UNIVERSITY OF CALIFORNIA,

IRVINE

Probabilistic Curve-Aligned Clustering and Prediction with Regression Mixture Models

DISSERTATION

submitted in partial satisfaction of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

in Information and Computer Science

by

Scott John Gaffney

Dissertation Committee:

Professor Padhraic Smyth, Chair

Professor Michael J. Pazzani

Professor Pierre Baldi

2004

The dissertation of Scott John Gaffney
is approved and is acceptable in quality
and form for publication on microfilm:

Committee Chair

University of California, Irvine

2004

To Andy Selden

the beauty of mathematics

is the reflection upon which we perceive

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

Let me start off by writing that I would never have been able to finish my Ph.D. without the many years of support from my advisor Padhraic Smyth. Even more so, I would like to thank my Mom and Dad for providing me with a lifetime's worth of support and encouragement, and the genuine belief in my pursuit of knowledge. I would also like to thank Mike Pazzani and Pierre Baldi for being on my committee. I would especially like to thank Mike for participating in the many years of my education at UCI.

I would like to recognize the important part that all of my closest friends have played over the years leading up to this moment. Although I have the unfortunate position of not being able to list them all here as they deserve, I would like to thank my friends from the neighborhood block—James Walker, Dave Salladay, and Jim Hastie—as well as from across the field—Darryl Faulstick, Darryl Oliver, Loren Gameros, and Larry Salcido. I would also like to thank Donovan Web for keeping my sanity intact all these years with our adventurous travels over and across US 395, I-5, SR-99, and the I-80. Special thanks goes to Rebecca Dodge for being there for me over these last few years. The members of my affiliated research group also deserve much thanks, in particular Ge Xianping and Igor Cadez were excellent sources of ideas and criticism.

Andy, to whom this dissertation is dedicated, and Rory Lentz have always been an inspiration to me. They were the cause that propelled me down this long path. Mike Schreiner also played a significant part in my education. I would like to mention my music partner Mike Rennie, who was always available for a good jam session, and I thank John Oliver for the many excellent discussions having to do with every topic under the sun. I would also like to thank Scott Rabon, my business partner, colleague, and most importantly, my friend. Finally, I would like to thank my sisters Chris, Cee, and Sharon, and my brothers John and Jamey for their unending support and enthusiasm over the years.

The cyclone datasets analyzed in Chapters 9 and 10 were provided by Andy Robertson and Suzana Carmago of the International Research Institute for Climate Prediction at Columbia University. I would like to thank Andy Robertson in particular for his many years of valued guidance with respect to the cyclone clustering applications presented in this dissertation. My research was supported by funding from the Institute for Scientific Computer Research at Lawrence Livermore National Laboratory.

CURRICULUM VITAE

Education

2001–2004 **Ph.D. Information and Computer Science**
University of California, Irvine

1997–2001 **M.S. Information and Computer Science**
University of California, Irvine

1994–1997 **B.S. Information and Computer Science**
University of California, Irvine

1994–1997 **B.A. Economics**
University of California, Irvine

Employment

2000–2001 **Chief Technology Officer**
LoudEnergy.com

1999–1999 **Research Intern**
AT&T Labs - Research

1996–1997 **Research Programmer**
University of California, Irvine

Awards

1997 Outstanding Research in Information and Computer Science (UCI)
1997 Cum Laude (Computer Science)
1997 Cum Laude (Economics)
1997 Phi Beta Kappa

ABSTRACT OF THE DISSERTATION

Probabilistic Curve-Aligned Clustering and Prediction
with Regression Mixture Models

By

Scott John Gaffney

Doctor of Philosophy in Information and Computer Science

University of California, Irvine, 2004

Professor Padhraic Smyth, Chair

Clustering and prediction of sets of curves is an important problem in many areas of science and engineering. Most clustering algorithms operate on fixed-dimensional feature vectors, and as a result, curve analysis is often forced into this unnatural paradigm. Perhaps more importantly, curves tend to be misaligned from each other in a continuous manner, either in space (across the measurements) or in time. However, the notion of time within a feature-vector is very rigid corresponding only to the discrete dimensional setup of the space itself.

In contrast to this, we develop a probabilistic framework that allows for the joint clustering and continuous alignment of sets of curves in *curve space*. Our proposed methodology integrates new probabilistic alignment models with model-based curve clustering algorithms. The probabilistic approach allows for the derivation of consistent EM-type learning algorithms for the joint clustering-alignment problem. Both simulated and real-world datasets are used for detailed experimentation, with two extensive applications to the clustering of cyclone trajectories presented.

Chapter 1

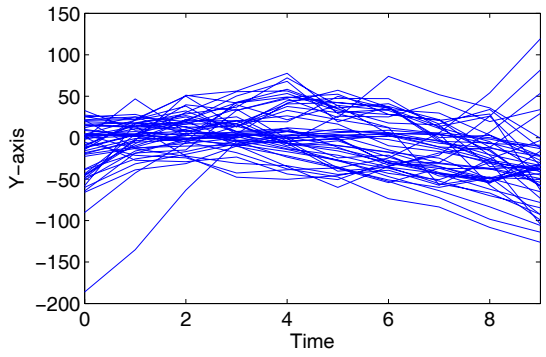
Introduction

This dissertation is concerned with the central hypothesis that clustering and alignment should not be carried out in isolation, but instead the symbiotic relationship between a clustering and an alignment can be exploited to increase the predictive modelling ability of each method in concert. We introduce a novel methodology for the clustering and prediction of sets of smoothly varying curves while jointly allowing for the learning of sets of continuous curve transformations.

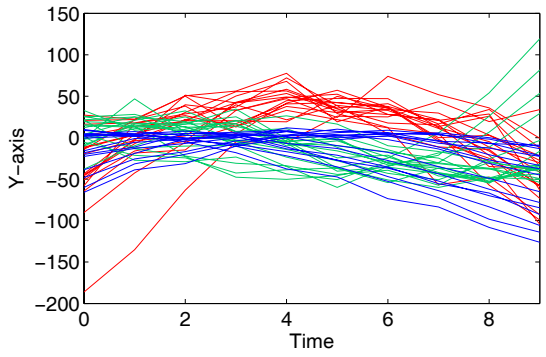
An isolated strategy that tackles the clustering and alignment problems in a two-step sequential manner is sub-optimal. For example, suppose a set of curve data is first preprocessed to effect an alignment and then used for subsequent clustering. In the presence of distinct cluster-specific alignment behavior, the initial alignment will be poor (see example below). The resulting clusters will not describe the true underlying group behavior since the preprocessing has incorrectly mixed-up the data. The opposite strategy of performing an initial clustering without regard for cluster-specific alignments, followed by a within-cluster alignment procedure, is also a suboptimal approach since the initial clustering may be misled by variation in the data due to misalignment.

An example of these effects with simulated data is shown in Figures 1.1 and 1.2. Curves were sampled (with additive noise) from three underlying polynomials at randomly translated points in time. The resulting curves are shown in Figure 1.1(a) with unknown class labels and alignments. The same curves are shown in Figure 1.1(b) with known class labels, and in Figure 1.1(c) with known labels and alignments. Application of the joint clustering-alignment methodology introduced in this thesis to the simulated data in Figure 1.1(a) results in the clusters shown in Figure 1.2(a). The recovery of both the class labels and the alignments is accurate. The plot in Figure 1.2(b) shows the clustering obtained by using a two-step procedure of first aligning the data and then clustering. Neither the class labels nor the alignments are accurately recovered with this procedure.

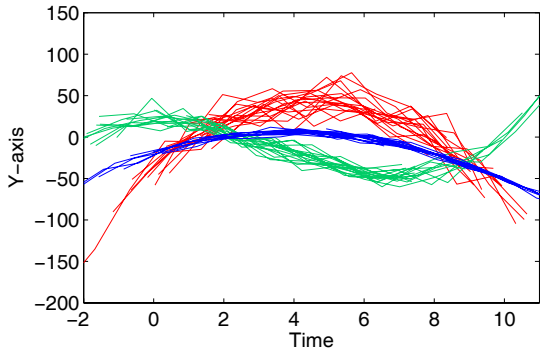
Our approach in solving this complex clustering and alignment problem is to formulate models for both the clustering and alignment sub-problems and integrate them into a unified probabilistic framework that allows for the derivation of consistent learning algorithms. For the alignment sub-problem, we introduce a novel curve alignment procedure employing model priors over the set of possible alignments and derive EM learning algorithms that formalize the so-called *Procrustes* approach for curve data. The Procrustes approach can be recognized as an iterative procedure that aligns a set of curve data to a current mean (or other target) which itself is then updated based on the current set of alignments (Ramsay & Silverman, 1997; Silverman, 1995). In this way, the iterations between the E- and M-steps in our EM alignment algorithms suggest this Procrustes behavior. These alignment models are then integrated into a finite mixture model setting in which the clustering is carried out. We make use of both polynomial and spline regression mixture models to complete the joint clustering-alignment framework.



(a) Simulated data

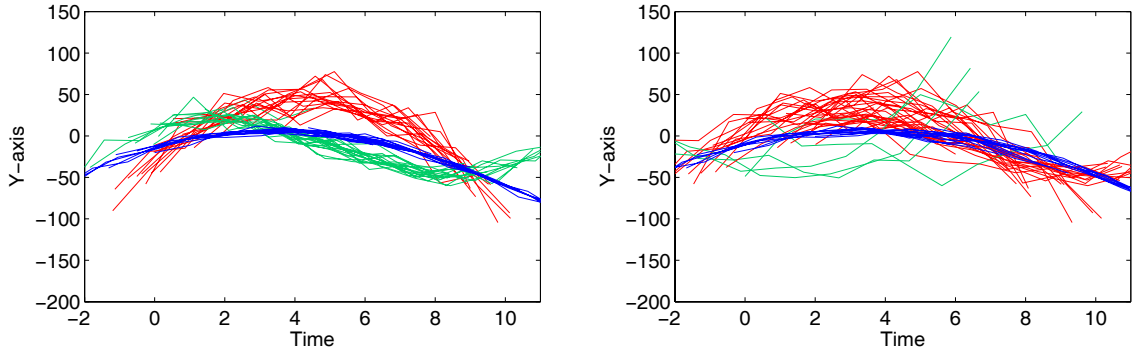


(b) Data with labels



(c) Data with labels and alignments

Figure 1.1: Simulated curve data with unknown clusters and alignments: (a) data as presented to clustering algorithm, (b) same data with known cluster labels, (c) same data with known cluster labels and alignments.



(a) Joint clustering-alignment

(b) Sequential alignment and clustering

Figure 1.2: Learned clusterings with joint and sequential approach: (a) clusters and alignments jointly learned using the clustering models introduced in this thesis, (b) clusters and alignments learned by first aligning and then clustering the data.

In the remainder of this chapter, we provide a brief introduction and motivation for our proposed methodology, followed by a chapter outline that emphasizes the main contributions of this work. The introduction is closed with a section defining the notation used throughout this document.

1.1 Motivation for curve clustering

Clustering is typically used as a tool for understanding and exploring large data sets. Curve clustering as a methodology can be seen to focus on specific types of data sets and those algorithms that are tailored to operate on *curves* as a unit. Traditionally, clustering algorithms have operated on points or on feature vectors of fixed-dimensional size. In contrast, however, curves commonly consist of a variable number of measurements, observed over measurement intervals of varying size, with any number of missing observations. Principal among the contrasts with feature vectors is that curves contain smoothness information which constrain the way in which observations vary from one measurement to the next. Clustering methods

like K-means (Hartigan & Wong, 1978) or Gaussian mixtures (Banfield & Raftery, 1993), for example, proceed without regard to this. This loss of potentially valuable smoothness information can lead to reduced clustering performance.

Curve-type datasets are increasingly available due in part to large-scale data collection in the scientific community. For example, Figure 1.3 shows a set of trajectory data from the atmospheric sciences. The trajectories represent of a number of extra-tropical cyclones that were tracked over the North Atlantic in the winter months (November to April) from 1980 to 1995 (Gaffney et al., 2001). The cyclone trajectories are plotted as tracks on a map of the North Atlantic at the corresponding latitude-longitude positions of the center of the cyclones. The circles indicate initial starting positions for each of the cyclone tracks. The curve data here is multidimensional with respect to time (i.e., there is a two dimensional lat-lon observation vector at each time point). The curves do not have equal lengths since cyclones have variable length lifetimes.

Extra-tropical cyclones are the cause of significant damage in the Northern hemisphere (Schubert et al., 1998), and their genesis, evolution, and links to large-scale atmospheric effects are not well understood (Murray & Simmonds, 1991). Clustering in this context provides a useful tool for exploring this data set.

Figure 1.4 depicts another set of trajectories tracking the center of a person's hand as it moves through a short video scene (Gaffney & Smyth, 1999). The x-axis is time while the y-axis is the vertical location in pixel coordinates of the centroid of a person's moving hand (relative to a fixed coordinate frame). The solid and dotted lines represent two different motions across the scene (a left-to-right movement, and a right-to-left movement). A clustering algorithm would attempt to recover the groups of hand movements represented by the curves.

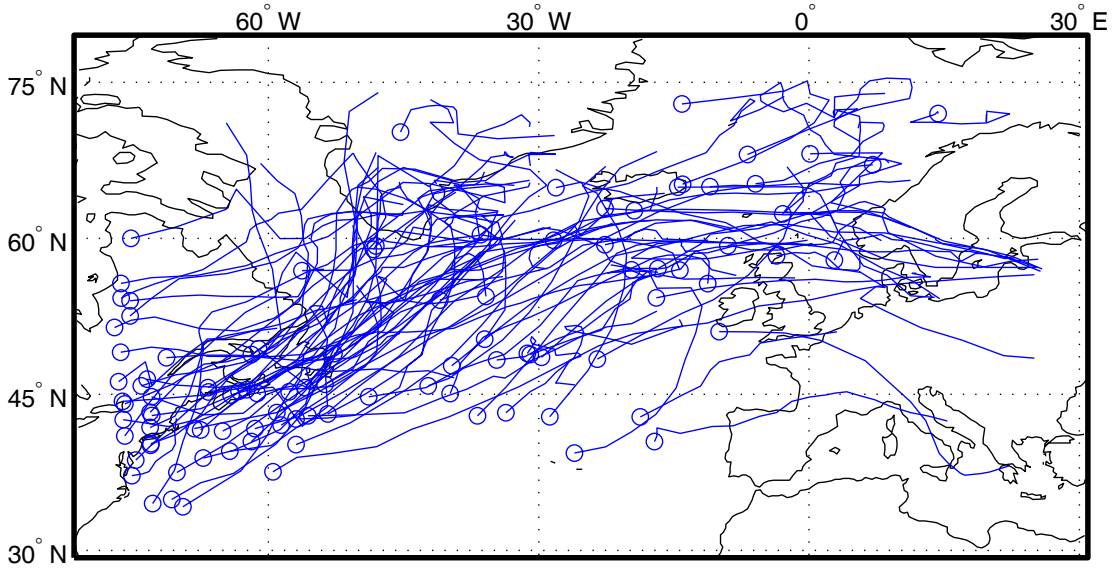


Figure 1.3: Cyclone trajectories tracked over the North Atlantic.

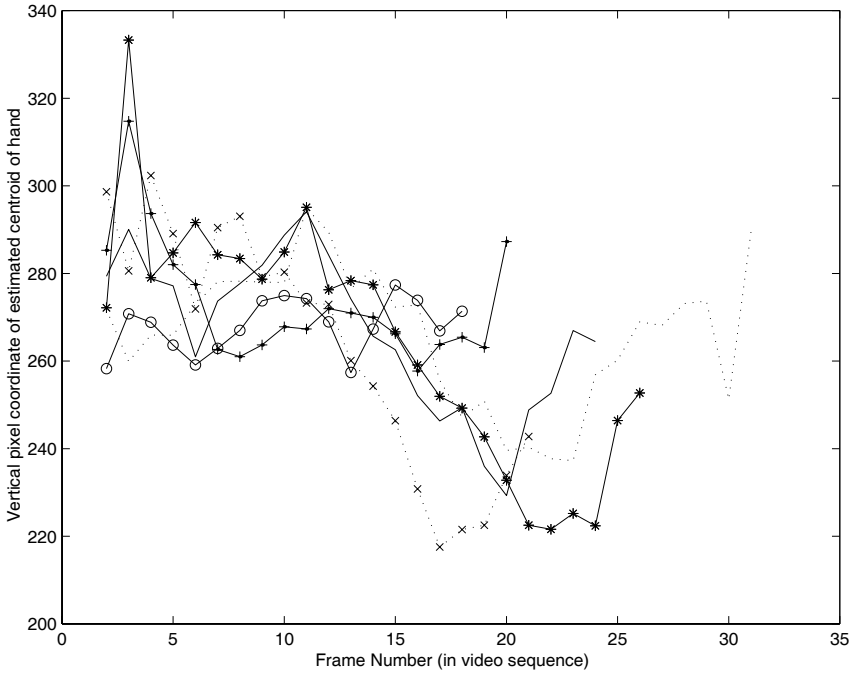


Figure 1.4: Trajectories of the estimated vertical position of a moving hand as a function of time, estimated from 6 different video sequences.

The dynamics of each hand movement differ considerably even for identical motions resulting in both unequal length curves and incorrectly aligned data. For example, a simple left-to-right movement can either be straight, slightly curved, begin with a hiccup, move up-and-down, move at different speeds, etc. Furthermore, the clustering algorithm should not only find cluster groups but it should also return smooth models that can be used to describe the underlying motions made by the hand movements.

Figure 1.5 gives a final example that more directly highlights the alignment problem in particular. The figure plots an estimated velocity curve for each of 39 boys whose heights were measured at 29 observation times over the ages of 1 to 18 (Ramsay & Silverman, 1997). Due to similarities in human growth development, the curves exhibit similar shape but are significantly misaligned due to differences in individual growth dynamics. Clustering and prediction in this situation can be difficult. A further complication is that the original measurement intervals are unequal for different subjects, thus making the analysis of the actual data in a vector space tricky. A good curve-based approach should be able to address all of these concerns and problems.

1.2 Outline of dissertation

The remainder of this thesis is concerned with the definition, learning, and application of our joint alignment-clustering models. In brief overview, this thesis sets out to do the following: (a) explain the inadequacy of standard clustering techniques for curves, (b) define and extend model-based clustering algorithms for curves, (c) introduce models for curve alignment in measurement space, (d) introduce models for curve alignment in time, (e) integrate alignment and clustering in a unified frame-

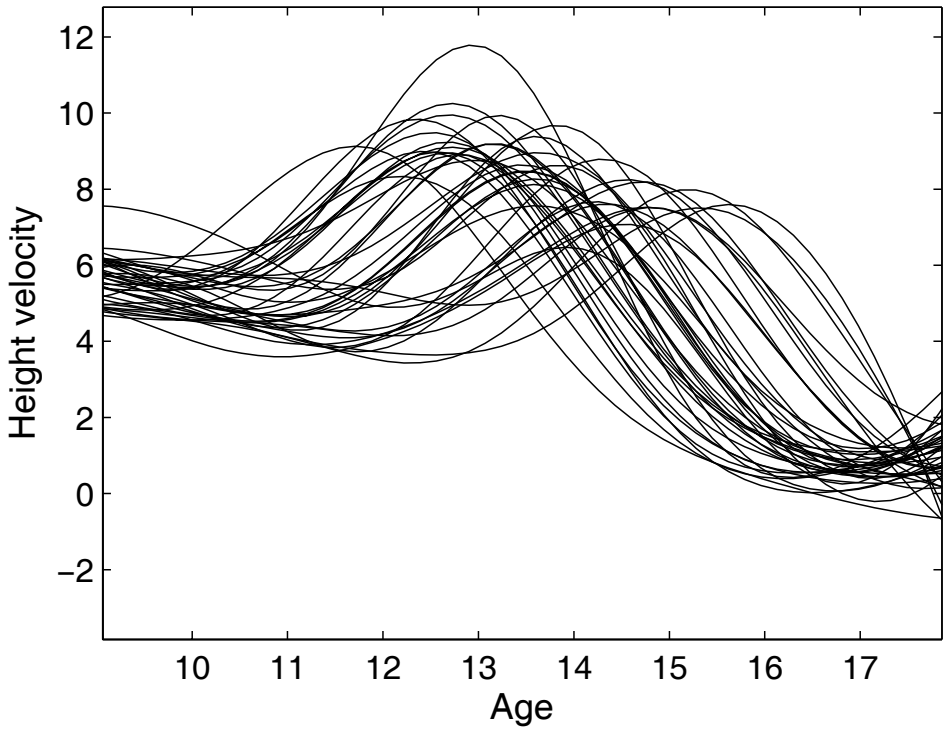


Figure 1.5: Estimated velocity of height measurements for 39 boys measured in cm/yr.

work, and (f) apply the joint methodology to real-world problems. The primary contribution of this thesis is the development of a novel probabilistic framework for the clustering and prediction of sets of smoothly varying curves while simultaneously allowing for the learning of sets of continuous curve transformations. A chapter outline with a summary of each chapter's main contributions follows.

In Chapter 2, a brief overview of existing clustering methodology (including vector- and non-vector-based) and how it relates to the curve clustering work in this thesis is given. Emphasis is placed on the range of applicability of each method for curve clustering in general. The main contribution of this chapter is in clearly setting forth the reasons why many standard clustering methods are inadequate for curve clustering.

In Chapter 3, the foundation for the clustering models and algorithms that are introduced in this thesis is given. We review standard model-based clustering with finite mixture distributions and show how curve clustering can be realized through the use of conditional mixture distributions. These distributions take the form of mixtures of regression models in which cluster-specific mean curves are modelled with regression functions (e.g., polynomial regression). At the end of this chapter, results are reported from simulated data experiments that show the benefits of using curve-based models for the clustering of curve data. The main contributions of this chapter are the extension of polynomial regression mixtures to multi-dimensional curve clustering (employing curve-level memberships), the introduction of mixtures of kernel regression models, and extensions to spline regression models.

In Chapter 4, we introduce curve clustering with mixtures of MAP (maximum a posteriori) random effects models. These models allow for certain heterogeneity within clusters through the use of prior distributions on cluster parameters in the form of mixture densities. Random effects regression mixtures (RERM) are a mix between regression mixture models and linear random effects models (Laird & Ware, 1982). The formulation of this problem as a hierarchical model allows for the derivation of efficient EM learning algorithms. Experimental results with simulated data are reported that show the increased performance of RERMs as compared to standard mixtures of regressions, K-means, and Gaussian mixtures. The main contribution of this chapter is in the introduction of the hierarchical MAP-based RERM and the associated experimental results.

In Chapter 5, we turn our attention to the curve alignment sub-problem. We introduce the use of probabilistic curve modelling techniques as the basis for a set of novel alignment models. These models allow for the alignment of curve data in measurement space. By "alignment in space" we mean as allowing for transforma-

tions on the curve measurements themselves. The main contribution of this chapter is in the formulation of the curve alignment problem in probabilistic terms. The formulation unifies the specification, learning, and prediction problems in a single, self-contained framework. The derived EM algorithm generalizes the classic Procrustes approach for curve alignment, and demonstrates the use of the Mahalanobis distance as a natural Procrustes distance metric. Experimental results with real and simulated data are reported that demonstrate the usefulness of the alignment models.

In Chapter 6, we address the more complex problem of curve alignment in time. Building on the foundation of the previous chapter, we define our time-alignment models and derive their associated learning algorithms. The main contribution is the formulation of the time-alignment problem using probabilistic curve modelling techniques and the exact calculation of the so-called Q-function for the case of polynomial regression models. Experimental results are presented at the end of this chapter with a real gene expression dataset and with simulated data. The results show the effectiveness of the probabilistic formulation.

In Chapter 7, we unify the individual space- and time-alignment models into a single joint framework that allows for transformations in both measurement space and in time. The derivation for the joint alignment model borrows much from the individual derivations of the component alignment models. Thus, this chapter is brief in its presentation. We also use this chapter to extend the alignment methodology to the case of multidimensional curves. The main contribution of this chapter is the introduction of the joint space- and time-alignment model.

In Chapter 8, we discuss the integration of the curve alignment models with the clustering algorithms of Chapter 3. This unification results in a model-based, joint clustering-alignment methodology for curve data. We use this chapter to define ap-

propriate out-of-sample test measures for model selection based on test log-likelihood and prediction SSE (sum of squared-error) scores. Extensive simulated-data experiments are reported. These experiments compare the different clustering-alignment models against each other and to other clustering algorithms. The main contribution is in the unification of the clustering and alignment problem, and the reporting of the experimental results.

In Chapter 9, we introduce a new methodology for the clustering of extra-tropical cyclone trajectories. The application requires initial detection and tracking of cyclone trajectories from raw MSLP (mean sea-level pressure) data maps. These data maps were generated by a computational climate model known as a general circulation model (GCM). We describe the developed procedure for the detection and tracking of cyclones from GCM data that was employed. The results of applying our joint clustering-alignment methodology to the resulting set of tracked cyclone trajectories are analyzed in detail. The main contribution of this chapter is in the application of our clustering methodology to the tracked cyclone dataset.

In Chapter 10 we present an application of our clustering methodology to an "observed" tropical cyclone dataset. Unlike the cyclone dataset mentioned above, this dataset consists of trajectories from actual cyclones observed in the tropical North Pacific. The resulting clusters are analyzed and the temporal behavior of these clusters over time is investigated. Finally, in Chapter 11, this thesis is closed with concluding remarks.

1.3 Notation

In this section, the general notational framework that is used throughout this thesis is briefly described. In later chapters, more specific notation is defined at the point in which it is first introduced.

Vectors and matrices

In general, a vector is represented in upright bolding as $\mathbf{x}$ or $\mathbf{y}$. A vector of zeros of arbitrary length is denoted as $\mathbf{0}$, while a vector of ones is denoted as $\mathbf{1}$. A prime is used to denote transpose so that $\mathbf{x}'$ represents a row-vector. Matrices are represented in capitalized upright bolding as $\mathbf{X}$ or $\mathbf{V}$. The identity matrix is denoted as $\mathbf{I}$, and a matrix of ones is denoted with the special notation $\mathbb{1}$. Note that $\mathbb{1} = \mathbf{1}\mathbf{1}'$.

Sets of curves

In this thesis, curves are represented as variable-length vectors. Thus, $\mathbf{y}_i$ is a curve that consists of a sequence of n_i observations or measurements. The j-th measurement of $\mathbf{y}_i$ is denoted as y_{ij} and is usually taken to be univariate (unless otherwise stated). The associated covariate of $\mathbf{y}_i$ is written as $\mathbf{x}_i$ in the same manner. $\mathbf{x}_i$ is usually represented as time so that x_{ij} gives the time at which y_{ij} was observed.

We define italic Y as the set $\{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$ of n curves. In a similar manner, we define italic X as the set $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$. We take the notation $\{\mathbf{y}_i\}$ to mean the entire set of all $\mathbf{y}_i$. So in particular $Y = \{\mathbf{y}_i\} = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$.

Probability distributions

We represent an unspecified probability density as $p(\mathbf{y}_i|\theta)$. The Gaussian distribution with mean μ and variance σ^2 is specifically denoted as $\mathcal{N}(\mu, \sigma^2)$. An arbitrary

Gaussian random variable x is denoted as $x \sim \mathcal{N}(\mu, \sigma^2)$, or as $p(x) = \mathcal{N}(x|\mu, \sigma^2)$. As a rule, all densities in this thesis are implicitly conditioned on an appropriate parameter vector. For example, if $p(\mathbf{y}_i) = \mathcal{N}(\mathbf{y}_i|\mathbf{X}_i\boldsymbol{\beta}, \Sigma)$, then the left-hand side is implicitly understood to be conditioned on the parameters $\theta = \{\boldsymbol{\beta}, \Sigma\}$ and on the non-random matrix $\mathbf{X}_i$.

Regression models

The standard p-th order regression model for curve $\mathbf{y}_i$ is written as

$$\mathbf{y}_i = \mathbf{X}_i\boldsymbol{\beta} + \boldsymbol{\epsilon}_i, \quad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}), \tag{1.1}$$

where $\boldsymbol{\beta}$ is a $(p+1) \times 1$ vector of regression coefficients, $\boldsymbol{\epsilon}_i$ is an $n_i \times 1$ noise vector, and the matrix $\mathbf{X}_i$ is the usual $n_i \times (p+1)$ Vandermonde regression matrix. We write the Vandermonde matrix for $\mathbf{x}_i$ as $\mathbf{X}_i$ and associate it with the expanded form

$$\mathbf{X}_i = \begin{bmatrix} 1 & x_{i1} & x_{i1}^2 & \cdots & x_{i1}^p \\ 1 & x_{i2} & x_{i2}^2 & \cdots & x_{i2}^p \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{in_i} & x_{in_i}^2 & \cdots & x_{in_i}^p \end{bmatrix}.$$

Chapter 2

Overview of Clustering

2.1 Introduction

Clustering is a generally useful tool for many kinds of applications. It can be applied as a preprocessing step in supervised classification problems to find prototype examples in training data (e.g., Duta et al., 1999). It can be used for image segmentation and representation by modelling images with mixture distributions (e.g.,Jepson & Black, 1996). It can also be used to directly learn density models of input data which themselves can then be used for classification and prediction of future data (e.g., Duda & Hart, 1973). More generally it can be used as an exploratory technique to summarize or describe complex data in useful ways. This chapter gives a brief overview of clustering in general and broadly classifies the different methods into two groups: vector-based, and pairwise distance-based. Special emphasis is placed throughout on the handling of curve data in each case.

2.2 Standard clustering techniques

There are many clustering methods available to the data analyst. However, many of these methods fail to be useful or practical when presented with curve data. Some require arbitrary preprocessing of the data, others are computationally prohibitive, while others fail to take advantage of the complete information available in a curve data set. In addition, prediction either does not make sense within the framework or is limited to some region or to points that exactly correspond to the fixed experimental design intervals. Finally, all of these methods ignore the dependence of the curve measurements on the dependent variable (usually considered as time); the conditional model is ignored.

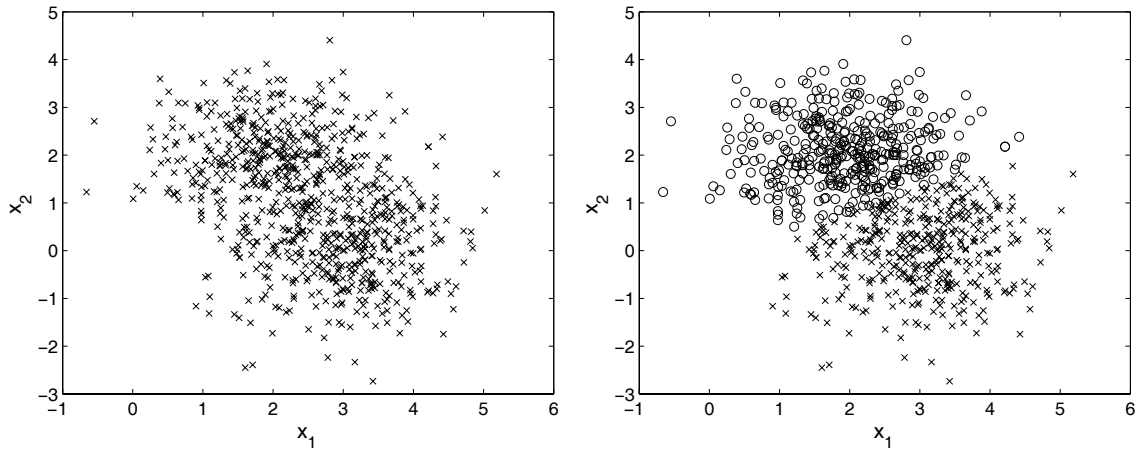
2.2.1 Vector-based methods

Many common clustering algorithms in wide use today fall into the vector clustering category. These standard multivariate clustering techniques require fixed-dimensional vector data. For example, K-means (Hartigan & Wong, 1978) is a classic example of a non-probabilistic vector clustering method that uses iterative relocation in an attempt to minimize within-cluster variance. The error term

$$E = \sum_{k} \sum_{i} z_{ik} \left(\mathbf{y}_i - \boldsymbol{\mu}_k\right)^2 \tag{2.1}$$

formally gives the criterion that K-means attempts to minimize by reassigning the points $\mathbf{y}_i$ among the K clusters. The z_{ik} are indicator variables with a value of 1 when $\mathbf{y}_i$ is assigned to cluster k, and μ_k is the mean value of cluster k.

Figure 2.1 shows an example of K-means applied to a simulated multivariate data set. The figure shows the unlabelled data on the left in (a), while the returned



(a) Unlabelled data

(b) K-means clustering

Figure 2.1: K-means clustering example.

clustering from K-means is given on the right in (b). The formation of linear separation planes in the vector space can be seen in the figure. These planes partition the vector space into distinct contiguous cluster regions such that every point falling in a particular region is assigned to the same cluster.

Gaussian mixtures is a natural extension of K-means to the probabilistic domain. There are two important differences that distinguish Gaussian mixtures from K-means. First, the error term in (2.1) is replaced with a generalized Mahalanobis distance based on the exponent of the normal distribution; and second, points are assigned to clusters with a probability of membership instead of with a binary decision. Gaussian mixture models are an example of a more general model-based clustering methodology (Banfield & Raftery, 1993) from which we derive our curve clustering algorithms in Chapter 3.

Gaussian mixtures can be defined in a mathematical sense by representing the probability density function of $\mathbf{y}_i$ as a multimodal mixture density. The form of a

mixture density is

$$p(\mathbf{y}_i|\Theta) = \sum_k \alpha_k \, p_k(\mathbf{y}_i|\theta_k), \tag{2.2}$$

with K component density functions p_k and K non-negative mixture weights α_k that sum to one. Each of the component densities can be seen as describing specific cluster behavior. A resulting log-likelihood function can be defined as the sum over all n points of the log density in (2.2):

$$L(\Theta|\mathbf{y}_1, \ldots, \mathbf{y}_n) = \sum_i \log \sum_k \alpha_k \, p_k(\mathbf{y}_i|\theta_k). \tag{2.3}$$

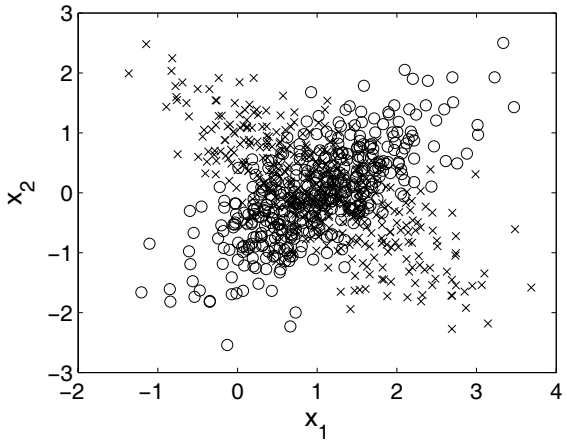
The clustering problem then becomes maximizing the log-likelihood function over the parameters Θ. The actual cluster labels are determined by assigning points to the class of maximum class membership based on resulting membership probabilities.

Figure 2.2 demonstrates the increased flexibility that the mixture framework enjoys over K-means. Figure 2.2(a) is a plot of two-dimensional simulated data generated by a two cluster model. The class labels are shown with symbols in the figure. Figure 2.2(b) is a representation of the clustering output from Gaussian mixtures on the simulated data, while Figure 2.2(c) shows the K-means clustering.

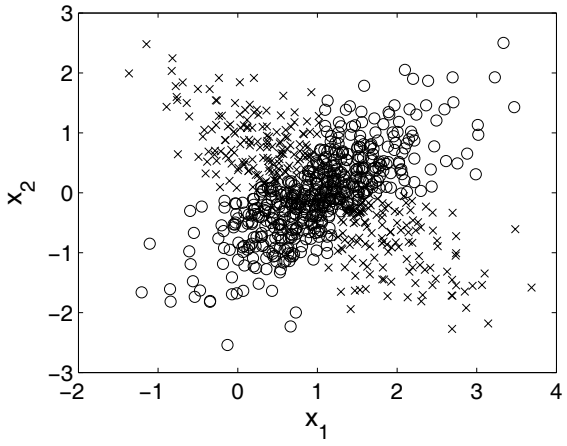
A well-known limitation of K-means is that it cannot model covariance in a data set. The effect of this can be seen by the incorrect grouping of the crossing clusters in Figure 2.2(c). Since K-means is limited to choosing linearly bounded, contiguous cluster regions, it can do no better than this. On the other hand, Gaussian mixtures has no trouble finding the covariant clusters.

Applications to curves

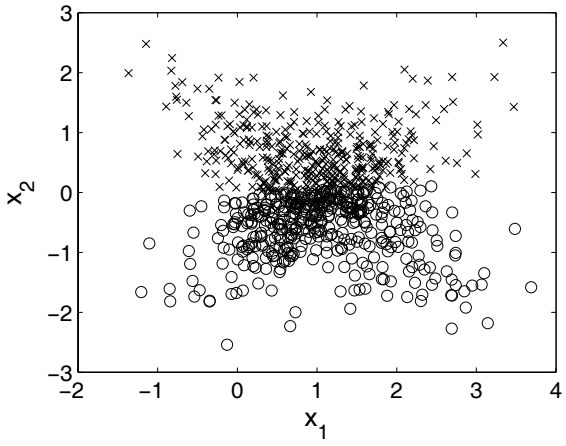
In order to use these vector-based techniques with curve data we must preprocess the data in such a way that reduces the curves to fixed-dimensional vectors. For



(a) Data w/ true labels



(b) Gaussian mixtures clustering



(c) K-means clustering

Figure 2.2: Clustering example with Gaussian mixtures and K-means.

example, in an analysis of cyclone trajectories, Blender et al. (1997) truncate all trajectories under consideration to a fixed length of 3 days worth of (x, y) latitude-longitude pairs (observed at 6-hour intervals). Upon concatenation of the x and y observations, they end up with 24-dimensional feature vectors and use K-means to cluster them. They demonstrate that clusters in this manner can be found; however, cyclones do not naturally conform to such a format and this type of methodology may result in a significant loss of valuable information.

Dougherty et al. (2002) develop an entire clustering toolbox to evaluate the effectiveness of clustering algorithms for gene expression clustering. The toolbox consists of K-means, fuzzy C-means (a fuzzy version of K-means; Duda & Hart, 1973), Kohonen-networks (a neural network motivated algorithm; Kohonen, 1995), and standard hierarchical clustering techniques (see Section 2.2.2). None of these methods handle curve data directly, they all operate on fixed-dimensional feature vectors. Yeung et al. (2001) employ model-based methods for gene expression clustering in which they use Gaussian mixtures to model curve data. Although this is a bit better than the use of K-means as in Dougherty et al. (2002), it still does not address the underlying curve problem directly.

This and other similar work demonstrate the use of vector-based methods for curve data; however, the reduction to fixed-dimensional space may not always be justifiable given its somewhat ad hoc nature in many circumstances. Furthermore, and more importantly, we lose the smoothness or temporal information contained in the sequence of events which is not explicit in vector form. Lastly, there is limited or no principled notion of prediction capability at points within, between, or beyond the fixed-dimensional measurement points. In Chapter 3, we show the applications of K-means and Gaussian mixtures to curve data produce clusters with less predictive power than with the curve clustering methodology that we propose.

2.2.2 Pairwise distance methods

Pairwise distance methods focus on the definition of a distance metric between every pair of points in the data set. With this metric in hand, a similarity (or dissimilarity) matrix $\mathbf{D}$ is defined which organizes the distances between each pair of points in matrix form:

$$\mathbf{D} = \begin{bmatrix} 0 & & & \\ d_{21} & 0 & & \\ d_{31} & d_{32} & 0 & \\ d_{41} & d_{42} & d_{43} & 0 \end{bmatrix}.$$

This matrix can then be used with any number of classical hierarchical clustering techniques (Everitt, 1993). In general, each technique can be described as agglomerative or divisive. Agglomerative techniques start with every point in its own cluster and then repeatedly merge two clusters based on the matrix $\mathbf{D}$ (divisive methods work in the opposite manner). Agglomerative methods differ in how they decide which clusters to merge. In *single linkage* clustering, the distance between two clusters c_1, c_2 is defined as

$$D_{c_1,c_2} = \min_{i \in c_1, j \in c_2} d_{ij}$$

and at each stage the nearest clusters are merged. The resulting clusters from this method tend to be rather elongated or "stringy" in one or a few directions since they tend to pick up points along a line. In *complete-linkage* clustering, we simply replace the min operator with max and proceed in the same manner. Since every point in each cluster contributes to the cluster distance, there is a noticeable reduction in the elongated effect. Still another method, the *minimum-variance* method (or Ward's method) seeks to find minimum variance clusters similar to K-means. However, in this case, merging of sub-clusters is used to achieve this goal as opposed to an

iterative reassignment of points.

Applications to curves

The distance metrics with these methods can be defined in many different ways on any kind of objects, whether they be points, vectors, or curves. Focusing on curve data, once the distances are calculated, the original curve data can be ignored and then any type of pairwise distance-based clustering algorithm can be used.

For example, Butte and Kohane (2000) presented a method that attempted to find functional genomic clusters in RNA expression data. Using a set of $2,467$ genes they constructed 22 different *relevance networks* or clusters by using a technique based on pairwise distance clustering. Their distance metric was based on mutual information between each pair of gene expression profiles (curves). Mutual information is a measure of the reduction in entropy (randomness) of one curve given knowledge of another. Thus, if knowing curve A provides no reduction in the entropy of curve B, then they have exactly zero mutual information. The clustering of genes in this case was carried out in a divisive (as opposed to agglomerative) manner by removal of edges in the fully connected network of genes whose mutual information did not meet a threshold. Their hypothesis was that the resulting clusters were scientifically valid.

An example of similar graph-based method was used by Ben-Dor et al. (1999). They defined their distance matrix in a slightly different manner and then identified a set of graph operations which led to a clustering of the nodes. Other example work followed more along the line of Eisen et al. (1998), in which they analyzed a set of gene expression data resulting from time-course experiments. They represented the curves as vectors and used a Euclidean distance metric for the generation the distance matrix. They pursued gene clusters using a standard agglomerative hierarchical

clustering algorithm on the distance matrix.

While these and other methods present plausible approaches, in practice it can be problematic to define appropriate distance measures for complex problems. Also, computationally, we are immediately saddled with $O(n^2)$ operations on the n curves even before we carry out any clustering. Nor is it obvious whether one can perform prediction with these methods either. Finally, none of these techniques take into account the smoothness information in the curves themselves during the clustering. In the next chapter, we define a set of curve-based clustering techniques and show that they systematically out-perform the methods discussed above by naturally handling each of these concerns.

2.3 Summary

In this chapter we presented an overview of current clustering methods. We set out to demonstrate the need for true curve clustering models by highlighting the deficiencies of each of the particular standard methods. We categorized clustering techniques into two broad categories and provided example applications to curve clustering from each category. The clustering of curves using vector-based or hierarchical methods provides limited capabilities for addressing the following issues:

- Clustering curves of different lengths
- Clustering curves with un-balanced designs (irregular sampling/observation intervals)
- Clustering curves while leveraging smoothness information
- Making predictions between or beyond the measurement observations
- Handling missing observations
- Dealing with multidimensional curves (e.g., three-dimensional measurements for the trajectory of a moving object in space)

- Accounting for sets of misaligned curves during the clustering

In the following chapter, and in the rest of this thesis, we define and introduce curve-based models and methodologies which specifically deal with each of these issues.

Chapter 3

Curve Clustering with Regression Mixtures

3.1 Introduction

In this chapter, we discuss the specific techniques, models, and algorithms that directly address the curve clustering problem. The methods presented here address each of the faults that were pointed out with the standard clustering techniques in Chapter 2 (except for the alignment problem, which is the main topic of this thesis, and is dealt with in the following chapters). The foundation for each of these curve clustering methods rests upon the regression mixtures framework, and more generally, are instances of model-based clustering.

In Section 3.2, a brief introduction to density estimation as it relates to mixture models is given. The relation between these statistical methods and clustering is discussed. In Section 3.3, the first curve clustering model is defined. The polynomial regression mixture model (PRM) employs conditional mixture density estimation to uncover cluster memberships among a set of curves. In Section 3.4, an extension

of the PRM framework to spline regression mixture models (SRM) is discussed. Section 3.5 introduces a novel nonparametric extension of PRM to kernel regression mixture models (KRM).

In Section 3.6, we report the results of simulated data experiments that show the curve-based clustering approaches defined in this chapter systematically out-perform the more common vector-based approaches. Gaussian mixtures is used as a proxy for the vector-based approaches. We show that Gaussian mixtures does not leverage the available smoothness information, does not handle variable-length curve data, and does not deal well with curves measured at different time points or that contain missing observations (which can be seen to be an equivalent situation). Finally, the chapter is concluded with a summary in Section 3.7.

3.2 Clustering by density estimation

Density estimation is a standard probabilistic technique that can be used to summarize a set of data. Nonparametric techniques such as histogram fitting (Silverman, 1986), local polynomial regression (Fan & Gijbels, 1996), and wavelet thresholding (Donoho et al., 1996) can be used when one does not wish to pre-specify the form of the density function.

Parametric techniques, on the other hand, first assume a functional form for the probability density function (PDF), for example, a normal distribution, and then *fit* the model to the data. The fitting process involves setting the values of a small set of distribution parameters (e.g., the mean and variance) so that the resulting density "matches" the distribution of the data. The fitting process is usually carried out using one of a number of methods: the method of moments, Maximum likelihood (ML) estimation, maximum a posteriori (MAP) estimation,

or fully Bayesian techniques. In this thesis, we primarily use the ML and MAP approach

3.2.1 Finite mixture models

We can think of finite mixture models (Everitt & Hand, 1981; Titterington et al., 1985; McLachlan & Basford, 1988) as a semi-parametric form of density estimation in which we assume that sub-regions of the data can be summarized with individual (component) PDFs. The overall density then takes the form of a convex combination of these component density functions. A key feature of the mixture model is its ability to model highly non-Gaussian, multimodal density functions using simpler (e.g., unimodal) component densities.

In the standard setup, we model the d-dimensional vector $\mathbf{y}_i$ by the mixture density

$$p(\mathbf{y}_i|\Theta) = \sum_{k}^{K} \alpha_k \, p_k(\mathbf{y}_i|\theta_k), \tag{3.1}$$

in which α_k is the k-th mixture weight and p_k is the k-th component density with parameter vector θ_k. The mixture weights α_k sum to one and are nonnegative.

The component densities $p_k(\cdot)$ model individual specific sub-regions of density, while the mixture $p(\cdot)$ summarizes all of these sub-regions according to the mixture weights α_k.

Maximum likelihood estimation and EM

The likelihood of a data set $Y = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$ is any function of Θ that is proportional to the probability of the data $p(Y|\Theta)$. The log-likelihood is the log of the likelihood

and takes the form

$$\begin{aligned}\mathcal{L}(\Theta|Y) &= \log p(Y|\Theta) \\ &= \sum_i \log p(\mathbf{y}_i|\Theta) \\ &= \sum_i \log \sum_k \alpha_k \, p_k(\mathbf{y}_i|\theta_k), \end{aligned} \tag{3.2}$$

assuming the $\mathbf{y}_i$ are i.i.d. (independently and identically distributed). ML estimates of the parameter vector Θ correspond to values of Θ that maximize (3.2).

The ML estimates for Θ do not, in general, yield closed-form solutions since the estimates depend non-linearly on each other. However the development of a general, iterative ML procedure called Expectation-Maximization (EM) provides an efficient framework for parameter estimation in the mixture context (Dempster et al., 1977; McLachlan & Krishnan, 1997). EM is an approximate root-finding procedure that is used to seek the root of the likelihood equation—it iteratively searches for a set of parameters $\hat{\Theta}$ that maximize the probability of the observed data. We review the necessary prerequisite EM theory that is needed for the rest of this thesis in Appendix A.

3.2.2 Model-based clustering

The use of mixture models for clustering is sometimes referred to as model-based probabilistic clustering (Fraley & Raftery, 1998, 2002), since a particular functional form for the component densities (such as a Gaussian model) must be assumed. Finite mixture models are widely used for clustering data in a variety of applications (e.g., see McLachlan & Basford, 1988).

In a clustering context, the data $Y = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$ are assumed to have been *generated* by a finite mixture model with K components. Each component density

is associated with a cluster, and each datum $\mathbf{y}_i$ is assumed to have been generated by one and only one cluster.

Given the data Y and having no knowledge of cluster labels, the parameters for the K component cluster models are inferred using, for example, the EM algorithm. Once the parameters of the mixture have been estimated the probability w_{ik} that $\mathbf{x}_i$ was generated by component k can be calculated using Bayes rule:

$$w_{ik} = p(k|\mathbf{y}_i, \hat{\Theta}) \approx \alpha_k \, p_k(\mathbf{y}_i|\hat{\theta}_k).$$

This, in turn, is interpreted as the membership probability of $\mathbf{y}_i$ belonging to cluster k, providing a clustering of the data points (e.g., by assigning each $\mathbf{y}_i$ to the cluster for which it has the largest membership probability).

3.2.3 Model-based curve clustering

A particular advantage of the probabilistic approach is that the component PDFs can be defined on non-vector data. For example, suppose that $\mathbf{y}_i$ is now a sequence of curve measurements of length n_i, observed at the n_i time points in $\mathbf{x}_i$. We can define a cluster-specific conditional probabilistic model $p_k(\mathbf{y}_i|\mathbf{x}_i, \theta_k)$ that relates $\mathbf{y}_i$ to $\mathbf{x}_i$.

Just as in the previous section, the overall density of $\mathbf{y}_i$ (now given $\mathbf{x}_i$) is a mixture of the component PDFs. In other words, the density takes the form

$$p(\mathbf{y}_i|\mathbf{x}_i, \Theta) = \sum_{k}^{K} \alpha_k \, p_k(\mathbf{y}_i|\mathbf{x}_i, \theta_k). \tag{3.3}$$

This conditional mixture density is now defined on curves as objects as opposed to fixed-length vectors. This *curve* density can be used in place of the mixture density

in (3.1) to arrive at a model-based clustering procedure for curves. The generative model for the curve mixture is as follows:

1. Assign the i-th curve to cluster k with probability α_k
2. Generate the *mean* curve for cluster k according to the component density model p_k
3. Define the i-th curve $\mathbf{y}_i$ to be equal to the mean curve plus some randomly generated noise (e.g., add a Gaussian error term)

The foundation of the curve clustering model defined in this way is two part. First, the mixture framework leads to efficient learning algorithms based on EM, and thus, to efficient clustering algorithms. And second, the conditional PDFs directly provide for curve models that handle variable length curves, random measurement intervals, missing observations, and explicit handling of smoothness constraints. All that remains is to define the functional form for the curve PDFs themselves. The remainder of this chapter will describe several functional forms for these curve PDFs which result in various types of curve clustering algorithms.

3.3 Polynomial regression mixtures

In this section, a curve clustering methodology based on polynomial regression mixture models (PRM) is described. PRMs employ polynomial regression models with Gaussian error terms as the component PDFs. The inclusion of these regression models into the model-based curve clustering framework outlined above leads to efficient EM learning algorithms for curve clustering.

In Section 3.3.1, the relevant prior work is discussed. There is a long history of the analysis of curve data using regression models. Often these models were used to describe two or more different behaviors within a single dataset. This work led to methods for the automated discovery of groups of curves described by unique

regressions. In Section 3.3.2, the model definition of PRMs is given. Section 3.3.3 derives the EM learning algorithm for PRMs. We will see that this derivation can be directly used to derive the learning algorithms for the other curve clustering models introduced in this chapter.

3.3.1 Prior work

Regression-based clustering has a relatively long history beginning with work on the simple two-cluster case right up to the general EM methodology for the K-cluster case and beyond. Most of the prior work focused on the univariate, non-curve case. In other words, the datasets consisted of individual univariate observations that were assumed to have been sampled from a regression curve.

One of the earliest works was that of Quandt (1972) who defined a two-component mixture likelihood for so-called *switching regressions*. The methodology demonstrated the ability to find underlying group behavior by maximizing the likelihood using a conjugate gradient algorithm. Later, Quandt and Ramsey (1978) developed a procedure using the method of moments to estimate the mixture parameters for switching regressions.

Hosmer (1974) also defined a two component mixture likelihood containing regression components but used maximum likelihood to estimate the mixture parameters in an iterative process. Essentially, he developed an EM algorithm for mixtures of regressions of two clusters. His paper also contains the first reference to the name *mixtures of regressions* that we have so far found.

Späth (1979) developed an algorithm called *clusterwise linear regression* that estimates the parameters for several different regression coefficient vectors simultaneously. Although there is no notion of a probabilistic model, the data is assumed to come from K groups of behavior, each explained by a different regression function.

The methodology is similar to K-means in that curves are iteratively relocated to minimize a squared-error criterion.

DeSarbo and Cron (1988) developed the modern EM-based procedure for mixtures of linear regressions with any number of clusters. However, as with the previous work above, they focus exclusively on the univariate, non-curve case. Jones and McLachlan (1992) extend this work to multivariate data. They develop a regression mixture model based on "three-mode" data, which is essentially a type of multivariate feature-vector data. It can be seen to be equivalent to a set of multidimensional curves of uniform length with uniform observation intervals and no missing observations.

There has since been the extension of the conditional mixtures framework to many types of regression models. For example, Lwin and Martin (1989) integrate binomial probit models, Wedel and DeSarbo (1993) integrate binomial logit models, Kamakura (1991) look at multinomial probit models, and Wang et al. (1998) develop Poisson regression mixture models.

Our work on linear (or polynomial) regression mixtures for curve data (Gaffney & Smyth, 1999) extended the work of DeSarbo and Cron (1988) with *clusterwise linear regression* by explicitly incorporating the notion of curves and curve membership into an overall general framework. This approach was new in the sense that all previous work on linear regression-based clustering did not focus on curve data but on individual, independent covariates $\mathbf{x}_i$ and their univariate dependent responses y_i (the classic multiple regression problem; Johnston, 1984). This translated into a data set that consisted of a random sample of *individual* scalar observations from K cluster-specific multiple regression functions. There was no notion of curves but only of single observations. The goal was to separately cluster the individual observations in the dataset. We show the effects of incorporating cluster memberships into the

regression mixtures framework in the next section.

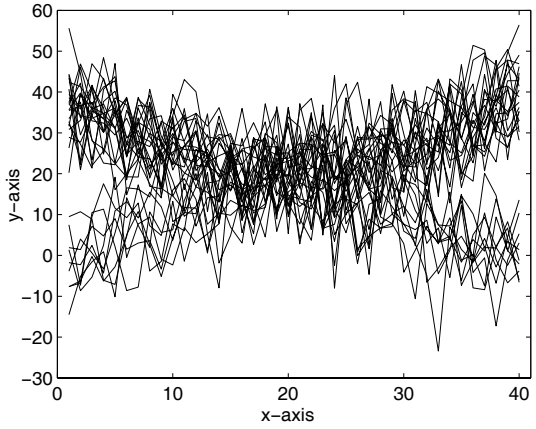
Finally, a number of authors have also pursued the Bayesian formulation of the problem. For example, Hurn et al. (2003) discuss solutions to the label-switching problem for Bayesian inference with regression mixtures, and Viele and Tong (2002) present consistency results of the posterior distribution.

Curve-level membership

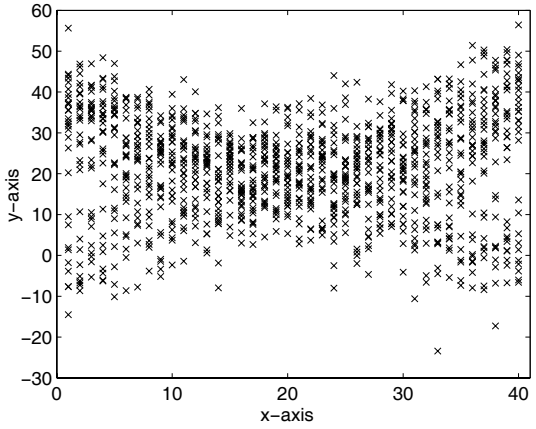
In this section, the novelty of employing curve-level memberships in the regression mixtures framework is discussed. In the original EM formulation for the general K cluster case (DeSarbo & Cron, 1988), the important notion of *sharing membership strength* along a curve is missing (and indeed, the notion of curves as data objects is also not addressed).

If a curve-specific membership probability w_{ik} is defined for each curve $\mathbf{y}_i$ and each cluster k, then it should collect membership information from each point in the curve to form the collective curve membership for $\mathbf{y}_i$. In contrast, previous methods defined (what amounted to) observation-specific membership probabilities w'_{ijk} for each univariate point y_{ij}, and cluster k. Although it is possible to define observation-specific membership probabilities to estimate the mixture parameters with curve data, the loss of consistent curve membership in the face of limited data will affect the clustering.

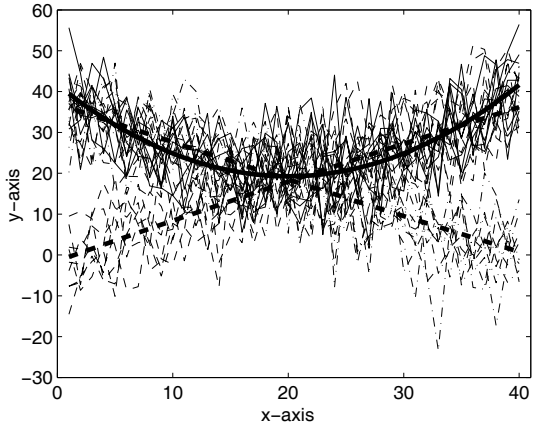
As an example, Figure 3.1 demonstrates the significance of curve-level membership. Figure 3.1(a) shows a set of simulated curves generated from three underlying polynomials. The same data is shown in Figure 3.1(b), but with the curve information removed. It is apparent that the problem is quite different without the curve information. Figure 3.1(c) shows the clustering that results from running a PRM on the curve data; the bolded lines are the cluster-specific models that were found, and



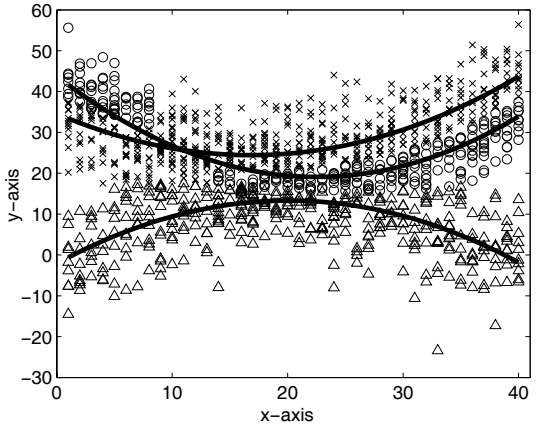
(a) Simulated curve data



(b) Simulated data w/o curve information



(c) Cluster results w/ curve information



(d) Cluster results w/o curve information

Figure 3.1: Example of the benefit of curve-level membership probabilities.

the different line styles (e.g., solid, dashed, etc.) give the classifications. The clustering in this figure matches the true underlying polynomials. In Figure 3.1(d), the clustering that results with the methods of DeSarbo and Cron (1988) is shown (i.e., without curve-level memberships). Again, the cluster-specific models are shown as bolded lines, and the classifications are given by the symbol styles (e.g., triangle, circle, etc.). Here we see that the cluster-specific models do not match the true models at all. Also of note is how the observation specific classifications divide up the space in a manner similar to K-means (i.e., specific regions of the space are assigned to specific clusters, and not specific curves). This effect can be summarized by stating that the clustering is carried out in observation space and not in curve space as is desired with curve-type data.

Mixtures of experts

Regression mixture models are also similar to mixtures of experts (ME; Jacobs et al., 1991; Waterhouse, 1997), or to the straightforward recursive extension to hierarchial mixtures of experts (Jordan & Jacobs, 1994). These models define gating networks which contain mixtures of generalized linear models (McCullagh & Nelder, 1983). The basic motivation is to model regions of the output space ($\mathbf{y}_i$) with region-tuned experts (generalized linear models), and then have the gating network decide which expert to use for a particular input ($\mathbf{x}_i$). This network/tree structure resolved from input space mirrors the way in which the classic procedures of CART (Breiman et al., 1984) and C4.5 (Quinlan, 1993) work.

Although mathematically similar in many respects, they differ from curve clustering models in a number of ways. Mixtures of experts focus on the easier problem of supervised learning as opposed to the more difficult unsupervised problem of clustering. Also, if we set the equivalence of "choosing an expert" in an ME model with

"choosing a cluster" in a PRM, then we see that ME models base this decision on the input variables $\mathbf{x}_i$, whereas PRMs use the unconditional mixture weight α_k which comes from the generative model for the unsupervised problem.

Furthermore, typically the input variables $\mathbf{x}_i$ for MEs are multidimensional and the output variables $\mathbf{y}_i$ are univariate. For PRMs, the situation is just the opposite; $\mathbf{x}_i$ is commonly represented as time and $\mathbf{y}_i$ is often multidimensional giving a vector of observations at each time point.

3.3.2 Model definition

Suppose we have a set Y of n curves as $\{\mathbf{y}_1, \ldots, \mathbf{y}_i, \ldots, \mathbf{y}_n\}$. Each curve has (a possibly unique) length of n_i with measurements observed at the points (or times) in $\mathbf{x}_i$. A p-th order polynomial regression relationship between $\mathbf{y}_i$ and $\mathbf{x}_i$ is assumed with an additive Gaussian error term (a common assumption in the presence of multiple exogenous, unexplained effects).

The regression of $\mathbf{y}_i$ on $\mathbf{x}_i$ can be summarized with the following equation:

$$\mathbf{y}_i = \mathbf{X}_i\boldsymbol{\beta} + \boldsymbol{\epsilon}_i, \qquad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}), \tag{3.4}$$

where the $n_i \times p$ regression matrix $\mathbf{X}_i$ is the Vandermonde matrix evaluated at $\mathbf{x}_i$, and $\boldsymbol{\beta}$ is the p-vector of regression coefficients. The p-th order Vandermonde matrix evaluated at $\mathbf{x}_i$ is equal to

$$\mathbf{X}_i = \begin{bmatrix} 1 & x_{i1} & x_{i1}^2 & \cdots & x_{i1}^p \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & x_{in_i} & x_{in_i}^2 & \cdots & x_{in_i}^p \end{bmatrix}.$$

This regression equation, along with the error model, defines the conditional PDF

of $\mathbf{y}_i$ given $\mathbf{x}_i$ as $\mathcal{N}(\mathbf{y}_i|\mathbf{X}_i\boldsymbol{\beta}, \sigma^2\mathbf{I})$. This PDF represents a probabilistic curve model that naturally allows for curves of variable length with unique measurement intervals and missing observations. Furthermore, the polynomial fit also takes advantage of smoothness information present in the data (see Section 3.6 for experimental results that demonstrate this effect).

We can incorporate this PDF into a mixture density by adding dependence of this PDF on k. In notation, this dependence is added in the form of subscripts on the parameters as $\{\boldsymbol{\beta}_k, \sigma_k^2\}$. The incorporation of these cluster-dependent PDFs into the conditional mixture density in (3.3) results in the definition of the PRM as

$$\begin{aligned} p(\mathbf{y}_i|\mathbf{x}_i, \Theta) &= \sum_k^K \alpha_k \, p_k(\mathbf{y}_i|\mathbf{x}_i \, \theta_k) \\ &= \sum_k^K \alpha_k \, \mathcal{N}(\mathbf{y}_i|\mathbf{X}_i\boldsymbol{\beta}_k, \sigma_k^2\mathbf{I}). \end{aligned} \tag{3.5}$$

The log-likelihood follows as the sum over all n curves of this conditional density. It takes the form

$$\log p(Y|X, \Theta) = \sum_i \log \sum_k^K \alpha_k \, p_k(\mathbf{y}_i|\mathbf{x}_i \, \theta_k). \tag{3.6}$$

We use this function to calculate the out-of-sample test log-likelihood scores for this model by substituting in an unseen dataset Y' for Y. This model definition is now used to derive the EM learning algorithm for curve clustering with PRMs.

3.3.3 EM algorithm for PRMs

In this section, we derive the EM algorithm for PRMs. It is assumed that the reader has familiarized themselves with the necessary EM theory in Appendix A. For the sake of notational simplicity, we also assume that every PDF is implicitly conditioned on a set of parameters (e.g., Θ or θ_k), and thus we leave out the explicit dependence

on the parameter vector in our notation.

We begin by letting z_i give the cluster membership for curve i, and we write the joint density of $\mathbf{y}_i$ and z_i as

$$\begin{aligned} p(\mathbf{y}_i, z_i|\mathbf{x}_i) &= \alpha_{z_i}\, p_{z_i}(\mathbf{y}_i|\mathbf{x}_i) \\ &= \alpha_{z_i}\, \mathcal{N}(\mathbf{y}_i|\mathbf{X}_i\boldsymbol{\beta}_{z_i}, \sigma^2_{z_i}\mathbf{I}). \end{aligned} \tag{3.7}$$

The cluster memberships $\{z_i\}$ are regarded as being hidden. The hidden-data density then becomes the posterior $p(z_i|\mathbf{y}_i, \mathbf{x}_i)$. The complete-data log-likelihood function $\mathcal{L}_c$ can be calculated by taking the sum over all n curves of the log joint density in (3.7):

$$\mathcal{L}_c = \sum_i \log \alpha_{z_i}\, \mathcal{N}(\mathbf{y}_i|\mathbf{X}_i\boldsymbol{\beta}_{z_i}, \sigma^2_{z_i}\mathbf{I}). \tag{3.8}$$

E-step

In the E-step, we calculate the posterior $p(z_i|\mathbf{y}_i, \mathbf{x}_i)$ which gives the *membership* probability that the i-th curve was generated from cluster z_i. The membership probability takes the form

$$\begin{aligned} w_{ik} = p(z_i = k|\mathbf{y}_i, \mathbf{x}_i) &\propto \alpha_k\, p_k(\mathbf{y}_i|\mathbf{x}_i) \\ &= \alpha_k\, \mathcal{N}(\mathbf{y}_i|\mathbf{X}_i\boldsymbol{\beta}_k, \sigma^2_k\mathbf{I}). \end{aligned} \tag{3.9}$$

The posterior expectation of $\mathcal{L}_c$ in (3.8) is then taken with with respect to the posterior above to get the Q-function. The Q-function is calculated as follows:

$$Q = \mathrm{E}[\mathcal{L}_c|\mathbf{y}_i, \mathbf{x}_i] = \sum_i \sum_k w_{ik} \log \alpha_k\, \mathcal{N}(\mathbf{y}_i|\mathbf{X}_i\boldsymbol{\beta}_k, \sigma^2_k\mathbf{I}). \tag{3.10}$$

M-step

In the M-step, we maximize Q with respect to the parameters $\{\boldsymbol{\beta}_k, \sigma_k^2, \alpha_k\}$. The solutions are straightforward and are given as

$$\hat{\boldsymbol{\beta}}_k = \left[\sum_i w_{ik}\mathbf{X}_i'\mathbf{X}_i\right]^{-1} \sum_i w_{ik}\mathbf{X}_i'\mathbf{y}_i, \tag{3.11}$$

$$\hat{\sigma}_k^2 = \frac{1}{\sum_i w_{ik}} \sum_i w_{ik} \left\|\mathbf{y}_i - \mathbf{X}_i\boldsymbol{\beta}_k\right\|^2, \tag{3.12}$$

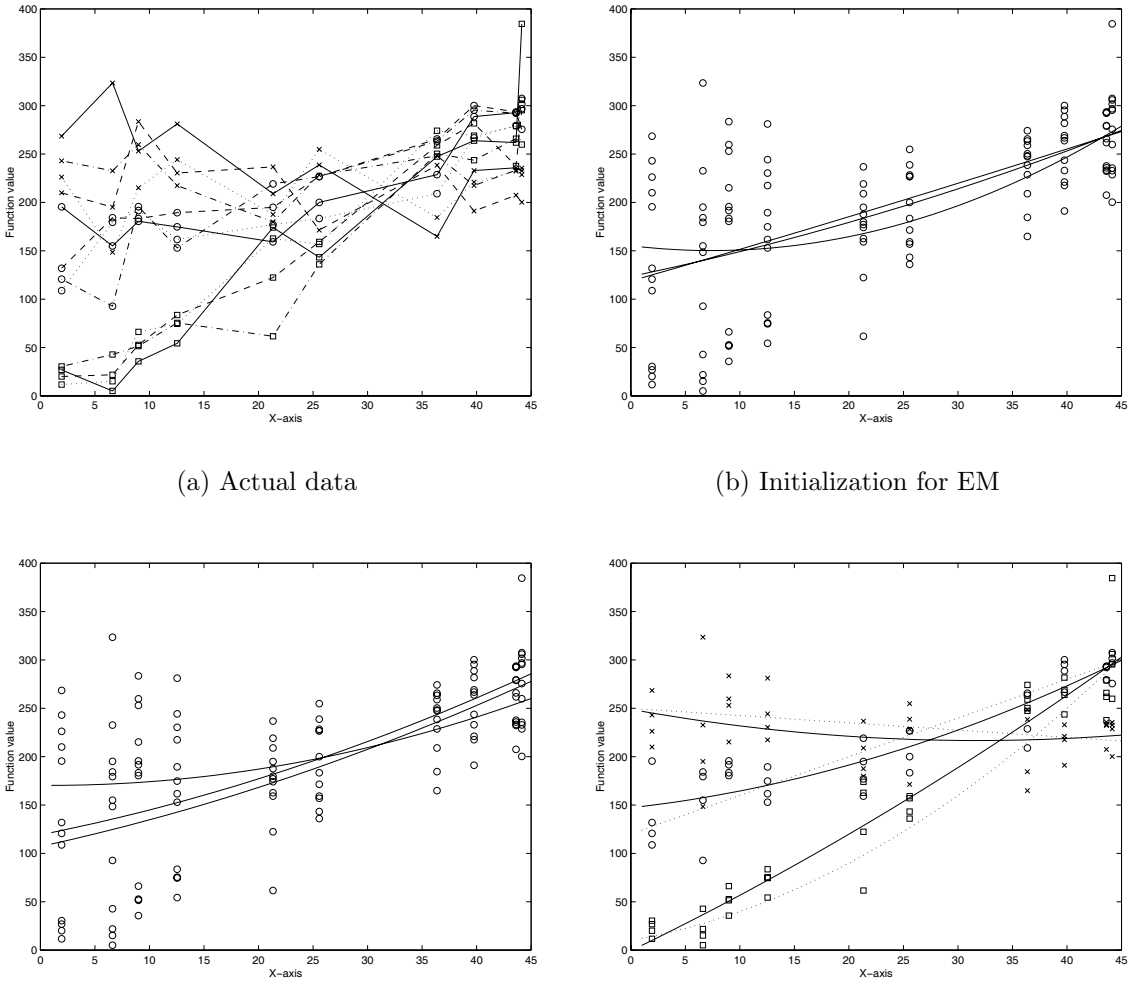
and

$$\hat{\alpha}_k = \frac{1}{n}\sum_i w_{ik}. \tag{3.13}$$

These update equations are equivalent to the well-known weighted least-squares solutions (Draper & Smith, 1981).

The computational complexity of this EM algorithm is linear in the number of curves (or the total number of points in these curves). Initialization is carried out by randomly sampling values for the membership probabilities and then beginning the iterations with the M-step. Convergence is detected when the ratio of the incremental improvement in log-likelihood to the initial incremental improvement during the second iteration drops below a threshold (e.g., $1\times\mathrm{E}^{-6}$).

Figure 3.2 graphically demonstrates a simulated data example of running EM for polynomial regression mixtures. Four curves were sampled from each of three different underlying polynomials and were clustered using a PRM. Figure 3.2(a) shows all curves presented to the EM algorithm. Notice that we plot the actual class labels here, but this information is not given to the algorithm. Figure 3.2(b) shows the initial guess of the algorithm for the three underlying polynomials. Figure 3.2(c) shows the same cluster centers after one iteration, and Figure 3.2(d) shows the final clustering as output after four iterations. The last plot also shows the classifica-



(a) Actual data

(b) Initialization for EM

(c) Centers after one iteration

(d) Final clusters (four iterations)

Figure 3.2: Trace of the EM algorithm for a PRM at various iterations: (a) all curves presented to the algorithm, (b) initial guess for the three cluster centers, (c) cluster centers after one iteration of EM, (d) cluster locations (solid) after EM convergence (iteration 4), and locations of the true data-generating trajectories (dotted).

tion resulting from the clustering (which is perfect in this example) and shows the underlying true polynomials as dotted-lines.

Experimental results with PRMs are presented at the end of this chapter in Section 3.6. But first we extend this model to spline regression mixtures and kernel regression mixtures.

3.4 Spline regression mixtures

In this section, we introduce a useful extension of the regression mixtures framework to spline regression mixtures (SRM). The extension allows for semi-parametric modelling of curve data as opposed to the strictly parametric polynomial regression model described in the previous section.

3.4.1 Related work

Maes and Hastie (1997) develop a related mixture of splines model that they use to discover curve-type features in data from the time-frequency domain generated from speech utterances. The data consist of "images" of frequency counts over a lattice of time-frequency points. The learning/discovery procedure sweeps across an image in a number of rounds, and an EM procedure learns model parameters from within sliding windows across time.

This problem is quite different than that of clustering sets of curves but nonetheless the mathematics of what they have done is similar. However, their methodology is quite specific and highly optimized to include the many constraints that are required for this particular problem domain. In contrast, we introduce the definition of mixtures of splines in a general model-based curve clustering setting that does not require any special constraints or procedures for any particular problem domain.

James and Sugar (2003) develop a *random effects* mixture of splines model similar to the model we define in Chapter 4. However, this model is different from the spline mixtures defined here since the incorporation of random effects leads to a two-level hierarchical mixture model. This type of model can be seen as founded on the spline mixtures that are defined here. We describe them as such when we discuss random effects regression mixtures (RERM) in Chapter 4.

3.4.2 Definition of splines

Splines are piece-wise polynomials that meet certain continuity conditions at the breakpoints (de Boor, 1978; Eubank, 1988; Green & Silverman, 1994). For example, we might require that a spline have two continuous derivatives throughout some valid interval. This results in curves which look and behave in a smooth manner. In spline theory, the set of breakpoints are often called knots.

We implement splines based on B-splines in this thesis (de Boor, 1986; Eilers & Marx, 1996). This is a common choice because B-splines are particularly efficient for computational purposes due to the block-diagonal basis matrices that result. Let $\zeta = \{t_1 < \cdots < t_N\}$ give a nondecreasing knot sequence, and let $[t_m, t_{m+1})$ be the half-open interval from t_m to t_{m+1}. Then the p-th order B-spline B_{mp} is a piece-wise polynomial that has finite support over $[t_m, t_{m+p})$ and is zero everywhere else. In general, the polynomial pieces of B_{mp} are of degree $p-1$. B_{mp} is defined in a special way so that

$$\sum_{m}^{L} B_{mp}(x) = 1,$$

where $L = N - p$ gives the number of B-splines defined over the knot sequence ζ.

The spline $s(x)$ is then defined as a linear combination of the B_{mp} as

$$s(x) = \sum_{m}^{L} B_{mp}(x) c_m, \tag{3.14}$$

where c_m gives the spline coefficients.

We can think of this as an expansion of $s(x)$ over the L basis functions B_{mp}. The basis functions can be calculated using a simple recurrence relation. All that is needed is to pick the order of the spline and then simply run the recurrence for each value of x. A common choice is to use fourth order B-splines which result in cubic spline functions (since the degree of each polynomial piece in a fourth order B-spline is at most 3).

To represent the curve $\mathbf{y}_i$ as a spline, we equate the j-th point y_{ij} to the value of the spline function evaluated at the j-th time point x_{ij}. In other words, we set $y_{ij} = s(x_{ij})$ for all $1 \leq j \leq n_i$.

The equation for $\mathbf{y}_i$ can be written in matrix form to simplify the notation. $B_{mp}(\mathbf{x}_i)$ is defined as the n_i-vector of the individual time points of $\mathbf{x}_i$ evaluated under B_{mp}. The spline basis matrix $\mathbf{B}_i$ is then the $n_i \times L$ matrix

$$\mathbf{B}_i = \left[\begin{array}{cccc} B_{1p}(\mathbf{x}_i) & B_{2p}(\mathbf{x}_i) & \cdots & B_{Lp}(\mathbf{x}_i) \end{array} \right]. \tag{3.15}$$

Use of this matrix allows us to represent the curve $\mathbf{y}_i$ using a spline in the form

$$\mathbf{y}_i = \mathbf{B}_i \mathbf{c}, \tag{3.16}$$

where $\mathbf{c}$ is a vector of spline coefficients.

3.4.3 EM algorithm for SRMs

The EM algorithm for SRMs can be derived using the same procedure as used for PRMs in Section 3.3.3. First, $\mathbf{B}_i$ is defined as the spline basis matrix evaluated at $\mathbf{x}_i$. Then, a regression relationship is assumed between $\mathbf{y}_i$ and $\mathbf{x}_i$ in the standard way:

$$\mathbf{y}_i = \mathbf{B}_i\boldsymbol{\beta}_k + \boldsymbol{\epsilon}_i, \qquad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \sigma_k^2\mathbf{I}). \tag{3.17}$$

The form of the regression and the error model result in the cluster-specific conditional PDF for $\mathbf{y}_i$:

$$p_k(\mathbf{y}_i|\mathbf{x}_i, \theta_k) = \mathcal{N}(\mathbf{y}_i|\mathbf{B}_i\boldsymbol{\beta}_k, \sigma_k^2\mathbf{I}). \tag{3.18}$$

This defines the probabilistic curve model for SRMs. The E- and M-steps of the EM algorithm can be derived by directly substituting this curve model into (3.7) which gives the joint PDF for curve $\mathbf{y}_i$ and cluster label z_i for PRMs. The EM solution that results is identical to that for PRMs with the Vandermonde matrix $\mathbf{X}_i$ replaced with the spline basis matrix $\mathbf{B}_i$.

The complexity, initialization procedure, and convergence condition are all identical to those with a PRM. In fact, an SRM can be learned using the exact same code that is used to train a PRM. All that is required is to pre-compute the spline basis matrix $\mathbf{B}_i$ and replace $\mathbf{X}_i$ with this new matrix as input to the EM algorithm for PRMs. The output regression coefficients will be spline coefficients that represent mean spline curves for each component PDF.

3.4.4 Discussion

Choosing the number and location of knots

Typically, the problem at hand is not overly sensitive to the number and location of knots. It is common to place a number of knots uniformly spaced across the interval in question, although more knots are needed in areas of rapid function change.

In any case, automatic techniques for the selection of the number of knots and their locations can be used. This area of research has received much attention (Friedman & Silverman, 1989; Kooperberg & Stone, 1991, 1992). Many schemes revolve around the idea of starting off with many knots and employing *knot deletion* to reduce the size of the knot set. In contrast, smoothing ideas can also be used in which a large number of knots is always used but measures are used to penalize the fitting function. Often the penalty measures are based on function curvature (O'Sullivan, 1986, 1988; Eilers & Marx, 1996).

One can extend the current definition of SRMs to handle this type of automatic complexity selection, but because the unsupervised clustering problem is difficult in itself, it is wise to fix the number and location of knots ahead of time. In this thesis, we will use knot sequences which are uniformly spaced across the interval in question unless stated otherwise.

Splines vs. polynomial regression

The EM algorithms for PRMs and SRMs are nearly equivalent. The main difference between the two EM algorithms is that an SRM requires the fitting of a larger number of parameters. For example, for cubic splines, the number of coefficients in a single spline coefficient vector is equal to $N - 4$, where N gives the length of the knot sequence.

The minimum size for the knot sequence with cubic splines is normally taken as 8 (this is because of required knot-end conditions). However, the maximum size is potentially infinite. The larger the relevant interval, the larger the knot sequence must be in order to efficiently model each region of the interval in question.

This results in a spline coefficient vector that can grow quite large. Even for small intervals, a typical length of coefficient vector might be 10 or 15 or even larger, depending on the situation. If we factor in the number of clusters, then the number of coefficients that must be fit to the data during a run of EM can grow quite large. This allows the problem of over-fitting to creep into the model learning process.

At a basic level, PRMs should be preferred if it can be determined that the curves are not "wildly" non-polynomial since the more simple model will suffer less from the over-fitting problem. However, SRMs provide an increased level of flexibility at almost no computational cost (except for the increase in parameters) that can be exploited in many domains where it is clear that the curves are not polynomial. In these cases, if over-fitting is of particular concern, then the number of knots can be decreased at the expense of less modelling flexibility.

In Section 3.6, we demonstrate another useful feature of SRMs as non-parametric curve simulators. Since the resulting sets of generated curve data are not tied to any particular parametric scheme, these sets of curves are ideal for simulated data experiments with curve and non-curve clustering techniques.

3.5 Kernel regression mixtures

In this section, we introduce a novel extension of PRMs by modelling the density function $p_k(\mathbf{y}_i|\mathbf{x}_i, \theta_k)$ as a non-parametric regression model (originally described in Gaffney & Smyth, 1999). These types of models can be used to relax the assumptions

placed on the form of the regression function even further than splines allow for. This approach is inherently more data-driven.

Non-parametric function estimation has been studied in a number of different settings, for example, kernel smoothing (Wand & Jones, 1995), local polynomial modelling (Fan & Gijbels, 1996), and density estimation (Silverman, 1986). In our context, by modelling the component densities as nonparametric regression models, we can cluster curve data for which the general relationship between y and x is uncertain, or for when we do not wish to make any such assumptions on our regression functions.

The basic idea behind kernel regression is that we can approximate any arbitrary function with a series of simple locally-weighted functions, such as linear regression functions. We approximate the unknown function at a point x_0, by running a locally-weighted linear regression (of order p) about the point x_0, and report the prediction $\hat{y}$ as the height of this fit. The weights are produced by a symmetric kernel (e.g., standard Gaussian density) centered about the point x_0, whose purpose is to *down-weight* points far away from x_0. When the random component for the locally fit regression model is Gaussian, the solution for the regression coefficients can be calculated using weighted least squares.

We include kernel regression model components into our regression mixture framework by modifying the EM algorithm in Section 3.3.3. Instead of calculating $\hat{\boldsymbol{\beta}}_k$ and $\hat{\sigma}_k^2$ in (3.11) and (3.12), we require the explicit calculation of the mean $\hat{y}_{ij}$ (predicted value) and variance $\hat{\sigma}_{ij}^2$ at every associated x_{ij}, for each cluster k. We do this by solving a locally-weighted least squares problem (the weights become the membership probabilities multiplied by the kernel weights) at each point. With this modification, the rest of the framework remains intact.

One other consideration is the *bandwidth* for the kernels. The bandwidth deter-

mines the spread of the density for a kernel. Much has been written on the subject of learning this parameter from the data (e.g., Fan & Gijbels, 1996). In practice, we assume a known fixed bandwidth but clearly one could generalize our algorithms to include a "data-adaptive bandwidth" component.

The complexity of the EM algorithm for kernel regression models scales as Nm, where N gives the total number of points in all the curves, and m gives the number of unique x-values in the dataset. This is due to the fact that we must perform a separate weighted-least squares regression at each of the unique x-points using (potentially) all of the N y-points for each curve.

The main drawback of KRMs is that they are computationally prohibitive. SRMs are computationally cheap and provide for similar flexibility. Thus, we do not pursue the use of KRMs in the remainder of this thesis. For more extensive details and further analysis with KRMs, see Gaffney and Smyth (1999).

3.6 Experimental results

In this section, we report experimental results with simulated data that show the importance of employing curve modelling techniques when clustering sets of curves. For these experiments we focus on using a spline mixture model (an SRM) as the data generating model and then compare PRMs and Gaussian mixtures on this data. In this way, the results are unbiased since the data generating model is different than each of the comparison models. Results for comparisons between SRMs and Gaussian mixtures on data generated from a PRM are similar and are not shown here.

Each of the experiments in this section was carried out as follows. A random, three component SRM (of order 4) was chosen by sampling three spline coefficient vectors (of length 8) from a normal distribution centered around zero (with standard

deviation 2). The sequence of knots (of length $8+4$) was linearly spaced among the x-axis from 0 to n, where n was set to either 20 or 50 depending on the experiment.

This model was used to generate 25 different training sets of 50 curves each and 25 testing sets of 100 curves each. Both PRM and Gaussian mixtures were trained on each training subset and test log-likelihood scores were calculated on the test sets and averaged across the 25 subsets (the log-likelihood scores were calculated based on the log-likelihood equations defined for each model above). This whole process was then repeated three times with three different randomly selected PRMs, resulting in test scores that were averaged over 75 different test subsets from three different spline mixture models. These averaged log-likelihood scores are reported in what follows. Three examples of the resulting generated data are provided in Figure 3.3. The bolded lines in the figure represent the mean spline curves for each of the three clusters.

Accounting for smoothness information

An assumption with curve data is that the underlying model is inherently smooth. Vector-based clustering methods such as Gaussian mixtures do not account for this information. For example, Figure 3.4 shows the results of fitting an SRM, a PRM, and a Gaussian mixture model to a set of curves generated from an SRM. The bolded lines represent the mean curves for each of the three clusters in this example.

Figure 3.4(a) is identical to the data generating model and shows the underlying spline curves as bolded lines. Figure 3.4(b) shows the curve models output from a PRM. The component regression models are of order seven in this example. The smoothness information is accurately accounted for with the PRM. Figure 3.4(c) shows the mean *vectors* output from Gaussian mixtures. It is clear that this model does not account for the smoothness information very well, if at all.

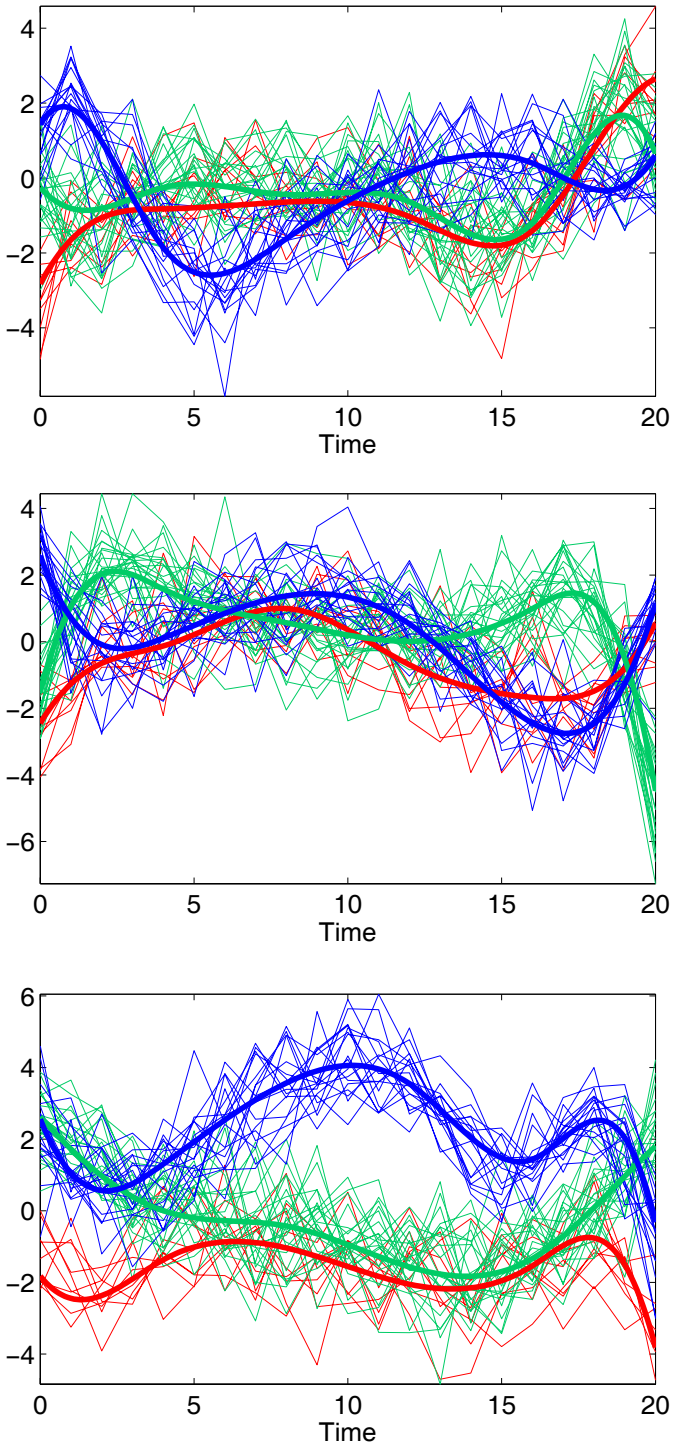
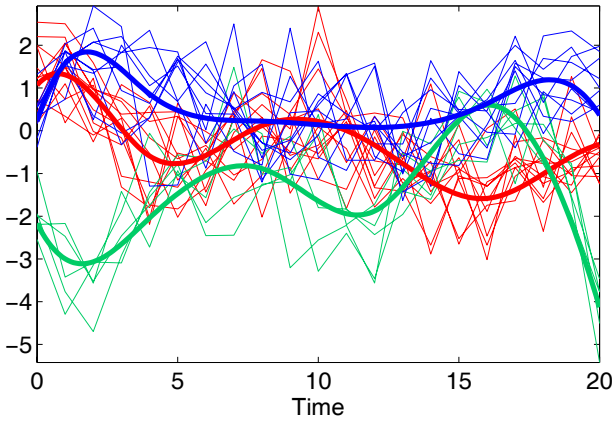
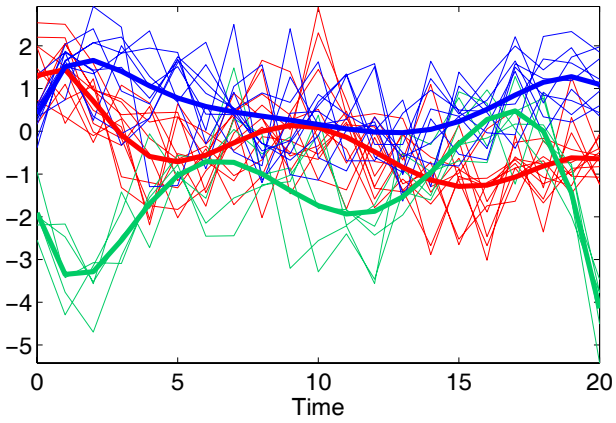


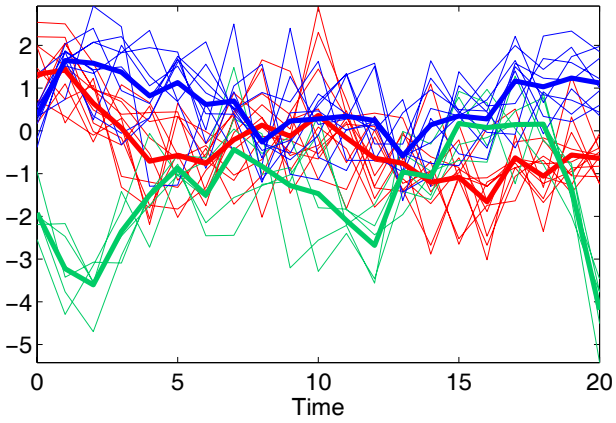
Figure 3.3: Example of the spline mixture data generated for the experiments in this section. The bolded lines represent the mean spline curves for each of the three clusters.



(a) Underlying SRM



(b) Output from PRM



(c) Output from Gaussian mixtures

Figure 3.4: Smoothness information accounted for by SRM, PRM, and Gaussian mixtures. The bolded lines are the mean cluster models.

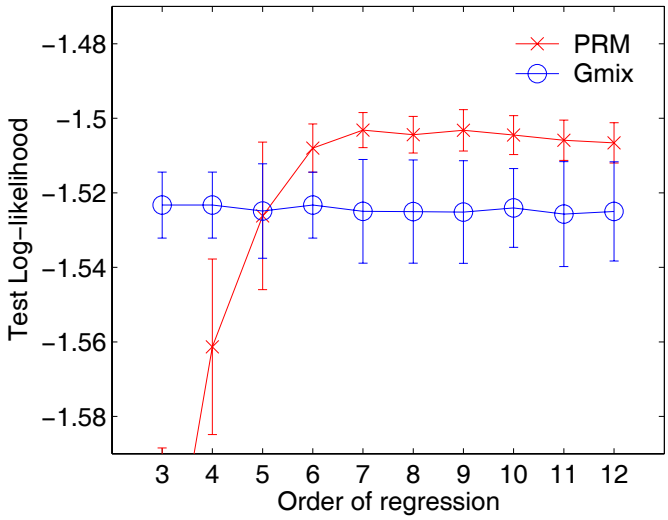


Figure 3.5: Comparison of Gaussian mixtures and PRMs for various orders of the regression fit with PRM. The error bars show one standard deviation.

Figure 3.5 shows the results of comparison experiments between Gaussian mixtures and different orders of regression models for PRM. The y-axis is averaged test log-likelihood, and the x-axis is the order of regression model used by PRM. The error bars show one standard deviation. For small orders such as cubic or quartic, the Gaussian mixture model out-performs PRM since the PRM model is too simple to model the rapidly changing curve data. But for orders of 5 or greater, the PRM is able to leverage the smoothness information to generate clusterings which are predictively better than those from Gaussian mixtures.

Accounting for variable length curves

A common problem with curve data is that curves often exhibit different lengths over the dataset. This creates problems for vector-based clustering methods since the curves do not conform to a fixed-dimensional vector space. For example, Figure 3.6 shows an example of variable-length curve data generated from random SRMs as described above. The plotted circles show the ends of each curve. A common solution

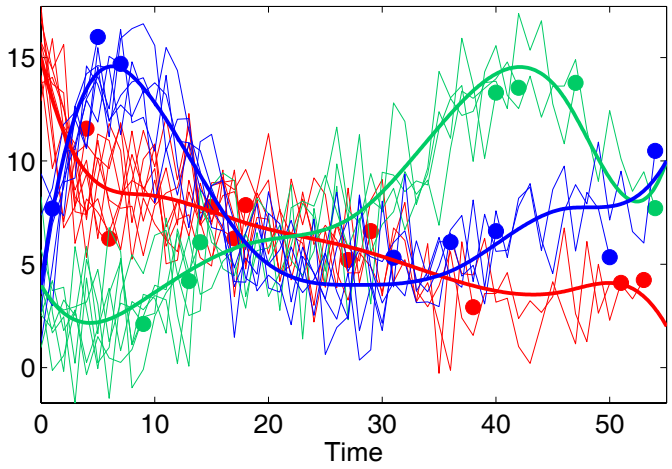


Figure 3.6: An example of variable length curve data generated from a spline mixture model. The circles indicate the ends of each curve.

for vector-based clustering methods when presented with such data is to truncate the curve data to a particular fixed-length and then cluster with these truncated curves. Figure 3.7 demonstrates that this methodology is inferior to using all of the curve data in a curve-based analysis.

The figure shows the results of experiments between Gaussian mixtures and PRMs with variable-length simulated SRM data. The curves all have a minimum length of 1 and a maximum length of 51. In the experiments, PRM was allowed to train on all of the variable length curves, while Gaussian mixtures truncated the curves to a particular fixed size before training. However, the test data was truncated for both PRM and Gaussian mixtures so that the score comparisons were fair.

At small truncation sizes (e.g., at a length of 5 on the x-axis in the figure), Gaussian mixtures only gets to "see" the first five points. However, since there is not much curve variance in the first five points, the difference between Gaussian mixtures and PRM is minimized on the truncated test set.

At larger truncation sizes, Gaussian mixtures is able to train on larger portions of the curve dataset. However, since not all curves meet this size, many of the

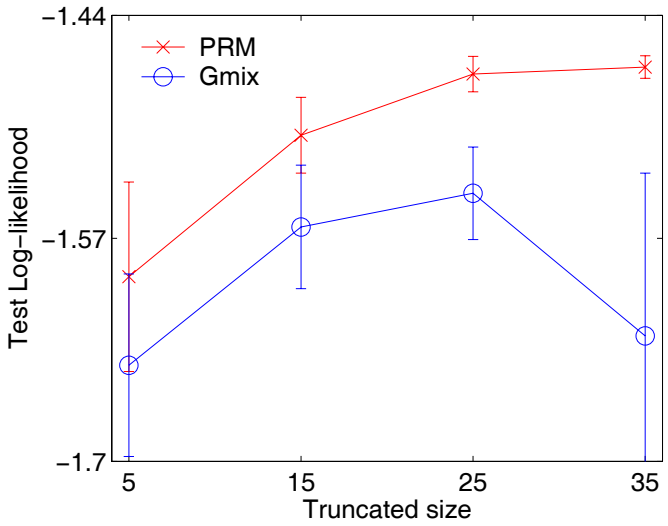


Figure 3.7: Results of experiments between Gaussian mixtures and PRM on variable-length curve data generated from an SRM.

curves are thrown out all together. This results in a degradation of prediction power as compared to PRMs on the now much larger test sets (the test sets are larger since the truncation size has increased). Furthermore, since a larger portion of each individual curve is included in the truncated test set, PRM is better able to demonstrate its leveraging of the smoothness information on the test set.

At the largest truncation sizes, Gaussian mixtures performs poorly since hardly any curves remain in the training set; most of the curves are smaller than the largest of the curves. Furthermore, now the test set includes almost the entire portion of each curve and thus the curve-based approach performs better. These results indicate that curve truncation is a particularly bad solution for dealing with variable-length curve data.

Accounting for irregular observation points

Another common problem with curve data is that each curve may have been observed/sampled at different points in time. This creates more problems for vector-

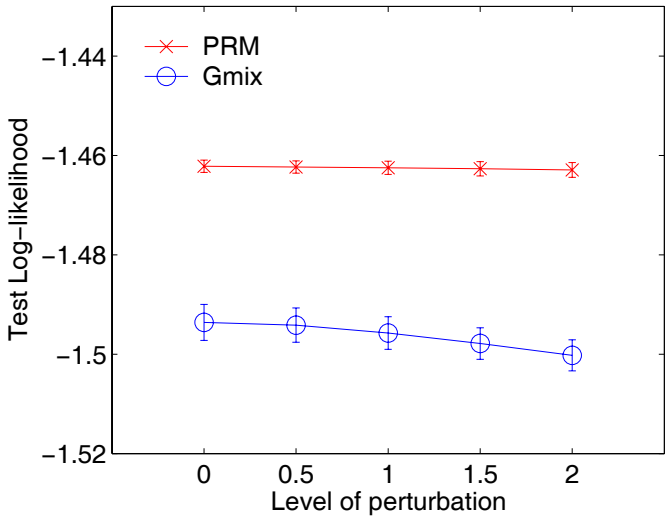


Figure 3.8: Results of experiments with irregularly sampled curve data generated from an SRM.

based clustering methods since none of these curves exactly correspond to any particular common vector-space. In practice, this situation is often ignored, and the data is treated as if it were measured at the same time points.

Figure 3.8 demonstrates the potential problems with this approach. The figure shows the results of comparison experiments between Gaussian mixtures and PRM on irregularly sampled curve data generated from an SRM. The experiments consisted of training PRM and Gaussian mixtures on the same datasets with various levels of perturbation added in time.

For example, at the 0.5 level of perturbation in the figure, a different random offset (a zero-mean normal offset with 0.5 standard deviation) was added to the time points before sampling the values of each individual training curve. The true value of the time points was given to PRM since curve-based methods model the conditional $p(\mathbf{y}_i|\mathbf{x}_i)$ for time $\mathbf{x}_i$. However, the Gaussian mixture model ignores this small difference and assumes that all the points are measured at the same exact time.

The results show that the predictive capability of the Gaussian mixture model degrades as the disparity between the actual and perceived time points at which each curve was sampled increases. This shows that curve-based approaches are more appropriate for clustering irregularly sampled curve data.

3.7 Summary

In this chapter, we discussed the specific techniques, models, and algorithms that directly address the curve clustering problem. Except for the alignment problem, the methods presented in this section addressed each of the faults that were pointed out with the standard clustering techniques in Chapter 2

We introduced polynomial regression mixtures (PRM), spline regression mixtures (SRM), and kernel regression mixtures (KRM) for curve clustering. The component models of PRMs are based on parametric regression functions that represent the mean curve using polynomials. The novelty of PRMs is in their extension of existing linear regression mixture models to explicitly handle curve data with curve-level membership functions.

SRMs relax this parametric requirement by using flexible spline models to represent the mean curves for the clusters. The extra flexibility is obtained at the expense of an increase in the number of parameters. However, there is no significant increase in computational complexity of the EM algorithm for SRMs as compared to that of the parametric PRMs. This makes SRMs an attractive alternative to PRMs for flexible curve clustering.

KRMs provide even more flexibility than SRMs. KRMs employ local polynomial modelling at each time point along the x-axis to determine the actual mean curves on a point-by-point basis. Unlike spline models, they do not enforce any smoothness

constraints, thereby allowing for maximum flexibility. The main problem with these models is that they are computationally prohibitive. For large amounts of data, SRMs are recommended as a computational alternative to the KRM. Nonetheless, KRMs are a novel extension of the regression mixtures framework

Finally, in this section, we reported extensive simulated data experiments that show the importance of employing curve modelling techniques for curve clustering. The experiments demonstrated that non-curve-based techniques such as Gaussian mixtures are not able to efficiently deal with variable length curves, do not efficiently handle irregular sampled curves or curves with missing measurements, and they do not account for the inherent smoothness information in curves.

Chapter 4

Random effects regression mixtures

4.1 Introduction

In this chapter, we introduce an extension of the regression mixtures framework that allows for the modelling of within-cluster heterogeneity. The clustering models of Chapter 3 can be used to effectively account for subpopulations of *homogeneous* behavior. However, more care should be taken when considerable variability exists within each subpopulation or group.

For example, suppose we have a set of individuals from K groups. The implicit assumption that is made when using a PRM is that each group of individuals are sufficiently homogeneous to appropriately fit the common group component model. However, in the presence of significant variability within any group, one would have to resort to fitting more groups K to be able to sufficiently describe the data, an option that is left undesirable.

What is needed is the ability to let an individual vary from the template for its

group, yet still exhibit the underlying behavior that distinguishes this group from the rest. This leads to the development of *random effects regression mixtures* (RERM). A hierarchical model structure is defined with a mixture on parameters at the top level (*parameter-level*) and an individual-specific regression model at the bottom level (*data-level*). An EM algorithm is then defined using MAP estimation to enable learning in the hierarchy.

In Section 4.2, the relevant prior work is discussed. Section 4.3 introduces the hierarchical model structure used in an RERM. In Section 4.4, the derivation of the MAP-based EM algorithm for RERMs is given. This is followed in Section 4.5 by brief experimental results comparing RERM to both PRM and to Gaussian mixtures. Finally, the chapter is concluded in Section 4.6 with a summary.

4.2 Prior work

EM as it relates to MAP estimation is discussed in the excellent book by McLachlan and Krishnan (1997). Also, Ormoneit and Tresp (1996) discuss the application of EM for MAP estimation using multivariate Gaussian mixture models. Cadez and Smyth (1999) discuss the use of MAP-based EM algorithms within a general probabilistic clustering framework using Bayesian hierarchical model structures.

Random effects mixtures in the Bayesian context were introduced by Lenk and DeSarbo (2000). There they focus on fully Bayesian inference for mixtures of generalized linear models with random effects. They use Markov Chain Monte Carlo (MCMC) techniques (Gelfand & Smith, 1990) for fully Bayesian inference (i.e., they produce posterior distributions on parameters instead of making point estimates). We take their lead and define a similar hierarchical model structure but instead develop a MAP-based EM procedure for parameter inference in the case of mixtures

of polynomial regression models and mixtures of splines.

In parallel to our development of RERMs (Gaffney & Smyth, 2003) James and Sugar (2003) developed a functional clustering model for sparsely sampled functional data. Their motivation extends from earlier work (James & Hastie, 2000) in linear discriminant analysis with irregularly sampled curves. They define a model similar to what is defined here but work with low-dimensional projections of group means and do not develop a full MAP-EM algorithm with conjugate hyperpriors.

4.3 Hierarchical model structure

Suppose we have a set of n individuals from K groups and that each individual i generates a curve $\mathbf{y}_i$ of length n_i according to the normal regression model

$$\mathbf{y}_i = \mathbf{X}_i\boldsymbol{\beta}_i + \boldsymbol{\epsilon}_i, \qquad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}). \tag{4.1}$$

This leads to the conditional density of the form

$$p(\mathbf{y}_i|\mathbf{X}_i\boldsymbol{\beta}_i, \sigma^2) = \mathcal{N}(\mathbf{y}_i|\mathbf{X}_i\boldsymbol{\beta}_i, \sigma^2\mathbf{I}), \tag{4.2}$$

with Vandermonde matrix $\mathbf{X}_i$ and coefficient vector $\boldsymbol{\beta}_i$ as with PRMs. Notice that each individual has its own regression model through the parameter $\boldsymbol{\beta}_i$ (i.e., there is no $\boldsymbol{\beta}_k$ here). This is the random effect. In fact, there is no dependence on group membership at all at this level, the bottom-level (or *data-level*) of the hierarchy. Instead at this level we allow for individual-specific heterogeneity.

At the top-level of the hierarchy there is a probabilistic model that describes the distribution of the parameters $\boldsymbol{\beta}_i$ for each individual. Let z_i give the group membership for curve $\mathbf{y}_i$. Knowledge of membership allows us to define a distribution

on $\boldsymbol{\beta}_i$ according to the group template as

$$p(\boldsymbol{\beta}_i|z_i, \phi_{z_i}) = \mathcal{N}(\boldsymbol{\beta}_i|\mu_{z_i}, \mathbf{R}_{z_i}), \qquad \phi_{z_i} = \{\mu_{z_i}, \mathbf{R}_{z_i}\},$$

where $\mathcal{N}$ is the multivariate normal density with mean μ_{z_i} and covariance $\mathbf{R}_{z_i}$. Unconditional of class membership, the prior for $\boldsymbol{\beta}_i$,

$$p(\boldsymbol{\beta}_i|\Phi) = \sum_k \alpha_k \mathcal{N}(\boldsymbol{\beta}_i|\mu_k, \mathbf{R}_k), \tag{4.3}$$

is a finite mixture with $\Phi = \{\alpha_1, \ldots, \alpha_K, \phi_1, \ldots, \phi_K\}$. At this level of the hierarchy we allow for the clustering of homogeneous group behavior. As a result, we now have a finite mixture model allowing for homogeneous group behavior at the top-level, and a regression model allowing for individual heterogeneity at the bottom-level.

One possible problematic issue with this model is that K distinct covariance matrices must be estimated. One solution to avoid possible estimation problems is to pool the K covariance matrices into a single representative matrix $\mathbf{R}$. Banfield and Raftery (1993) introduced a number of methods to reparameterize covariance matrices so that instead of all clusters sharing a single $\mathbf{R}$, they only share certain chosen characteristics (e.g., orientation, size, or shape).

We can also introduce a Bayesian regularization methodology to the framework to curb problematic estimations. We define hyperpriors for $\mathbf{R}_k$ and α_k in this regard. The standard conjugate priors for $\mathbf{R}_k^{-1}$ and $\alpha = (\alpha_1, \ldots, \alpha_K)'$ are the Wishart density $W(\mathbf{R}_k^{-1}|\mathbf{R}_0, \nu)$ and the Dirichlet density $D(\alpha|\eta)$ (Buntine, 1994; Gelman et al., 1995; Ormoneit & Tresp, 1996). With the estimation problem addressed, the model is completed by assuming a simple non-informative prior for both σ^2 and μ_k.

4.3.1 MAP estimation

The hierarchical model specification naturally leads to a MAP estimation. In other words, it is natural to define the posterior of the parameters given the data as being proportional to the likelihood of the bottom-level times the prior of the top-level. Let $\Theta = \{\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_n, \sigma^2\}$ be the parameters at the bottom-level and let Φ be the parameters at the top-level. Then, for the set of n curves $Y = \{\mathbf{y}_i\}_i^n$ and the corresponding set of time points $X = \{\mathbf{x}_i\}_i^n$, the MAP objective function $\mathcal{M}$ is proportional to the posterior $p(\Theta, \Phi|Y, X)$ of the parameters. The objective function for the parameters Θ and Φ is defined as

$$\begin{aligned} \mathcal{M}(\Theta, \Phi) &= \log\left[p(Y|X, \Theta, \Phi)p(\Theta, \Phi)\right] \\ &= \log\left[p(Y|X, \Theta)p(\Theta|\Phi)p(\Phi)\right], \end{aligned}$$

where

$$\begin{aligned} p(Y|X, \Theta) &= \prod_i \mathcal{N}(\mathbf{y}_i|\mathbf{X}_i\boldsymbol{\beta}_i, \sigma^2\mathbf{I}), \\ p(\Theta|\Phi) &= \prod_i \sum_k^K \alpha_k \mathcal{N}(\boldsymbol{\beta}_i|\mu_k, \mathbf{R}_k), \end{aligned}$$

and

$$p(\Phi) = D(\alpha|\eta) \prod_k W(\mathbf{R}_k^{-1}|\mathbf{R}_0, \nu).$$

4.4 MAP-based EM algorithm

Analysis of $\mathcal{M}$ leads to the conclusion that direct maximization is not feasible. However, we can derive a MAP-based EM algorithm that produces consistent parameter estimates. The derivation proceeds by first declaring both the group memberships

z_i and the individual-specific regression coefficients $\boldsymbol{\beta}_i$ as being hidden. This results in the joint posterior $p(\boldsymbol{\beta}_i, z_i|\mathbf{y}_i)$ as the hidden-data density.

In the EM framework, the function $\mathcal{M}$ is referred to as the *incomplete-data function* since it does not contain all the missing data. It is the missing data that makes the problem complex. Therefore, to make the problem easier we simply define another function that *does* contain the missing data. Let $\mathbf{Z}$ be the complete set of memberships z_i for all individuals, and notate the set of all unobservable $\boldsymbol{\beta}_i$ as $\boldsymbol{\beta}$. The *complete-data* MAP objective function of σ^2 and Φ takes the form

$$\begin{aligned} \mathcal{M}_C(\sigma^2, \Phi) &= \log\left[p(Y, \mathbf{Z}, \boldsymbol{\beta}|X, \sigma^2, \Phi)p(\Phi)\right] \\ &= \log\left[p(Y|X, \Theta)p(\Theta, \mathbf{Z}|\Phi)p(\Phi),\right] \end{aligned}$$

with

$$\begin{aligned} p(Y|X, \Theta) &= \prod_i \mathcal{N}(\mathbf{y}_i|\mathbf{X}_i\boldsymbol{\beta}_i, \sigma^2\mathbf{I}), \\ p(\Theta, \mathbf{Z}|\Phi) &= \prod_i \alpha_{z_i}\mathcal{N}(\boldsymbol{\beta}_i|\mu_{z_i}, \mathbf{R}_{z_i}), \end{aligned}$$

and

$$p(\Phi) = D(\alpha|\eta)\prod_k W(\mathbf{R}_k^{-1}|\mathbf{R}_0, \nu).$$

Notice that the logarithm of the summation has been removed due to the move from $p(\Theta|\Phi)$ to $p(\Theta, \mathbf{Z}|\Phi)$. The true values for $\mathbf{Z}$ and $\boldsymbol{\beta}$ aren't known, so expectations with respect to their joint posterior is taken in their place.

The remaining derivation for the EM algorithm consists of two steps: (1) the expected value of $\mathcal{M}_C$ is taken with respect to the posterior *hidden distribution* $p(\mathbf{Z}, \boldsymbol{\beta}|Y)$, and (2) this expectation is maximized over the parameters σ^2 and Φ to yield the new parameter values.

E-Step

The posterior $p(\mathbf{Z}, \boldsymbol{\beta}|Y)$ factors into $p(\mathbf{Z}|Y)p(\boldsymbol{\beta}|\mathbf{Z}, Y)$. This leaves two factors to be calculated: the *membership* probability $p(z_i = k|\mathbf{y}_i)$, and the expected value of the posterior $p(\boldsymbol{\beta}_i|z_i, \mathbf{y}_i)$. First, we calculate the *membership* probability

$$\begin{aligned} w_{ik} &= p(z_i = k|\mathbf{y}_i) \\ &\propto \alpha_k \, p(\mathbf{y}_i|\sigma^2, \phi_k) \end{aligned} \tag{4.4}$$

that curve i was generated from cluster k. Note that we are not given $\boldsymbol{\beta}_i$ in $p(\mathbf{y}_i|\sigma^2, \phi_k)$; this is the marginal model of $\mathbf{y}_i$. Second, we set the expected value of $\boldsymbol{\beta}_i$ given $\mathbf{y}_i$ and $z_i = k$ to

$$\hat{\boldsymbol{\beta}}_{ik} = (1/\sigma^2 \mathbf{X}_i' \mathbf{X}_i + \mathbf{R}_k^{-1})^{-1} (1/\sigma^2 \mathbf{X}_i' \mathbf{y}_i + \mathbf{R}_k^{-1} \mu_k)$$

which is the mean of the posterior $p(\boldsymbol{\beta}_i|\mathbf{y}_i, z_i = k)$. Note that $\hat{\boldsymbol{\beta}}_{ik}$ is simply the result of Bayesian regression with prior μ_k. Also, for simplicity, we set

$$\mathbf{V}_{\hat{\beta}_{ik}} = (1/\sigma^2 \mathbf{X}_i' \mathbf{X}_i + \mathbf{R}_k^{-1})^{-1}$$

which gives the posterior covariance.

M-Step

In the M-step, we use the results from the E-step to update the model parameters. First, we maximize the top-level (the mixture model on parameters), and then we maximize the bottom-level (the regression model on y and x data). For the top-level

we update the parameters

$$\begin{aligned}\hat{\alpha}_k &= \frac{\sum_i^n w_{ik} + (\eta_k - 1)}{n + (\sum_k \eta_k - K)}, \\ \hat{\mu}_k &= \frac{\sum_i^n w_{ik}\hat{\boldsymbol{\beta}}_i}{\sum_i^n w_{ik}},\end{aligned}$$

and

$$\hat{\mathbf{R}}_k = \frac{\sum_i^n w_{ik}\left[\left\|\hat{\boldsymbol{\beta}}_i - \hat{\mu}_k\right\|^2 + \mathbf{V}_{\hat{\beta}_{ik}}\right] + \mathbf{R}_0^{-1}}{\sum_i^n w_{ik} + (\nu - (p+1))}, \tag{4.5}$$

while on the bottom-level we update the parameter

$$\hat{\sigma^2} = \frac{\sum_{ik} w_{ik}\left[\left\|\mathbf{y}_i - \mathbf{X}_i\hat{\boldsymbol{\beta}}_i\right\|^2 + \mathbf{V}_{\hat{\beta}_{ik}}\right]}{N}$$

where $N = \sum_i n_i$.

There is one small issue of setting the hyperparameters for the hyperprior $p(\Phi)$. In practice we set ν to the neutral value of $p+1$ so that it cancels in the denominator of (4.5). We also set $\mathbf{R}_0^{-1}$ to $\omega\mathbf{I}$ for some positive ω. In this way, ω acts as a type of smoothing parameter. Unless otherwise stated, we also set the Dirichlet to neutral values (e.g., $\eta_1 = \cdots = \eta_k = 1$); however, it can be used to deal with issues such as background clusters. In this case, you may want to enforce a rule that for every 100 curves, there "should" be at least one in the background.

The computational complexity of this EM algorithm is still linear in the number of data points as with PRM and SRM. Initialization is carried out by sampling random values for the membership weights w_{ik}, and then solving for $\hat{\boldsymbol{\beta}}_{ik}$ by removing the terms involving $\mathbf{R}$ and μ_k. The EM algorithm can then be started with the M-step. Convergence is detected in the same manner as with PRM, SRM, and KRM; by monitoring the log-likelihood until the incremental improvement drops below a

threshold value (see Section 3.3.3 for the exact criterion as described for PRMs).

4.5 Experimental results

Figure 4.1 shows simulated curve data generated from the hierarchical model structure described in Section 4.3. Underlying the data are three different quadratic polynomials each allowing heterogeneity among the curves common to its group. The curves were generated by choosing cluster k with probability α_k, drawing $\boldsymbol{\beta}_i$ from $\mathcal{N}(\boldsymbol{\beta}_i|\mu_k, \mathbf{R}_k)$, and then producing $\mathbf{y}_i$ from $\mathcal{N}(\mathbf{y}_i|\mathbf{X}_i\boldsymbol{\beta}_i, \sigma^2\mathbf{I})$.

The left plot of Figure 4.1 shows the generated curve data with no classification labels. The right plot shows a view of parameter space. In this plot, the horizontal axis gives the value of $\boldsymbol{\beta}_{i0}$ (the y-intercept) while the vertical axis gives $\boldsymbol{\beta}_{i2}$ (the coefficient of x^2). The variance in parameter space can be seen from this view.

This data was given to RERM and was set to find three groups. The results are shown in Figure 4.2. The left plot of Figure 4.2 shows the data as clustered/classified by the mixture model. The clustering is shown as solid, dashed, and dashed-dotted curve groups. The mean curve for each group is shown by bolded lines. The right plot shows the clustered data in parameter space. As with the previous plot, the symbols for the means of the groups are bolded/filled-in.

Figure 4.3 gives the actual or true clustering from the generated data. You can see that the hierarchical model was able to find the means and the variance both at the bottom-level and the top-level of the structure.

It is instructive to understand how a standard PRM deals with this dataset. The results from this test are shown in Figure 4.4. There is no plot for the top level in this figure since this model does not have any notion of a hierarchy. However, the linear regression model spreads the mean curves out across the mass of data so that

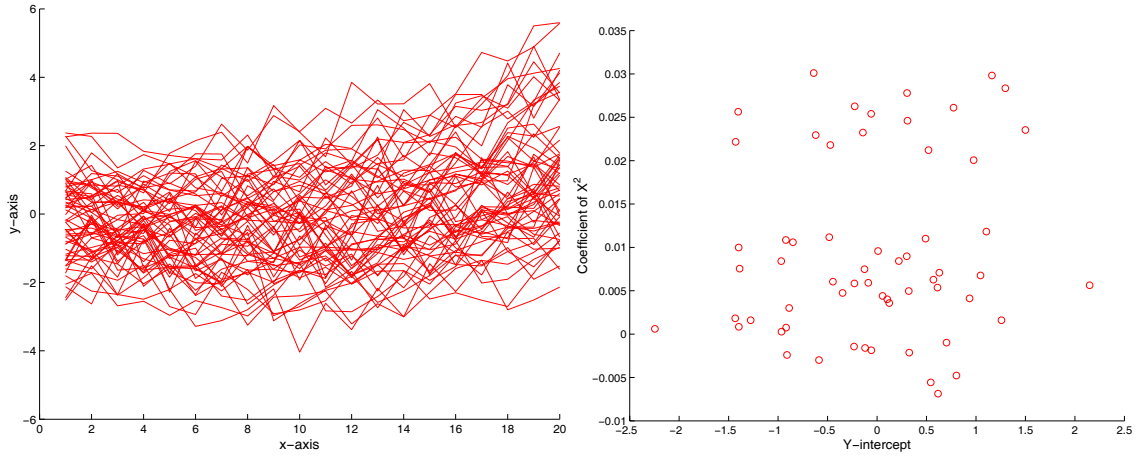


Figure 4.1: Simulated data from three underlying quadratic polynomials. The left plot shows the data-level and the right plot shows a view of the parameter-level.

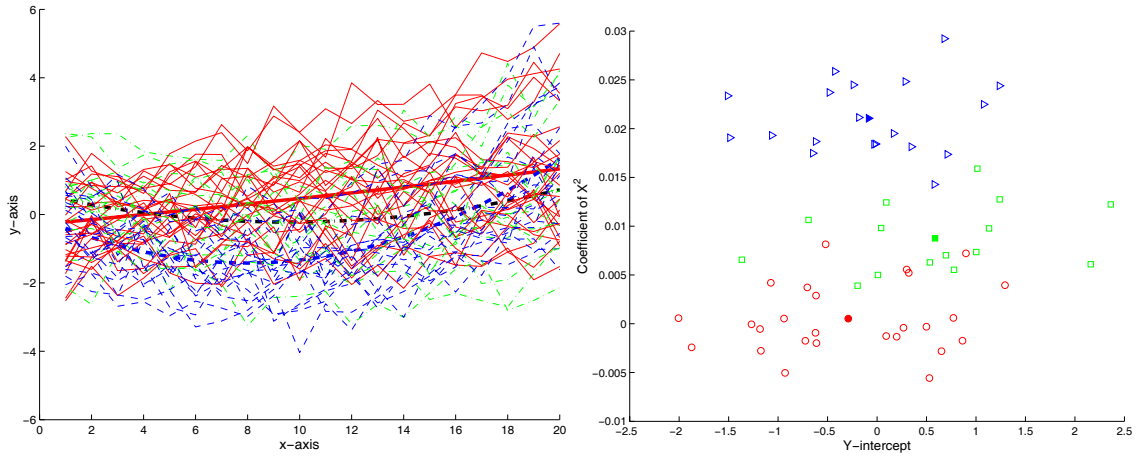


Figure 4.2: Clustering results with random effects regression mixtures. The "means" are shown in bold.

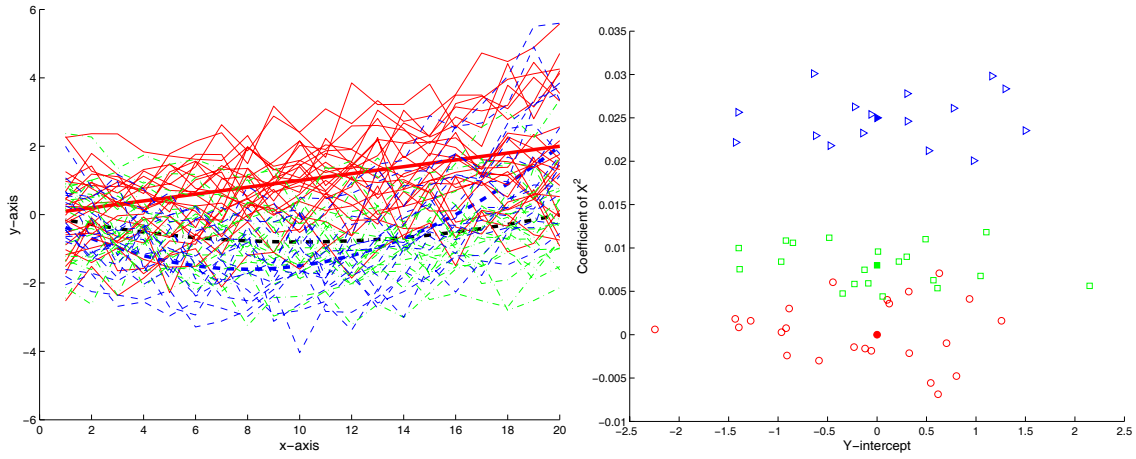


Figure 4.3: Simulated data from Figure 4.1 with the true class labels and underlying curves added.

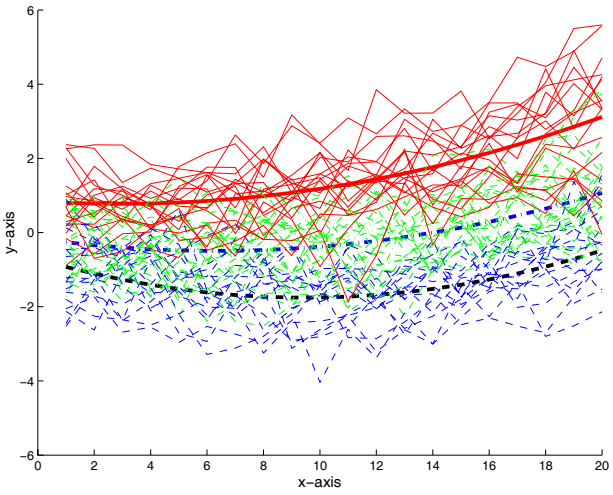


Figure 4.4: Clustering results with standard linear regression mixtures. Note there is no notion of a parameter level with this model.

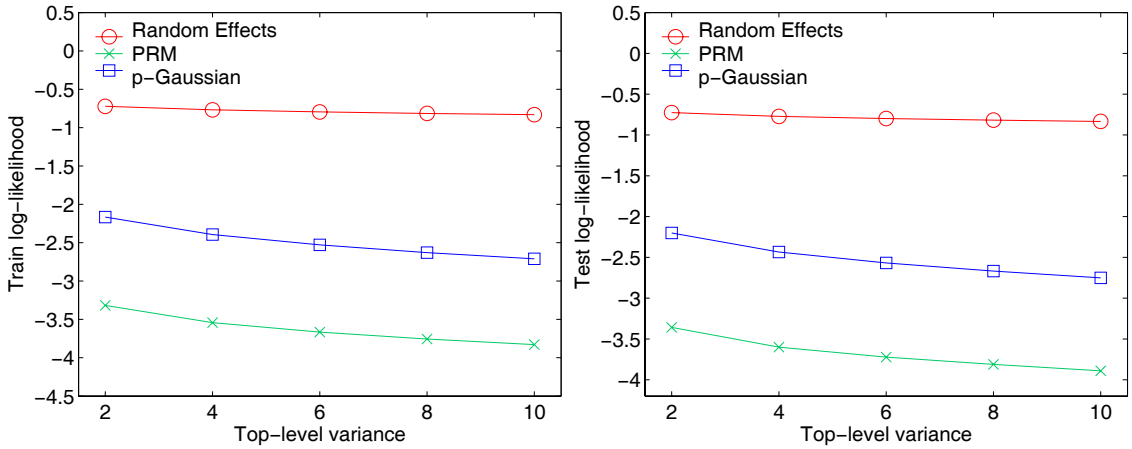


Figure 4.5: Comparisons between random effects linear regression mixtures, linear regression mixtures, and p-Gaussian mixtures. The left plot shows the training log-likelihood and the right plot shows the test log-likelihood.

it can account for all of the variance at the bottom-level in the most likely fashion. Of course, this is not the state of nature, but it is doing the best it can in this case. Absent the ability to model the variance in some other way, it must incorrectly infer the placement of the underlying groups of behavior.

One thing we might want to test is the comparison between random effects re-

gression mixtures and the method of fitting a separate regression line of order $p-1$ to each curve in your data set and then use straight Gaussian mixtures to cluster the fitted regression coefficients, and thereby the original curve data. We call this the p-Gaussian method since one firsts projects the original curve data down into a p-dimensional vector space using a polynomial fit of order $p-1$ and then applies Gaussian mixtures in this p-dimensional space.

Conceptually, we can think of random effects regression mixtures as doing this in an iterative fashion, where at each iteration the bottom-level uses the results from the top-level to perform the projection, and the top-level uses the results from the bottom-level to perform the clustering. In this way the method attempts to utilize information from both levels in the most effective manner.

Figure 4.5 shows the results of comparison tests between the $p-$Gaussian method, RERM, and PRM. For the experiments, twenty-five training and test data sets were randomly generated from a hierarchical model structure at each of five different variance levels, resulting in 250 different data sets. The top-level of the model consisted of a mixture on the coefficients for three underlying linear polynomials. The covariance matrix was set as diagonal and shared by all three clusters, with the diagonal elements being the measure of variance that was changed during data generation. The variance at the data-level was fixed during the generation of the training and test sets.

Each of the three clustering methods were presented with the training and test sets, and the resulting training and test log-likelihood scores per data point were recorded. In Figure 4.5 we see that the random effects linear regression model consistently beat the other two over the testing interval in both training and test log-likelihood scores. Again we see evidence that standard linear regression mixtures cannot cope with the variance structure in these types of data sets as it gets beat

by both p-Gaussian mixtures and RERM.

4.6 Summary

In this chapter, we introduced a novel extension to the regression mixtures framework of the previous chapter. This extension allows for the modelling of significant within-cluster variability without resorting to the undesirable solution of arbitrarily increasing the number of clusters to provide a better model fit. The framework can be defined in a natural way as a two-level hierarchical model in which the clustering of individuals occurs at the top-level and the modelling of the output data from the individuals occurs at the bottom-level. The definition of the hierarchy leads to an efficient MAP-based EM algorithm for the estimation of both the individual-specific model parameters and the top-level hyperparameters which describe the cluster-specific variability.

Simulated data experiments were presented that showed the effectiveness of the two-level hierarchy for the modelling of clustered data with significant within-cluster variability. This variability is well described through the use of top-level distributions on individual-specific regression parameters. Reported results showed that the PRM is particularly bad at modelling this type of two-level data. It was out-performed by a projection-based Gaussian mixtures method in which the underlying dataset was first projected into the parameter space in which Gaussian mixtures was run.

We leave this model aside in the rest of this thesis, and instead focus on the integration of the more basic regression mixtures of the previous chapter with the alignment models introduced in Chapters 5, 6, and 7. The interested reader is referred to Gaffney and Smyth (2003) for more detailed experimental results with RERMs.

Chapter 5

Curve Alignment in Measurement Space

In previous chapters, we looked at the clustering problem. In this and the following two chapters we set this problem aside and focus on the curve alignment problem. We unify both clustering and alignment in Chapter 8.

5.1 Introduction

It is common for sets of curves that result from a data measuring process to be misaligned from each other. This can be caused by factors such as incompatible measurement procedures or underlying differences in the curve generation process. In some instances an explicit alignment is not desired since the unaligned curves may provide useful information that is inherent to the process and may yield scientific interpretation. For example, signal delays in gene regulation networks provide clues to the underlying network structure. However, in other cases the search for an alignment is of primary importance; we shall assume this in what follows.

The main contribution of this chapter lies in the introduction of a novel probabilistic curve alignment model that allows for the alignment of curve data in measurement space. We refer to *alignment in measurement space* as allowing for transformations on the curve measurements themselves as opposed to the more difficult problem of allowing for transformations in time which is discussed separately in Chapter 6. Often we refer to alignment in measurement space as *alignment in space* or *space-alignment*.

Much of the previous work in curve alignment is founded in optimization theory. Often the resulting procedures are complex with many external constraints that help make the problem well-defined. In contrast, we formulate the problem from a probabilistic viewpoint that unifies the specification, learning, prediction, and constraint problems in a single, self-contained framework.

This chapters is organized as follows. We define the alignment problem and discuss the relation of our new methodology to previous work in Section 5.2. In Section 5.3 we introduce our space-translation alignment model that allows for translations in measurement space. We use this section to lay the foundation for the introduction of the rest of our alignment models described in this thesis.

In Section 5.4, the extension to the more complex affine alignment model which adds scaling in measurement space is derived. This introduces joint priors on translations and scalings, and can be described by a simple Bayesian network (Pearl, 1988; Jensen, 1996). Section 5.5 presents a set of experimental results with both real and simulated data that demonstrate the benefits gained with the probabilistic formulation. The chapter is concluded with a summary in Section 5.6.

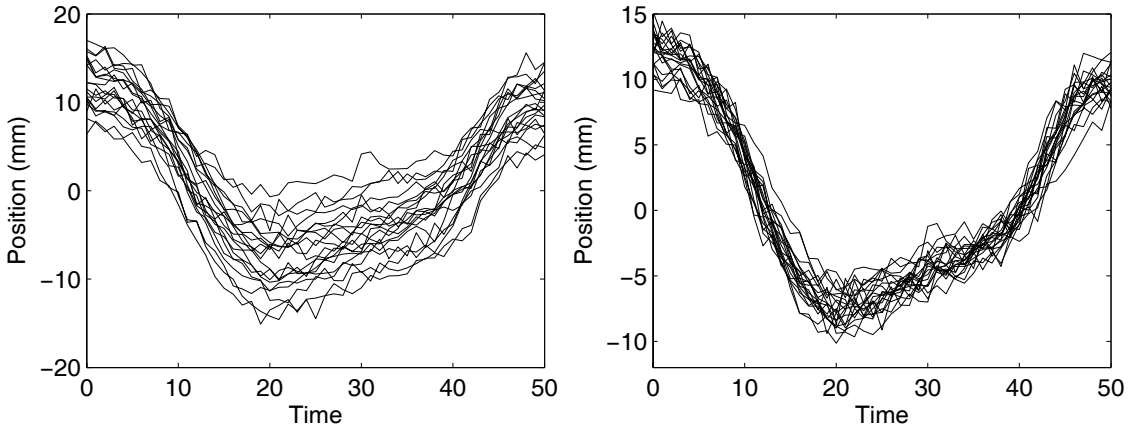


Figure 5.1: Estimated (and randomly translated) position of the center of the lower lip during the speaking of the syllable "bob"; (left) unregistered curves, (right) registered curves.

5.2 Problem definition and prior work

An example alignment problem is given in Figure 5.1 which shows simulated data motivated by experiments in speech modelling. The figure shows twenty curves that were simulated based on the "lip" data set analyzed in Ramsay and Silverman (1997). The experiment involved observing the position of the center of the lower lip at the moment the syllable "bob" was uttered.

The curves shown in the left plot are randomly translated versions of the original data (not shown) with noise added. The curves shown in the right plot are the registered versions of the curves that are obtained using our spline alignment model described below. It's clear from the figure that the underlying shape is more clearly expressed in the aligned curves.

The general curve alignment problem has active ongoing research in many fields under different names. For example, in statistics, self-modelling regression methods employ a parametric *shape invariant model* that allows for space- as well as time-alignment (Lawton et al., 1972; Kneip & Gasser, 1988). They use non-linear

curve models (e.g., non-linear regression models) to represent the curve data and develop iterative optimization procedures to solve for the alignments. A related method known as *structural averaging* also employs landmark points to aid in the alignment (Kneip & Engel, 1995).

Curve alignment is known as *curve registration* in functional data analysis (Silverman, 1995; Ramsay & Silverman, 1997; Ramsay & Li, 1998). Curves are represented as smooth functionals (e.g., using splines) and alignments are learned in an iterative fashion by minimizing integrated criteria in spline space (e.g., squared-error).

The method of dynamic time warping (DTW) grew out of speech recognition (Sakoe & Chiba, 1978) and specifically addresses the time-alignment problem. However, by defining transformation-invariant distance measures, alignment in measurement space can also be achieved (Keogh & Pazzani, 1998). DTW makes use of dynamic programming techniques to discover an alignment that minimizes the warping distance or cost (we will address DTW techniques in the next chapter on time-alignment).

Finally, *point-set matching* is important in medical imaging (among other applications) where 2D and 3D shapes are aligned within images (Goodall, 1991; Kendall, 1984; Neumann & Lorenz, 1998; Dryden & Mardia, 1998). Much of the work in this area is relevant to the joint problem of clustering and aligning. We discuss this work in Chapter 8 that addresses the joint problem.

In a broad sense, each of these methods contains a very specific iterative optimization routine to effect an alignment. In general, they iterate between estimating an alignment and updating a reference curve that represents the ideal to which the set should be aligned. This type of iterative procedure is common in multivariate statistics; it is loosely known as Procrustes (Mardia et al., 1979; Ramsay & Silverman, 1997). Many of the above methods also employ self-tailored normalization

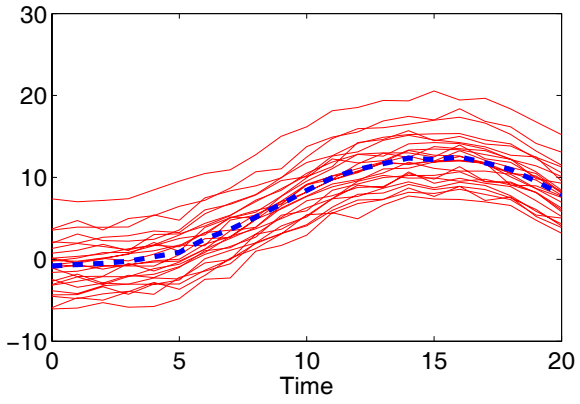


Figure 5.2: Example of cross-sectional mean curve.

procedures that address well-known identifiability issues (discussed below) with the alignment problem (see, e.g., Kneip & Gasser, 1988).

In contrast, we look at the problem from a probabilistic model-based viewpoint which unifies the specification, learning, prediction, and identifiability problems in a single, self-contained framework. This leads to an iterative EM algorithm that naturally encompasses the Procrustes procedure and leads to a simple-to-implement algorithm.

5.2.1 Curve preprocessing

Looking at the translation problem in Figure 5.1, the problem seems rather straightforward. For example, a similar alignment can be obtained by subtracting off the individual mean of each curve. In other words, the aligned curve $\mathbf{y}_i'$ is calculated as $\mathbf{y}_i' = \mathbf{y}_i - \sum_j y_{ij}/n_i$, for the curve of length n_i. This is a standard technique for normalizing a set of curves (i.e, conform in a standard way) before any further analysis is performed. However, this will cause all curves to artificially vary about zero. The underlying shape is well represented but the overall level has been shifted (which is important for curve prediction, among other things).

The cross-sectional mean curve, defined as $\mu = \sum_i \mathbf{y}_i / n_i$, represents the mean level of all curves at each point along the x-axis. The cross-sectional mean curve is represented in Figure 5.2 by the dashed (blue) line. An alignment that preserves the overall level of the curve dataset can be obtained by subtracting from each curve, the mean deviation of the curve from the cross-sectional mean curve. This results in the exact same shape as the first case except now the curves vary about the cross-sectional mean curve instead of zero. In other words, this procedure globally translates the first alignment to be centered over the cross-sectional mean curve. We refer to these types of "one-off" techniques as curve preprocessing or curve normalization.

We can attempt to further improve this alignment by modelling an overall mean by something other than the cross-sectional mean curve, and then calculate deviations from this new mean curve. In fact, we might refer to this as *model-based alignment* since the alignment is carried out with respect to an assumed model and not in relation to a simple *model-free* mean calculation. This is the strategy that is pursued in this thesis.

A useful feature of the alignment models proposed in this thesis is that they can be used in addition to prior curve preprocessing. In other words, a particular type of preprocessing can be employed to give a rough estimate of the curve alignments. Then, the application of our alignment models to this rough estimate can be used to give a more refined alignment. In Chapter 9, we show an application to cyclone clustering in which this is the optimal strategy. This procedure results in the implicit modification of the alignment priors as will be seen below.

5.3 Translations in space

We begin with an alignment model that allows for translations in measurement space. The exposition below defines the foundation upon which all our alignment models are based. We will follow a standard template for the description of new models throughout the rest of this thesis. The five-step procedure is as follows:

1. Provide the model definition
2. Define the transformation priors
3. Calculate the resulting joint and marginal probability models
4. Define the log-likelihood function
5. Derive the associated EM algorithm

5.3.1 Model definition

The derivation that follows equally applies to several types of regression models (e.g., in particular, both polynomial regression and spline regression models). However, we describe this section from the viewpoint of applying spline regression models to the problem. The alternative viewpoint will be taken in the next chapter with models for alignment in time.

We model a set of curves using the same spline regression model as defined in Section 3.4. The spline regression takes the form

$$\mathbf{y}_i = \mathbf{B}_i\boldsymbol{\beta} + \boldsymbol{\epsilon}_i, \qquad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}), \tag{5.1}$$

where $\mathbf{y}_i$ gives the i-th curve of length n_i, $\mathbf{B}_i$ is the associated $n_i \times L$ spline basis matrix, $\boldsymbol{\beta}$ is the $p \times 1$ *mean* spline coefficient vector, and $\mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I})$ gives the Gaussian noise model.

We add a curve-specific translation scalar d_i that allows for the entire curve to be translated as a unit. We incorporate this translation into the regression model as

$$\mathbf{y}_i = \mathbf{B}_i\boldsymbol{\beta} + d_i + \boldsymbol{\epsilon}_i, \qquad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}). \tag{5.2}$$

This results in a spline regression model that allows for arbitrary translations in measurement space. The associated curve probability model $p(\mathbf{y}_i|\theta, d_i)$ is conditioned both on the global parameters $\theta = \{\boldsymbol{\beta}, \sigma^2\}$ and also on the new unknown translation d_i (it is also implicitly conditioned on the non-random matrix $\mathbf{B}_i$). In order to fit the model to the available data, both θ and each of the d_i must be found.

Before we define the EM algorithm which performs this fitting task, we motivate the algorithm by quickly walking through a possible fitting procedure. The estimation is difficult because we have missing data. The problem would be straightforward if we knew the values of d_i; however, these values are hidden from the observer. Suppose we initially set each of the d_i to zero and then solve the regression equation in (5.2) for the ML estimate $\hat{\theta}$. This is a sensible thing to do; however, we are also in search of the values for the translation parameters themselves. Each $\hat{d}_i$ can be found as the solution to the associated minimization problem:

$$\hat{d}_i = \arg\min_{d_i} \left\| \mathbf{y}_i - \mathbf{B}_i\hat{\boldsymbol{\beta}} - d_i \right\|^2. \tag{5.3}$$

These solutions provide estimates for each of the parameters $\hat{\boldsymbol{\beta}}, \hat{\sigma}^2$ and $\hat{d}_i$. However, the estimate $\hat{\sigma}^2$ is incorrect since it includes variance from the translations that were not subtracted out of the regression (since we set all $d_i = 0$).

An improved estimate can be obtained by resolving the regression equation while substituting $\hat{d}_i$ for d_i. The new solution not only produces an improved $\hat{\sigma}^2$ but as a result a new estimate for $\hat{\boldsymbol{\beta}}$ is produced as well. These values can then be used to

set up a new minimization problem which again leads to a new regression equation and so forth and so on.

This demonstrates the use of a model-based Procrustes alignment strategy for curve data. As pointed earlier, the above estimation procedure seems to mirror the way in which the EM algorithm works. This similarity is exploited and the associated EM algorithm is derived in Section 5.3.2. The above iterative procedure is a non-probabilistic, model-based Procrustes alignment method. We use this type of method in Section 5.5.2 as a comparison to the full probabilistic EM approach. We give the name non-probabilistic Procrustes (NPP) to this type of procedure.

There is one more problem, however. The estimation in this case is not well-defined (i.e., the model is not identifiable). There are infinitely many valid solutions (in a maximum likelihood sense) for the values of the translation parameters $\{d_i\}$ and the global distribution parameters θ for any particular set of curves. It is easy to generate new solutions with the same likelihood as any current solution. First, pick any real number and add it to the current value of each d_i. Then, re-estimate the new value for $\boldsymbol{\beta}$ with the set $\{d_i\}$ fixed. The mean curve will simply be translated by the chosen real number added to the d_i, the variance will remain unchanged, and most importantly, the likelihood will also remain unchanged.

The problem is that there are no restrictions on the translations themselves. Borrowing a term from machine learning, the method does not employ an *inductive bias* over translations (Mitchell, 1997). Many different normalization schemes have been used to deal with this predicament. For example, Härdle and Marron (1990) chose the first curve as the reference curve and fixed its alignment parameters to some particular values (e.g., in this case, we might fix d_1 to zero) which pins down a particular "frame of reference". Other schemes require that the mean of the set $\{d_i\}$ be zero which achieves the same sort of goal. This problem is handled automatically

in our framework through the use of alignment priors, which we now turn to.

Prior model: $p(d_i)$

In the preceding example, we initially estimated θ by setting the d_i to zero. This initialization makes a couple of implicit assumptions: (1) the most likely translation is the zero translation, and (2) negative or positive translations are equally likely. We can make these assumptions explicit by considering d_i to be a random variable with an associated prior probability density attached to it. A useful prior model for the random variable d_i is the zero-mean Gaussian:

$$p(d_i) = \mathcal{N}(d_i|0, v^2). \tag{5.4}$$

This encompasses the two implicit assumptions above and also discounts large translations over smaller ones. The variance v^2 determines the degree to which larger translations are discounted. This variance will be learned from the data within the ensuing EM algorithm.

Specifying the transformation parameters as random variables is done in Kneip and Gasser (1988), but they do this for asymptotic reasons only (they do not actually employ priors in the algorithm). Rønn (2001) specifies priors on time shifts, as we will see in the next chapter, but he does not use them to develop a full EM algorithm.

Integration of this prior into the joint probability density for curve $\mathbf{y}_i$ and translation d_i leads naturally to identifiable estimation procedures. We discus this integration after presenting the graphical model structure associated with this model.

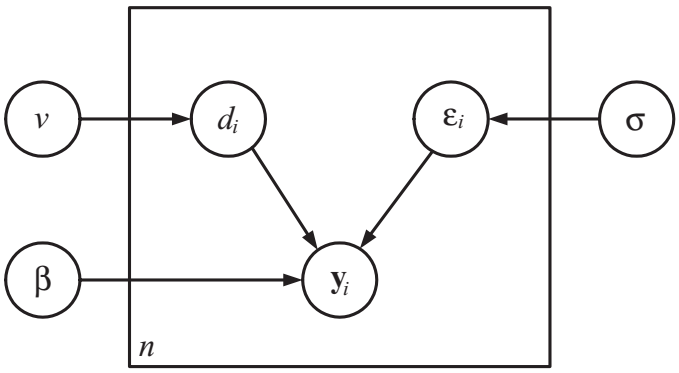


Figure 5.3: Graphical model structure for the specified alignment model. The use of plates allows for the explicit notation of the curve samples and the model parameters.

Plate representation of the model structure

Figure 5.3 shows a diagram of the graphical model structure for this problem using plates (Buntine, 1994). Plates are an extension of standard graphical models that explicitly represent the data samples and the parameters as nodes in the graph. The nodes "inside" of the box (or plate) in the figure are thought of as being repeated n times (consisting of a stack of n plates), representing the n curve samples. This is denoted by the n in the lower-left corner of the ("top") plate. The nodes outside of the plate have only one instance each, but directly affect the nodes of the n plates to which they point. This means each of their arcs are also repeated n times.

The plate structure clearly describes the set of priors that are employed (and not employed) for this alignment model. Notice that only variables are located in the plate and only parameters are located outside. The model does not use hyperpriors. In particular, notice that there are no incoming arcs to any of the parameters outside of the plate. One can easily employ hyperpriors in this model if the need arises. For example, the conjugate hyperprior for the prior parameter v^2 is the inverse gamma distribution (Gelman et al., 1995).

Joint, marginal, and log-likelihood

The model specification in (5.2) results in the conditional probability density for $\mathbf{y}_i$ as

$$p(\mathbf{y}_i|d_i) = \mathcal{N}(\mathbf{y}_i|\mathbf{B}_i\boldsymbol{\beta} + d_i, \sigma^2\mathbf{I}). \tag{5.5}$$

If we assume that d_i is a random variable with the prior specified in (5.4), then the joint probability density for $\mathbf{y}_i$ and d_i takes the product form

$$\begin{aligned} p(\mathbf{y}_i, d_i) &= p(\mathbf{y}_i|d_i)p(d_i) \\ &= \mathcal{N}(\mathbf{y}_i|\mathbf{B}_i\boldsymbol{\beta} + d_i, \sigma^2\mathbf{I})\mathcal{N}(d_i|0, v^2). \end{aligned} \tag{5.6}$$

The joint density contains complete information about the model from which all other densities can be derived. For example, we can integrate over d_i to obtain the marginal density for $\mathbf{y}_i$. This density can be calculated as follows.

$$\begin{aligned} p(\mathbf{y}_i) &= \int p(\mathbf{y}_i, d_i)\, dd_i \\ &= \int p(\mathbf{y}_i|d_i)\; p(d_i)\, dd_i \\ &= \int \mathcal{N}(\mathbf{y}_i|\mathbf{B}_i\boldsymbol{\beta} + d_i, \sigma^2\mathbf{I})\; \mathcal{N}(d_i|0, v^2)\, dd_i \\ &= \mathcal{N}(\mathbf{y}_i|\mathbf{B}_i\boldsymbol{\beta}, v^2\mathbb{1} + \sigma^2\mathbf{I}), \end{aligned} \tag{5.7}$$

where $\mathbb{1}$ notates an $n_i \times n_i$ matrix of ones. The marginal density on the left is implicitly conditioned on the parameters $\theta = \{\boldsymbol{\beta}, \sigma^2\}$ and on the non-random matrix $\mathbf{B}_i$ as is always the case in this thesis (unless otherwise stated). The reader should not confuse this marginal density with the fully Bayesian marginal in which θ has been integrated out of the model.

The marginal density for $\mathbf{y}_i$ has a non-diagonal covariance matrix. The marginal

model that results from the standard regression equation in (5.1) necessitates independence among the curve measurements of $\mathbf{y}_i$. However, by considering d_i as a random variable we have added non-zero covariance between the curve measurements themselves. This is due to the fact that d_i affects the entire curve. The variance of any particular y_{ij} is $v^2 + \sigma^2$, and the covariance between y_{ij} and y_{ik} is v^2 with a correlation of $v^2/(v^2 + \sigma^2)$.

This density leads directly to the definition of the log-likelihood for the set $Y = \{\mathbf{y}_i\}_1^n$ of n curves. The log-likelihood is the sum over all n curves of the log marginal of $\mathbf{y}_i$:

$$\log p(Y) = \sum_i \log \mathcal{N}(\mathbf{y}_i | \mathbf{B}_i \boldsymbol{\beta}, v^2 \mathbb{1} + \sigma^2 \mathbf{I}). \tag{5.8}$$

It is this function which we attempt to maximize using the EM algorithm developed next.

5.3.2 EM translation algorithm

In this section, we derive the new EM space-translation algorithm. A review of the necessary prerequisite EM theory is provided in Appendix A. The derivation covers four steps. First, we specify the hidden or missing data and define the *hidden-data* density (the posterior of the hidden data given curve $\mathbf{y}_i$). Second, we define the complete-data log-likelihood function $\mathcal{L}_c$ which is the joint log-likelihood of Y and the hidden data. Third, we calculate the Q-function by taking the expectation of $\mathcal{L}_c$ (w.r.t. the hidden-data density). And finally, we derive the parameter re-estimation equations by maximizing the Q-function. The first two steps are initial specification steps, while step 3 is the E-step, and step 4 is the M-step.

In the first step, we declare the hidden data as the translations $\{d_i\}$ since these data cannot be directly observed. The posterior $p(d_i|\mathbf{y}_i)$, then, represents the hidden-

data density (giving us a distribution on the values for the unknown translations).

In the second step we define the complete-data log-likelihood function as the joint log-likelihood of Y and the hidden data $\{d_i\}$. This can be written as the sum over all n curves of the log joint density in (5.6). This function takes the form

$$\begin{aligned} \mathcal{L}_c &= \sum_i \log p(\mathbf{y}_i|d_i)p(d_i) \\ &= \sum_i \log \mathcal{N}(\mathbf{y}_i|\mathbf{B}_i\boldsymbol{\beta} + d_i, \sigma^2\mathbf{I})\, \mathcal{N}(d_i|0, v^2). \end{aligned} \tag{5.9}$$

The EM algorithm will iterate between calculating the posterior $p(d_i|\mathbf{y}_i)$ in the E-step and calculating new parameter estimates in the M-step. We address these last two steps next.

E-step

In the E-step we first calculate the posterior $p(d_i|\mathbf{y}_i)$ and then use this to take expectations of the complete-data log likelihood function in (5.9). The posterior can be calculated analytically in this case. Upon expansion of the posterior

$$\begin{aligned} p(d_i|\mathbf{y}_i) &\propto p(\mathbf{y}_i|d_i)p(d_i) \\ &\propto \exp\left\{ - \left\|\mathbf{y}_i - \mathbf{B}_i\boldsymbol{\beta} - d_i\right\|^2 / 2\sigma^2 - d_i^2/2v^2 \right\}, \end{aligned}$$

we recognize this as the normal density $\mathcal{N}(d_i|\hat{d}_i, V_{d_i})$ since we have an exponential function of a quadratic polynomial in d_i. The mean can be identified as

$$\hat{d}_i = \frac{V_{d_i}}{\sigma^2}(\mathbf{y}_i - \mathbf{B}_i\boldsymbol{\beta})'\mathbf{1} \tag{5.10}$$

with a variance of

$$V_{d_i} = (n_i/\sigma^2 + 1/v^2)^{-1}. \tag{5.11}$$

From this we see the posterior mean $\hat{d}_i$ is the weighted difference between the actual curve and the model-based mean curve (which plays the role of the cross-sectional mean here). This difference is down-weighted as the noise model grows (increase in σ^2) or as the *likely* range of translations shrinks (decrease in V_{d_i}). Thus, we find that the probabilistic framework naturally produces a weighted form of the basic normalization technique described at the beginning of Section 5.3 (i.e., set d_i to the mean of the difference between the curve and the cross-sectional mean curve). The variance equation is the inverse of the sum of the two relevant precisions, a common result in Bayesian estimation (Gelman et al., 1995).

Calculating the Q-function

We now turn to the calculation of the Q-function as the posterior expectation of $\mathcal{L}_c$ from (5.9) with respect to the hidden-data density calculated above. We will use this function in the M-step to derive the parameter re-estimation equations. The expectation can be calculated by expanding the normal densities and substituting in the posterior mean $\hat{d}_i$ for $\mathrm{E}[d_i|\mathbf{y}_i]$ and $(\hat{d}_i^2 + V_{d_i})$ for $\mathrm{E}[d_i^2|\mathbf{y}_i]$:

$$\begin{aligned} Q &= \sum_i \mathrm{E}\left[\log \mathcal{N}(\mathbf{y}_i|\mathbf{B}_i\boldsymbol{\beta} + d_i, \sigma^2\mathbf{I})\, \mathcal{N}(d_i|0, v^2)\Big|\, \mathbf{y}_i\right] \\ &= \sum_i -\frac{n_i}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}\, f(\hat{d}_i|\mathbf{y}_i) - \frac{1}{2}\log 2\pi v^2 - \frac{1}{2v^2}\, g(\hat{d}_i) \end{aligned} \tag{5.12}$$

where

$$\begin{aligned} f(\hat{d}_i|\mathbf{y}_i) &= \mathrm{E}\left[\, \|\mathbf{y}_i - \mathbf{B}_i\boldsymbol{\beta} - d_i\|^2 \Big|\, \mathbf{y}_i\right] \\ &= \left\|\mathbf{y}_i - \mathbf{B}_i\boldsymbol{\beta} - \hat{d}_i\right\|^2 + n_i V_{d_i}, \end{aligned} \tag{5.13}$$

and

$$\begin{aligned} g(\hat{d}_i) &= \mathrm{E}\left[d_i^2 \middle| \mathbf{y}_i \right] \\ &= \hat{d}_i^2 + V_{d_i}. \end{aligned} \tag{5.14}$$

M-step

The last step that must be defined is the M-step. In the M-step we maximize the Q-function in (5.12) over the set of parameters $\{\boldsymbol{\beta}, \sigma^2, v^2\}$. The maximization is straightforward and resembles the standard least squares solution for regression. The parameter re-estimation equations are

$$\hat{v}^2 = 1/n \sum_i g(\hat{d}_i), \tag{5.15}$$

$$\hat{\sigma}^2 = 1/N \sum_i \hat{f}(\hat{d}_i), \tag{5.16}$$

and

$$\hat{\boldsymbol{\beta}} = \left[\sum_i \mathbf{B}_i' \mathbf{B}_i \right]^{-1} \sum_i \mathbf{B}_i' (\mathbf{y}_i - \hat{d}_i), \tag{5.17}$$

where $\hat{f}$ is the function f with $\boldsymbol{\beta}$ replaced by $\hat{\boldsymbol{\beta}}$.

An NPP alignment procedure (defined in Section 5.3.1) for this model consists of calculating $\hat{\boldsymbol{\beta}}$ exactly as it is here (assuming that you replace the cross-sectional mean curve with $\mathbf{B}_i\boldsymbol{\beta}$, i.e., a model-based NPP). However, the equation for $\hat{\sigma}^2$ would not contain the $n_i V_{d_i}$ term and there would be no calculation for $\hat{v}^2$ at all (since this is not present in a non-random approach).

Further details

A typical initialization for this algorithm consists of sampling random values for $\hat{d}_i$ and V_{d_i}, and then proceeding directly to the M-step. The computational complexity of this algorithm is linear in the total number of points $N = \sum_i n_i$.

We provide experimental results that measure the performance of this algorithm in Section 5.5. This framework is now used to extend the methodology to allow for translations as well as scaling in measurement space.

5.4 Affine transformations in space

In this section we focus on extending the methodology of the previous section to include translations as well as possible scaling in measurement space. The derived extension demonstrates the flexibility of the probabilistic framework.

5.4.1 Model definition

We turn to a graphic example of the affine alignment problem using simulated data. Figure 5.4 shows twenty curves which were simulated from the "pinch" data set analyzed in Ramsay and Silverman (1997). The experiment involved measuring the exerted force between the thumb and forefinger during a brief impulse of pinching. The curves in the left plot are randomly transformed versions of the original data (not shown). The curves in the right plot are the registered versions output from our affine-alignment spline model introduced below. Note that the shape as well as the overall mean-level are well represented in the aligned curves.

The model definition is an extension of the spline regression model in (5.2). We add a new scale parameter c_i to this model which results in the following affine-

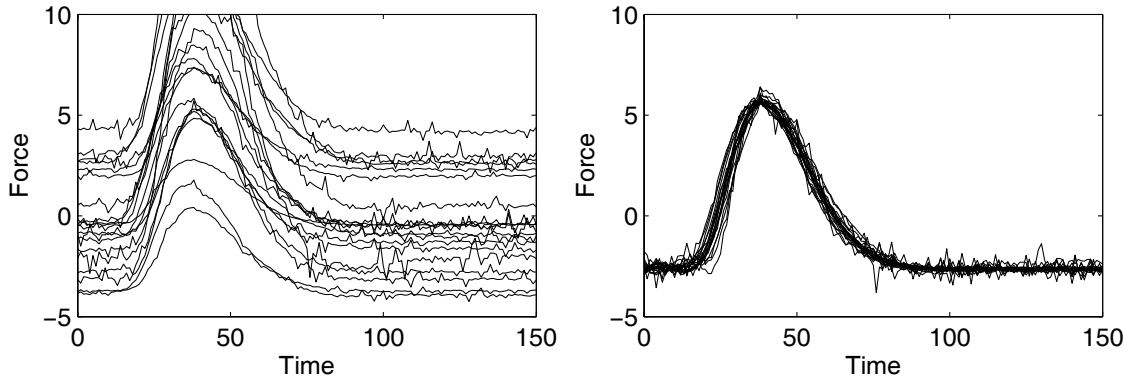


Figure 5.4: Curves measuring the force exerted between the forefinger and thumb; (left) randomly transformed versions, (right) registered curves.

alignment spline regression model:

$$\mathbf{y}_i = c_i \mathbf{B}_i \boldsymbol{\beta} + d_i + \boldsymbol{\epsilon}_i, \qquad \boldsymbol{\epsilon}_i \sim N(\mathbf{0}, \sigma^2 \mathbf{I}). \tag{5.18}$$

This spline model now allows for arbitrary translation and scaling in measurement space. The associated curve probability density $p(\mathbf{y}_i|\theta, c_i, d_i)$ is conditioned both on the global parameters $\theta = \{\boldsymbol{\beta}, \sigma^2\}$ and also on the unknown transformation variables $\{c_i, d_i\}$.

The same identifiability problem must be dealt with for this model as in the previous modelling case in Section 5.3.1. This problem is handled automatically in our framework in the same manner as before, through the use of priors on the possible set of transformations.

Prior model: $p(c_i, d_i)$

We begin by making an initial independence assumption for the joint prior model. That is, given no other information, we set $p(c_i, d_i) = p(c_i)p(d_i)$. This is a natural assumption since given no data, we have no reason to believe c_i covaries with d_i

unless the specific situation calls for it.

Given our independence assumption, the problem is reduced to specifying two separate priors. The translation prior was specified earlier in Equation (5.4) and we use the same prior here. For the scaling prior, it is desirable that a value of 1 be the most likely value (i.e., no scaling at all) with successive values decreasing in likelihood. Again an extremely useful prior for this job is the Gaussian density. In other words, we set

$$p(c_i, d_i) = \mathcal{N}(c_i|1, u^2)\mathcal{N}(d_i|0, v^2), \tag{5.19}$$

where u^2 gives the variance for the prior model on c_i, and v^2 gives the variance for d_i as before.

This scale prior technically allows for negative scaling (and is also symmetric about one). In practical terms, the ensuing EM algorithm won't allow for negative scaling unless the data set actual contains it. Other priors which prevent negative scaling values can be used. A log-normal density is an example of this type of prior. However, preliminary results did not show any significant improvements over the more analytically friendly Gaussian priors, and thus we use the prior defined in (5.19) for the results in this thesis.

Integration of this prior into the joint probability density for curve $\mathbf{y}_i$ and transformation parameters $\{c_i, d_i\}$ leads naturally to identifiable estimation procedures. The corresponding plate structure for the affine alignment model is shown in Figure 5.5. There are two extra nodes for this model. The unobservable scaling variable c_i and its associated variance prior parameter u^2.

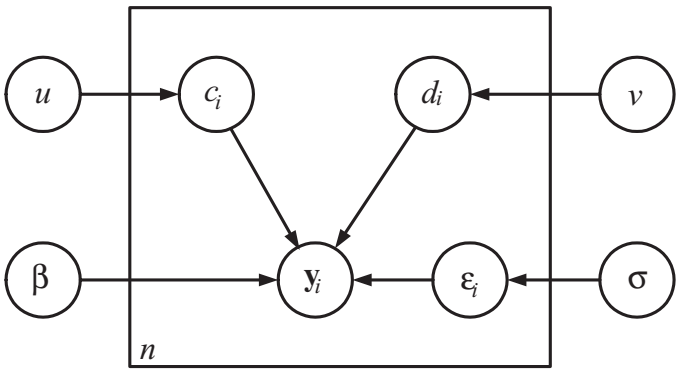


Figure 5.5: Graphical model structure for the affine alignment model.

Joint, marginals, and log-likelihood

The model specification in (5.18) results in the conditional probability density for $\mathbf{y}_i$ as

$$p(\mathbf{y}_i|c_i, d_i) = \mathcal{N}(\mathbf{y}_i|c_i\mathbf{B}_i\boldsymbol{\beta} + d_i, \sigma^2\mathbf{I}). \tag{5.20}$$

If we consider both c_i and d_i as random variables with prior density (5.19), then the joint probability density for the curve $\mathbf{y}_i$ and the transformation parameters $\{c_i, d_i\}$ takes the product form

$$\begin{aligned} p(\mathbf{y}_i, c_i, d_i) &= p(\mathbf{y}_i|c_i, d_i)p(c_i)p(d_i) \\ &= \mathcal{N}(\mathbf{y}_i|c_i\mathbf{B}_i\boldsymbol{\beta} + d_i, \sigma^2\mathbf{I})\mathcal{N}(c_i|1, u^2)\mathcal{N}(d_i|0, v^2). \end{aligned} \tag{5.21}$$

We can integrate over each of the transformation parameters in the joint model to obtain all of the marginal densities. For example, the marginal of $\mathbf{y}_i$ conditioned on c_i is

$$\begin{aligned} p(\mathbf{y}_i|c_i) &= \int p(\mathbf{y}_i, d_i|c_i)\, dd_i \\ &= \mathcal{N}(\mathbf{y}_i|c_i\mathbf{B}_i\boldsymbol{\beta}, \mathbf{V}), \qquad \mathbf{V} = v^2\mathbb{1} + \sigma^2\mathbf{I}. \end{aligned}$$

Note the covariance terms in $\mathbf{V}$ are not zero but are equal to v^2 while the variances are $v^2 + \sigma^2$. The covariance $\mathbf{V}$ contains all model variance except that associated with the conditioned variable c_i. For the marginal of $\mathbf{y}_i$ conditioned on d_i we get

$$\begin{aligned} p(\mathbf{y}_i|d_i) &= \int p(\mathbf{y}_i, c_i|d_i)\, dc_i \\ &= \mathcal{N}(\mathbf{y}_i|\mathbf{B}_i\boldsymbol{\beta} + d_i, \mathbf{U}), \qquad \mathbf{U} = u^2\mathbf{B}_i\boldsymbol{\beta}\boldsymbol{\beta}'\mathbf{B}_i' + \sigma^2\mathbf{I}, \end{aligned}$$

where the covariance terms are essentially cross-products of $\mathbf{B}_i\boldsymbol{\beta}$ weighted by u^2. The covariance $\mathbf{U}$ contains all model variance except that associated with the conditioned variable d_i.

The unconditional marginal of $\mathbf{y}_i$ can be calculated by integrating over both c_i and d_i. Again, we can calculate this analytically as

$$\begin{aligned} p(\mathbf{y}_i) &= \int\int p(\mathbf{y}_i, c_i, d_i)\, dc_i dd_i \\ &= \mathcal{N}(\mathbf{y}_i|\mathbf{B}_i\boldsymbol{\beta}, \mathbf{U} + \mathbf{V} - \sigma^2\mathbf{I}) \end{aligned}$$

Note that $\mathbf{U} + \mathbf{V} - \sigma^2\mathbf{I}$ contains all of the model variance. This marginal density then leads directly to the definition of the log-likelihood for the set $Y = \{\mathbf{y}_i\}_1^n$ of n curves. The log-likelihood is the sum over all n curves of the log marginal of $\mathbf{y}_i$:

$$\log p(Y) = \sum_i \log \mathcal{N}(\mathbf{y}_i|\mathbf{B}_i\boldsymbol{\beta}, \mathbf{U} + \mathbf{V} - \sigma^2\mathbf{I}). \tag{5.22}$$

We attempt to maximize this function using the EM algorithm developed next.

5.4.2 EM affine algorithm

We again follow our four step procedure (see Section 5.3.2) for deriving EM algorithms. In the first step, we begin by regarding the unknown transformation parameters $\{c_i, d_i\}$ as hidden. The hidden-data density is then the joint posterior $p(c_i, d_i|\mathbf{y}_i)$. Even though the prior model $p(c_i, d_i)$ factors independently, the joint posterior does not. This can be seen from the plate diagram shown in Figure 5.5. The node representing $\mathbf{y}_i$ in the Bayesian network inside of the plate *activates* the link between c_i and d_i. Therefore, knowledge of $\mathbf{y}_i$ makes c_i and d_i dependent.

In the second step, we define the complete-data log-likelihood function as the joint log-likelihood of Y and the hidden data $\{c_i, d_i\}$. This can be written as the sum over all n curves of the log joint density in (5.21). This function has the form

$$\mathcal{L}_c = \sum_i \log \mathcal{N}(\mathbf{y}_i|c_i\mathbf{B}_i\boldsymbol{\beta} + d_i, \sigma^2\mathbf{I})\mathcal{N}(c_i|1, u^2)\mathcal{N}(d_i|0, v^2). \tag{5.23}$$

The final two steps, the E- and M-steps, are given next.

E-step

In the E-step we first calculate the joint posterior $p(c_i, d_i|\mathbf{y}_i)$ and then use this to take expectations of Equation (5.23). The posterior can be calculated analytically in this case. Upon expansion of the density we have

$$\begin{aligned} p(c_i, d_i|\mathbf{y}_i) &\propto p(\mathbf{y}_i|c_i, d_i)p(c_i)p(d_i) \\ &\propto \exp\Big\{-\|\mathbf{y}_i - c_i\mathbf{B}_i\boldsymbol{\beta} - d_i\|^2/2\sigma^2 \\ &\quad -(c_i-1)^2/2u^2 - d_i^2/2v^2\Big\}, \end{aligned} \tag{5.24}$$

which can be recognized as a bi-variate normal density since this is an exponential function of a quadratic polynomial in two variables. All that remains is to find the five parameters that identify this distribution: two means, two variances, and a single covariance. After some algebraic manipulation the resulting parameters can be found. For the posterior means we have

$$\hat{c}_i = V_{c_i}(\boldsymbol{\beta}'\mathbf{B}_i'\mathbf{V}^{-1}\mathbf{y}_i + 1/u^2) \tag{5.25}$$

and

$$\hat{d}_i = V_{d_i}(\mathbf{y}_i - \mathbf{B}_i\boldsymbol{\beta})'\,\mathbf{U}^{-1}\mathbf{1}, \tag{5.26}$$

where V_{c_i} and V_{d_i} are the posterior variances given below. Notice that the equation for $\hat{d}_i$ is again the weighted difference between the actual curve and the model-based mean curve. The difference between this equation and that of (5.10) is that the noise model has been augmented and replaced by $\mathbf{U} = u^2\mathbf{B}_i\boldsymbol{\beta}\boldsymbol{\beta}'\mathbf{B}_i' + \sigma^2\mathbf{I}$ which is the variance associated with the random term $c_i\boldsymbol{\beta}\mathbf{B}_i + \boldsymbol{\epsilon}_i$. In other words, $\mathbf{U}$ is the variance associated with everything except d_i. Likewise, the equation for $\hat{c}_i$ is discounted by $\mathbf{V}$ which is the variance associated with everything except c_i.

After a little more work, the posterior variances can be written as

$$V_{c_i} = (\boldsymbol{\beta}'\mathbf{B}_i'\mathbf{V}^{-1}\mathbf{B}_i\boldsymbol{\beta} + 1/u^2)^{-1}, \tag{5.27}$$

$$V_{d_i} = (\mathbf{1}'\mathbf{U}^{-1}\mathbf{1} + 1/v^2)^{-1}, \tag{5.28}$$

with the posterior covariance being

$$V_{c_i d_i} = -uv\sqrt{\lambda V_{c_i} V_{d_i}}\;\; \mathbf{1}'\mathbf{B}_i\boldsymbol{\beta}. \tag{5.29}$$

The equation for λ is

$$\lambda = (u^2\boldsymbol{\beta}'\mathbf{B}_i'\mathbf{B}_i\boldsymbol{\beta} + \sigma^2)^{-1}(n_i v^2 + \sigma^2)^{-1}.$$

The variance equations are straightforward to interpret. For example, the posterior variance of d_i is sensible in that it is the inverse of the sum of the precisions $1/v^2$ and $\mathbf{U}^{-1}$, which are the precisions associated with d_i and everything but d_i, respectively.

It is satisfying that the posterior covariance is negative so that larger values of c_i would tend to *explain away* larger values of d_i, given knowledge of $\mathbf{y}_i$. This is exactly what we would expect given that in the Bayes net described by the plate diagram in Figure 5.5, $\mathbf{y}_i$ activates the link between c_i and d_i. Using these results we now turn to the calculation of the Q-function.

Calculating the Q-function

The calculation of the Q-function consists of taking the posterior expectation of (5.23) with respect to $p(c_i, d_i|\mathbf{y}_i)$ that we just calculated above. We can greatly simplify this operation by first taking the posterior expectation of the noise term $\boldsymbol{\epsilon}_i = (\mathbf{y}_i - c_i\mathbf{B}_i\boldsymbol{\beta} - d_i)$. We can write this expectation and the related variance as

$$\hat{\boldsymbol{\epsilon}}_i = \mathrm{E}[\boldsymbol{\epsilon}_i|\mathbf{y}_i] = \mathbf{y}_i - \hat{c}_i\mathbf{B}_i\boldsymbol{\beta} - \hat{d}_i, \tag{5.30}$$

and

$$V_{\boldsymbol{\epsilon}_i} = \mathrm{Var}[\boldsymbol{\epsilon}_i|\mathbf{y}_i] = V_{c_i}\mathbf{B}_i\boldsymbol{\beta}\boldsymbol{\beta}'\mathbf{B}_i' + V_{d_i}\mathbb{1} + 2V_{c_i d_i}\mathbf{B}_i\boldsymbol{\beta}\mathbf{1}'. \tag{5.31}$$

We also note that $\mathrm{E}[\boldsymbol{\epsilon}_i'\boldsymbol{\epsilon}_i|\mathbf{y}_i] = \hat{\boldsymbol{\epsilon}}_i'\hat{\boldsymbol{\epsilon}}_i + \mathrm{tr}(V_{\boldsymbol{\epsilon}_i})$. With this we now take the expectation of (5.23) with respect to $p(c_i, d_i|\mathbf{y}_i)$. First we expand the normal densities and carry

the expectation across the non-random terms.

$$\begin{aligned} Q &= \sum_i \int\int [\log p(\mathbf{y}_i|c_i, d_i)p(c_i, d_i)]\; p(c_i, d_i|\mathbf{y}_i)\, dc_i\, dd_i \\ &= \sum_i -\frac{n_i}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}\mathrm{E}\left[\boldsymbol{\epsilon}_i'\boldsymbol{\epsilon}_i|\mathbf{y}_i\right] \\ &\quad -\frac{1}{2}\log 2\pi u^2 - \frac{1}{2u^2}\mathrm{E}\left[(c_i-1)^2|\mathbf{y}_i\right] - \frac{1}{2}\log 2\pi v^2 - \frac{1}{2v^2}\mathrm{E}\left[d_i^2|\mathbf{y}_i\right]. \quad (5.32) \end{aligned}$$

Notice that we are left with taking expectations that only require substitution of known sufficient statistics from the E-step. The substitutions result in the final equation for the Q-function:

$$\begin{aligned} Q &= \sum_i -\frac{n_i}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}\left[\hat{\boldsymbol{\epsilon}}_i'\hat{\boldsymbol{\epsilon}}_i + \mathrm{tr}(V_{\boldsymbol{\epsilon}_i})\right] \\ &\quad -\frac{1}{2}\log 2\pi u^2 - \frac{1}{2u^2}\left[(\hat{c}_i-1)^2 + V_{c_i}\right] - \frac{1}{2}\log 2\pi v^2 - \frac{1}{2v^2}\left[\hat{d}_i^2 + V_{d_i}\right]. \end{aligned}$$

M-step

The final step that must be defined is the M-step. In the M-step the Q-function is maximized over the parameters $\{\boldsymbol{\beta}, \sigma^2, u^2, v^2\}$. The maximization is straightforward as is usually the case with EM since most of the hard work is done in the more difficult E-step. The parameter re-estimation equations are

$$\hat{u}^2 = 1/n \sum_i \left[(\hat{c}_i - 1)^2 + V_{c_i}\right], \quad (5.33)$$

$$\hat{v}^2 = 1/n \sum_i \left[\hat{d}_i^2 + V_{d_i}\right], \quad (5.34)$$

$$\hat{\sigma}^2 = 1/N \sum_i \left[\hat{\boldsymbol{\epsilon}}_i'\hat{\boldsymbol{\epsilon}}_i + \mathrm{tr}(V_{\boldsymbol{\epsilon}_i})\right], \quad (5.35)$$

and

$$\hat{\boldsymbol{\beta}} = \left[\sum_i \mathbf{B}_i'\mathbf{B}_i(\hat{c}_i^2 + V_{c_i})\right]^{-1} \sum_i \mathbf{B}_i'(\hat{c}_i(\mathbf{y}_i - \hat{d}_i) - V_{c_i d_i}). \tag{5.36}$$

These re-estimation equations are quite similar to (5.15)–(5.17) that give the re-estimation equations in the translation case. The only difference is in accounting for the extra variance added due to the uncertainty of c_i and in locating the mean curve coefficients $\hat{\boldsymbol{\beta}}$ while handling the scaling effect.

Further details

A typical initialization of EM consists of sampling random values for the posterior means $\hat{c}_i, \hat{d}_i$, the posterior variances V_{c_i}, V_{d_i}, and the posterior covariances $V_{c_i d_i}$. EM can then be directly started at the M-step. The computational complexity of the EM-affine algorithm is identical to the EM-translation algorithm; it is linear in the total number of points N.

5.5 Experimental results

In this section, we report experimental results on simulated as well as real data that show the EM alignment models out-perform model-free preprocessing and the non-probabilistic model-based alignment method of NPP. Results for both translation- and affine-alignment methods using spline regression as well as polynomial regression models are given.

The fundamental motivating factor for the development of the alignment models in this chapter is to facilitate the integration of curve alignment and curve clustering in a unified framework. However, a valid question is how the new alignment methodology itself compares to more basic alignment methods such as model-free curve normalization.

Table 5.1: NPP Space-Translation Procedure.

1. Initialize all $\hat{d}_i$ to random values.
2. Set $\hat{\boldsymbol{\beta}} = \left[\sum_i \mathbf{B}_i'\mathbf{B}_i\right]^{-1} \sum_i \mathbf{B}_i'(\mathbf{y}_i - \hat{d}_i)$ and $\hat{\sigma}^2 = 1/n \sum_i \left[\left\|\mathbf{y}_i - \mathbf{B}_i\boldsymbol{\beta} - \hat{d}_i\right\|^2\right]$.
3. Set all $\hat{d}_i = \frac{1}{n_i}(\mathbf{y}_i - \mathbf{B}_i\boldsymbol{\beta})'\mathbf{1}$ and enforce identifiability with $\sum_i \hat{d}_i = 0$.
4. Jump back to step (2) until convergence (no change in the parameters).

Recall in the translation case that model-free curve normalization consists of setting the i-th translation d_i to the mean difference between the curve $\mathbf{y}_i$ and the cross-sectional mean curve. In Section 5.3.2, we showed that the estimate for $\hat{d}_i$ in Equation (5.10) is a weighted form of the above curve normalization estimate. The weighted estimate is

$$\hat{d}_i = \frac{V_{d_i}}{\sigma^2}(\mathbf{y}_i - \mathbf{B}_i\boldsymbol{\beta})'\mathbf{1}. \tag{5.37}$$

There are two main differences between (5.37) and curve normalization. First, the term $\mathbf{B}_i\boldsymbol{\beta}$ replaces the cross-sectional mean curve; and second, the weight V_{d_i}/σ^2 determines the degree to which the difference $(\mathbf{y}_i - \mathbf{B}_i\boldsymbol{\beta})'\mathbf{1}$ affects the translation. It is this weighting in the EM algorithm that increases alignment performance.

In addition to comparing the EM alignment models to model-free normalization, we also compare results with a non-probabilistic replica of the alignment models introduced in this chapter. We give the name non-probabilistic Procrustes (NPP) to this method. It follows a Procrustes procedure based on our EM algorithms except that it does not consider the transformation parameters as random variables, and hence, it does not model the uncertainty associated with these hidden data. We provide a listing of the NPP procedure for the translation case in Table 5.1. A listing for the affine version of NPP is similar and is not provided.

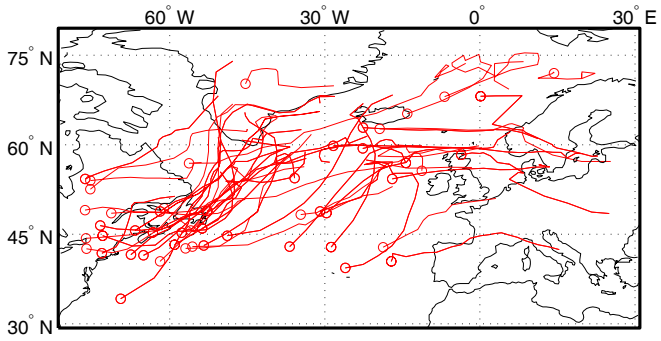


Figure 5.6: Example trajectories from the cyclone dataset that were tracked over the North Atlantic.

5.5.1 Experiments with cyclone data

In this section, we report experimental results using a real dataset. The dataset consists of 614 cyclone trajectories tracked over the North Atlantic from 1980 to 1995 (Gaffney et al., 2001). An example plot of these trajectories shown on a map of the North Atlantic is given in Figure 5.6. The plotted circles indicate the initial positions of each trajectory. The trajectories are two-dimensional, giving the latitude and longitude positions of the cyclones at each time point. These multidimensional curves are modelled with a separate regression model for each output dimension. In other words, both the latitude vs. time and the longitude vs. time dimensions are represented with a regression model as in Equation (5.18). The probabilistic model for the two-dimensional trajectory is then the product of the two individual density models.

The general use of multidimensional curves within our alignment methodology is explicitly derived in Chapter 7. Further details regarding this cyclone dataset and the specific modelling issues involved are discussed in detail in Chapter 9.

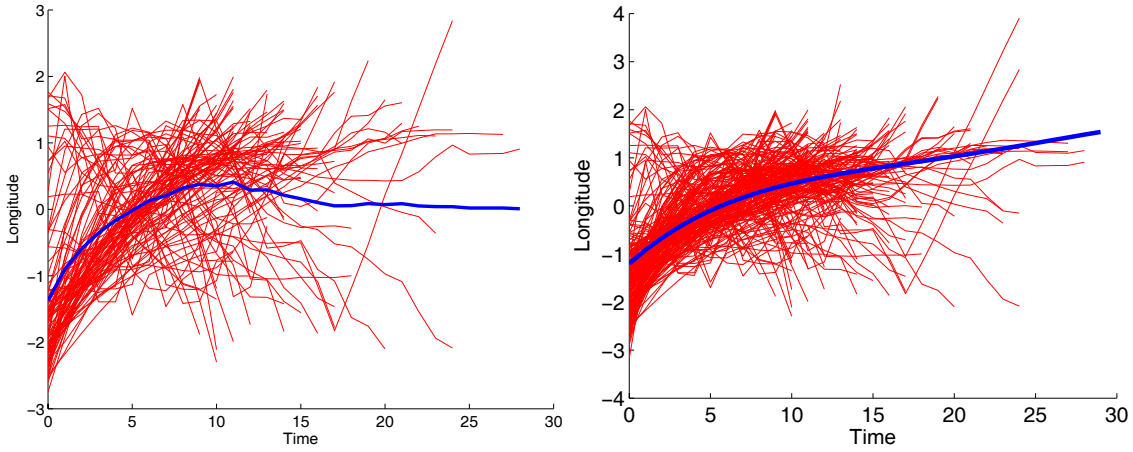


Figure 5.7: Graphic example of the alignments achieved with (left) preprocessing, and (right) EM-Affine on a real cyclone dataset. The bolded lines in each plot give the mean curves.

Graphical comparison to preprocessing

Before presenting explicit quantitative analysis with this dataset, we provide qualitative analysis in the form of graphs. Figure 5.7 shows the results of applying curve preprocessing (left) and the polynomial EM-affine algorithm (right) to a subset of the cyclone trajectories. Second-order polynomial regression models are used in the EM-affine case. The preprocessing involved aligning to the cross-sectional mean and then dividing each curve through by its standard deviation. That is, we set $\mathbf{y}_i' = (\mathbf{y}_i - d_i)/\sigma_i$, where d_i gives the offset for alignment with the cross-sectional mean, and σ_i gives the standard deviation of the curve $\mathbf{y}_i$.

The figure shows the curves of longitude versus time for each of the trajectories in the subset. The thick bolded lines give the representation of the mean curve for each alignment. It is quite obvious that the alignment resulting from the preprocessing is not as compact as that output from the EM-affine algorithm. In fact, the alignment from preprocessing is quite diffuse; its variance is twice that of the alignment from EM.

The mean curve for the preprocessing alignment is also not smooth. It tends to follow the noise of the various trajectories. Furthermore, it doesn't particularly describe any of the trajectories. It does not describe "curve behavior", but instead point-set behavior. The mean curve from the EM alignment appears to match the trajectories more closely and is smooth.

Cross-validation comparisons with NPP

In this section, we describe cross-validation experiments between NPP and the EM alignment algorithms. The experiments used Monte Carlo cross-validation as defined in Appendix B.

The test scores calculated during the cross-validation were based on prediction SSE (sum-of-squared error) scores (the prediction SSE score is described in detail in Section 8.5.3). The score is calculated by taking the learned model and predicting the *test* curve point $\hat{y}_{ij}$ at x_{ij} given the partial test curve $\mathbf{y}_{i(j-1)}$ (which contains all the points up to time $j-1$). This prediction is subtracted from the true value y_{ij}, the result is squared and summed across all the predictions along the curve. Adding these values across all curves in a test set and dividing by the number of predictions gives us the mean prediction SSE score for the test set.

The experiments consisted of 25 runs of MCCV. During each run, a random subset of 70 cyclones was used for training and a random subset of 30 cyclones was used for testing. Predictions for the entire last half of each curve were made. The test scores were averaged over the 25 runs and are reported in Table 5.2.

Eight models were compared, four using polynomial regression and four using spline regression. The spline models are denoted with the suffix "Spline" appended to the end of each name. The translation models are denoted as "Trans" and the affine models as "Affine".

Table 5.2: MCCV results for the EM alignment and NPP models on the cyclone data. The run-averaged SSE score for each of the models is shown under column μ. The corresponding standard deviation is shown under σ.

Model	Prediction SSE Scores	
	μ	σ
EM Affine	14.3611	4.1480
NPP Affine	14.5215	4.4493
EM Affine Spline	15.9214	4.5133
NPP Affine Spline	18.8216	8.1180
EM Trans	72.1898	15.3944
NPP Trans	72.4254	15.4808
EM Trans Spline	88.0819	33.8302
NPP Trans Spline	88.3507	33.8767

The EM-affine model performed the best overall, just edging out its model-based NPP counterpart. The results show a clear distinction between the translation models and the affine models on this dataset. The more complex affine models showed a fundamental increased capacity for predictive generalization with the cyclones.

It appears that the spline-based alignment models suffered from over-fitting as they are out-performed by the polynomial-based alignment models for each of the translation- and affine-alignment problems. Experimental results presented with a real-world gene expression dataset in the next chapter demonstrate an application in which spline-based alignment models lead to better out-of-sample prediction than the polynomial-based methods.

5.5.2 Experiments with simulated data

In this section, we present experimental results with simulated data. The results show that the probabilistic EM alignment models were able to uncover the underlying "true" transformations with greater accuracy than the non-probabilistic NPP methods at increasing levels of measurement noise.

The simulated data used in these experiments were generated by random spline models. The spline models were of order 4 with 12 knots uniformly spaced across the interval from 0 to 20. The spline coefficients were randomly drawn from a normal distribution with vector mean $\mathbf{1}$ and scalar variance 64.

The spline models were used to generate two different data sets. One with added normal translations in measurement space used to test the translation-based algorithms, and the other with added affine transformations in measurement space used to test the affine-based algorithms.

The experiments for the translation-based algorithms were run as follows. Twenty-five different sets of 25 random spline curves with added translations were generated from a single underlying spline model (i.e., the same spline coefficient vector was used in each case). Each of the curves was evaluated at a random set of 21 time points (uniformly distributed). The translation models were then run on each of the datasets. The output translation parameters from each model were compared to the "true" translations and the mean sum-of-squared error was recorded in each case. This process was repeated at each of four different levels of the measurement noise σ^2, resulting in the evaluation of 100 different subsets of curve data from a single underlying random spline model. Finally, this entire procedure was carried out over three different randomly generated spline models. This resulted in the evaluation of 300 different subsets of data.

The experiments for the affine-based algorithms were carried out in the exact same manner except that random affine transformations were added to the curve datasets instead of only translations. Figure 5.8 shows four examples of the randomly generated data. The top-row in the figure shows two translation-based datasets, while the bottom-row shows two affine-based datasets. The plots in the right of the figure contain more noise than those in the left.

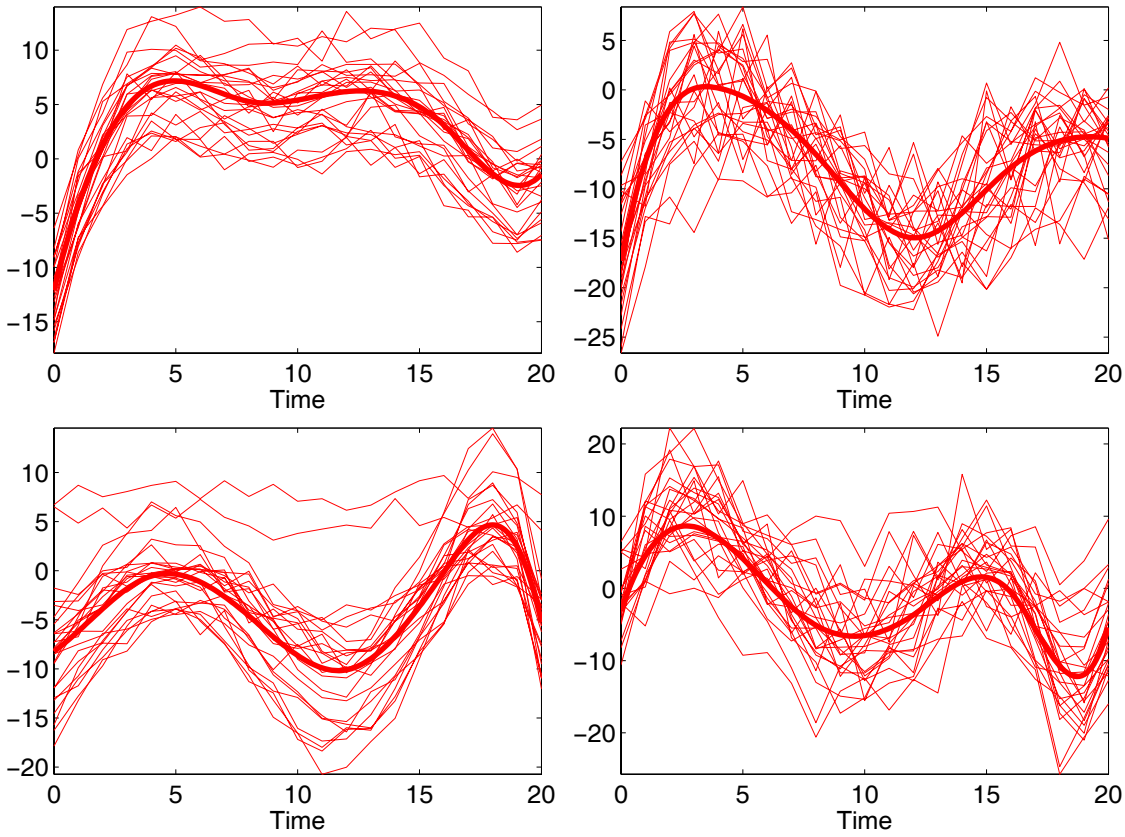


Figure 5.8: Example generated data from random spline models with different levels of noise. The top row shows two data sets generated with random translations; the bottom row shows two data sets generated with random affine transformations. The plots in the left of the figure demonstrate the lower levels of noise present in the sampled data, and the plots in the right demonstrate the higher levels of noise.

The results for the space-translation case can be seen in the left of Figure 5.9. Five models were compared; the same naming scheme used in the previous section is used here except for the new model "Norm" which means preprocessing by aligning to the cross-sectional mean.

These results confirm our intuition that separate modelling of both the measurement noise and the transformation noise should lead to improved performance. At low levels of noise, all of the methods are able to discern the true translations from the measurement noise except for Norm. This is because the cross-sectional mean is not a good representation of the "mean curve" when curves have been measured at different time points, as is common with curve data. As the measurement noise grows, the non-probabilistic alignment methods have a more difficult time at separating the translations from the curve noise. The results also indicate that both the polynomial and spline regression versions exhibit identical performance.

The results for the affine-transformation case are shown in the right of Figure 5.9. (Note that the Norm method has not been included in these results.) We see similar behavior in these results. At low levels of noise, the problem is easy, and thus all of the methods can recover the correct transformations. But at increased levels of noise, the non-probabilistic methods perform poorly.

5.6 Summary

In this chapter, we introduced a novel probabilistic curve alignment model that allows for the alignment of curve data in measurement space. The new methodology represents data objects using a curve representation that eliminates many of the common problems in curve-based analysis. The use of two such representations was demonstrated: polynomial regression models, and spline regression models. A proba-

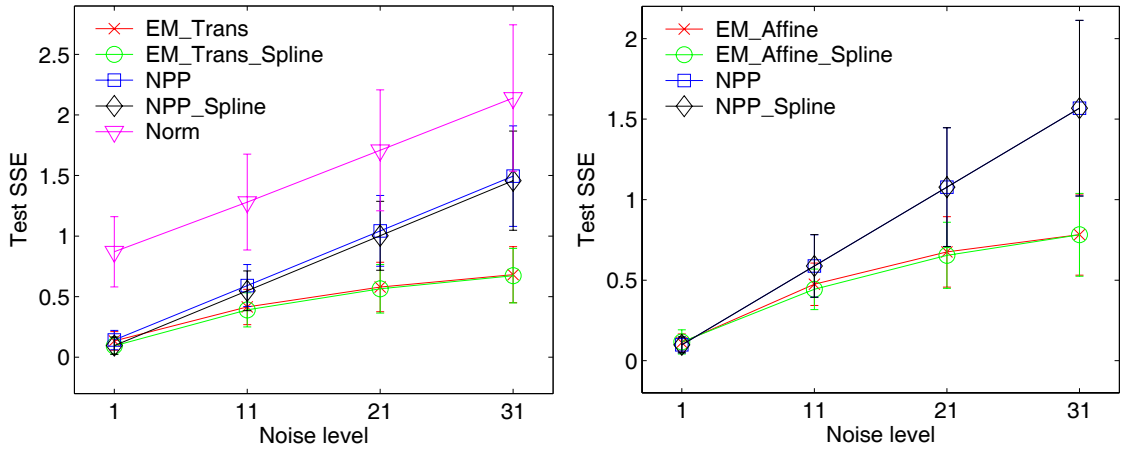


Figure 5.9: Cross-validation results for EM, NPP, and normalization alignment methods. Both polynomial and spline models are shown in each plot. The left plot was generated from the results for translations in space, and the right plot was generated from the results for affine transformations in space. Error bars denote one standard deviation on each side of the plotted mean.

bilistic framework was established employing priors over the set of possible alignment transformations allowing for the identifiable estimation of the unobservable "true" dataset alignment. The framework naturally leads to iterative EM algorithms that provide for the alignment estimation. The main contributions of this chapter can be listed as follows:

- Probabilistic formulation of the space-alignment problem employing priors over the set of possible transformations (resulting in identifiable learning procedures).
- EM algorithm derivation that formalizes the use of a Mahalanobis distance in a Procrustes-type alignment procedure.
- Derivation of the analytic solution of the general affine alignment problem in measurement space.
- Use of curve models in the alignment methodology allowing for the handling of irregular sampled data, variable length curves, missing observations, and leveraging of smoothness information.
- Experimental results with real and simulated data that demonstrate the value of the probabilistic formulation.

Chapter 6

Curve Alignment in Time

6.1 Introduction

In the previous chapter, we introduced an alignment model for curves that allowed for transformations in measurement space. A related and more difficult problem deals with curves which are misaligned in time. In this chapter we propose a novel probabilistic curve alignment model that allows for the continuous alignment of curves in time.

Much of the previous work in time-alignment is couched in optimization theory resulting in complex iterative algorithms with additional constraints that help make the problem well-defined. In contrast, we extend the approach of the previous chapter and apply probabilistic modelling to time-alignment for curves.

We define the time-alignment problem and discuss the novelty of our new methodology in relation to previous work in Section 6.2. In Section 6.3 we introduce our time-translation alignment model and derive the supporting EM time-translation algorithm. In Section 6.4 we introduce a time-alignment model that allows for linear transformations (affine) in time. In Section 6.5, we report experimental results

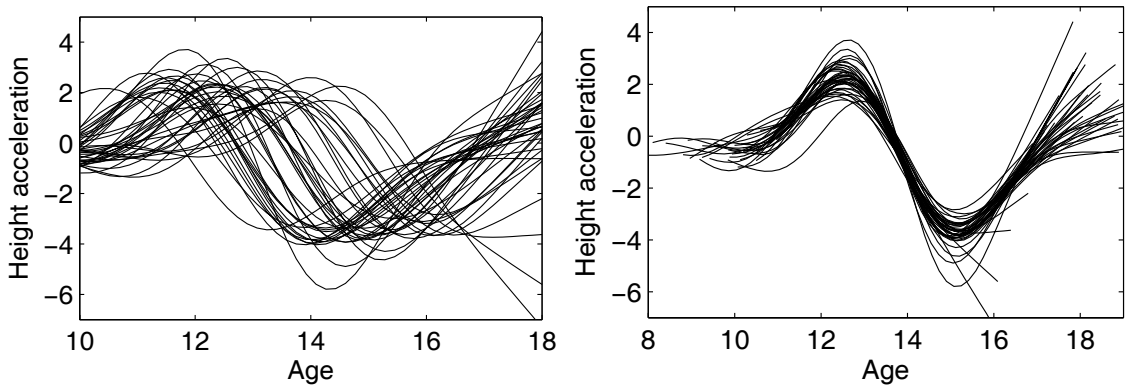


Figure 6.1: Curves measuring the height acceleration for 39 boys; (left) smoothed versions of raw observations, (right) aligned curves.

with a real gene expression dataset and with simulated data that show the benefits of the probabilistic formulation. Finally, we close the chapter with a summary in Section 6.6 .

6.2 Problem definition and prior work

An example alignment problem is given in Figure 6.1 which focuses on the underlying dataset shown earlier in Figure 1.5. The left graph shows a set of spline curves representing the acceleration of height for each of 39 boys whose heights were measured at 29 observation times over the ages of 1 to 18 (Ramsay & Silverman, 1997). The right plot shows the aligned versions output from our spline alignment model (allowing for translations in time). The aligned curves represent the average behavior in a much clearer way. For example, it appears there is an interval of 2.5 years from peak (age 12.5) to trough (age 15) that describes the average cycle that all boys go through. The example demonstrates a common problem with curve data in that the important features of curves tend to be randomly translated in time.

There are two main approaches to aligning curves in time. The first requires

the identification of landmarks usually associated with maxima, minima, or other critical or inflection points of a curve. A set of curves is then aligned so that the landmarks of each curve are synchronized. Landmarks can either be automatically identified or can be defined by an external expert. Gasser and Kneip (1995) coined the term *structural points* that encompass both critical (maxima, minima, singular) and inflection points. They regard these points as defining the structure of curves. In their paper, they develop a nonparametric technique that automatically locates landmarks associated with regions of high *structural intensity* among a set of curves. A related technique known as *structural averaging* (Kneip & Gasser, 1992) jointly aligns curves and identifies these structural points using an iterative algorithm.

A related form of landmark alignment is also used in areas such as medical imaging where it is sometimes referred to as *point-set matching* (Goodall, 1991; Kendall, 1984; Neumann & Lorenz, 1998; Dryden & Mardia, 1998). 2D and 3D shapes are described by sets of points, often consisting of expert-defined landmarks. Iterative Procrustes scaling algorithms are used to estimate the correspondences between the landmarks and the transformations on those landmarks that best align the shapes.

The second main approach to curve alignment in time does not require landmarks, but instead a global fitting function is defined that is optimized to achieve an alignment. Dynamic time warping (DTW) is an example of this type of method. Its origins are based in speech recognition but have since been applied to many other types of problems (Sakoe & Chiba, 1978; Rabiner & Schmidt, 1980). DTW searches for a monotonic warping of the time axis that minimizes a chosen distance function. It is common to choose a distance function that is invariant to selected transformations such as translation in measurement space. However, DTW in its standard form is not a curve modelling technique (it is model-free) and can be considered a

discrete alignment method since it only allows time points to be repeated, skipped or selected. There is no notion of warping to intermediate points along the axis. Despite the absence of a curve model, it has been applied to time-series data sets for query retrieval and clustering in large databases (Keogh & Pazzani, 1998, 1999).

Wang and Gasser (1997) develop a *continuous* time-alignment DTW technique that does take advantage of parametric and semi-parametric curve models. This method requires sets of noise-free curves, thus, requiring the fitting of smooth functionals to the entire curve data set as a preprocessing procedure.

A similar approach is taken in *functional data analysis* (Ramsay & Silverman, 1997). As with the continuous time-alignment DTW technique, functional data analysis requires the fitting of a smooth function (e.g., a spline) to the data from each curve. The set of estimated smooth functions is then used as a proxy for the actual data which is not used in further analysis. One of the tools used in functional data analysis is *curve registration*. Ramsay and Li (1998) detail a Procrustes algorithm that aligns functional data objects by learning an appropriate monotone transformation for each curve that minimizes a penalized squared-error alignment criterion. A similar framework is demonstrated by Silverman (1995) in functional principal components analysis which allows for time shifts through the iterative optimization of integrated sum-of-squares objective functions.

In relation to this body of prior work, we can say that our new approach is characterized by a number factors: it formulates the problem from a probabilistic viewpoint, is not confined to landmarks, allows for continuous time-alignment, and employs curve models to deal with issues such as variable length sequences and irregular sampled curves. The framework naturally leads to an iterative EM algorithm that is similar in many respects to the way in which many of these previous methods operate. However, the formulation of this problem in probabilistic terms is

important for a number of reasons: (1) the explicit use of priors on transformations help define the alignment operation more clearly in a self-contained manner, (2) the resulting EM algorithm formalizes the iterative Procrustes alignment method and demonstrates the use of the Mahalanobis distance metric for alignment, (3) the probabilistic framework helps with the integration of alignment into other more complex problems (such as joint alignment and clustering), and (4) the flexibility of probabilistic models easily allows for the addition of different prior models on transformations in a principled manner.

6.3 Translations in time

In this section we derive an alignment model for translations in time. We follow our previously defined template for the description of new models. The five-step procedure is as follows:

1. Provide the model definition
2. Define the transformation priors
3. Calculate the resulting joint and marginal probability models
4. Define the log-likelihood function
5. Derive the associated EM algorithm

Unlike the space-alignment models in the previous chapter, the derivations for the time-alignment models in this chapter are specific to the particular regression models that are used. Because of the way in which the spline basis matrix $\mathbf{B}_i$ is defined (see Section 3.4), it is not possible to find closed-form solutions for the required expectations in the E-step of the resulting EM algorithms. For this reason, we show the derivations in this chapter using polynomial regression models which,

in particular, allow for closed-form solutions of the Q-function. An identical spline-based EM time-alignment algorithm can still be derived, with the only exception being that the calculation of the Q-function in Equation (6.20) does not contain all of the required posterior variance terms. We discuss the use of spines within our time-alignment methodology during the discussion of the exact calculation of the Q-function in Section 6.3.2.

6.3.1 Model definition

We model a set of curves using a polynomial regression model of the form

$$\mathbf{y}_i = \mathbf{X}_i\boldsymbol{\beta} + \boldsymbol{\epsilon}_i, \qquad \boldsymbol{\epsilon}_i \sim N(\mathbf{0}, \sigma^2\mathbf{I}). \tag{6.1}$$

We allow for a continuous random translation in time by adding the time translation parameter b_i. In other words, we posit that the curve $\mathbf{y}_i$ was actually measured not at $\mathbf{x}_i$ but at some translated time $\mathbf{x}_i - b_i$. This translation cannot be added to the model in the same manner as in (5.2) for translations in space. Instead, the Vandermonde matrix $\mathbf{X}_i$ must be modified to use $\mathbf{x}_i - b_i$ as its input.

To facilitate the writing of equations, we introduce two forms of non-standard notation. First, we write $\mathcal{X}_i$ for the Vandermonde matrix evaluated at any particular transformed time (e.g., $\mathbf{x}_i - b_i$ in this case). Second, we also want to be able to emphasize a particular transformation that is inside of $\mathcal{X}_i$. For this we use the notation $[\![\mathbf{x}_i - b_i]\!]$ which emphasizes the translation b_i in this case. Using this notation, we

write the translated Vandermonde matrix $\mathcal{X}_i$ as

$$[\![\mathbf{x}_i - b_i]\!] = \begin{bmatrix} 1 & (x_{i1} - b_i) & (x_{i1} - b_i)^2 & \cdots & (x_{i1} - b_i)^p \\ 1 & (x_{i2} - b_i) & (x_{i2} - b_i)^2 & \cdots & (x_{i2} - b_i)^p \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & (x_{in_i} - b_i) & (x_{in_i} - b_i)^2 & \cdots & (x_{in_i} - b_i)^p \end{bmatrix}.$$

The time-translated regression model can be written with this notation as either

$$\mathbf{y}_i = \mathcal{X}_i\boldsymbol{\beta} + \boldsymbol{\epsilon}_i, \qquad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}), \tag{6.2}$$

or

$$\mathbf{y}_i = [\![\mathbf{x}_i - b_i]\!]\boldsymbol{\beta} + \boldsymbol{\epsilon}_i, \qquad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}), \tag{6.3}$$

depending on our needs.

Prior model: $p(b_i)$

We consider b_i to be a random variable with an associated prior probability density attached to it. The prior model for the time translation should encode the idea that the most likely translation is the zero translation and should also discount the likelihood of large translations. A zero-mean Gaussian prior is a good fit for this and so we set $p(b_i) = \mathcal{N}(b_i|0, s^2)$, where s^2 gives the variance. This variance will be learned from the data within the ensuing EM algorithm. Rønn (2001) also specifies a prior on time translations in this manner but he does not use them to develop an EM algorithm as we do here.

The graphical plate structure for the time-translation model is shown in Figure 6.2. We have introduced a new node $\mathbf{x}_i$ representing the sequence of "times" at which curve $\mathbf{y}_i$ was observed. The time translation b_i affects $\mathbf{y}_i$ through this node

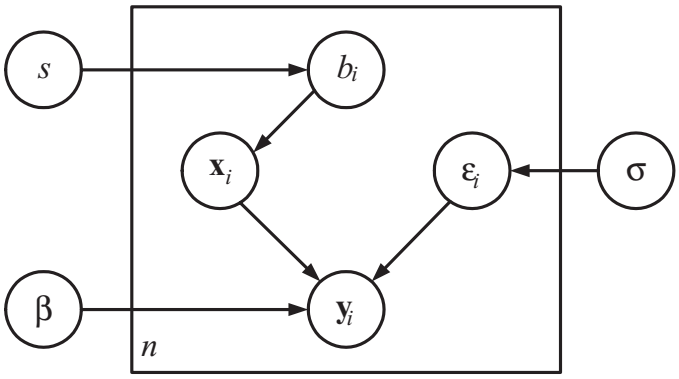


Figure 6.2: Graphical model structure for the time-translation alignment model.

in the graphical model.

Joint, marginal, and log-likelihood

The model specification in (6.2) results in the conditional probability density for $\mathbf{y}_i$ as

$$p(\mathbf{y}_i|b_i) = \mathcal{N}(\mathbf{y}_i|\mathcal{X}_i\boldsymbol{\beta}, \sigma^2\mathbf{I}). \tag{6.4}$$

If we assume that b_i is a random variable with prior $\mathcal{N}(b_i|0, s^2)$, then we can write the joint probability density for $\mathbf{y}_i$ and b_i in the form

$$\begin{aligned} p(\mathbf{y}_i, b_i) &= p(\mathbf{y}_i|b_i)p(b_i) \\ &= \mathcal{N}(\mathbf{y}_i|\mathcal{X}_i\boldsymbol{\beta}, \sigma^2\mathbf{I})\mathcal{N}(b_i|0, s^2). \end{aligned} \tag{6.5}$$

From this joint probability density we can produce the marginal density for $\mathbf{y}_i$ by integrating over b_i. However, the time-alignment model does not afford an analytic solution for this integration. We can, instead, approximate this integral using numerical integration. Since it is relatively easy to obtain samples from the prior distribution $\mathcal{N}(b_i|0, s^2)$, we can use a standard Monte Carlo integration technique for this purpose (Lange, 1999; Gentle, 1998; Press et al., 1992). The approximation

becomes

$$\begin{aligned} p(\mathbf{y}_i) &= \int p(\mathbf{y}_i|b_i)p(b_i)\,db_i \\ &\approx \frac{1}{M}\sum_m p(\mathbf{y}_i|b_i^{(m)}), \end{aligned}$$

where

$$b_i^{(m)} \sim \mathcal{N}(0, s^2), \quad \text{for } m = 1, \ldots, M.$$

The log-likelihood is then the sum over all n curves of the approximate log marginal of $\mathbf{y}_i$:

$$\begin{aligned} \log p(Y) &= \sum_i \log \int p(\mathbf{y}_i, b_i)\,db_i \\ &\approx \sum_i \log \sum_m p(\mathbf{y}_i|b_i^{(m)}) - n\log M \end{aligned} \tag{6.6}$$

It turns out, exact calculation of the log-likelihood is not required for the alignment model. In fact, the EM algorithm only requires calculation of the joint density in (6.5). We will see in Chapter 8, that the integration of alignment and clustering do require the explicit calculation of the marginal and the log-likelihood. But for now, the only reason for any calculation of the log-likelihood is to monitor the EM algorithm for convergence.

However, it is possible to employ a stopping criterion that does not compute the log-likelihood function and thus avoids the numerical integration altogether (e.g., by allowing for a fixed number of iterations or by monitoring the change in the values of variables). In any case, the approximation is shown to be sufficiently accurate even for a sample size of $M = 100$ (see Section 8.4.1 for a discussion of this behavior).

6.3.2 EM time-translation algorithm

In this section, we follow our four step procedure for deriving EM algorithms (see Section 5.3.2). We begin by regarding the unknown time-translation parameters $\{b_i\}$ as hidden. The hidden-data density, then, becomes the posterior $p(b_i|\mathbf{y}_i)$, giving us a distribution on the possible unknown values for the time translations.

In the second step, we define the complete-data log-likelihood function as the joint log-likelihood of Y and $\{b_i\}$. This is the sum over all n curves of the log joint density in (6.5). This function takes the form

$$\begin{aligned} \mathcal{L}_c &= \sum_i \log p(\mathbf{y}_i|b_i)p(b_i) \\ &= \sum_i \log \mathcal{N}(\mathbf{y}_i|\mathcal{X}_i\boldsymbol{\beta}, \sigma^2\mathbf{I})\mathcal{N}(b_i|0, s^2). \end{aligned} \tag{6.7}$$

The EM algorithm will iterate between calculating the expected value of $\mathcal{L}_c$ with respect to $p(b_i|\mathbf{y}_i)$ in the E-step and calculating the new parameter estimates in the M-step.

E-step

In the E-step, we first calculate the posterior $p(b_i|\mathbf{y}_i)$ and then use this to take expectations of the complete-data log-likelihood function in (6.7). For the posterior we have

$$\begin{aligned} p(b_i|\mathbf{y}_i) &\propto p(\mathbf{y}_i|b_i)p(b_i) \\ &\propto \exp\left\{-\left\|\mathbf{y}_i - [\![\mathbf{x}_i - b_i]\!]\boldsymbol{\beta}\right\|^2/2\sigma^2 - b_i^2/2s^2\right\} \end{aligned} \tag{6.8}$$

which, in general, cannot be identified with a known parametric density and does not provide closed form solutions for the required sufficient statistics except for

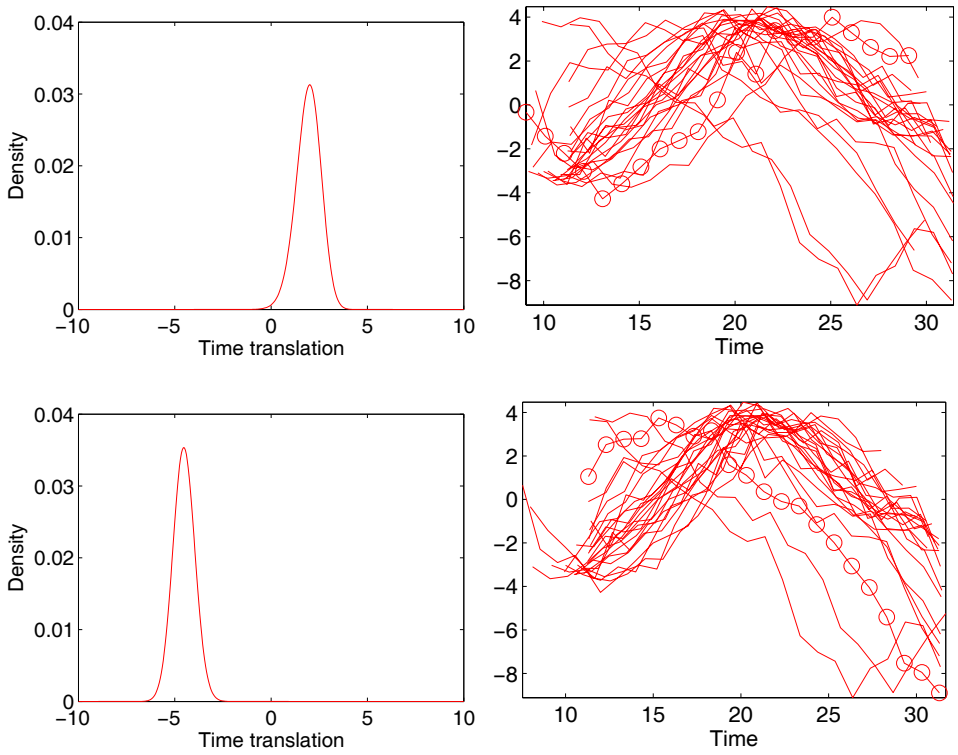


Figure 6.3: Example plots of the normalized posterior $p(b_i|\mathbf{y}_i)$. The top-row shows an estimate of the normalized posterior in the left axes, for the curve plotted with circle symbols in the right. The bottom-row shows the same situation for another curve in the same dataset.

the specific case of polynomial regression models of order one. For the general case, we must seek an approximation. In the following subsections, we describe the approximation problem and show how we handle this within our framework. We then turn to the problem of exact calculation of the Q-function given these approximations.

Posterior approximation

The fact that posterior densities tend towards highly peaked Gaussian densities has been widely noted (e.g, Gelman et al., 1995, Tanner, 1996) and leads to the normal approximation of posterior densities. Figure 6.3 shows the data-driven estimates of

the posterior $p(b_i|\mathbf{y}_i)$ for two different curves during a run of the EM-translation algorithm. The estimates were generated by evaluating Equation (6.8) over a fine grid of time points with the results normalized and plotted.

The top-row shows an estimate of the normalized posterior (on the left) for the curve plotted with circle symbols (on the right). The bottom-row shows the associated plots for another curve in the same dataset. The plots clearly show that the posteriors resulting from the EM-translation algorithm do appear Gaussian.

In the companion figure (Figure 6.4), the same posteriors are estimated a few iterations later. The posteriors are now more highly peaked about the mean than they were earlier. These results suggest that the normal approximation is appropriate for the current situation.

There isn't a need to approximate the entire posterior in order to complete the EM algorithm since the EM algorithm only requires the computation of sufficient statistics in the E-step. Only two quantities are needed: $\mathrm{E}[b_i|\mathbf{y}_i]$ and $\mathrm{E}[b_i^2|\mathbf{y}_i]$.

We can obtain relatively good approximations of these two quantities. First we use a univariate unconstrained maximization technique to find the mode $\hat{b}_i$ of the log posterior density $l \propto \log p(b_i|\mathbf{y}_i)$ which coincides with $\mathrm{E}[b_i|\mathbf{y}_i]$ when the posterior is approximately normal. Then we estimate the posterior variance V_{b_i} by evaluating the inverse of the *observed information* I at $\hat{b}_i$:

$$\begin{aligned} V_{b_i} &= I^{-1}(\hat{b}_i|\mathbf{y}_i) \\ &= \left(-\frac{d^2 l}{db_i^2}\right)^{-1}\Bigg|_{\hat{b}_i}. \end{aligned}$$

The observed information can be computed analytically; it is the negative second derivative of l. The Bayesian justification for this approximation is based on expanding the log-posterior density about the mode using the first two derivatives (Tanner,

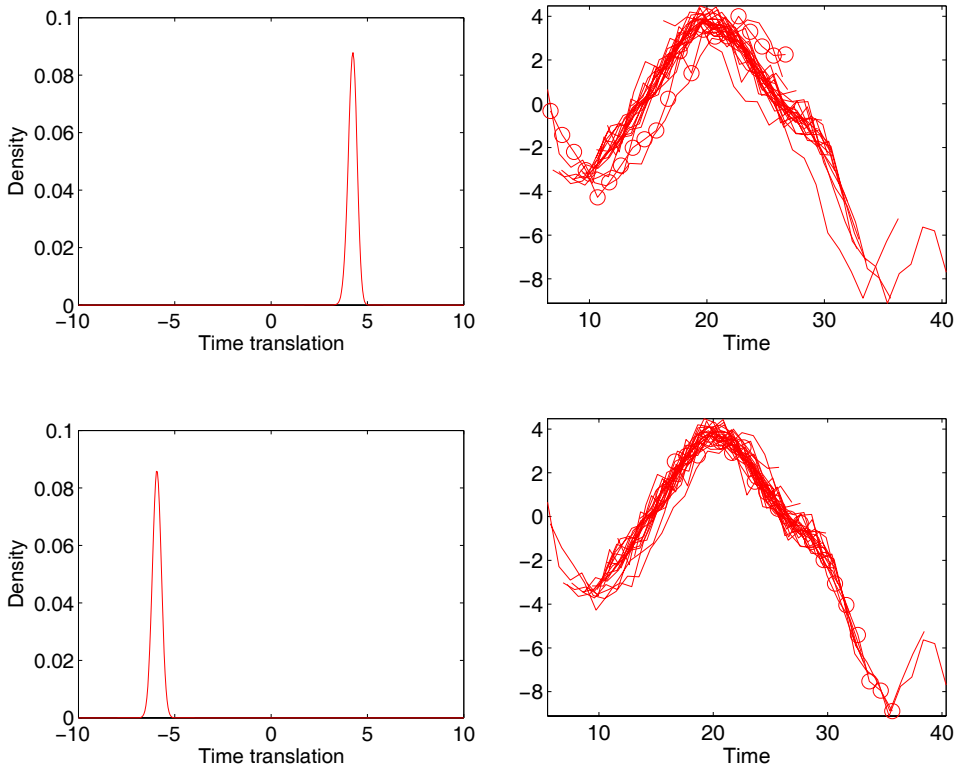


Figure 6.4: Example plots of the normalized posterior $p(b_i|\mathbf{y}_i)$ corresponding to a few iterations after the same posteriors estimated in Figure 6.3. The top- and bottom-rows show the corresponding estimates to those in the companion figure.

1996). The approximation is valid asymptotically.

Calculating the exact Q-function

Calculation of the Q-function is quite complex but can be computed efficiently once the analytic solution is found. We start by writing out the general form for the posterior expectation of $\mathcal{L}_c$ from (6.7):

$$\begin{aligned} Q &= \sum_i \int \Big[\log p(\mathbf{y}_i|b_i)p(b_i)\Big]\; p(b_i|\mathbf{y}_i)\, db_i \\ &= \sum_i \int \Big[\log \mathcal{N}(\mathbf{y}_i|\mathcal{X}_i\boldsymbol{\beta}, \sigma^2\mathbf{I})\mathcal{N}(b_i|0, s^2)\Big]\; p(b_i|\mathbf{y}_i)\, db_i \end{aligned}$$

$$= \sum_i \int \left[-\frac{n_i}{2} \log 2\pi\sigma^2 - \frac{1}{2\sigma^2} \left\| \mathbf{y}_i - \mathcal{X}_i\boldsymbol{\beta} \right\|^2 - \frac{1}{2} \log 2\pi s^2 - \frac{b_i^2}{2s^2} \right] p(b_i|\mathbf{y}_i)\, db_i.$$

After carrying through the integral we end up with the form

$$Q = \sum_i -\frac{n_i}{2} \log 2\pi\sigma^2 - \frac{1}{2\sigma^2} \mathrm{E}\left[\left\| \mathbf{y}_i - \mathcal{X}_i\boldsymbol{\beta} \right\|^2 \right] - \frac{1}{2} \log 2\pi s^2 - \frac{1}{2s^2} \left[\hat{b}_i^2 + V_{b_i} \right], \quad (6.9)$$

in which the E operator encloses the last part of the integration that has not been carried out. We proceed by isolating this calculation in the form

$$\mathrm{E}\left[\left\| \mathbf{y}_i - \mathcal{X}_i\boldsymbol{\beta} \right\|^2 \right] = \mathbf{y}_i'\mathbf{y}_i - 2\mathbf{y}_i'\mathrm{E}[\mathcal{X}_i]\boldsymbol{\beta} + \boldsymbol{\beta}'\mathrm{E}[\mathcal{X}_i'\mathcal{X}_i]\boldsymbol{\beta}. \quad (6.10)$$

In order to complete the Q-function we must calculate the two remaining expectations $\mathrm{E}[\mathcal{X}_i]$ and $\mathrm{E}[\mathcal{X}_i'\mathcal{X}_i]$. A useful solution is a decomposable one; one in which the expectation fits the form $\mathrm{E}[\mathcal{X}_i] = \hat{\mathcal{X}}_i + \Delta$, where $\hat{\mathcal{X}}_i$ is $[\![\mathbf{x}_i - \hat{b}_i]\!]$ and Δ is whatever is left over. We derive such a solution next.

Calculating $\mathbf{E}[\mathcal{X}_i]$

The expectation of $\mathcal{X}_i$ is equal to the matrix of component-wise expectations:

$$\mathrm{E}[\mathcal{X}_i] = \mathrm{E}\left[[\![\mathbf{x}_i - b_i]\!] \right] = \begin{bmatrix} 1 & \mathrm{E}\left[(x_{i1} - b_i)\right] & \cdots & \mathrm{E}\left[(x_{i1} - b_i)^p\right] \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \mathrm{E}\left[(x_{in_i} - b_i)\right] & \cdots & \mathrm{E}\left[(x_{in_i} - b_i)^p\right] \end{bmatrix}. \quad (6.11)$$

The component-wise expectations can be further broken down through use of the binomial formula:

$$\mathrm{E}[(x_{ij} - b_i)^p] = \mathrm{E}\left[\sum_{m=0}^{p} (-1)^m C_m^p x_{ij}^{p-m} b_i^m \right]$$

$$= \sum_{m=0}^{p} (-1)^m C_m^p x_{ij}^{p-m} \mathrm{E}\left[b_i^m\right], \tag{6.12}$$

where C_m^p is the number of combinations of p items taken m at a time. We are now left with the problem of taking the expectation of a random variable raised to an arbitrary power, or to calculating an arbitrary moment. However, the only two statistics that we have at our disposal from the posterior $p(b_i|\mathbf{y}_i)$ are the mean $\hat{b}_i$ and the variance V_{b_i} that we approximated. Therefore, we need a decomposable closed-form solution for the m-th moment of a normally distributed random variable in terms of its mean and variance.

Interestingly, during a cursory search of the relevant literature, we were not able to find such a solution. Fortunately, we were able to derive the solution for the general case. Suppose that $z \sim \mathcal{N}(\mu, \sigma^2)$, then we can write the m-th moment of z in terms of just μ and σ as

$$\mathrm{E}[z^m] = \mu^m + \gamma_{mz}, \qquad \gamma_{mz} = \sum_{q=1}^{\lfloor m/2 \rfloor} G_q C_{2q}^m \sigma^{2q} \mu^{m-2q}, \tag{6.13}$$

where $G_q = \prod_{j=1}^{q}(2j-1)$ is the product of the first q odd numbers. We can now substitute this closed-form solution into (6.12) and run the binomial theorem in reverse:

$$\begin{aligned} \mathrm{E}[(x_{ij} - b_i)^p] &= \sum_{m=0}^{p} (-1)^m C_m^p x_{ij}^{p-m} \mathrm{E}\left[b_i^m\right] \\ &= \sum_{m=0}^{p} (-1)^m C_m^p x_{ij}^{p-m} [\hat{b}_i^m + \gamma_{mb_i}] \\ &= \mathrm{E}[(x_{ij} - \hat{b}_i)^p] + \Delta_b^p(x_{ij}), \end{aligned} \tag{6.14}$$

where

$$\Delta_b^p(x_{ij}) = \sum_{m=0}^{p} (-1)^m C_m^p x_{ij}^{p-m} \gamma_{mb_i}.$$

The full matrix expectation $\mathrm{E}[\mathcal{X}_i]$ can then be written succinctly in the form

$$\mathrm{E}[\mathcal{X}_i] = [\![\mathbf{x}_i - \hat{b}_i]\!] + \mathbf{V}_{\mathbf{X}i}, \tag{6.15}$$

in which the matrix $\mathbf{V}_{\mathbf{X}i}$ is

$$\mathbf{V}_{\mathbf{X}i} = \begin{bmatrix} 0 & 0 & \Delta_b^2(x_{i1}) & \cdots & \Delta_b^p(x_{i1}) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \Delta_b^2(x_{in_i}) & \cdots & \Delta_b^p(x_{in_i}) \end{bmatrix}, \tag{6.16}$$

and where $\Delta_b^0(\mathbf{x}_i) = \Delta_b^1(\mathbf{x}_i) = 0$.

Calculating $\mathbf{E}[\mathcal{X}_i'\mathcal{X}_i]$

The calculation of $\mathrm{E}[\mathcal{X}_i'\mathcal{X}_i]$ follows directly from $\mathrm{E}[\mathcal{X}_i]$. First we let

$$\mathrm{E}^m = \sum_j^{ni} \mathrm{E}\left[(x_{ij} - b_i)^m\right], \tag{6.17}$$

which is just the sum of column $m+1$ of $\mathrm{E}[\mathcal{X}_i]$ in (6.11). This allows us to write $\mathrm{E}[\mathcal{X}_i'\mathcal{X}_i]$ as

$$\mathrm{E}[\mathcal{X}_i'\mathcal{X}_i] = \begin{bmatrix} n_i & \mathrm{E}^1 & \mathrm{E}^2 & \cdots & \mathrm{E}^p \\ \mathrm{E}^1 & \mathrm{E}^2 & \mathrm{E}^3 & \cdots & \mathrm{E}^{p+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathrm{E}^p & \mathrm{E}^{p+1} & \mathrm{E}^{p+2} & \cdots & \mathrm{E}^{2p} \end{bmatrix}. \tag{6.18}$$

In other words, we can find $\mathrm{E}[\mathcal{X}_i'\mathcal{X}_i]$ simply by summing down the columns of $\mathrm{E}[\mathcal{X}_i]$ and placing the column-sums in the right positions of this matrix. In notation,

summing (6.14) over j results in the decomposable solution for E^m as

$$\mathrm{E}^m = \sum_j^{n_i} (x_{ij} - \hat{b}_i)^m + \sum_j^{n_i} \Delta_b^m(x_{ij}).$$

The full matrix expectation $\mathrm{E}[\mathcal{X}_i'\mathcal{X}_i]$ can then be written in the form

$$\mathrm{E}[\mathcal{X}_i'\mathcal{X}_i] = [\![\mathbf{x}_i - \hat{b}_i]\!]'[\![\mathbf{x}_i - \hat{b}_i]\!] + \mathbf{V}_{\mathbf{xx}i}, \tag{6.19}$$

where $\mathbf{V}_{\mathbf{xx}i}$ is

$$\mathbf{V}_{\mathbf{xx}i} = \begin{bmatrix} 0 & 0 & \sum_j \Delta_b^2(x_{ij}) & \cdots & \sum_j \Delta_b^p(x_{ij}) \\ 0 & \sum_j \Delta_b^2(x_{ij}) & \sum_j \Delta_b^3(x_{ij}) & \cdots & \sum_j \Delta_b^{p+1}(x_{ij}) \\ \sum_j \Delta_b^2(x_{ij}) & \sum_j \Delta_b^3(x_{ij}) & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \sum_j \Delta_b^p(x_{ij}) & \cdots & \cdots & \cdots & \sum_j \Delta_b^{2p}(x_{ij}) \end{bmatrix}.$$

What results is a straightforward computation for Q. The computational effort is spent computing $\Delta_b^m(x_{ij})$ for $2 \leq m \leq 2p$ which has complexity $O(p^3)$. However, p is the order of the regression model which is usually just 2, 3, or 4.

The complete Q-function

We are now able to complete the specification of the Q-function. Using the results for $\mathrm{E}[\mathcal{X}_i]$ and $\mathrm{E}[\mathcal{X}_i'\mathcal{X}_i]$ we can rewrite (6.9) as

$$\begin{aligned} Q &= \sum_i -\frac{n_i}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}\mathrm{E}\left[\left\|\mathbf{y}_i - \mathcal{X}_i\boldsymbol{\beta}\right\|^2\right] - \frac{1}{2}\log 2\pi s^2 - \frac{1}{2s^2}\left[\hat{b}_i^2 + V_{b_i}\right] \\ &= \sum_i -\frac{n_i}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}\left[\left\|\mathbf{y}_i - \hat{\mathcal{X}}_i\boldsymbol{\beta}\right\|^2 - 2\mathbf{y}_i'\mathbf{V}_{\mathbf{X}i}\boldsymbol{\beta} + \boldsymbol{\beta}'\mathbf{V}_{\mathbf{xx}i}\boldsymbol{\beta}\right] \\ &\quad -\frac{1}{2}\log 2\pi s^2 - \frac{1}{2s^2}\left[\hat{b}_i^2 + V_{b_i}\right], \end{aligned} \tag{6.20}$$

where $\hat{\mathcal{X}}_i = [\![\mathbf{x}_i - \hat{b}_i]\!]$.

This result is valid only for the polynomial-based time-alignment model. Equation (6.20) cannot be computed for the spline-based time-alignment model. In general, this is because we are not able to expand the components of the spline basis matrix using the binomial formula as in (6.12). Although we can expand the components using the underlying recursive definition of the associated B-spline basis functions (de Boor, 1978), this results in a series of integrations involving partial error functions that cannot be computed analytically.

As a result, the corresponding Q-function for the spline-based time-alignment model can be written as

$$Q = \sum_i -\frac{n_i}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}\left[\left\|\mathbf{y}_i - \hat{\mathbf{B}}_i\boldsymbol{\beta}\right\|^2\right] - \frac{1}{2}\log 2\pi s^2 - \frac{1}{2s^2}\left[\hat{b}_i^2 + V_{b_i}\right], \quad (6.21)$$

where the posterior variance terms associated with the error function $\left\|\mathbf{y}_i - \hat{\mathbf{B}}_i\boldsymbol{\beta}\right\|^2$ have been removed. Note that $\hat{\mathbf{B}}_i$ is the time transformed version of the usual spline basis matrix.

M-step

In the M-step we maximize the Q-function over the set of parameters $\{s^2, \sigma^2, \boldsymbol{\beta}\}$. The parameter re-estimation equations can be easily solved for since the previously hidden-data has been filled-in from the E-step. The derived solutions are as follows:

$$\hat{s}^2 = 1/n \sum_i \left[\hat{b}_i^2 + V_{b_i}\right], \quad (6.22)$$

$$\hat{\sigma}^2 = 1/N \sum_i \left[\left\|\mathbf{y}_i - \hat{\mathcal{X}}_i\boldsymbol{\beta}\right\|^2 - 2\mathbf{y}_i'\mathbf{V}_{\mathbf{X}i}\boldsymbol{\beta} + \boldsymbol{\beta}'\mathbf{V}_{\mathbf{X}\mathbf{X}i}\boldsymbol{\beta}\right], \quad (6.23)$$

and

$$\hat{\boldsymbol{\beta}} = \left[\sum_i \hat{\mathcal{X}}_i' \hat{\mathcal{X}}_i + \mathbf{V}_{\mathbf{X}\mathbf{X}i}\right]^{-1} \sum_i \hat{\mathcal{X}}_i' \mathbf{y}_i - \mathbf{V}_{\mathbf{X}i}' \mathbf{y}_i. \tag{6.24}$$

If you were to perform a non-probabilistic Procrustes alignment for this model, you would remove all terms involving $\mathbf{V}_{\mathbf{X}i}$ and $\mathbf{V}_{\mathbf{X}\mathbf{X}i}$ from the above equations, and you would not calculate $\hat{s}^2$ at all (since this is not present in a non-random approach). What you end up with is an exact least squares solution for non-probabilistic Procrustes. The EM approach, on the other hand, incorporates the uncertainty of b_i into the solutions.

Further details

Initialization for the time-translation model can be carried out by randomly sampling values for the posterior mean $\hat{b}_i$, and variances V_{b_i}, and then starting the iterations at the M-step. The complexity of the time-translation algorithm is $O(NMI)$, where N is the total number of points, M gives the average number of iterations of the minimization procedure to find the posterior modes, and I gives the average number of iterations of EM.

We provide experimental results for this model in Section 6.5. We next use this framework to extend the methodology to handle scaling as well as translations in time.

6.4 Affine transformations in time

In this section we focus on extending the methodology of the previous section to include linear scaling transformations (e.g., "stretching" or "compression") as well as possible translations in time. We begin by presenting the model definition and then develop an EM alignment algorithm for affine transformations.

6.4.1 Model definition

We use a polynomial regression model of the form

$$\mathbf{y}_i = \mathbf{X}_i\boldsymbol{\beta} + \boldsymbol{\epsilon}_i, \qquad \boldsymbol{\epsilon}_i \sim N(\mathbf{0}, \sigma^2\mathbf{I}). \tag{6.25}$$

We augment this model to allow for random affine transformations in time (i.e., $a_i\mathbf{x}_i - b_i$). As in the previous section, we will need to modify the Vandermonde matrix $\mathbf{X}_i$ to be evaluated at the transformed time $a_i\mathbf{x}_i - b_i$ instead of at its usual $\mathbf{x}_i$.

The notation of the previous section is reused so that $\mathcal{X}_i$ notates the Vandermonde matrix evaluated at any particular transformed time (e.g., $a_i\mathbf{x}_i - b_i$ in this section). We also adopt the notation that $\mathcal{X}_i = [\![a_i\mathbf{x}_i - b_i]\!]$ to emphasize the role of the transformation variables. With this notation the time-transformed regression model takes the form

$$\mathbf{y}_i = [\![a_i\mathbf{x}_i - b_i]\!]\boldsymbol{\beta} + \boldsymbol{\epsilon}_i, \qquad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}). \tag{6.26}$$

Prior model: $p(a_i, b_i)$

For the joint prior, we make the independence assumption $p(a_i, b_i) = p(a_i)p(b_i)$ (we do not assume any prior knowledge about covariance between scaling and translations). We take the joint prior from Section 5.4.1 and reuse it here. In other words, we assume that $a_i \sim \mathcal{N}(1, r^2)$ and $b_i \sim \mathcal{N}(0, s^2)$, and thus, the priors specify that the most likely transformation is the identity transformation. The associated plate diagram for the time-affine model is shown in Figure 6.5. The unobservable alignments a_i, b_i affect $\mathbf{y}_i$ through the $\mathbf{x}_i$ node in the graphical model.

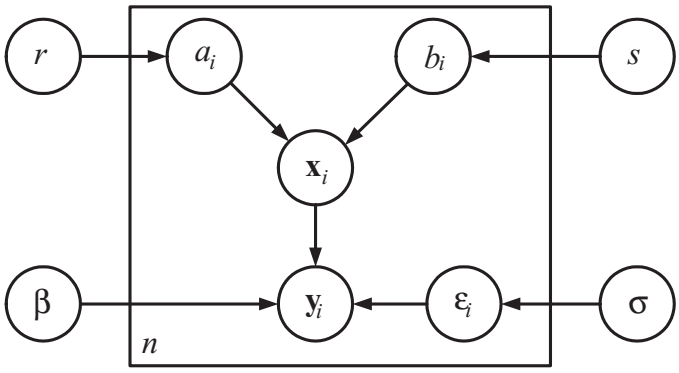


Figure 6.5: Plate diagram describing the time-affine model structure.

Joint, marginal, and log-likelihood

The model specification in (6.26) results in the conditional probability density for $\mathbf{y}_i$ as

$$p(\mathbf{y}_i|a_i, b_i) = \mathcal{N}(\mathbf{y}_i|[\![a_i\mathbf{x}_i - b_i]\!]\boldsymbol{\beta}, \sigma^2\mathbf{I}). \tag{6.27}$$

With the assumption that a_i and b_i are random variables, the joint probability density for $\mathbf{y}_i, a_i$ and b_i takes the form

$$\begin{aligned} p(\mathbf{y}_i, a_i, b_i) &= p(\mathbf{y}_i|a_i, b_i)p(a_i)p(b_i) \\ &= \mathcal{N}(\mathbf{y}_i|[\![a_i\mathbf{x}_i - b_i]\!]\boldsymbol{\beta}, \sigma^2\mathbf{I})\mathcal{N}(a_i|1, r^2)\mathcal{N}(b_i|0, s^2). \end{aligned} \tag{6.28}$$

From the joint model we obtain the marginal density for $\mathbf{y}_i$; however, it cannot be computed analytically. Instead, we use Monte Carlo integration for this task since it is relatively easy to obtain samples from the prior distribution $p(a_i, b_i)$. The approximation becomes

$$\begin{aligned} p(\mathbf{y}_i) &= \int\int p(\mathbf{y}_i|a_i, b_i)p(a_i)p(b_i)\, da_i db_i \\ &\approx \frac{1}{M}\sum_m p(\mathbf{y}_i|a_i^{(m)}, b_i^{(m)}), \end{aligned}$$

where

$$a_i^{(m)} \sim \mathcal{N}(1, r^2), \quad \text{and} \quad b_i^{(m)} \sim \mathcal{N}(0, s^2), \quad \text{for } m = 1, \ldots, M.$$

The log-likelihood follows directly from this approximation and takes the form

$$\log p(Y) \approx \sum_i \log \sum_m p(\mathbf{y}_i | a_i^{(m)}, b_i^{(m)}) - n \log M. \tag{6.29}$$

As in the previous case (Section 6.3.1), the EM algorithm does not explicitly require the evaluation of this approximation. However, it can be calculated to monitor algorithm convergence as noted previously.

6.4.2 EM affine algorithm

In this section we derive the EM time-affine algorithm. The derivation borrows much from the derivations of Sections 5.4.2 and 6.3.2. As such, this information and discussion is not unnecessarily repeated (except where appropriate).

We begin by regarding the unknown transformation parameters $\{a_i, b_i\}$ as hidden, and thus the hidden-data density becomes the posterior $p(a_i, b_i|\mathbf{y}_i)$ (giving us a distribution on the values for the unknown transformation variables). The complete-data log-likelihood function is then the joint log-likelihood of Y and $\{a_i, b_i\}$. This is the sum over all n curves of the log joint density in (6.28). This function takes the form

$$\begin{aligned} \mathcal{L}_c &= \sum_i \log p(\mathbf{y}_i | a_i, b_i) p(a_i) p(b_i) \\ &= \sum_i \log \mathcal{N}(\mathbf{y}_i | \mathcal{X}_i \boldsymbol{\beta}, \sigma^2 \mathbf{I}) \mathcal{N}(a_i | 1, r^2) \mathcal{N}(b_i | 0, s^2). \end{aligned} \tag{6.30}$$

E-step

In the E-step we calculate the posterior $p(a_i, b_i|\mathbf{y}_i)$ and then use this to take expectations of the complete-data log-likelihood function in (6.30). For the posterior we have

$$\begin{aligned} p(a_i, b_i|\mathbf{y}_i) &\propto p(\mathbf{y}_i|a_i, b_i)p(a_i)p(b_i) \\ &\propto \exp\left\{-\left\|\mathbf{y}_i - [\![a_i\mathbf{x}_i - b_i]\!]\boldsymbol{\beta}\right\|^2/2\sigma^2 - (a_i-1)^2/2s^2 - b_i^2/2s^2\right\}, \end{aligned}$$

which, in general, cannot be identified with a known parametric density and does not provide closed form solutions for the required sufficient statistics. Thus, we must seek an approximation.

Posterior approximation

We use the same normal approximation as in Section 6.3.2 except that now we are working in two dimensions. We must find the vector $(\hat{a}_i, \hat{b}_i)$ representing the multi-dimensional mode of $p(a_i, b_i|\mathbf{y}_i)$ and calculate the covariance matrix for $(\hat{a}_i, \hat{b}_i)$.

To find the mode we use a multi-dimensional minimization technique to solve for the vector $(\hat{a}_i, \hat{b}_i)$:

$$(\hat{a}_i, \hat{b}_i) = \underset{(a_i, b_i)}{\arg\min}\{-2\log p(a_i, b_i|\mathbf{y}_i)\}.$$

We use a Nelder-Mead optimization method to perform the minimization (Nelder & Mead, 1965).

For the covariance matrix we evaluate the inverse of the observed information matrix I at $(\hat{a}_i, \hat{b}_i)$ which is the negative Hessian of the log-posterior density (see Section 6.3.2). For convenience, we keep to our earlier notation so that the posterior variance of a_i is V_{a_i}, the posterior variance of b_i is V_{b_i}, and their covariance is $V_{a_i b_i}$.

Calculating the Q-function

Calculation of the Q-function is slightly more complex with affine transformations than with translations, but we carry much of the notation over from Section 6.3.2. We start by writing out the general form for the posterior expectation of $\mathcal{L}_c$:

$$\begin{aligned} Q &= \sum_i \int\int \left[\log p(\mathbf{y}_i|a_i,b_i)p(a_i)p(b_i)\right]\ p(a_i,b_i|\mathbf{y}_i)\,da_i db_i \\ &= \sum_i \int\int \left[\log \mathcal{N}(\mathbf{y}_i|\mathcal{X}_i\boldsymbol{\beta},\sigma^2\mathbf{I})\mathcal{N}(a_i|1,r^2)\mathcal{N}(b_i|0,s^2)\right]\ p(a_i,b_i|\mathbf{y}_i)\,da_i db_i. \end{aligned}$$

After carrying through the logarithm and the integral we end up with the form

$$\begin{aligned} Q &= \sum_i -\frac{n_i}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}\mathrm{E}\left[\|\mathbf{y}_i - \mathcal{X}_i\boldsymbol{\beta}\|^2\right] \\ &\quad -\frac{1}{2}\log 2\pi r^2 - \frac{1}{2r^2}\left[(\hat{a}_i-1)^2 + V_{a_i}\right] - \frac{1}{2}\log 2\pi s^2 - \frac{1}{2s^2}\left[\hat{b}_i^2 + V_{b_i}\right] \quad (6.31) \end{aligned}$$

in which the E operator encloses the last part of the integration that has not been carried out. The remaining integration hinges on the calculation of $\mathrm{E}[\mathcal{X}_i]$ and $\mathrm{E}[\mathcal{X}_i'\mathcal{X}_i]$ which we turn to next

Calculating $\mathbf{E}[\mathcal{X}_i]$

The calculation of $\mathrm{E}[\mathcal{X}_i]$ centers around the solution of $\mathrm{E}[(a_i x_{ij} - b_i)^p]$ (see Section 6.3.2). Using the binomial theorem we can expand this as

$$\begin{aligned} \mathrm{E}[(a_i x_{ij} - b_i)^p] &= \mathrm{E}\left[\sum_{m=0}^{p}(-1)^m C_m^p x_{ij}^{p-m} a_i^{p-m} b_i^m\right] \\ &= \sum_{m=0}^{p}(-1)^m C_m^p x_{ij}^{p-m}\mathrm{E}\left[a_i^{p-m} b_i^m\right]. \quad (6.32) \end{aligned}$$

We focus on the joint expectation $\mathrm{E}\left[a_i^{p-m}b_i^m\right]$ which can be expanded further into the form

$$\mathrm{E}\left[a_i^{p-m}b_i^m\right] = \mathrm{E}[a_i^{p-m}]\mathrm{E}[b_i^m] + \mathrm{Cov}(a_i^{p-m}, b_i^m). \tag{6.33}$$

The product of expectations can be expanded further still by using the closed-form solution for Gaussian moments in Equation (6.13):

$$\mathrm{E}[a_i^{p-m}]\mathrm{E}[b_i^m] = (\hat{a}_i^{p-m} + \gamma_{(p-m)a_i})(\hat{b}_i^m + \gamma_{mb_i}). \tag{6.34}$$

Multiplying through and collecting the terms we find that

$$\mathrm{E}[a_i^{p-m}]\mathrm{E}[b_i^m] = \hat{a}_i^{p-m}\hat{b}_i^m + \Gamma_i^{mp}, \tag{6.35}$$

where

$$\Gamma_i^{mp} = (\hat{a}_i^{p-m}\gamma_{mb_i} + \hat{b}_i^m\gamma_{(p-m)a_i} + \gamma_{(p-m)a_i}\gamma_{mb_i}).$$

Substituting this result back into the joint expectation (6.33) we have

$$\mathrm{E}\left[a_i^{p-m}b_i^m\right] = \hat{a}_i^{p-m}\hat{b}_i^m + \left(\Gamma_i^{mp} + \mathrm{Cov}(a_i^{p-m}, b_i^m)\right). \tag{6.36}$$

Finally, we substitute this result into (6.32) and reverse the binomial theorem:

$$\begin{aligned}
\mathrm{E}[(a_i x_{ij} - b_i)^p] &= \sum_{m=0}^{p} (-1)^m C_m^p x_{ij}^{p-m} \mathrm{E}\left[a_i^{p-m}b_i^m\right] \\
&= \sum_{m=0}^{p} (-1)^m C_m^p x_{ij}^{p-m} \left[\hat{a}_i^{p-m}\hat{b}_i^m + \left(\Gamma_i^{mp} + \mathrm{Cov}(a_i^{p-m}, b_i^m)\right)\right] \\
&= \mathrm{E}[(\hat{a}_i x_{ij} - \hat{b}_i)^p] + \Delta_{ab}^p(x_{ij}),
\end{aligned} \tag{6.37}$$

where

$$\Delta_{ab}^{p}(x_{ij}) = \sum_{m=0}^{p} (-1)^m C_m^p x_{ij}^{p-m} \left(\Gamma_i^{mp} + \mathrm{Cov}(a_i^{p-m}, b_i^m) \right).$$

The full matrix expectation $\mathrm{E}[\mathcal{X}_i]$ can then be written succinctly in the form

$$\mathrm{E}[\mathcal{X}_i] = [\![\hat{a}_i \mathbf{x}_i - \hat{b}_i]\!] + \mathbf{V}_{\mathbf{X}i}, \tag{6.38}$$

in which the matrix $\mathbf{V}_{\mathbf{X}i}$ is

$$\mathbf{V}_{\mathbf{X}i} = \begin{bmatrix} 0 & 0 & \Delta_{ab}^2(x_{i1}) & \cdots & \Delta_{ab}^p(x_{i1}) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \Delta_{ab}^2(x_{in_i}) & \cdots & \Delta_{ab}^p(x_{in_i}) \end{bmatrix},$$

and where $\Delta_{ab}^0(\mathbf{x}_i) = \Delta_{ab}^1(\mathbf{x}_i) = 0$.

Calculating $\mathbf{E}[\mathcal{X}_i'\mathcal{X}_i]$

Using our updated calculations for $\mathrm{E}[\mathcal{X}_i]$ and Δ_{ab}^m, the calculation of $\mathrm{E}[\mathcal{X}_i'\mathcal{X}_i]$ follows exactly that in Section 6.3.2. Hence we only give the final results here. The full matrix expectation $\mathrm{E}[\mathcal{X}_i'\mathcal{X}_i]$ is

$$\mathrm{E}[\mathcal{X}_i'\mathcal{X}_i] = [\![\hat{a}_i \mathbf{x}_i - \hat{b}_i]\!]' [\![\hat{a}_i \mathbf{x}_i - \hat{b}_i]\!] + \mathbf{V}_{\mathbf{X}\mathbf{X}i}, \tag{6.39}$$

where $\mathbf{V_{xx}}_i$ is

$$\mathbf{V_{xx}}_i = \begin{bmatrix} 0 & 0 & \sum_j \Delta_{ab}^2(x_{ij}) & \cdots & \sum_j \Delta_{ab}^p(x_{ij}) \\ 0 & \sum_j \Delta_{ab}^2(x_{ij}) & \sum_j \Delta_{ab}^3(x_{ij}) & \cdots & \sum_j \Delta_{ab}^{p+1}(x_{ij}) \\ \sum_j \Delta_{ab}^2(x_{ij}) & \sum_j \Delta_{ab}^3(x_{ij}) & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \sum_j \Delta_{ab}^p(x_{ij}) & \cdots & \cdots & \cdots & \sum_j \Delta_{ab}^{2p}(x_{ij}) \end{bmatrix}$$

Approximation of Cov(a_i^n, b_i^k)

Despite the elegant solution for $\mathrm{E}[\mathcal{X}_i]$, we have one remaining problem. The solutions of $\mathrm{Cov}(a_i^l, b_i^k)$ for all $l, k \geq 1$ such that $l + k = p$ are needed. There are only $(p-1)$ such combinations, which means there aren't many to calculate since p (the order of the regression model) is usually 2,3, or 4. In the quadratic case, which is quite common, we only need one of these, $\mathrm{Cov}(a_i, b_i)$. But we already have this in the form of $V_{a_i b_i}$. So in the quadratic case there is no extra required calculation.

For the case when $p \geq 3$, we use sampling to estimate the needed covariances. The procedure is to sample from the posterior $p(a_i, b_i|\mathbf{y}_i)$ using the approximate normal fit that we made in the E-step. Then, using the samples, we can estimate all of the needed covariances.

We have found that a small number of samples (e.g., $M = 50$) leads to sufficient approximations. In fact, the method performs well even in the extreme case of setting all the covariances to zero.

The complete Q-function

The complete calculation of the Q-function follows by substitution of (6.38) and (6.39) into (6.31):

$$\begin{aligned} Q &= \sum_i -\frac{n_i}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}\left[\left\|\mathbf{y}_i - \hat{\mathcal{X}}_i\boldsymbol{\beta}\right\|^2 - 2\mathbf{y}_i'\mathbf{V}_{\mathbf{X}i}\boldsymbol{\beta} + \boldsymbol{\beta}'\mathbf{V}_{\mathbf{X}\mathbf{X}i}\boldsymbol{\beta}\right] \\ &\quad -\frac{1}{2}\log 2\pi r^2 - \frac{1}{2r^2}\left[(\hat{a}_i - 1)^2 + V_{a_i}\right] - \frac{1}{2}\log 2\pi s^2 - \frac{1}{2s^2}\left[\hat{b}_i^2 + V_{b_i}\right] \quad (6.40) \end{aligned}$$

where $\hat{\mathcal{X}}_i = [\![\hat{a}_i\mathbf{x}_i - \hat{b}_i]\!]$.

The Q-function for the spline-based alignment model cannot be computed exactly just as in the former case described in Section 6.3.2. Consequently, the Q-function for the spline-based alignment model is identical to Equation (6.40) with the removal of the posterior variance terms associated with the error function $\left\|\mathbf{y}_i - \hat{\mathcal{X}}_i\boldsymbol{\beta}\right\|^2$ as shown earlier.

M-step

In the M-step we maximize the Q-function over the set of parameters $\{r^2, s^2, \sigma^2, \boldsymbol{\beta}\}$. The solutions can be derived in a straight-forward manner. The derived equations are as follows:

$$\hat{r}^2 = 1/n\sum_i\left[(\hat{a}_i - 1)^2 + V_{a_i}\right], \quad (6.41)$$

$$\hat{s}^2 = 1/n\sum_i\left[\hat{b}_i^2 + V_{b_i}\right], \quad (6.42)$$

$$\hat{\sigma}^2 = 1/N\sum_i\left[\left\|\mathbf{y}_i - \hat{\mathcal{X}}_i\hat{\boldsymbol{\beta}}\right\|^2 - 2\mathbf{y}_i'\mathbf{V}_{\mathbf{X}i}\hat{\boldsymbol{\beta}} + \hat{\boldsymbol{\beta}}'\mathbf{V}_{\mathbf{X}\mathbf{X}i}\hat{\boldsymbol{\beta}}\right], \quad (6.43)$$

and

$$\hat{\boldsymbol{\beta}} = \left[\sum_i \hat{\mathcal{X}}_i'\hat{\mathcal{X}}_i + \mathbf{V}_{\mathbf{X}\mathbf{X}i}\right]^{-1}\sum_i \hat{\mathcal{X}}_i'\mathbf{y}_i - \mathbf{V}_{\mathbf{X}i}'\mathbf{y}_i. \quad (6.44)$$

In the non-probabilistic Procrustes version applied to this model you would remove all terms that contain $\mathbf{V}_{\mathbf{X}i}$ and $\mathbf{V}_{\mathbf{X}\mathbf{X}i}$ from the above equations, and you would not solve for $\hat{r}^2$ or $\hat{s}^2$ since they do not appear in such a version. This results in a standard least squares solution for NPP.

Further details

Initialization for the time-affine model can be carried out by randomly sampling values for the posterior means $\hat{a}_i, \hat{b}_i$, variances V_{a_i}, V_{b_i}, and the covariance $V_{a_i b_i}$, and then starting the iterations at the M-step. The complexity of the time-affine algorithm is $O(NMI)$, where N is the total number of points, M gives the average number of iterations of the minimization procedure to find each of the posterior modes, and I gives the average number of iterations of EM.

6.5 Experimental results

In this section, we present experimental results with both simulated and real data. The motivating factor for the development of these probabilistic alignment models is to facilitate the integration of curve alignment and curve clustering. However, it is still important to demonstrate to what extent these alignment models are useful in themselves.

We compare the EM time alignment algorithms of this chapter to a time-alignment version of NPP from the last chapter. Note that NPP primarily differs from the full EM alignment algorithms in that it does not treat the transformation parameters as random. It still gets the benefit of using curve models to represent the set of sequence data and uses an iterative algorithm to optimize the alignments. The goal of these experiments is to determine whether the probabilistic modelling is benefi-

Table 6.1: NPP Time-Translation Procedure

1. Initialize all $\hat{b}_i$ to random values.
2. Set $\hat{\boldsymbol{\beta}} = \left[\sum_i \hat{\mathcal{X}}'_i \hat{\mathcal{X}}_i\right]^{-1} \sum_i \hat{\mathcal{X}}'_i \mathbf{y}_i$ and $\hat{\sigma}^2 = 1/N \sum_i \left\|\mathbf{y}_i - \hat{\mathcal{X}}_i \boldsymbol{\beta}\right\|^2$.
3. Set all $\hat{b}_i = \arg\min_{b_i} \|\mathbf{y}_i - [\![\mathbf{x}_i - b_i]\!]\boldsymbol{\beta}\|^2$ and enforce $\sum_i \hat{b}_i = 0$.
4. Jump back to step (2) until convergence (no change in the parameters).

cial for curve alignment. A listing for the time-translation version of NPP is given in Table 6.1. A listing for the affine-version is similar and is not provided.

6.5.1 Experiments with gene expression data

In this section, we present experimental results with the time-alignment models on a real gene expression dataset. This dataset consists of normalized gene expression observations over time. The expression profiles measure the activity of cell cycle-regulated genes in yeast. The full dataset contains 800 curves giving the expression profiles for 800 different genes. Clustering is often used in gene expression analysis to reveal groups of genes with similar profiles that may be physically related to some underlying biological process (e.g., Spellman et al., 1998; Eisen et al., 1998; Ben-Dor et al., 1999; Aach & Church, 2001). However, here we use this dataset to demonstrate the time-alignment problem since there are known time-delay effects in these profiles. Further details about the creation and basic analysis of this dataset can be found in Spellman et al. (1998).

Figure 6.6 shows 100 randomly selected genes from this dataset. The y-axis is the normalized log-ratio of expression, and the x-axis is time. The curves show a definite periodicity that is characteristic of these cell cycle-regulated genes.

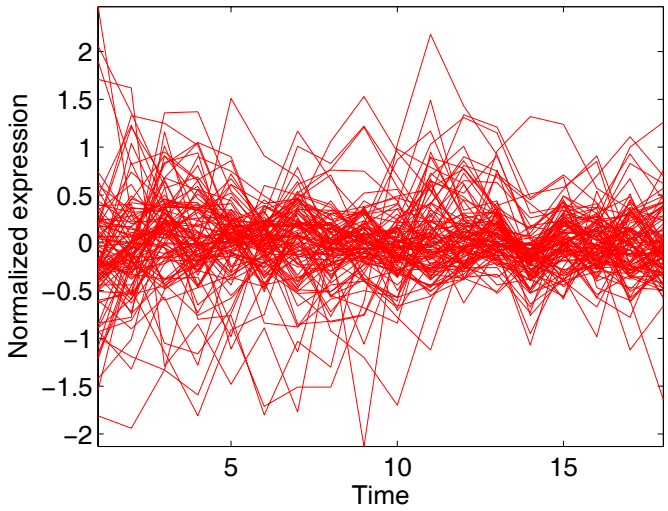


Figure 6.6: Example expression profiles from the gene dataset.

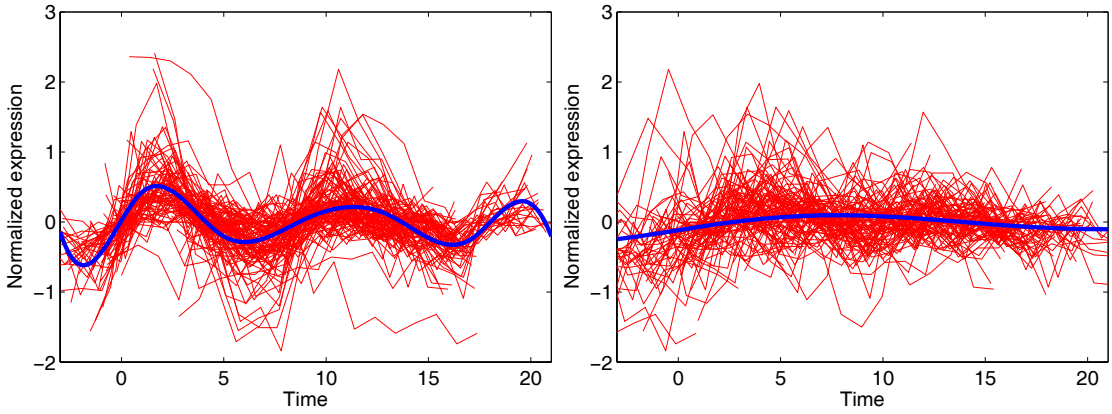


Figure 6.7: Example alignments output from (left) spline-based EM time-translation (4th order) and (right) polynomial-based EM time-translation (5th order) for the gene dataset.

Figure 6.7 shows the alignment of 100 randomly selected genes for both spline EM time-translation (left) and polynomial EM time-translation (right). The mean curves are shown with bolded lines. It is obvious that the spline models are more suited for the alignment problem with this dataset. The 5th order polynomials are not able to discern the cyclic behavior of these profiles.

Table 6.2 shows the MCCV prediction SSE test scores obtained on the gene

Table 6.2: MCCV results for the EM time-alignment and NPP models with the gene expression data. The run-averaged prediction SSE score for each model is shown in column μ. The corresponding standard deviation is shown in σ.

Model	Prediction SSE Scores	
	μ	σ
EM Trans Spline	0.1458	0.0527
NPP Trans Spline	0.1458	0.0487
EM Trans	0.1712	0.0252
EM Affine	0.1734	0.0268
NPP Affine	0.1921	0.0273
NPP Trans	0.2004	0.0248

expression dataset for the EM time-alignment models (polynomial and spline) and with NPP (polynomial and spline). The experiments consisted of 25 runs. At each run, a random sample 150 genes was selected. The models were trained on half of this subset and tested on the remaining half. Curve predictions for the last half of each curve in the test set were made (using the same technique as that described in Section 5.5). The test SSE scores were averaged over the 25 runs and are reported in the table.

The out-of-sample scores show that both of the polynomial EM alignment algorithms out-perform polynomial NPP, with the EM-translation model scoring the best of the non-spline methods. This demonstrates the usefulness of the full probabilistic formulation.

The spline-based translation algorithms, however, show the best performance overall. Since the full posterior expectation in Equation (6.20) cannot be computed for the spline-based EM models, the full probabilistic formulation is not able to be leveraged; and thus, both NPP and spline-based EM-translation show similar performance.

6.5.2 Comparisons with simulated data

In this section, we present experimental results with simulated data. The results show similar characteristics to those from the space-alignment models of the previous chapter. That is, the EM alignment models demonstrate a prowess at recovering the hidden transformations even when faced with large levels of measurement noise (σ^2).

The experiments in this section were based on the exact same data generation procedure described in the experimental results section of the previous chapter. However, a slightly different set of spline models were used. We describe the procedure again for completeness.

The simulated data was generated by random spline models. The spline models were of order 4 with 15 knots uniformly spaced across the interval from 0 to 40. The spline coefficients were randomly drawn from a normal distribution with vector mean $\mathbf{1}$ and scalar variance 64.

The spline models were used to generate two different data sets. One with added normal translations in time used to test the translation-based algorithms, and the other with added affine transformations in time used to test the affine-based algorithms.

The experiments for the translation-based algorithms were run as follows. Twenty-five different sets of 50 random spline curves with added translations in time were generated from a single underlying spline model (the same spline coefficient vector was used in each case). Each of the curves was evaluated across a linear span of 21 time points over the interval of 10 to 30 (the same points were used for each curve).

The translation models were then run on each of the datasets. The output translation parameters from each model were compared to the "true" translations and the mean sum-of-squared error was recorded in each case. This process was repeated at each of of four different levels of the measurement noise σ^2, resulting in

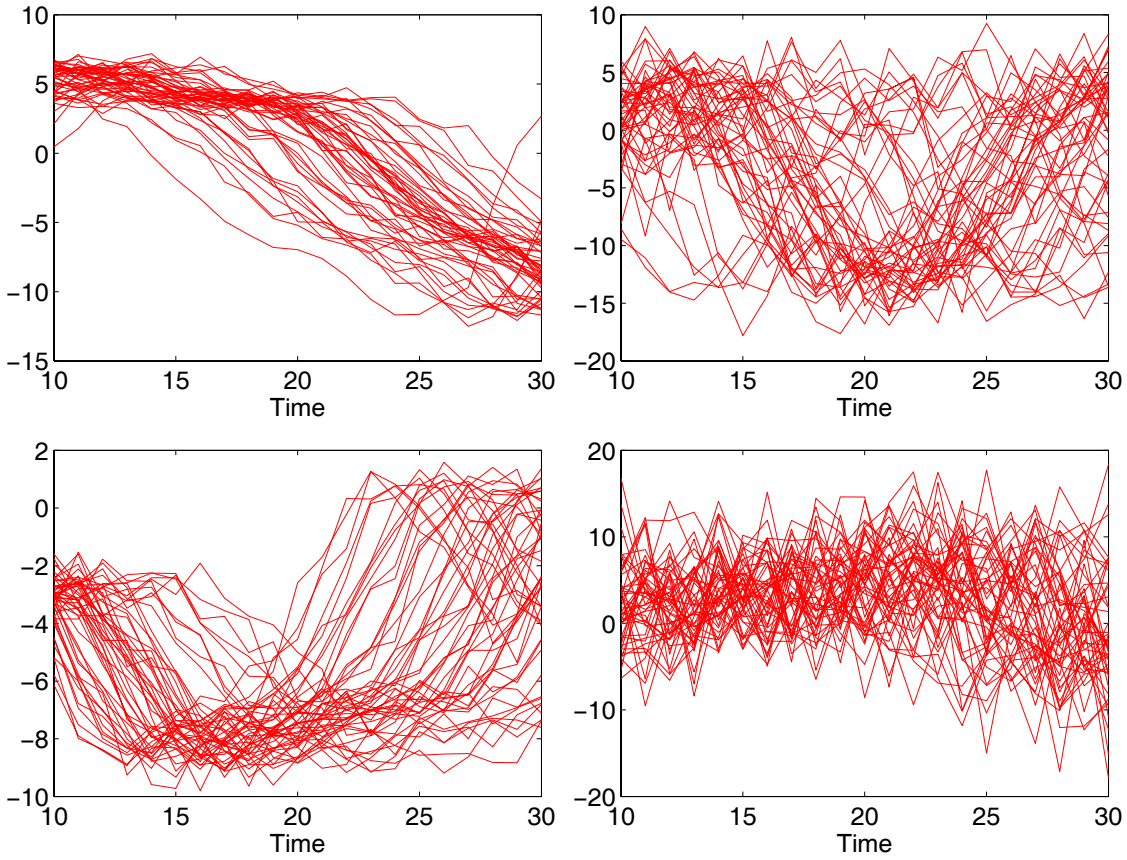


Figure 6.8: Example generated data from random spline models with different levels of noise. The top row shows two data sets generated with random translations in time; the bottom row shows two data sets generated with random affine transformations in time. The plots in the right of the figure contain more noise than those in the left

the evaluation of 100 different subsets of curve data from a single underlying random spline model. Finally, this entire procedure was carried out over three different randomly generated spline models. This resulted in the evaluation of 300 different subsets of data.

The experiments for the affine-based algorithms were carried out in the exact same manner except that random affine transformations in time were added to the curve datasets instead of only translations. Figure 6.8 shows four examples of the randomly generated data. The top-row in the figure shows two translation-based

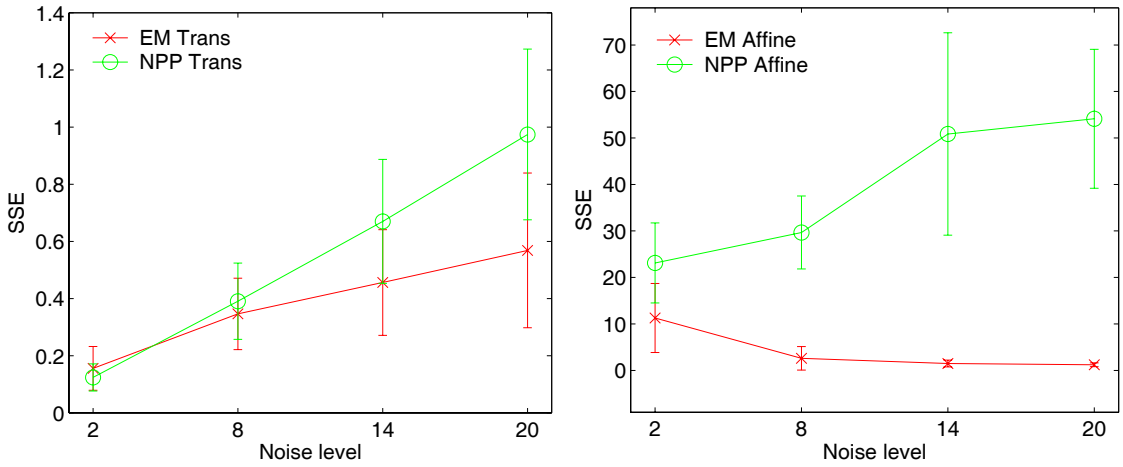


Figure 6.9: Cross-validation results for the EM and NPP time-alignment models: (left) results for translations in time, and (right) results for affine transformations in time. Error bars denote one standard deviation on each side of the plotted mean.

datasets, while the bottom-row shows two affine-based datasets. The plots in the right of the figure contain more noise than those in the left.

The results for the time-translation case can be seen in the left of Figure 6.9. The spline-based alignment models were not used in these comparisons since they match the generated model exactly, and thus they out-perform all other models by default.

At low levels of noise, both of the methods are able to discern the true translations from the measurement noise. However, as the measurement noise grows, NPP has a more difficult time at separating the translations from the curve noise.

Similar but more pronounced results for the affine-transformation case are shown in the right of Figure 6.9. The same relationships between the level of noise and the relative accuracy of NPP hold for this case as well.

6.6 Summary

In this chapter, we introduced a novel probabilistic curve alignment model that allows for the alignment of curve data in time. We derived two specific alignment models, one that allows for translations in time, and another that allows for arbitrary affine transformations in time. Our new approach allows for *continuous* alignments in time by using curve representations of data objects embedded into a probabilistic framework. This formulation naturally leads to iterative EM algorithms that can be used to discover the underlying "true" alignment of a dataset. The main contributions of this chapter can be listed as follows:

- Probabilistic formulation of the alignment problem employing priors over the set of possible transformations (resulting in identifiable alignments).
- Allowance for true continuous time-alignment that does not depend on a set of defined landmarks.
- EM algorithm derivation that formalizes the use of a Mahalanobis distance in a Procrustes-type alignment procedure.
- Derivation for the exact calculation of the expected complete-data log-likelihood function (i.e., the Q-function) in the EM-time alignment algorithm for polynomial regression curve models.
- Use of curve models in the alignment methodology allowing for the handling of irregular sampled data, variable length curves, missing observations, and leveraging of smoothness information.

Chapter 7

Joint Space- and Time-Alignment Models

7.1 Introduction

In this brief chapter, we discuss two extensions to the novel probabilistic alignment methodologies introduced in Chapters 5 and 6. The primary purpose of this chapter is to introduce models that simultaneously allow for alignments in both measurement space and in time. However, this chapter is also used to develop an extended framework that allows for the incorporation of multi-dimensional curves.

This chapter is organized as follows. In Section 7.2, we merge the two space- and time-alignment models into a joint alignment model that simultaneously allows for alignments in space and time. These models provide for maximum curve alignment flexibility. However, as such, they should be used in parallel to analysis of whether or not important aspects of the underlying curve dataset may be inappropriately removed. The derivation in this section is given in a succinct manner since much of the work in the previous two chapters can be directly applied here.

In Section 7.3, the framework is explicitly extended to include the handling of multidimensional curves. A multidimensional curve can contain a vector of observations at each time point instead of the univariate y_{ij}. Thus, the j-th time point of curve $\mathbf{y}_i$ consists of a vector of observations $\mathbf{y}_{ij}$. The extension to multi-dimensional curves is required in Chapters 9 and 10, where we discuss the application of our joint clustering-alignment models to the problem of clustering two-dimensional cyclone trajectories. Finally, the chapter is concluded in Section 7.4 with a summary.

7.2 Joint space- and time-alignment

In this section, we merge the alignment methodologies of the previous two chapters to allow for space as well as time alignment in a single model. The derivation for the complete alignment model allowing for affine transformations in both measurement space and time is given here, with the understanding that the other joint alignment models (e.g., allowing for only translations in measurement space and time) can be considered as special cases of this complete model. Much of the necessary concepts have been extensively detailed in the previous two chapters, and we refer to these chapters as needed.

7.2.1 Model definition

We begin with the model definition for the joint-affine alignment model. The regression models and the transformation priors for the two space- and time-alignment models can be merged together to form one complete specification. This specification can be defined as follows:

$$\mathbf{y}_i = c_i[\![a_i\mathbf{x}_i - b_i]\!]\boldsymbol{\beta} + d_i + \boldsymbol{\epsilon}_i, \tag{7.1}$$

with the time transformation priors

$$a_i \sim \mathcal{N}(1, r^2), \quad b_i \sim \mathcal{N}(0, s^2), \tag{7.2}$$

and the measurement space priors

$$c_i \sim \mathcal{N}(1, u^2) \quad , d_i \sim \mathcal{N}(0, v^2), \tag{7.3}$$

with the noise model

$$\boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}). \tag{7.4}$$

This regression model allows for translation and scaling in time with the inclusion of a_i, b_i, and for translation and scaling in measurement space with c_i, d_i. Further equations can be simplified by grouping the transformation variables into the set $\Phi_i = \{a_i, b_i, c_i, d_i\}$.

Joint, marginals, and log-likelihood

The model specification in Equations (7.1)–(7.4) results in the conditional probability density for $\mathbf{y}_i$ as

$$p(\mathbf{y}_i|\Phi_i) = \mathcal{N}(\mathbf{y}_i|c_i[\![a_i\mathbf{x}_i - b_i]\!]\boldsymbol{\beta} + d_i, \sigma^2\mathbf{I}), \tag{7.5}$$

with the joint taking the form

$$p(\mathbf{y}_i, \Phi_i) = p(\mathbf{y}_i|\Phi_i)p(\Phi_i), \tag{7.6}$$

where

$$p(\Phi_i) = \mathcal{N}(a_i|1, r^2)\mathcal{N}(b_i|0, s^2)\mathcal{N}(c_i|1, u^2)\mathcal{N}(d_i|0, v^2). \tag{7.7}$$

Just as in the individual space and time alignment cases, the marginals can be computed analytically over the space transformation parameters, but not over the time transformation parameters.

For example, the space transformation parameters can be separately integrated out of (7.6) resulting in the marginal of $\mathbf{y}_i$ conditioned only on the time transformation parameters. This conditional marginal takes the form

$$\begin{aligned} p(\mathbf{y}_i|a_i, b_i) &= \int\int p(\mathbf{y}_i, a_i, b_i, c_i, d_i)\, dc_i, dd_i \\ &= \mathcal{N}(\mathbf{y}_i|\mathcal{X}_i\boldsymbol{\beta}, \mathbf{U} + \mathbf{V} - \sigma^2\mathbf{I}), \end{aligned} \tag{7.8}$$

with $\mathbf{U} = u^2\mathcal{X}_i\boldsymbol{\beta}\boldsymbol{\beta}'\mathcal{X}_i' + \sigma^2\mathbf{I}$ and $\mathbf{V} = v^2\mathbb{1} + \sigma^2\mathbf{I}$. The unconditional marginal for $\mathbf{y}_i$; however, cannot be computed analytically. As previously, we use Monte Carlo integration for this task. The resulting unconditional marginal for $\mathbf{y}_i$ is approximated by

$$\begin{aligned} p(\mathbf{y}_i) &= \int\int p(\mathbf{y}_i|a_i, b_i)p(a_i)p(b_i)\, da_i\, db_i \\ &\approx \frac{1}{M}\sum_m p(\mathbf{y}_i|a_i^{(m)}, b_i^{(m)}), \end{aligned} \tag{7.9}$$

where

$$a_i^{(m)} \sim \mathcal{N}(1, r^2), \quad \text{and} \quad b_i^{(m)} \sim \mathcal{N}(0, s^2), \quad \text{for } m = 1, \ldots, M. \tag{7.10}$$

The log-likelihood follows directly from this approximation and takes the form

$$\log p(Y) = \sum_i \log \sum_m p(\mathbf{y}_i|a_i^{(m)}, b_i^{(m)}) - n\log M. \tag{7.11}$$

7.2.2 Joint EM alignment algorithm

In this section, we briefly outline the joint EM alignment algorithm. The steps are quite similar to the particular steps of the individual alignment models. We refer to those steps were appropriate.

In the first step, we regard the unknown set of transformation parameters Φ_i as hidden. This results in setting the hidden-data density to $p(\Phi_i|\mathbf{y}_i)$. In the second step, we define the complete-data log-likelihood function $\mathcal{L}_c$ as the sum over all n curves of the log joint density in (7.6):

$$\mathcal{L}_c = \sum_i \log p(\mathbf{y}_i|\Phi_i)p(\Phi_i). \tag{7.12}$$

The remainder of the algorithm is specified in the E- and M-steps below.

E-step

In the E-step, the posterior $p(\Phi_i|\mathbf{y}_i)$ is calculated and then used to take the posterior expectation of $\mathcal{L}_c$. This results in the Q-function which is maximized in the M-step. The E-step can be broken down into two sub-steps. First, the E-step defined in Section 6.3.2 for the time alignment models is carried out. Then, the E-step defined in Section 5.4.2 for the space alignment models is completed, conditioned on the results from the first sub-step.

Specifically, the posterior $p(\Phi_i|\mathbf{y}_i)$ can be factored as $p(c_i, d_i|a_i, b_i, \mathbf{y}_i)p(a_i, b_i|\mathbf{y}_i)$. The first sub-step involves solving for the vector $(\hat{a}_i, \hat{b}_i)$ representing the multi-dimensional mode of $p(a_i, b_i|\mathbf{y}_i)$ and estimating the associated covariance matrix for $(\hat{a}_i, \hat{b}_i)$. This approximate procedure can be carried out using the methods defined in Section 6.3.2.

The second sub-step consists of using the analytic solution provided in Sec-

tion 5.4.2 to solve for the posterior means and covariances of $p(c_i, d_i|\hat{a}_i, \hat{b}_i, \mathbf{y}_i)$. The only difference from the solutions in Section 5.4.2 is that now the results are conditioned on the values for $\hat{a}_i$ and $\hat{b}_i$ estimated in the first sub-step. This does not change the mathematics since we just replace the spline basis matrix $\mathbf{B}_i$ in the solutions provided in Equations (5.25)–(5.29) with the time transformed matrix $\hat{\mathcal{X}}_i = [\![\hat{a}_i \mathbf{x}_i - \hat{b}_i]\!]$.

For example, the corresponding solutions for the posterior means given in Equation (5.25) are

$$\hat{c}_i = V_{c_i}(\boldsymbol{\beta}' \hat{\mathcal{X}}_i' \mathbf{V}^{-1} \mathbf{y}_i + 1/u^2) \tag{7.13}$$

and

$$\hat{d}_i = V_{d_i}(\mathbf{y}_i - \hat{\mathcal{X}}_i \boldsymbol{\beta})' \mathbf{U}^{-1} \mathbf{1}, \tag{7.14}$$

where V_{c_i} and V_{d_i} are the associated posterior variances.

Calculating the Q-function

The calculation of the Q-function consists of taking the posterior expectation of (7.12) with respect to $p(\Phi_i|\mathbf{y}_i)$ calculated above. The mathematics of this calculation are a bit more complex than with the individual time-alignment model, but the general procedure is the same. Here, we just give the final result. The complete Q-function can be written as

$$Q = \sum_i -\frac{n_i}{2} \log 2\pi\sigma^2 - \frac{1}{2\sigma^2} f(\hat{\Phi}_i|\mathbf{y}_i) + g(\hat{\Phi}_i), \tag{7.15}$$

where

$$f(\hat{\Phi}_i|\mathbf{y}_i) = \left[\left\| \mathbf{y}_i - \hat{c}_i \hat{\mathcal{X}}_i \boldsymbol{\beta} - \hat{d}_i \right\|^2 - 2\mathbf{y}_i' \mathbf{V}_{\mathbf{X}i} \boldsymbol{\beta} + \boldsymbol{\beta}' \mathbf{V}_{\mathbf{X}\mathbf{X}i} \boldsymbol{\beta} + 2\boldsymbol{\beta}' \mathbf{V}_{\mathbf{X}cd} \mathbf{1} + n_i V_{d_i} \right],$$

and

$$
\begin{aligned}
g(\hat{\Phi}_i) &= -\frac{1}{2}\log 2\pi r^2 - \frac{1}{2r^2}\left[(\hat{a}_i - 1)^2 + V_{a_i}\right] - \frac{1}{2}\log 2\pi s^2 - \frac{1}{2s^2}\left[\hat{b}_i^2 + V_{b_i}\right] \\
&\quad -\frac{1}{2}\log 2\pi u^2 - \frac{1}{2u^2}\left[(\hat{c}_i - 1)^2 + V_{c_i}\right] - \frac{1}{2}\log 2\pi v^2 - \frac{1}{2v^2}\left[\hat{d}_i^2 + V_{d_i}\right].
\end{aligned}
$$

The definitions of $\mathbf{V}_{\mathbf{X}i}, \mathbf{V}_{\mathbf{XX}i}$, and $\mathbf{V}_{\mathbf{X}cd}$ are similar to those in the time alignment case and can be derived using the methods described in Section 6.4.2.

M-step

In the M-step, we maximize the Q-function over the complete set of parameters $\{r^2, s^2, u^2, v^2, \sigma^2, \boldsymbol{\beta}\}$. The solutions for the transformation variances $\{r^2, s^2, u^2, v^2\}$ are identical to those in Sections 5.4.2 and 6.4.2 and the reader is encouraged to review those solutions there. We provide the solutions for $\hat{\boldsymbol{\beta}}$ and $\hat{\sigma}^2$ since they are somewhat different than for the previous models:

$$
\hat{\boldsymbol{\beta}} = \left[\sum_i \hat{c}_i^2 \hat{\mathcal{X}}'_i \hat{\mathcal{X}}_i + \mathbf{V}_{\mathbf{XX}i}\right]^{-1} \left[\sum_i \hat{c}_i \hat{\mathcal{X}}'_i (\mathbf{y}_i - \hat{d}_i) + \mathbf{V}'_{\mathbf{X}i}\mathbf{y}_i - \mathbf{V}'_{\mathbf{X}cd}\mathbf{1}\right], \tag{7.16}
$$

and

$$
\hat{\sigma}^2 = 1/N \sum_i \hat{f}(\hat{\Phi}|\mathbf{y}_i), \tag{7.17}
$$

where $N = \sum_i n_i$ and $\hat{f}$ is the function f in which $\boldsymbol{\beta}$ has been replaced by $\hat{\boldsymbol{\beta}}$.

The initialization and convergence procedures are simply borrowed from the respective procedures of the individual alignment models. So that we initialize the algorithm by sampling values for the various posterior means and variances and then proceed to the M-step. The convergence can be detected by monitoring the log-likelihood for a threshold drop in incremental improvement (this is what is done in this thesis). However, the change in the values of the parameters can also be

monitored to determine a convergence criterion that does not require the calculation of the log-likelihood.

7.2.3 Discussion

The joint model described above must solve n four-dimensional search problems associated with finding the four optimal transformation parameters for each of the n curves in the dataset. This leads to an increased propensity to over-fit as compared to either of the individual alignment models. Thus, the extra flexibility that the joint models provide must be evaluated in parallel to the risk of over-fitting.

Nonetheless, the joint alignment models can always be useful tools for exploratory analysis. For example, the resulting learned alignment models can be used to assess the level of transformation variability that is inherent in any particular curve dataset. Whether this variability is associated with random transformation noise or due to underlying scientific causes is a question that should be analyzed carefully.

The complexity of the joint alignment models are similar to the complexity of the individual time alignment models since the discovery of the time-alignment transformations dominate the complexity. The complexity of the joint alignment model is $O(nLI)$ where n is the number of curves, L is some measure of the length of curves in the dataset (e.g., the mean or maximum length of all curves), and I gives the number of iterations of EM.

Typically, L for curve data is much smaller than that for classic time-series data. For example, L is approximately 15 for the application to cyclone trajectories discussed in Chapter 9 and it is about 30 for the application discussed in Chapter 10. Whereas for time-series data, typical values for L might be 200, or even 2000, or more.

The joint alignment models in this chapter were pursued mostly for completeness.

We did not extensively experiment with these methods due to the factorial explosion of the number of experiments that must be carried out to investigate each of the joint alignment models (there are eight of them, four each of spline and polynomial), and each of the individual alignment models (there are eight of them also), plus the different orders of each of the models. In addition, our primary concern was the integration of clustering and alignment. This adds an increased layer of complexity which creates even more model choices that must be evaluated.

As such, we primarily focus on the individual alignment models in the remainder of this thesis. However, we do report some experimental results with the joint alignment models in the application chapters of 9, and 10.

7.3 Multidimensional curves

In this section, we introduce the extension of the curve alignment methodology to multidimensional curves. Thus far, we have implicitly assumed that the curve $\mathbf{y}_i$ consisted of a sequence of univariate curve measurements. However, in many applications these curves are multidimensional. That is, at each time point, a D-dimensional vector of measurements may have been observed.

Often, it is useful to explicitly emphasize that a particular curve is multidimensional. We denote multidimensional curves as $\tilde{\mathbf{y}}_i$. Then, $\tilde{\mathbf{y}}_i$ consists of D columns such that the qth column $\mathbf{y}_i^{(q)}$ contains the sequence of univariate curve measurements for the q-th observation variable. In other words, each $\mathbf{y}_i^{(q)}$ corresponds to a standard curve $\mathbf{y}_i$ packed into a matrix multidimensional curve $\tilde{\mathbf{y}}_i$.

7.3.1 Multidimensional space-alignment regression models

Multidimensional curves can be incorporated into an alignment regression model in a straight-forward manner. For example, the multidimensional curve $\tilde{\mathbf{y}}_i$ can be included into a regression model that allows for translations in measurement space by defining a regression model for each dimension separately. For example, the regression model for the q-th dimension is defined as

$$\mathbf{y}_i^{(q)} = \mathbf{X}_i\boldsymbol{\beta}_q + d_{iq} + \boldsymbol{\epsilon}_i, \quad d_{iq} \sim \mathcal{N}(0, v_q^2), \quad \boldsymbol{\epsilon}_i \sim \mathcal{N}(\mathbf{0}, \sigma_q^2\mathbf{I}), \tag{7.18}$$

where $\boldsymbol{\beta}_q$ gives the regression coefficients for the q-th dimension (i.e., the regression coefficients for the q-th column of $\tilde{\mathbf{y}}_i$), and d_{iq} gives the translation for the q-th dimension. The parameters v_q^2, σ_q^2 can be used to allow for separate variance terms in each dimension. If desired, this dependence can be removed.

The model specification results in the joint density for the multidimensional curve $\tilde{\mathbf{y}}_i$ and the D translation parameters $\{d_{iq}\}$ as follows:

$$p(\tilde{\mathbf{y}}_i, d_{i1}, \ldots, d_{iD}) = \prod_q \mathcal{N}(\mathbf{y}_i^{(q)}|\mathbf{X}_i\boldsymbol{\beta}_q + d_{iq}, \sigma_q^2\mathbf{I})\mathcal{N}(d_{iq}|0, v_q^2). \tag{7.19}$$

The joint density factors for two necessary reasons: (1) conditional independence is assumed between the dimensions of $\tilde{\mathbf{y}}_i$, and (2) each dimension is assumed to have its own set of translation parameters. Absent either of these two conditions, the density would not factor. The marginal density $p(\tilde{\mathbf{y}}_i)$ also factors as $\prod_q p(\mathbf{y}_i^{(q)})$, and so the log-likelihood of $Y = \{\tilde{\mathbf{y}}_i\}_1^n$ takes the form

$$\begin{aligned} \log(Y) &= \sum_i \log p(\tilde{\mathbf{y}}_i) \\ &= \sum_{iq} \log \int p(\mathbf{y}_i^{(q)}|d_{iq})p(d_{iq})\, dd_{iq}. \end{aligned} \tag{7.20}$$

It is apparent from the sum over i, q in the last equation that the log-likelihood is treated as if $n(D-1)$ extra curves have been added to the dataset. The integration in the last equation can be carried out analytically which gives the log-likelihood in the form

$$\log(Y) = \sum_{iq} \log \mathcal{N}(\mathbf{y}_i^{(q)} | \mathbf{X}_i \boldsymbol{\beta}_q, v_q^2 \mathbb{1} + \sigma_q^2 \mathbf{I}). \tag{7.21}$$

7.3.2 Multidimensional time-alignment regression models

It makes sense to have D separate transformation parameters for alignment in measurement space since the individual dimensions may need to be translated and scaled separately. However, for alignment in time, the situation is contrary. It is natural to assume that the dynamic behavior of each dimension has occurred over the same time scale. Therefore, the time-transformation parameters will need to be shared over the D dimensions.

In the time-translation case, each of the one-dimensional curves of $\tilde{\mathbf{y}}_i$ share a single translation parameter b_i. The conditional density of $\tilde{\mathbf{y}}_i$ is

$$\begin{aligned} p(\tilde{\mathbf{y}}_i | b_i) &= \prod_q p(\mathbf{y}_i^{(q)} | b_i) \\ &= \prod_q \mathcal{N}(\mathbf{y}_i^{(q)} | \mathcal{X}_i \boldsymbol{\beta}_q, \sigma_q^2 \mathbf{I}), \end{aligned} \tag{7.22}$$

in which there is only one b_i for all q. The conditional density factors, but the marginal density $p(\tilde{\mathbf{y}}_i)$ does not since the dimensions exhibit dependence through the translation b_i. Therefore, for the log-likelihood of Y we get a different result:

$$\begin{aligned} \log(Y) &= \sum_i \log p(\tilde{\mathbf{y}}_i) \\ &= \sum_i \log \int p(b_i) \prod_q p(\mathbf{y}_i^{(q)} | b_i) \, db_i. \end{aligned} \tag{7.23}$$

The product over the dimensions is now trapped inside of the integral. This results in a slightly more complex problem. However, the approximate log-likelihood can still be computed using Monte Carlo techniques in the same way as before. The approximation can be computed as follows:

$$\log(Y) \approx \sum_i \log \sum_m \prod_q p(\mathbf{y}_i^{(q)} | b_i^{(m)}) - n \log M, \tag{7.24}$$

where

$$b_i^{(m)} \sim \mathcal{N}(0, s^2), \quad \text{for } m = 1, \ldots, M. \tag{7.25}$$

7.3.3 Discussion

The remainder of the derivation for multidimensional curves directly follows from the previous derivations except for the handling of the various individual q subscripts on the dimension-dependent parameters. The result is that each of the alignment models that have so far been defined have an equivalent multidimensional extension.

We demonstrate the application of these multidimensional extensions in Chapters 9 and 10. However, we return to the implicit denoting of multidimensional curves throughout the remainder of this thesis since the extra notation required for multidimensional curves is not needed in general.

7.4 Summary

The main contribution of this chapter was in the introduction of a set of new joint space- and time-alignment models that result from the merging of the individual alignment models of the previous two chapters. We demonstrated an example derivation for these models by focusing on the complete alignment model that allows for

affine transformations in both measurement space and time. The straight-forward extension demonstrated the flexibility of the probabilistic formulation for curve alignment.

This chapter also introduced an extension of our alignment methodology to multidimensional curves. This allows for a vector of observations to be accounted for at each point along a curve. For example, 3D trajectories are curves that consist of three-dimensional vectors of position observations at each time point. The probabilistic formulation easily allows for the incorporation of multidimensional curves by defining an appropriate conditional density for the curve measurements. The conditional density can then be substituted into the alignment methodology without further modification, resulting in the alignment of multidimensional curves.

This chapter concludes the three-part introduction of the novel probabilistic curve alignment methodology used in this thesis. The remainder of this dissertation is concerned with the methods and procedures that result in a joint clustering and alignment methodology for the analysis of sets of smoothly varying curves. We develop such a joint clustering-alignment methodology in Chapter 8, and then present two extensive applications of the joint methodology to the clustering of cyclone trajectories in the following two chapters.

Chapter 8

Curve-Aligned Clustering

8.1 Introduction

In this chapter, we unify the alignment models of the previous three chapters with the clustering models of Chapter 3. The alignment models were specifically developed so that their integration into a model-based clustering algorithm was natural. Most of the foundation work for the model specifications and learning was covered extensively in these previous chapters. As such, the first half of this chapter deals with the integration issues in a broad sense—leaning on the the foundation work of the previous chapters—and the second half details extensive simulated experiments with our new joint alignment and clustering methodology. We present the application of this joint methodology to two "real-world" datasets in Chapters 9, and 10.

There is much prior work focusing on each of the separate clustering and alignment problems as we have pointed out in previous chapters. However, there is only a minimal amount of prior work that has looked at the joint curve clustering-alignment problem itself. The main contribution of this chapter is in the introduction of new

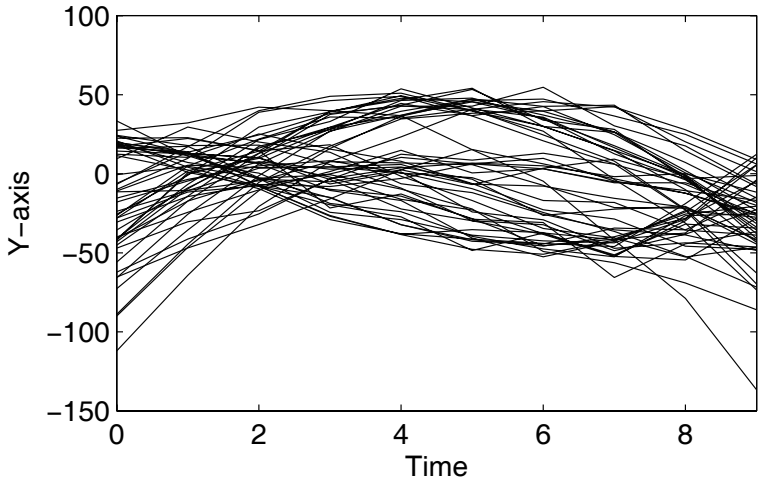


Figure 8.1: Simulated dataset with random translations in time and added measurement noise.

models and algorithms that directly address this problem.

This chapter is organized as follows. We discuss the relevant prior work in Section 8.2 and point-out the novelty of our contribution. In Section 8.3 we derive our joint clustering-alignment framework by demonstrating how to add cluster dependency to the space-affine alignment model of Section 5.4. In Section 8.4 we discuss the extrapolation of this derivation to all of our alignment models. We also make use of this section to discuss some special issues regarding the time-alignment models.

In Section 8.5 we change tack and go from specification to evaluation. In this section we describe the techniques and measures that we use to compare and evaluate our new joint methodology. We discuss cross-validation methods (Burman, 1989) for model selection and derive the test log-likelihood and prediction SSE measures. We follow this in Section 8.6 by detailing the results of systematic evaluations of our new models and methods on simulated data. The evaluations show the effectiveness of our new approach. We conclude the chapter with a summary in Section 8.7.

Before we leave the introduction, we present a simple illustrative example. Figure 8.1 shows a simulated dataset with random translations in time and added

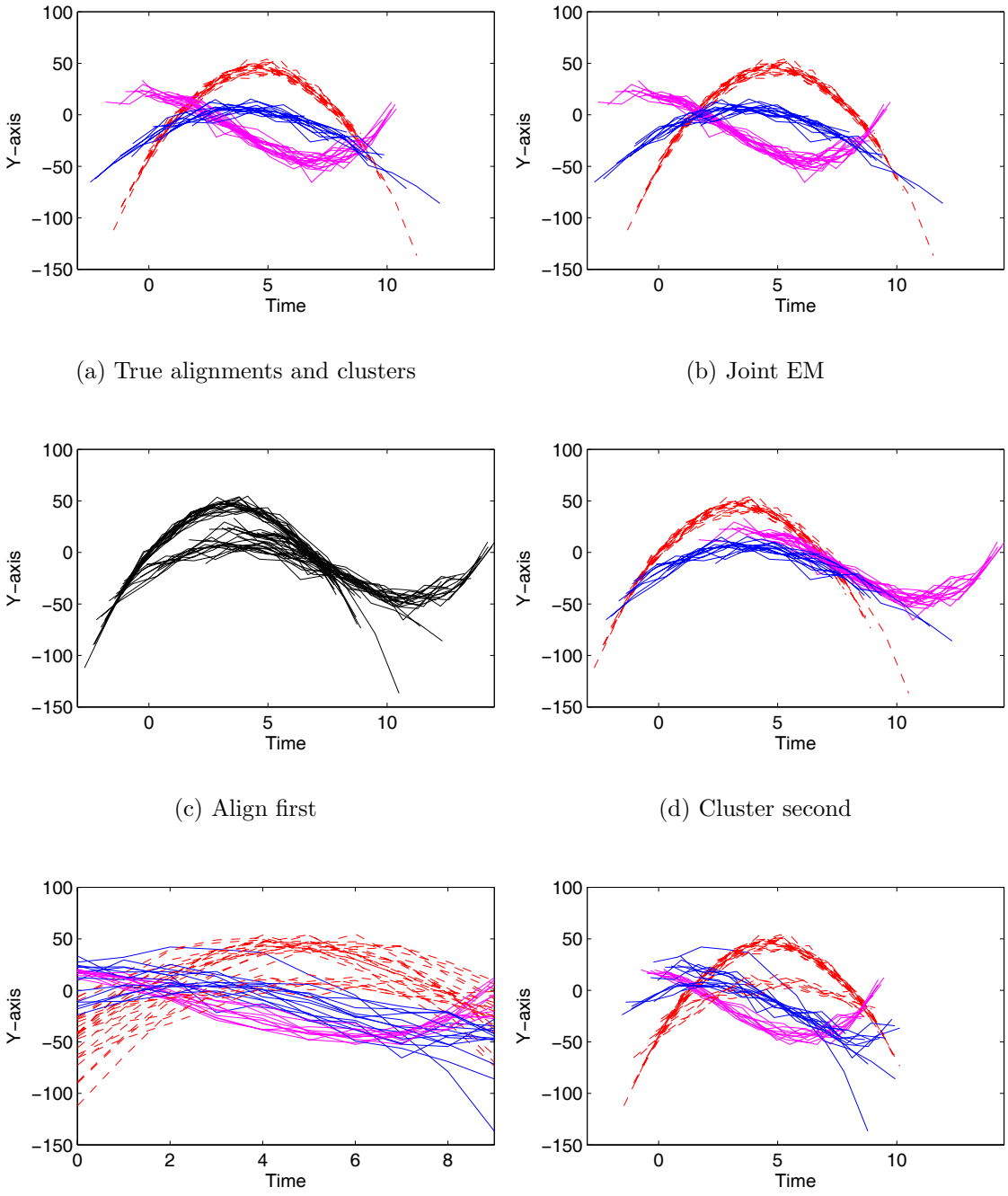
measurement noise. The underlying generative model contains three clusters each described by a cubic polynomial (not shown). Figure 8.2 shows the clustering that results using our new joint EM-based approach (described below) and two sequential approaches (align first, then cluster; and cluster first, then align).

Figure 8.2(a) shows the true alignments and clustering that each method must uncover. The output of our joint approach is shown in Figure 8.2(b). We can see that it closely resembles the true picture in Figure 8.2(a). The clustering and the alignment are both accurate. Figure 8.2(c) shows what happens if you align the data first. The resulting clustering is shown adjacent to this in Figure 8.2(d). The alignment is clearly incorrect; however, many of the classifications are correct (there are only a few misclassified examples). Figure 8.2(e) shows the result of clustering first. The within-cluster alignment is shown adjacent to this in Figure 8.2(f). This sequential approach results in significant misclassification and incorrect alignment.

The example demonstrates that the best approach to the clustering problem is to apply a joint clustering-alignment methodology. Either of the sequential approaches will most likely fail in one way or another given significant misalignment in clustered data.

8.2 Prior work

Despite the dearth of joint curve clustering and alignment work, there has been related work in other areas. One area where there has been some success in simultaneous alignment and clustering is in the modelling of image data using mixture densities. Although there is no notion of sets of curves or curve models, the joint architecture is related. For example, the *transformed mixture of Gaussians* (TMG) model uses a probabilistic setup and an EM algorithm to learn mixtures of images



(a) True alignments and clusters

(b) Joint EM

(c) Align first

(d) Cluster second

(e) Cluster first

(f) Align second

Figure 8.2: Comparison of joint EM and sequential clustering-alignment: (top-row) ground truth and joint EM, (middle-row) align and then cluster, (bottom-row) cluster and then align. The original data is shown in Figure 8.1

jointly subject to various forms of linear transformations (Frey & Jojic, 1999, 2002, 2003). However, this model only considers *discrete* sets of transformations that shift pixels in images, whereas we are focused on curve modelling that allows for arbitrary *continuous* alignment in time. In other work, Jojic et al. (2000) demonstrate that the probabilistic framework allows for a natural extension of the TMG to *transformed hidden Markov* models which they use to cluster video sequences in time.

A related area where there has been much work on a similar problem is in medical imaging. Aligning of 2D and 3D shapes in images is important in medical imaging. It is common to represent such shapes by a set of points (commonly called a *point-set*). The representation is readily applicable to the methods of *statistical shape analysis* which is why the technique is a popular choice (Cootes et al., 1994; Neumann & Lorenz, 1998; Staib & Duncan, 1992). A problem with this approach, however, is that the *correspondences* between different point-sets are in general unknown. Without knowledge of the correspondences, the alignment of shapes is difficult.

The joint correspondence-alignment problem has similarities to the joint cluster-alignment problem that we address here. There has been several papers that have dealt with this problem in one way or another. For example, Cross and Hancock (1998) develop a dual-step EM algorithm formulated using a graph-based representation of correspondence constraints that simultaneously solves for the correspondences and the alignment parameters. Several *softassign* influenced algorithms have been developed that also handle this problem directly (Rangarajan et al., 1997; Gold et al., 1998). The joint correspondence-alignment problem is solved in an iterative optimization framework that employs deterministic annealing in which softassign is used to rid the objective function of problematic penalty terms. Chui et al. (2004) follow a similar tack, but instead formulate the problem in probabilistic terms and then use an iterative process with an embedded deterministic annealing procedure

employing clustering as a de-noising process. Xue et al. (2001) apply an iterative fuzzy algorithm that jointly finds the correspondences and affine transformation parameters for the point-sets.

In many ways, these correspondence-alignment algorithms can be seen as similar to our joint curve clustering-alignment algorithms. However, the various solutions presented in these papers are quite specific to the problem at hand. The higher dimensional space requires more complex procedures and external constraints that are needed to make the problem manageable. In contrast, the novelty of our approach lies in its self-contained probabilistic formulation in which priors on transformations naturally provide for clustering models that are transformation-invariant without any extra specialized constraints or procedures. In addition, our focus is on the application to curve analysis and those curve models that are used to represent them. Our framework naturally leads to easy-to-understand EM algorithms that are easily coded-up in MATLAB using only the provided routines.

In related work (Chudova et al., 2003, 2003), we have shown how transformation-invariant curve clustering can be incorporated into a Gaussian mixtures framework. In this work, we did not model transformations as random variables with prior probability densities. Instead we treated them as extra variables to be optimized separately as an addition to the standard Gaussian mixtures framework. We also showed how a general Bayesian network can be used for simultaneous local (non-linear) time-warping and clustering of curve data.

8.3 Adding cluster dependence

We can view the integration of alignment and clustering as either adding cluster dependence to the alignment models or as adding alignment to the clustering models.

Because of the way in which we defined the alignment models, the viewpoints are equivalent; however, we present this section from the former viewpoint—we add cluster dependence to the alignment models.

Instead of deriving the joint alignment-clustering algorithms for each of the models defined in Chapters 5 and 6, we instead present an example derivation for a single model which then allows the reader to extrapolate the methodology to the remaining cases. We chose to derive the joint algorithm for the space-affine model of Section 5.4.1 since it includes the space-translation model as a special case. The derivations for the time-alignment models follow closely with what we present here; however, there are a couple of issues which require special attention in the time-alignment case. We discuss these in Section 8.4.

To this end, we begin with the space-affine spline alignment model defined in Section 5.4.1 which we reproduce here as Equation (8.1):

$$\mathbf{y}_i = c_i \mathbf{B}_i \boldsymbol{\beta} + d_i + \boldsymbol{\epsilon}_i, \qquad \boldsymbol{\epsilon}_i \sim N(\mathbf{0}, \sigma^2 \mathbf{I}), \tag{8.1}$$

with priors $c_i \sim \mathcal{N}(1, u^2)$ and $d_i \sim \mathcal{N}(0, v^2)$. We add cluster dependence to this by repeating this model over K different clusters. This results in affixing k to each of the parameters $\{\boldsymbol{\beta}, \sigma^2, u^2, v^2\}$ to arrive at the cluster-dependent regression model:

$$\mathbf{y}_i = c_i \mathbf{B}_i \boldsymbol{\beta}_k + d_i + \boldsymbol{\epsilon}_i, \qquad \boldsymbol{\epsilon}_i \sim N(\mathbf{0}, \sigma_k^2 \mathbf{I}), \tag{8.2}$$

with priors $c_i \sim \mathcal{N}(1, u_k^2)$ and $d_i \sim \mathcal{N}(0, v_k^2)$. Instead of placing dependence on every parameter, we may instead only wish to treat some subset of them. For example, we may posit that there are only single u^2,v^2 parameters shared among all of the clusters and only $\boldsymbol{\beta}_k$ and σ_k^2 provide the cluster-specific behavior. We describe the general case in which every parameter is dependent on k.

We denote the cluster-dependent conditional density for $\mathbf{y}_i$ using a subscripted k as $p_k(\mathbf{y}_i|c_i, d_i)$ and write

$$p_k(\mathbf{y}_i|c_i, d_i) = \mathcal{N}(c_i\mathbf{B}_i\boldsymbol{\beta}_k + d_i, \sigma_k^2\mathbf{I}). \tag{8.3}$$

When cluster membership is unknown, we have a conditional mixture distribution in the form

$$p(\mathbf{y}_i|c_i, d_i) = \sum_k \alpha_k \, p_k(\mathbf{y}_i|c_i, d_i), \tag{8.4}$$

with non-negative mixture weights α_k that sum to one. This is a mixture model that assumes the transformation parameters c_i, d_i are known. The next step is to write down the densities for when both (or one or the other) of the transformation parameters are unknown.

8.3.1 Joint, marginals and log-likelihood

The updated k-dependent joint and marginal densities (see Section 5.4.1) follow in a straightforward manner. Essentially, there are now K separate joints and marginals, one of each for each cluster. The cluster-dependent joint takes the product form

$$\begin{aligned} p_k(\mathbf{y}_i, c_i, d_i) &= p_k(\mathbf{y}_i|c_i, d_i)p_k(c_i)p_k(d_i) \\ &= \mathcal{N}(\mathbf{y}_i|c_i\mathbf{B}_i\boldsymbol{\beta}_k + d_i, \sigma_k^2\mathbf{I})\mathcal{N}(c_i|1, u_k^2)\mathcal{N}(d_i|0, v_k^2). \end{aligned} \tag{8.5}$$

We can integrate over each of the transformation parameters in the joint model to obtain all of the cluster-dependent marginal densities. For example, the cluster-dependent marginal of $\mathbf{y}_i$ given c_i is

$$p_k(\mathbf{y}_i|c_i) = \int p_k(\mathbf{y}_i, d_i|c_i) \, dd_i$$

$$= \mathcal{N}(\mathbf{y}_i | c_i \mathbf{X}_i \boldsymbol{\beta}_k, \mathbf{V}_k), \qquad \mathbf{V}_k = v_k^2 \mathbb{1} + \sigma_k^2 \mathbf{I}. \tag{8.6}$$

Notice that now the covariance matrix $\mathbf{V}_k$ is indexed by cluster k. The cluster-dependent marginal of $\mathbf{y}_i$ given d_i is

$$\begin{aligned} p_k(\mathbf{y}_i | d_i) &= \int p_k(\mathbf{y}_i, c_i | d_i)\, dc_i \\ &= \mathcal{N}(\mathbf{y}_i | \mathbf{X}_i \boldsymbol{\beta}_k + d_i, \mathbf{U}_k), \qquad \mathbf{U}_k = u_k^2 \mathbf{X}_i \boldsymbol{\beta}_k \boldsymbol{\beta}_k' \mathbf{X}_i' + \sigma_k^2 \mathbf{I}. \end{aligned} \tag{8.7}$$

The remaining cluster-dependent marginal results from integration over both of the transformation parameters as in

$$\begin{aligned} p_k(\mathbf{y}_i) &= \int\int p_k(\mathbf{y}_i, c_i, d_i)\, dc_i\, dd_i \\ &= \mathcal{N}(\mathbf{y}_i | \mathbf{B}_i \boldsymbol{\beta}_k, \mathbf{U}_k + \mathbf{V}_k - \sigma_k^2 \mathbf{I}). \end{aligned} \tag{8.8}$$

Since these equations all depend on k, we then naturally obtain mixture densities when cluster membership is unknown. For example, the mixture density for the unconditional marginal of $\mathbf{y}_i$ is

$$p(\mathbf{y}_i) = \sum_k \alpha_k\, p_k(\mathbf{y}_i). \tag{8.9}$$

The marginal density then leads directly to the definition of the log-likelihood for the set $Y = \{\mathbf{y}_i\}_1^n$ of n curves. The log-likelihood is the sum over all n curves of the log marginal of $\mathbf{y}_i$:

$$\log p(Y) = \sum_i \log \sum_k \alpha_k\, \mathcal{N}(\mathbf{y}_i | \mathbf{B}_i \boldsymbol{\beta}_k, \mathbf{U}_k + \mathbf{V}_k - \sigma_k^2 \mathbf{I}). \tag{8.10}$$

8.3.2 Joint EM clustering-alignment algorithm

The derivation of the joint clustering-alignment algorithm follows closely with that in Section 5.4.2 while handling the hidden cluster memberships. In this section, we again follow our four-step template for describing EM algorithms.

In the first step, we let z_i give the cluster membership for curve $\mathbf{y}_i$. We then regard the transformation parameters $\{c_i, d_i\}$ as well as the cluster memberships $\{z_i\}$ as being hidden. The hidden-data density then becomes the posterior $p(z_i, c_i, d_i|\mathbf{y}_i)$.

In the second step, we define the complete-data log-likelihood function as the joint log-likelihood of Y and the hidden data $\{c_i, d_i, z_i\}$. This can be written as the sum over all n curves of the log of the product of α_{z_i} and the cluster-dependent joint density in (8.5). This function takes the form

$$\mathcal{L}_c = \sum_i \log \alpha_{z_i} p_{z_i}(\mathbf{y}_i|c_i, d_i)\, p_{z_i}(c_i)\, p_{z_i}(d_i). \tag{8.11}$$

The remaining two steps (the E- and M-steps) are defined next.

E-step

In the E-step we calculate the joint posterior $p(z_i, c_i, d_i|\mathbf{y}_i)$ and then use this to take expectations of the complete-data log-likelihood function in (8.11). We can make use of the previous work in Section 5.4.2 by factoring the posterior $p(z_i, c_i, d_i|\mathbf{y}_i) = p(c_i, d_i|z_i, \mathbf{y}_i)p(z_i|\mathbf{y}_i)$ and taking expectations first with respect to $p(c_i, d_i|z_i, \mathbf{y}_i)$ and then with respect to $p(z_i|\mathbf{y}_i)$.

We begin with the first factor of the posterior. The form of this factor as

$$\begin{aligned} p(c_i, d_i|z_i = k, \mathbf{y}_i) &\propto p_k(\mathbf{y}_i|c_i, d_i)p_k(c_i)p_k(d_i) \\ &\propto \exp\left\{-\left\|\mathbf{y}_i - c_i\mathbf{B}_i\boldsymbol{\beta}_k - d_i\right\|^2/2\sigma_k^2 - (c_i-1)^2/2u_k^2 - d_i^2/2v_k^2\right\} \end{aligned}$$

is identical to that in (5.24) except that now we have K different posteriors, one for each value of z_i. The result is that these K posteriors can each be identified with an individual bi-variate normal posterior (as in Section 5.4.2) which is fully described by its set of means, variances, and covariances. For completeness we give identical equations to those in (5.25)–(5.29) but include cluster dependence. For the posterior means we have

$$\hat{c}_{ik} = V_{c_{ik}}(\boldsymbol{\beta}_k' \mathbf{B}_i' \mathbf{V}_k^{-1} \mathbf{y}_i + 1/u_k^2), \tag{8.12}$$

$$\hat{d}_{ik} = V_{d_{ik}}(\mathbf{y}_i - \mathbf{B}_i \boldsymbol{\beta}_k) \mathbf{U}_k^{-1} \mathbf{1}, \tag{8.13}$$

the posterior variances are

$$V_{c_{ik}} = (\boldsymbol{\beta}_k' \mathbf{B}_i' \mathbf{V}_k^{-1} \mathbf{B}_i \boldsymbol{\beta}_k + 1/u_k^2)^{-1}, \tag{8.14}$$

$$V_{d_{ik}} = (\mathbf{1}' \mathbf{U}_k^{-1} \mathbf{1} + 1/v_k^2)^{-1}, \tag{8.15}$$

and finally the posterior covariance is

$$V_{c_{ik} d_{ik}} = -u_k v_k \sqrt{\lambda_k V_{c_{ik}} V_{d_{ik}}}\ \mathbf{1}' \mathbf{B}_i \boldsymbol{\beta}_k. \tag{8.16}$$

The equation for λ_k is

$$\lambda_k = (u_k^2 \boldsymbol{\beta}_k' \mathbf{B}_i' \mathbf{B}_i \boldsymbol{\beta}_k + \sigma_k^2)^{-1} (n_i v_k^2 + \sigma_k^2)^{-1}.$$

All that remains is to calculate the remaining factor of the posterior, namely, $p(z_i = k|\mathbf{y}_i)$. This is the membership probability w_{ik} that $\mathbf{y}_i$ was generated by cluster $z_i = k$. Its calculation is straightforward:

$$w_{ik} = p(z_i = k|\mathbf{y}_i) \quad \propto \quad p(\mathbf{y}_i|z_i = k) p(z_i = k)$$

$$= \alpha_k \, p_k(\mathbf{y}_i) \tag{8.17}$$

$$= \alpha_k \, \mathcal{N}(\mathbf{y}_i | \mathbf{B}_i \boldsymbol{\beta}_k, \mathbf{U}_k + \mathbf{V}_k - \sigma_k^2 \mathbf{I}). \tag{8.18}$$

Calculating the Q-function

The calculation of the Q-function consists of taking the posterior expectation of (8.11) with respect to $p(c_i, d_i | \mathbf{y}_i, z_i) p(z_i | \mathbf{y}_i)$ calculated above. We can simplify this operation by first taking the posterior expectation of the noise term $\boldsymbol{\epsilon}_i = (\mathbf{y}_i - c_i \mathbf{B}_i \boldsymbol{\beta}_k - d_i)$ with respect to $p(c_i, d_i | \mathbf{y}_i, z_i = k)$. We can write this expectation and the related variance as

$$\hat{\boldsymbol{\epsilon}}_{ik} = \mathrm{E}[\boldsymbol{\epsilon}_i | \mathbf{y}_i, z_i = k] = \mathbf{y}_i - \hat{c}_{ik} \mathbf{B}_i \boldsymbol{\beta}_k - \hat{d}_{ik}, \tag{8.19}$$

$$V_{\boldsymbol{\epsilon}_{ik}} = \mathrm{Var}[\boldsymbol{\epsilon}_i | \mathbf{y}_i, z_i = k] = V_{c_{ik}} \mathbf{B}_i \boldsymbol{\beta}_k \boldsymbol{\beta}_k' \mathbf{B}_i' + V_{d_{ik}} \mathbb{1} + 2 V_{c_{ik} d_{ik}} \mathbf{B}_i \boldsymbol{\beta}_k \mathbf{1}'. \tag{8.20}$$

We also note that $\mathrm{E}[\boldsymbol{\epsilon}_i' \boldsymbol{\epsilon}_i | \mathbf{y}_i, z_i = k] = \hat{\boldsymbol{\epsilon}}_{ik}' \hat{\boldsymbol{\epsilon}}_{ik} + \mathrm{tr}(V_{\boldsymbol{\epsilon}_{ik}})$. With this, we now take the expectation of (8.11) with respect to $p(z_i, c_i, d_i | \mathbf{y}_i)$. First we expand the normal densities and carry the expectation across the non-random terms.

$$\begin{aligned} Q &= \sum_{ik} w_{ik} \int \int [\log \alpha_k \, p_k(\mathbf{y}_i | c_i, d_i) p_k(c_i) p_k(d_i)] \; p(c_i, d_i | \mathbf{y}_i, k) \, dc_i \, dd_i \\ &= \sum_{ik} w_{ik} \log \alpha_k - \frac{w_{ik} n_i}{2} \log 2\pi\sigma_k^2 - \frac{w_{ik}}{2\sigma_k^2} \mathrm{E}\left[\boldsymbol{\epsilon}_i' \boldsymbol{\epsilon}_i | \mathbf{y}_i, k\right] \\ &\quad - \frac{w_{ik}}{2} \log 2\pi u_k^2 - \frac{w_{ik}}{2u_k^2} \mathrm{E}\left[(c_i - 1)^2 | \mathbf{y}_i, k\right] \\ &\quad - \frac{w_{ik}}{2} \log 2\pi v_k^2 - \frac{w_{ik}}{2v_k^2} \mathrm{E}\left[d_i^2 | \mathbf{y}_i, k\right]. \end{aligned} \tag{8.21}$$

We are left with taking expectations that only require substitution of known sufficient statistics from the E-step. The substitutions result in the final equation for the Q-

function:

$$\begin{aligned} Q &= \sum_{ik} w_{ik} \log \alpha_k - \frac{w_{ik} n_i}{2} \log 2\pi\sigma_k^2 - \frac{w_{ik}}{2\sigma_k^2} \left[\hat{\boldsymbol{\epsilon}}'_{ik}\hat{\boldsymbol{\epsilon}}_{ik} + \mathrm{tr}(V_{\boldsymbol{\epsilon}_{ik}})\right] \\ &\quad -\frac{w_{ik}}{2} \log 2\pi u_k^2 - \frac{w_{ik}}{2u_k^2}\left[(\hat{c}_{ik}-1)^2 + V_{c_{ik}}\right] \\ &\quad -\frac{w_{ik}}{2} \log 2\pi v_k^2 - \frac{w_{ik}}{2v_k^2}\left[\hat{d}_{ik}^2 + V_{d_{ik}}\right]. \end{aligned} \tag{8.22}$$

M-step

The M-step is identical to that in Section 5.4.2 except for the appearance of the membership probabilities and the solutions for the mixture weights $\hat{\alpha}_k$. The parameter re-estimation equations can be written for $1 \leq k \leq K$ as follows. For the mixture weights we have

$$\hat{\alpha}_k = \frac{\sum_i w_{ik}}{n}. \tag{8.23}$$

The transformation variances have the solutions

$$\hat{u}_k^2 = \frac{1}{\sum_i w_{ik}} \sum_i w_{ik}\left[(\hat{c}_{ik}-1)^2 + V_{c_{ik}}\right], \tag{8.24}$$

$$\hat{v}_k^2 = \frac{1}{\sum_i w_{ik}} \sum_i w_{ik}\left[\hat{d}_{ik}^2 + V_{d_{ik}}\right]. \tag{8.25}$$

The measurement noise can be estimated by

$$\hat{\sigma}_k^2 = \frac{1}{\sum_i w_{ik} n_i} \sum_i w_{ik}\left[\hat{\boldsymbol{\epsilon}}'_{ik}\hat{\boldsymbol{\epsilon}}_{ik} + \mathrm{tr}(V_{\boldsymbol{\epsilon}_{ik}})\right]; \tag{8.26}$$

and finally, we estimate the regression coefficients using

$$\hat{\boldsymbol{\beta}}_k = \left[\sum_i w_{ik}\mathbf{B}_i'\mathbf{B}_i(\hat{c}_{ik}^2 + V_{c_{ik}})\right]^{-1} \sum_i w_{ik}\mathbf{B}_i'(\hat{c}_{ik}(\mathbf{y}_i - \hat{d}_{ik}) - V_{c_{ik}d_{ik}}). \tag{8.27}$$

Essentially, the joint EM algorithm simultaneously solves K different alignment problems in the E-step and fits K different curve models in the M-step. The role of the membership probabilities is to determine how much influence each curve has in each of the K fitting problems in the M-step. If you consider that one can hold up and look at each one of the K joint alignment and fitting problems as parallel panes of glass, then each pane of glass can be seen as a separate EM alignment algorithm (from Chapter 5, 6, or 7) with the membership probabilities determining which panes of glass are etched with which curves and to what degree.

Having derived the joint EM algorithm for affine transformations in measurement space, we now discuss the extrapolation of the derivation to the other alignment models.

8.4 Extrapolation to other models

In this section we discuss the extrapolation of the derivation of the previous section to the remaining EM alignment models introduced in the previous chapters. Special issues regarding the memberships probabilities and the log-likelihood in the time-alignment case are also discussed.

8.4.1 General derivation of joint clustering algorithms

The derivation of the joint EM clustering-alignment algorithms for the other models defined in Chapters 5, 6, and 7 can be extrapolated from the example procedure described in Section 8.3. The basic idea is to represent each cluster with its own probabilistic alignment model and then form a mixture over these clusters. The joint EM algorithm is then identical to the specific EM alignment algorithm but making allowances for the membership probabilities and the mixture weights. This

Table 8.1: Labels used to refer to each of the joint clustering-alignment models based on the PRM model.

Name	Measurement Space		Time	
	Trans	Affine	Trans	Affine
PRM_TM	X			
PRM_AM		X		
PRM_TT			X	
PRM_AT				X
PRM_TM_TT	X		X	
PRM_AM_TT		X	X	
PRM_TM_AT	X			X
PRM_AM_AT		X		X

natural integration is an important asset of the model-based probabilistic formulation introduced in this thesis. Table 8.1 lists the abbreviations that are used in the rest of this thesis to refer to each combination of clustering and alignment model. Only the versions for the PRM model are listed in the table, but there are an equivalent set of abbreviations for the SRM model also. There are some issues that require special attention in the integration process. We discuss these next.

Calculating memberships and the log-likelihood

The calculations for the membership probabilities and the log-likelihood are straightforward when only transformations in measurement space are considered. But when transformations in time are allowed, we discover that the equations do not reveal closed-form solutions. The reason for this is because both the membership probabilities and the log-likelihood are functions of the cluster-dependent marginal density for $\mathbf{y}_i$, a density which cannot be computed analytically. For the time-translation case, this density is

$$p_k(\mathbf{y}_i) \;=\; \int p_k(\mathbf{y}_i|b_i)p_k(b_i)\,db_i$$

$$= \int \mathcal{N}\left(\mathbf{y}_i \middle| [\![\mathbf{x}_i - b_i]\!]\boldsymbol{\beta}_k, \sigma_k^2\mathbf{I}\right)\mathcal{N}(b_i|0, v_k^2)\, db_i.$$

We saw in Section 6.3.1 that this integral cannot be solved for analytically. Instead, because sampling from the prior $p_k(b_i)$ is rather easy, we can use Monte Carlo integration. The approximation becomes

$$\begin{aligned} p_k(\mathbf{y}_i) &= \int p_k(\mathbf{y}_i|b_i)\, p_k(b_i)\, db_i \\ &\approx \frac{1}{M}\sum_m p_k(\mathbf{y}_i|b_{ik}^{(m)}) \end{aligned} \tag{8.28}$$

where

$$b_{ik}^{(m)} \sim \mathcal{N}(0, s_k^2), \quad \text{for } m = 1, \ldots, M.$$

Plugging this into our generic equation for membership probabilities in (8.17) leads to the approximation for the memberships w_{ik}:

$$\begin{aligned} w_{ik} &\propto \alpha_k\, p_k(\mathbf{y}_i) \\ &\approx \alpha_k \frac{1}{M}\sum_m p_k(\mathbf{y}_i|b_{ik}^{(m)}). \end{aligned} \tag{8.29}$$

The accuracy of this approximation is discussed below.

Like the membership probabilities, the log-likelihood must also be approximated. The log-likelihood is the sum over all n curves of the unconditional marginal of $\mathbf{y}_i$ (unconditional of cluster membership). The approximate unconditional marginal of $\mathbf{y}_i$ is a mixture density of the form

$$\begin{aligned} p(\mathbf{y}_i) &= \sum_k \alpha_k\, p_k(\mathbf{y}_i) \\ &\approx \frac{1}{M}\sum_{k,m} \alpha_k\, p_k(\mathbf{y}_i|b_{ik}^{(m)}). \end{aligned} \tag{8.30}$$

The mixture density arises because cluster membership must be summed out of the equation. The approximate log-likelihood of Y follows directly from (8.30):

$$\begin{aligned} \log p(Y) &= \sum_i \log p(\mathbf{y}_i) \\ &\approx \sum_i \log \sum_{k,m} \alpha_k \, p_k(\mathbf{y}_i | b_{ik}^{(m)}) - n \log M. \end{aligned} \tag{8.31}$$

Approximation accuracy

In this section, we look at the accuracy of the approximations used for both the membership probabilities and the log-likelihood. Although they are both based on the same underlying density, the nature of the problem is different in each case.

In practice, the membership probabilities do not require overly accurate approximations for successful application of EM. Membership probabilities often tend toward polar opposites of 1 and 0. This effect is particularly noticeable with curve clustering since each "individual" consists of many different points that share membership along the curve. This tendency eases the difficulty in estimating the memberships. The algorithm is not likely to be negatively affected by using approximate membership probabilities unless the error is relatively large.

Figure 8.3 shows an example that depicts the evolution of the approximation accuracy of the membership probabilities during a run of EM. The data was generated from a three-cluster time-translation model. Each of the clusters was represented by a polynomial of order one (i.e., linear). For such a model, an exact calculation of the E-step (and the membership probabilities in particular) is possible.

Each of the 12 "boxes" in the figure contains two different plots separated by a thick horizontal axis line. The upper plot graphs the approximate and the exact values of w_{ik} on the y-axis for each of the curves along the x-axis (the approximate curve cannot be seen for the most part since it tracks the exact value so closely).

The lower plot graphs the error of the approximation on the y-axis for each of the same curves along the x-axis (this is used as a proxy for not being able to distinguish between the approximate and exact curves in the upper plot). Although the x-axis is discrete (i.e., there is a curve 1 and a curve 2, but no curve 1.5), the graph is displayed as if it was continuous.

The boxes are organized by cluster and iteration. For example, all the boxes related to cluster 1 are given in the first column. If we scan down this column, we are able to see the time-evolution of the memberships for cluster 1. Initially, after the E-step in the first iteration, none of the first 23 curves have any membership in this cluster, while the remaining curves have about 0.5 membership. The approximation error is almost zero at this point. The second and third iterations show an increase in the approximation error as the curves move from cluster to cluster. As the memberships approach 1 or 0, the error dies back down. There were actually 21 iterations for this run of EM. However, the memberships were essentially fixed beyond iteration 6. The clustering is almost perfect in this case (curve 45 belongs to cluster 1, but it was put into cluster 2).

A similar picture can be seen in the second column for cluster 2. The approximation error initially grows and then settles back down. Cluster 3 shows a unique picture since its memberships were learned immediately in the very first iteration. It never shows any approximation error at all.

Overall the approximation is quite good. We are not able to see any significant deviation of the approximate curve from the exact curve. Where we do see deviation is when the EM algorithm gets "stuck" in bad local maxima. That is, when the resulting EM solution is not very good. For example, Figure 8.4 shows a particularly bad iteration for a run of EM on the same exact dataset. The local maximum of the likelihood surface in which the algorithm is "trapped" does not allow EM to discover

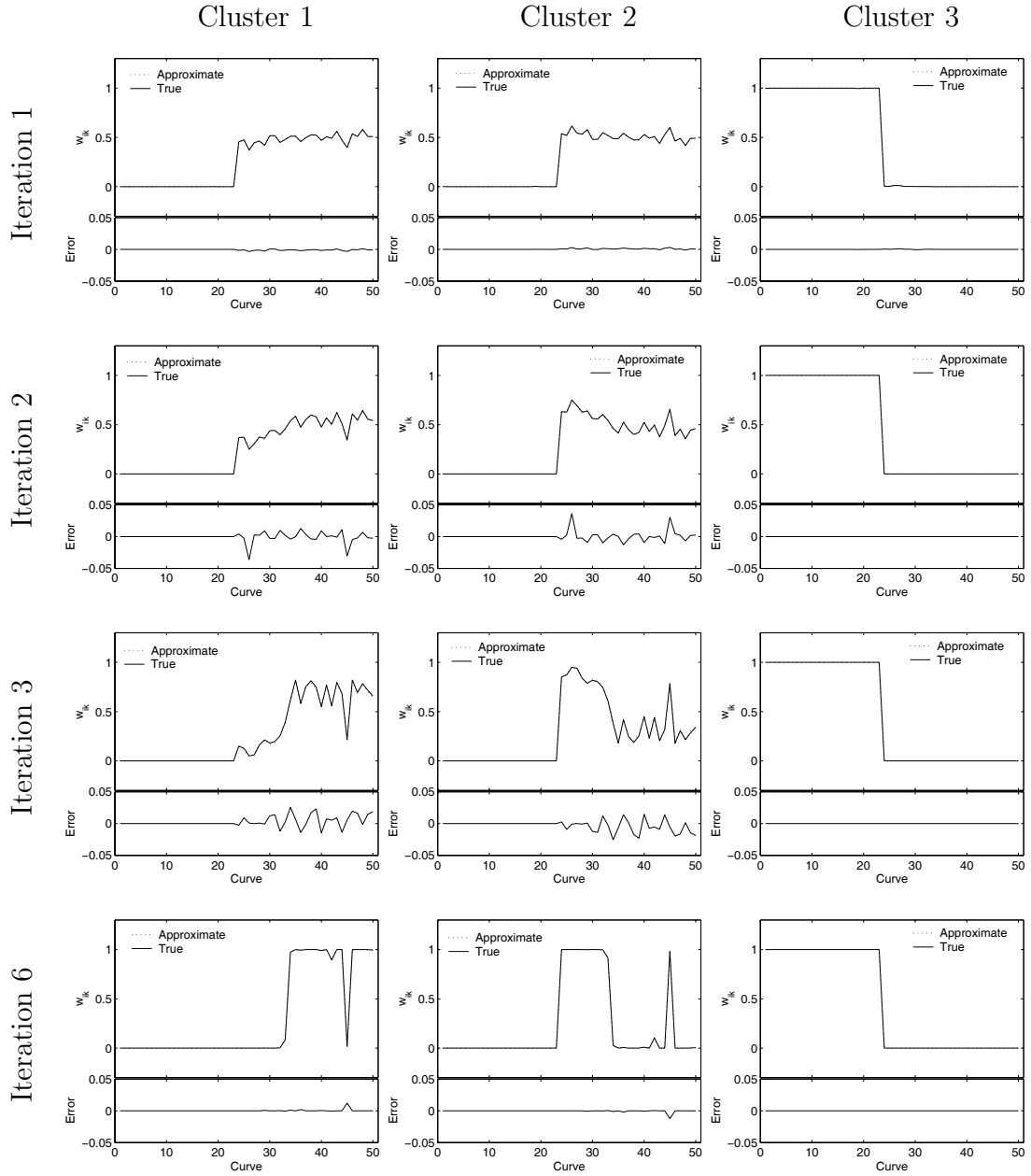


Figure 8.3: Approximate and true membership probabilities for each cluster at iterations 1, 2, 3, and 6. Each of the 12 "boxes" contains two separate, but related plots. The upper y-axis gives the value of w_{ik}, and the lower y-axis gives the error in the approximation. Both the exact and the approximate values of w_{ik} are plotted in the upper plot; however, the approximate curve can almost never be distinguished from the exact curve since it tracks the exact value closely. There is a noticeable tendency for the error to first increase and then die back down over time.

the true clustering during this run. The approximation is visibly poor in this case. However, because the resulting EM solution is not high in likelihood, this run will be discarded anyway.

We can also look at the approximation accuracy of the log-likelihood. The membership probabilities and the log-likelihood are both functions of the marginal density $p_k(\mathbf{y}_i)$. Thus, they both stem from the same approximation. The main difference is that the memberships are put through a normalization process whereas the log-likelihood can be seen as a sum over the un-normalized memberships.

Figure 8.5 graphs the log-likelihood from the EM run shown in Figure 8.3. Again, both the approximated values and the exact values are given. In the left plot, we see that the approximation follows closely with the exact curve. A close-up of the behavior after the 5th iteration can be seen in the right plot. The variance in the approximation can be estimated by calculating the differences between the two curves in this plot. The resulting standard deviation is 4.8×10^{-3}.

If there is a need to decrease the approximation error, this can be addressed by either increasing the number of samples or by using a different sampling scheme (e.g., acceptance/rejection importance sampling, constrained sampling, etc.; Gentle, 1998). For our purposes, we found that increasing the number of samples was a useful solution. For example, we set M to about 100 during normal runs of EM. But when calculating the log-likelihood at the end of EM, or for out-of-sample test purposes, we used values of M as large as 1500.

8.5 Testing methodology

In this section we address the problem of comparing sets of alignment models to other sets of both alignment and non-alignment models. Out-of-sample test statistics are

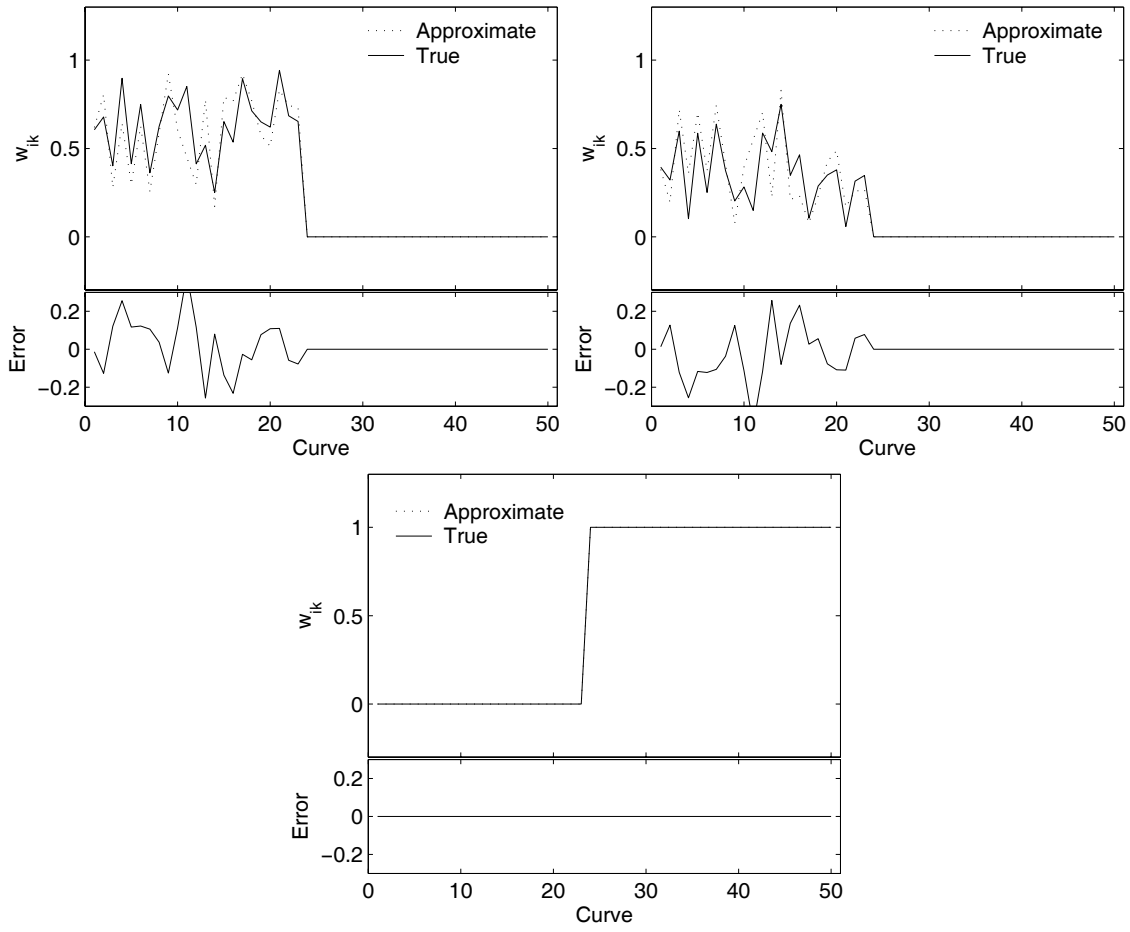


Figure 8.4: Approximate membership probabilities for each cluster during an EM run that returns an incorrect solution.

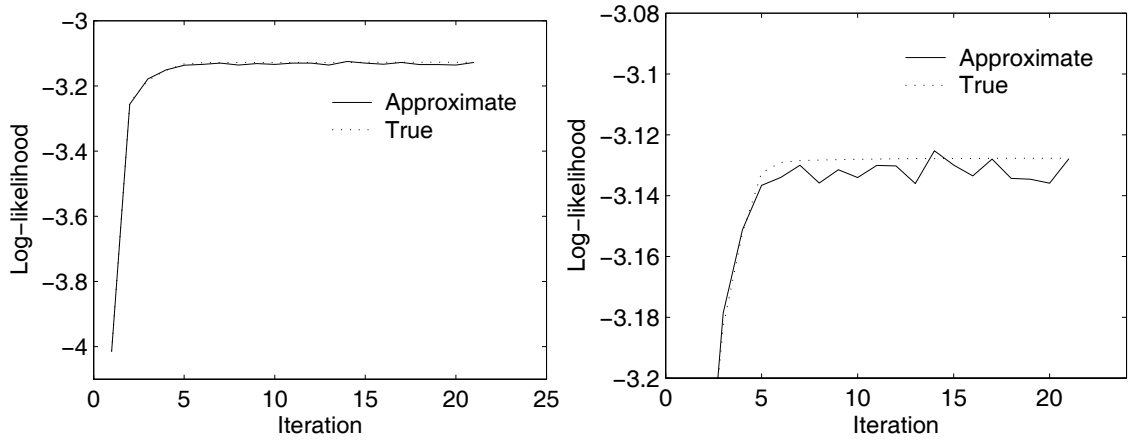


Figure 8.5: Approximate and true log-likelihood versus iteration. (left) plot over all iterations, (right) close-up of the tail-end of iterations.

essential when comparing models with different complexities since a more complex model (such as a model with 100 clusters) can always provide a better fit to the training data than a less complex model (such as a single cluster model). We use two main test statistics for comparing models throughout this thesis: test log-likelihood, and predicted squared error. In each case, to calculate the out-of-sample scores, we either generate many training and test datasets (e.g., when running simulated data experiments) or use cross-validation in a finite dataset setting.

8.5.1 Cross-validation

Cross-validation is a general model-selection technique that attempts to select the best predictive model from among a set of candidates. Prediction on *unseen* test data measures the true predictive power of a model. Cross-validation estimates each model's predictive power on a single data set by repeatedly training on one random subset and testing (or scoring) on another disjoint random subset. The resulting test scores are averaged across the random subsets, the scores are ranked, and the model with the best score is chosen.

When this procedure is carried out using v different partitions such that each point is tested only once, the method is called v-fold cross-validation (v is commonly chosen as 10). We use both v-fold cross-validation and another variant known as Monte Carlo cross-validation (MCCV; see Appendix B for details).

The one remaining problem is to choose the score or test statistic that is used during the testing phase of each run of cross-validation. In the next two sections, we describe the test log-likelihood as one such measure, and then derive the prediction SSE score for our alignment models (a useful measure in a curve analysis setting).

8.5.2 Test log-likelihood

The natural score in a mixture framework is the log-likelihood, or $\log p(Y'|\Theta)$, evaluated on an unseen dataset Y', where we explicitly list the parameter vector Θ. From Equation (8.10), we can write the equation for the test log-likelihood of the joint clustering model that allows for affine transformations in space as

$$\log p(Y'|\Theta) = \sum_i \log \sum_k \alpha_k \mathcal{N}(\mathbf{y}_i|\mathbf{B}_i\boldsymbol{\beta}_k, \mathbf{U}_k + \mathbf{V}_k - \sigma_k^2\mathbf{I}), \tag{8.32}$$

where $\Theta = \{\alpha_j, \boldsymbol{\beta}_j, \mathbf{U}_j, \mathbf{V}_j, \sigma_j^2\}_{j=1}^K$. This is a fair and principled measure because the log-likelihood is an "integrated" measure. In other words, only the observed test data Y' appears on the left of $p(Y'|\Theta)$, and only the fixed parameter set appears on the right. All other variables are integrated out of the measure.

For example, in Equation (8.32), the hidden membership probabilities have been integrated out of the mixture likelihood through the use of the sum over k. It would not be fair if a clustering model was allowed to first calculate the membership probability for a particular test curve, and then evaluate $\log p_k(\mathbf{y}_i|\theta_k)$ as the score for the test curve $\mathbf{y}_i$. The cluster model is not allowed to use the test data to choose any specific cluster. It must integrate over the clustering instead.

Suppose we want to compare an alignment model (e.g., a space-affine clustering-alignment model) and a non-alignment model (e.g., a standard PRM). In this case, we must integrate out the effect of the allowed transformations from the log-likelihood for the alignment model before comparison. That is, the alignment model is not allowed to calculate an alignment on a test curve before the score is recorded. Indeed, we see in Equation (8.32) that there are no transformation parameters (i.e., c_i, d_i) present on the right side. They have been integrated out of the model as is required.

We can view comparison by log-likelihood as a game. Each model is allowed to

position its finite density throughout the probability space in any way it desires. An outside expert then presents a number of test examples to a set of competing models. The model that has positioned its finite density in a way that best coincides with the test examples is the winner. To say that a model must use an integrated measure, is to say that the model cannot reposition its density after it sees the test examples. That would violate the rules of the game.

8.5.3 Prediction squared error

The test log-likelihood is a density modelling measure. It determines how well a learned density matches a set of new instances. It does not measure one-step-ahead prediction capability. For example, it does not measure how well a model can predict the next point in a curve given all of the previous points along the curve. Such a measure is important in curve analysis problems since curve prediction is a fundamental question.

We use a prediction SSE (sum-of-squared error) score to measure this ability. This measure can be used by probabilistic and non-probabilistic methods. We take the learned model and predict the *test* curve point $\hat{y}_{ij}$ at $\mathbf{x}_{ij}$ using the learned model parameters and the partial *test* curve $\mathbf{y}_{i(j-1)}$ (which contains all the points up to time $j-1$). Then we subtract this prediction from the true value y_{ij}, square the result, and sum this across the predictions made along the curve. Adding these values across all curves in a test set and dividing by the number of predictions gives us a mean SSE per-point test measure.

We can perform a forward prediction in which we predict the next point given all previous points, or we can perform a forward-backward prediction (called smoothing in state-space sequential modelling) in which we predict the current point given all previous as well as future points.

In order to use such a measure we need to be able to calculate the expected value of y_{ij} given the initial partial curve up to time $j-1$. As an example, we calculate this prediction for the time-translation alignment model. We begin with an in-cluster prediction:

$$\mathrm{E}[y_{ij}|\mathbf{y}_{i(j-1)}, z_i = k] = \int y_{ij}\, p_k(y_{ij}|\mathbf{y}_{i(j-1)})\; dy_{ij}. \tag{8.33}$$

We remove the dependence of y_{ij} on the partial curve $\mathbf{y}_{i(j-1)}$. The dependence is removed if we introduce the translation parameter b_i. For if we know the translation b_i and the model parameters $\{\boldsymbol{\beta}_k, \sigma_k^2\}$, then the prediction is always $[\![\mathbf{x}_i - b_i]\!]\boldsymbol{\beta}_k$ no matter what the value of $\mathbf{y}_{i(j-1)}$. We remove the dependence by introducing the time translation b_i and then integrating so that we leave the equation undisturbed:

$$\begin{aligned} \mathrm{E}[y_{ij}|\mathbf{y}_{i(j-1)}, z_i = k] &= \int y_{ij} \int p_k(y_{ij}, b_i|\mathbf{y}_{i(j-1)})\; db_i\, dy_{ij} \\ &= \int y_{ij} \int p_k(y_{ij}|b_i)\; p_k(b_i|\mathbf{y}_{i(j-1)})\, db_i\, dy_{ij}. \end{aligned} \tag{8.34}$$

We are left with a likelihood factor $p_k(y_{ij}|b_i)$, which equals $\mathcal{N}([\![\mathbf{x}_i - b_i]\!]\boldsymbol{\beta}_k, \sigma_k^2\mathbf{I})$, and a posterior factor $p_k(b_i|\mathbf{y}_{i(j-1)})$ for which there is no simpler form (this is because we are working with a time-alignment model; for the space-alignment models, the calculation is exact). We cannot compute the integration directly, but we do know that the value of the integrand will be small for values of b_i which are not close to the posterior mean. In fact, because posteriors are narrowly peaked densities in general, we can assume that almost all of the area underneath the density is centered around the posterior mean $\hat{b}_{ik}$. Therefore, we can make an approximation of the inner-most integral by replacing it with $p_k(y_{ij}|\hat{b}_{ik})$ and writing

$$\mathrm{E}[y_{ij}|\mathbf{y}_{i(j-1)}, z_i = k] \approx \int y_{ij}\, p_k(y_{ij}|\hat{b}_{ik})\; dy_{ij}. \tag{8.35}$$

We can also see this as a Monte Carlo integration approximation in which the sample size is just one, the most likely sample. The remaining integration is straightforward; it is just the expected value of $\mathcal{N}([\![\mathbf{x}_i - \hat{b}_{ik}]\!]\boldsymbol{\beta}_k, \sigma_k^2\mathbf{I})$. Therefore, we have the final result:

$$\mathrm{E}[y_{ij}|\mathbf{y}_{i(j-1)}, z_i = k] \approx [\![\mathbf{x}_i - \hat{b}_{ik}]\!]\boldsymbol{\beta}_k. \tag{8.36}$$

The expectation when cluster membership is unknown is

$$\mathrm{E}[y_{ij}|\mathbf{y}_{i(j-1)}] \approx \sum_k w'_{ik}\, [\![\mathbf{x}_i - \hat{b}_{ik}]\!]\boldsymbol{\beta}_k, \tag{8.37}$$

where w'_{ik} is just the membership probability computed using the partial curve $\mathbf{y}_{i(j-1)}$.

8.6 Simulation results

In this section we report on systematic experiments with simulated data that show the effectiveness of our new joint clustering-alignment methodology. We begin in Section 8.6.1 by demonstrating that the framework itself is viable. We report on 16 experiments that determine whether or not our joint models can identify generated datasets. The results also validate the testing methodology, i.e., they show that the test statistics are able to distinguish between models on test data.

In Section 8.6.2 we compare joint alignment-clustering to methods that simply ignore alignment during clustering. The goal is to show the importance of accounting for misaligned curves when clustering.

Finally, in Section 8.6.3, we describe experiments that show the ineffectiveness of the sequential approach to the joint problem. The experiments in this section compare methods that sequentially cluster and then align (or vice versa) against our

joint EM algorithms.

8.6.1 Identification tests

In this section we undertake a number of identification tests. The primary goal is to demonstrate the viability of our new models and to show that the testing methodology is valid. We do this by generating curve data from one alignment-clustering model and then show that this model out-performs all other models on test data. An important factor is to show that the most complex model doesn't beat the "true" model. If this were the case, then either the model, the learning algorithm, or the testing methodology is flawed.

The experiments were carried out as follows. Each of four candidate models (listed below), in turn, was chosen as the data model. This model was used to generate 25 different random training and testing sets. Then each of the four models was trained and tested on each pair of datasets. Their test SSE and log-likelihood scores were recorded and averaged over all sets.

The averaged test SSE scores are shown in Table 8.2 and the averaged test log-likelihood scores are given in Table 8.3. The models are denoted using the abbreviations listed in Table 8.1. The data models are listed down the left column of each table so that the datasets used in the tests for each row were generated by the model listed at the far left of that row. For example, in the row labelled PRM_TT of Table 8.2, we see that the lowest averaged SSE score attained on the 25 datasets generated from the PRM_TT was 542.49, which was recorded by the PRM_TT itself. The best score in each row is bolded.

As the bolding indicates, the models, the learning algorithms, and the testing statistics are able to identify the correct generative models in each of the cases. The results reveal the common theme in model selection, i.e., a model that is too

Table 8.2: Polynomial prediction SSE scores of all models trained/tested on data generated from a specific model. The models which generated the data are listed along the left column. The lowest score in each row is bolded.

	Prediction SSE Scores			
Model	PRM	PRM_TM	PRM_AM	PRM_TT
PRM	**404.42**	404.65	407.36	414.43
PRM_TM	484.03	**481.99**	482.99	502.78
PRM_AM	869.68	863.74	**681.22**	875.58
PRM_TT	707.15	706.54	718.79	**542.49**

Table 8.3: Polynomial test log-likelihood scores of all models trained/tested on data generated from a specific model. The models which generated the data are listed along the left column. The highest score in each row is bolded.

	Test Log-likelihood Scores			
Model	PRM	PRM_TM	PRM_AM	PRM_TT
PRM	**-4.5435**	-4.5437	-4.5452	-4.5461
PRM_TM	-4.6130	**-4.6115**	-4.6135	-4.6193
PRM_AM	-4.7892	-4.7915	**-4.6959**	-4.7967
PRM_TT	-4.8125	-4.7570	-4.7650	**-4.6600**

complex tends to overfit and thus show worse out-of-sample performance. Another thing that can be gleaned from the statistics is that the more complex the data generating model, the larger the disparity in the test score of the "true" model and the remaining models.

8.6.2 Comparisons with non-alignment methods

In this section we compare joint alignment-clustering to basic clustering that does not allow for any alignment. The goal is to show the importance of accounting for misaligned curves when clustering.

The tests were carried out as follows. Twenty-five different training and testing sets were generated from a three-cluster PRM_TT. Then, Gaussian mixtures (Gmix), K-means (Kmeans), PRM, and PRM_TT were evaluated on the datasets and the

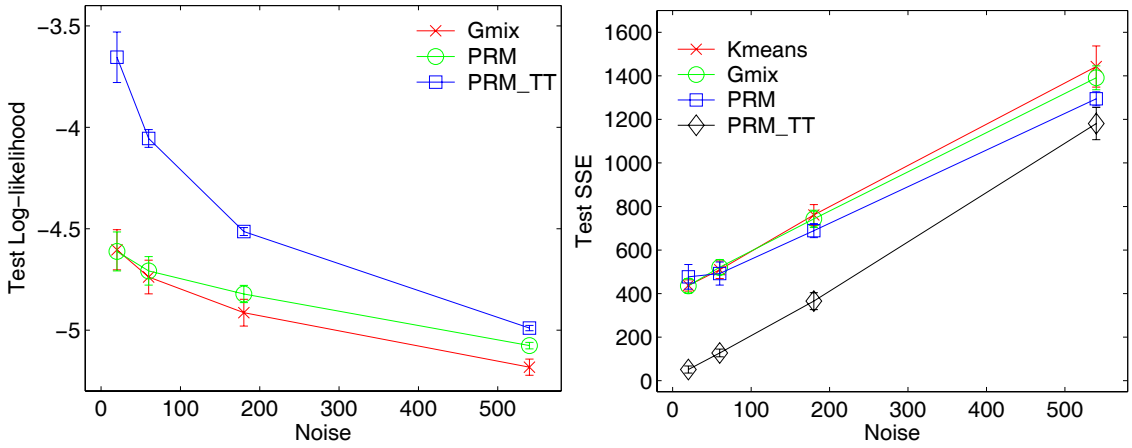


Figure 8.6: Average test scores that compare alignment and non-alignment clustering methods. The test scores are given on the y-axis and the level of measurement noise is given on the x-axis. Error bars indicate one standard deviation on each side of the plotted mean score.

test scores were recorded and averaged. This was repeated at each of four different levels of measurement noise resulting in the evaluation of 100 different test sets for each model.

Figure 8.6 graphically depicts the results from these tests. The average scores are plotted along the y-axis in both graphs, and the noise level is plotted along the x-axis. Error bars indicate one standard deviation on each side of the plotted mean.

In the left plot, the test log-likelihood scores demonstrate the importance of accounting for alignment in a curve dataset. The overall density modelling performance of the alignment method PRM_TT is well above that of the two non-alignment clustering methods at all levels of noise. In addition, the curve-based clustering model PRM out-performed the vector-based method of Gaussian mixtures at increasing levels of noise.

A similar picture can been seen from the SSE scores in the right plot of Figure 8.6. PRM_TT demonstrated significant improvement in prediction power over the non-alignment methods. The improvement is due to the fact that this model is able

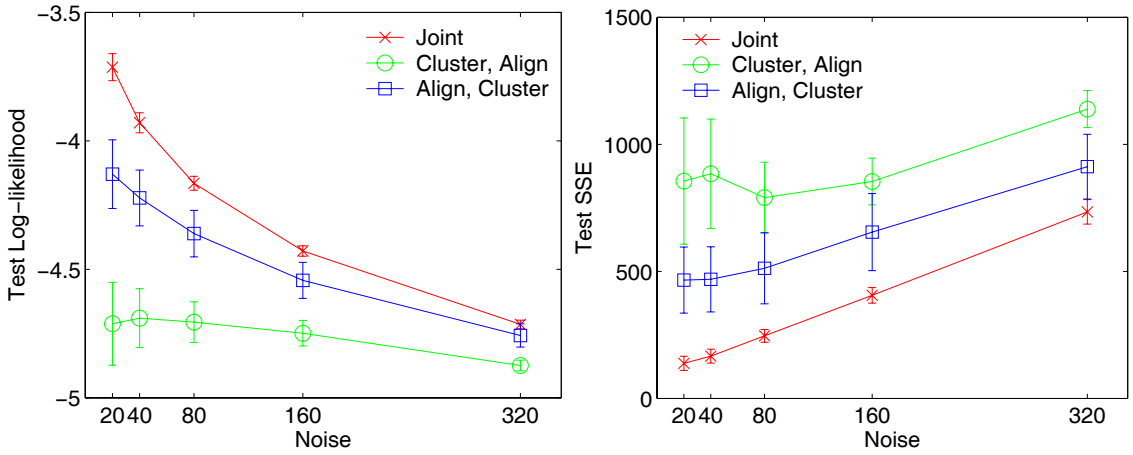


Figure 8.7: Average test scores that compare joint and sequential clustering methods. The test scores are given on the y-axis and the level of measurement noise is given on the x-axis. Error bars indicate one standard deviation on each side of the plotted mean score.

to accurately predict the alignment of a partial curve before the point-prediction is made. The non-alignment methods only use the partial curve to predict membership probabilities. This makes the joint models much more accurate at the prediction problem.

8.6.3 Comparisons on joint methodology

In this section we compare joint clustering-alignment with that of clustering first and then aligning second or vice versa. The goal is to show the benefit of the joint approach.

The experiments were carried out as follows. Twenty-five different training and testing sets were generated from a three-cluster PRM_TT. Then, PRM_TT and two sequential methods (cluster first, align second; and align first, cluster second) were evaluated on the datasets and the test scores were recorded and averaged. This was repeated at each of five different levels of measurement noise resulting in the evaluation of 125 different test sets for each model.

The results from these experiments are shown in Figure 8.7. The average scores are plotted along the y-axis in both graphs, and the noise level is plotted along the x-axis. Error bars indicate one standard deviation on each side of the plotted mean.

The left plot depicts the average test log-likelihood results. The joint approach clearly outperforms the two sequential methods on the test density modelling task. As the noise level increases we see the performance gap narrow just as we saw in the previous section. It appears that for a sequential approach, it is preferable to align first and then cluster second.

The right plot depicts the average test SSE scores. Again, we see the joint approach clearly outperforms the two sequential methods on the partial curve prediction task. It appears that the noise level does not dramatically affect the performance gap in this case. The align-first, cluster-second approach again appears superior to the alternate sequential method.

8.7 Summary

In this chapter, we introduced a new unified methodology that integrates both alignment and clustering, augmenting each individual problem with iterative input from the other in a joint estimation framework. The novelty of the approach lies in its self-contained probabilistic formulation in which priors on transformations naturally provide for clustering models that are transformation-invariant without any extra specialized constraints or procedures. This joint clustering-alignment problem has not previously been addressed in curve analysis, and thus this work is seminal.

We derived a useful predictive measure consistent with the framework that allows for fair comparison of dissimilar models based on future expected predictive capability. We used this measure along with test log-likelihood scores to show the

effectiveness of our joint methodology in a number of systematic simulation experiments. The main contributions of this chapter can be listed as follows:

- Unification of both alignment and clustering in a joint estimation framework.
- Probabilistic formulation naturally resulting in transformation-invariant clustering models without any added specialized procedures.
- Incorporation of curve models that provide for a joint clustering and alignment of sets of curves in *curve space*.
- Experimental results that demonstrate the benefit of treating the joint problem as opposed to addressing each of the clustering and alignment problems in isolation.

Chapter 9

Identification, Tracking, and Clustering of ETC Cyclones

9.1 Introduction

In this chapter, we describe the application of our clustering models to the problem of extra-tropical cyclone (ETC) clustering. We develop a methodology for the detection and tracking of cyclones from mean sea-level pressure (MSLP) data, generated by a general circulation model (GCM), and then apply our joint clustering-alignment models to the resulting set of cyclone trajectories.

GCMs are computational models that are used for climate prediction. As such, the cyclone dataset that is used in this application is technically *simulated* as opposed to *observed*. However, GCMs are quite complex dynamical models that generate global weather states consisting of hundreds of meteorological output variables "observed" at (potentially) every point on the earth. In many ways, the output from GCM models are thought of as "real" data in the atmospheric sciences. Thus, we can think of this chapter as describing an application to a "real-world" dataset. In the

next chapter, we describe an application of our clustering models to an "observed" cyclone dataset, compiled by the Joint Typhoon Warning Center (JTWC).

This chapter is organized as follows. In Section 9.2, we motivate the importance of understanding ETCs and how they fit within the global climate picture. Section 9.3 discusses the problem definition and describes the relevant prior work in this area. Section 9.4 describes the GCM model and the related *raw* MSLP dataset that were used for the analyses in this chapter.

In Section 9.5, we introduce our identification and tracking methodology that was used to produce the set of cyclone trajectories from the GCM data. Section 9.6 discusses the modelling of cyclone trajectories using regression models. This section justifies the use of relatively simple models for the modelling of highly non-linear dynamical weather phenomena (i.e., cyclones)

In Section 9.7, the model selection problem is addressed. This section makes up the bulk of the experimental work with the alignment models for cyclone clustering. Experimental results are reported that were used to make decisions about the optimal order of the cyclone regression models, the most suitable type of trajectory preprocessing, the best predictive alignment model, and the number of clusters that best describes the cyclone dataset.

Following this, Section 9.8 analyzes the cluster results in detail produced by the previously selected "best" methodology. This section provides graphical as well as quantitative analysis of the clustering results. The temporal behavior of the clusters is briefly discussed in the latter part of this section. Finally, the chapter is concluded with a summary in Section 9.9.

9.2 Motivation

Extra-tropical cyclones (ETCs) are important for a number of reasons. They are responsible for severe and highly damaging winter weather over North America and Western Europe (Schubert et al., 1998). In the last decade, they were second only to hurricanes (which are caused by tropical cyclones) in total insurance loss due to weather.

Atmospheric scientists are interested in the special role ETCs play as intermediary between large-scale components of climate and more regional local weather patterns. For example, it is not well-understood how long-term climate changes (such as global warming) may influence ETC frequency, strength, and spatial distribution; and how this, in turn, may influence the regional climate (Murray & Simmonds, 1991). A better understanding of the behavior of ETCs in the context of climate variability could have important societal implications.

This chapter is concerned with ETCs over the North Atlantic and Western Europe. It is common for North Atlantic ETCs to propagate from west to east over the ocean as they go through their life cycle. It is the tail-end of this life cycle that is most responsible for the climate variability over Western Europe (Blender et al., 1997). Analysis of North Atlantic ETCs from GCM simulations (or observed datasets) may lead to a better understanding of the cyclones themselves and their associated frontal weather systems. This may also provide clues as to how atmospheric events in distant parts of the world (such as El Nino in the tropical Pacific) are able to affect regional weather over Europe (Schubert et al., 1998). Such analyses are important since direct study of the atmosphere and its effects is a complex undertaking.

In the next section, we discuss the problems encountered in cyclone analysis such

as cyclone identification, tracking, and clustering, and describe the relevant prior work that addresses these issues.

9.3 Problem definition and prior work

Historically, cyclone data analysis relied upon scientists who performed the tedious job of manually analyzing weather charts to identify sets of observed cyclone tracks. The results of these analyses were then used to generate statistics such as cyclone genesis rate, geographic distribution, cyclone lifetime, track orientation, etc. Often the cyclones were categorized into groups or clusters based on the shapes of their tracks (e.g., straight, recurving, noisy; Hodanish & Gray, 1993).

More recently, methods that allow for the automatic identification, tracking, and clustering of cyclones have received much interest (Hodges, 1998). This is largely due to the proliferation of general circulation models (GCMs). GCMs are parameterized computational models used for climate prediction. They can be used to generate a potentially unlimited amount of simulated meteorological data. It would be infeasible to manually chart each cyclone resulting from hundreds of years of simulated output from a single GCM.

The identification and tracking of cyclones from GCM data is important for a number of reasons. First, validation of GCMs is of great interest since future predictions of climate depend upon the accuracy of such models. A common validation step is to compare particular spatial averages over time to observed averages from "real" meteorological data (Gates, 1992). However, it can be valuable to compare smaller-scale transient behavior in the same manner also (Hodges, 1994). In this case, the existence of cyclones and their derived statistics from GCM output can be used as comparison measures with those from real data.

GCMs can also be used to run large-scale experiments that provide clues as to how global climate change affects local weather patterns such as seasonal rain (which is dominated by the existence of cyclones and other weather phenomena). For example, GCMs can be used to simulate years of normal climate effects, and then these same years can be simulated again with an increased greenhouse gas concentration (Schubert et al., 1998). After detecting and analyzing the resulting cyclones in each case, comparisons can be made to determine if cyclones are significantly affected by increased greenhouse gasses.

There has been much progress in automatic cyclone detection. Identification methods range from the relatively simple approach of finding minima in surface pressure fields (König et al., 1993) to more complex approaches such as the use of image processing and computer vision techniques (Hodges, 1994, 1998). These algorithms are usually coupled with a tracking scheme to produce a final set of trajectories. Proposed methods for tracking include a number of different schemes; for example, nearest-neighbor search (Blender et al., 1997; König et al., 1993), numerical prediction schemes with cost minimizing optimizations (Murray & Simmonds, 1991), and image-based feature tracking methods (Hodges, 1994, 1995).

In contrast to cyclone detection, there has not been much work in automatic categorization or clustering of cyclone trajectories. Blender et al. (1997) introduced the idea of using K-means to cluster cyclone trajectories of fixed length. To apply K-means to cyclone trajectory data, one must convert the variable-length trajectories into fixed-dimensional vectors. To do this, Blender et al. constrain each of their storm trajectories to be exactly 3 days in length (12 measurements, 6 hours apart) and then concatenate each of the latitude and longitude measurements to form 24-dimensional vectors to be used with K-means.

This type of vector-based clustering has limitations when applied directly to cy-

clones. For example, embedding of the sequence of time and space measurements into a vector-space loses spatio-temporal smoothness information related to the underlying dynamics of the cyclone process. Furthermore, trajectories of different lengths do not admit a fixed-dimensional representation unless truncated in some manner, which results in a potential loss of useful information. For example, Simmons and Hoskins (1978) identified a cyclone life cycle of about 10 days. This is much longer than what the truncated trajectories of Blender et al. (1997) allow for; and thus, the fixed-dimensional setup does not seem natural for this problem

Our goal is to demonstrate the application of our curve clustering models to cyclone clustering in a GCM setting. This methodology eliminates the problems associated with K-means type approaches and provides for a joint clustering-alignment, something that has not previously been attempted in atmospheric science.

This application requires the development of an identification and tracking component prior to the clustering. In the following sections, we describe the GCM dataset that was used in this application, and then go on to describe our identification and tracking procedures.

9.4 GCM model and raw dataset

The GCM that was used to generate the raw MSLP dataset used in this chapter was the National Center for Atmospheric Research Community Climate Model, version 3 (CCM3; Hack et al., 1998). It was run with observed sea surface temperatures specified at the lower boundary over the 1980–1995 period. The atmospheric pressure at mean sea level (MSLP) was given on an approximate $2.8^\circ \times 2.8^\circ$ Gaussian grid over the globe. The data are available every 6 hours and the subset of data analyzed in this chapter consisted of measurements for the winter months (1 November to 30

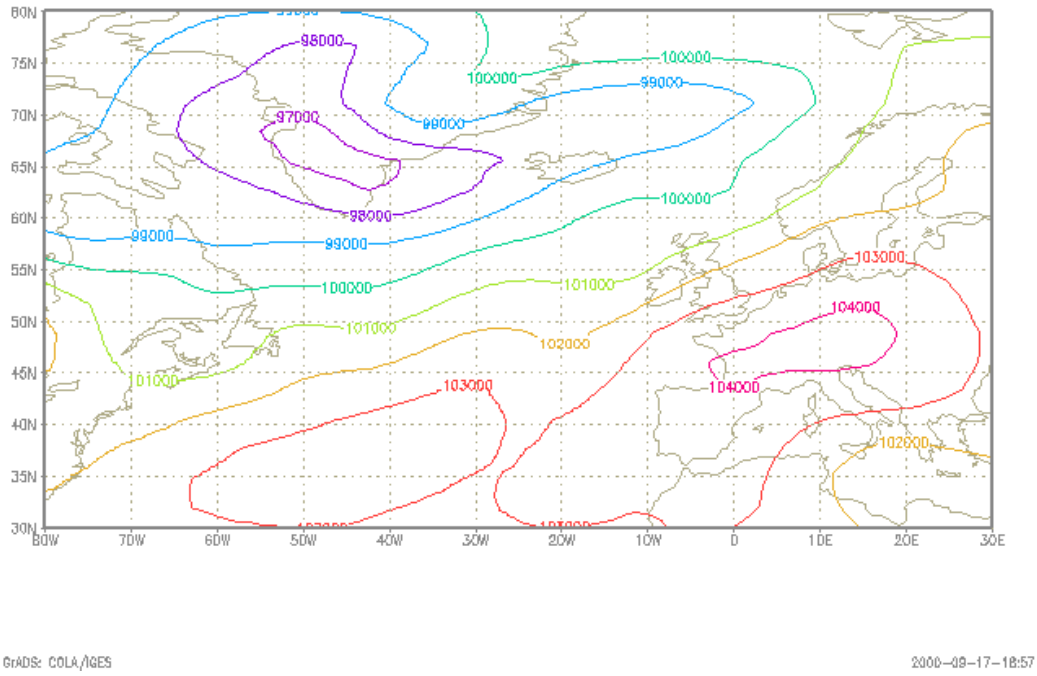


Figure 9.1: Contour plot showing MSLP in the North Atlantic at a particular date.

April) from 1980 to 1995.

Interest lies on North Atlantic ETCs in the area 30°N–70°N and 80°W–10°E. In each winter there are 181 days. Therefore there are $181 \times 4 = 724$ grids of data for each winter, where each grid is a 19×41 array of MSLP pixel values—we refer to a single grid of such pixels at a specific time as a *frame*. A snapshot of the resulting data can be seen in Figure 9.1.

9.5 Identification and tracking of cyclones

This section introduces our identification and tracking scheme—this is based on standard methods proposed in the literature (Blender et al., 1997; König et al., 1993) and requires relatively few parameters to implement. We use the term *trajectory* to

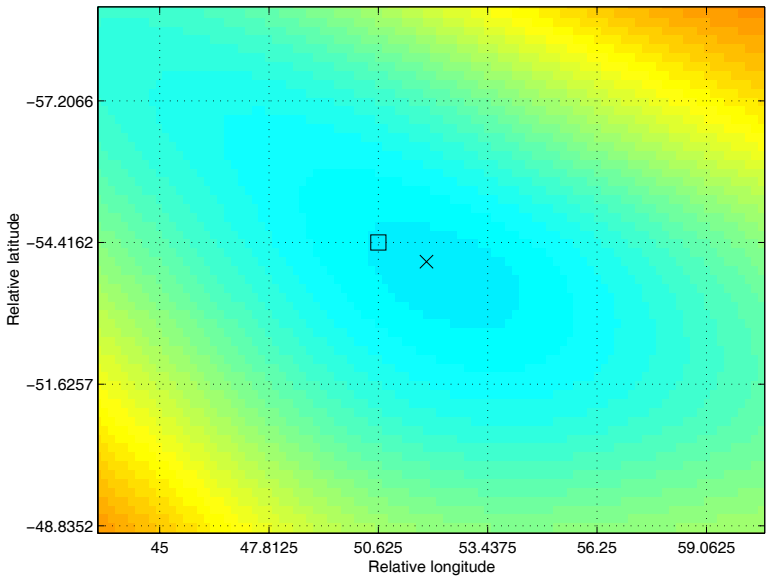


Figure 9.2: An example off-grid minimum found using gradient descent. We see an image of the interpolated MSLP data at one instant in time. The grid lines represent the location of the actual data grid in our data set. The square shows the grid-located minimum found using sliding neighborhoods. The 'x' shows the approximate minimum found using gradient descent with bicubic interpolation.

refer to a set of lat-lon (latitude-longitude) tuples corresponding to the path of a cyclone, defined from a start time t_1 contiguously through to an end time t_2.

9.5.1 Cyclone identification

Cyclones are characterized by well-defined pressure minima. In order to distinguish these minima more easily from larger-scale, low-pressure areas, the gridded MSLP data were prefiltered in space at each time so as to remove the largest planetary-wave scales (Hoskins & Hodges, 2002; Anderson et al., 2003). This was accomplished by transforming the data to spherical harmonics, removing the gravest four wavenumbers, and transforming back to grid-point space. These spectral transforms are exact to within roundoff error for a Gaussian grid.

A bicubic interpolation method coupled to an iterative scheme is used to find

pressure minima using gradient descent. First the frames are scanned over time and all local minima are found using a simple sliding neighborhood method. A "pixel" is declared to be at a minimum if its value is less than all eight of its neighbors on the grid. Then, gradient descent with bicubic interpolation is used to descend to the point "inside" of the pixel that is at an approximate interpolated minimum. This point provides an approximate off-grid center of a candidate cyclone. Figure 9.2 shows an example of this procedure.

Spurious minima can arise using this procedure, usually in one of two situations: (1) in high-pressure regions not associated with cyclonic activity, and (2) within the outskirts of a single cyclonic system with an already located central minimum. Both situations can be dealt with by thresholding the MSLP data at a particular pressure level to form individual low-pressure regions within the data. This results in "pixel blobs" or contiguous pixel regions, where each blob corresponds to the estimated spatial extent of a single candidate cyclone at a specific time.

Spurious minima are removed by (1) rejecting minima that are spatially located outside the low-pressure blobs and (2) rejecting all but only the deepest minimum within each individual blob. A conservative threshold value of -17 mb is used for the results in this paper, chosen based on some preliminary experiments. (The sensitivity of this value to the resulting set of tracked cyclones is discussed below.) Note that although removal of spurious minima is desirable, it is not critical to the results since it is rather unlikely for these minima to persist over time and to be tracked as distinct cyclones. However, removal of the most obvious offenders can lead to more accurate tracking.

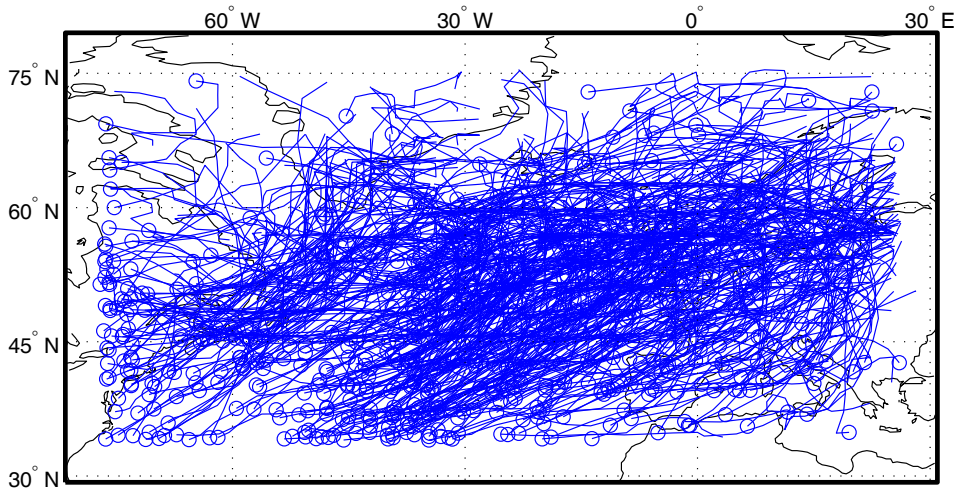


Figure 9.3: The full set of cyclone trajectories that were tracked using our methodology.

9.5.2 Tracking of cyclones

Given candidate cyclone centers determined using the procedure above, the following two steps are performed to complete the tracking. The frames are scanned sequentially from beginning to end and each candidate cyclone center is examined to determine if it can be associated with a candidate cyclone center from the previous frame. If there exists a center in the previous frame located within a small neighborhood region (window) surrounding a center in the current frame, then they are linked. If there does not exist an associated center within the window then the candidate center is designated as newly *born*.

A window size that allows up to 7 degrees of longitude and 5 degrees of latitude movement over each 6-hour measurement interval is used (this corresponds to an overall window size of 14×10 degrees, longitude-by-latitude). This allows a maximum cyclone displacement velocity of approximately 129 km/h in longitude and 92 km/h in latitude. Empirical results indicate that a typical tracked cyclone from the MSLP data used in this chapter has an approximate average velocity of 50 km/h, and thus

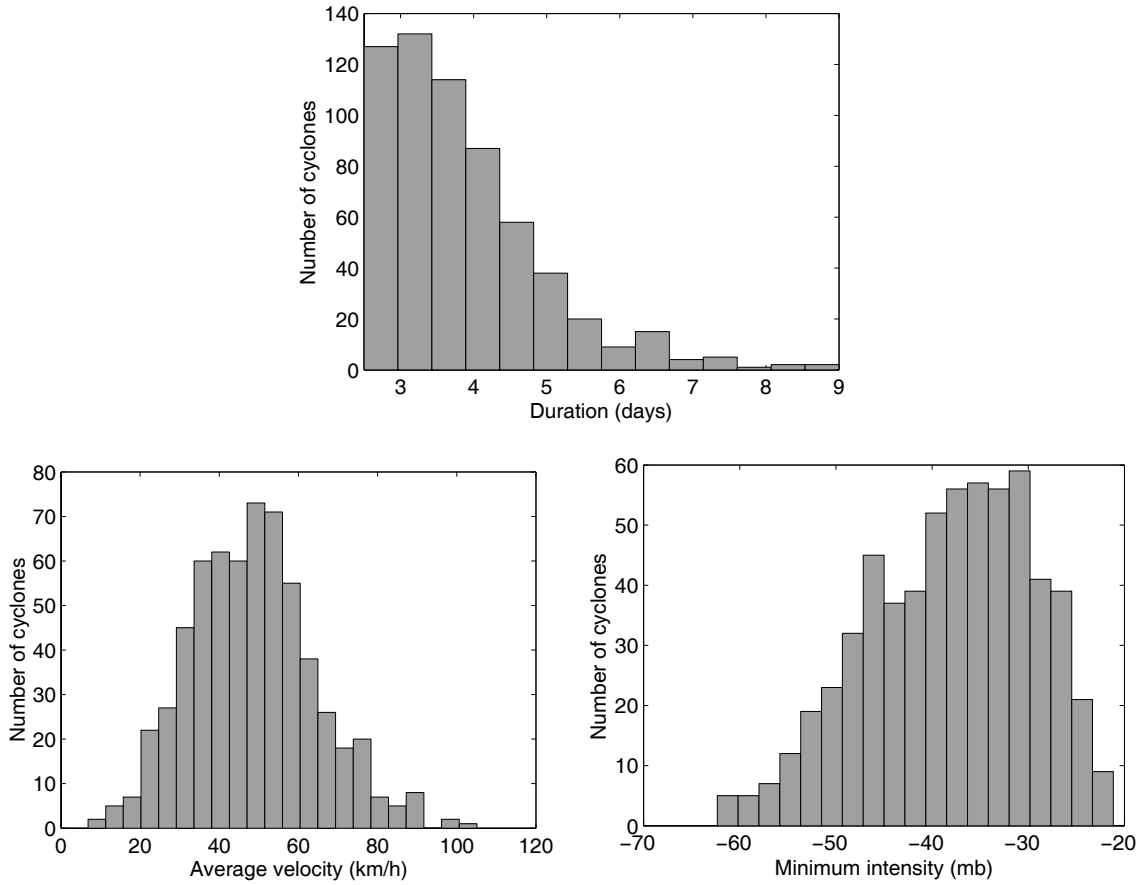


Figure 9.4: Summary histograms for the cyclone data set: (a) top, cyclone duration; (b) bottom-left, average velocity; and (c) bottom-right, minimum intensity (MSLP).

the maximum displacement velocities are rarely ever reached.

In the second step, the set of associated centers (trajectories) over time is taken and all those that exist for less than 2.5 days are eliminated. This step removes many small noisy tracks that correspond to local small-scale weather disturbances not usually considered to be cyclones.

Application of the identification and tracking procedures described above to the MSLP data produced 614 cyclones of different durations, each with a minimum of 10 observations (i.e., at least 2.5 days long). Figure 9.3 shows the complete set of tracked cyclones. Figure 9.4 contains three summary histograms describing the statistical

Table 9.1: Sensitivity of the identification and tracking procedure to small changes in window size and threshold. N is the number of cyclones tracked, μ_L is the average trajectory length, and μ_I is the average of cyclone minimum intensities.

Window size	Threshold	N	μ_L	μ_I
14×10	-17	614	15.26	-38.1
14×10	-18	614	15.26	-38.1
12×8	-17	616	14.98	-37.5
12×8	-18	590	14.94	-38.0

characteristics of the same set of trajectories. This specific set of trajectories is used for all further analysis in this chapter.

There are two parameters that primarily affect the set of trajectories detected by the identification and tracking algorithm and the sensitivity of the results to these parameters is a potential concern. The first, the threshold value used to remove spurious minima, can be lowered or raised, decreasing or increasing the number of candidate cyclone centers. The second, the window size used in the nearest-neighbor search, can also be reduced or enlarged.

We have found empirically that small changes in either of these parameters do not lead to large changes in the resulting sets of trajectories. For example, when the window size is decreased from 14×10 to 12×8, a slightly smaller set of cyclones is produced, in which only the fastest (or longest) cyclones are either shortened or removed. Table 9.1 lists the effect of selected small parameter changes on three different summary statistics of the tracked cyclones: number of cyclones, average length, and the average minimum intensity.

9.6 Regression models for cyclone trajectories

The goal of cyclone clustering—and clustering in general—is to capture individual clusters that exhibit unique characteristics. The most obvious characteristic of cy-

clones is their shape, or their movement over time through latitude-longitude space (or lat-lon space). This shape is determined by the ratio of velocities in lat-vs-time and lon-vs-time space. The slope of the line representing the direction of travel at any point in time is given by the relative latitude change in position divided by the relative longitude change in position. Thus, cyclone "shape clustering" can be carried out by modelling the component velocity profiles over time.

A cyclone trajectory is modelled with two separate polynomial regression models: one for lat-vs-time and one for lon-vs-time. Since the shape of the velocity profiles closely resemble those of polynomial functions (see, e.g., Figure 9.5), we do not pursue the use of spline regression models for cyclone trajectories.

Suppose we have a set of n two-dimensional latitude-longitude cyclone trajectories measured over time. Each trajectory $\mathbf{y}_i$ is an $n_i \times 2$ matrix containing the sequence of n_i latitude-longitude measurements (note that n_i may be different for each trajectory $\mathbf{y}_i$). The associated $n_i \times 1$ vector of times at which the $\mathbf{y}_i$ measurements were observed is denoted as $\mathbf{x}_i$.

As a simple illustration, consider a hypothetical trajectory with $n_i = 4$ measurements:

$$\mathbf{y}_i = \begin{bmatrix} 0.5 & 0.1 \\ 1 & 0.2 \\ 2.5 & 0.4 \\ 3.3 & 0.7 \end{bmatrix}$$

where the longitude measurements are in the first column and the latitude in the

second column, and

$$\mathbf{x}_i = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix}$$

gives the measurement times. Note that this represents a trajectory moving in a horizontally-dominated direction.

The longitude profile is modelled with an order p polynomial regression model in which time ($\mathbf{x}_i$) is the dependent variable; in a similar manner a regression model is given for the latitude profile. Both regression equations can be defined succinctly in terms of the matrix $\mathbf{y}_i$. The exact form of the regression equation for $\mathbf{y}_i$ is

$$\mathbf{y}_i = \mathbf{X}_i\boldsymbol{\beta} + \epsilon_i, \qquad \epsilon_i \sim \mathcal{N}(\mathbf{0}, \Sigma), \tag{9.1}$$

where $\mathbf{X}_i$ is the standard $n_i \times (p+1)$ Vandermonde regression matrix associated with vector $\mathbf{x}_i$ (see Section 5.3.1), $\boldsymbol{\beta}$ is a $(p+1) \times 2$ matrix of regression coefficients ($\boldsymbol{\beta}$ contains the longitude coefficients in the first column and the latitude coefficients in the second column) and ϵ_i is an $n_i \times 2$ zero-mean matrix multivariate normal error term with a 2×2 covariance matrix Σ (see Appendix C for the definition of the matrix multivariate normal density).

The covariance matrix Σ contains three covariances: (1) the noise variance σ_1^2 for each longitude measurement, (2) the noise variance σ_2^2 for each latitude measurement, and (3) the covariance between any two longitude and latitude measurements. We make the simplifying assumption that $\Sigma = \text{diag}(\sigma_1^2,\ \sigma_2^2)$ so that the latitude and longitude dimensions are treated as conditionally independent given the model—this could be extended to allow for covariate lat-lon dependence if desired.

The choice of error model (Gaussian) leads to a conditional density model of the form

$$p(\mathbf{y}_i|\mathbf{x}_i, \theta) = f(\mathbf{y}_i|\mathbf{X}_i\boldsymbol{\beta}, \Sigma) \tag{9.2}$$

for the lat-lon trajectory data from the i-th cyclone. The density f is matrix multivariate normal with matrix mean $\mathbf{X}_i\boldsymbol{\beta}$ and covariance matrix Σ (note that $\theta = \{\boldsymbol{\beta}, \Sigma\}$).

Given this definition, cyclones can be clustered using any of our clustering-alignment models defined in Chapter 8 by substituting (9.2) into the appropriate mixture density setting. All that remains is to choose which model best describes the particular cyclone dataset to be clustered.

9.7 Model selection

In this section, the model selection problem is addressed. This section makes up the bulk of the experimental work with the alignment models for cyclone clustering. Experimental results are reported that were used to make decisions about the optimal order of the cyclone regression models, the most suitable type of trajectory preprocessing, the best predictive alignment model, and the number of clusters that best describes the cyclone dataset.

In Section 9.7.1, brief experimental results are reported that choose quadratic regression models as optimal for the cyclone dataset. In Section 9.7.2 we investigate the many preprocessing techniques that are common in cyclone analysis. This section demonstrates how each of these techniques affects the automatic alignment that is achieved with our alignment models.

Section 9.7.3 investigates the suitability of each joint clustering-alignment model for cyclone clustering with the GCM cyclone dataset. It is shown that the space-

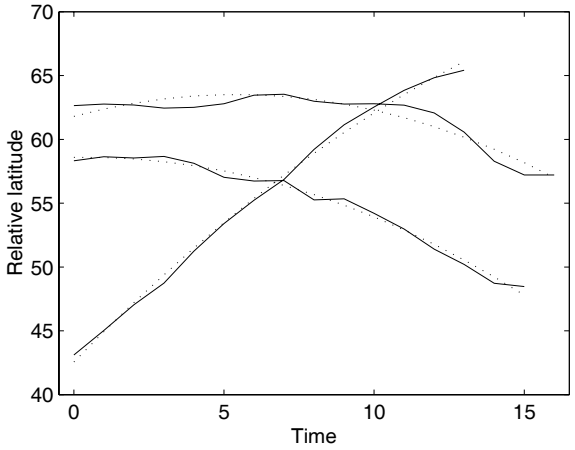


Figure 9.5: Second-order polynomial regression models (dotted) fit to three cyclone trajectories (solid) in latitude-time space.

affine alignment model gives the best performance improvement while keeping model complexity to a minimum. Finally, Section 9.7.4 details experimental results that were used to identify nine clusters that best model the cyclone dataset. We refer the reader to Table 8.1 which lists the abbreviations for the clustering-alignment models that are used extensively throughout this section.

9.7.1 Choosing the order of regression model

An important point is that polynomial regression models are too simple to capture the full spatio-temporal dynamics of ETCs. However, we believe that they provide a useful first-order approximation of the ETC tracks.

In this chapter, the choice to use second-order polynomials for the cyclone regression models (as opposed to other order polynomials) is made for two reasons: (a) visual inspection leads to this sufficient choice and (b) the objective data-driven method of cross-validation (Smyth, 2000; Smyth et al., 1999) also confirms second-order as the optimal order in this case.

Figure 9.5 shows an example of three such second-order polynomial regression

Table 9.2: Test log-likelihood scores using MCCV on the cyclone data for K-values 1 to 4 with PRM and fitted polynomials of linear to cubic. A quadratic fit achieves the highest score for all values of K from 1 to 4.

K	Linear	Quadratic	Cubic
1	-3.6024	-3.5902	-3.5929
2	-3.4380	-3.4169	-3.4198
3	-3.3279	-3.3016	-3.3051
4	-3.2355	-3.2120	-3.2170

models (dotted) fit to three cyclone trajectories (solid) in latitude-time space. The regression models provide a good fit to the cyclone trajectories as can be seen in the figure.

Table 9.2 lists the test log-likelihood scores obtained with PRM on the ETC data using cross-validation. The experiments were carried out as follows. A random sample of 50 cyclones was selected from the complete dataset. PRM was trained on this dataset using polynomials of linear to cubic, and over the K values from 1 to 4. These trained models were evaluated on a random hold-out set of 50 cyclones and test log-likelihood scores were recorded. This procedure was repeated 25 times and the scores were averaged across the runs.

The table shows that the highest score is achieved with second-order (quadratic) polynomials across all values of K. Similar results can be seen for each of the other clustering models. However, due to the complexity of the remaining model selection problem, we do not continue to train all models and choices on each and every order of regression model. Instead, we make the choice of using quadratic polynomials for the remainder of this chapter.

9.7.2 Preprocessing techniques

In this section, the optimal trajectory preprocessing technique is chosen. We begin with a discussion of the types of techniques which are commonly used for cyclone analysis. This is followed by the investigation of the effects of each of these techniques on the clustering-alignment models. Finally, we present the model selection results for choosing the best trajectory preprocessing methodology.

In order to enhance the shape aspect of the clustering, it is common to "zero" all cyclone trajectories before any clustering is carried out. For example, Blender et al. (1997) subtract the initial position from the trajectory of each cyclone before cluster analysis. This process attempts to remove initial starting position—that is, geographic location—as a factor in the clustering.

It is instructive to investigate how this affects the results from the alignment models. For example, suppose PRM_TM (PRM with translations in measurement space) is run with *zeroing* and then without *zeroing*. Since the model is freely allowed to translate the trajectories in measurement space, it is not clear if the initial translation (i.e., the zeroing) will affect the model output.

If we think of the zeroing as an initial starting position for PRM_TM, it is plausible, it seems, that this zeroing might result in a bad initial position. Thus, leading to a less desirable solution than what would have originally been output. However, given that our primary goal is to organize cyclones based on shape, it seems probable that the initial zeroing will give a better starting position than no zeroing at all. In this way, the initial zeroing can be seen as modifying the prior models for the translation parameters. For example, in the latitude dimension, if the translation d_i has initial prior $\mathcal{N}(0, v^2)$, then after the initial zeroing, d_i will now have the prior $\mathcal{N}(y_{i0}, v^2)$, where y_{i0} is the initial latitude position of the i-th cyclone.

This kind of preprocessing is another form of the curve normalization discussed

in the introductory section of Chapter 5. Many other types of normalization can be used. For example, instead of subtracting initial position, the mean of each curve could be subtracted; after which, the standard deviation of each curve could then be divided through.

We investigate the effects of five types of normalization on the clustering process. Each of these methods is given an emphasized name to simplify further discussion:

- `zero`: subtract initial cyclone position
- `mean`: subtract the curve mean
- `znorm`: first `zero` and then divide by the standard deviation
- `norm`: first subtract the mean and then divide by the standard deviation
- `nozero`: leave the trajectory unprocessed

Cyclone shape clustering

Figure 9.6 shows the results of running standard PRM on a random subset of 400 non-zeroed cyclone trajectories with K (the number of clusters) set to 3. The trajectories are plotted as tracks on a map of the North Atlantic at corresponding lat-lon positions. The circles indicate initial starting positions for the cyclone tracks. Each of the three maps represents one of the three found clusters.

It is clear that these clusters are dependent on geographic location. There are clusters for each of the western, central, and eastern North Atlantic regions. Although it may be useful to pursue such a clustering, clusters which are independent of geographic location are often sought. Clusters which result from a zeroing of the cyclone trajectories do not exhibit this geographic dependence.

Figure 9.7 shows the results of running standard PRM on this same random subset in which each trajectory has been zeroed. The geographic dependence is

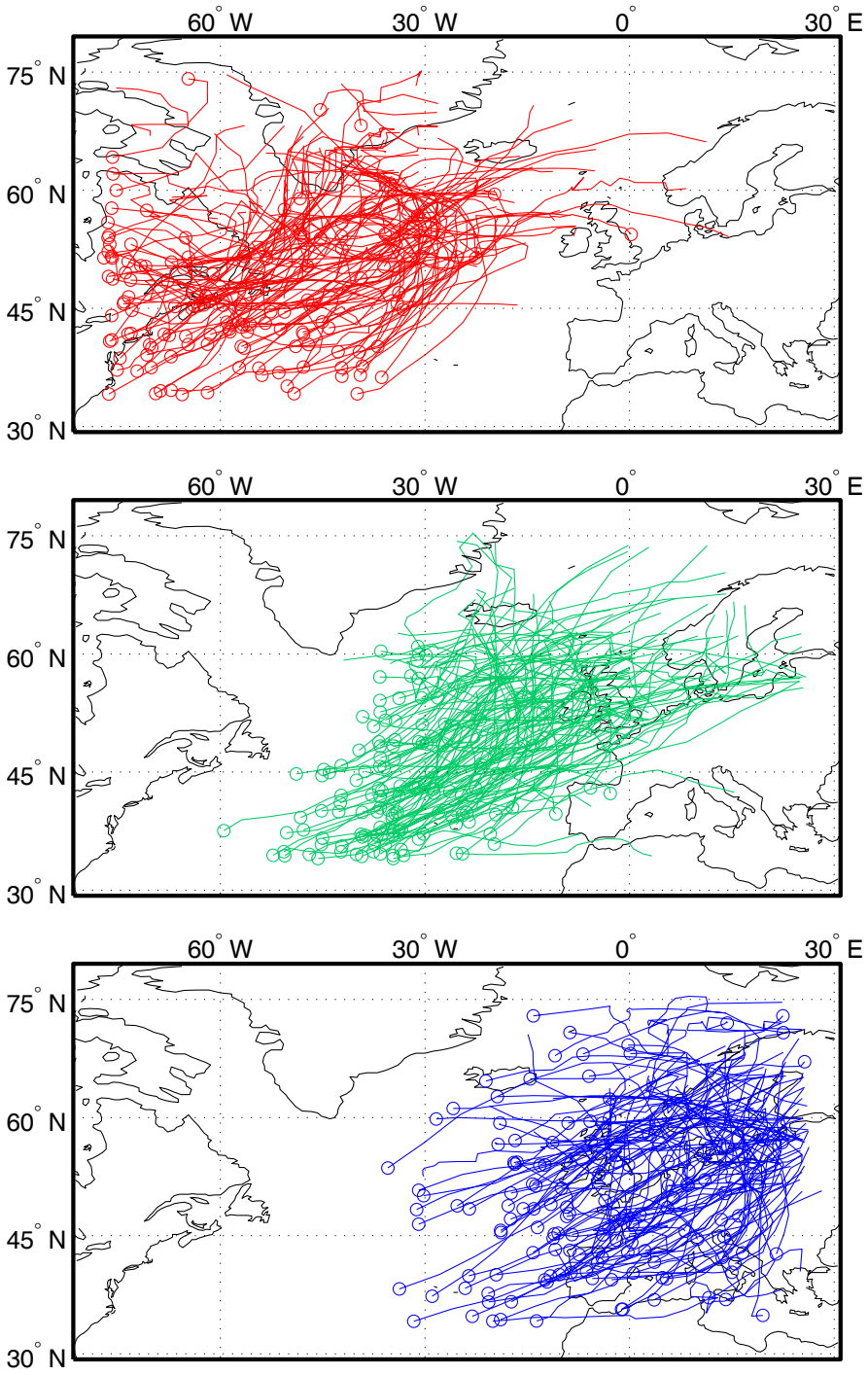


Figure 9.6: Clustering results from running PRM on the cyclone data with no pre-processing. Note the dependence of the clusters on geographic location.

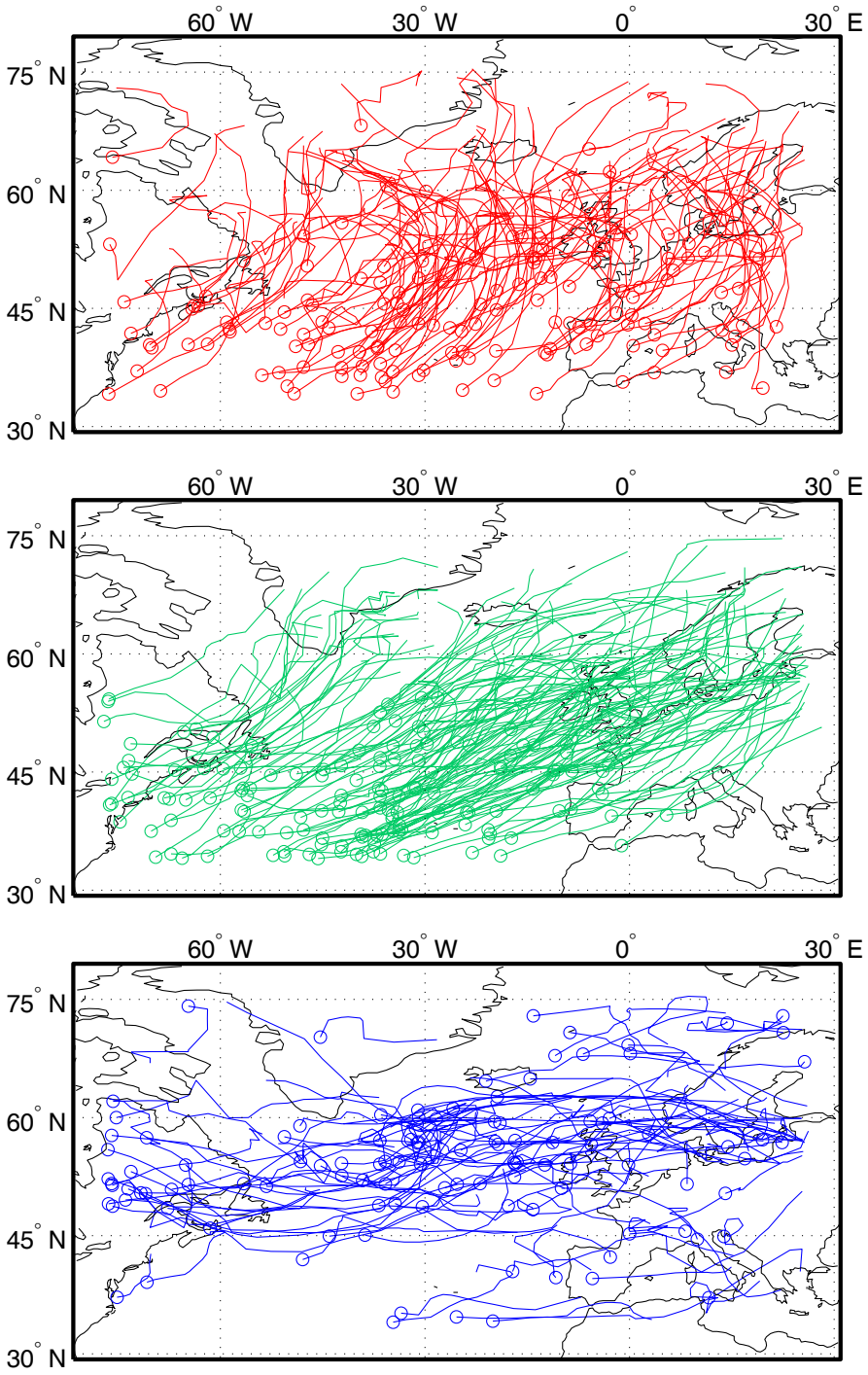


Figure 9.7: Clustering results from running PRM on the zeroed cyclone data. Note the dependence of the clusters on geographic location has largely been removed.

now removed and we are left with clusters based on shape (or based on speed and direction). These clusters loosely represent the three dominant North Atlantic cyclone track-types that have been previously reported (Blender et al., 1997; Gaffney & Smyth, 1999). These three clusters can be described as consisting of (a) cyclones moving primarily northward, often ending with a turn to the west, (b) cyclones moving northeast across the Atlantic, and (c) cyclones that move predominantly due east, often into Western Europe.

Automatic shape clustering with PRM_TM

Suppose the clustering is carried out using the space-translation model PRM_TM. This model automatically learns translations in measurement space as it jointly clusters the data. What kind of clusters will this model produce if it clusters non-zeroed data? Will the clusters be dependent on geographic location?

Figure 9.8 shows the results of running PRM_TM on non-zeroed cyclone data. The resulting clusters are similar to those from PRM on zeroed data. However, there are discernable differences. For example, the vertical (red) cluster from PRM_TM consists mainly of unpredictable, northward meandering cyclones, whereas the corresponding vertical (red) cluster from PRM seems to contain other cyclones that tend to move in a manner similar to cyclones in the other two clusters. In any case, the results indicate that PRM_TM is inherently invariant to initial geographic starting position and thus produces clusters based on shape.

The agreement between the two clustering methods can be calculated based on the number of cyclones grouped in common clusters between the two methods. The agreement between the two clusterings in this case is only 63%. In other words, approximately 147 of the 400 cyclones are assigned to different groups between the two methods.

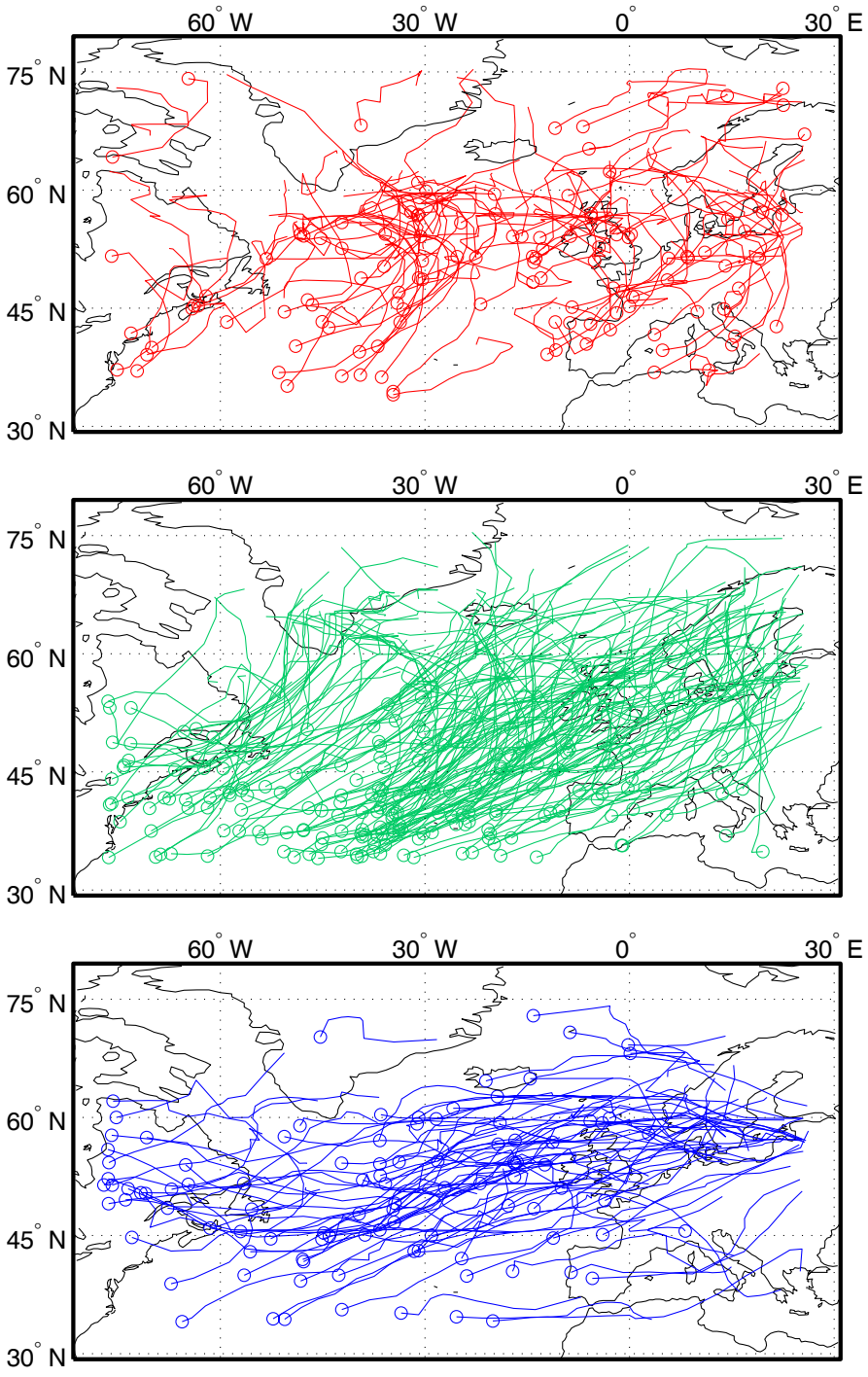


Figure 9.8: Clustering results from running PRM_TM on the non-zeroed cyclone data. Notice that PRM_TM is naturally invariant to initial geographic starting position.

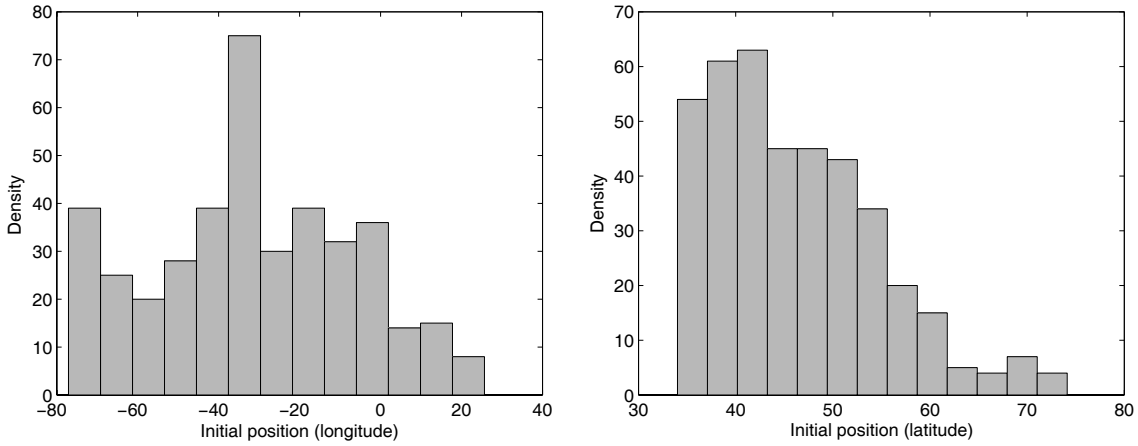


Figure 9.9: Histograms of the initial starting positions subtracted when zeroing the trajectory data. (left) Longitude positions, and (right) latitude positions.

Effect of prior zeroing on PRM_TM

We demonstrate that even though the clusters resulting from PRM_TM are invariant to initial geographic position, the clusters are not invariant to the individual zeroing of each trajectory prior to the clustering. Note that this zeroing is not the same as a global shift applied to all of the trajectories which does not change anything except the overall measurement level.

The difference between running standard PRM on zeroed data (thus, using an initial static zeroing), and running PRM_TM on non-zeroed data (thereby, allowing for an "automatic" zeroing) can best be seen by looking at the histograms of the translations in each case. Figure 9.9 contains two histograms that show the distributions of the initial starting positions that are *statically* subtracted off during the zeroing process. These histograms give the implicit translations when using PRM on zeroed data. The left histogram in the figure shows the distribution in the longitude dimension and the right histogram depicts the distribution in the latitude dimension. Note these distributions are not Gaussian in nature.

The corresponding histograms for PRM_TM on non-zeroed data are given in

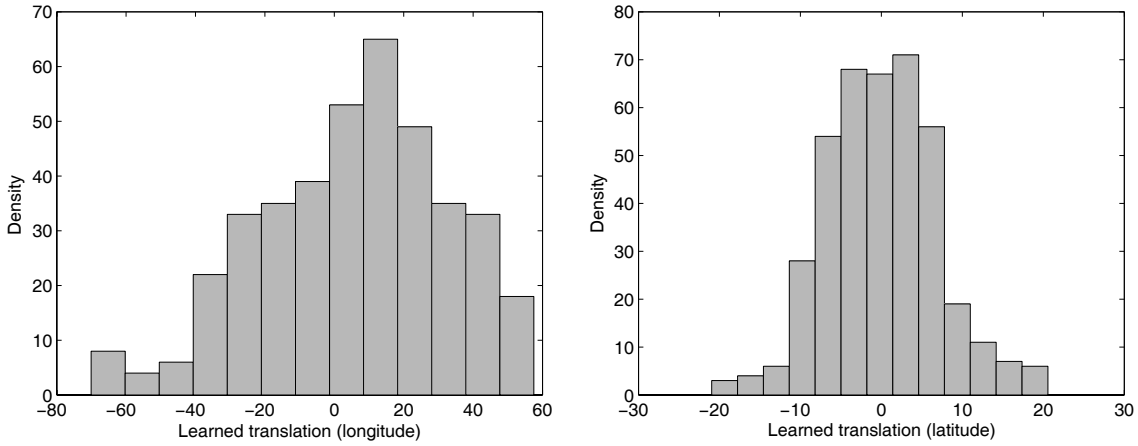


Figure 9.10: Histograms of learned translations with PRM_TM on non-zeroed data. (left) Longitude translations, and (right) latitude translations.

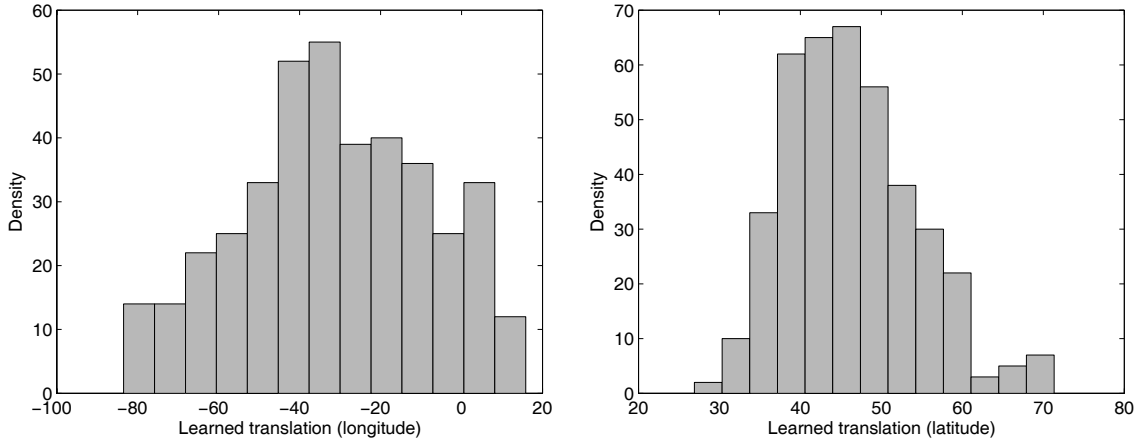


Figure 9.11: Histograms of learned translations plus the prior zeroed offsets for the output of PRM_TM on zeroed data. (left) Longitude translations and offsets, and (right) latitude translations and offsets.

Figure 9.10. These figures show the distribution of the learned translations during the clustering. The distributions appear Gaussian as the prior model on translations requires. The distributions are not only shaped differently than those for the zeroed data, but are also located along the axis at different positions (they are located around zero since the translation prior has zero mean). In other words, the alignments in each case are not similar.

Table 9.3: Example test log-likelihood scores for PRM and PRM_TM used on both zeroed and non-zeroed cyclone trajectories. The scores were evaluated on a hold-out set of 214 cyclones not included in a random training set of 400 cyclones. The bolded score is the best attained.

	PRM	PRM_TM
`nozero`	-3.5698	-2.9365
`zero`	-3.3501	**-2.8736**

A natural question is what happens if we combine the two methodologies. That is, what results from running PRM_TM on zeroed data. Intuitively, what results is a Gaussian distribution is overlayed over the location of the initial-starting-position histograms of Figure 9.9. This is shown in Figure 9.11, which gives the distributions of the learned translations when PRM_TM is run on zeroed data. The x-axis in each histogram gives the learned translations plus the initial starting positions that were initially subtracted off so that the complete translation is depicted.

The shape of these distributions is similar to those from PRM_TM in the non-zeroing case, but the locations of the axes are similar to those from PRM in the zeroing case. The translations here are optimizations or refinements of the initial "shape-zeroing" translations. Mathematically, the translation priors have been shifted from $\mathcal{N}(0, v^2)$ to $\mathcal{N}(y_{i0}, v^2)$, where y_{i0} is the initial position of the i-th cyclone.

As a precursor to the complete results reported below, we show that the best zeroing methodology can be objectively chosen by evaluating each of the above methods on a hold-out dataset. For example, Table 9.3 gives the test log-likelihood scores for both of PRM and PRM_TM on zeroed and non-zeroed data. The scores were evaluated on a hold-out set of 214 cyclones not included in the set of 400 cyclones used to produce the maps in Figure 9.8. The best score is achieved when zeroed data is applied to PRM_TM.

Choosing the best methodology

As described at the beginning of this section, five forms of preprocessing were considered for each trajectory in the cyclone dataset. We applied each of these preprocessing techniques to the cyclone data and evaluated out-of-sample scores for all of the clustering models. The results in this section indicate that there is no overall best preprocessing technique shared among all of the clustering-alignment models. Different models perform better with different types of preprocessing. However, we show that only three of the preprocessing techniques (`mean`, `norm`, and `znorm`) need further consideration during the subsequent model selection in Section 9.7.3.

The experiments were carried out as follows. A random sample of 150 cyclones was chosen from the set of 614 (resulting in approximately 4,500 total training points). Each pair of preprocessing technique and clustering model was then executed on this sample. The resulting models were evaluated on a random hold-out set of 100 cyclones and the test log-likelihood and prediction SSE scores were recorded. This process was then repeated 10 times with the test scores averaged across the 10 different runs.

Figure 9.12 shows both of the test log-likelihood and prediction SSE scores obtained with the non-alignment method PRM. The figure compares the performance for each of the `zero`, `mean`, and `nozero` preprocessing methods. The results indicate that subtracting the mean leads to better performance than zeroing or performing no preprocessing at all.

The results are somewhat different when transformations are allowed in measurement space. Figure 9.13 shows the test log-likelihood scores for PRM_TM (on the left) and PRM_AM (on the right) under the same preprocessing methods of `zero`, `mean`, and `nozero`. PRM_TM allows for translations in measurement space, and so there is only a slight difference between the `zero` and `mean` methods (the `nozero`

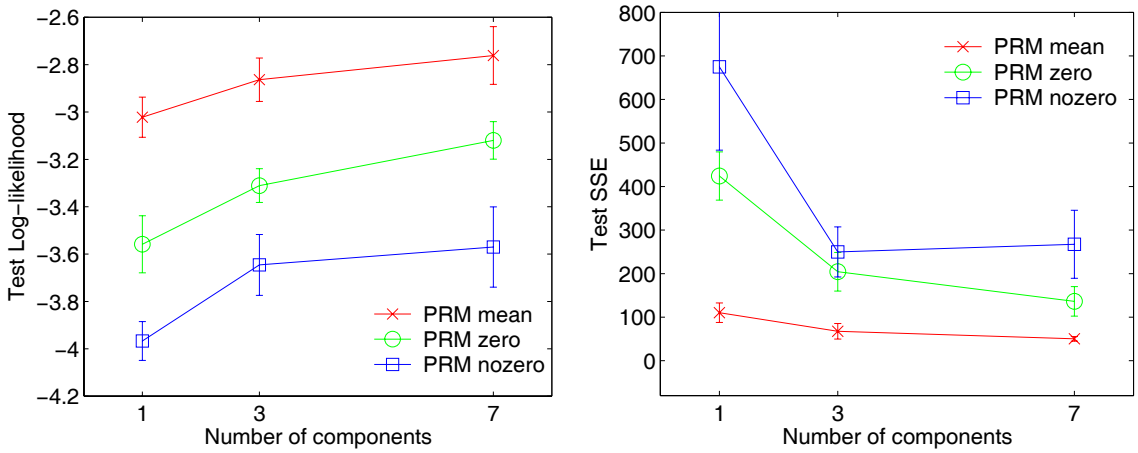


Figure 9.12: Cross-validation with PRM on `mean`, `zero`, and `nozero` cyclone data: (left) test log-likelihood, and (right) prediction SSE scores. Error bars denote one standard deviation on each side of the mean.

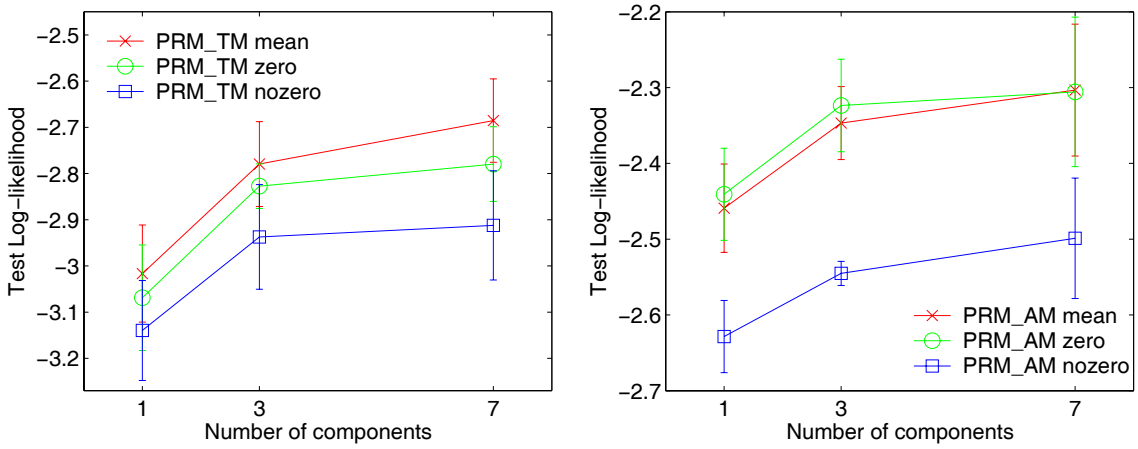


Figure 9.13: Cross-validation with (left) PRM_TM and (right) PRM_AM on `mean`, `zero`, and `nozero` data. Test log-likelihood scores are shown in each case.

curve is far behind). This distinction is blurred even more with PRM_AM which allows for scaling as well as translation in measurement space. The results indicate that there is less importance on the choice of zeroing or subtracting the mean for these models (the `nozero` technique still performs poor).

The results for the time-alignment models are similar to those of PRM since, for the most part, the time-alignment models cannot recover alignments in measurement

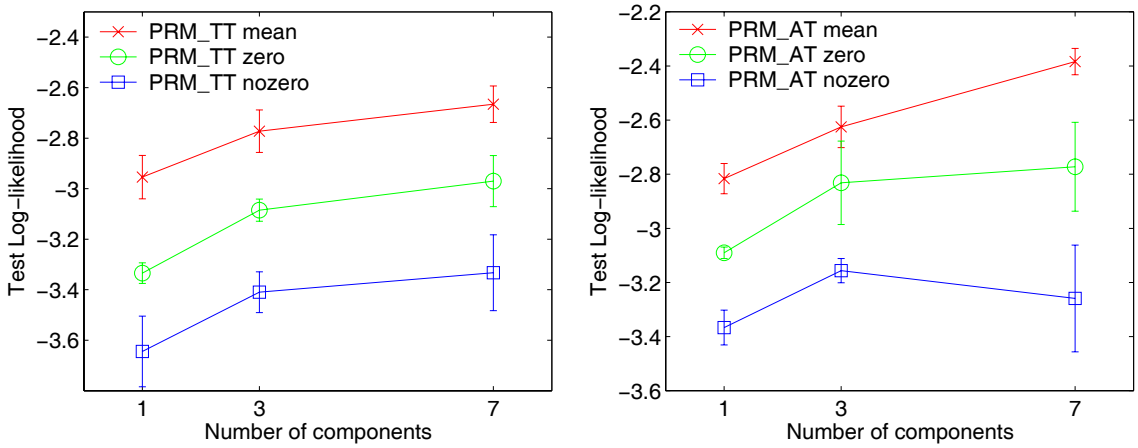


Figure 9.14: Cross-validation with PRM_TT (left) and PRM_AT (right) on `mean`, `zero`, and `nozero` data. Test log-likelihood scores are shown in each case.

space (technically, however, some transformations in measurement space can be accounted for by transformations in time). Figure 9.14 shows the results for PRM_TT (on the left) and PRM_AT (on the right) for the same preprocessing methods of `mean`, `zero`, and `nozero`. The results show a large performance gap between each of the methods; the best method again consists of subtracting the mean.

Models that allow for alignment in both measurement space and in time can also be employed for cyclone clustering. For example, the test results for PRM_TM_TT which jointly allows for translations in measurement space and in time are shown in Figure 9.15. The same three preprocessing techniques are again compared. The test log-likelihood and prediction SSE scores are both shown in the figure. The joint space- and time-alignment model does not seem largely affected by the choice of preprocessing. In fact, even `nozero` seems to match the performance of the other two. The ability to translate in measurement space and in time trumps the choice of arbitrary initial alignment.

Evaluation of the results leads to the conclusion that mean-subtraction performs best on average. The results for the `norm` and `znorm` preprocessing methods require

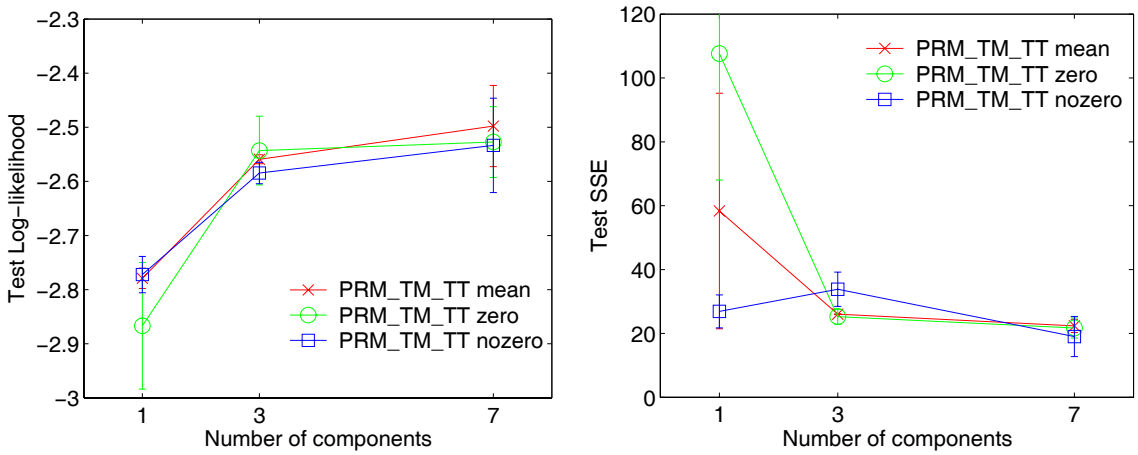


Figure 9.15: Cross-validation with PRM_TM_TT on `mean`, `zero`, and `nozero` data: (left) test log-likelihood, and (right) prediction SSE scores.

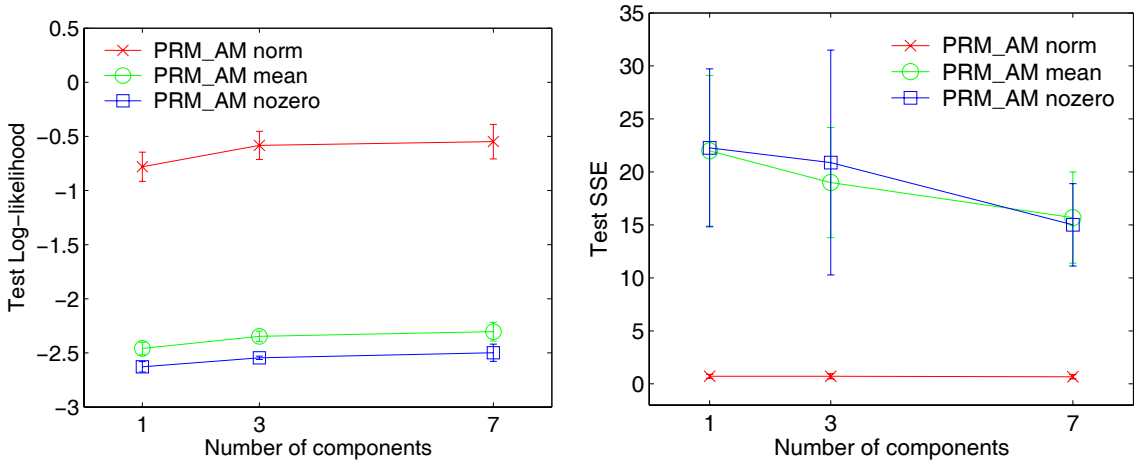


Figure 9.16: Cross-validation with PRM_AM on `norm`, `mean`, and `nozero` data: (left) test log-likelihood, and (right) prediction SSE scores. Notice the prior scaling performed in the `norm` case results in test scores that cannot be compared to the other methodologies.

special handling. The test scores for `norm` and `znorm` aren't directly comparable to the other methods. The prior scaling of the cyclone trajectories leads to learned density models that cannot be fairly compared to the learned density models resulting from training on an unscaled dataset. The complexity of the learned mixture density hampers any attempt at "unscaling" the resulting test scores.

Figure 9.16 shows the results of this effect from training PRM_AM on `norm`, `mean`, and `nozero` preprocessed cyclone data. Both the test log-likelihood and predicted SSE scores are given in the figure. Notice the large scale difference that arises between the score curves for `norm` and those of the other methods. An objective comparison between `norm` and `mean` cannot be made with these results.

In the next section, the three preprocessing techniques `mean`, `norm`, and `znorm` are evaluated on each of the alignment models in order to choose a best alignment model and preprocessing technique.

9.7.3 Choosing an alignment model

In this section, we report on the results that were used in choosing the specific clustering-alignment model. The experiments reported in this section were carried out in the same manner as those in the previous section. In particular, a random sample of 150 cyclones was chosen from the set of 614 (resulting in approximately 4,500 total training points). Each pair of preprocessing technique and clustering model was then executed on this sample. The resulting models were evaluated on a random hold-out set of 100 cyclones and the test log-likelihood and prediction SSE scores were recorded. This process was then repeated 10 times with the test scores averaged across the 10 different runs.

The model selection is commenced by first eliminating the `mean` technique in competition with the `norm` technique. The `norm` methodology is then compared to the best model arising from `znorm` data, with PRM_AM and `znorm` ultimately chosen as the best joint methodology.

Figure 9.17 shows a comparison between all four individual space- and time-alignment models for `mean` and `norm` preprocessing. PRM_AM performs best for the `mean` cyclones while the time-alignment model PRM_AT performs best for the

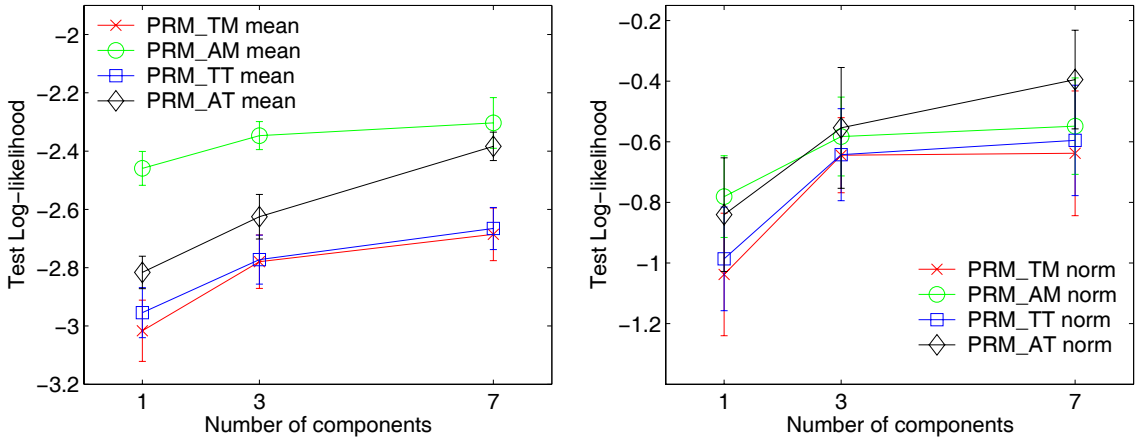


Figure 9.17: Cross-validation for each of the four individual space- and time-alignment models on (left) `mean` and (right) `norm` data.

`norm` cyclones. The time-alignment model is out-performed by PRM_AM with `mean` preprocessing since PRM_AT is not capable of handling the large scaling effects in measurement space. These scaling effects are important with this dataset.

However, once a rough estimate of the trajectory scaling is removed through the normalizing process, it appears that PRM_AT is able to take advantage of its ability to align in time to gain some performance. Even so, PRM_AM exhibits good performance in both of the `mean` and `norm` situations, which suggests that for this dataset the ability to find automatic time alignments is not as important as accounting for the transformations in measurement space. Though, with the measurement transformations roughly accounted for by the normalizing process, PRM_AT results in the best performance between the two competing methodologies.

The joint space- and time-alignment models can also be used for cyclone clustering. For example, Figure 9.18 reports the test log-likelihood scores on `mean` and `norm` cyclone data for a number of competing clustering models. Although PRM_TM_TT edges out the space-alignment model PRM_AM on `norm` data, it does not show enough potential improvement over PRM_AT to warrant the extra complexity in-

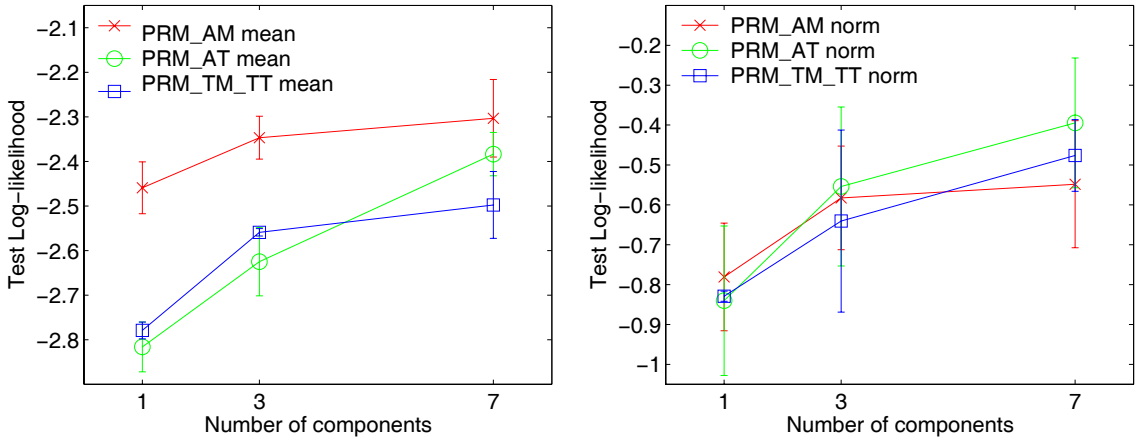


Figure 9.18: Comparison of PRM_TM_TT with the best individual alignment models on (left) `mean` and (right) `norm` data.

curred with this model. Results for the most complex joint space- and time-alignment model (PRM_AM_AT) do show improvement over PRM_AT; however, the improvement is not significant enough to justify the selection of the most complex model of the entire set.

The only remaining task is to compare PRM_AT to the best model resulting from the `znorm` process. Figure 9.19 highlights the results for the two best competing models. The left plot gives the test log-likelihood scores for PRM_AT on `norm` data and PRM_AM on `znorm` data. The log-likelihood results indicate that PRM_AT holds a slight advantage over PRM_AM as K increases.

The right plot depicts a different picture for the prediction SSE scores. PRM_AM shows a large performance advantage over PRM_AT that narrows as K increases. The log-likelihood measures the overall density modelling efficiency of the two methods, while the prediction SSE score measures how well the cyclone trajectories are represented by the cluster-specific curve models. Higher prediction accuracies lead to better modelling of each individual curve.

There is no "correct" choice in this case; however, there are several reasons why

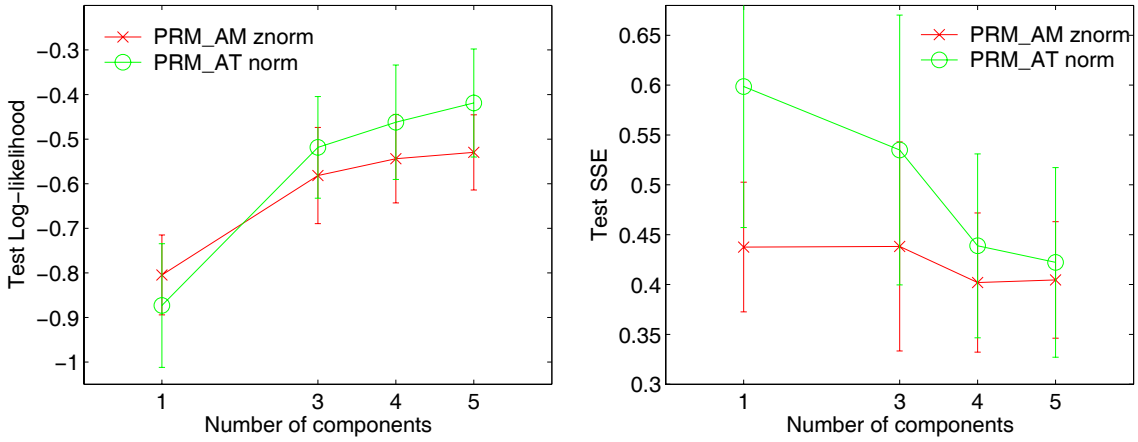


Figure 9.19: Cross-validation for the best competing models on `norm` and `znorm` data. (left) Test log-likelihood, and (right) prediction SSE scores.

PRM_AM can be seen as the better choice. First, the models both perform well on the cyclone dataset; however, PRM_AM is a simpler model, and thus it should be preferred on average. Second, earlier we noted that the most important alignment factor in this dataset is the scaling in measurement space. Time transformations do not seem to be particularly important from a modelling aspect with these cyclone trajectories. Thus, there isn't a good reason to move up to PRM_AT from PRM_AM. Finally, the prediction problem is more innately associated with cyclone analysis. Thus, PRM_AM should be preferred. PRM_AM with `znorm` preprocessing is what is used for all further analysis in this chapter.

9.7.4 Choosing K

An important question in cyclone clustering—and for clustering in general—is the selection of the number of cyclone clusters. Previous studies (e.g., Blender et al., 1997) have found it useful to set the number of clusters to three based on various meteorological considerations. In this section, we address the issue of choosing the optimal number of clusters in an objective fashion. Only the chosen alignment model

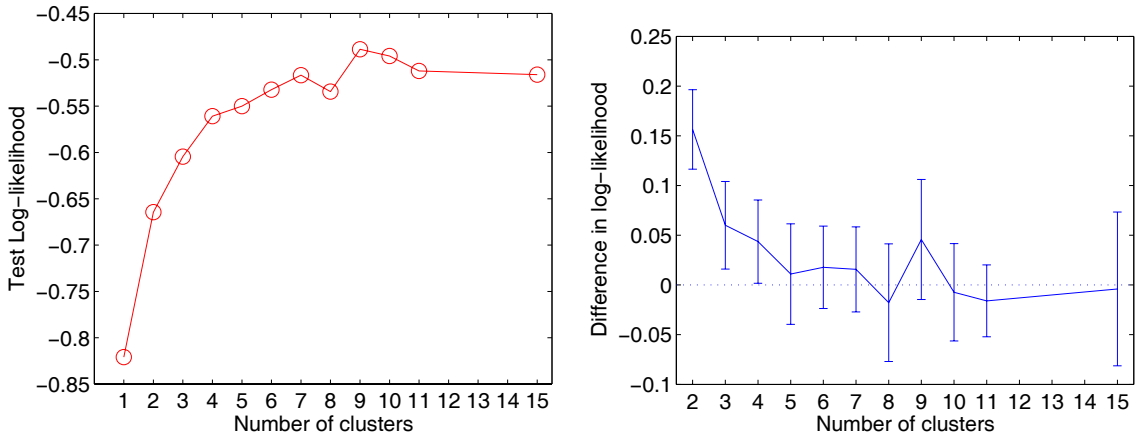


Figure 9.20: Cross-validation for PRM_AM on `znorm` cyclone data for various values of K. The test log-likelihood scores are shown in the left plot. The right plot is explained in the text. The maximum value attained over the experiment was at $K = 9$.

PRM_AM from above is considered in this section.

The reported experiments in this section were carried out in the same manner as in the previous sections. The only difference is that the results are based on twice as many runs (i.e., twenty different training and test sets were sampled, with the scores averaged over the twenty runs).

The left plot of Figure 9.20 shows the values of the recorded test log-likelihood scores for various values of K, from 1 up to 15. The value of $K = 9$ corresponds to the maximum log-likelihood point attained along the entire curve.

The right plot attempts to show how much better on average the value of K is over $K - 1$. The difference in the mean score from the value at K and at $K - 1$ is plotted (at K) along with the standard deviation in this difference over the experimental runs. A positive value indicates that the mean value at K is larger (or better in this case) than the value at $K - 1$. If the extent of the error bars do not cross below zero, then the value of K is "always" better than the value of $K - 1$. Note that even the "true" value of K (that is, when the true model is included in your model

space) may test below the value of $K-1$ for any particular training and testing set. Nonetheless, the plot gives a valuable view into the results.

The plot does not show any decrease in performance noticed over any of the runs until $K = 5$, though the average value at $K = 5$ still shows an increase in performance. The plot shows that the largest sustained drop-off in performance occurs after $K = 9$.

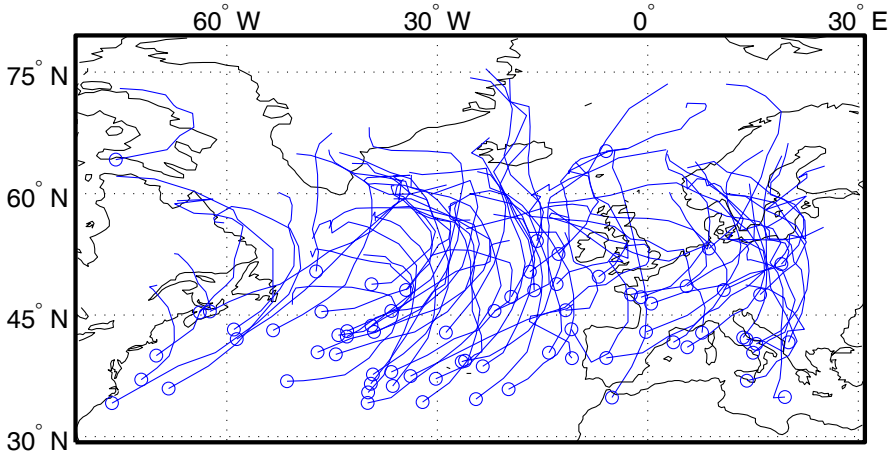
The competing value of $K = 7$ offers an alternative choice since there does not seem to be dramatic improvement for ensuing values of K. However, the results show that $K = 9$ is better on average, with the performance immediately tailing off after $K = 9$. Thus, the value of $K = 9$ is chosen as the "best" number of clusters for this dataset.

9.8 Clustering analysis

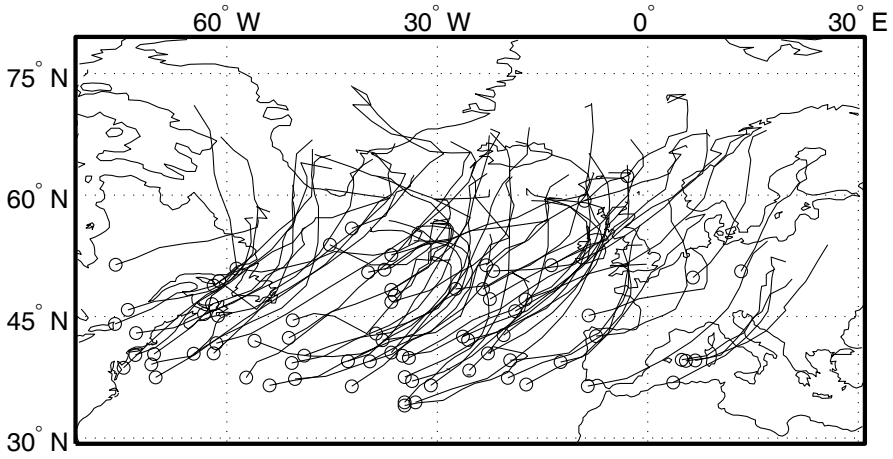
In this section, we analyze the cyclone clusters resulting from the application of PRM_AM to the `znorm` cyclone data with K set to nine. Both graphical and quantitative analysis of the clustering is given. Analysis of the daily temporal behavior of the cyclone clusters is also given towards the end of this section.

Figures 9.21 to 9.24 geographically depict the cyclones from each of the nine clusters. The clusters are organized into three main groups (vertical, diagonal, and horizontal) based on the mean orientation of the tracks within each cluster. A single highly variable background cluster is placed into its own group.

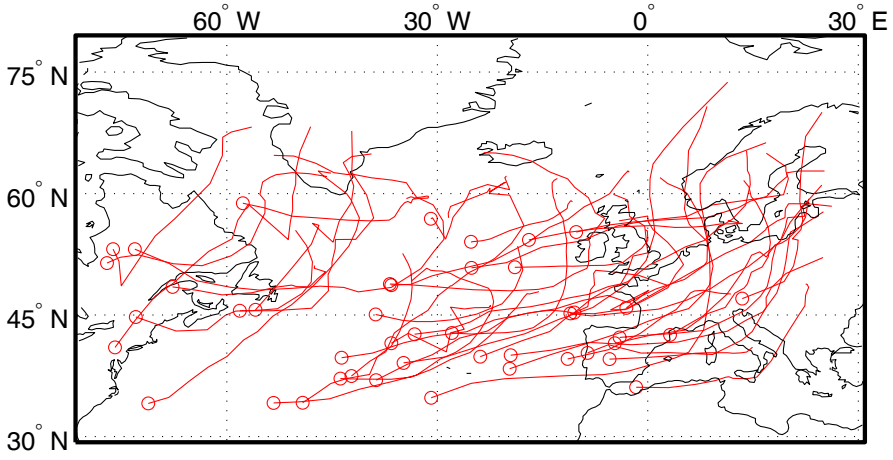
Names are given (in `typewriter` font) to each of the clusters that generally describe the shape of the tracks found in each cluster (two-letter abbreviations used in later figures are given in parentheses). For example, in Figure 9.21, the three north- or vertically-oriented clusters are shown. The `VertCurveWest` cluster consists



(a) `VertCurveWest` (VW)



(b) `VertCurveNorth` (VN)



(c) `VertBend` (VB)

Figure 9.21: Northward moving cyclone clusters.

of tracks that begin moving northeast and curve towards the northwest. The tracks in `VertCurveNorth` begin moving northeast and curve to a final due north orientation. The remaining north-oriented cluster `VertBend` consists of tracks that primarily being moving east and then finish with a bend to the north.

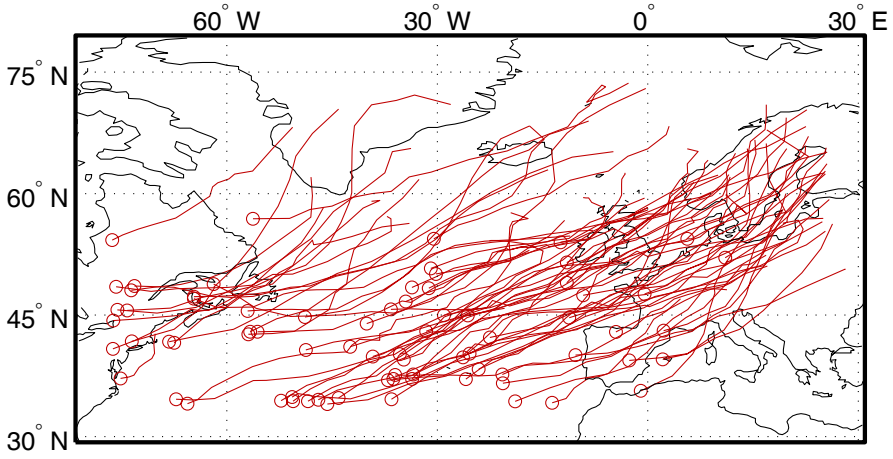
Similarly, in Figure 9.22, the three northeast- or diagonally-oriented clusters are shown. `DiagStraight` consists of tracks that begin and end moving in a northeast direction. Cluster `DiagBend` consists of tracks that begin moving northeast and then abruptly bend to the east primarily before they reach the 60th parallel. `DiagTurn` consists of tracks which begin moving northeast and undergo some sort of turning action before they tail off into an east-northeast direction; many of these tracks have an S-curve shape to them.

The two east- or horizontally-oriented clusters are shown in Figure 9.23. The `HorzWave` cluster consists of tracks that resemble the shape of an ocean wave. The `HorzTail` cluster consists of tracks that primarily begin moving northeast and then tail-off in a due east direction. Finally, Figure 9.24 shows the remaining cluster `Back` which consists of highly variable tracks that tend to meander in several directions over time.

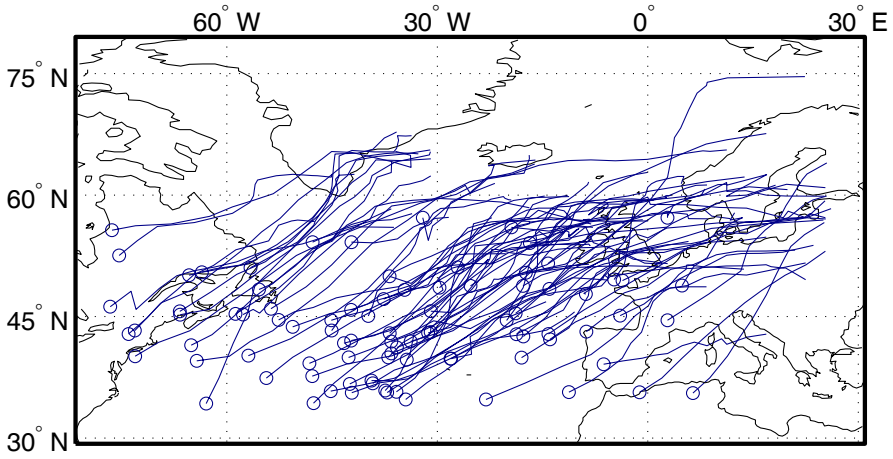
9.8.1 Cluster descriptions

In this section, several figures and tables reporting cluster-specific statistics are initially introduced. This is followed by a number of subsections which provide individual analysis of each cluster.

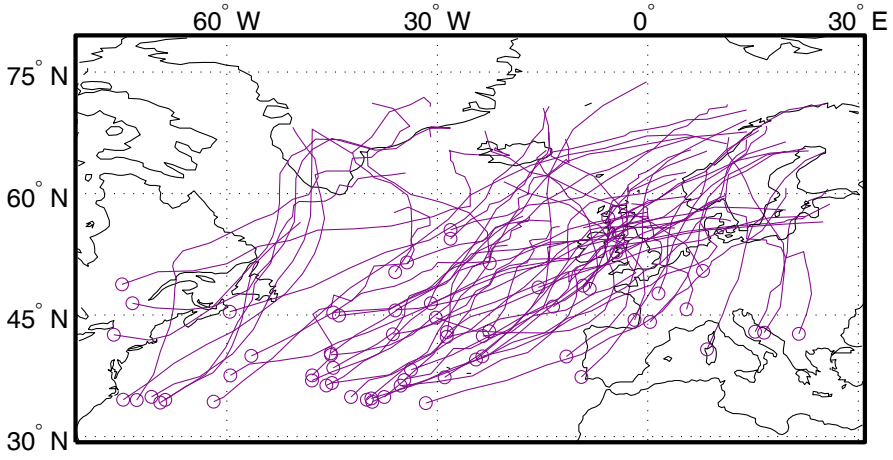
A number of empirically-derived summary statistics for each of the nine clusters are given in Table 9.4. The values under the columns give cluster-wide means and deviations of the summary statistics. For example, μ for column `VertCurveWest` in Table 9.4 reports the mean of all the minimum intensities attained by the cyclones in



(a) `DiagStraight` (DS)

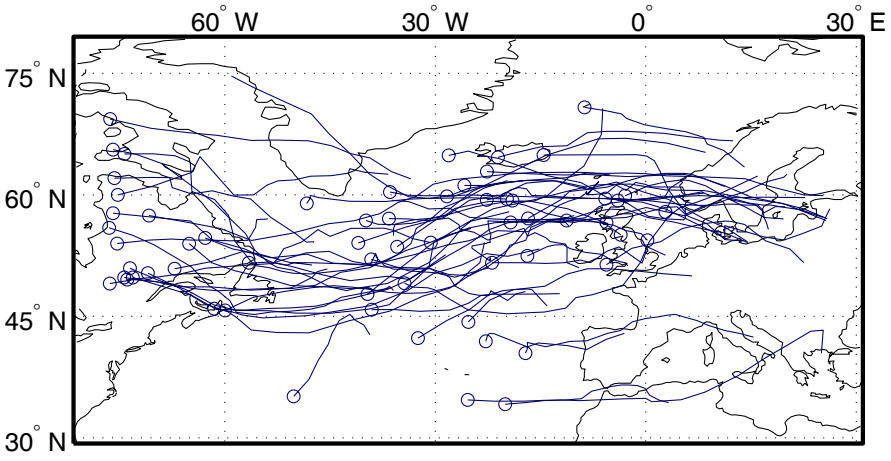


(b) `DiagBend` (DB)

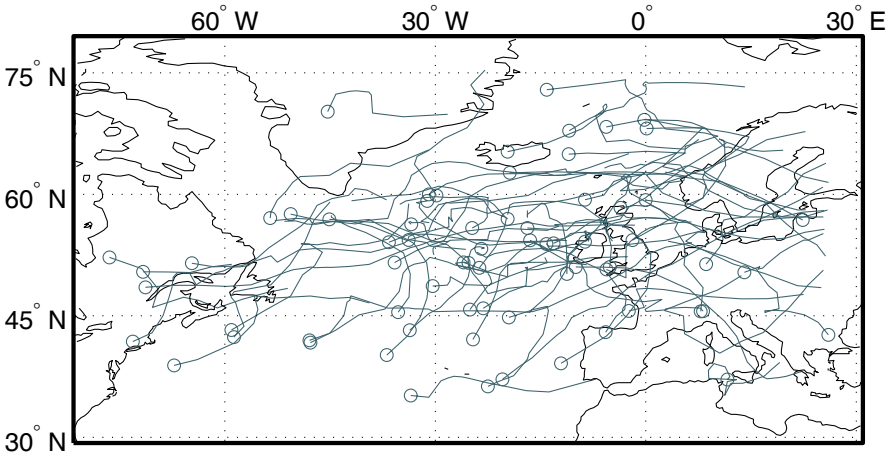


(c) `DiagTurn` (DT)

Figure 9.22: Northeastward moving cyclone clusters.

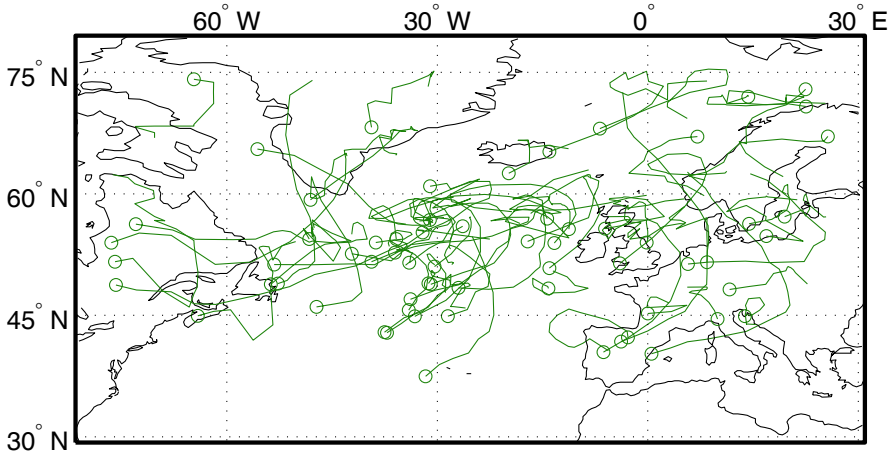


(a) `HorzWave` (HW)



(b) `HorzTail` (HT)

Figure 9.23: Eastward moving cyclone clusters.



(a) `Back` (BK)

Figure 9.24: Background or noise cluster.

cluster `VertCurveWest`. The standard deviation for this value is shown under σ for the same column. Bolded entries denote the largest value among all of the clusters and underlined entries denote the smallest (for intensity, bolded denotes the most intense or most negative value). Three of the cyclone-specific statistics require more detailed explanation.

- The *average acceleration* of a cyclone is calculated using the *absolute* rate of change of velocity for each trajectory since interest lies both in the increase and the decrease of acceleration.
- The *curvature* of a cyclone is calculated by taking the average of the instantaneous curvature values along the trajectory. Instantaneous curvature is defined in the standard way as $|d\varphi/ds|$ where φ is the angle of inclination at a time point and the derivative is taken with respect to the displacement s. Note that a straight line has a curvature of 0 and a circle has constant curvature.
- The *instability* of a cyclone estimates the degree of "erratic" departure from a smooth path (whether straight or otherwise). Instability in effect is the standard deviation of instantaneous curvature along a cyclone's trajectory.

Figure 9.25 shows the distributions for cyclone duration found in each cluster. The graph shows the number of cyclones in each cluster as a function of cyclone

Table 9.4: Cluster-wide averaged measures of various cyclone-specific statistics for the nine clusters. Both means (μ) and standard deviations (σ) are given for each cluster column. Bolded entries give the largest value among all nine clusters and underlined entries give the smallest. The units are given as follows: minimum intensity (mb), velocity (km/h), absolute acceleration (km/h^2), duration (days), curvature (radians/km$\times 10^{-3}$), and instability (radians/km$\times 10^{-3}$)

Cyclone statistics	VertCurveWest		VertCurveNorth		VertBend	
	μ	σ	μ	σ	μ	σ
Intensity	**-41.71**	8.82	-39.79	8.88	-37.69	8.41
Velocity	46.32	12.82	47.61	13.43	46.50	11.54
Acceleration	16.01	6.09	15.97	5.74	16.35	5.37
Duration	3.64	0.98	<u>3.49</u>	0.74	4.22	1.14
Curvature	1.78	1.81	1.82	1.83	3.63	5.01
Instability	2.66	3.68	2.89	3.93	6.90	12.31

Cyclone statistics	DiagStraight		DiagBend		DiagTurn	
	μ	σ	μ	σ	μ	σ
Intensity	-37.07	8.77	-39.30	8.21	-41.70	8.12
Velocity	**58.46**	16.21	54.24	16.34	50.61	14.18
Acceleration	16.20	6.72	16.44	5.29	<u>15.34</u>	4.26
Duration	4.13	1.32	3.55	0.81	**4.43**	1.22
Curvature	<u>1.10</u>	1.36	2.31	2.70	1.37	1.43
Instability	<u>1.81</u>	2.32	4.30	5.65	2.24	3.34

Cyclone statistics	HorzWave		HorzTail		Back	
	μ	σ	μ	σ	μ	σ
Intensity	<u>-33.92</u>	7.75	-35.19	6.90	-36.23	9.05
Velocity	54.21	16.02	42.13	13.73	<u>33.27</u>	11.14
Acceleration	17.96	6.90	**19.59**	7.57	17.44	7.95
Duration	3.71	1.18	3.70	1.19	3.74	1.23
Curvature	3.01	2.41	5.59	7.00	**7.17**	9.54
Instability	5.65	5.24	9.42	14.49	**11.66**	19.45

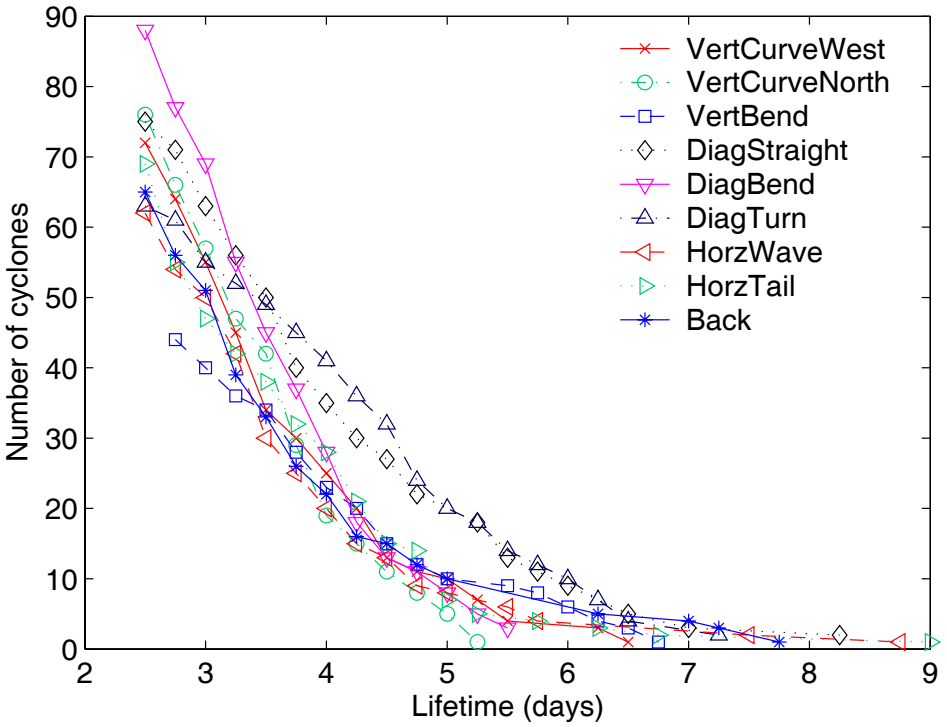


Figure 9.25: Number of cyclones in each cluster as a function of cyclone lifetime. The graph shows the lifetime decay rate for each cluster.

duration. The curves graphically show the the lifetime decay rates for each cluster (or more specifically, the average lifetime decay rate for the *cyclones* of each cluster).

At $t = 2.5$ days, the plotted values reflect the total number of cyclones in each cluster with a lifetime of at least 2.5 days (the minimum required length for any tracked cyclone). At $t = 4$ days, the figure shows that less than 50% of the original number of cyclones are still active.

There appear to be two main decay rates shared by the clusters. The majority of the clusters exhibit a large decay rate such as that shown by `DiagBend`. Two of the clusters, `DiagTurn`, and `DiagStraight`, (and also to some degree `VertBend`) show a much lower decay rate. These clusters also demonstrate larger then average velocities and (lower than average) minimum intensities which could provide a cause for their low decay rates.

Below, the nine clusters are discussed in detail using the cluster maps, the table of data, and the figure as references.

`VertCurveWest`

The cyclones in this cluster initially begin moving in a northeast direction and then curve to the northwest. The cluster is highly shape-consistent. It contains the most intense cyclones (-41.71 mb) of all nine clusters. The genesis region is primarily south of 47° latitude with a mean genesis point of 43°N by 24°W. This cluster is one of the largest (fourth, with 72 cyclones) and consists of below average-length cyclone tracks.

`VertCurveNorth`

The cyclones in this cluster begin moving ease-northeast and tail-off to due north. Closer inspection reveals that they begin moving in a more easterly direction than those of the previous cluster. This cluster contains the overall shortest duration cyclones on average, while exhibiting the third most intense group of cyclones. This cluster has the largest cluster-average cyclone speed among the three vertically-oriented clusters. The genesis region is a bit more westward than that of `VertCurveWest`, with a mean genesis point of 43°N by 37°W. This cluster is the second-largest overall with a total of 76 cyclones. It also exhibits the second-largest cyclone lifetime decay rate.

`VertBend`

This cluster consists of cyclones that primarily begin moving along an eastward direction with an abrupt bend to the north. Unlike `VertCurveWest`, this cluster is about average on the shape-consistency scale. These cyclones are the least intense

of those from the three vertically-oriented clusters. While it is the smallest of all nine clusters with 44 cyclones, it has the longest duration cyclones, thus giving a low lifetime decay rate. The mean genesis point is 44°N by 33°W, similar to that of `VertCurveNorth`.

`DiagStraight`

The cyclones in this cluster move due northeast for their complete lifetime duration. This cluster has the lowest curvature and is the most stable among all nine clusters. The cyclones of this cluster have the largest overall velocity (58.46 km/h), yet they exhibit below average intensity. This cluster is the third largest overall with 75 cyclones. It's lifetime decay rate is significantly lower than the other clusters (except for `DiagTurn`), resulting in a large number of longer-duration cyclones. The primary genesis region is south of 45°N latitude and west of 30°W longitude with a mean genesis point of 42°N by 38°W.

`DiagBend`

This cluster consists of cyclones that begin in a similar fashion as those of `DiagStraight` but then abruptly bend east, primarily south of 60°N latitude. These cyclone are a bit more intense than those of `DiagStraight` but are somewhat slower on average also. This cluster is the largest overall with 88 cyclones. Its lifetime decay rate is the largest among all nine clusters resulting in the cluster having the second shortest-duration set of cyclones. The genesis region is tightly focused in the southwest region of the North Atlantic with a mean genesis point of 44°N by 36°W.

`DiagTurn`

The cyclones in this cluster begin and end moving east-northeast; however, they exhibit some sort of turning action in the middle lifetime. In fact, many of these tracks have an S-curve shape to them. In relation to the three other diagonal clusters, this cluster contains the slowest moving cyclones. Yet, these cyclones are the second-most intense compared to those in all nine clusters. It also consists of the longest duration cyclones overall, and it has the overall lowest average lifetime decay rate. This cluster is the third smallest with 63 cyclones. The genesis region is not well compacted; the mean genesis point is 42°N by 32°W.

`HorzWave`

This cluster consists of cyclone tracks that trace a horizontal S-curve. These cyclones are the least intense, yet they move at considerable speed (third overall). This is only the second largest cluster with 62 cyclones. The curvature and instability measures are very low, second only to the diagonal cluster `DiagStraight`. This cluster initially shows one of the largest lifetime decay rates, but this steadily decreases over time due to the existence of extremely long duration cyclones that propagate across the North Atlantic. The genesis region is almost completely north of the 45th parallel with a mean genesis point of 54°N by 39°W.

`HorzTail`

The cyclones of this cluster mainly begin moving northeast and then tail-off in a due east direction. The cluster is not very shape-consistent. The cyclones of this cluster have high curvature and exhibit large instability. They show the largest absolute acceleration, speeding-up and slowing-down often. Compared to `HorzWave`, the cyclones of this cluster are much slower on average (54 km/h to 42 km/h). The

cluster is of average size, having 69 cyclones. The genesis region is the most eastward of all nine clusters with a mean genesis point of 51°N by 23°W.

Back

The cyclones in this cluster give highly variable, meandering tracks. The cluster consists of cyclones that are nearly the least intense, the slowest, the most unstable, and show the largest curvature. The genesis region is almost exactly centered in the North Atlantic with a mean genesis point of 53°N by 23°W. This cluster soaks up all of those cyclones that don't fit anywhere else.

9.8.2 Temporal analysis of cyclone clusters

In this section we analyze the daily temporal behavior of ETC clusters. We classify each day as being in one of ten cluster regimes corresponding to the nine clusters, or in the tenth case, to none of the clusters. The following heuristic procedure is used to perform the classification. If only a single cluster is active on a given day, then we assign that day to the regime corresponding to that cluster. On the other hand, if no cluster is active then the day is assigned to the quiescent regime. For days with overlap, the regime corresponding to the cluster with the largest number of active cyclones on that day is chosen. In the case of a tie between two or more active clusters, the regime which was most recently selected corresponding to one of the "tied" clusters is chosen (this can be thought of as a type of "momentum bias").

When applied to the clustered ETC data, this assignment procedure yields the daily regime sequences shown in Figure 9.26. The plot in this figure has fifteen rows depicting the regime classification for each day in each of the fifteen winters (top-to-bottom) from 1980–1994. The key at the right gives the gray-value-to-regime mapping. A number of things can be seen from this picture—for example, there is

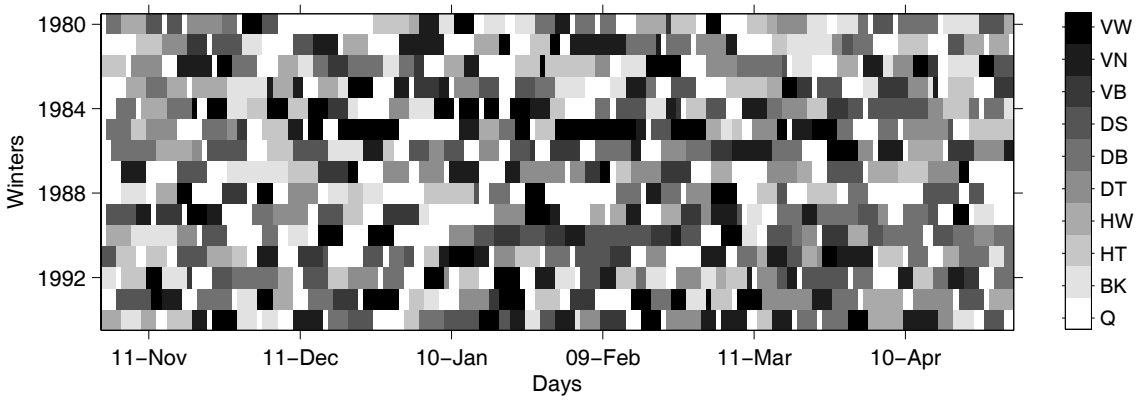


Figure 9.26: Daily regime classification for winters (top-to-bottom) 1980–1994. Each of the fifteen rows depicts the regime classification for each day in a winter. The key at the right gives the gray-value-to-regime mapping where the two-letter abbreviations denote the names of the cluster regimes as defined in Figures 9.21–9.24. Q represents the quiescent regime.

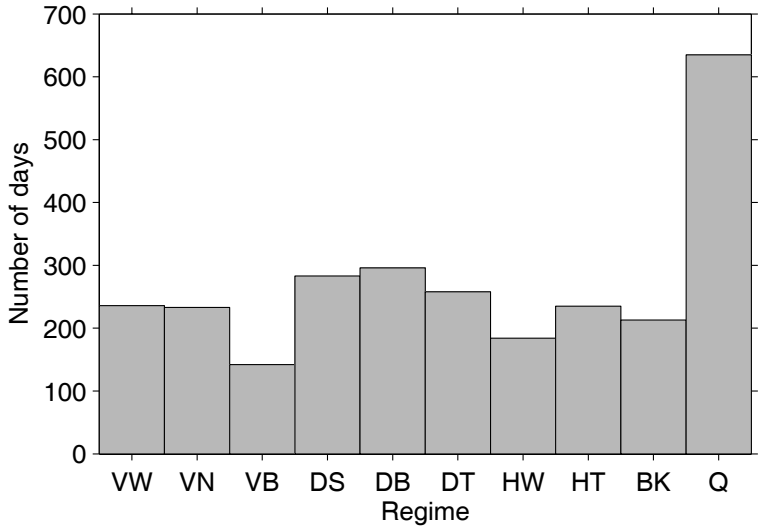


Figure 9.27: Histogram of regime activity corresponding to the regime classification in Figure 9.26.

considerably more regime activity resulting from the vertically oriented clusters in winters 1984 and 1985 than there is in winters 1980 and 1988.

The distribution of regime activity corresponding to Figure 9.26 is plotted in Figure 9.27. This plot gives the number of days that each regime is active over all

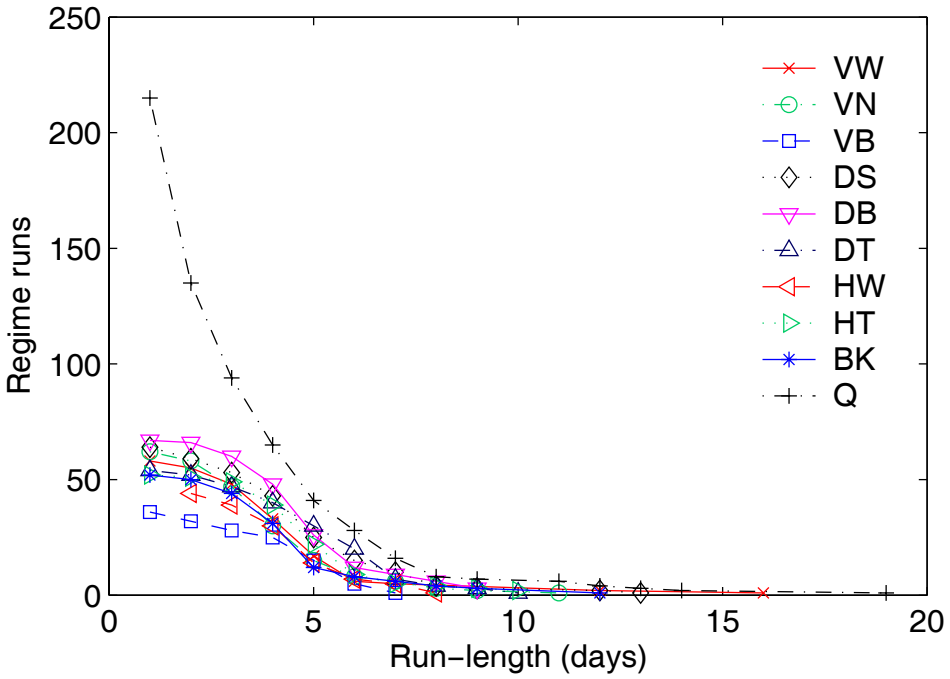


Figure 9.28: Distribution of run-lengths for each regime corresponding to the regime classifications in Figure 9.26.

winters. Of note is the relatively large size of the quiescent regime, accounting for about 22% of all days.

Another interesting feature of Figure 9.26 is the distribution of regime run-lengths. Figure 9.28 shows the decay rate (distribution of run-lengths) for each regime. The run-lengths for the quiescent regime Q decay in an exponential fashion. The cluster regimes have different persistence with run-length distributions that are non-geometric in nature: runs of length 1, 2, and 3 days are roughly equally likely—for run-lengths of 4 days or longer the distribution starts to decay in an exponential fashion. The diagonal regimes appear to have slightly more persistence (a tendency for longer runs) than the other regimes.

A couple of considerations arise if we consider the regime sequences as arising from a stochastic process. First, the regime process appears to demonstrate some

form of memory. And second, the process does not appear to demonstrate first-order Markov dependence. If the process were first-order Markov we would expect to see an approximate geometric distribution of run-lengths for each regime. The non-geometric distribution of run-lengths can be at least partly attributed to individual cyclone duration and sparsity of cyclone genesis. Since on average cyclones have a duration of roughly 4 days and since on average new cyclones are only generated approximately every 18 days, cyclone duration will play a large role in the run-length distribution.

9.9 Summary

In this chapter, an application of our joint clustering-alignment methodology to extra-tropical cyclone trajectories was presented. A methodology was developed for the detection and tracking of cyclones from mean sea-level pressure (MSLP) data generated by a general circulation model (GCM). The detection and tracking procedures applied to the output from the NCAR-CCM3 general circulation model resulted in a dataset of 614 cyclone trajectories.

A number of experiments were conducted to test the suitability of various trajectory preprocessing methods for application with our clustering-alignment models. Five different trajectory preprocessing techniques were investigated. Subtracting the mean was shown to be generally applicable for many of the clustering models. However, the best model/preprocessing combination was found to be with PRM_AM and `znorm` preprocessing.

A further set of experiments showed that the optimal number of clusters for use with PRM_AM on the cyclone data was found to be nine. The resulting nine clusters naturally grouped into four "super-clusters": vertically-oriented, diagonally-

oriented, horizontally-oriented, and one background cluster. These clusters were separately analyzed using various quantitative statistical quantities.

Regime classifications were assigned to each of the days in the winters from 1980 to 1994. The classifications were based on the cluster with the most active cyclones on each day. The daily regime classifications were plotted and summary statistics describing the distributions of regimes and their run-lengths were reported. The distribution of run-lengths suggest a process with some form of memory, not associated with first-order Markov.

Chapter 10

Clustering Observed Tropical Cyclones

10.1 Introduction

In this chapter, we describe the application of our clustering models to an "observed" tropical cyclone dataset. The application is shown to group cyclones based on identifiable characteristics such as speed, acceleration, duration, and track-type. The clusters are also shown to likely correspond to known states of circulation in the atmosphere such as that associated with a reverse-oriented monsoon trough (Lander, 1996).

This chapter stands in contrast to the previous chapter in several respects. First, the dataset analyzed in this chapter consists of actual observed data from the "real-world", not data generated from a mathematical GCM model. An interesting question is whether the cyclone dataset of the previous chapter that was implicitly generated during the run of the CCM3 GCM, closely matches data from actual observed cyclones. Second, the dataset in this chapter does not require the development of

an identification or tracking methodology since it already consists of the cyclone trajectories themselves. Finally, the cyclones analyzed in this chapter are tropical cyclones (as opposed to extra-tropical) that originate in the western North Pacific Ocean, just east of China and the Philippines.

This chapter is organized as follows. In Section 10.2, the problem definition and motivation is given along with a discussion of prior work on tropical cyclone analysis. Section 10.2 describes the JTWC (Joint Typhoon Warning Center) Western North Pacific Best Track dataset that was used for the results described in this chapter.

In Section 10.4, the model selection problem with the tropical cyclone dataset is addressed. As in the previous chapter, this section makes up the bulk of the experimental work with the alignment models. Experimental results are reported that were used to make decisions about the optimal order of the cyclone regression models, the most suitable type of trajectory preprocessing, the best predictive alignment model, and the number of clusters that best describes the tropical cyclone dataset.

In Section 10.5, detailed analysis of the results from the application of the selected model to the JTWC cyclone dataset is given. Graphical, statistical, and temporal analysis of the resulting clustering is reported. Finally, the chapter is concluded with a summary in Section 10.6.

10.2 Problem definition and prior work

Tropical cyclones over the western North Pacific are the cause of much damage in Southeast Asia. A better understanding of their structure and behavior may lead to improved track prediction and/or warning indicators, yielding positive impacts upon society.

Scientists are interested in tropical cyclones as they relate to the large-scale

circulation of the atmosphere and their effects on regional climate. Changes in activity, genesis location, and track-type are influenced by the large-scale circulation present during cyclone lifetime (Hodanish & Gray, 1993).

Harr and Elsberry (1995a) attempted to characterize the variability in the large-scale circulation by modelling the continuous circulation by a small set of recurrent patterns or clusters. These clusters were used to establish relationships between the large-scale circulation and cyclone characteristics. Four cyclone track-types were identified and contingency tables relating cyclone characteristics (such as track type) to cluster number were built and analyzed. In particular, the results indicate a significant relationship between circulation pattern and both track-type and genesis location.

In a companion paper, Harr and Elsberry (1995b) fit a Markov model to the transitions of the large-scale circulation among the identified cluster patterns. They demonstrated that the links between clusters and cyclone characteristics remain intact during transitions between the cluster patterns. This work prompts our analysis of the temporal behavior of the reported cyclone clusters in this chapter.

Taking the opposite tack, Harr and Elsberry (1991) build composites of wind data according to the types of cyclones active at any point in time and show that this leads to informative clusters of large-scale circulation as described by the wind composites. They show that relatively accurate predictions of track-type can be made by considering the genesis location and the associated composite clusters.

Hodanish and Gray (1993) provide a detailed study of cyclone track recurvature. The Joint Typhoon Warning Center (JTWC) classifies tropical cyclone tracks as either recurving or nonrecurving. The point of recurvature is that point at which a tropical cyclone changes from a north-northwest heading to a north-northeast heading. Hodanish and Gray identified four types of cyclones: sharply recurving,

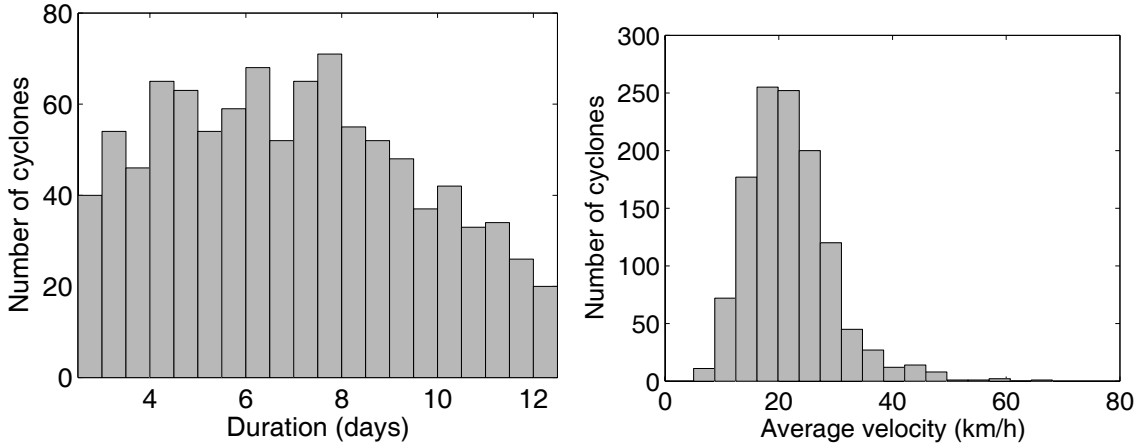
gradually recurving, left-turning, and nonrecurving. Using observed wind data from North Pacific rawindsondes, the circumstances that favor each of the four track types for tropical cyclones are investigated. The quantitative analysis reveals direct links between synoptic-scale circulation and cyclone track type.

This and other prior work demonstrates a connection between identified cyclone characteristics (whether clustered or not) and the large-scale circulation (e.g., as measured by various wind fields). What has not been done is an objective analysis of the type and number of cyclone clusters that exist in the tropical North Pacific. If cyclone clusters are required for particular analyses, then most authors usually group the cyclones based on whether or not they follow a straight or curved path over a particular area of interest. This grouping may be fine for many types of analyses; however, an objective out-of-sample analysis of the type and number of clusters in the tropical North Pacific is pursued in this chapter.

10.3 Best Track dataset

The dataset used in this chapter is a subset of the Best Track data compiled by the JTWC. The JTWC Western North Pacific Best Track dataset consists of six-hourly observations for tropical cyclones which occur in the western North Pacific. The subset reported on in this chapter covers the years from 1950 to 2001 during the months from June to November. Only tropical cyclones that reach tropical storm intensity or higher (a minimum wind speed of 33 knots) are included. The dataset contains 1,198 tropical cyclones of varying duration giving a total of 72,356 observations.

A further filtering processed was employed to reduce the dataset size and remove the outlier cyclones. All those cyclones that had a duration of less than 2.5 days or



(a) Cyclone duration histogram

(b) Average velocity histogram

Figure 10.1: Summary histograms describing the JTWC cyclone dataset.

more than 12.5 days were discarded. This process resulted in a set of 984 cyclones giving a total of 55,934 observations.

Summary histograms of this dataset are shown in Figure 10.1. The average North Pacific cyclone in this dataset has a duration of approximately 7 days. This is twice as long as those of the North Atlantic ETCs in the previous chapter which have an average duration of only 3.5 days. The average velocity for the tropical cyclones is about 22 km/h, whereas the ETCs show an average velocity of 48 km/h.

In Figure 10.2, a map of the genesis points for all cyclones is shown; genesis points are plotted by small circles. Tropical western North Pacific cyclones share a fairly compact genesis region south of 30°N and west of 180°E.

A map showing all observed cyclone tracks from 1990 to 2001 is given in Figure 10.3. The two major types of cyclone tracks can be seen in the map: (a) predominantly straight path tracks that approach and make landfall in the Philippines, Vietnam, and Southern China, and (b) "recurving" tracks that either make landfall over the Korean Peninsula or Japan, or head back eastward to open ocean.

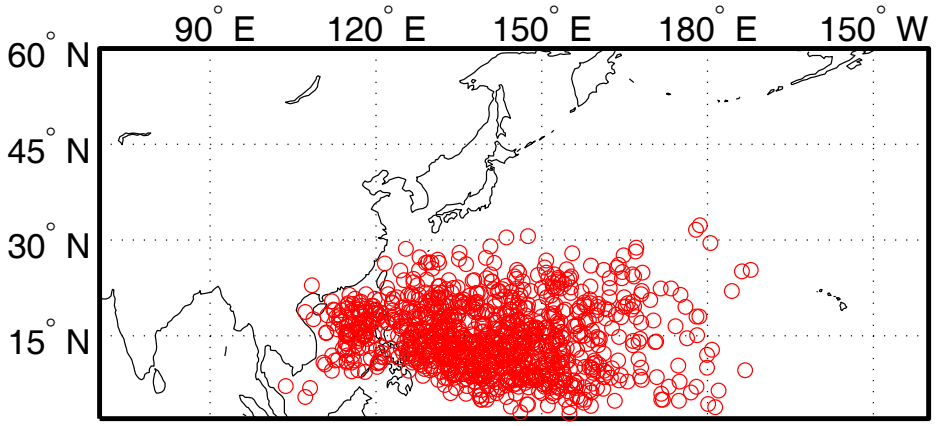


Figure 10.2: Map showing the genesis points (plotted by circles) of every cyclone in our selected subset of the Best Track dataset.

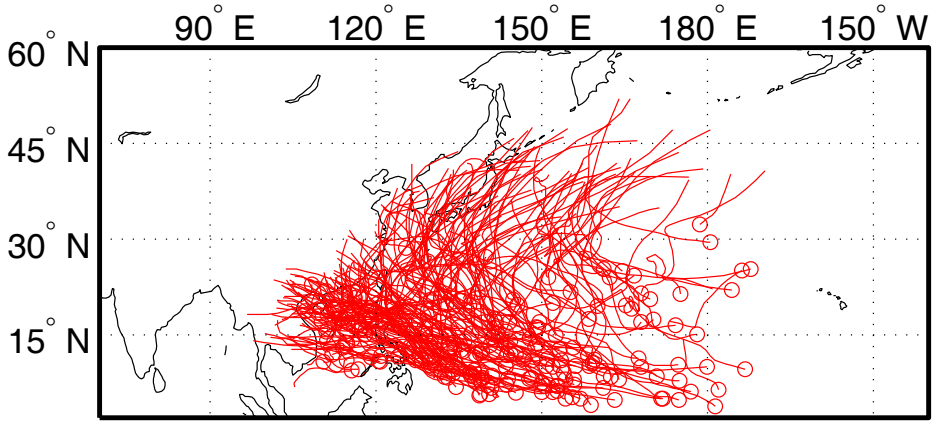


Figure 10.3: Map of all cyclone tracks from 1990 to 2001.

10.4 Model selection

In this section, our clustering algorithms are applied to the JTWC cyclone dataset with the goal of choosing an optimal methodology that is used in all further analysis. The model selection process can be broken down into four steps. The four steps involve choosing the order and type of cyclone regression model, the employed preprocessing method (if any at all), the type of alignment model, and the number of clusters that best describes the underlying cyclone dataset. These issues are discussed in the following four subsections.

10.4.1 Choosing the order of regression model

The cyclone regression models of Section 9.6 are again used to model the tropical cyclones of this chapter. All that remains is to choose the order of the regression model that best describes the North Pacific tropical cyclones. The order of the regression model is chosen based on the results of cross-validation experiments. The experiments reported here were run only with PRM. The results for the other models are similar.

The experiments were carried out as follows. A random sample of 50 cyclones was selected from the JTWC dataset. PRM was trained on this dataset using polynomials of linear to cubic, and over the K values from 1 to 4. These trained models were evaluated on a random hold-out set of 50 cyclones and test log-likelihood scores were recorded. This procedure was repeated 10 times and the scores were averaged across the runs. The results from these experiments are shown in Table 10.1. The highest score is achieved with quadratic polynomials across all values of K.

Table 10.1: Test log-likelihood scores from PRM on the cyclone data for K-values 1 to 4 and fitted polynomials of linear to cubic. A quadratic fit achieves the highest score for all values of K.

K	Linear	Quadratic	Cubic
1	-3.3886	-3.3885	-3.3897
2	-3.1604	-3.1545	-3.1565
3	-3.0847	-3.0788	-3.0814
4	-3.0201	-3.0049	-3.0259

10.4.2 Choosing the alignment model

The effect of various types of preprocessing on the output of the clustering-alignment models was discussed in detail in Section 9.7.2. Clusters that share similar shape characteristics are also sought in the tropical cyclone case. The five different types of trajectory preprocessing defined in Section 9.7.2 are also considered for preprocessing with the tropical cyclones.

In short, the results reported in this section show much similarity to those generated with the ETC data. This fact lends support to the validity of GCMs for climate modelling and prediction. The discussion of the selection of preprocessing method and alignment model is consolidated in this section, since the individual effects in each case were discussed in the previous chapter (as noted above).

The results given below were generated in the exact same manner as those in Section 9.7.2. Namely, a sample of 150 cyclones was chosen at random from the complete set of cyclones. On average, this results in a training set of roughly 9,000 individual observations. Each pair of preprocessing technique and clustering model was evaluated on the training sample. The trained models were then scored on a separate hold-out set of 100 cyclones and the test log-likelihood and prediction SSE scores were recorded. This process was repeated 10 times with the test scores averaged across the 10 runs.

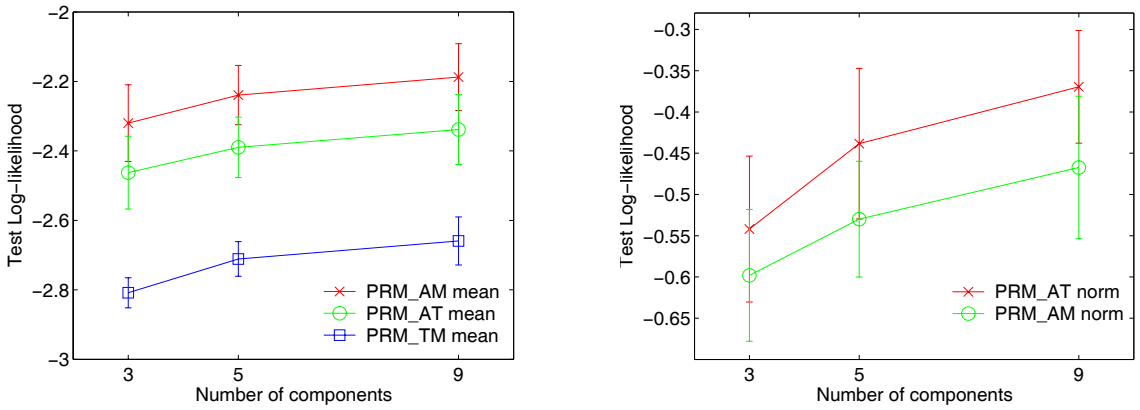


Figure 10.4: Test log-likelihood scores for the best performing `mean` (left) and `norm` (right) alignment models.

Since direct comparison between the normalizing techniques (`norm` and `znorm`) and the subtraction techniques (`nozero`, `zero`, and `mean`) is not possible due to the scaling effects discussed in Section 9.7.2, the results cannot be presented in a single graph representing all of the models. Instead, the relevant results are shown over two figures. The first figure is used to choose the `norm`-based models over those of `mean`, and the second figure is used to choose between the top competing models of `norm` and `znorm`.

The best performing models for both the `mean` (left) and `norm` (right) methods are shown in Figure 10.4. None of the `nozero`- or `zero`-based models are shown since they do not compete with the `mean`-based models.

PRM_AM performs best for the `mean` cyclones but it is out-performed on the `norm` cyclones by the time-alignment model PRM_AT. This resembles the situation reported in Figure 9.17 of Section 9.7.3 for the ETC dataset.

The time-alignment model PRM_AT is out-performed by PRM_AM under `mean` preprocessing since PRM_AT is not capable of detecting the large scaling effects present in measurement space. But just as with the ETC dataset, once a rough estimate of the trajectory scaling is removed through the normalizing process, it

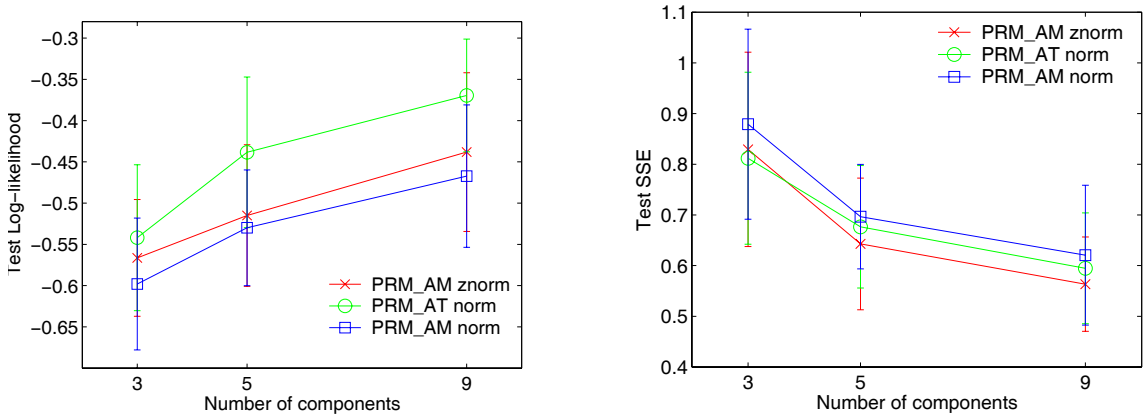


Figure 10.5: Test log-likelihood (left) and prediction SSE (right) scores for the best performing `norm` and `znorm` clustering models.

appears that PRM_AT is able to take advantage of its ability to align in time to gain a performance edge.

Comparisons with the joint space- and time-alignment models were not systematically carried out for this dataset in the interest of computational and analysis time. This decision was also based on preliminary results with these joint alignment models (on the cyclone dataset) that did not demonstrate enough of a gain in performance to justify their systematic use.

All that remains is to compare the `norm`-based PRM_AT with the best competing `znorm`-based model. Figure 10.5 compares PRM_AM `znorm` with the two best `norm` models (PRM_AT and PRM_AM). Interestingly, these results show the same exact scenario as that for the ETC data. That is, on the density modelling task, the time-alignment model performs best, but for the curve modelling and prediction task, PRM_AM out-performs the time alignment model. These two models are the only two that show this sort of inverse relationship over the two test scores.

Since the curve prediction task is more important in the cyclone domain, and because for equivalent performance, simpler models should be preferred on average, the less complex space-alignment model PRM_AM is chosen as the best predictive

model on unseen data.

10.4.3 Choosing K

An objective analysis of the type and number of cyclone clusters that exist in the tropical North Pacific has not been done in atmospheric science. Previous studies (e.g., Harr & Elsberry, 1995a, 1995b; Lander, 1996) have found it useful to partition cyclones into a few clusters mostly based on large-scale shape characteristics. In this section, we address the issue of choosing the optimal number of clusters in an objective fashion. Only the chosen alignment model PRM_AM from above is considered in this section.

The reported experiments in this section were carried out in the same manner as in the previous sections. However, these results are based on twice as many runs (i.e., twenty different training and test sets were sampled, with the scores averaged over the twenty runs).

Figure 10.6 shows the plotted values of the test log-likelihood (top) and prediction SSE (bottom) scores for selected values of K. The results do not indicate a clear choice for the best value of K. Both the log-likelihood and SSE scores steadily improve as K increases up until the K values of 10 or 11. At this point, the incremental improvement begins to show random fluctuations, sometimes negative, other times positive. K values as large as 30 exhibit similar behavior.

A close-up view of the SSE score curve is provided in Figure 10.7. At the point $K = 10$, the incremental improvement has been reduced to just 3% of the initial improvement from $K = 1$ to $K = 2$. Beyond this K value, the SSE curve does not show any steady improvement. Based on these observations, 10 is chosen as that value of K which leads to the best predictive modelling performance with minimum model complexity. Thus, the model selection is complete with the choice of alignment

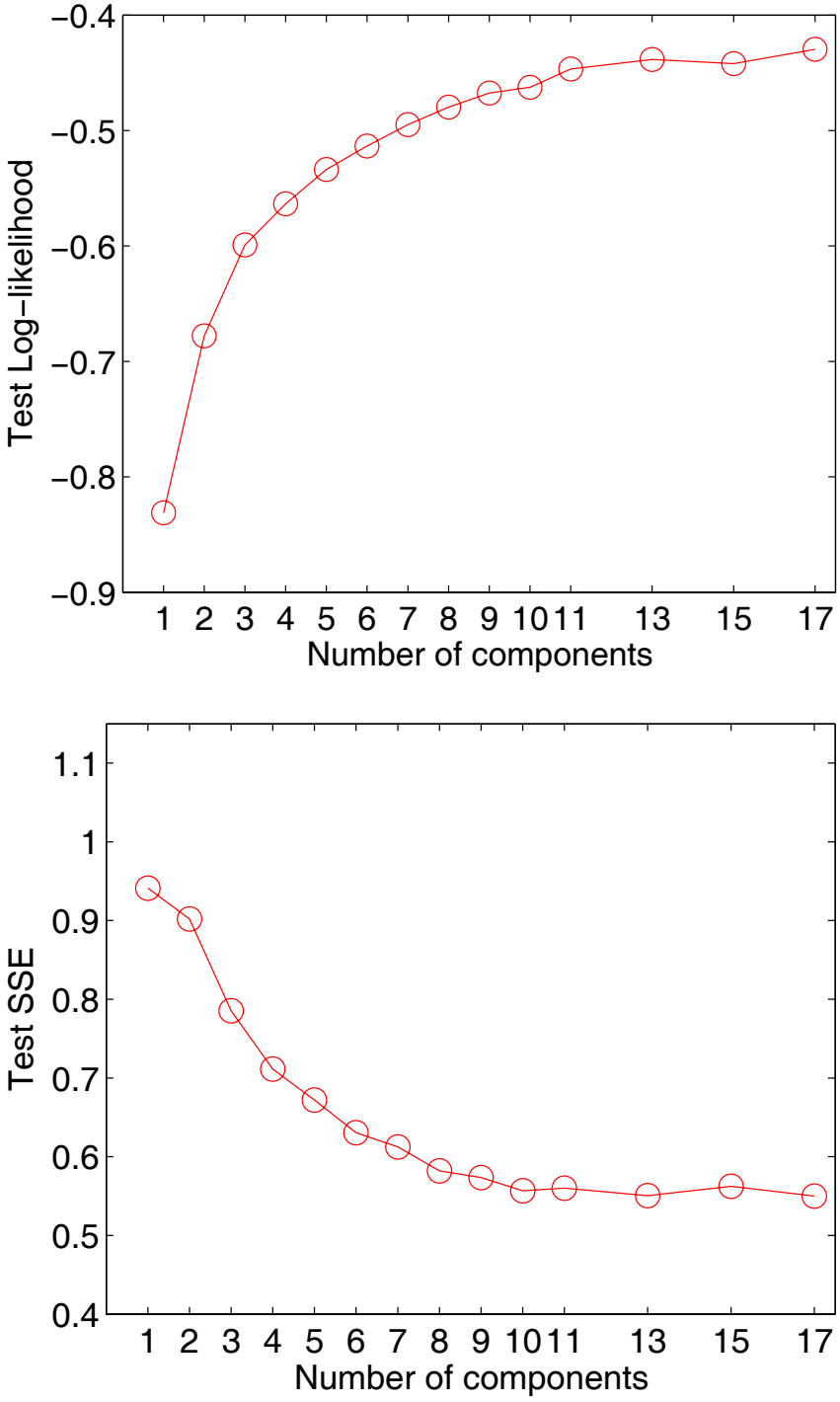


Figure 10.6: Test log-likelihood (top) and prediction SSE (bottom) scores for PRM_AM applied to `znorm` cyclone data for various values of K. The value of $K = 10$ was chosen as the best compromise selection.

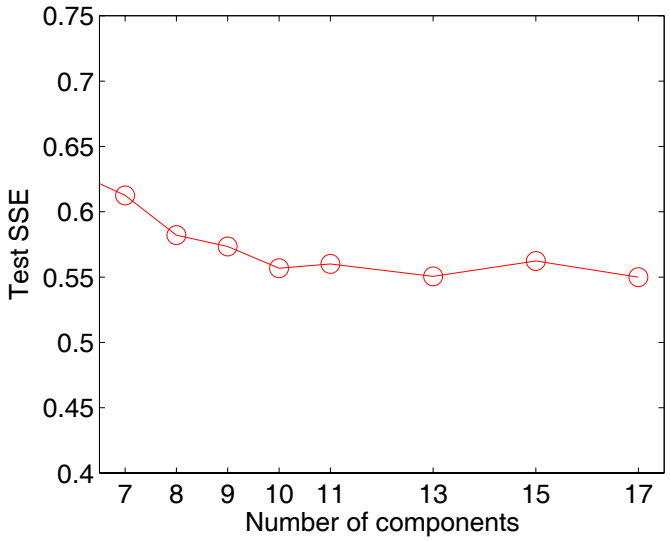


Figure 10.7: Close-up view of the prediction SSE score curve for PRM_AM applied to `znorm` cyclone data for various values of K.

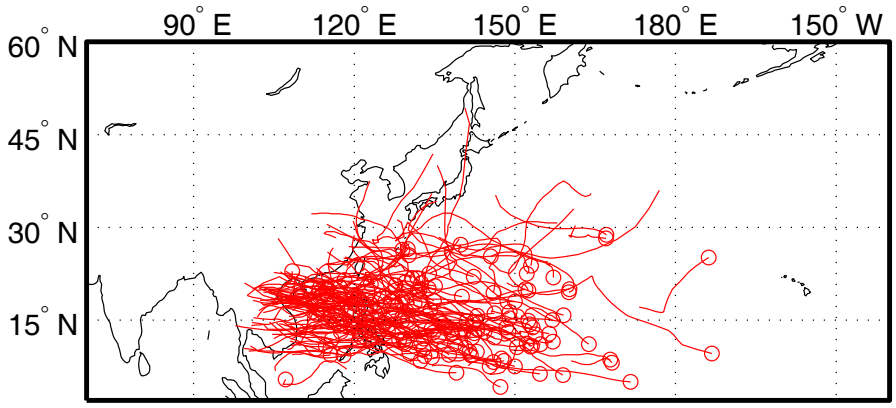
model (PRM_AM), preprocessing method (`znorm`), and the number of clusters (10).

10.5 Clustering analysis

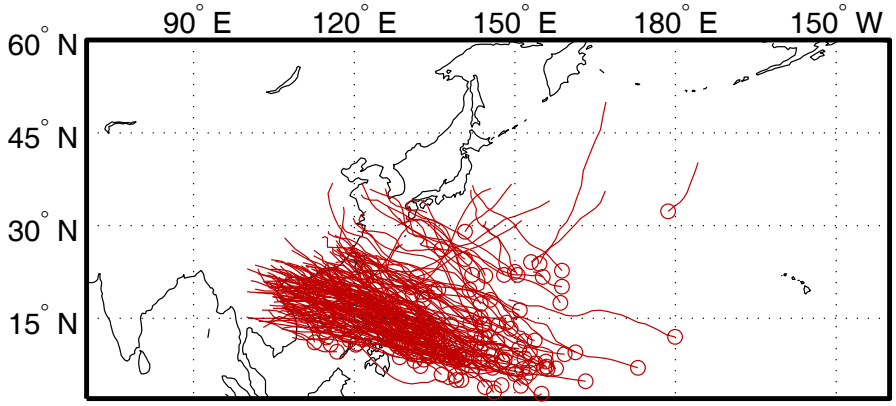
In this section, we analyze the resulting tropical cyclone clusters from the application of PRM_AM to the `znorm` cyclone data. The cluster track-types are analyzed and various cluster-specific statistics are discussed. The temporal behavior of the cyclone clusters was investigated and is reported at the end of this section.

Figures 10.8 to 10.11 geographically depict the cyclones from each of the resulting ten clusters. The clusters are organized into five main groups based on the general track-type of each cluster's cyclones: (a) straight-path, (b) north recurving, (c) east recurving, (d) vertically-oriented, and (d) transient.

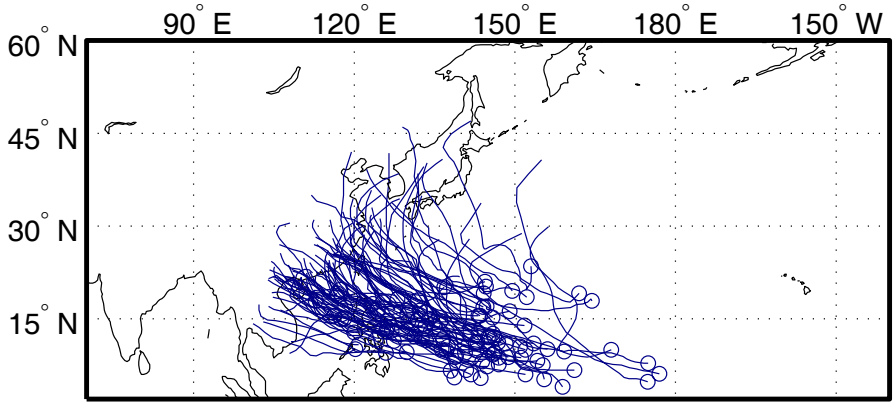
Useful names are given (in `typewriter` font) to each cluster for referential purposes. The names are briefly explained. In Figure 10.8, the three straight-path clusters are shown. The `HorzStraight` cluster consists of cyclones that move along



(a) `HorzStraight` (HS)

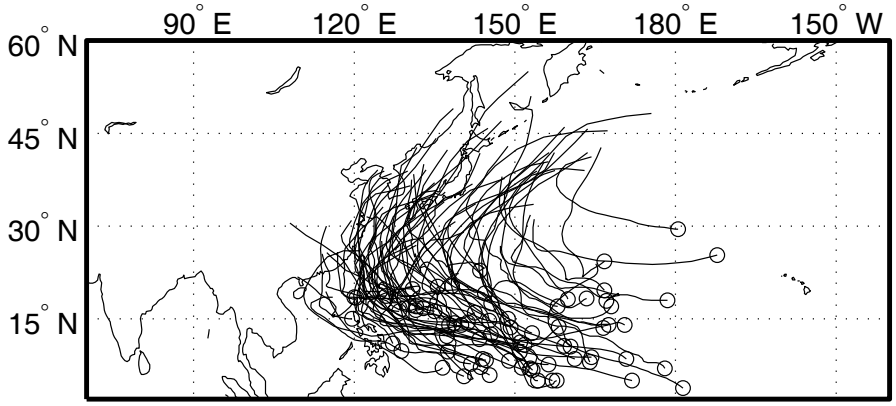


(b) `DiagStraight` (DS)

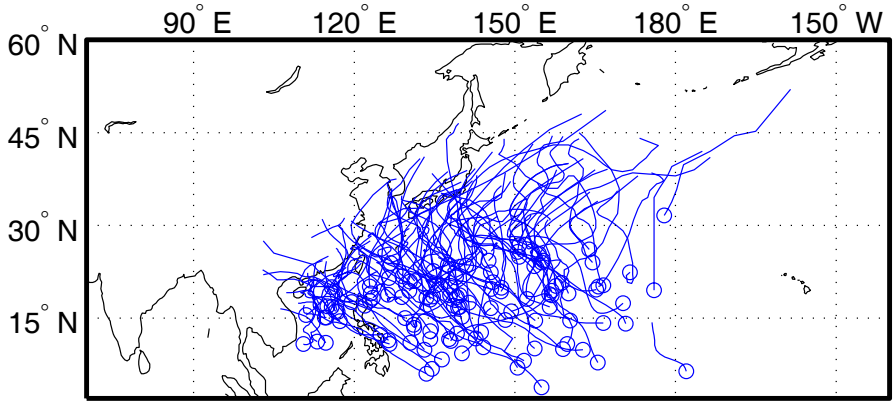


(c) `DiagStraightTail` (DT)

Figure 10.8: Straight-path clusters.



(a) `HorzCurveNorth` (HN)



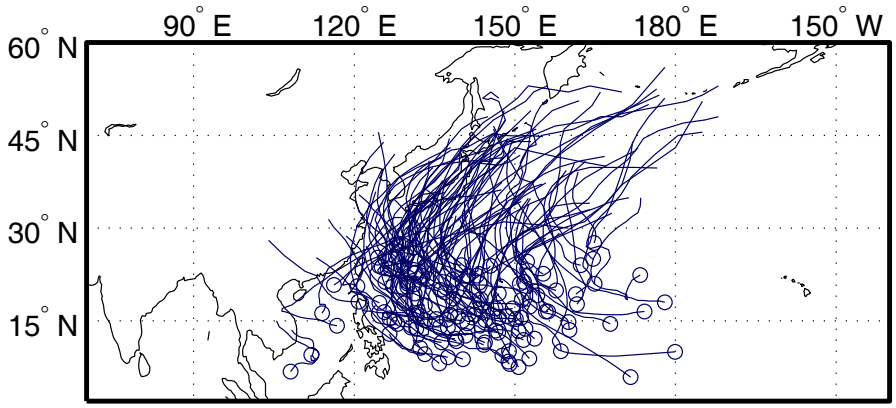
(b) `DiagCurveTail` (DCT)

Figure 10.9: North recurving clusters

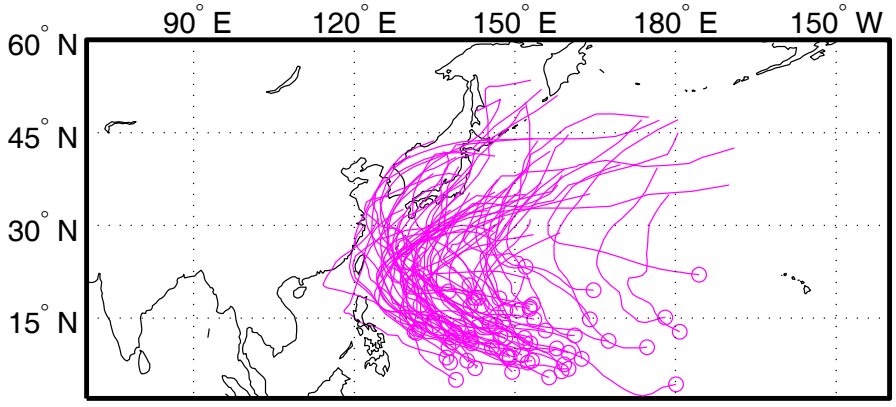
a due west track, mostly making landfall (if at all) south of Hong Kong. The `DiagStraight` cluster consists of cyclones that move along a straight path in a west-northwest direction. The third straight-path cluster `DiagStraightTail` consists of cyclones that move along a similar west-northwest track but tail off to the north.

Figure 10.9 shows the two north recurving clusters. The `HorzCurveNorth` cluster consists of cyclones that initially begin moving along a due west track and then curve mostly to the north. The `DiagCurveTail` cluster primarily consists of cyclones that begin moving along a northwest track and then curve to the north.

The two east recurving clusters are shown in Figure 10.10. The `DiagCurveEast`



(a) `DiagCurveEast` (DCE)

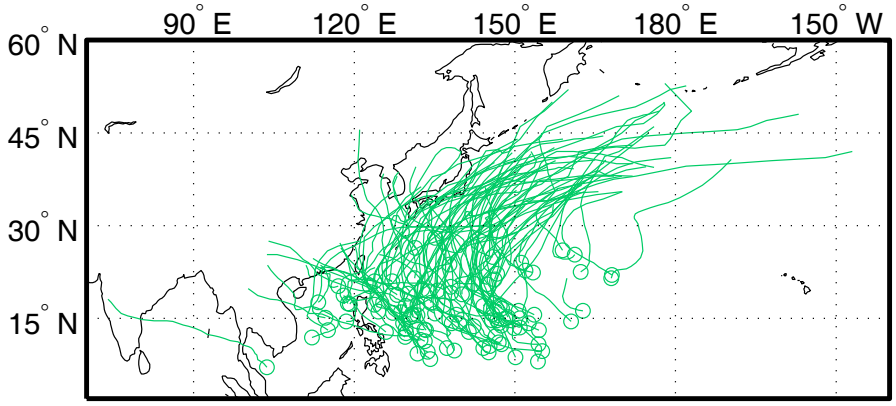


(b) `DiagBendEast` (DBE)

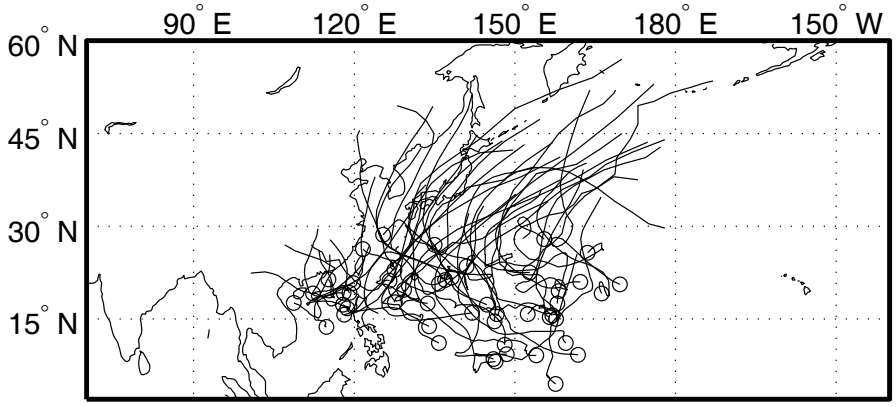
Figure 10.10: East recurving clusters.

cluster consists of cyclones that begin moving along a north-northwest track and then turn to the northeast. The `DiagBendEast` cluster consists of cyclones that follow a west-northwest track that completely *bends* around without exceeding 45°N.

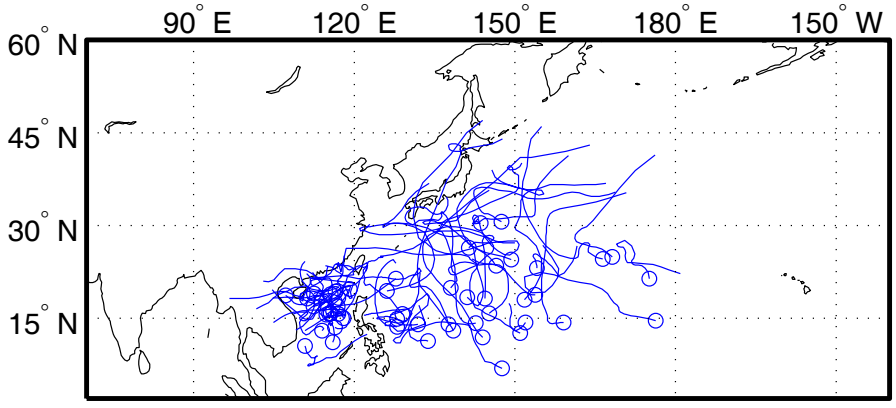
The two vertical-path clusters are shown in Figure 10.11, along with the lone transient cluster. The cyclones of the `VertCurveEast` cluster primarily move along an initial north track and then make a hard turn to the east. The cyclones of the `ReverseTrough` cluster start-out in many different initial directions but then primarily resemble a track-type associated with a reverse-oriented (southwest-to-northeast) monsoon trough in the North Pacific (Lander, 1996). Note that the



(a) `VertCurveEast` (VE)



(b) `ReverseTrough` (RT)



(c) `SouthChinaSea` (SCS)

Figure 10.11: The two vertical-path and the single transient cluster.

cyclones of `VertCurveEast` also resemble tracks which seem to have been influenced by this effect, which is why they are grouped here together (this is discussed in detail below).

The final cluster `SouthChinaSea` largely consists of highly variable tracks not associated with other track-types. This cluster contains many of the random track-types commonly associated with the cyclones of the South China Sea.

10.5.1 Cluster descriptions

In this section, a detailed analysis of the above clustering is given. Cyclone characteristics such as velocity, duration, frequency, and genesis region are used to draw distinctions between the clusters. Several tables and figures are initially introduced that are used as reference for the ensuing cluster analysis.

Table 10.2 lists a number of empirically-derived cyclone characteristics for each of the ten tropical cyclone clusters. The values under the columns give cluster-wide means and deviations of the summary statistics. For example, μ for column `HorzStraight` in Table 10.2 reports the mean of all the minimum intensities attained by the cyclones in cluster `HorzStraight`. The standard deviation for this value is shown under σ for the same column. Bolded entries denote the largest value among all of the clusters and underlined entries denote the smallest. Explicit definitions for the acceleration, curvature, and instability measures are given in Section 9.8.1.

Figure 10.12 shows the distributions for cyclone duration found in each cluster. The plotted values at each point of the x-axis report the number of cyclones in each cluster with a duration greater than or equal to the chosen x point. The curves generally describe the lifetime decay rate for the cyclones in each cluster.

There appears to be three distinct decay rates: the `SouthChinaSea` and the `ReverseTrough` clusters show a below average rate, `DiagStraight` registers a larger

Table 10.2: Cluster-wide averaged measures of various cyclone-specific statistics for the ten clusters. Both means (μ) and standard deviations (σ) are given for each cluster column. Bolded entries give the largest value among all ten clusters and underlined entries give the smallest. The units are given as follows: velocity (km/h), absolute acceleration (km/h^2), duration (days), curvature (radians/km$\times 10^{-3}$), and instability (radians/km$\times 10^{-3}$).

Cyclone statistics	HorzStraight		DiagStraight		DiagStraightTail	
	μ	σ	μ	σ	μ	σ
Velocity	19.28	6.14	21.87	5.12	20.56	4.69
Acceleration	3.69	2.23	3.60	1.76	<u>3.29</u>	1.65
Duration	6.76	2.93	6.40	2.42	7.32	2.50
Curvature	2.20	3.00	<u>0.90</u>	0.74	1.02	0.81
Instability	3.97	6.90	<u>1.20</u>	1.88	1.50	2.25

Cyclone statistics	HorzCurveNorth		DiagCurveTail		DiagCurveEast		DiagBendEast	
	μ	σ	μ	σ	μ	σ	μ	σ
Velocity	21.76	5.98	21.36	9.19	25.85	8.05	**25.91**	5.95
Acceleration	3.86	1.84	4.61	4.10	5.08	2.34	4.99	2.03
Duration	**9.11**	2.21	<u>6.29</u>	2.69	6.46	2.16	8.57	1.63
Curvature	1.77	2.07	2.09	1.85	1.77	1.57	1.34	2.01
Instability	3.36	5.24	3.63	4.64	3.12	3.74	2.54	7.12

Cyclone statistics	VertCurveEast		ReverseTrough		SouthChinaSea	
	μ	σ	μ	σ	μ	σ
Velocity	25.75	10.06	21.90	6.68	<u>17.25</u>	6.10
Acceleration	**5.49**	2.99	4.55	1.76	4.02	2.91
Duration	6.76	2.28	8.92	2.26	7.54	2.41
Curvature	1.74	2.71	4.91	3.45	**5.06**	3.62
Instability	3.05	6.50	**11.43**	8.94	10.32	9.28

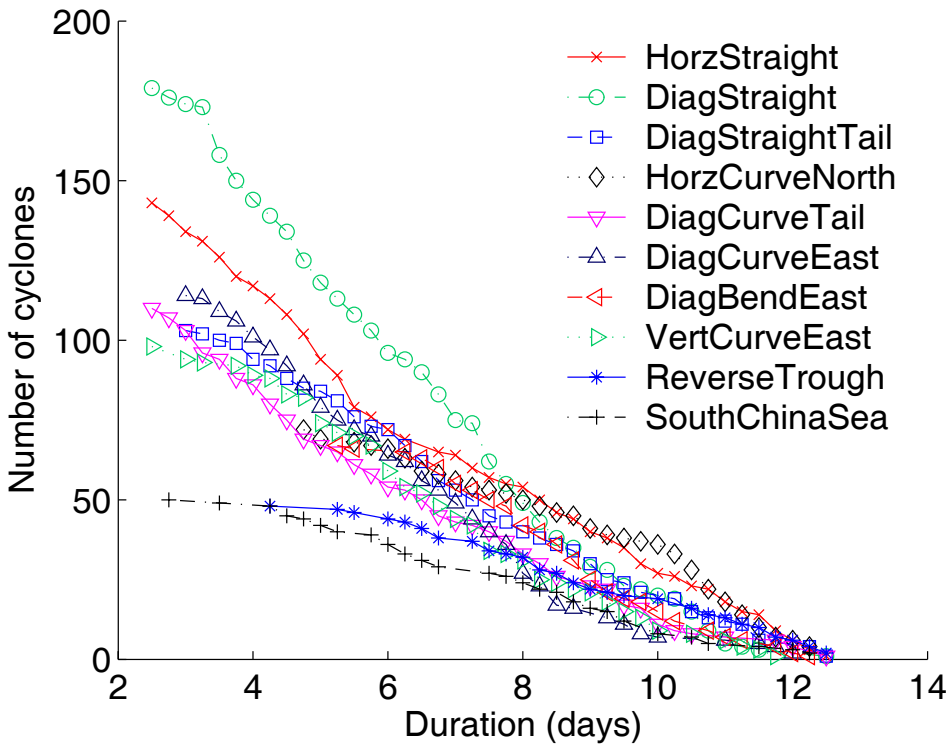


Figure 10.12: Number of cyclones in each cluster as a function of cyclone duration. The graph shows the lifetime decay rate for each cluster.

than average decay rate, while the other clusters share a median lifetime decay rate. Interestingly, the cyclones of the tropical North Pacific appear to have linear decay rates. This is in contrast to the decay rates from the ETC clusters shown earlier in Figure 9.25 that clearly demonstrate an exponential decay.

Below, the ten clusters are discussed in detail using the cluster maps, the table of data, and the figure as references.

`HorzStraight`

The cyclones in this cluster loosely follow a straight, due west track. The tracks exhibit large curvature and instability. In fact, only the two most variable clusters (`ReverseTrough`, and `SouthChinaSea`) are able to match the instability of these

cyclones. All three of the straight-path clusters have low mean acceleration, though these cyclones seem to demonstrate the largest acceleration of the three.

This is the second largest cluster with 143 cyclones. The genesis region is more northwest and spread out than the other straight-path clusters, with many genesis events located in the South China Sea (only the `SouthChinaSea` cluster itself is comparable). The mean genesis point is 15°N by 137°E.

`DiagStraight`

This is the largest cluster with 179 cyclones occurring over the 52 years under consideration. The cyclones primarily follow straight west-northwest tracks. This cluster has an average curvature that is nearly zero (0.0009), the lowest overall, and is the most stable.

The associated lifetime decay rate is the largest among all clusters leading to the second-smallest average cyclone duration. The genesis region is the most southern overall, with almost all events occurring below 15°N. The mean genesis point is 12°N by 138°E.

`DiagStraightTail`

This straight-path cluster consists of cyclones that follow a similar track to those of `DiagStraight`, but tail off to the North. The associated lifetime decay rate is much smaller, however, than that of `DiagStraight`. This results in an increased average duration. This cluster is of average size with 103 total cyclones and rivals `DiagStraight` as the second-most stable cluster.

The genesis region is located a bit more northeast than that of `DiagStraight`, but it is still well below 15°N. The mean genesis point is 13°N by 140°E.

`HorzCurveNorth`

These cyclones initially begin moving along a due west track and then curve to the north; some tracks extend all the way to the northeast. These tracks undergo most of their curvature almost exactly at 23°N latitude, with only slight curvature occurring beyond.

This cluster is of below average size with 72 total cyclones, yet it contains the longest duration cyclones overall. As such, the lifetime decay rate is below average. The genesis region is the most eastern of all the clusters, most occurring in the Southeast portion of the overall genesis region. The mean genesis point is 13°N by 150°E.

`DiagCurveTail`

The tracks of this cluster are mostly oriented to the northwest, many times tailing off to the north. These cyclones are the shortest-duration overall with a mean lifetime of 6.29 days. Many of these tracks exhibit an S-track type motion which has been shown to be associated with a reverse monsoon trough (Lander, 1996). However, the tracks more resemble what you would expect under a normal southeast-to-northwest circulation as seen by the primarily northwest track orientation. This cluster is probably best explained as having occurred from a mix (or a transition) between the two circulation patterns.

The genesis region is quite diffuse and is the most northern of all the clusters except for that of the `ReverseTrough` cluster itself. The mean genesis point is 17°N by 141°E.

`DiagCurveEast`

The cyclones of this cluster follow northwest or north-northwest tracks that curve to the northeast. A distinguishing feature of this cluster is that the cyclones undergo nearly constant curvature throughout their lifetime (unlike the related `DiagBendEast` cluster discussed below). These cyclones are the second-fastest cyclones overall with an average speed of 25.9 km/h, and undergo the second-largest acceleration on average.

This cluster is larger than average with 114 total cyclones. The genesis region is nearly identical to that of `DiagCurveTail`, only more compact. The mean genesis point is 17°N by 142°E.

`DiagBendEast`

The tracks of this cluster initially point in a west-northwest direction and then quickly bend around and head in the opposite direction. These cyclones undergo rapid curvature over a short period of time near 20°N latitude. These cyclones are the fastest overall, undergo the second-largest acceleration, and have the third longest average duration.

This cluster is below average in size with only 67 total cyclones. The genesis region is a small compact subregion located to the southeast. The mean genesis point is 13°N by 149°E.

`VertCurveEast`

This is an average size cluster with 98 total cyclones. The cyclones of this cluster primarily follow north-oriented tracks that curve due east; however, many follow tracks that start-out in an already eastward heading to begin with. As pointed-out earlier, these types of tracks are associated with a reverse monsoon trough. In

contrast to the cyclones of `ReverseTrough`, the tracks here are much smaller, less random, and considerable faster. Furthermore, the genesis region is significantly more compact and localized in the southern region. The mean genesis point is 16°N by 139°E.

`ReverseTrough`

The cyclones of this cluster are highly indicative of a reverse-oriented monsoon trough. Track-types associated with this event are unusual since they do not follow the normal climatological tracks (Lander, 1996). Track-types tend to be north- or even north-eastward-oriented and often exhibit random behavior. Lander demonstrated that many cyclones associated with a reverse-oriented monsoon trough follow distinctive S-shaped tracks.

Many of these attributes can be associated with the tracks in this cluster. For example, this cluster has the least stable tracks (11.43) and the second largest curvature (4.91), which demonstrates highly random behavior. Furthermore, many of the cyclone tracks in this cluster move in a completely north-eastward heading (rather unusual under normal circumstances).

This cluster is the smallest in total size with only 48 cyclones, which also gives support for its association with an unusual event. The genesis region is very diffuse, only matched by the noisy `SouthChinaSea` cluster. The mean genesis point is centrally located at 18°N by 140°E.

`SouthChinaSea`

This cluster naturally groups together the well-known South China Sea cyclones (Harr & Elsberry, 1995a) with other highly variable cyclones that do not fit within any other cluster. The South China Sea cyclones tend to form and remain within the sea,

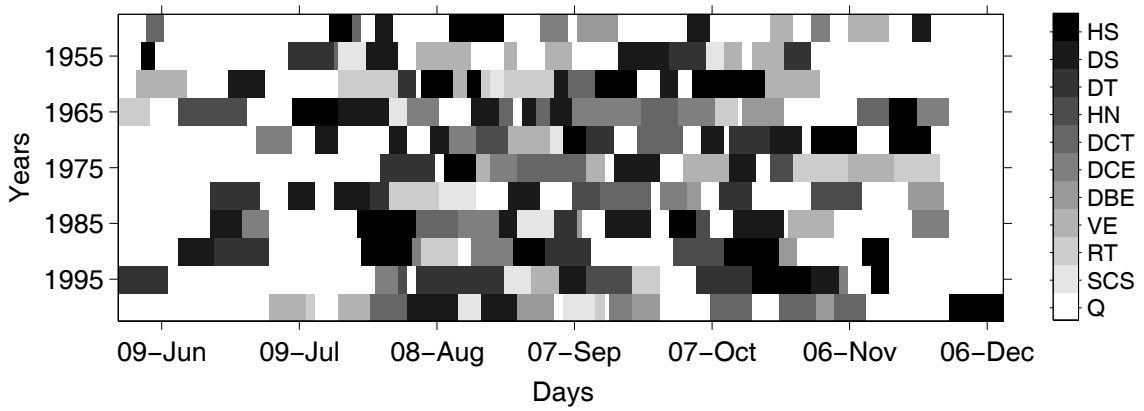


Figure 10.13: Daily regime classification for every five years (top-to-bottom) from 1950–2000. Each of the eleven rows depicts the regime classification for each day in the associated year. The key at the right gives the gray-value-to-regime mapping where the two- or three-letter abbreviations denote the names of the cluster regimes as defined in Figures 10.8–10.11. Q represents the quiescent regime.

mostly meandering in random directions (the South China Sea is located between the Philippines and Vietnam or China in the bottom-left of the map in Figure 10.11(c)).

The cyclones of this cluster have the slowest velocity, the largest curvature, and are the second-most unstable. The cluster is the second-smallest with only 50 total cyclones. The genesis region is concentrated around the South China Sea with various other random genesis events occurring outside this region. The mean genesis point is 17°N by 134°E.

10.5.2 Temporal analysis of cyclone clusters

In this section we analyze the daily temporal behavior of the tropical cyclone clusters described above. We classify each day as being in one of eleven cluster regimes corresponding to the ten clusters, or in the eleventh case, to none of the clusters. The same heuristic procedure is used to perform the daily classifications as that in Section 9.8.2.

Applied to the clustered cyclone data, the assignment procedure yields the daily

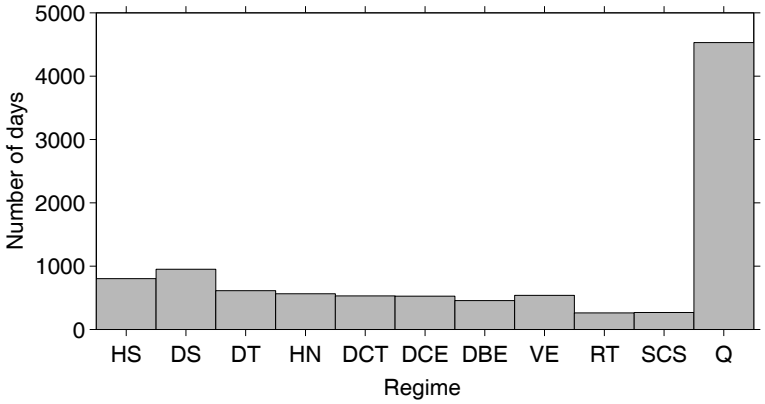


Figure 10.14: Histogram of regime activity corresponding to the regime classification in Figure 10.13.

regime sequences shown in Figure 10.13. The plot in this figure has eleven rows corresponding to every fifth year from 1950 to 2000. The rows show the regime classification for each day by color-coded pixels. The key at the right gives the gray-value-to-regime mapping where the two- or three-letter abbreviations denote the names of the cluster regimes as defined in Figures 10.8–10.11. Q represents the quiescent regime, or the regime not associated with any of the ten cyclone clusters.

Some edge effects are to be expected since only cyclones that began on or after 30 May are included in this dataset. Thus, you would expect to see a larger proportion of white-space at the left of the picture (a similar situation holds for the right side as well). However, regardless of the edge effects, it is clear that almost all cyclone generations happen between 1 August and 1 November.

The distribution of regime activity is plotted in Figure 10.14. This plot gives the number of days that each regime is active over all 52 years from 1950–2001. For the most part, the largest clusters generate the most active regimes. However, this is not so in every case. For example, `DiagCurveEast` (DCE) is the third largest cluster and yet it shows less regime activity than six other clusters.

An interesting feature of Figure 10.13 is the distribution of regime run-lengths.

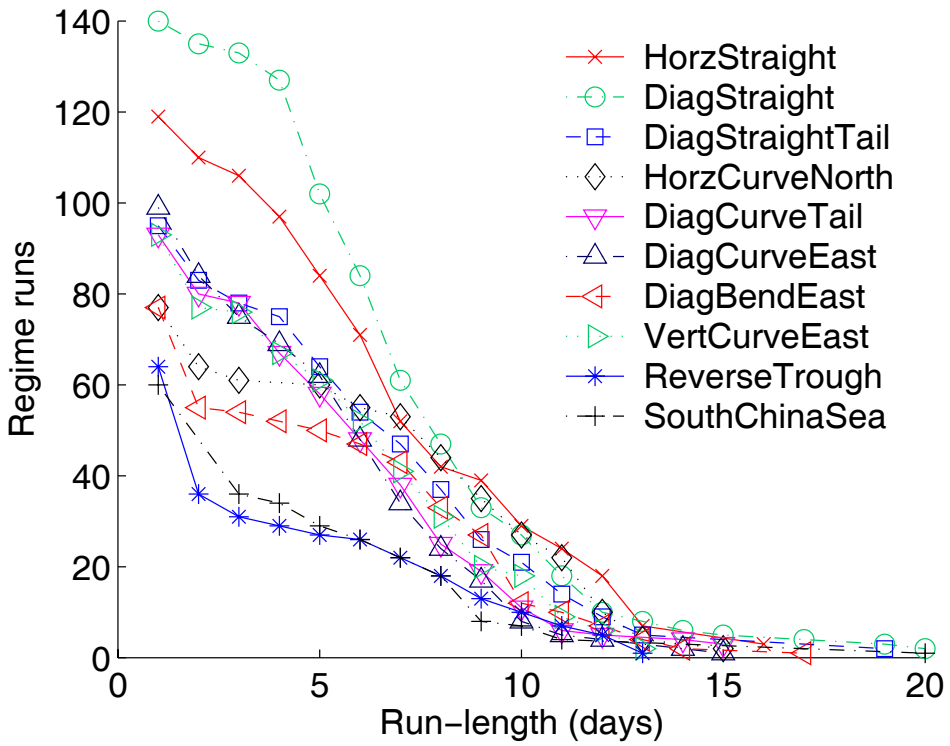


Figure 10.15: Distribution of run-lengths for each regime corresponding to the regime classifications in Figure 10.13.

Figure 10.15 shows the distribution of run-lengths for each regime. The curves show the decay rate for each regime.

The cluster regimes show definite persistence with run-length distributions that are non-geometric in nature. For example, `DiagBendEast`, `HorzCurveNorth`, and `ReverseTrough` show uniform persistence from 3 to 6 plus days. Other regimes show linear persistence, for example, `DiagStraightTail`. It appears that regime persistence in the tropical North Pacific is decidedly non-geometric, at least until large values of run-lengths are reached. This suggests a non-first order Markov model might best describe the temporal regime-like behavior of the tropical North Pacific.

10.6 Summary

In this chapter, an application of our joint alignment-clustering methodology to observed North Pacific tropical cyclones was presented. Unlike the GCM derived ETC tracks of the previous chapter, the JTWC dataset analyzed in this chapter consists of "real-world" cyclone track observations.

This chapter focused on an objective analysis of the type and number of cyclone clusters that exist in the tropical North Pacific. All prior cyclone clustering work with tropical cyclones simply grouped cyclones based on whether or not they followed a straight or curved path over a particular area of interest. This grouping may be fine for many types of analyses; however, an objective out-of-sample analysis of the type and number of clusters in the tropical North Pacific has never been pursued.

Probabilistic model selection procedures based on out-of-sample log-likelihood and prediction SSE scores were used to select the best predictive modelling methodology. This selection process consisted of choosing a specific order cyclone regression model, a trajectory preprocessing method, a joint clustering-alignment model, and the number of cyclone clusters that best describes the dataset. The model selection process applied to the JTWC cyclone dataset resulted in the following methodology:

- Order of polynomial regression model: quadratic
- Trajectory preprocessing: `znorm`
- Clustering-alignment model: PRM_AM
- Number of clusters: 10

The resulting ten clusters were analyzed in detail and shown to group cyclones based on identifiable characteristics such as speed, acceleration, duration, and track-type. The clusters were also shown to correspond to known states of circulation in the atmosphere such as that associated with a reverse-oriented monsoon trough.

Daily regime classifications were assigned to each day based on the number of active cyclones during each 24-hour period. The regime classifications were plotted and summary statistics describing the distributions of regimes and their run-lengths were reported. The regimes were shown to exhibit ranges of uniform and linear persistence over time. This suggests a non-first order Markov assumption is necessary for describing the temporal regime-like behavior in the tropical North Pacific.

Chapter 11

Conclusion

This dissertation was concerned with the central hypothesis that clustering and alignment should not be carried out in isolation since significant relationships exist between the two problems that can be leveraged in a joint formulation. We introduced a novel methodology for the clustering and prediction of sets of smoothly varying curves while jointly allowing for the learning of sets of continuous curve transformations.

The methodology was two-fold. First, we introduced new probabilistic alignment models that employed curve modelling techniques and defined priors over the sets of allowable curve transformations. This self-contained formulation of the alignment problem resulted in iterative EM alignment algorithms that resembled generalized Procrustes-type alignment procedures (Mardia et al., 1979). Experiments with a "real-world" gene expression dataset showed the effectiveness of the probabilistic formulation.

Second, we integrated these new alignment models into a model-based curve clustering framework based on mixtures of regression models. The integration was natural since the definition of the alignment models were also founded in (curve) model-

based techniques. The resulting clustering algorithms were naturally transformation-invariant without the need for additional specialized procedures or constraints.

Experiments with simulated data showed that the joint methodology was superior to the isolated approach. Further experiments with simulated data showed that the curve-based modelling techniques employed in the clustering-alignment models effectively handled common problems encountered with curve datasets: (a) incorporation of variable-length curves, (b) correct accounting of irregular sampled/observed curve measurements, (c) the handling of randomly missing curve observations, and (d) the leveraging of inherent smoothness information available in curves.

Two extensive applications of the joint clustering-alignment methodology were reported. The applications concerned the clustering of cyclone trajectories tracked over the North Atlantic and the North Pacific. A detailed model selection was presented for each application that resulted in the selection of an optimal cyclone regression model, trajectory preprocessing procedure, clustering-alignment model, and an optimal number of clusters. The resulting optimal clustering in each case was presented and analyzed in detail. The applications demonstrated the practical use of the ideas introduced in this dissertation.

References

Aach, J., & Church, G. M. (2001). Aligning gene expression time series with time warping algorithms. *Bioinformatics*, *17*(6), 495–508.

Anderson, D., Hodges, K. I., & Hoskins, B. J. (2003). Sensitivity of feature-based analysis methods of storm tracks to the form of background field removal. *Monthly Weather Review*, *131*(3), 565–573.

Banfield, J. D., & Raftery, A. E. (1993). Model-based Gaussian and non-Gaussian clustering. *Biometrics*, *49*, 803–821.

Ben-Dor, A., Shamir, R., & Yakhini, Z. (1999). Clustering gene expression patterns. *Journal of Computational Biology*, *6*, 281–297.

Blender, R., Fraedrich, K., & Lunkeit, F. (1997). Identification of cyclone-track regimes in the North Atlantic. *Quart J. Royal Meteor. Soc.*, *123*, 727–741.

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees.* Monterey, CA: Wadsworth and Brooks/Cole.

Buntine, W. (1994). Operations for learning with graphical models. *Journal of the Artificial Intelligence Research*, *2*, 159–225.

Burman, P. (1989). A comparitive study of ordinary cross-validation, v-fold cross-validation, and the repeated learning-testing methods. *Biometrika*, *76*(3), 503–514.

Butte, A., & Kohane, I. (2000). Mutual information relevance networks. *Pacific Symposium on Biocomputing*, *5*, 415–426.

Cadez, I., & Smyth, P. (1999). *Probabilistic clustering using hierarchical models* (Tech. Rep. No. TR-99-16). Department of Information and Computer Science, University of California, Irvine.

Chudova, D., Gaffney, S. J., , & Smyth, P. J. (2003). Probabilistic models for joint clustering and time-warping of multi-dimensional curves. In *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence (UAI-2003), Acapulco, Mexico, August 7–10.*

Chudova, D., Gaffney, S. J., Mjolsness, E., & Smyth, P. J. (2003). Translation-invariant mixture models for curve clustering. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington D.C., August 24–27.* New York: ACM Press.

Chui, H., Zhang, J., & Rangarajan, A. (2004). Unsupervised learning of an atlas from unlabeled point-sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*(2), 160–172.

Cootes, T. F., Hill, A., Taylor, C. J., & Haslam, J. (1994). Use of active shape models for locating structures in medical images. *Image and Vision Computing*, *12*(6), 355–366.

Cross, A. D. J., & Hancock, E. R. (1998). Graph matching with a dual-step EM algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *20*(11), 1236–1253.

de Boor, C. (1978). *A practical guide to splines.* New York, NY: Springer-Verlag.

de Boor, C. (1986). B(asic)-spline basics. In C. de Boor (Ed.), *Extension of b-sline curve algorithms to surfaces* (Vols. Course #5, ACM SIGGRAPH 86, pp. 18–22). Dallas, TX: ACM.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, *39*, 1–38.

DeSarbo, W. S., & Cron, W. L. (1988). A maximum likelihood methodology for clusterwise linear regression. *Journal of Classification*, *5*(1), 249–282.

Donoho, D. L., Johnstone, I. M., Kerkyacharian, G., & Picard, D. (1996). *Density estimation by wavelet thresholding* (Tech. Rep. No. 426). Stanford, CA: Department of Statistics, Stanford University.

Dougherty, E. R., Barrera, J., Brun, M., Kim, S., Cesar, R. M., Chen, Y., Bittner, M., & Trent, J. M. (2002). Inference from clustering with application to gene-expression microarrays. *Journal of Computational Biology*, *9*(1), 105–126.

Draper, N. R., & Smith, H. (1981). *Applied regression analysis* (2nd ed.). New York, NY: John Wiley and Sons.

Dryden, I. L., & Mardia, K. V. (1998). *Statistical shape analysis.* New York: John Wiley & Sons.

Duda, R. O., & Hart, P. E. (1973). *Pattern classification and scene analysis* (1st ed.). New York, NY: John Wiley and Sons.

Duta, N., Jain, A. K., & Dubuisson-Jolly, M.-P. (1999). Learning 2D shape models. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2008–2012). Fort Collins, CO: IEEE.

Eilers, P. H. C., & Marx, B. D. (1996). Flexible smoothing with B-splines and penalties. *Statistical Science*, *11*(2), 89–121.

Eisen, M. B., Spellman, P. T., Brown, P. O., & Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Science*, *95*(25), 14863–68.

Eubank, R. L. (1988). *Spline smoothing and nonparametric regression.* New York, NY: Dekker.

Everitt, B. S. (1993). *Cluster analysis* (3rd ed.). London: Gower Publications.

Everitt, B. S., & Hand, D. J. (1981). *Finite mixture distributions.* London: Chapman and Hall.

Fan, J., & Gijbels, I. (1996). *Local polynomial modelling and its applications.* London: Chapman and Hall.

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

Fraley, C., & Raftery, A. E. (2002). Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, *97*(458), 611–631.

Frey, B. J., & Jojic, N. (1999). Estimating mixture models of images and inferring spatial transformations using the EM algorithm. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition* (pp. 416–422). IEEE.

Frey, B. J., & Jojic, N. (2002). Fast, large-scale transformation-invariant clustering. In T. G. Dietterich, S. Becker, & Z. Ghahramani (Eds.), *Advances in Neural Information Processing Systems 14.* Cambridge, MA: MIT Press.

Frey, B. J., & Jojic, N. (2003). Transformation-invariant clustering using the EM algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *25*(1), 1–17.

Friedman, J., & Silverman, B. W. (1989). Flexible parsimonious smoothing and additive modeling. *Technometrics*, *31*, 3–39.

Gaffney, S., & Smyth, P. (1999). Trajectory clustering with mixtures of regression models. In S. Chaudhuri & D. Madigan (Eds.), *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, August 15–18* (pp. 63–72). N.Y.: ACM Press.

Gaffney, S. J., Robertson, A., & Smyth, P. (2001). Clustering of extra-tropical cyclone trajectories using mixtures of regression models. In *Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Fourth Workshop on Mining Scientific Datasets.*

Gaffney, S. J., & Smyth, P. (2003). Curve clustering with random effects regression mixtures. In C. M. Bishop & B. J. Frey (Eds.), *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics.* Key West, FL.

Gasser, T., & Kneip, A. (1995). Searching for structure in curve samples. *Journal of the American Statistical Association*, *90*(432), 1179–1188.

Gates, W. L. (1992). *The validation of atmospheric models* (PCMDI Report No. 1). Livermore, CA: Lawrence Livermore National Laboratory.

Gelfand, A. E., & Smith, A. F. M. (1990). Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association*, *85*, 398–409.

Gelman, A., Carlin, J. B., Stern, H. S., & Rubin, D. B. (1995). *Bayesian data analysis.* New York, NY: Chapman & Hall.

Gentle, J. E. (1998). *Random number generation and Monte Carlo methods.* New York, NY: Spring-Verlag.

Gold, S., Rangarajan, A., Lu, C.-P., Pappu, S., & Mjolsness, E. (1998). New algorithms for 2D and 3D point matching: pose estimation and correspondence. *Pattern Recognition*, *31*(8), 1019–1031.

Goodall, C. (1991). Procrustes methods in the statistical analysis of shape. *Journal of the Royal Statistical Society B*, *53*(2), 285–339.

Green, P. J., & Silverman, B. W. (1994). *Nonparametric regression and generalized linear models: A roughness penalty approach.* London: Chapman and Hall.

Hack, J. J., Kiehl, J. T., & Hurrell, J. W. (1998). The hydrologic and thermodynamic characteristics of the NCAR CCM3. *Journal of Climate*, *11*(6), 1179–1206.

Härdle, W., & Marron, S. (1990). Semiparametric comparison of regression curves. *Annals of Statistics*, *18*, 63–90.

Harr, P. A., & Elsberry, R. L. (1991). Tropical cyclone track characteristics as a function of large-scale circulation anomalies. *Monthly Weather Review*, *119*(6), 1448–1468.

Harr, P. A., & Elsberry, R. L. (1995a). Large-scale circulation variability over the tropical western north pacific. Part I: Spatial patterns and tropical cyclone characteristics. *Monthly Weather Review*, *123*(5), 1225–1246.

Harr, P. A., & Elsberry, R. L. (1995b). Large-scale circulation variability over the tropical western north pacific. Part II: Persistence and transition characteristics. *Monthly Weather Review*, *123*(5), 1247–1268.

Hartigan, J. A., & Wong, M. A. (1978). Algorithm AS 136: a K-means clustering algorithm. *Applied Statistics*, *28*, 100–108.

Hodanish, S., & Gray, W. M. (1993). An observational analysis of tropical cyclone recurvature. *Monthly Weather Review*, *121*(10), 2665–2689.

Hodges, K. I. (1994). A general method for tracking analysis and its applications to meteorological data. *Monthly Weather Review*, *122*(11), 2573–2586.

Hodges, K. I. (1995). Feature tracking on the unit sphere. *Monthly Weather Review*, *123*(12), 3458–3465.

Hodges, K. I. (1998). Feature-point detection using distance transforms: Application to tracking tropical convective complexes. *Monthly Weather Review*, *126*(3), 785–795.

Hoskins, B. J., & Hodges, K. I. (2002). New perspectives on the northern hemisphere winter storm tracks. *Journal of the Atmospheric Sciences*, *59*(6), 1041–1061.

Hosmer, D. (1974). Maximum likelihood estimates of the parameters of a mixture of two regression lines. *Communications in Statistics*, *3*(10), 995–1006.

Hurn, M., Justel, A., & Robert, C. P. (2003). Estimating mixtures of regressions. *Journal of Computational and Graphical Statistics*, *12*(1), 55-79.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural Computation*, *3*(1), 79–87.

James, G. M., & Hastie, T. J. (2000). *Functional linear discriminant analysis for irregularly sampled curves* (Tech. Rep.). Los Angeles, CA: Marshall School of Business, University of Southern California.

James, G. M., & Sugar, C. A. (2003). Clustering for sparsely sampled functional data. *Journal of the American Statistical Association*, *98*, 397–408.

Jensen, F. V. (1996). *An introduction to Bayesian networks* (1st ed.). New York: Springer-Verlag.

Jepson, A., & Black, M. (1996). *Mixture models for image representation* (PRECARN ARK Project Technical Report No. ARK96-PUB-54). Department of Computer Science, University of Toronto.

Johnston, J. (1984). *Econometric methods* (3rd ed.). New York, NY: McGraw-Hill.

Jojic, N., Petrovic, N., Frey, B. J., & Huang, T. S. (2000). Transformed hidden Markov models: Estimating mixture models of images and inferring spatial transformations in video sequences. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition.* IEEE.

Jones, P. N., & McLachlan, G. J. (1992). Fitting finite mixture models in a regression context. *Australian Journal of Statistics*, *34*(2), 233–240.

Jordan, M. I., & Jacobs, R. A. (1994). Hiearchical mixtures of experts and the em algorithm. *Neural Computation*, *6*, 181–214.

Kamakura, W. A. (1991). Estimating flexible distributions of ideal-points with external analysis of preference. *Psychometrika*, *56*, 419–448.

Kendall, D. G. (1984). Shape-manifolds, Procrustean metrics and complex projective spaces. *Bulletin of the London Mathematical Society*, *16*, 81-121.

Keogh, E. J., & Pazzani, M. J. (1998). An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. In R. Agrawal, P. E. Stolorz, & G. Piatetsky-Shapiro (Eds.), *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining, New York, Aug. 27–31* (pp. 239–243). New York: AAAI Press.

Keogh, E. J., & Pazzani, M. J. (1999). Scaling up dynamic time warping to massive datasets. In J. M. Zytkow & J. Rauch (Eds.), *Proceedings of the Third European Conference on Principles of Data Mining and Knowledge Discovery, Prague, Czech Republic, Sept. 15–18* (pp. 1–11). New York: Springer.

Kneip, A., & Engel, J. (1995). Model estimation in nonlinear regression under shape invariance. *Annals of Statistics*, *23*(2), 551–570.

Kneip, A., & Gasser, T. (1988). Convergence and consistency properties for self-modeling nonlinear regression. *Annals of Statistics*, *16*(1), 82–113.

Kneip, A., & Gasser, T. (1992). Statistical tools to analyze data representing a sample of curves. *Annals of Statistics*, *20*(3), 1266–1305.

Kohonen, T. (1995). *Self Organizing Maps.* New York, NY: Springer-Verlag.

König, W., Sausen, R., & Sielman, F. (1993). Objective identification of cyclones in GCM simulations. *Journal of Climate*, *6*(12), 2217-2231.

Kooperberg, C., & Stone, C. J. (1991). A study of logspline density estimation. *Computational Statistics and Data Analysis*, *12*, 327–347.

Kooperberg, C., & Stone, C. J. (1992). Logspline density estimation for censored data. *Journal of Computational Graphics and Statistics*, *1*, 301–328.

Laird, N. M., & Ware, J. H. (1982). Random effects models for longitudinal data. *Biometrics*, *38*, 963–974.

Lander, M. A. (1996). Specific tropical cyclone track types and unusual tropical motions associated with a reverse-oriented monsoon trough in the western North Pacific. *Weather and Forecasting*, *11*(2), 170–186.

Lange, K. (1999). *Numerical analysis for statisticians.* New York, NY: Springer-Verlag.

Lawton, W. H., Sylvestre, E. A., & Maggio, M. S. (1972). Self modeling nonlinear regression. *Technometrics*, *14*, 513–532.

Lenk, P. J., & DeSarbo, W. S. (2000). Bayesian inference for finite mixtures of generalized linear models with random effects. *Psychometrika*, *65*(1), 93–119.

Lwin, T., & Martin, P. J. (1989). Probits of mixtures. *Biometrics*, *45*, 721–732.

Mardia, K. V., Kent, J. T., & Bibby, J. M. (1979). *Multivariate analysis.* London: Academic Press.

McCullagh, P., & Nelder, J. A. (1983). *Generalized linear models.* New York, NY: Chapman and Hall.

McLachlan, G., & Krishnan, T. (1997). *The EM algorithm and extensions.* New York, NY: John Wiley and Sons.

McLachlan, G. J., & Basford, K. E. (1988). *Mixture models: Inference and applications to clustering.* New York: Marcel Dekker.

Mitchell, T. (1997). *Machine learning.* New York, NY: McGraw-Hill.

Murray, R. J., & Simmonds, I. (1991). A numerical scheme for tracking cyclone centres from digital data Part I: development and operation of the scheme. *Australian Meteorological Magazine*, *39*, 155–166.

Nelder, J. A., & Mead, R. (1965). *Computer Journal*, *7*, 308–313.

Neumann, A., & Lorenz, C. (1998). Stastical shape model-based segmentation of medical images. *Computerized Medical Imaging and Graphics*, *22*, 133–143.

Ormoneit, D., & Tresp, V. (1996). Improved Gaussian mixture density estimates using Bayesian penalty terms and network averaging. In D. Touretzky, M. Mozer, & M. Hasselmo (Eds.), *Advances in Neural Information Processing Systems 9* (pp. 542–548). New York: MIT Press.

O'Sullivan, F. (1986). A statistical perspective on ill-posed inverse problems. *Statistical Science*, *1*, 505–527.

O'Sullivan, F. (1988). Fast computation of fully automated log-density and log-hazard estimators. *SIAM Journal on Scientific and Statistical Computing*, *9*, 363–379.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems* (Revised 2nd ed.). San Francisco: Morgan Kaufmann.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (1992). *Numerical recipies in c* (2nd ed.). New York, NY: Cambridge University Press.

Quandt, R. E. (1972). A new approach to estimating switching regressions. *Journal of the American Statistical Association*, *67*, 306–310.

Quandt, R. E., & Ramsey, J. B. (1978). Estimating mixtures of normal distributions and switching regressions. *Journal of the American Statistical Association*, *73*, 730–738.

Quinlan, J. R. (1993). *C4.5: Programs for machine learning.* San Mateo, CA: Morgan Kaufmann.

Rabiner, L., & Schmidt, C. (1980). Applications of dynamic time warping to connected digit recognition. *IEEE Transaction Acoustic Speech Signal Processing*, *28*, 377–388.

Ramsay, J., & Silverman, B. W. (1997). *Functional data analysis.* New York, NY: Springer-Verlag.

Ramsay, J. O., & Li, X. (1998). Curve registration. *Journal of the Royal Statistical Society B*, *60*, 351–363.

Rangarajan, A., Chui, H., & Bookstein, F. L. (1997). The Softassign Procrustes matching algorithm. *Information Processing in Medical Imaging*, 29–42.

Rønn, B. B. (2001). Nonparametric maximum likelihood estimation for shifted curves. *Journal of the Royal Statistical Society B*, *63*(2), 243–259.

Sakoe, H., & Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustic Speech Signal Processing*, *26*, 43–49.

Schubert, M., Perlwitz, J., Blender, R., Fraedrich, K., & Lunkeit, F. (1998). North Atlantic cyclones in CO_2-induced warm climate simulations: frequency, intensity, and tracks. *Climate Dynamics*, *14*, 827–837.

Shao, J. (1993). Linear model selection by cross-validation. *Journal of the American Statistical Association*, *88*(422), 486–494.

Silverman, B. W. (1986). *Density estimation for statistics and data analysis.* New York, NY: Chapman and Hall.

Silverman, B. W. (1995). Incorporating parametric effects into functional principal components analysis. *57*(4), 673–689.

Simmons, A. J., & Hoskins, B. J. (1978). The life cycles of some nonlinear baroclinic waves. *Journal of the Atmospheric Sciences*, *35*(3), 414–432.

Smyth, P. (2000). Model selection for probabilistic clustering using cross-validated likelihood. *Statistics and Computing*, *10*(1), 63–72.

Smyth, P., Ide, K., & Ghil, M. (1999). Multiple regimes in northern hemisphere height fields via mixture model clustering. *Journal of the Atmospheric Sciences*, *56*(21), 3704–3723.

Späth, H. (1979). Algorithm 39: clusterwise linear regression. *Computing*, *2*, 367–373.

Spellman, P. T., Sherlock, G., Zhang, M. Q., Iyer, V. R., Anders, K., Eisen, M. B., Brown, P. O., Botstein, D., & Futcher, B. (1998). Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. *Molecular Biology of the Cell*, *9*(12), 3273–3297.

Staib, L. H., & Duncan, J. S. (1992). Boundary finding with parametrically deformable models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *14*(11), 1061–1075.

Tanner, M. A. (1996). *Tools for statistical inference* (Third ed.). New York, NY: Springer-Verlag.

Titterington, D. M., Smith, A. F. M., & Makov, U. E. (1985). *Statistical analysis of finite mixture distributions.* New York, NY: John Wiley and Sons.

Viele, K., & Tong, B. (2002). Modeling with mixtures of linear regressions. *Annals of Statistics*, *27*, 439–460.

Wand, M. P., & Jones, M. C. (1995). *Kernel smoothing.* London: Chapman and Hall.

Wang, K., & Gasser, T. (1997). Alignment of curves by dynamic time warping. *Annals of Statistics*, *25*, 1251–1276.

Wang, P. M., Cockburn, I. M., & Puterman, M. L. (1998). Analsis of patent data: A mixed Poisson regression model. *Journal of Business and Economic Statistics*, *16*, 27–41.

Waterhouse, S. (1997). *Classification and regression using mixtures of experts.* Ph.d. thesis, Cambridge University Engineering Department, Cambridge, UK.

Wedel, M., & DeSarbo, W. S. (1993). A latent class binomial logit methodology for the analysis of paried comparison data: An application re-investigating the determinants of perceived risk. *Decision Science*, *24*, 1157–1170.

Xue, Z., Shen, D., & Teoh, E. K. (2001). An efficient fuzzy algorithm for aligning shapes under affine transformations. *Patter Recognition*, *34*, 1171–1180.

Yeung, K. Y., Fraley, C., Murua, A., Raftery, A. E., & Ruzzo, W. L. (2001). Model-based clustering and data transformations for gene expression data. *Bioinformatics*, *17*(10), 977–987.

Appendices

A EM algorithm

The EM algorithm is an iterative ML procedure that provides a general and efficient framework for parameter estimation. At a basic level, EM is an approximate root-finding procedure that is used to seek the root of the likelihood equation. It iteratively searches for a set of parameters that maximize the probability of the observed data.

Due to the presence of local maxima on the likelihood surface, the EM solution is not guaranteed to correspond to a global maximum of the likelihood function. However, under fairly broad conditions we are guaranteed to find a stationary point of the likelihood equation corresponding to a local maximum (McLachlan & Krishnan, 1997). Furthermore, by running the EM algorithm multiple times from different starting points in parameter space and selecting the parameters from the run that results in the highest likelihood, we increase the chances of finding the global maximum.

We define the necessary prerequisite EM theory that is needed to understand the algorithm derivations given in this thesis. The EM algorithm is primarily used for estimating ML parameters in *missing-* or *hidden-data* problems. Parameter estimation in hidden-data problems is difficult because the missing data causes the

likelihood to take on a complex form.

For example, suppose we have data Y, hidden data Z, and parameter vector Φ. Then the likelihood of Φ given Y can be defined in a general way as

$$\mathcal{L}(\Phi|Y) = \int_Z p(Y|Z,\Phi)p(Z|\Phi)\, dZ. \tag{A.1}$$

It is understood in hidden-data problems that this integration cannot be easily calculated. The EM algorithm circumvents this problem by defining another likelihood function known as the *complete-data likelihood* that contains the missing data. The complete-data likelihood $\mathcal{L}_c$ is defined as

$$\mathcal{L}_c(\Phi|Y,Z) = p(Y,Z|\Phi). \tag{A.2}$$

However, the actual value of Z is unknown. So the EM algorithm "fills-in" values for the unknown Z by taking the expectation of $\mathcal{L}_c$ with respect to the posterior distribution $p(Z|Y,\Phi')$ for a fixed parameter vector Φ'.

This posterior distribution is known as the *hidden-data* distribution since it gives the distribution of the hidden data given the observed data Y and the current set of parameters Φ'. The expectation is taken as follows:

$$\mathrm{E}[\mathcal{L}_c|Y] = Q(\Phi,\Phi') = \int p(Y,Z|\Phi)p(Z|Y,\Phi')\, dZ. \tag{A.3}$$

We refer to this filled-in likelihood as the Q-function. The E-step is concerned with the procedures required in calculating Q.

The Q-function can be easily maximized since it does not contain any missing data. All of the missing data has been replaced by the operation of taking the expectation in the E-step. In the M-step, the Q-function is maximized with respect

to Φ to arrive at $\hat{\Phi}$:

$$\hat{\Phi} = \arg\max_{\Phi} Q(\Phi, \Phi'). \tag{A.4}$$

At the next iteration, Φ' is replaced by the new $\hat{\Phi}$ in the hidden-data distribution $p(Z|Y, \Phi')$ and the E-step is commenced again. Dempster et al. (1977) showed that under fairly general conditions, the likelihood will never decrease during the E- and M-step iterations.

B Monte Carlo cross-validation

In this appendix, we briefly outline the Monte Carlo cross-validation (MCCV) procedure as used in this thesis. The MCCV procedure (Shao, 1993; Smyth, 2000) consists of M runs where for each run the available data set is partitioned into a training and testing subset in which the testing subset is a fraction ν of the complete data set. The candidate models are then trained and tested on the corresponding subsets and the test scores are averaged over the M runs. MCCV is related to standard 10-fold cross-validation by setting $M = 10$, $\nu = 0.1$ and requiring the M test sets to be disjoint. However, in general the MCCV test subsets are not disjoint but instead each point has equal chance of being in any test set. Shao (1993) showed that in a regression context the estimation variability on the test sets can be reduced over standard cross-validation for relatively large values of ν (e.g., values of $\nu > 0.1$ in this context). Smyth (2000) found the value of $\nu = 0.5$ to be useful in a mixture context (for choosing the number of components or clusters). In this thesis, we used intermediate values of ν from 0.3 to 0.4.

C Matrix multivariate normal density

The matrix multivariate normal density is a useful rewriting of the standard multivariate normal density for multiple data vectors with special covariance. The standard multivariate normal density defined for d-dimensional data vector $\mathbf{x}$ with mean μ_x and covariance $\sigma_x^2\mathbf{I}$ is

$$p(\mathbf{x}|\mu_x, \sigma_x^2) = (2\pi\sigma_x^2)^{-d/2} \exp\left\{-\frac{1}{2\sigma_x^2}(\mathbf{x}-\mu)'(\mathbf{x}-\mu)\right\}. \tag{C.5}$$

Suppose we now introduce another d-dimensional data vector $\mathbf{y}$ distributed normally with mean μ_y and covariance $\sigma_y^2\mathbf{I}$ in which the covariance between $\mathbf{x}$ and $\mathbf{y}$ is $\sigma_{xy}^2\mathbf{I}$. A parsimonious way to define the joint distribution of $\mathbf{x}$ and $\mathbf{y}$ is to define their distribution in terms of the matrix $\mathbf{Z} = (\mathbf{x},\ \mathbf{y})$. The density of $\mathbf{Z}$ is matrix multivariate normal and is written as

$$p(\mathbf{Z}|\boldsymbol{\mu}, \Sigma) = (2\pi)^{-dm/2}|\Sigma|^{-d/2} \exp\left\{-\frac{1}{2}\mathrm{tr}\left((\mathbf{Z}-\boldsymbol{\mu})\Sigma^{-1}(\mathbf{Z}-\boldsymbol{\mu})'\right)\right\}, \tag{C.6}$$

with matrix mean $\boldsymbol{\mu} = (\mu_x,\ \mu_y)$ and covariance matrix

$$\Sigma = \begin{bmatrix} \sigma_x^2 & \sigma_{xy}^2 \\ \sigma_{xy}^2 & \sigma_y^2 \end{bmatrix}.$$

Note that $m = 2$ in this case and gives the number of columns of $\mathbf{Z}$ in general.