window for any pass is of fixed length, hence this method considers only exactly equal duration sequences as potential matches. Secondly, the sliding window method defines a motif as a *pair* of closest neighbour sequences. However motifs are understood intuitively not as a single pair of subsequences, but as a *frequently repeated* subsequence. Our motif-finding methodology was designed to overcome these two problems.

First we derive the equivalent of eigenworms for larval *Drosophila*, termed eigenmaggots. The ECTS of both larval *Drosophila* and *C. elegans* are then analysed using our novel motif-finding method. The ECTS motifs are used as the basis of a probabilistic behavioural annotator, the eigenshape annotator (ESA). We show that the resulting annotation corresponds well to hand annotation, although a number of behaviours can not be unambiguously classified. The ESA analysis is also applied to the behaviour of a state-based simulated maggot to show that the ambiguity is not inherent in the method, but reflects a greater continuity between behavioural states in these organisms than is generally assumed. In summary, our new method both confirms the results of previous behavioural annotation and reveals some of its limitations.

7.2 Methods

Our aim is to go from video of a behaving animal to annotation of its behavioural states, where those states are determined using bottom-up discovery of motifs in the sequence of postures. We start by recording freely foraging *Drosophila* larva, extract their midline as a set of angles, and apply principal component analysis to obtain a low dimensional description of postures, the eigenshape coefficient time series (ECTS). Equivalent information for the worm is available from the *C. elegans* behavioural database (CBD). Discovering motifs in the multidimensional ECTS is a non-trivial problem and there are no existing adequate tools. We developed a two step process to first extract subsequences and then fit a statistical model to cluster the subsequences. Briefly (details are given below), we use changes in the dynamics of the ECTS to divide the

sequence into variable length subsequences, with the intent that each subsequence contains a single ‘action’. The subsequences are aligned and then clustered using a spline regression model [Gaffney and Smyth, 2004, Gaffney, 2004], a method for analysing curves analogous to Gaussian mixture models. The resulting clusters constitute motifs by which the animal’s behaviour can be annotated. The results are compared to alternative annotation systems and to hand annotation provided by a human expert, which is treated as ground truth.

7.2.1 Data collection

Canton-S flies are maintained on conventional cornmeal-agar molasses medium at 22°C and kept in a 12 hour dark-light cycle. For the behavioural experiments larvae are placed on 3 % agarose and are allowed to freely forage. Across 33 individuals 14 hours of video was recorded at 30 fps. The videos are segmented (see below) into a total of 11613 actions. The tracking and data acquisition hardware used for this publication are described in detail at [Schulze et al., 2015]. Briefly, the larva moving over a fixed stage was imaged using a camera (Basler A622f) on top. The camera was mounted on a moving stage to follow the animal. The software for image capture and stage control was written in C using the OpenCV libraries.

To analyse worm behaviour we used data from the *C. elegans* behavioural database [Yemini et al., 2013]. The database consists of videos of worms (recorded at 30 fps) browsing in bacteria. For every video there is a corresponding feature file, which contains many precalculated statistics of worm morphology. The feature files also contain the eigenworm coefficient time series. The worm analysis in this paper uses this precalculated ECTS. 22066 actions are analysed from 100 experiments with N2 worms, corresponding to 25 hours of video.

7.2.2 Constructing eigenmaggots

In each video frame, the larva was separated from the background by a thresholding algorithm. The resulting binary images are skeletonised using the built-in MATLAB function. Midlines are rotated such that the endpoints, corresponding to the head and tail of the animal, lie along the x-axis. This operation removes the overall rotation of the animal's body relative to the plate. The midlines are normalised such that they consist of 71 points placed equidistant from each other. The length of the larva can change, but is neglected in this analysis, i.e., we treat every midline as if it is the same length. The eigenshapes on Figures 3.1 and 7.2 have been reconstructed to reflect the average physical size of the midlines. The angles among consecutive points defining the midline are restricted to the interval $-\pi < \theta_i \leq \pi$. As a result of these operations each frame is associated with a 70 dimensional vector, where the i^{th} component is θ_i (Figure 3.1C). These vectors are concatenated to form an $n * 70$ data matrix where n is the number of frames. Principal component analysis is applied to this data matrix to construct the eigenshapes and the associated eigenshape coefficient time series.

7.2.3 Eigenshape coefficient time series

For both the larval and worm analysis the coefficients of the three most significant eigenshapes are included in the ECTS, that is $ECTS(t) = [\alpha_1(t), \alpha_2(t), \alpha_3(t)]$, see Eq.7.1. After principal component analysis the inspection of the eigenvalues reveals that for both organisms three coefficients account for approximately 90% of the posture variance [Jolliffe, 2002], thus provide an accurate description of posture. At the same time a three dimensional ECTS is small enough to avoid 'the curse of dimensionality', that could lead to difficulties during the clustering step [Verleysen and François, 2005].

7.2.4 Dropped frames

Both the larval and the maggot ECTS contains dropped frames. If a gap was short ($<$0.5s), then ECTS was linearly interpolated. After the interpolation 1.1% of the *Drosophila* and 4.2% of the *C. elegans* frames are still missing. For both organisms on a significant portion of the dropped frames the animal was curled up in a 'doughnut shape' from which it is difficult to extract a biologically meaningful skeleton. For *C. elegans* more frames are dropped, because the worms are browsing in food. The layer of bacteria can obscure the worm in the image making separation of the body of the worm from the background more challenging. Note that the inability to analyse curled up postures introduces a bias to the pipeline, as no posture with self intersection is included.

7.2.5 Segmentation

The intuition behind the segmentation algorithm is that boundaries between windows should be located where the dynamics of ECTS changes. ECTS was smoothed using a weighted running average filter with a window size of four frames and weights inversely proportional to the distance from the window's centre. Segmentation operates on a 'body score' time series that is created by calculating a weighted sum of the separate dimensions of ECTS, where the weights are set by the eigenvalues associated with the eigenshapes. The segmentation algorithm scans the body score to find local minima and maxima. An action is defined as a local maxima in body score bounded by minimas. The minimas define the start and end of the segmented subsequence. Figure 7.1 shows the result of segmentation for *Drosophila* and *C. elegans* with the corre-

sponding body score time series.

The maxima/minima finding algorithm is controlled by a master parameter. The results are not strongly dependent on the precise parameter setting: adjusting it by $\pm 25\%$ leaves 92% of the annotation unchanged.

The behavioural videos of *C. elegans* are recorded while the worms are browsing in food. In this environment worms often show low activity. Our segmentation was designed to identify periods where the body score rapidly changes, hence the identification of low activity periods required an extra step. Low activity periods are identified by intervals where the time derivative of body score remained under half of its average value for more than 0.5s. These periods are added to the collection of actions prior to proceeding to the clustering step. If the two parameters (under 50% of average body score for more then 0.5s) are adjusted $\pm 25\%$ then 97% of the action's classifications are not altered. Thus fine tuning of the parameters is not necessary.

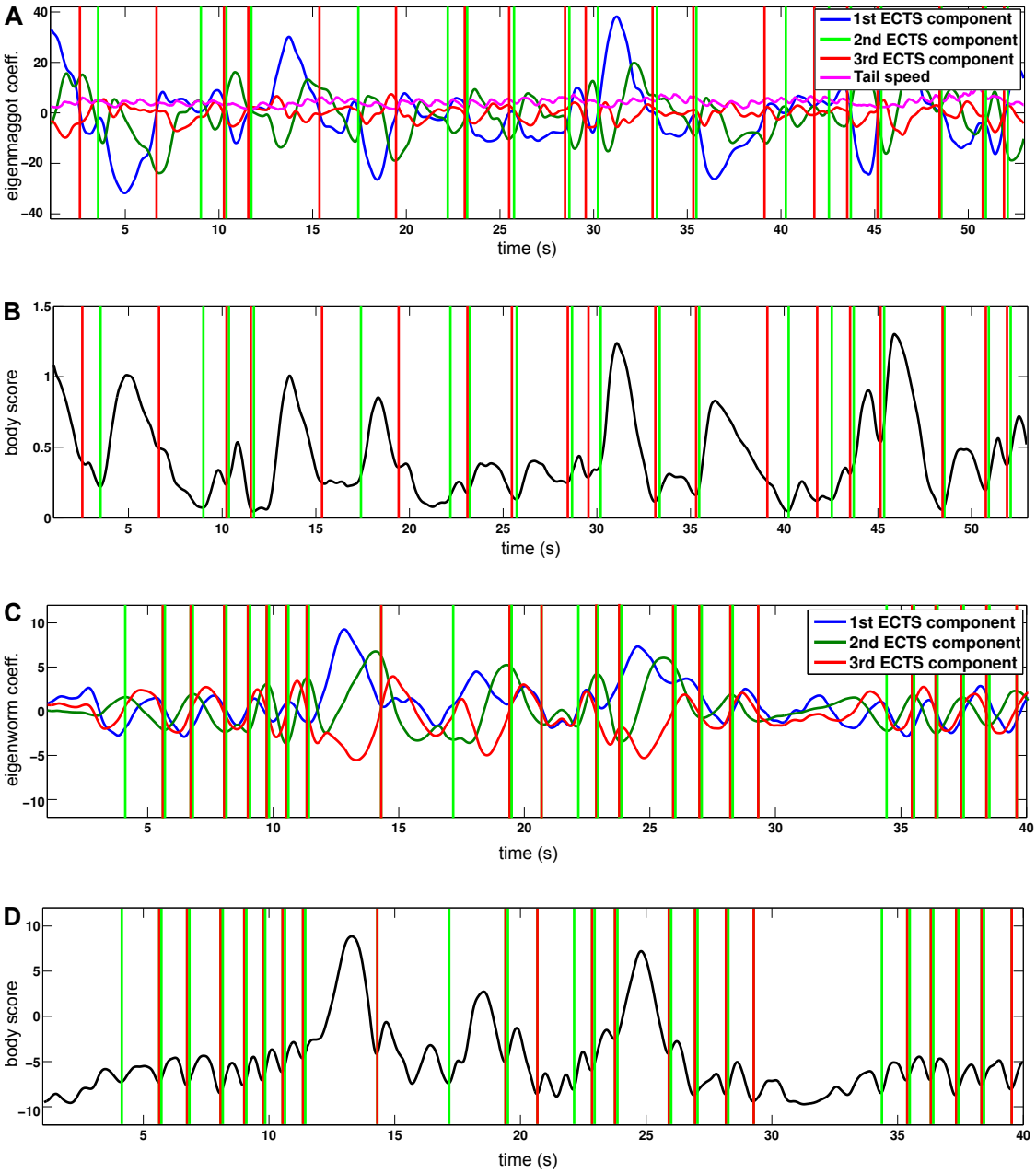


Figure 7.1: The segmentation algorithm. **A** shows a screenshot of the larval ECTS and the tail speed time series. Panel **B** shows the corresponding body score, calculated as a weighted average of the ECTS dimensions, where the weights are set by the eigenvalue associated with each eigenshape. Local maximas and minimas in body score determine boundaries between actions, marked as green and red vertical lines for the beginning and end of actions respectively, in both Panels **A** and **B**. **C** and **D** show the same information as **A** and **B** for *C. elegans*. The sinusoidal segments correspond to locomotion, note that segmentation resolves these into 'steps'.

7.2.6 Curve alignment and clustering

Segmentation produces a large set of subsequences, or actions, each of which is a continuous ECTS curve. Hence splines, locally smooth piecewise polynomials, are a natural choice to parameterise actions. Spline regression [Gaffney and Smyth, 2004, Gaffney, 2004] was used to assign the actions to clusters. This method is analogous to Gaussian mixture models, but instead of Gaussian distributions, clusters are parameterised by splines.

To improve the consistency of spline fitting, the ECTS subsequences are aligned in the time domain. The frame with the highest body score was used as a reference, and actions are shifted in time such that their point of highest body score coincides, see Figure 6.1 for illustration. Note that if $ECTS = [0,0,0]$, then the posture is a flat line (for both organisms). The higher the coefficients are, generally the more curved the postures are (although the bend caused by the coefficients can be in opposite directions and cancel each other). Therefore the maxima of the body score corresponds to the frame with the most bent posture and as such this frame is a rational choice to define a reference point in time by which subsequences of different lengths can be aligned.

Splines had three internal knot points and each polynomial had an order of 3. An expectation-maximisation (EM) algorithm [Gaffney, 2004] was used to learn model parameters. EM was initiated 500 times with random boundary conditions and the solution with the highest likelihood was kept. Bayesian Information Criteria (BIC) [Schwarz et al., 1978, Fraley and Raftery, 1998, Konishi and Kitagawa, 2008] was used to identify the optimal number of clusters. BIC is defined as:

$$BIC = 2ln(\mathscr{L}_{model}) - k * ln(n), \tag{7.2}$$

where $\mathscr{L}_{model}$ is the likelihood of the fitted model, k is the number of free parameters and n is the number of observations. The first term reflects goodness of fit of the model, and the second is a penalty term is for the number of free parameters.

Spline regression clustering produces a membership probability that a given action belongs to a cluster. Therefore this method avoids rigid cluster assignments and also allows classification uncertainty to be quantified. To measure the classification

uncertainty Shannon entropy [Shannon, 2001] was used, defined as:

$$H = -\sum_i p_i log_2 p_i, \tag{7.3}$$

where p_i is the probability that a given action belongs to a cluster i. Note that the most uncertain situation is when the probability is equally distributed among the clusters, correspondingly H has a maximum when all $p_i = 1/i_{max}$ (i_{max} is the number of clusters).

7.2.7 Comparison of behavioural annotations

In the following a 'behavioural event' means an interval of consecutive frames tagged with the same behaviour. A behavioural event marked by an automated annotator (ESA, JAABA or CBD) was counted as true positive if at least 50% of it was also tagged by ground truth annotation with the same behaviour. Otherwise the event was either counted as a false positive (automated annotator marked a behavioural event that had less then 50% overlap with an identically annotated behavioural event in the ground truth annotation) or a false negative (ground truth marked a behavioural event that had less then 50% overlap with an identically annotated behavioural event in the automated annotation).

Furthermore we had to consider the problem that different annotations used different behavioural state spaces. The behaviours are always matched to the closest behaviour in the ground truth annotation. Specifically, for larval *Drosophila* ESA's *turning manoeuvre* was treated as a match to both *stop cast* and *turn* in the ground truth annotation. That is if ground truth contained either a *turn* or a *stop cast* behaviour and at least 50% of the frames are tagged as a *turning manoeuvre* by ESA, then it was counted as a true positive. *Run casts* are the same behaviour across ground truth, JAABA and ESA. For *C. elegans* the ground truth hand annotation's *dwelling* was treated as a match to CBD's *pause* and ESA's *passive* state. The CBD's Upsilon and Omega turns are both treated as a match to the ground truth's *turn* behaviour.

Parts of the time series are excluded from the analysis when the video frames could not be segmented and hence midline information was not accessible. Note that JAABA, CBD and ground truth annotation is available for these periods as they do not exclusively rely on contour information.

We modified the output of JAABA to avoid the problem of 'flickering annotation'. Flickering annotation occurs when single frames within a behavioural event are not classified as part of the event, e.g., the sequence 0011011100 (where 1 means that the frame corresponds to a given behaviour, 0 means it does not). JAABA works on a frame by frame basis, hence these sequences are present when an event is near threshold value. To avoid the false positives caused by the small gaps, we have connected behavioural events that are less then 3 frames apart. Hence the sequence above would become 0011111100.

To summarise annotation accuracy we report the precision (positive predictive value) and sensitivity (also known as recall and true positive rate) [Powers, 2011] in Table 1 and 2. Sensitivity is the percentage of events recognised by the annotator, and precision is the proportion of events tagged by the annotator that are true positive. Furthermore these two measures are combined as the F-score, defined as:

$$F = \frac{2(\text{precision} \times \text{sensitivity})}{\text{precision} + \text{sensitivity}}, \tag{7.4}$$

which is commonly used to quantify the goodness of classification.

7.2.8 Visualisation, density cross sections and feature histograms

To produce Figure 7.3B, 7.4B and 7.5 the standard MATLAB (2014a) implementation of metric multidimensional scaling was used. The distance matrix was constructed using weighted dynamic time warping (DTW), where the weights are set by the eigenvalue associated with each dimension of ECTS. DTW is a standard measure of similarity in time series analysis that uses a non-linear time warping to find the optimal match between a pair of subsequences [Müller, 2007]. Note that the Euclidean distance among the points (corresponding to the actions) on the map correlates with the DTW distance among the subsequences, but the distances on the map are in arbitrary units. To construct each map a random sample of 5000 actions are used. The algorithm was run 500 times with random initial conditions and the solution with the highest R^2 was kept.

The density cross sections of aggregated ECTS curves was visualised to see possible density fluctuations (see 7.3.4). Sets of stereotypical curves would form high density regions in the cross sections. Hence the cross sections can be used to detect stereotypical curves corresponding to a stereotypical posture sequences. Density cross sections are measured on aggregated and aligned ECTS curves at specific 'time slices' as shown in Figure 7.6A. To estimate the density of curves a kernel density estimation method was used [Botev et al., 2010]. Figure 7.6 only shows the cross section for one time slice, see figure B.1 for additional cross sections.

To create the histograms of *C. elegans* behavioural features, data was directly imported from the *C. elegans* behavioural database feature files. These features are defined in [Yemini et al., 2013]. The hardware and software that was used to obtain the behavioural features for larval *Drosophila* is described in [Gomez-Marin et al., 2011].

7.3 Results

7.3.1 Eigenshapes

The eigenworm analysis pipeline extracts a vector of angles between consecutive points along the animal's midline, and applies principle component analysis to reduce the dimensionality of this description. The same method was adapted to create the analogous set of shapes for *Drosophila* larva, the eigenmaggots (Figure 7.2). We find that eigenmaggots (Figure 7.2B) are as efficient to describe larval postures as the eigenworms (Figure 3.1D) are to describe worm postures. The inspection of eigenvalues reveals that three eigenmaggots account for over 90% of the postural variance [Jolliffe, 2002] (Figure 7.2A). Thus eigenmaggots provide an accurate low dimensional description of larval postures.

In contrast to eigenworms, eigenmaggots do not capture forward locomotion [Stephens et al., 2008]. This difference is due to the different mode of locomotion. *C. elegans* propels itself by moving its body in a sinusoidal wave perpendicularly to the direction of motion [Berri et al., 2009]. Larval *Drosophila* crawls forward utilizing peristaltic contraction waves [Heckscher et al., 2012]. The peristaltic waves can be recognized by the contraction of the abdominal sections, but this contraction does not alter the animal's midline shape from the camera's top view, and therefore is not captured by the eigenmaggot description. It is noted here that we have experimented with supplementing the larval ECTS with the tail speed time series as an extra dimension. The idea is that tail speed captures the state of peristalsis. However the additional information did not improve the classification when evaluated against the ground truth annotation.

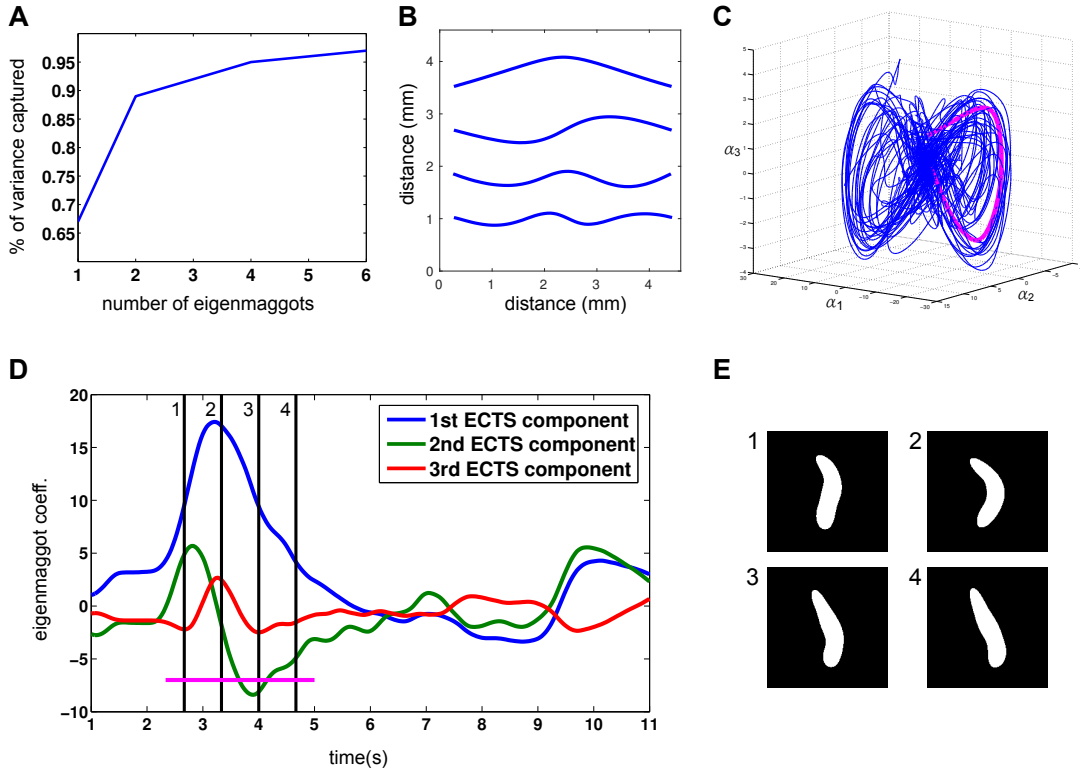


Figure 7.2: Results of eigenmaggot analysis. Panel **A** shows the percentage of the original data's variance recovered given the dimensionality of the representation. **B** shows the eigenmaggots with the most significant eigenmaggot on top, the second below etc. These shapes can be added up in different proportions to reproduce the larval postures (see figure 1). Panel **C** shows a 3D behavioural trajectory in eigenmaggot space, that is the time evolution of the first three eigenmaggot coefficients. The subtrajectory highlighted is an example of what we call a *turning manoeuvre*, see 3.2. **D** shows a part of the same trajectory as three separate one-dimensional time series; the subsequence underlined corresponds to the highlighted subtrajectory on Panel **C**. **E** shows binary images of the maggot at the corresponding time slices from Panel **D**.

7.3.2 Motifs for *Drosophila* larva

For foraging *Drosophila* larva, the BIC for the spline regression model gave the best fit when assuming the presence of two behavioural motifs. The first motif we call a *run cast*. A run cast is a low amplitude head cast while the larva is moving approximately straight [Ohashi et al., 2014, Gomez-Marin and Louis, 2014]. Successive run casts make up the larva's typical forward locomotion. The second motif corresponds to high amplitude head casts that may or may not be followed by a sharp change of direction. Some previous analyses of larval behaviour distinguish 'stop casts' (or simply 'casts'), where the larva stops locomotion and sweeps its head laterally, from 'turns', which start in a bent body shape and end as the larva resumes locomotion in a new direction [Gomez-Marin et al., 2011, Green et al., 1983]. This classification scheme is not unique; others have proposed alternatives [Kane et al., 2013]. We do not find evidence to support the distinction between 'stop casts' and 'turns' instead our analysis describes these behaviours as a single motif, the *turning manoeuvre*. See supplementary video SV1, and Figure 7.3 for an annotated trajectory and a visualisation of the relationship among the motifs.

The maggot behavioural states were also clustered with the $tailSpeed$ added as an additional component to the ECTS. However the same states have been found as without the addition of $tailSpeed$. This finding hints that there are no 'straight' movements of the maggot, rather locomotion is always accompanied by run casting. This conclusion is confirmed by another recent analysis of maggot motion as well [Wystrach et al., 2016].

ESA annotation was evaluated against hand annotation. Across all behaviours ESA produced an F-score of 0.72 (precision=0.67 and sensitivity=0.77), where the dominant source of error was a large number of false positive run casts. On the same behavioural experiments JAABA annotation produced an F-score of 0.68 with many false positive events as well. See Table 7.1 for the precision, sensitivity and F-score statistics for each behaviour for both JAABA annotation and ESA. Supplementary video SV3 shows the binary video of the larva, hand annotation, JAABA and ESA annotations next to each other, so that the reader can gain a good understanding of how the different annotations relate to the larva's behaviour.

Typically, disagreements happen between ESA and hand annotation when an action has high classification uncertainty. Classification uncertainty is quantified by the Shannon entropy [Shannon, 2001] and it is denoted by H. 73% of the ESA actions have a low uncertainty, meaning $H < H_{max}/4$, where $H_{max} = log_2 2$ because two states have been found. For these low uncertainty actions hand annotation and ESA agree on 87%. When classification entropy is high, $H > H_{max}/4$, then the agreement rate between the two annotations drops to 49%. In short, action labels typically differ where ESA is uncertain. When hand annotation and ESA are in disagreement it is often debatable which one is correct. In Section 7.3.4 we argue that the difficulty to resolve disagreements is due to an unbroken continuity between the two behavioural motifs.

	Run Cast			Stop Cast			Turn			All behaviours		
	Pre.	Sen.	F	Pre.	Sen.	F	Pre.	Sen.	F	Pre.	Sen.	F
JAABA	0.49	0.95	0.65	0.67	0.89	0.76	0.53	0.98	0.69	0.54	0.94	0.68
ESA	0.64	0.91	0.75	0.74	0.75	0.75	0.7	0.51	0.59	0.67	0.77	0.72

Table 7.1: Statistics of the annotation of larval *Drosophila* behaviour. Precision, sensitivity and F-score values have been derived from supplementary table S1. See supplementary video SV3 that shows the larva's behaviour ground truth, JAABA annotation and ESA annotation next to each other.

7.3.3 Motifs for *C. elegans*

ESA was developed with the analysis of larval *Drosophila* in mind, but can also be applied to *C. elegans*. The worm behavioural data was obtained from the *C. elegans* behavioural database (CBD). The database contains movies of worms browsing in bacteria, an environment where worms tend to pause for long periods. These pauses required an extra step in the segmentation process, see *Methods* for details.

In this case BIC for the spline regression model fit indicated the presence of three behavioural motifs, corresponding to *locomotion*, *turns* and *passive periods*. Segmentation divides locomotion into 'steps', where each step is a $\pi/2$ advancement of the locomotion wave. Multiple locomotion steps make up the characteristic undulatory motion of the worm. The turn behaviour as defined by ESA includes classic Omega turns, lower amplitude turns and sharp pirouettes as well [Huang et al., 2006]. The

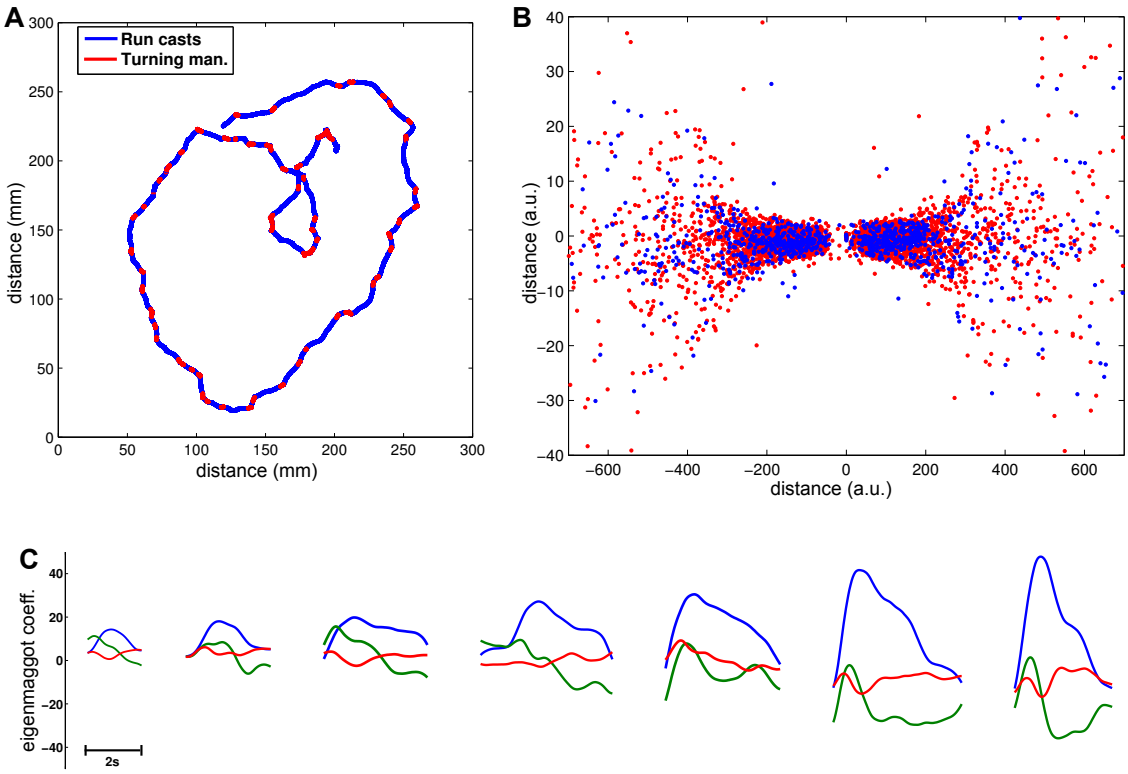


Figure 7.3: The structure of behavioural motifs for larval *Drosophila*. ESA identifies two motifs in the larva's behaviour, **A** shows a trajectory colour coded for the two motifs. Note that turning manoeuvres tend to happen when direction changes. **B** shows a 2D map of the distances among actions as measured by dynamic time warping ($R^2 = 0.85$), see 2.9 for details. **B** uses the same colour scheme as Panel **A** to distinguish behaviours. The symmetry in the figure correspond to the left/right symmetry in the animal's behaviour. Note that the points corresponding to the two behavioural motifs are concentrated in separate regions, yet there is no clear boundary between the two set of points. **C** illustrates that similar ECTS subsequences can be found at every scale. These actions have been selected by starting in the middle of the map in **B** and picking example actions at regularly spaced distances along the x-axis, going from left to right.

passive periods are a mixture of pauses, dwelling and quiescence [Gallagher et al., 2013]. Figure 7.4 shows a visualisation of the relationship between the motifs and an annotated trajectory, and supplementary video SV2 provides a dynamic illustration of the annotation.

To benchmark ESA its performance was compared against hand annotation. ESA produced an F-score of 0.82 (precision=0.74 and sensitivity=0.95), where the domi-

nant source of error was a large number of false positive turn events. This finding is not surprising given that the turning behaviour as defined by ESA is very permissive. Existing automated behavioural annotation of the *C. elegans* behavioural database resulted in an F-score of 0.88 (precision=0.86 and sensitivity=0.9). See Table 7.2 for the precision, sensitivity and F-score statistics for each behaviour for both CBD annotation and ESA. Furthermore see supplementary video SV4 that shows the video of the worm, hand annotation, CBD and ESA annotations next to each other.

As for larval *Drosophila*, there is a significantly increased chance of a *C. elegans* action to be labelled differentially by ESA and hand annotation if the action has a high classification uncertainty ($H > H_{max}/4$, where $H_{max} = log_2 3$ as three behavioural states have been detected) according to ESA. The probability that hand annotation labels these uncertain actions identically decreases to 39% from the population average 77%.

	Locomotion			Turn			Dwelling			All behaviours		
	Pre.	Sen.	F	Pre.	Sen.	F	Pre.	Sen.	F	Pre.	Sen.	F
CBD	0.77	1	0.87	0.96	0.79	0.87	0.89	0.94	0.92	0.86	0.9	0.88
ESA	0.83	0.93	0.9	0.67	1	0.8	0.73	0.83	0.77	0.74	0.95	0.82

Table 7.2: Statistics of the annotation of *C. elegans* behaviour. Precision, sensitivity and F-score values have been derived from supplementary table S2. See supplementary video SV4 that shows the worm's behaviour ground truth, CBD annotation and ESA annotation next to each other.

7.3.4 Do the larva and the worm exhibit discrete behaviours?

For both animals the above analysis produces a substantial proportion of actions (around 25%) for which classification uncertainty is high. This suggests that the identified behaviours are not discrete, where 'discrete' means clearly distinguishable and stereotypical. Rather we see a continuous spectrum of behaviour. This is in contrast with the overwhelming majority of the literature that treats behaviour of these animals as a set of discrete states, although we are not the first to suggest a continuum among

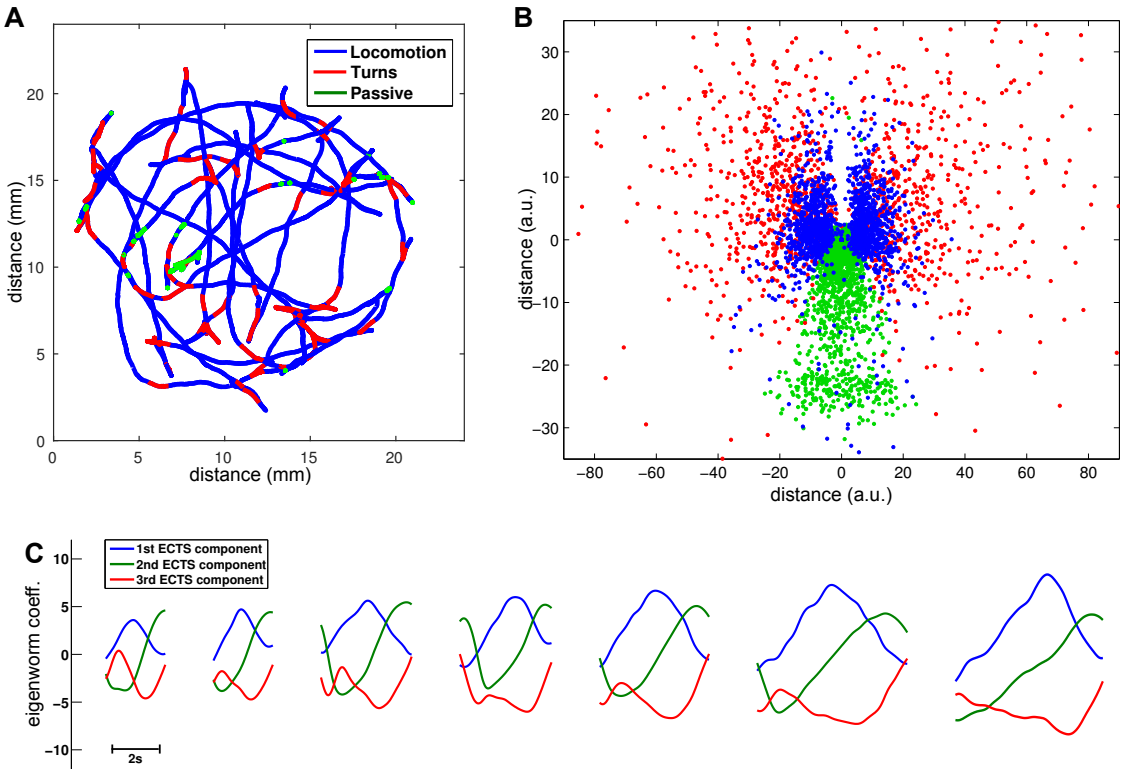


Figure 7.4: The structure of behavioural motifs for *C. elegans*. **A** shows a trajectory colour coded for behaviour. **B** shows a 2D map of the distance among actions as measured by dynamic time warping ($R^2 = 0.78$), see 2.9 for details. The symmetry in the figure correspond to the dorsal/ventral symmetry in the animal's behaviour. Note that turn events are more dense on the negative side of the x-axis. This effect is due to the ventral bias of Omega turns [Huang et al., 2006]. **C** illustrates that the ECTS subsequence corresponding to turns can be found at various scales, indicating that Omega turns are not distinct behaviour, but a part of the continuum of turning behaviours.

behavioural states for *C. elegans* [Gallagher et al., 2013].

To compare our results to what might be expected if there are discrete states, ESA was used to annotate the behaviour of an agent based simulation of *Drosophila* larva which had been developed independently to study chemotaxis [Davies et al., 2015]. The agent's behaviour is controlled by a Markov chain model with three states: stop cast, run cast and straight run. Within each state the precise motion (e.g. body bend) is determined by the current sensory conditions so can vary significantly. Videos are recorded of the agent in its virtual world and the videos are put through the ESA pipeline (i.e. extracting eigenshape representation, segmentation, clustering). In this

way we test the ESA pipeline for its ability to detect underlying discrete states. We also present several alternative analyses that reveal distinct actions in the simulation but suggest a continuum of actions in the real animals.

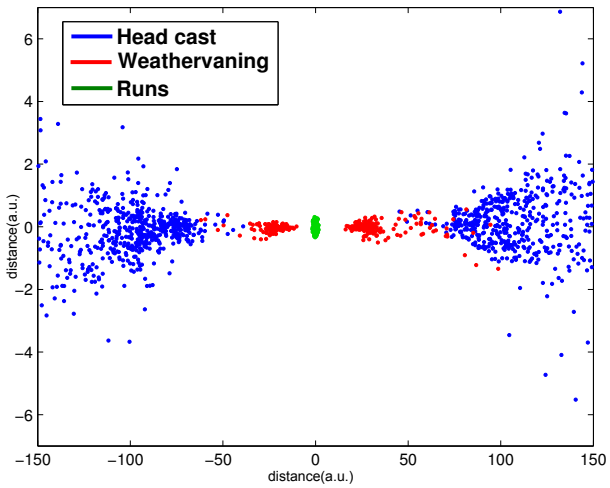


Figure 7.5: Multidimensional scaling visualisation of a virtual agent's behavioural state space ($R^2 = 0.74$), see the *Methods* for details. Note that unlike for the two organisms there is a distinct boundary separating behavioural motifs. This separation is due to the agent behaviour being driven by a Markov model that consists of discrete states.

Clustering results

For the simulated agent ESA produced three clusters and for 94% of the time it produced the same behavioural classification as ground truth annotation. Bayesian information criteria indicated a difference between the agent and the animals. For the agent BIC provided strong evidence to distinguish the three clusters ($\Delta BIC_{min} = 7.57$). In contrast for both *Drosophila* larva and *C. elegans* there was weak statistical evidence to justify the number of clusters (in both cases $\Delta BIC_{min} < 3.75$) [Kass and Raftery, 1995]. In other words BIC is confident that there are three distinct clusters among the agent's actions, but for the two animals, the cluster structure is statistically much less justified.

Structure in aggregated ECTS segments

We can directly examine this difference in cluster structure by visualising the presence or absence of clear density bands in the aggregated ECTS subsequences (see 7.2.8). Sets of stereotypical curves form high density regions in the cross sections, hence the cross sections can be used to detect stereotypical curves corresponding to a stereotypical posture sequences. Figure 7.6A shows the aggregated ECTS curves for the 1st ECTS component of larval *Drosophila*. Figure 7.6B, C and D show the density cross sections for larval *Drosophila*, *C. elegans* and the agent respectively. Note that the positive/negative asymmetry of ECTS values along the x-axis corresponds to the left/right asymmetry in larval behaviour and to the dorsal/ventral distinction for *C. elegans*. For both organisms there is a single band in each half of the x-axis. This profile is in contrast with the two distinct bands of the agent's density cross section. The curves forming each high density band correspond to one Markov state of the agent. Seven cross sections at various x-values are examined in each dimension for both the *C. elegans* and *Drosophila* (Figure B.1), but they all had the same qualitative features as the cross section shown on Figure 7.6, i.e., the animals do not have distinct bands that would support the inference of separable behavioural states.

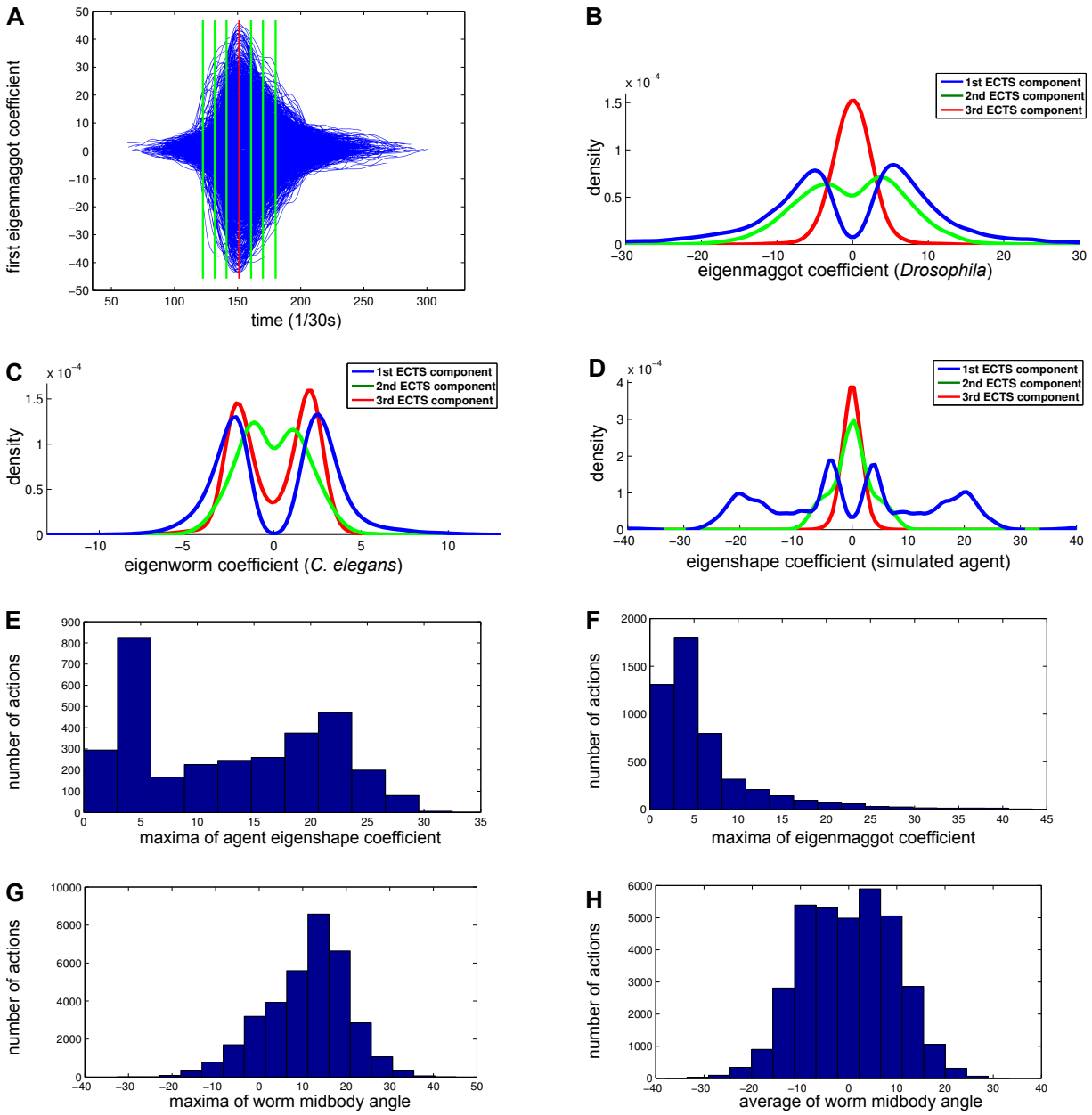


Figure 7.6: Continuity among behavioural states. **A** shows the cross section taken across the aggregated ECTS curves. The cross section across the time of peak curvature (red line on **A**) is shown for *Drosophila* larva, *C. elegans* and the simulated agent on Panel **B**, **C** and **D** respectively. For clarity straight runs are removed from the agent's cross section. **E** and **F** shows the histogram of the maxima of 1st ECTS component during actions for the agent and *Drosophila* respectively. For the agent the bimodal distribution indicates two distinct behaviours, but there is no clear cut-off amplitude for the real organism. **G** and **H** shows the histogram of the maxima and the average of midbody bend for *C. elegans* actions. Again we do not find a multimodal distributions, indicating that there is no data defined threshold to distinguish separate behaviours.

Structure in behavioural features

Weathervaning, or klinotaxis, is a steering process that results in the animal's trajectory bending towards higher concentration of odour [Lockery, 2011]. For *Drosophila* larva, low amplitude head casts are hypothesized to be responsible for weathervaning [Gomez-Marin and Louis, 2014]. These weathervaning casts are distinguished from head casts by the amplitude of body angle [Gomez-Marin and Louis, 2014,Ohashi et al., 2014], which is very closely related to the amplitude of the 1st ECTS component, see Figure 7.2B. The agent's behaviour was coded with this distinction in mind, so head casts tend to cause a higher body angle then weathervaning casts. Figure 7.6E shows the histogram of the maxima of 1st ECTS component during the agent's actions. The bimodal distribution clearly indicates two distinct behaviours. Based on this observation we examined the maxima and average of a number of features of larval *Drosophila* (head speed, head angle, body angle, body angle speed, and head angle speed) and *C. elegans* (eccentricity, head, midbody and tail angles) actions, see Figure 7.6E-H and B.2, B.3. We hoped to find multimodal distributions and possibly sharp cut-off values because these could be used as data defined thresholds to distinguish actions. However in all cases a smooth, unimodal distribution was found.

Multidimensional scaling

A final way to examine this issue is to use multidimensional scaling to visualise the distance matrix of actions. Weighted dynamic time warping (DTW) was used to measure distance, where the weights are set by the eigenvalue associated with each dimension of ECTS. Figure 7.3B and 7.4B show the larval *Drosophila* and *C. elegans* maps respectively. As can be seen there is no clear boundary in either figure to unambiguously separate behavioural motifs. This is in contrast with the agent's map, figure 7.5, where clearly separated regions can be seen.

7.4 Discussion

This paper introduces eigenshape annotation (ESA), a bottom-up unsupervised method that searches for frequently repeated posture sequences in behavioural data. This problem is closely related to behavioural annotation, but not identical to it. Most behavioural annotators recognise behaviours through user defined thresholds or training data [Kabra et al., 2013, Ohyama et al., 2013, Gomez-Marin et al., 2011, Salvador et al., 2014, Yemini et al., 2013]. In both cases the set of possible behaviours and the description of those behaviours are determined by the user. In contrast ESA is trying to discover the behavioural states directly from the data without any user input. Note that this task is considerably more challenging than behavioural annotation due to the lack of *a priori* constraints. Thus the novelty of this work is to create a data processing pipeline that discovers behavioural motifs in an unsupervised manner, where a behavioural motif is defined as a frequently repeated posture sequence.

The behavioural motifs discovered are generally consistent with behaviours described in the literature. However many ESA motifs are more permissive then the definitions in other studies. For example the ESA 'turning manoeuvre' for larva includes turns and high amplitude head casts [Gomez-Marin et al., 2011], while the ESA 'turning behaviour' for the worm is a mixture of classic and wide Omega turns [Huang et al., 2006, Yemini et al., 2013]. In both cases, there was no justification in the data for making any further subdivision of turns. Note that it can also be difficult for human observers to distinguish these behaviours consistently.

ESA was also unable to unambiguously classify many actions. The seeming continuity of the action distance maps, figures 7.3B and 7.4B, motivated us to further consider whether there are 'defining features' that could objectively distinguish behaviours. In a simulated agent that was coded with distinct behavioural states, it is straightforward to find such features, e.g., the amplitude of body bend (Figure 7.6E). We searched for multimodal distributions in a variety of features of the *Drosophila* and *C. elegans* data, but failed in both cases. It remains possible that some feature we did not consider might reveal multimodality, or that discrete behaviours can be distinguished by considering a combination of multiple features.

There is an extensive literature that treats the behaviour of these animals as a set of discrete states. Despite our observation of continuity among behavioural states, our results are not necessarily in contradiction with the discrete treatment of behaviour. Discrete states can be seen as coarse graining (or binning) the continuous behavioural states. For example the *C. elegans* behavioural database defines Omega turns as a bend greater then $\pi/6$ propagating through the body. If the bend is between $\pi/12$ and $\pi/6$ then the event is called an Upsilon turn. Thus this classification scheme treats turning as a two state variable (Omega/Upsilon turn). In contrast ESA produces a membership probability that an action is a turn, instead of discretizing non-turns, Upsilon and Omega turns at arbitrary thresholds. Coarse graining simplifies the underlying postural dynamics and it can be an appropriate simplification for many studies. For example the *C. elegans* behavioural database's turn annotation is appropriate for studies looking at the worm's biased random walk. On the other hand if an analysis requires the precise characterisation of the worm's turning behaviour then the continuous classification scheme of ESA can be advantageous.

However, adopting a coarse grained description for convenience does not justify the widespread treatment in the research literature of behaviour as actually consisting of a set of discrete states, an assumption that needs to be independently evaluated. There is a risk that initially arbitrary distinctions between behaviours have become reified as qualitatively distinct behaviours of the animal, and treated as a set of actions between which it selects. For example, it is sometimes assumed that the underlying neural activity has a modularity that matches the behavioural states, and that this should guide investigation of neural circuits. In our results, the lack of stereotypical and distinguishable behavioural states suggests that the underlying neural activity is not stereotypical or modular. It remains possible that a highly stereotypical activity pattern of neurons implements a behavioural state, but due to biomechanical effects the resulting posture sequences are not so stereotypical. These alternate possibilities can only be addressed by studies of neural activity that do not exclusively depend on behavioural annotators that make *a priori* assumptions about the existence of discrete states.

A further possibility is that the lack of discrete actions observed in our study was a consequence of the particular behavioural conditions in which the animals are tested. Both environments are free of stimulus gradients: larval *Drosophila* was crawling on plain agar; while *C. elegans* was browsing in bacteria (although the bacterial layer

could have minor inhomogeneities leading to shallow gradients). In future work we will examine if the behavioural space changes under different environmental conditions, for example, during directed chemotaxis in larval *Drosophila*.

Eigenshape annotation could be improved by advances in computer vision. Standard thresholding and skeltonising algorithms fail when the animal intersects itself (7.2.4). The exclusion of self-intersecting postures introduces a bias to the pipeline, as no posture with self intersection is included in the analysis. It is a possibility that there are discrete elements of behaviour in the self intersecting sequences of postures.

The idea behind ESA is to find motifs in behaviour. We represented behaviour as posture, and posture as an eigenshape coefficient time series (ECTS), but the framework presented is not specific to either. ECTS can be replaced with any time series capturing behavioural features, or alternatively ECTS can be supplemented with such time series. Time series of higher level features provide extra information for the classifier, potentially increasing its accuracy. For example, including a 'direction of locomotion' time series could lead to the detection of reversals as a separate state.

Alternative motif-finding algorithms could be used on ECTS as well. For example the subsequences yielded by segmentation can also be clustered using distance based methods. We have experimented with several methods [Rodriguez and Laio, 2014, Ankerst et al., 1999] in combination with standard distance measures (Euclidean and dynamic time warping), but it always led to results inferior to spline regression clustering in terms of the classification performance evaluated against hand annotation. We think that the performance difference is due to the ambiguous separation of clusters. Because of its probabilistic nature, spline regression clustering is better equipped to deal with datasets where many of the entries can not be unambiguously classified.

Finally we note that motif discovery is a challenging problem and it is an area of intense research in the machine learning community. Due to the abundance of sequencing data most of the effort is focused on discrete, one dimensional time series. To the best of our knowledge the combination of segmentation and clustering is a novel approach to multidimensional motif finding. As discussed earlier the framework is not specific to ECTS, therefore we expect that with minor modifications the framework

could make contributions in other applications as well.

Chapter 8

Conclusions and future work

8.1 Introduction

One of the most popular aims in current biology is to link genes to behaviour. This task can only be completed successfully using sophisticated tools to analyse behaviour. In Chapter 2, the study of Omega turn annotations, it was demonstrated that the current practices are inadequate, as both expert annotation and automated methods are inconsistent. Given the inconsistency, one could easily imagine a scenario in which a given method indicates a phenotype, but another method yields a null result. Such disagreements are unacceptable, as they question the objectivity of biological research.

A potential answer to both problems (lack of consistency and learnt biases) is to use unsupervised methods that directly construct behavioural annotation from data. This thesis has developed such a pipeline, the eigenshape annotator (ESA). While conceptually, the eigenshape annotator represents an advance, it is not without faults. In this final chapter, I will present a brief overview of the thesis and then critically discuss ESA's weak points and make suggestions for improvement. Finally, I will discuss some of the conceptual issues that emerged during this study.

8.2 Overview and novel aspects

A study of *C. elegans*'s Omega turn behaviour was presented (Chapter 2). The findings demonstrate the inconsistency of expert annotation and current threshold-based methods, thus justifying why unsupervised methods should be explored.

A core element of ESA is that it represents behaviour as a posture time series. This translation is accomplished through eigenshapes (Chapter 3). The use of eigenshapes allowed me to investigate the behaviour of *C. elegans*, larval *Drosophila* and an artificial agent using the same conceptual tools.

In order to avoid the pitfalls of the commonly used sliding window motif discovery approaches (Chapter 4), I have developed an alternative: the segmentation clustering-framework. The key insight from the segmentation-clustering framework is that motif discovery can be decomposed into two separate tasks. First, the subsequences of potential interest are found (see Chapter 5) and then the subsequences are clustered (see Chapter 6) to obtain motifs. The main improvement with this method is that it does not rely on the 'closest neighbours' definition of motifs, whereby every motif is defined as having exactly two instances. Second, it does not require the assumption that motifs of the same class will be exactly equal in length.

The resulting motif discovery pipeline, the ESA, was used to investigate the behaviour of *Drosophila* and *C. elegans*. The main scientific finding was that many elements of behaviour cannot be unambiguously classified (Chapter 7). This finding suggests that the identified behaviours are not discrete (where 'discrete' means clearly distinguishable and stereotypical) but rather form a continuous spectrum of behaviours.

8.3 Limitations and possible extensions

Coiling shapes

One inherent limitation of my work is that coiling shapes could not be processed. Both *C. elegans* and *Drosophila* coil when performing sharp turns; therefore, the characterisation of these behaviours remains incomplete (see 3.2.2 and 7.2.1). Efforts have been made by other researchers to solve this problem [Broekmans et al., 2016, Huang et al., 2006], but at the time of my work, no robust method was publicly available. A straightforward way to improve ESA would be to solve the issue of coiling shapes (i.e. obtain the eigenshape coefficients during coils) and to fill the gaps in the eigenshape coefficient time series. The inclusion of these frames would yield a more complete description of behaviour and could lead to insights regarding the hypothesized continuous nature of turning behaviours.

Extending the analysis to the temporal domain

ESA discovers behavioural states from data, but it does not analyse the temporal relation of states. Consider two behavioural states, *A* and *B*. *ABABAB* and *AAABBB* are clearly distinct behavioural patterns, but in its current form, ESA would not be able to distinguish them, as they are composed of the same elements.

A common technique used to study temporal relations is the Markov chain models. However the Markov property, or 'memorylessness,' i.e. the assumption that the next state is only dependent on the current state, is likely to be false for behaviour. For example, neuromodulation provides a clear example of how intermediate scale processes modulate behaviour [Bargmann, 2012, Cohn et al., 2015]. Therefore, alternative frameworks such as additive Markov chains could be considered [Melnyk et al., 2006]. Combining the automated state discovery of ESA with temporal analysis could provide a powerful framework for investigating behaviour.

Phenotyping and comparison of experimental conditions

Identifying the links between genes and behaviour is one of the key driving topics in biology. I have not explored ESA as a phenotyping pipeline, but it could be used in that capacity as well. By running the ESA on several genotypes, the structures of the resulting states could be compared. A phenotype would be identified if the behavioural states of two genotypes were different beyond chance.

To the best of my knowledge, such an approach would be a novel way to look at the phenotyping problem. Most phenotyping tools work with a fixed set of features and predefined behaviours. For example, [Yemini et al., 2013] has a definition of Omega turns and using this same method, estimates the Omega turn probability for many genotypes. However, the possibility that Omega turns could be altered by genetic manipulations is never addressed. This possibility of 'non-static' or 'fluid' behavioural states could be examined using ESA to discover the behavioural states from the data.

A similar problem is characterising behaviour under various environmental conditions. For example, the switch in the order of eigenworms when on- and off-food is a clear sign that the behaviour is altered in these situations (see 3.4). A systematic study of many environmental and stimulus conditions (e.g. stimulus-free environment,

chemotaxis, phototaxis, etc.) using ESA could yield insights into how sensory stimuli modulate the structure of behavioural states.

Real-time behavioural classification

At the moment, the eigenshape annotator is an off-line method, but it could be modified to act as an on-line (real time) classifier. This is an important issue, as optogenetics have made it feasible to activate neuronal groups at will. For example, the activation of reward circuits when the animal encounters a specific stimulus could yield insights into the mechanisms of learning.

Recall that in ESA, a segment is defined as a local maximum bounded by two local minimums. This seemingly makes live behavioural classification impossible, since after a local maximum, the method needs to wait until a local minimum has been reach to recognise a segment. However, by predetermining the set of states, the segmentation step could be eliminated. Once the ECTS curves for each action are known, a particular behaviour could be detected if, based on the time series observed so far, the probability of the fragments constituting a behaviour reaches a certain threshold value (it is assumed here that the speed of converting live behavioural video to an eigenshape time series would not be prohibitive). Expanding the use of ESA in this manner could make it a more versatile tool.

8.4 Conceptual considerations

8.4.1 Discrete behavioural states versus a spectrum of behaviour

Due to their inherent variability, the classification of biological phenomena often reveals hard-to-classify, borderline cases. As a consequence, finding such clearly separated behavioural states as in the case of the simulated agent larva (see Section 7.3.4) is unlikely. Therefore, the question arises of how distinct behavioural states must be in order to consider them discrete?

Consider that in a given situation, 1% of the behavioural events have a considerable membership probability ($15\% \leq p_{membership_i}$) in all states. In this case, most investigators would think of the hard-to-classify 1% as noise and treat these states as discrete. However, what if 5%/10%/15% of the events have a considerable membership prob-

ability in all states? In the 15% case, it is not so easy to dismiss the hard-to-classify elements as noise. The point is that there is no objective threshold that could help to determine when a set of actions is discrete. Hence, I think that the question of whether behaviour is a spectrum or a set of discrete states is also a question of community consensus.

As argued in 7.4, despite our observation of continuity among behavioural states, our results are not necessarily in contradiction with the discrete treatment of behaviour. Depending on the specific question in mind, a coarse graining of behavioural state space into discrete entities could make sense. However, it should be kept in mind that the hypothesis that behaviour consists of discrete elements should be evaluated independently. There is a risk that initially arbitrary distinctions between behaviours have become reified as qualitatively distinct behaviours of the animal.

The current literature on behavioural analysis is dominated by analysis that relies on discrete states. I believe this is partially inspired by the convenience of working with discrete data. For example, once discrete states are constructed, it is straightforward to conduct a Markov chain analysis. In contrast, there are far fewer tools available for analysing continuous data. Therefore, for the spectrum of behaviour hypothesis to be further explored, more analysis methods are first needed.

8.4.2 Tools for studying the behavioural spectrum

If behaviour is a spectrum, what analytic tools should be used to study it? Eigenshape annotation provides a means to capture this spectrum through the membership probabilities it produces. For example, consider two behavioural elements of larval of *Drosophila*: one is 50% head-cast and 50% turning manoeuvre, while the other is 20% head-cast and 80% turning manoeuvre. In this case, it is clear that neither event is a good example of either state, but that they are positioned differently on the behavioural spectrum.

While eigenshape annotation offers a way to capture the behavioural spectrum, it was not designed with this possibility mind. Specifically, the segmentation of the eigenshape coefficient time series assumes that it makes sense to separate behaviour into discrete elements. Therefore, eigenshape annotation implicitly assumes discrete

states and then reveals some of the limitations of this framework.

One way to avoid the segmentation step (and with it, the implicit assumption of states) is to represent the eigenshape coefficient time series using wavelets. Wavelets offer a compromise between time and frequency domain representations, while simultaneously capturing time series at multiple length scales. Therefore, they are an ideal representation for finding behavioural patterns without predetermined lengths. The combination of eigenshapes and wavelet analysis has been explored previously [Berman et al., 2014, Todd et al., 2016], but these studies exclusively looked for behavioural states, never exploring the possibility of a behavioural spectrum.

8.4.3 The problem with ground truth comparisons

In the case of behavioural studies, automated methods are often evaluated against ground truth datasets. In practice, ground truth datasets are produced by laboratory members and are therefore an artificial reference. The issue with this practice is that by using human annotation as an evaluation metric, the automated methods essentially learn the same biases that make human annotation undesirable in the first place.

The evaluation of unbiased methods raises methodological and philosophical problems [Todd et al., 2016, Jain et al., 1999]. The core issue is that since the comparison with ground truth is undesirable, there is no obvious metric for evaluating such methods. On the other hand, unsupervised methods should certainly not be trusted blindly; the validity of their results should be verified in one way or another. These questions are beyond the scope of this thesis, but the reader should be aware of them. Conceptual advancements in model validation should be followed and applied to behavioural studies.

Finally, I note that throughout this thesis I have evaluated various methods against ground truth data. However, in no case was this meant as an absolute verification of the method. Rather, it was meant as a safety check to verify that the method was not producing incomprehensible results.

Appendix A

Quantitative evaluation

Thrughout the thesis ground truth data is compared with the annotation of an automated tool. Here the terms used for the compariosn are defined.

A 'behavioural event' means an interval of consecutive frames tagged with the same behaviour. A behavioural event marked by an automated annotator was counted as true positive if at least 50% of it was also tagged by ground truth annotation with the same behaviour. Otherwise the event was either counted as a false positive (automated annotator marked a behavioural event that had less then 50% overlap with an identically annotated behavioural event in the ground truth annotation) or a false negative (ground truth marked a behavioural event that had less then 50% overlap with an identically annotated behavioural event in the automated annotation).

To summarise annotation accuracy we report the *Precision* (also known as positive predictive value) and *Sensitivity* (also known as recall and true positive rate) [Powers, 2011] in Table 5.1. These statistics are defined as:

$$Precision = \frac{TP}{TP+FP}, \tag{A.1}$$

$$Sensitivity = \frac{TP}{TP+FN}, \tag{A.2}$$

where TP, FP and FN stands for the number of *true positive*, *false positive* and *false negative* events respectively. In words *Precision* is the proportion of events tagged by the annotator that are true positive and $Sensitivity$ is the percentage of true events recognised by the annotator. Furthermore these two measures are combined in the *F-score* defined as:

$$F = \frac{2(Precision \times Sensitivity)}{Precision + Sensitivity}, \tag{A.3}$$

which is combination of the *Precision* and *Sensitivity* and is commonly used as a one number summary for the goodness of classification [Powers, 2011].

Furthermore we had to consider the problem that different annotations used different behavioural state spaces. The behaviours are always matched to the closest behaviour in the ground truth annotation. Specifically, for larval *Drosophila turning manoeuvre* was treated as a match to both *stop cast* and *turn* in the ground truth annotation. That is if ground truth contained either a *turn* or a *stop cast* behaviour and at least 50% of the frames are tagged as a *turning manoeuvre*, then it was counted as a true positive. For *C. elegans* the ground truth hand annotation's *dwelling* was treated as a match to CBD's *pause* and the *passive* state. The CBD's Υ and Ω turns are both treated as a match to the ground truth's *turn* behaviour.

Parts of the time series are excluded from the analysis when the video frames could not be segmented and hence midline information was not accessible. Note that JAABA, CBD and ground truth annotation is available for these periods as they do not exclusively rely on contour information.

Appendix B

Supplementary figures

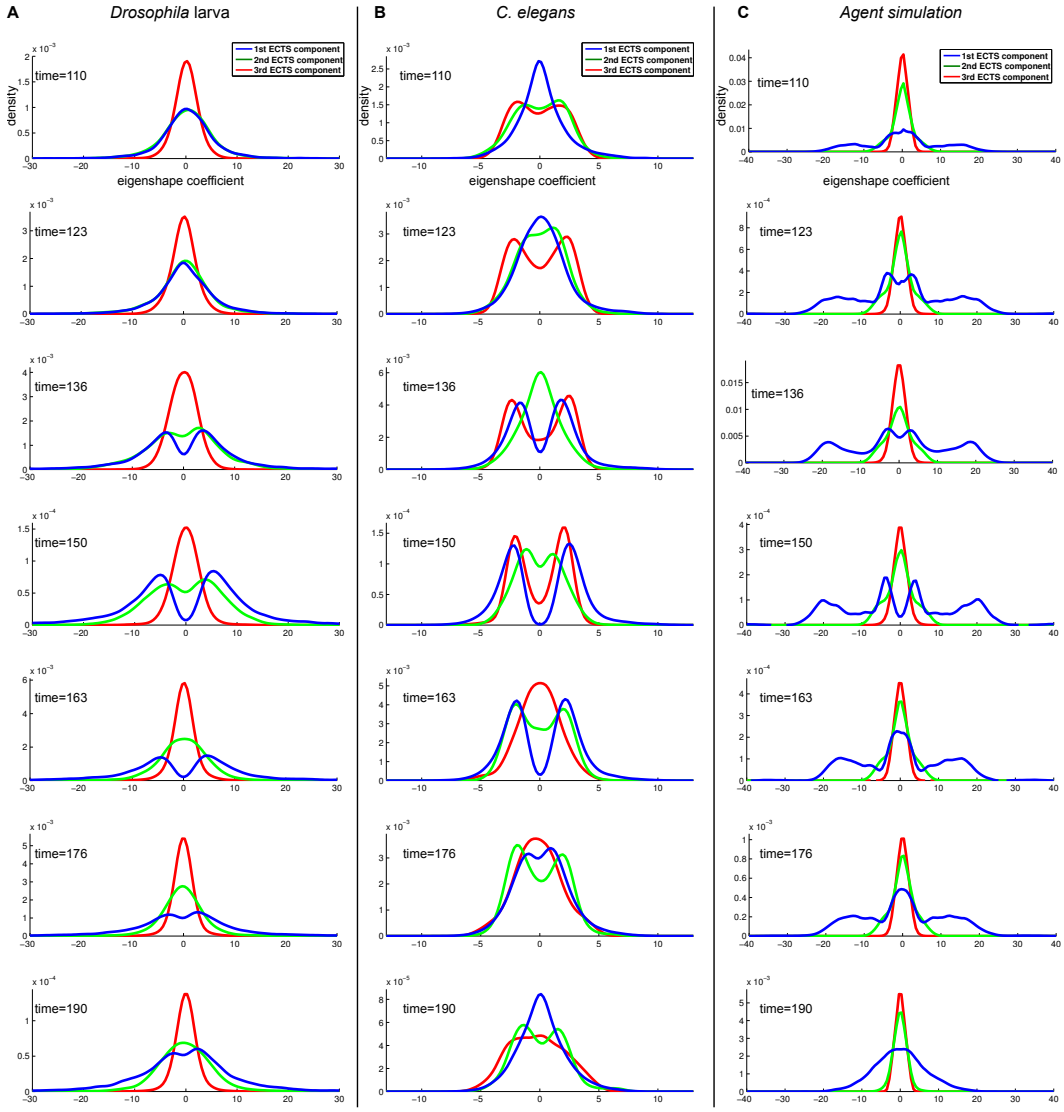


Figure B.1: Density cross sections of aggregated ECTS curves. Each column shows density cross sections for larval *Drosophila*, *C. elegans* and the agent simulation. Each row corresponds to a different time slice, see Figure 5A for explanation. Note that in this case time is given in frame numbers, that is each increment in t corresponds to 1/30s. Bands in this figure correspond to typical ECTS curves, which represent typical sequences of body shapes. With the exception of the agent simulation, each curve is smooth on each side of 0, indicating a lack of well defined discrete states.

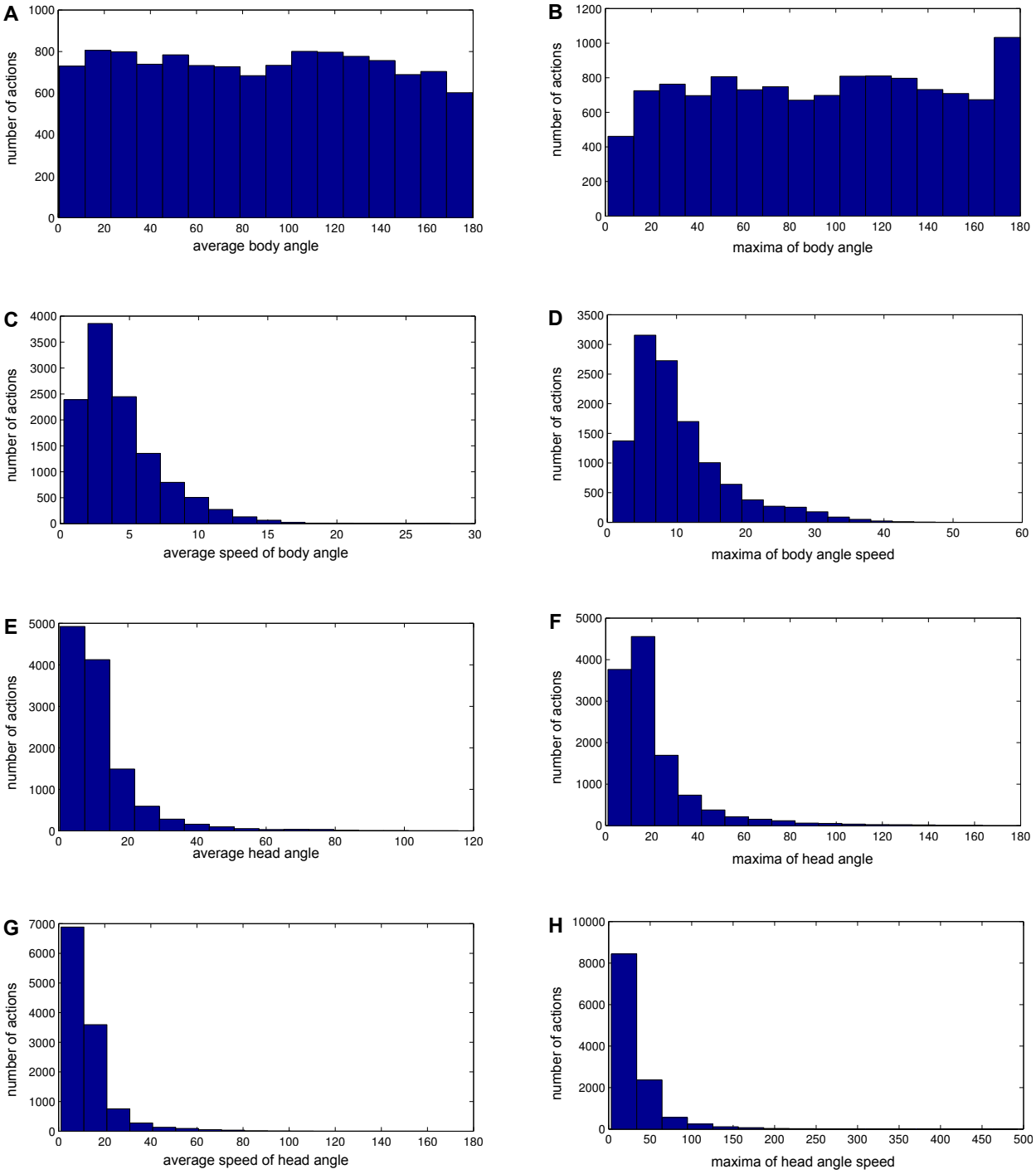


Figure B.2: Histograms of larval *Drosophila* action features. In the following both the maxima and the averages are measured on individual actions, that is each action contributes a single data point to these histograms. **A** and **B** displays the histogram of average and maxima of body angles, while **C** and **D** shows the histogram of average and maxima of the speed body angle change. Similarly **E** and **F** depicts the histogram of average and maxima of head angles, while **G** and **H** shows the histogram of average and maxima of the speed head angle change. For a precise definition of these features see [Gomez-Marin et al., 2011].

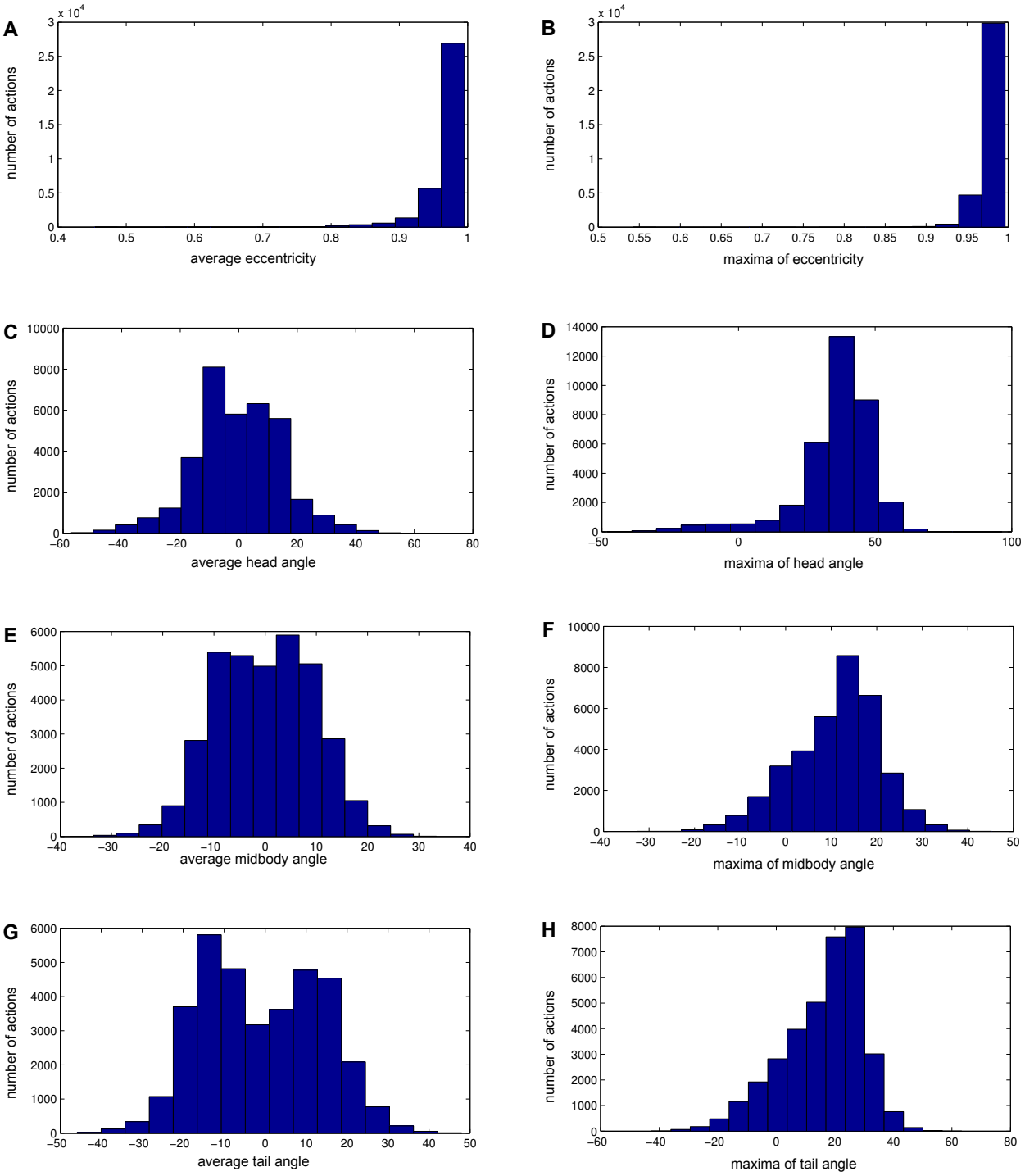


Figure B.3: Histograms of *C. elegans* action features. In the following both the maxima and the averages are measured on individual actions, that is each action contributes a single data point to these histograms. **A** and **B** displays the histogram of average and maxima of eccentricity. **C** and **D** shows the histogram of average and maxima of head angle. Similarly **E** and **F** depicts the histogram of average and maxima of mid body angles, note that the asymmetry along the x-axis corresponds to the dorsal/ventral asymmetry in behaviour. Panels **G** and **H** show the histogram of average and maxima of tail angle. For a precise definition of these features see [Yemini et al., 2013].

Appendix C

Supplementary tables

	Trianlges			R.A. triangles			Sinusoidal			All patterns		
	TP	FP	FN	TP	FP	FN	TP	FP	FN	TP	FP	FN
CS	501	2	23	499	1	36	493	5	35	1493	7	94

Table C.1: Counting the number of TP/FP/FN (true positive / false positive / false negative) annotations of the segmentation-clustering method on sythetic data. For details of the experiment and the derived statistics ($Precision$,$Sensitivty$,$F-score$) see section 4.4.2.

	Run Cast			Stop Cast			Turn			All behaviours		
	TP	FP	FN	TP	FP	FN	TP	FP	FN	TP	FP	FN
K-means	87	83	33	33	13	11	24	20	37	144	116	81
Regression	109	61	11	34	12	11	31	13	30	174	86	52

Table C.2: Counting the number of TP/FP/FN (true positive / false positive / false negative) annotations of the SC algorithm using K-means/spline regression clustering on larval*Drosophila* data. For details of the experiment and the derived statistics ($Precision$,$Sensitivty$,$F-score$) see section 6.3.2 and for the procedure of the compariosn see A in the Supplementary materials.

	Locomotion			Turn			Dwelling			All behaviours		
	TP	FP	FN	TP	FP	FN	TP	FP	FN	TP	FP	FN
K-means	45	33	21	60	44	10	21	18	15	126	95	46
Regression	65	13	1	70	34	0	29	12	7	164	59	8

Table C.3: Counting the number of TP/FP/FN (true positive / false positive / false negative) annotations of the SC algorithm using K-means/spline regression clustering on *C. elegans* data. For details of the experiment and the derived statistics ($Precision$,$Sensitivty$,$F-score$) see section 6.3.2 and for the procedure of the compariosn see A in the Supplementary materials.

	Run Cast			Stop Cast			Turn			All behaviours		
	TP	FP	FN	TP	FP	FN	TP	FP	FN	TP	FP	FN
JAABA	114	117	6	54	27	7	44	39	1	212	183	14
ESA	109	61	11	34	12	11	31	13	30	174	86	52

Table C.4: Counting the number of TP/FP/FN (true positive / false positive / false negative) annotations of the segmentation-clustering and the JAABA methods on larval *Drosophila* behaviour. For details of the experiment and the derived statistics ($Precision$,$Sensitivty$,$F-score$) see section 7.3.2.

	Locomotion			Turn			Dwelling			All behaviours		
	TP	FP	FN	TP	FP	FN	TP	FP	FN	TP	FP	FN
CBD	66	20	0	55	2	15	34	4	2	155	26	17
ESA	65	13	1	70	34	0	29	12	7	164	59	8

Table C.5: Counting the number of TP/FP/FN (true positive / false positive / false negative) annotations of the segmentation-clustering and the CBD methods on *C. elegans* behaviour. For details of the experiment and the derived statistics ($Precision$,$Sensitivty$,$F-score$) see section 7.3.3.

Appendix D

Video captions

SV1 (OmegaSurvey_ambiguous.mp4) Ambiguous Omega turns. This video shows the 20 Omega like events that all participants of the Omega turn survey were required to score. For details how these events were selected see Section 2.2.3.

SV2 (OmegaSurvey_ranking.mp4) The video shows Omega-like events with increasing tightness score. This movie demonstrates that the tightness score ranks events from wide amplitude to sharp turns. See Section 2.2.3 for details how the tightness score was constructed.

SV3 (maggotESAannotation.mp4) Segmentation of larval ECTS. Top left panel shows a binary video of a tracking experiment. Top right panel shows the midline of the larva reconstructed from ECTS. The middle panel show the time evolution of the first three components of ECTS. Green and red vertical lines mark the beginning and the end of actions respectively. Note that if the end of an action coincidences with the beginning of the next action, then only red vertical line will show. The blue vertical line in the middle marks the time corresponding to the video frame in the two top panels. Bottom panel shows the behavioural annotation of ESA. The annotation is probabilistic, height corresponds to the probability that the action is an example of a certain behaviour.

SV4 (wormESAannotation.mp4) Segmentation of worm ECTS. Top left panel shows video of a tracking experiment with the midline and contour of the worm highlighted. Data was acquired from the *C. elegans* behavioural database. Top right panel shows the midline of the worm reconstructed from ECTS. The middle panel show the time

evolution of the first three components of ECTS. Green and red vertical lines mark the beginning and end of actions respectively. Note that if the end of an action coincidences with the beginning of the next action, then only red vertical line will show. The blue vertical line in the middle marks the time corresponding to the video frame in the two top panels. Bottom panel shows the behavioural annotation of ESA. The annotation is probabilistic, height corresponds to the probability that the action is an example of a certain behaviour.

SV5 (maggotAnnotationCompare.mp4) Comparison of larval behavioural annotations. Top left panel shows a binary video of a tracking experiment, while the top right panel shows the midline of the larva reconstructed from ECTS. The bottom panel shows the ground truth, JAABA and ESA annotation going from top to bottom. For better visualization all three annotations are shown on the same panel, but height only has significance for the ESA annotation. ESA annotation is probabilistic, height corresponds to the probability that the action is an example of a certain behaviour. For ground truth and JAABA, the annotation is binary (either 1 or 0). The blue vertical line in the middle marks the time corresponding to the video frame in the two top panels. When midline disappears in the top right panel, the vision algorithm could not isolate the larva from the background. For small gaps (less then 15 frames) ECTS was interpolated. Wider gaps could not be reliable interpolated, hence these times were excluded from the ESA analysis.

SV6 (wormAnnotationCompare.mp4) Comparison of worm behavioural annotations. Top left panel shows a video of a tracking experiment, while the top right panel shows the midline of the larva reconstructed from ECTS. The bottom panel shows the ground truth, CBD and ESA annotation going from top to bottom. For better visualization all three annotations are shown on the same panel, but height only has significance for the ESA annotation. ESA annotation is probabilistic, height corresponds to the probability that the action is an example of a certain behaviour. For ground truth and CBD, the annotation is binary (either 1 or 0). The blue vertical line in the middle marks the time corresponding to the video frame in the two top panels. When midline disappears in the top right panel, the vision algorithm could not isolate the larva from the background. For small gaps (less then 15 frames) ECTS was interpolated. Wider gaps could not be reliable interpolated, hence these times were excluded from the ESA analysis.

Bibliography

[Albrecht and Bargmann, 2011] Albrecht, D. R. and Bargmann, C. I. (2011). High-content behavioral analysis of *Caenorhabditis elegans* in precise spatiotemporal chemical environments. *Nature methods*, 8(7):599–605.

[Aleman-Meza et al., 2015] Aleman-Meza, B., Jung, S.-K., and Zhong, W. (2015). An automated system for quantitative analysis of *Drosophila* larval locomotion. *BMC Developmental Biology*, 15(1):11.

[Ankerst et al., 1999] Ankerst, M., Breunig, M. M., Kriegel, H.-P., and Sander, J. (1999). Optics: ordering points to identify the clustering structure. In *ACM Sigmod Record*, volume 28, pages 49–60. ACM.

[Baek et al., 2002] Baek, J.-H., Cosman, P., Feng, Z., Silver, J., and Schafer, W. R. (2002). Using machine vision to analyze and classify *Caenorhabditis elegans* behavioral phenotypes quantitatively. *Journal of Neuroscience Methods*, 118(1):9–21.

[Bargmann, 2012] Bargmann, C. I. (2012). Beyond the connectome: how neuromodulators shape neural circuits. *Bioessays*, 34(6):458–465.

[Berman et al., 2014] Berman, G. J., Choi, D. M., Bialek, W., and Shaevitz, J. W. (2014). Mapping the stereotyped behaviour of freely moving fruit flies. *Journal of The Royal Society Interface*, 11(99):20140672.

[Berri et al., 2009] Berri, S., Boyle, J. H., Tassieri, M., Hope, I. A., and Cohen, N. (2009). Forward locomotion of the nematode *C. elegans* is achieved through modulation of a single gait. *HFSP journal*, 3(3):186–193.

[Berry et al., 2007] Berry, M. W., Browne, M., Langville, A. N., Pauca, V. P., and Plemmons, R. J. (2007). Algorithms and applications for approximate nonnegative matrix factorization. *Computational statistics & data analysis*, 52(1):155–173.

[Botev et al., 2010] Botev, Z., Grotowski, J., Kroese, D., et al. (2010). Kernel density estimation via diffusion. *The Annals of Statistics*, 38(5):2916–2957.

[Boyle, 2009] Boyle, J. H. (2009). *C. elegans locomotion: an integrated approach*. PhD thesis, University of Leeds.

[Broekmans et al., 2016] Broekmans, O. D., Rodgers, J. B., Ryu, W. S., and Stephens, G. J. (2016). Resolving coiled shapes reveals new reorientation behaviors in *C. elegans*. *arXiv preprint arXiv:1603.04023*.

[Brown et al., 2013] Brown, A. E., Yemini, E. I., Grundy, L. J., Jucikas, T., and Schafer, W. R. (2013). A dictionary of behavioral motifs reveals clusters of genes affecting *Caenorhabditis elegans* locomotion. *Proceedings of the National Academy of Sciences*, 110(2):791–796.

[Cassisi et al., 2012] Cassisi, C., Montalto, P., Aliotta, M., Cannata, A., and Pulvirenti, A. (2012). Similarity measures and dimensionality reduction techniques for time series data mining. *Advances in data mining knowledge discovery and applications*.

[Chiu et al., 2003] Chiu, B., Keogh, E., and Lonardi, S. (2003). Probabilistic discovery of time series motifs. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 493–498. ACM.

[Cho and Fryzlewicz, 2015] Cho, H. and Fryzlewicz, P. (2015). Multiple-change-point detection for high dimensional time series via sparsified binary segmentation. *Journal of the Royal Statistical Society Series B*, 77(2):475–507.

[Chu and Wong, 1999] Chu, K. K. W. and Wong, M. H. (1999). Fast time-series searching with scaling and shifting. In *Proceedings of the eighteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 237–248. ACM.

[Cohn et al., 2015] Cohn, R., Morantte, I., and Ruta, V. (2015). Coordinated and compartmentalized neuromodulation shapes sensory processing in *Drosophila*. *Cell*, 163(7):1742–1755.

[Croll, 1975] Croll, N. A. (1975). Components and patterns in the behaviour of the nematode *Caenorhabditis elegans*. *Journal of Zoology*, 176(2):159–176.

[Davies et al., 2015] Davies, A., Louis, M., and Webb, B. (2015). A model of *Drosophila* larva chemotaxis. *PLoS Comput Biol*, 11(11):e1004606.

[De Boor et al., 1978] De Boor, C., De Boor, C., Mathématicien, E.-U., De Boor, C., and De Boor, C. (1978). *A practical guide to splines*, volume 27. Springer-Verlag New York.

[Ding et al., 2008] Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., and Keogh, E. (2008). Querying and mining of time series data: experimental comparison of representations and distance measures. *Proceedings of the VLDB Endowment*, 1(2):1542–1552.

[Ester et al., 1996] Ester, M., Kriegel, H.-P., Sander, J., and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231.

[Fraley and Raftery, 1998] Fraley, C. and Raftery, A. E. (1998). How many clusters? which clustering method? answers via model-based cluster analysis. *The Computer Journal*, 41(8):578–588.

[Fu, 2011] Fu, T.-c. (2011). A review on time series data mining. *Engineering Applications of Artificial Intelligence*, 24(1):164–181.

[Gaffney, 2004] Gaffney, S. J. (2004). *Probabilistic curve-aligned clustering and prediction with regression mixture models*. PhD thesis, Citeseer.

[Gaffney and Smyth, 2004] Gaffney, S. J. and Smyth, P. (2004). Joint probabilistic curve clustering and alignment. In *Advances in neural information processing systems*, pages 473–480.

[Gallagher et al., 2013] Gallagher, T., Bjorness, T., Greene, R., You, Y.-J., and Avery, L. (2013). The geometry of locomotive behavioral states in *C. elegans*. *PloS One*, 8(3):e59865.

[Geng et al., 2003] Geng, W., Cosman, P., Baek, J.-H., Berry, C. C., and Schafer, W. R. (2003). Quantitative classification and natural clustering of *Caenorhabditis elegans* behavioral phenotypes. *Genetics*, 165(3):1117–1126.

[Godoy-Herrera and Connolly, 2007] Godoy-Herrera, R. and Connolly, K. (2007). Organization of foraging behavior in larvae of cosmopolitan, widespread, and endemic *Drosophila* species. *Behavior Genetics*, 37(4):595–603.

[Gomez-Marin and Louis, 2013] Gomez-Marin, A. and Louis, M. (2013). Multilevel control of run orientation in *Drosophila* larval chemotaxis. *Frontiers in Behavioral Neuroscience*, 8:38–38.

[Gomez-Marin and Louis, 2014] Gomez-Marin, A. and Louis, M. (2014). Multilevel control of run orientation in *Drosophila* larval chemotaxis. *Frontiers in Behavioral Neuroscience*, 8.

[Gomez-Marin et al., 2014] Gomez-Marin, A., Paton, J. J., Kampff, A. R., Costa, R. M., and Mainen, Z. F. (2014). Big behavioral data: psychology, ethology and the foundations of neuroscience. *Nature Neuroscience*, 17(11):1455–1462.

[Gomez-Marin et al., 2011] Gomez-Marin, A., Stephens, G. J., and Louis, M. (2011). Active sampling and decision making in *Drosophila* chemotaxis. *Nature Communications*, 2:441.

[Green et al., 1983] Green, C., Burnet, B., and Connolly, K. (1983). Organization and patterns of inter-and intraspecific variation in the behaviour of *Drosophila* larvae. *Animal Behaviour*, 31(1):282–291.

[Hartigan and Wong, 1979] Hartigan, J. A. and Wong, M. A. (1979). Algorithm as 136: A k-means clustering algorithm. *Applied Statistics*, pages 100–108.

[Heckscher et al., 2012] Heckscher, E. S., Lockery, S. R., and Doe, C. Q. (2012). Characterization of *Drosophila* larval crawling at the level of organism, segment, and somatic body wall musculature. *The Journal of Neuroscience*, 32(36):12460–12471.

[Huang et al., 2006] Huang, K.-M., Cosman, P., and Schafer, W. R. (2006). Machine vision based detection of omega bends and reversals in *C. elegans*. *Journal of Neuroscience Methods*, 158(2):323–336.

[Jain et al., 1999] Jain, A. K., Murty, M. N., and Flynn, P. J. (1999). Data clustering: a review. *ACM Computing Surveys*, 31(3):264–323.

[Jolliffe, 2002] Jolliffe, I. (2002). *Principal component analysis*. Wiley Online Library.

[Kabra et al., 2013] Kabra, M., Robie, A. A., Rivera-Alba, M., Branson, S., and Branson, K. (2013). Jaaba: interactive machine learning for automatic annotation of animal behavior. *Nature Methods*, 10(1):64–67.

[Kane et al., 2013] Kane, E. A., Gershow, M., Afonso, B., Larderet, I., Klein, M., Carter, A. R., de Bivort, B. L., Sprecher, S. G., and Samuel, A. D. (2013). Sensorimotor structure of *Drosophila* larva phototaxis. *Proceedings of the National Academy of Sciences*, 110(40):E3868–E3877.

[Kass and Raftery, 1995] Kass, R. E. and Raftery, A. E. (1995). Bayes factors. *Journal of the American Statistical Association*, 90(430):773–795.

[Kato et al., 2015] Kato, S., Kaplan, H. S., Schrödel, T., Skora, S., Lindsay, T. H., Yemini, E., Lockery, S., and Zimmer, M. (2015). Global brain dynamics embed the motor command sequence of *Caenorhabditis elegans*. *Cell*, 163(3):656–669.

[Konishi and Kitagawa, 2008] Konishi, S. and Kitagawa, G. (2008). Bayesian information criteria. *Information Criteria and Statistical Modeling*, pages 211–237.

[Krishnan and McLachlan, 1997] Krishnan, T. and McLachlan, G. (1997). The em algorithm and extensions. *Wiley*, 1(997):58–60.

[Laurent et al., 2015] Laurent, P., Soltesz, Z., Nelson, G. M., Chen, C., Arellano-Carbajal, F., Levy, E., and de Bono, M. (2015). Decoding a neural circuit controlling global animal state in *C. elegans*. *eLife*, 4:e04241.

[Lee and Seung, 1999] Lee, D. D. and Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791.

[Li and Lin, 2010] Li, Y. and Lin, J. (2010). Approximate variable-length time series motif discovery using grammar inference. In *Proceedings of the Tenth International Workshop on Multimedia Data Mining*, page 10. ACM.

[Lin et al., 2007] Lin, J., Keogh, E., Wei, L., and Lonardi, S. (2007). Experiencing sax: a novel symbolic representation of time series. *Data Mining and Knowledge Discovery*, 15(2):107–144.

[Liu et al., 2013] Liu, S., Yamada, M., Collier, N., and Sugiyama, M. (2013). Change-point detection in time-series data by relative density-ratio estimation. *Neural Networks*, 43:72–83.

[Lockery, 2011] Lockery, S. R. (2011). The computational worm: spatial orientation and its neuronal basis in *C. elegans*. *Current opinion in neurobiology*, 21(5):782–790.

[Lonardi and Patel, 2002] Lonardi, J. L. E. K. S. and Patel, P. (2002). Finding motifs in time series. In *Proc. of the 2nd Workshop on Temporal Data Mining*, pages 53–68.

[Martin, 2004] Martin, J.-R. (2004). A portrait of locomotor behaviour in *Drosophila* determined by a video-tracking paradigm. *Behavioural Processes*, 67(2):207–219.

[McGovern et al., 2011] McGovern, A., Rosendahl, D. H., Brown, R. A., and Droegemeier, K. K. (2011). Identifying predictive multi-dimensional time series motifs: an application to severe weather prediction. *Data Mining and Knowledge Discovery*, 22(1-2):232–258.

[McLachlan and Krishnan, 1997] McLachlan, G. J. and Krishnan, T. (1997). Wiley series in probability and statistics. *The EM Algorithm and Extensions, Second Edition*, pages 361–369.

[Melnyk et al., 2006] Melnyk, S., Usatenko, O., and Yampol'skii, V. (2006). Memory functions of the additive markov chains: applications to complex dynamic systems. *Physica A: Statistical Mechanics and its Applications*, 361(2):405–415.

[Minnen et al., 2007] Minnen, D., Isbell, C. L., Essa, I., and Starner, T. (2007). Discovering multivariate motifs using subsequence density estimation and greedy mixture learning. In *Proceedings of the National Conference on Artificial Intelligence*, volume 22, page 615. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

[Mohammad and Nishida, 2009] Mohammad, Y. and Nishida, T. (2009). Robust singular spectrum transform. In *Next-Generation Applied Intelligence*, pages 123–132. Springer.

[Mohammad et al., 2012] Mohammad, Y., Ohmoto, Y., and Nishida, T. (2012). G-stex: greedy stem extension for free-length constrained motif discovery. In *Advanced Research in Applied Artificial Intelligence*, pages 417–426. Springer.

[Mueen et al., 2009] Mueen, A., Keogh, E. J., Zhu, Q., Cash, S., and Westover, M. B. (2009). Exact discovery of time series motifs. In *SDM*, pages 473–484. SIAM.

[Müller, 2007] Müller, M. (2007). Dynamic time warping. *Information Retrieval for Music and Motion*, pages 69–84.

[Nevill-Manning and Witten, 1997] Nevill-Manning, C. G. and Witten, I. H. (1997). Identifying hierarchical strcture in sequences: A linear-time algorithm. *Journal of Artificial Intelligence Research (JAIR)*, 7:67–82.

[Nunthanid et al., 2011] Nunthanid, P., Niennattrakul, V., and Ratanamahatana, C. A. (2011). Discovery of variable length time series motif. In *2011 8th International Conference on ECTI*, pages 472–475. IEEE.

[Oates, 2002] Oates, T. (2002). Peruse: An unsupervised algorithm for finding recurring patterns in time series. In *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on*, pages 330–337. IEEE.

[Ohashi et al., 2014] Ohashi, S., Morimoto, T., Suzuki, Y., Miyakawa, H., and Aonishi, T. (2014). A novel behavioral strategy, continuous biased running, during chemotaxis in *Drosophila* larvae. *Neuroscience Letters*, 570:10–15.

[Ohyama et al., 2013] Ohyama, T., Jovanic, T., Denisov, G., Dang, T. C., Hoffmann, D., Kerr, R. A., and Zlatic, M. (2013). High-throughput analysis of stimulus-evoked behaviors in *Drosophila* larva reveals multiple modality-specific escape strategies. *PLoS one*, 8(8):e71706.

[Oliver et al., 1998] Oliver, J. J., Baxter, R. A., and Wallace, C. S. (1998). Minimum message length segmentation. In *Research and Development in Knowledge Discovery and Data Mining*, pages 222–233. Springer.

[Otsu, 1975] Otsu, N. (1975). A threshold selection method from gray-level histograms. *Automatica*, 11(285-296):23–27.

[Pierce-Shimomura et al., 1999] Pierce-Shimomura, J. T., Morse, T. M., and Lockery, S. R. (1999). The fundamental role of pirouettes in *Caenorhabditis elegans* chemotaxis. *The Journal of Neuroscience*, 19(21):9557–9569.

[Powers, 2011] Powers, D. M. (2011). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *Journal of Machine Learning Technologies*, 2(1):37–63.

[Rodriguez and Laio, 2014] Rodriguez, A. and Laio, A. (2014). Clustering by fast search and find of density peaks. *Science*, 344(6191):1492–1496.

[Salvador et al., 2014] Salvador, L. C., Bartumeus, F., Levin, S. A., and Ryu, W. S. (2014). Mechanistic analysis of the search behaviour of *Caenorhabditis elegans*. *Journal of The Royal Society Interface*, 11(92):20131092.

[Schulze et al., 2015] Schulze, A., Gomez-Marin, A., Rajendran, V. G., Lott, G., Musy, M., Ahammad, P., et al. (2015). Dynamical feature extraction at the sensory periphery guides chemotaxis. *Elife*, 4:e06694.

[Schwarz et al., 1978] Schwarz, G. et al. (1978). Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464.

[Shannon, 2001] Shannon, C. E. (2001). A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55.

[Shen et al., 2011] Shen, W. L., Kwon, Y., Adegbola, A. A., Luo, J., Chess, A., and Montell, C. (2011). Function of rhodopsin in temperature discrimination in *Drosophila*. *Science*, 331(6022):1333–1336.

[Stephens et al., 2008] Stephens, G. J., Johnson-Kerner, B., Bialek, W., and Ryu, W. S. (2008). Dimensionality and dynamics in the behavior of *C. elegans*. *PLoS computational biology*, 4(4):e1000028.

[Szigeti et al., 2015] Szigeti, B., Deogade, A., and Webb, B. (2015). Searching for motifs in the behaviour of larval *Drosophila melanogaster* and *Caenorhabditis elegans* reveals continuity between behavioural states. *Journal of The Royal Society Interface*, 12(113):20150899.

[Todd et al., 2016] Todd, J. G., Kain, J. S., and de Bivort, B. (2016). Systematic exploration of unsupervised methods for mapping behavior. *bioRxiv*, page 051300.

[Treutwein, 1995] Treutwein, B. (1995). Adaptive psychophysical procedures. *Vision Research*, 35(17):2503–2522.

[van der Maaten et al., 2009] van der Maaten, L. J., Postma, E. O., and van den Herik, H. J. (2009). Dimensionality reduction: A comparative review. *Journal of Machine Learning Research*, 10(1-41):66–71.

[Varshney et al., 2011] Varshney, L. R., Chen, B. L., Paniagua, E., Hall, D. H., Chklovskii, D. B., et al. (2011). Structural properties of the *Caenorhabditis elegans* neuronal network. *PLoS Computational Biology*, 7(2):e1001066.

[Verleysen and François, 2005] Verleysen, M. and François, D. (2005). The curse of dimensionality in data mining and time series prediction. In *International Work-Conference on Artificial Neural Networks*, pages 758–770. Springer.

[Vogelstein et al., 2014] Vogelstein, J. T., Park, Y., Ohyama, T., Kerr, R. A., Truman, J. W., Priebe, C. E., and Zlatic, M. (2014). Discovery of brainwide neural-behavioral maps via multiscale unsupervised structure learning. *Science*, 344(6182):386–392.

[Wystrach et al., 2016] Wystrach, A., Lagogiannis, K., and Webb, B. (2016). Continuous lateral oscillations as a core mechanism for taxis in drosophila larvae. *eLife*, 5:e15504.

[Yemini et al., 2013] Yemini, E., Jucikas, T., Grundy, L. J., Brown, A. E., and Schafer, W. R. (2013). A database of *Caenorhabditis elegans* behavioral phenotypes. *Nature Methods*, 10(9):877–879.