UNIVERSITY OF Southampton

University of Southampton Research Repository
ePrints Soton

Copyright © and Moral Rights for this thesis are retained by the author and/or other copyright owners. A copy can be downloaded for personal non-commercial research or study, without prior permission or charge. This thesis cannot be reproduced or quoted extensively from without first obtaining permission in writing from the copyright holder/s. The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the copyright holders.

When referring to this work, full bibliographic details including the author, title, awarding institution and date of the thesis must be given e.g.

AUTHOR (year of submission) "Full thesis title", University of Southampton, name of the University School or Department, PhD Thesis, pagination

http://eprints.soton.ac.uk

School of Electronics and Computer Science
Faculty of Engineering, Sciences and Mathematics
University of Southampton

Daniel Thorpe

January 2010

On Shape-mediated Analysis of Spatiotemporal Distributions

Supervisors: Professor Mark Nixon
Professor Peter Atkinson

Thesis submitted for PhD

Abstract

This research describes a new general framework to automate object-based analysis of a spatiotemporal property of a study phenomenon. Raw data is smoothly interpolated to provide a temporal series of surfaces represented as digital images. From each time sample, objects of activity are automatically extracted. These objects may be queried to provide spatial properties and shape descriptive Tchebichef moments. Objects close together from adjacent time samples are associated together to effectively track the phenomenon's property through space and time. A complete series of tracked objects, from spontaneous appearance to eventual disappearance is termed a Phenomenon Event, and is used principally as a means to analyse how various aspects, such as the raw value or centre of mass of the phenomenon's property changes over time.

Data sources of secondary, possibly explanatory variables, called covariates, are queried and processed in conjunction with the study phenomenon's data. By correlating properties between covariate objects and phenomenon objects, the nature of any relationship between them may be examined.

This general framework was developed using weekly surveillance of Influenza Like Illness (ILI) cases at participating general practitioners (GPs) across France since 1988 as the study phenomenon. Covariate data came in the form of three hourly weather observations at locations across France. These two disparate datasources expose the generality of the framework, as the Influenza data are digital images on disc, and the Covariate data are network accessed raw values stored in a database.

This novel approach employs modern shape description techniques from Computer Vision accompanied by geographical methods and traditional statistics. Such a treatment of surveillance data is new to epidemiology, and we hope it will provide a new perspective in the analyis of public health.

Acronyms

GP general practitioner

ILI Influenza Like Illness

WHO World Health Organisation

UNICEF United Nations Children's Fund

RBM Roll Back Malaria

ITN insecticide-treated net

IRS indoor residual spraying

GIS Geographic Information System

BMJ British Medical Journal

BLUE Best Linear Unbiased Estimator

SK Simple Kriging

OK Ordinary Kriging

BADC British Atmospheric Data Centre

NASA National Aeronautics and Space Administration

USGS United States Geological Survey

UTM Universal Transverse Mercator

CCP	conformal conic projection
IGN	Institut Geographical National
MSE	mean squared error
MSL	mean sea level
LOESS	locally weighted scatterplot smoothing
API	application programming interface
GPS	Global Positioning System
MODIS	moderate-resolution imaging spectroradiometer
SIR	Susceptible Infectious Removed
GVF	Gradient Vector Flow
MSER	Maximally Stable Extremal Regions

Acknowledgements

I would like to acknowledge the following people for their assistance:

- Professors Mark Nixon and Peter Atkinson, my supervisors, for providing me with guidance throughout this research.
- Sentinelles, Inserm-UPMC, Paris, http://www.sentiweb.org, for providing Influenza data which is central to this research.
- The Met Office and the British Atmospheric Data Centre for providing access to a rich source of global weather data.

Contents

Abstract i

Acronyms iv

Acknowledgements v

Contents ix

List of Figures xiv

1 Contributions 1

1.1 Spatiotemporal Phenomena 2

1.2 An Object-based Framework 3

2 Introduction 7

3 Source Data 11

3.1 Influenza Data 11

3.2 MIDAS Weather Data 13

4 Shape Description 15

4.1 Boundary Based Shape Description 15

4.2 Area Based Shape Description 20

4.3 Centralized Moments 21

4.4 Orthogonal Moments 23

4.5 Discrete Orthogonal Moments 26

4.6 Orthonormal Tchebichef Moments 29

5 Covariate Interpolation **39**

5.1 Implicit Covariates 40

5.2 MIDAS Weather Observations 41

5.3 Interpolation Techniques 43

5.4 Semivariogram 46

5.5 Kriging 49

5.6 Interpolating Covariate Data 51

6 An Object Based Framework **53**

6.1 Defining Shapes And Objects 53

6.2 Spatial Properties of Objects 58

6.3 Creating Objects from Raw Data 58

6.4 Phenomenon Events 60

6.5 Tracking Objects 61

6.6 Branching And Merging Objects 67

6.7 Forecasting Future Activity 69

7 Event Analysis **73**

7.1 Analysis with Covariates 73

7.2 Confirming Influenza's Seasonal Behaviour 76

7.3 Cross-correlation Between Shapes 78

7.4 Cross-correlation between Shapes and Samples 82

8 Conclusions **86**

8.1 Shape Discovery 88

8.2 Orthonormal Tchebichef Moments 90

8.3 Covariate Shape Extraction 92

8.4 Data 92

9 Further Work **94**

9.1 Phenomenon Event Centre Of Mass 94

9.2 Covariate Shapes 95

9.3 Population Density 97

9.4 Forest Fires: Further Phenomenons 98

9.5 Real Time Surveillance 100

References **102**

A Application Framework Architecture **109**

B Implementation Details **113**

B.1 Co-ordinate Conversion 113

B.2 Object Isolation 117

B.3 Orthonormal Tchebichef Moments 121

B.4 Covariate Interpolation 126

List of Figures

2.1 The number of confirmed (97) and probable (260) cases of swine-origin influenza A (H1N1) virus infections by the date of onset, in Mexico, during March 15 to April 26, 2009 [11]. 8

3.1 An example activity map of Influenza Like Illness 12

3.2 The location of weather stations used in this research, shown using Google Maps. 14

4.1 The result of the active contour algorithm. The contour was initially a circle encompassing all of France. 17

4.2 Using Gradient Vector Flow forces active contours to move into concavities, but still fails in many cases. 18

4.3 Legendre Polynomials, $P_n(x),\ n = 0, 1, \ldots, 5$ 24

4.4 Zernike polynomial, $V_{31} = \rho^2(-10 + 60\rho^2 - 105\rho^4 + 56\rho^6)\cos 2\theta$ 25

4.5 Orthonormal Tchebichef Polynomials, $\hat{t}_n(x),\ n = 0, 1, \ldots, 5; N = 64$ 31

4.6 Computing Tchebichef moments of shapes removed from their background automatically ensures translation invariance. 32

4.7 Mean squared error of image reconstruction using all moments from order zero up to and including the x-axis data point. 34

4.8 Image reconstruction from ever increasing numbers of Tchebichef Moments. 35

4.9 Demonstrating the linear response of Tchebichef Moments. 38

5.1 An example semivariogram, using the standard models. 47

5.2 Average air temperature across France for 2nd calendar week of the year with trend. 48

5.3 Variogram cloud of air temperature across France for 2nd calendar week of the year with fitted exponential semivariogram model. 49

5.4 Covariate data evenly interpolated from weather stations using semivariogram model shown in Figure 5.3. 52

6.1 Defining a *Phenomenon Shape*. 54

6.2 Examples of influenza activity 55

6.3 Example graph of connected shapes. Each shape is represented as a labelled circle. Blue circles are those at the the lowest intensity which only have a single branch. The coloured bands represent the quantised intensity levels present in the Influenza image data, which have an associated grey scale intensity, shown on the right. 56

6.4 Normalising evenly sampled spatial data 59

6.5 1998 week 50. The starting time sample used to generate the Phenomenon Events shown in Figures 6.6, 6.7 and 6.8. 61

6.6 Tracking disease outbreaks (year/week), part 1 64

6.7 Tracking disease outbreaks (year/week), part 2 65

6.8 Tracking disease outbreaks (year/week), part 3 66

6.9 Merging objects 68

6.10 Branching objects 69

6.11 Fitting cubic, quadratic and straight line models to example Tchebichef Moments, labelled as $(p : q)$, for the Phenomenon Event that starts in week 39 of 1998, in Figure 6.6d on page 64, and denoted by the white circles thereafter. 70

6.12 Estimate of week 40, 1998, only the black hexagon event had enough data to create a forecast. 71

6.13 Estimate of week 41, 1998 71

7.1 Shapes of air temperature for the first four weeks of event starting with in week 39 of 1998, see Figure 6.6d. 75
7.2 Phenomenon settings 76
7.3 Variation in influenza object mass for a phenomenon event starting in September of 1998: labeled with white circles starting on Figure 6.6d, and air temperature at the centre of mass of objects over the entire phenomenon season. 78
7.4 Correlation between Tchebichef moments of Influenza shapes with those of Air Temperature shapes for the 1998 39 event. The cross-correlations are normalised by the length of the event. 80
7.5 Correlation between Tchebichef moments of Influenza shapes with those of Dew Point shapes for the 1998 39 event. The cross-correlations are normalised by the length of the event. 80
7.6 Average cross-correlation of Tchebichef moments of Influenza with those of Dew Point and Air Temperature shapes for the 1998 41 event, which starts in Figure 6.6f on page 64. The cross-correlations are normalised by the length of the event. 81
7.7 Average cross-correlation of Tchebichef moments of the first 10 weeks of Influenza with those of Air Temperature and Dew Point shapes for the 1998 39 event. Additional plot of phenomenon object mass on a logarithmic scale. The cross-correlations are normalised by the length of the event. 82
7.8 Average cross-correlation of Influenza values with those of Air Temperature and Dew Point for the 1998 39 event. The cross-correlations are normalised by the length of the event. 84
7.9 Viewing Phenomenon event images from within the framework. 85
7.10 Performing correlation of a single event with air temperature from within the framework. 85

8.1 Hierarchy of area based objects. 87

8.2 Tracking Shapes and Creating Samples. 89

9.1 Quantising interpolated sampled spatial data 96
9.2 Examples of MODIS Burned Area Product 99
9.3 Sample GeoTiff MODIS Burned Area Product, from window 3, March 2008. The images are $13,654 \times 6144$ pixels in size and $\approx$ 160MB. 99
9.4 Twitter sourced real-time snow surveillance, showing my surveillance contribution at the time of writing. 101

A.1 Framework architecture 112

B.1 Developing an algorithm to isolate these objects from their surroundings. . 118
B.2 Flow diagram of object isolation algorithm. 119
B.3 Rectangles used to determine iteration variables when computing Tchebichef moments. 124

Chapter 1

Contributions

In experimental analysis it is often possible to measure the properties of phenomena in isolation and by doing so, explore their nature. The usual aim of pattern recognition studies is the development of a group of measures which recognise the study phenomenon with the least uncertainty. As the exact nature of the phenomenon most likely depends on several variables, it follows that any such group of measures likewise depends on several variables. Consequently, the established experimental procedure is to measure the effect on the phenomenon's property, by modifying each variable in turn while holding the others constant. To control such variables, scientists perform their experiments in a laboratory. For example, in studies of recognising subjects by the way they walk, called gait biometrics, video data files are recorded of subjects walking in a controlled environment thereby removing any transient lighting effects and background noise. While such variables most certainly exist in the real world, they complicate the problem unnecessarily for the development of a gait biometric; the use of a laboratory is to remove such factors. Furthermore, once a set of measures has been formulated to perform satisfactorily in laboratory conditions, the data may be augmented to include these and other variables, and structured to allow for their analysis. For a gait biometric study this might include recording subjects outside, or to vary the subjects clothing and footwear.

Unfortunately, not all phenomena are amenable to traditional research methodologies, as their nature depends on a conflation of factors: an unknown, inseparable combination of variables. Therefore, the focus of this research is a new approach, which is needed to understand the behaviour and attributes of such phenomena. This thesis does not outline a complete scientific methodology, but instead introduces a paradigm shift through the application of a novel *object* based framework to analyse spatiotemporal distributions.

1.1 Spatiotemporal Phenomena

Different spatiotemporal phenomena exist everywhere, and in many cases are entirely conceptual and immeasurable, therefore the development of a truly general method to explore their nature is impossible. For example, it is not possible to measure the act of imagination, yet clearly innovation and creativity vary between people across the globe and throughout history. However, the properties of some spatiotemporal phenomena are measured, or their presence observed, and they are already the focus of current research efforts. For example, since the birth of our ancestors, humankind has watched and measured the change in weather, for our survival often depended on it. Now in the modern world daily bulletins provide forecasts for the week ahead, and farmers are able to estimate their harvest months in advance. Weather observations are an archetype of spatiotemporal distributions, and indeed they feature prominently in this research. Flash forest fires, which occur in many different parts of the globe, are another example, and one of increased research focus given the recent devastation caused in California [9]. Distributions such as these are most commonly modeled using a random field as their properties are measured at numerous locations over time. For example, weather stations are distributed around the globe and measure air temperature, wind speed, and other factors. Forest fires are monitored using satellites and Geographic Information Systems (GISs) which determine their location and extent. The surveillance of infectious diseases provides data that are studied extensively by epidemiologists and statisticians. Again the raw data generated are measurements of a property, such as the number of cases of Influenza Like Illness (ILI) presented to a general practitioner (GP), over a specified time frame and given the position of the GP's practice, a spatiotemporal distribution is developed.

1.2 An Object-based Framework

It is proposed that formulating spatiotemporal distributions into an object-based framework will provide further insight into their characteristics and enable forecasting of those characteristics. A spatiotemporal distribution is a surface that changes over successive time samples and, therefore, current research methodology treats it as a random field that changes over time. This research develops a novel approach to analyse the surface by extracting distinct objects, or areas, where the intensity of the measured property is considered interesting or important. These objects exhibit spatial attributes of the study phenomenon's measured property, but over sub-samples of the domain and at positions of interest. A formulation of the data in this way will be a boon to research as the objects possess spatial attributes such as its extent, spatial position, total "mass", the centre of mass, and even high frequency spatial changes or curvature.

In application, for example with infectious disease surveillance, it would be possible to forecast not only the expected levels of disease activity but where it will be most concentrated and the specific area where public health agencies should focus their response. This information may increase their effectiveness and provide significant cost savings. The possibility of accurate, location aware, forecasting of forest fires, infectious disease and other naturally occurring events is a very exciting prospect. To this end the contribution of this research is presented in four areas, which are introduced below and discussed extensively in the remainder of this thesis.

This thesis is organised as follows. Chapter 2 introduces the need for the research, most importantly that effective analysis of complex spatiotemporal phenomena can be achieved with a paradigm shift in methodology. Chapter 3 describes the data used as the study phenomenon and the source of multiple covariates. Chapter 4 is a detailed overview of the current status of shape descriptive techniques, including the computation of orthonormal Tchebichef moments and image reconstruction based on the said moments. It also presents new insight into the properties of orthonormal Tchebichef moments which are used to describe the spatial properties of objects in this methodology. Chapter 5 describes the spatial statistical methods used to generate digital images from spatially sampled data.

Chapter 6 develops the notion of objects. This includes a discussion of their abstract definition and what information about the measured property of the phenomenon is intended to be captured. Tchebichef Moments are computed for every object, thereby providing an avenue to query the object of its own spatial properties such as mass and curvature. Indeed, each object can approximate its own shape using a sub-set of moments, mathematical models may then be developed to describe the general shape of the phenomenon's measured property. The chapter continues with an examination of the methodology developed to realise objects from the raw data. This new approach isolates subsamples of data that are determined to be of interest. This decision process will most likely depend on the study phenomenon. For example in observing instances of forest fires, the measured data are binary in nature, as the fire is either observed or not. But measurements of air temperature are signed floating point values, and the researchers might be interested only in sub-zero temperatures.

Once objects have been created from each time sample of the spatiotemporal distribution, each object from each time sample is associated with up to two other objects: one from the previous time sample and one from the next time sample. In effect, this process tracks the objects and as a result, encapsulates the spatial deformation of the measured property over time. Section 6.5 examines the evolution of the tracking procedure from its simplistic beginnings to its intelligent and adaptive final state.

The result of object tracking is a series of objects that are connected over time. A set of connected objects is termed a *phenomenon event*. It is possible to forecast the future behaviour of events based solely on the history of the objects. The novel method used to perform this forecasting, which is founded on the variation of the objects' spatial properties, is discussed extensively in Section 6.7 and followed with a presentation of the result of forecasting arbitrary events from a study phenomenon.

At the beginning of this chapter, the notion that a set of unknown variables influence the properties of phenomena was introduced. It is proposed that analysis of these variables, or covariates, in context with the study phenomenon's properties will aid understanding of the phenomenon, and increase forecast accuracy. Such covariates will most likely be spatiotemporal distributions themselves. For example, forest fires will not be present in locations where they have recently existed. Therefore, the historical presence of the phenomenon is itself a covariate for the phenomenon. For many phenomena, the complex relationships that exist between their covariates, form a problem domain that is largely impenetrable without the development of highly specialised methods and equipment tailored to the study phenomenon. To research effectively gait biometrics, introduced earlier, a highly specialised apparatus, used to record subjects under controlled conditions and covariates, has been developed [47]. Instead of bespoke experiments, it is shown that an object-based methodology provides a general approach to analyse the covariate space of a study phenomenon. Chapter 7 continues by developing the approach to analyse covariate data in the same context as the study phenomenon, and presents results indicating how such analysis can augment the object-based forecasting process.

Chapter 8 presents the conclusions drawn from the research, and Chapter 9 highlights avenues of further research that could be explored given more resources. Lastly Appendix A details the architecture developed to realised the methodology and Appendix B contains some specific implementation details of the novel aspects of the framework.

Chapter 2

Introduction

This research is aimed at providing an object-framework to analyse spatiotemporal phenomena and understand the impact that covariates can have on them. Covariates are variables that may influence the outcome under study. For example, infectious diseases will be influenced by some degree by population density, air temperature or quality of life measures such as amount of disposable income. A greater understanding of the symbiotic relationships that exist between phenomena and their covariates will provide a basis for further research into either diminishing, augmenting or predicting the phenomenon under study.

Surveillance of influenza [59] shows that it is highly dynamic and accordingly unpredictable. It is postulated that a framework to explore how the properties of influenza can vary with the those of covariates will be invaluable to public health organisations. For example, it is hoped that such a system will benefit international healthcare, by providing accurate estimations, of the required quantity of drug, for a given locality and future date. Currently weather forecasts are part of daily bulletins, yet regular epidemiological forecasting designed for the general public is non-existent, a point stressed by Viboud et al. [64].

In 2000, the British Medical Journal (BMJ) carried 3 articles [66] regarding the unexpected influenza rates in the United Kingdom, and the pressure on health care facilities due to record levels of hospital admissions. In January 2008, the number of norovirus cases reported was double that of the previous year [24], although more effective diagnosis, and increased public awareness are contributing factors to this sharp rise. The BBC reported that at least 100 hospital wards were closed to new patients across England and Wales, and it is estimated that between 600,000 and one million people will become infected each year [8].

On April 25th 2009, the World Health Organisation (WHO) declared a "public health emergency of international concern" [13] after an outbreak in Mexico of a new strain of influenza in mid-March. This new strain is referred to in the media as "swine flu" due to its origins, but more accurately it is an Influenza A (H1N1) subtype where several gene segements match those of the 1998 North American swine influenza strains [69]. The announcement by the WHO of an *international* emergency occured just 39 days after 60% of the population of La Gloria, Veracruz, a small town in Mexico, was sickened by a respiratory illness. The illness was confirmed as a Swine-Origin Infleunza A (H1N1) virus infection and separate cases were confirmed in Imperial County and San Diego County, California in late March. On April 12th, the earliest known fatality related to the outbreak was confirmed. By April 27th swine-origin influenza cases are confirmed in Canada, Spain and the United Kingdom, while in Mexico there are seven confirmed deaths. Figure 2.1 shows the initial reports of the ILI cases in Mexico during March and April 2009 [11].

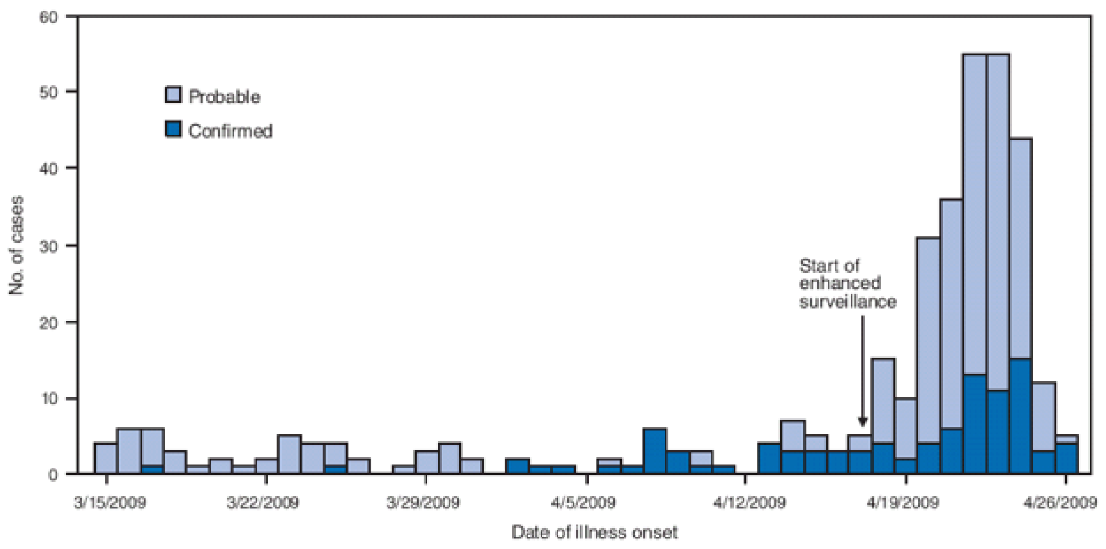


Figure 2.1: The number of confirmed (97) and probable (260) cases of swine-origin influenza A (H1N1) virus infections by the date of onset, in Mexico, during March 15 to April 26, 2009 [11].

At the time of writing the latest weekly report from the WHO [68], that of September

20th 2009, states that there have been more than 300,000 laboratory confirmed cases of pandemic influenza H1N1 and 3917 deaths, in 191 countries and territories reported to the WHO. This demonstrates how quickly a pandemic becomes a global emergency, and that the response from public health agencies must be just as swift to minimize the damages. The speed at which the 2009 strain spread reflects the ease of international travel; indeed the first reported case in the United Kingdom was a Scotsman returning home from Mexico [10]. The ready movement of people ensures that perfect containment of infectious diseases is practically impossible, so instead health agencies must coordinate their efforts to provide a global response.

The recent pandemic scares of SARS in 2002, avian-origin influenza (H5N1) in 2005 and the aforementioned swine-origin influenza of 2009 highlight that these global events are not uncommon. While this research does not focus on pandemics, they illustrate that a greater understanding of complex space-time varying phenomena is required. Indeed the WHO stressed it needed to "intensify surveillance of unusual flu-like illness and severe pneumonia" in a statement following the 5th meeting of the Emergency Committee in September 2009 [12].

I propose that a model which describes disease activity accurately will be founded on a reliable understanding of its covariate structure. While this research focuses on infectious diseases, it is proposed that other research areas will benefit from an approach to improve the understanding of their covariate structure. For example, facial expressions can be viewed as covariates in a face recognition system; they are variables that may affect the outcome under study. Likewise, changes over time of a subjects biometrics, may be regarded as a covariate. While these are examples of research areas where the covariate structure is ill-defined, the remainder of this research focuses on infectious diseases.

Infectious diseases can disrupt the daily routine of humans considerably. Contracting influenza may cause days of discomfort and lost productivity, while a mosquito bite in some parts of the world may cause eventual death. Malaria has become "one of the world's leading re-emerging infectious diseases, infecting an estimated half billion people a year and killing up to 2 million" [19]. In the World Malaria Report 2005, [33], the WHO state that "107 countries and territories have reported areas at risk of malaria transmission. Although this number is considerably less than in the 1950s, ... 3.2 billion people are still at risk". The prevention strategies recommended by the WHO, which consist mainly of using insecticide-treated nets (ITNs) and indoor residual spraying (IRS), rely on accurate knowledge of the current spatial distribution of the disease, with a degree of insight into the future distribution. Likewise, reliable information about the expected spatial distribution of influenza is necessary, for the efficient manufacture and delivery of the yearly injection, that minimises lost human productivity and money, during the flu season.

As disease varies in a spatio-temporal fashion, it is proposed to utilise techniques developed in image processing research to analyse past disease activity available from surveillance sources. Information from GISs and national census results can provide covariate data. By comparing these disparate data-sets in the same context accurate conclusions about the relationships between the disease and covariates may be drawn.

Chapter 3

Source Data

While this research aims to develop a *general* framework to analyse multi-dimensional phenomena, it is actually realised in the context of a study phenomenon. As an example application we are using influenza data made available through the Sentinelles Network project as discussed in the following section. As we aim to consider the effect of various covariates for influenza, data for these is also required. Section 3.2 details the treatment of weather data which is investigated as a possible covariate.

3.1 Influenza Data

The Réseau Sentinelles, or Sentinelles Network is a French collaboration between general practitioners and researchers [59]. The network currently consists of 1260 volunteering GPs, who work throughout the metropolitan area of France. The aim is to provide clinical surveillance in France for 14 health indicators including ILI and Mumps, surveillance of which started in 1984. The epidemiological data is collected in real-time, and is used internally within Inserm[1], the regulating body, for analysis, forecasting and redistribution.

The data is made available to the public in a weekly report containing activity maps of the monitored diseases. The raw data is not available to the public due to restrictions in

[1]French National Institute of Health and Medical Research

French law. Figure 3.1 shows an example activity map, and such maps can be created for each week since November 1984; almost 1200 images.

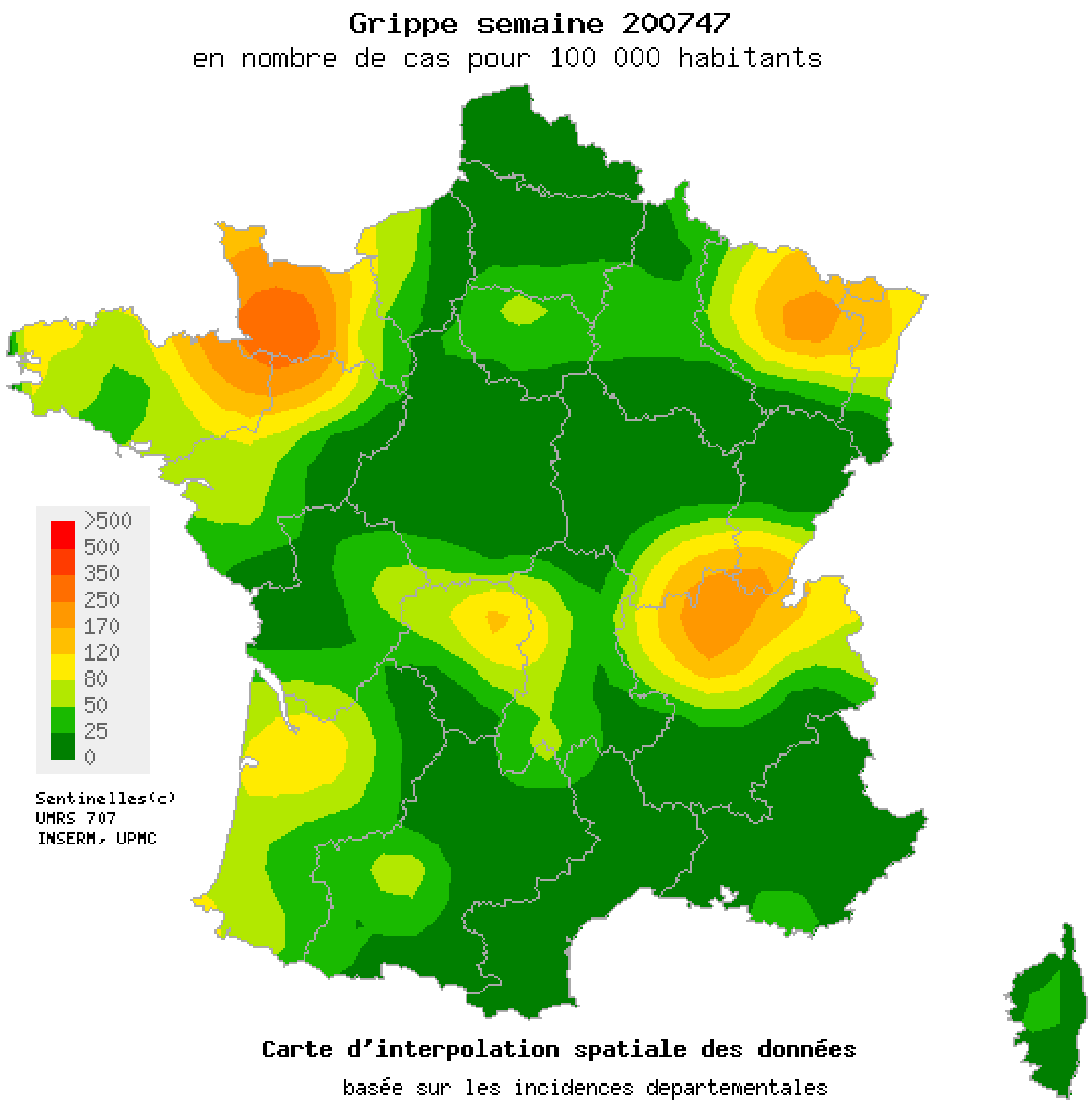

Grippe semaine 200747
en nombre de cas pour 100 000 habitants
>500
500
350
250
170
120
80
50
25
0
Sentinelles(c)
UMRS 707
INSERM, UPMC
Carte d'interpolation spatiale des données
basée sur les incidences departementales

Figure 3.1: An example activity map of Influenza Like Illness

There are a number of issues with using such images as the basis for this research, and access to the raw data would clearly be preferable. However their provenance should not detract from the thrust of this research, which is the development of an analysis technique not a specific result. As such the techniques described here will be applicable in other problem domains. Additionally, having the data in these image formats introduces some practical problems that must be surmounted, as discussed briefly in this section.

The activity maps show the number of cases for 100,000 habitants, although the spatial resolution of this population density cannot be determined. 83,933 pixels are used to represent continental France, the land area of which is 543,015 km^2, therefore the resolution of the map is approximately 1 pixel to 6.5km^2. We are assuming that the image faithfully represents the actual geography of France. The curvature of the shapes in Figure 3.1 varies smoothly, hence we can infer that the resolution of the population density used to generate the map is at least comparable to that of the image itself, if it were less the shapes would appear pixellated. Lastly, the geodesic projection used to create the map is a Lambert II conformal conic projection [26], which is discussed later in this chapter.

The number of cases of ILI is quantized into 9 bands, so the actual number of disease cases cannot be retrieved. The range of the bands increases exponentially, with the highest band sample being greater than 500. The overall result of this is that we cannot effectively draw any quantitative conclusions from the analysis with regard to the physical number of ILI cases. Consequently the analysis is focused on revealing *trends* in the data. Additionally when we consider the image to be a varying surface, it is not smooth, which leads to a treatment of the data as a number of slices through the surface, commonly called level sets.

3.2 MIDAS Weather Data

Global weather data is available from the British Atmospheric Data Centre (BADC) through the UK Met Office as part of their MIDAS database program. Each observation records the weather station, observation period and the observed values. The observation period used for this research is from 1990-01-01 to 2005-12-31 and is three-hourly with additional daily averages. The observed values, or covariates are

- wind direction
- wind speed
- visibility
- air temperature (minimum, maximum and mean average)
- dew point

- mean sea level pressure
- precipitation hour count
- precipitation amount

This wealth of data, over 2.1 million individual weather observations, covering 257 weather stations across Europe, is stored in a MySQL database. This allows it to be fully searchable, remotely accessible and tightly integrated into the framework. An aspect of this integration is also demonstrated by Figure 3.2 which uses Google Maps to plot the location of the weather stations.

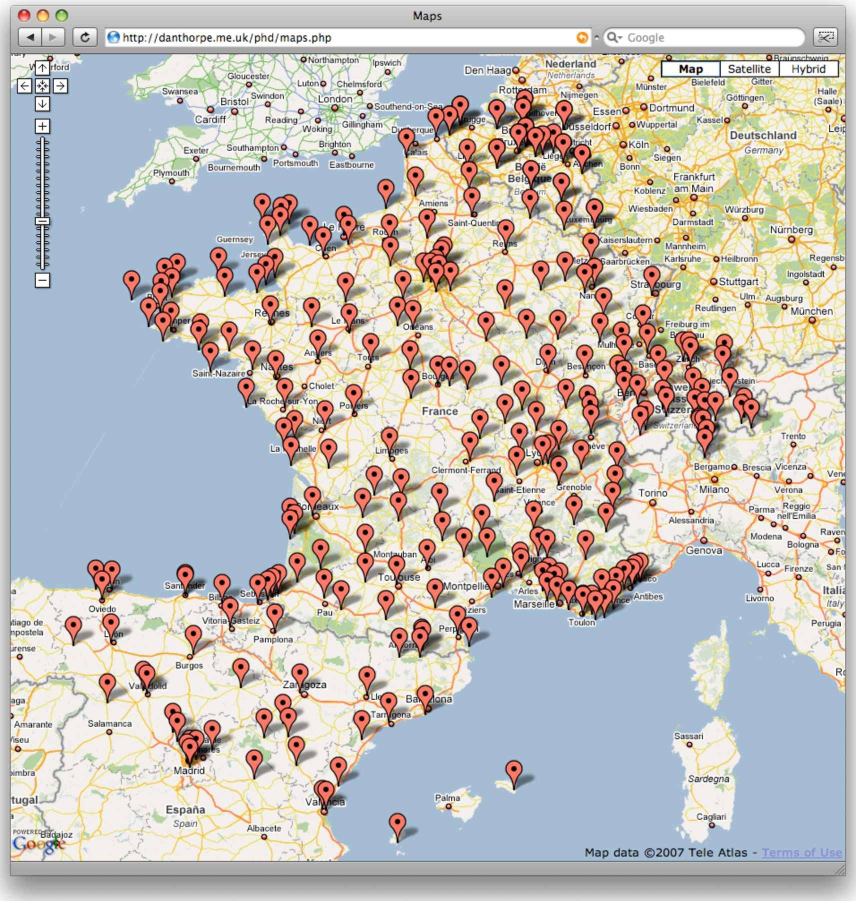

Maps
http://danthorpe.me.uk/phd/maps.php
Google
Map
Satellite
Hybrid
France
España
Deutschland
Map data ©2007 Tele Atlas - Terms of Use

Figure 3.2: The location of weather stations used in this research, shown using Google Maps.

Chapter 4

Shape Description

Shape description techniques aim to represent the spatial features of a digital image as a unique set of numbers. There are two distinct methodologies, boundary descriptions and area based descriptions.

4.1 Boundary Based Shape Description

Boundary based shape description techniques are a family of methods which perform object segmentation by extracting the dividing line between a foreground object and its background.

4.1.1 Active Contours

Active Contours, or Snakes [28] are curves defined in the image domain that may be influenced by the curve's own properties and by external forces derived from the image itself. When given the correct external force models and initial parameters, at their lowest energy the active contour may describe the boundary of an object inside the image as a pair of distributions. The distributions are the displacement of the object's boundary in the image's x and y axes over the length of the contour. A pair of such signals may be used to compute Elliptic Fourier Descriptors.

Active contours are formally defined as a curve $\mathbf{C}(s) = [x(s), y(s)], s \in [0, 1]$, which is calculated as a series of points in the image domain. The final contour is mapped to image objects by moving the points iteratively to minimise an energy functional in the form

$$E_{snake} = \int_0^1 E_{cont}(\mathbf{C}(s)) + E_{curv}(\mathbf{C}(s)) + E_{img}(\mathbf{C}(s))\,\mathrm{d}s \tag{4.1}$$

The first term, $E_{cont}(\mathbf{C}(s))$ represents the contour's *continuity* and determines the spacing of the points. It is calculated as

$$E_{cont}(\mathbf{C}(s)) = \alpha(s)\left|\mathbf{C}'(s)\right|^2 \tag{4.2}$$

and measures the energy due to stretching the snake. The contour's tension and rigidity maybe varied by $\alpha(s)$. Low values of α imply that the contour's points can vary greatly in spacing, whereas high values will ensure the contour tends toward points evenly sampled along the contour.

The second term corresponds to the contour's *curvature*, and is calculated as

$$E_{curv}(\mathbf{C}(s)) = \beta(s)\left|\mathbf{C}''(s)\right|^2 \tag{4.3}$$

Low values of β reduce the imperative to minimise curvature, which allows for sharp corners to be formed. The derivatives in equations (4.2) and (4.3), $\mathbf{C}'$ and $\mathbf{C}''$, represent the first and second derivatives of the curve with respect to s, the position along the curve, and for computation may be approximated using finite differencing. If $\mathbf{C}_i = (x_i, y_i)$ is a point on the contour the following approximation may be used

$$\left|\mathbf{C}_i'\right|^2 = \left|\frac{\mathrm{d}\mathbf{C}_i}{\mathrm{d}s}\right|^2 \approx \left|\mathbf{C}_i - \mathbf{C}_{i+1}\right|^2 \tag{4.4}$$

$$\left|\mathbf{C}_i''\right|^2 = \left|\frac{\mathrm{d}^2\mathbf{C}_i}{\mathrm{d}s^2}\right|^2 \approx \left|\mathbf{C}_{i-1} - 2\mathbf{C}_i + \mathbf{C}_{i+1}\right|^2 \tag{4.5}$$

The third term in equation Eq (4.1) represents the image energy. It is designed so that regions of interest, which in this case are boundaries or edges, have smaller values. Typically this external energy will be the inverted magnitude of an edge detector such as the Sobel operator. The inversion is required because the energy functional is to be minimised, so image edges should appear as minimum values.

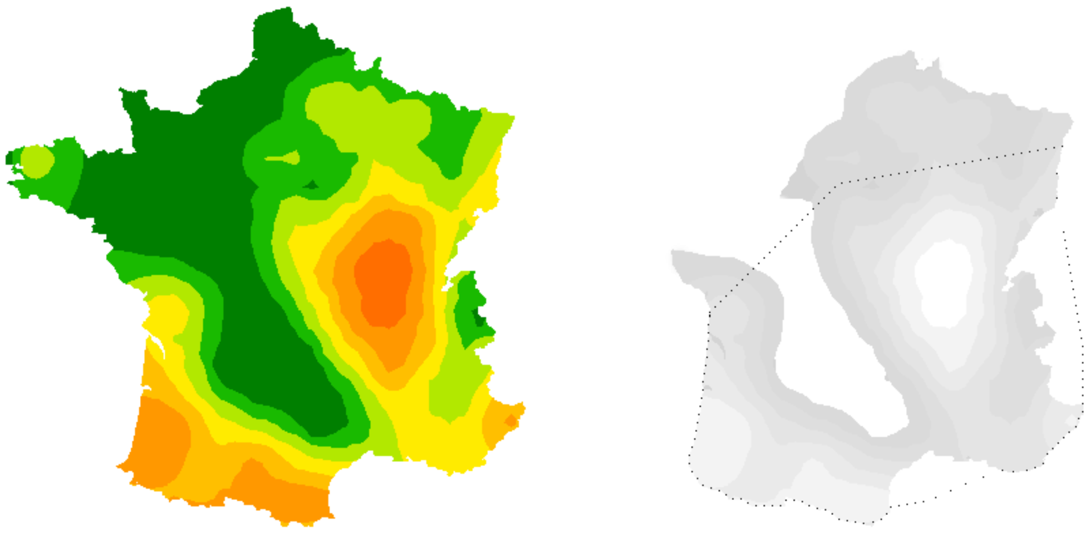

(a) Input image **(b)** Final iteration of active contour

Figure 4.1: The result of the active contour algorithm. The contour was initially a circle encompassing all of France.

Figure 4.1 shows the result of computing an active contour on a typical example of Sentiweb image data. The algorithm used to generate this result is based on the initial research in this field and significant research has been undertaken to improve the performance of active contours. This result is included as it highlights some of the major issues with applying active contours in this context. The objects that will require description are amorphous blobs, and as such will quite likely feature protrusions, appendages and concavities such as those in Figure 4.1a. The algorithm described thus far is not able to calculate negative curvatures, nor inject extra points which would be required to evolve the snake into the concave regions of the object. Techniques to overcome these inadequacies have been developed. Notably Gradient Vector Flow (GVF) snakes advances the contour further into concavities by using an external force that diffuses the gradient vectors thereby improving convergence of the snake with the image boundary.

4.1.2 Gradient Vector Flow

Standard parametric snakes fail to advance into concavities because the external forces within the concavity point towards the object boundary, and therefore at either side of the concavity the forces point in opposite directions. Hence, when the contour evolves it reaches the mouth of the concavity and never proceeds into it. There is no choice of α and β that will resolve this problem. A new external force that rectified this was introduced by Xu[72] and called Gradient Vector Flow. It is defined to be the vector field, $\mathbf{v}(x,y) = [u(x,y), v(x,y)]$ that minimises the energy functional

$$E_{gvf} = \iint \mu(u_x^2 + u_y^2 + v_x^2 + v_y^2) + |\nabla f(x,y)|^2 \, |\mathbf{v} - \nabla f(x,y)|^2 \, \mathrm{d}x \, \mathrm{d}y \tag{4.6}$$

where $f(x,y)$ is the edge map of $I(x,y)$, and $\nabla f(x,y)$ is the gradient of the edge map. Clearly this introduces another layer of iterative computation as $\mathbf{v}$ must be calculated before the active contour can start evolving to its minimum state.

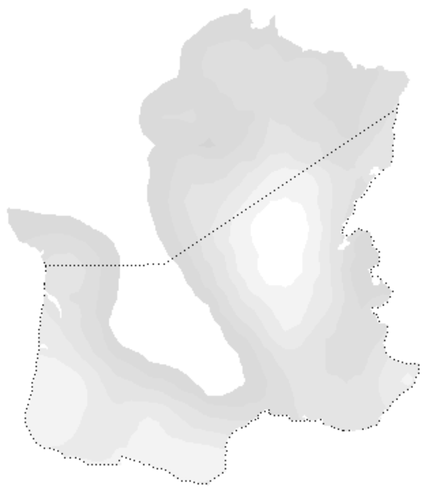


Figure 4.2: Using Gradient Vector Flow forces active contours to move into concavities, but still fails in many cases.

Figure 4.2 shows that the use of a GVF field produces a better active contour, however it still fails to advance into extreme concavities. Additionally, the introduction of GVF adds another layer of computation.

4.1.3 Level Set Methods

Level Set Methods provide a different approach to object boundary detection by increasing the dimension so that a surface, rather than a curve, is used. The surface is evolved by advancing it in a direction normal to its own curvature. The level sets are slices through the surface parallel to the image plane, so that the zeroth level set is the intersection the image makes with the surface, and hence, once the surface has evolved represents the object boundary. The advantage of this method, is that because the level sets are intersections a plane makes with a curved surface, is it possible for the place to intersect the surface in more than one place. For example, the curve at which a plane intersects a Gaussian surface is a circle. Hence, it is possible to extract complex shapes that may contain holes and concavities. The disadvantages of the method are that it is fundamentally very complex to implement, and at the time this topic was researched, there were no library routines available for level set methods.

4.1.4 Object Boundaries as a basis for Shape Description

The amount of research devoted to extracting object boundaries is significant. Largely this research is aimed at developing new methods to overcome the inadequacies of active contours, such as GVF [71, 54, 72] and Level Set Methods [1, 53, 50, 60]. These inadequacies are principally that the technique relies heavily on its initial parameters, is easily affected by noise and is not able to extract complex shape boundaries. The initial parameters of an active contour algorithm are choices for α, β and other constraining variables required to maintain a balanced energy functional. Additionally the starting contour has a significant effect on the final contour. Methods introduced to improve upon the result of active contours often introduce further initialisation parameters, boundary constraints and computation before addressing ways of maintaing the result while reducing the parameters.

It is necessary to consider what information is provided by the object boundary. Fourier Descriptors can consume the result of parametric active contours to provide a description of the shape boundary. So called implicit contours, such as that produced by level set methods, are less useful as a basis for description, as the final contour is not in a parametric form. Moreover, in general, a shape is not defined solely by its boundary, but instead by the spatial distribution of all its pixels and their intensities. In this research the shapes are not physical objects but actually represent the observed properties of a study phenomenon, therefore the boundary of a shape only represents where the spatial sampling of the property was bounded, not the property itself. In the case of the Sentiweb data, the boundary of the shape is that of France itself, so any descriptions of shapes based on this boundary would not provide a wide range of descriptions.

In conclusion, boundary based shape description methods are computationally expensive, have numerous initialisation parameters and do not capture the information the shapes in this research represent. Therefore, area based shape description methods are considered.

4.2 Area Based Shape Description

Equation (4.7) is the general moment function; where Ψ_{pq} is the $p+q$ order moment of the image $f(x,y)$, calculated using the basis function $\psi_{pq}(x,y)$.

$$\Psi_{pq} = \int_x \int_y \psi_{pq}(x,y) f(x,y)\,\mathrm{d}x\,\mathrm{d}y \tag{4.7}$$

Geometric moments are computed using this form with the basis set $\{x^p y^q\}$ as show in equation (4.8), where (W,H) is the size of the image.

$$m_{pq} = \sum_x^{W-1} \sum_y^{H-1} x^p y^q f(x,y) \quad p,q = 0,1,2,\ldots,\infty \tag{4.8}$$

Specific geometric moments provide some properties of the image. The zeroth moment, m_{00}, is the sum over all the image pixels, and therefore represents the *mass*[1] of the image. The image's centre of mass is effectively the location in each axis of the pixel's peak intensity

[1]Mass in this context refers to the image's pixel intensity, and the total summation of the intensities can be considered the image's mass in the Newtonian sense.

distribution. This can be calculated by dividing the first order moments, m_{01} and m_{10} by the zeroth order moment.

$$\begin{aligned} m_{01} &= \sum_{x}^{W-1}\sum_{y}^{H-1} y f(x,y) \\ m_{10} &= \sum_{x}^{W-1}\sum_{y}^{H-1} x f(x,y) \\ (\bar{x},\bar{y}) &= \left(\frac{m_{01}}{m_{00}}, \frac{m_{10}}{m_{00}}\right) \end{aligned} \tag{4.9}$$

4.3 Centralized Moments

Centralized moments are geometric moments that are translation invariant, because the pixels are indexed not from the image origin, but from the centre of mass.

$$\mu_{pq} = \sum_{x}^{W-1}\sum_{y}^{H-1} (x-\bar{x})^p (y-\bar{y})^q f(x,y) \tag{4.10}$$

Hu, [25] incorporated scale invariance:

$$\begin{aligned} \eta_{pq} &= \frac{\mu_{pq}}{\mu_{00}^{\gamma}} \\ \gamma &= 1 + \frac{p+q}{2} \end{aligned} \tag{4.11}$$

and seven rotation invariant moments can be computed from these normalized centralized moments [48].

Low order moments provide low level information: mass, position, size and orientation [61]. Mukundan [46] shows that third-order moments $\mu_{30,03}$ denote the *skewness* of the image projections, which indicates symmetry about the centre of inertia. Furthermore fourth-order moments, $\mu_{40,04}$ denote *kurtosis* of the image, which is "a measure of the flatness or *peakedness* of a curve" [46]. Moments above fourth-order do not not have analogous statistical measures, but instead represent larger variation in the spatial distribution of the image's intensity; the high frequency information. [25, 61, 46] explore the relationship between image moments and the Fourier transform, and their similar properties cement the notion that the moment orders are effectively a filter, whereby low order moments execute a low-pass filter and represent the low level information, and high order moments represent in detail in the image.

The basis set, $\{x^p, y^q\}$, increases in magnitude exponentially as the moment order increases. Accordingly the distribution of moment values is skewed by the high order moment values so that the dynamic range of the set of moment is very large, which makes numerical interpretation problematic. All geometric kernels share these characteristics. Additionally the set of moments is not bounded, which results in an increasing degree of information redundancy in the high order moments. This is a significant deficiency as it follows that it is not possible to completely describe a shape with a finite set of geometric moments. Therefore exact shape reconstruction from geometric moments is effectively impossible, and reconstruction of credible approximations are computationally expensive due to the large number of moments required.

In order to achieve complete shape description it is necessary to use an orthogonal basis so that the information provided by each moment is unique. This removes redundant information from the moment set and limits the complete description to a finite, and therefore realizable, set of numbers.

Two vectors, $f(x)$ and $g(x)$ are orthogonal if their inner product is zero:

$$\int_a^b f(x)g(x)w(x)\,\mathrm{d}x = 0$$

where $w(x)$ is a non-negative weight function.

A basis set is orthogonal if all the vectors it defines are orthogonal. The basis set of geometric moments (and by extension centralized moments) are not orthogonal, this can be proved mathematically as follows, assuming a weighting function $w(x) = 1$:

$$\begin{aligned}
\int_{0}^{N-1}\int_{0}^{N-1} x^p y^q \,\mathrm{d}x\,\mathrm{d}y &= \int_{0}^{N-1} x^p \,\mathrm{d}x \int_{0}^{N-1} y^q \,\mathrm{d}y \\
&= \left[\frac{x^{p+1}}{p+1}\right]_0^{N-1} \left[\frac{y^{q+1}}{q+1}\right]_0^{N-1} \\
&= \left(\frac{(N-1)^{p+1}}{p+1}\right)\left(\frac{(N-1)^{q+1}}{q+1}\right) \\
&\neq 0 \qquad \forall\, p,q = 0,1,2,\ldots,\infty
\end{aligned}$$

4.4 Orthogonal Moments

There are a number of popular type of orthogonal image moments such as Legendre, Zernike and Jacobi-Fourier moments, about which exists a significant body of literature [45, 18, 16, 23, 31, 52, 4, 27, 29, 30, 34, 17, 65].

The solutions to Legendre's differential equation:

$$\frac{\mathrm{d}}{\mathrm{d}x}\left[(1-x^2)\frac{\mathrm{d}}{\mathrm{d}x}P_n(x)\right] + n(n+1)P_n(x) = 0$$

for $n = 0, 1, 2, \ldots$ form a set of polynomials referred to as the Legendre polynomials:

$$P_n(x) = \frac{1}{2^n n!}\frac{\mathrm{d}^n}{\mathrm{d}x^n}\left[(x^2-1)^n\right] \tag{4.12}$$

which are orthogonal on the interval $-1 \leq x \leq 1$. The first 6 orders are shown in Figure 4.3.

Teague [61] defines the Legendre moment of $f(x, y)$ as

$$\lambda_{pq} = \frac{(2q+1)(2p+1)}{4}\int_{-1}^{1}\int_{-1}^{1} P_p(x)P_q(y)f(x,y)\,\mathrm{d}x\,\mathrm{d}y \tag{4.13}$$

Chong *et. al.* [18] adds translation and scale invariance directly from the Legendre polynomial. Teague [61] also outlines orthogonal moments founded on the Zernike polynomial,

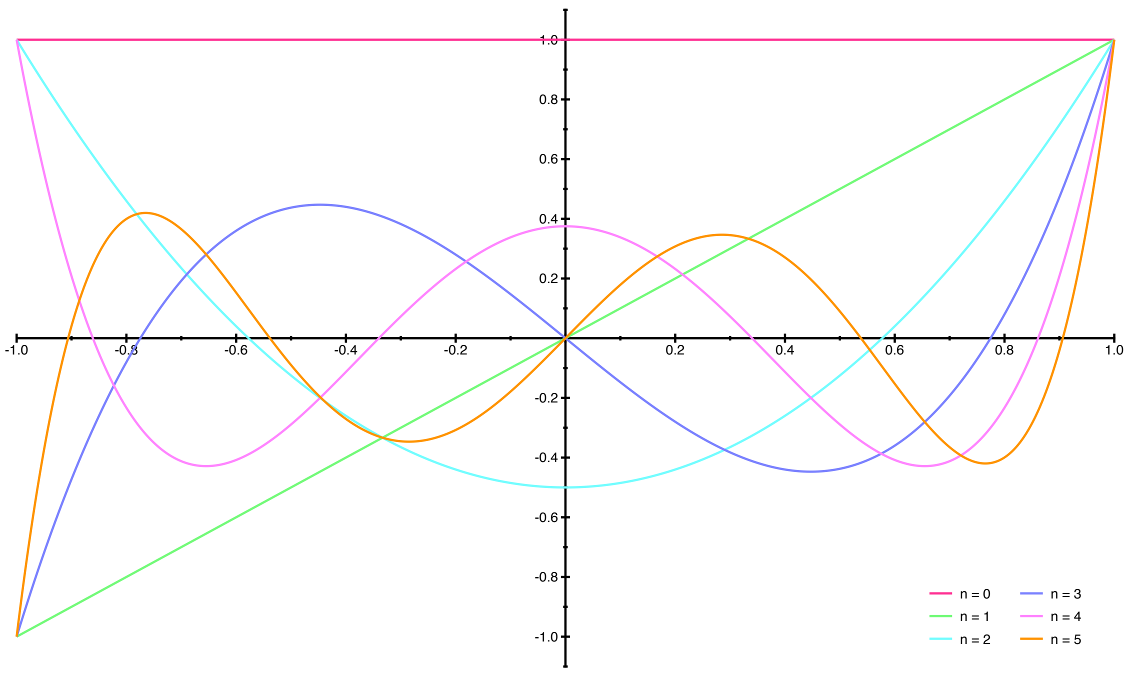


Figure 4.3: Legendre Polynomials, $P_n(x),\ n = 0, 1, \ldots, 5$

for order n and repetition l

$$V_{nl}(x, y) = V_{nl}\left(\rho \sin\theta, \rho\cos\theta\right) = R_{nl}(\rho)\exp(jl\theta)$$

where $n = 0, 1, 2, \ldots, \infty$ and l takes on positive and negative integer values subject to the conditions

$$n - |l| \text{ is even}, \quad |l| < n$$

The Zernike polynomial is orthogonal within the unit circle. An example polynomial, V_{31} is plotted in Figure 4.4. The Zernike moment is defined as

$$Z_{nl} = \frac{(n+1)}{\pi} \int\limits_0^{2\pi} \int\limits_0^{1} V_{nl}^*(\rho, \theta) f(\rho, \theta) \rho \,\mathrm{d}\rho \,\mathrm{d}\theta, \quad \rho \leq 1 \tag{4.14}$$

Zernike moments can naturally provide radial invariance as they are defined using radial polynomials. Chong *et. al.* [17] again provide formulations to render the moments translation invariant.

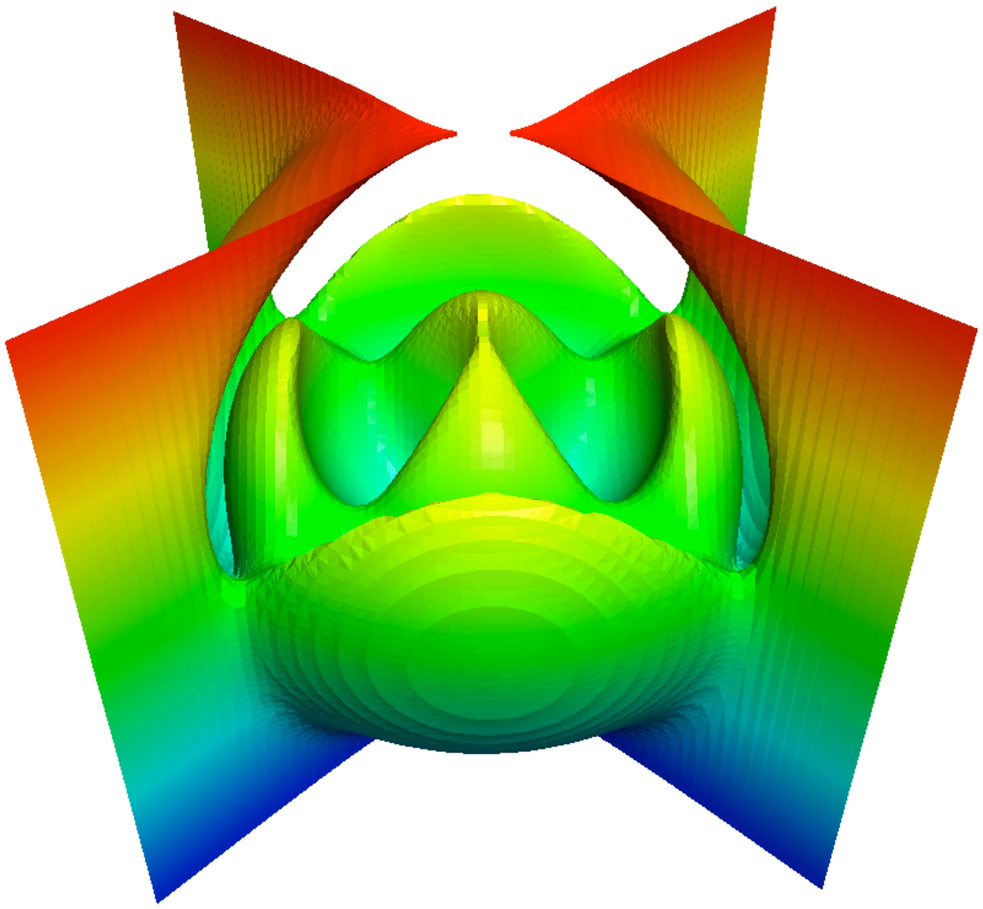


Figure 4.4: Zernike polynomial, $V_{31} = \rho^2(-10 + 60\rho^2 - 105\rho^4 + 56\rho^6)\cos 2\theta$

Both Legendre and Zernike moments offer complete shape description using a finite set of moments. However, they both suffer from some significant deficiencies which make implementation problematic.

The polynomials are defined to be orthogonal inside the domain $[-1, 1]$ and $0 \leq \rho \leq 1; 0 \leq \theta \leq 2\pi$ for Legendre and Zernike respectively. Neither of these is suited to the digital image domain, which is $[0, N - 1]$ for square images. Therefore before calculating these moments the image must be transformed into the moment coordinate space, which compromises the original shape especially in the case of Zernike moments. A secondary

Name	Notation	a	b	$w(x)$
Tchebichef	$t_n(x)$	0	$N-1$	1
Krawtchouk	$k_n^{(p)}(x)$	0	N	$p^x(i-p)^{N-x}\binom{N}{x}$
Charlier	$c_n^{(a)}(x)$	0	α	$\frac{e^{-a}a^x}{x!}$
Meixner	$m_n^{(c,\zeta)}(x)$	0	α	$\frac{c^x(\epsilon)_x}{x!}$
Hahn	$h_n^{(\epsilon,\phi,\zeta)}(x)$	0	$N-1$	$\frac{(\epsilon)_x(\phi)_x}{x!(\zeta)_x}$

Table 4.1: Properties of some important discrete orthogonal polynomials. $p, \alpha, c, \epsilon, \phi, \zeta$ are parameters associated with the respective polynomials, n denotes the degree [3].

limitation of the Zernike and Legendre schemes is their continuous definition, which results in a further approximation by calculating integrals using summations, thereby introducing numerical errors. These factors have led to the emergence of discrete orthogonal moments, notably Tchebichef Moments proposed by Mukundan [41].

4.5 Discrete Orthogonal Moments

When considering a discrete orthogonal system, $\{f_n(i)\}$, where $a \leq i \leq b$, the orthogonality property can be written

$$\sum_{i=a}^{i=b} w(i) f_m(i) f_n(i) = \rho(n, a, b)\delta_{mn} \tag{4.15}$$

as stated by Mukundan [41]. Where $w(i)$ is the weighting function, $\rho(\cdot)$ is the squared norm and δ_{mn} is the Kronecker delta, equal to 1 if $i = j$ and 0 if $i \neq j$. The properties of five discrete orthogonal polynomials are shown in Table 4.1.

The simplest system is the Tchebichef polynomial, as it has unit weight, and ideally fits the digital image domain, $[0, N-1]$. Therefore we may present the definition of discrete orthogonal moments of an image $f(x, y)$ using the following theorem: If $\{t_n(x)\}$ is a set of discrete orthogonal polynomials with unit weight, satisfying the condition

$$\sum_{x=0}^{N-1} t_p(x)t_q(x) = \rho(p,N)\delta_{pq}, \quad 0 \le p,q \le N-1 \tag{4.16}$$

then any bounded function $f(x,y), 0 \le x,y \le N-1$, has the following polynomial representation in terms of the functions $t_p(x)$

$$f(x,y) = \sum_{p=0}^{N-1}\sum_{q=0}^{N-1} T_{pq}t_p(x)t_q(y) \tag{4.17}$$

where the coefficients T_{pq} are given by

$$T_{pq} = \frac{1}{\rho(p,N)\rho(q,N)} \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} t_p(x)t_q(y)f(x,y) \tag{4.18}$$

$$p,q = 0,1,2,\ldots,N-1$$

We should observe that the above theorem defines the moment generating function Eq (4.18) using the Tchebichef basis set $\{t_n(x)\}$, and Eq (4.17) is its corresponding inverse function. The moment function Eq (4.18) eliminates the approximation of continuous integrals and does not require any coordinate transformations, which results in significantly reduced numerical errors and consequently enables perfect image reconstruction.

The discrete Tchebichef polynomials [3] are defined as

$$t_n(x) = (1-N)_n\, {}_3F_2(-n,-x,1+n;1,1-N;1),$$
$$n,x,y = 0,1,2,\ldots,N-1$$

where $(a)_k$ is the Pochhammer symbol $(a)_k = a(a+1)(a+2)\cdots(a+k+1)$ and ${}_3F_2(\cdot)$ is the generalized hypergeometric function

$${}_3F_2(a_1,a_2,a_3;b_1,b_2;z) = \sum_{k=0}^{\infty} \frac{(a_1)_k(a_2)_k(a_3)_k}{(b_1)_k(b_2)_k}\frac{z^k}{k!}$$

Mukundan [41] states that the above definitions may be rewritten as

$$t_n(x) = n! \sum_{k=0}^{n} (-1)^{n-k} \binom{N-1-k}{n-k} \binom{n+k}{n} \binom{x}{k} \tag{4.19}$$

and that the polynomials satisfy the orthogonality property (4.16), with

$$\rho(n,N) = (2n)! \binom{N+n}{2n+1}, \quad n = 0, 1, \dots, N-1 \tag{4.20}$$

and have the following recurrence relation

$$(n+1)t_{n+1}(x) - (2n+1)(2x-N+1)t_n(x) + n(N^2-n^2)t_{n-1}(x) = 0, \tag{4.21}$$
$$n = 1, 2, \dots, N-1$$

This set of polynomials is not ideally suited to image moments however, as the value of $t_n(x)$ grows as N^n, so that the moment T_{pq} given in Eq (4.18) grows as $N^{-(p+q)}$, which will result in biased high order image moment terms. Therefore we scale the polynomials using

$$\tilde{t}_n(x) = \frac{t_n(x)}{\beta(n,N)} \tag{4.22}$$

with a scaled squared norm

$$\tilde{\rho}(n,N) = \frac{\rho(n,N)}{\beta(n,N)^2} \tag{4.23}$$

which replaces $\rho(n,N)$ in Eq (4.18). The simplest choice for $\beta(n,N)$ is

$$\beta(n,N) = N^n \tag{4.24}$$

so that

$$\tilde{\rho}(n,N) = \frac{N\left(1-\frac{1}{N^2}\right)\left(1-\frac{2^2}{N^2}\right)\cdots\left(1-\frac{n^2}{N^2}\right)}{2n+1} \tag{4.25}$$
$$n = 0, 1, \dots, N-1$$

These scaled polynomials may be defined, and calculated, using the following recurrence relation

$$\tilde{t}_n(x) = \frac{(2n-1)\tilde{t}_1(x)\tilde{t}_{n-1}(x) - (n-1)\left(1 - \frac{(n-1)^2}{N^2}\tilde{t}_{n-2}(x)\right)}{n} \tag{4.26}$$

$$n = 2, 3, \ldots, N-1$$

where

$$\tilde{t}_0(x) = 1$$
$$\tilde{t}_1(x) = \frac{2x+1-N}{N}$$

However Mukundan highlights [43] that in situations where the image size is large, $N \to 200$, and the polynomial degree approaches N, the scaled squared norm, Eq (4.25) tends toward zero, which results in an undefined T_{pq} due to a division by zero as seen in Eq (4.18). This motivates the development of *orthonormal* Tchebichef moments.

4.6 Orthonormal Tchebichef Moments

Orthonormal Tchebichef moments are based on an orthonormal variant of the Tchebichef polynomials which can be defined by using an appropriate scale factor, modifying Eq (4.24) to

$$\beta(n, N) = \sqrt{\frac{N(N^2-1)(N^2-2^2)\cdots(N^2-n^2)}{2n+1}} \tag{4.27}$$

These orthonomal polynomials are denoted as $\{\hat{t}_n(x)\}$, and the recurrence relation given in Eq (4.26) is changed to

$$\hat{t}_n(x) = \hat{t}_{n-1}(x)\,(x\alpha_1 + \alpha_2) + \alpha_3\hat{t}_{n-2}(x) \tag{4.28}$$

$$n = 2, 3, \ldots, N-1; \quad x = 0, 1, \ldots, N-1$$

where

$$\alpha_1 = \frac{2}{n}\sqrt{\frac{4n^2-1}{N^2-n^2}} \tag{4.29a}$$

$$\alpha_2 = \frac{1-N}{n}\sqrt{\frac{4n^2-1}{N^2-n^2}} \tag{4.29b}$$

$$\alpha_3 = \frac{n-1}{n}\sqrt{\frac{2n+1}{2n-3}}\sqrt{\frac{N^2-(n-1)^2}{N^2-n^2}} \tag{4.29c}$$

with initial values

$$\hat{t}_0(x) = \frac{1}{\sqrt{N}} \tag{4.30a}$$

$$\hat{t}_1(x) = (2x+1-N)\sqrt{\frac{3}{N(N^2-1)}} \tag{4.30b}$$

Figure 4.5 shows the first 6 orthonormal polynomials for $N = 64$. The squared norm of $\{\hat{t}_n(x)\}$ is now denoted

$$\hat{\rho}(n, N) = \sum_{x=0}^{N-1}\{\hat{t}_n(x)\}^2 = 1 \tag{4.31}$$

so that the moment generating function, Eq (4.18) is now

$$T_{pq} = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1}\hat{t}_p(x)\hat{t}_q(y)f(x,y) \tag{4.32}$$

$$p, q = 0, 1, \ldots, N-1$$

The framework provides an implementation of orthonormal Tchebichef polynomials given the above definitions and based on the recurrence schemes in Equations (4.28), (4.29) and (4.30), and the specifics implementation is provided in Appendix B.3.

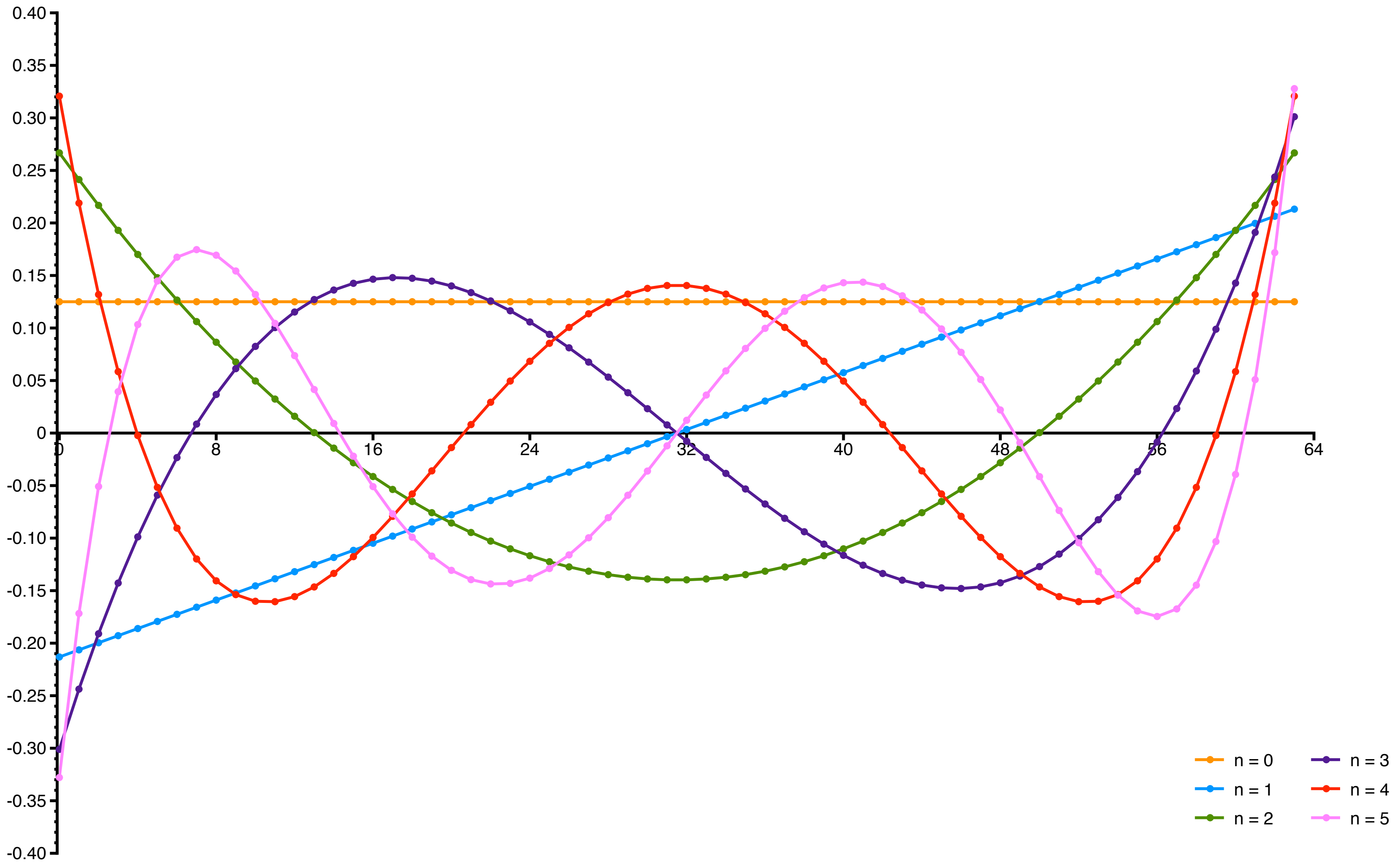


Figure 4.5: Orthonormal Tchebichef Polynomials, $\hat{t}_n(x),\ n = 0, 1, \ldots, 5; N = 64$

4.6.1 Moment Invariance

Mukundan developed rotational invariant Tchebichef moments [42, 44] by using the single dimensional Tchebichef polynomial introduced previously with a circular function. Given the discrete image domain of square size N, the parameter r, the radius of the circle, varies from 0 to $N/2$, and θ varies from 0 to 2π in n discrete steps. From this it follows that the moments will not be computed over the entire square image, which will adversely impact the description and reconstruction. Additionally this method requires the co-ordinate transformation of the square image to a circle, which was a principle motivator in the development of discrete moments in the first instance.

For this research radial or scale invariance of shapes is not required, as these are inherent to the very properties we wish to examine. However, we do require translation invariance which is provided as a natural consequence of the image segmentation process, as discussed in Section 6.1. Figure 4.6 shows the square areas over which the Tchebichef moments are computed for the detected shapes. Because we access the pixels using the origin of each square, rather than the origin of the whole image, each shape is effectively removed from its context, and henceforth considered in isolation.

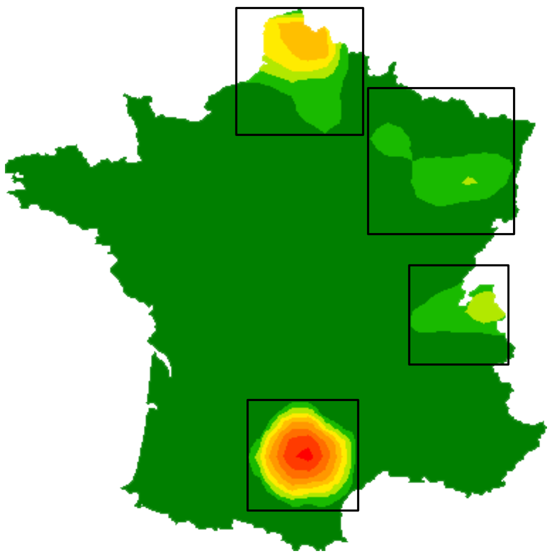


Figure 4.6: Computing Tchebichef moments of shapes removed from their background automatically ensures translation invariance.

4.6.2 Shape Reconstruction

It is possible to reconstruct an image from a set of image moments in a manner that is similar to computing the reverse Fourier transform compared to the forward transform. Eq (4.33) shows the formula for image reconstruction.

$$\tilde{f}(x,y) = \sum_{p=0}^{M}\sum_{q=0}^{M} \tilde{t}_p(x)\tilde{t}_q(y)T_{pq} \tag{4.33}$$

where $\tilde{f}(x,y)$ is the reconstructed image, and M is the maximum moment order used for the reconstruction. For reconstruction from the complete set, $M = N-1$. Figure 4.7 plots the mean squared error (MSE) in the image reconstruction, computed as

$$\epsilon = \sqrt{\sum_{y=0}^{N-1}\sum_{x=0}^{N-1}(f(x,y) - \tilde{f}(x,y))^2} \tag{4.34}$$

where $f(x,y)$ is the original image. The MSE reduces significantly as the reconstructed moment order is initially increased. This is expected as the low order moments contain the overall image properties. The MSE of the reconstruction from the complete set of moments is very low: 1.56% of that based solely on the zeroth order. However, theoretically its absolute value should be zero, yet in practice this is not the case. This is a consequence of the limited numerical precision in the implementation of the recurrence schemes used to compute the Tchebichef polynomials. Figure 4.7 on page 34 plots the reduction in MSE for image reconstruction for a 256 pixel square image shown in Figure 4.8.

Figure 4.8, on page 35, shows the results of image reconstruction based on varying degrees of a complete set of moments. Figure 4.8a is the original 256 pixel square grayscale image. Orthonormal Tchebichef moments have been computed over the whole image. Figures 4.8b through to 4.8e show the result of image reconstruction from a limited set of moments. The images have been reconstructed using moments from order zero up to $n = \{50, 100, 150, 200\}$ respectively. Finally 4.8f shows the reconstruction from the complete set. The difference between the final two reconstructions is not discernable to the human eye which is a result of the MSE reduction as the number of moments tends toward N.

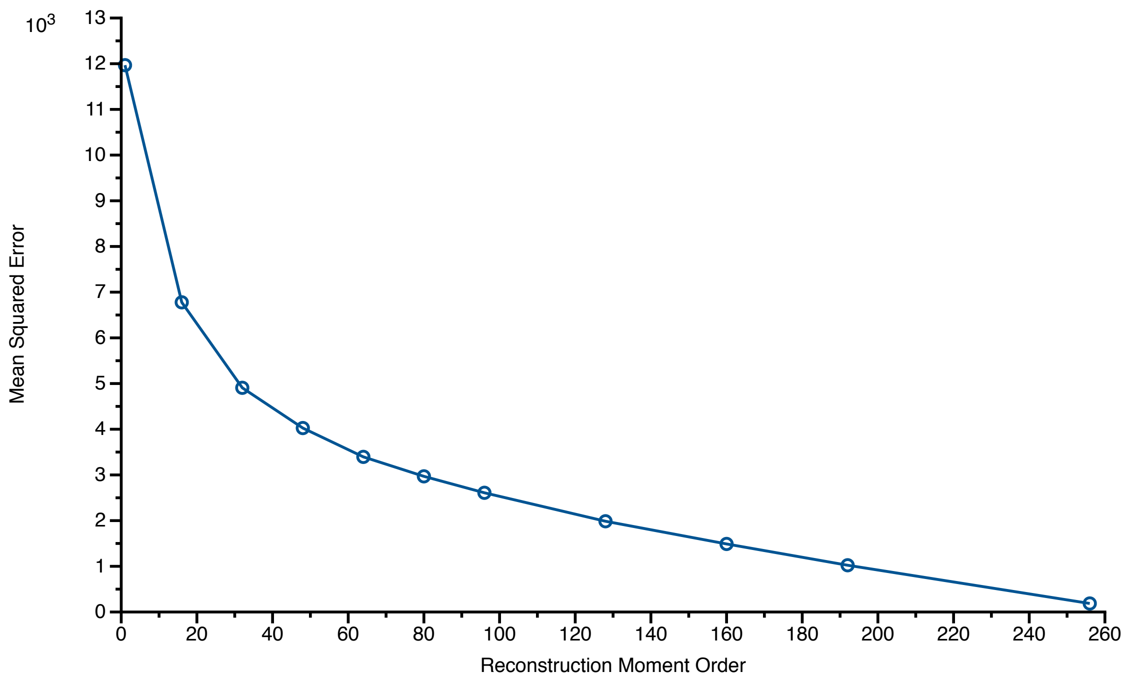


Figure 4.7: Mean squared error of image reconstruction using all moments from order zero up to and including the x-axis data point.

4.6.3 Properties of Tchebichef Moments

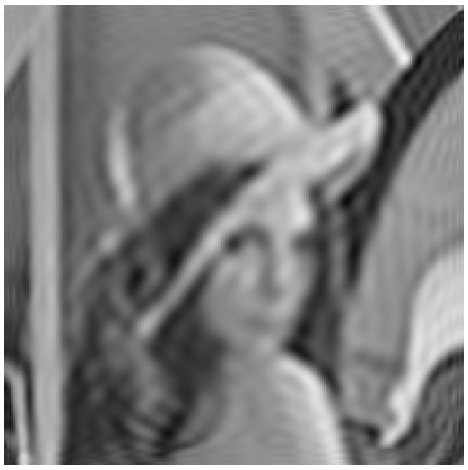
Mukundan's research on Tchebichef Moments [41, 43] focus on the development of the technique and computational aspects. These proposed implementation aspects minimise the error between original images and those reconstructed from their moments. Further analysis of the descriptive properties of Tchebichef Moments, unrelated to reconstruction error, has not been published. As this research is employing Tchebichef Moments for their descriptive power, and less for their image reconstruction potential, further analysis of the moments was required.

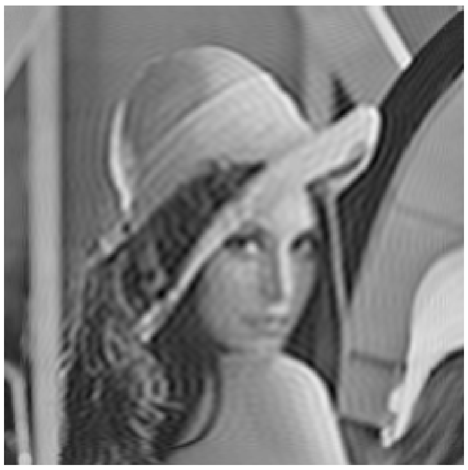
The ordinary, or Euclidean distance measure can be used to calculate the distance between two points in n-dimensional space.

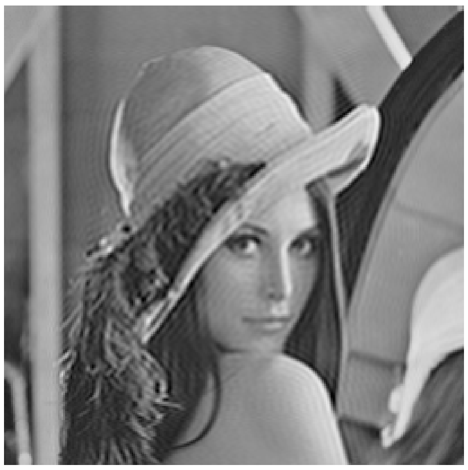
$$d = \sqrt{\sum_p \sum_q (A_{pq} - B_{pq})^2} \tag{4.35}$$

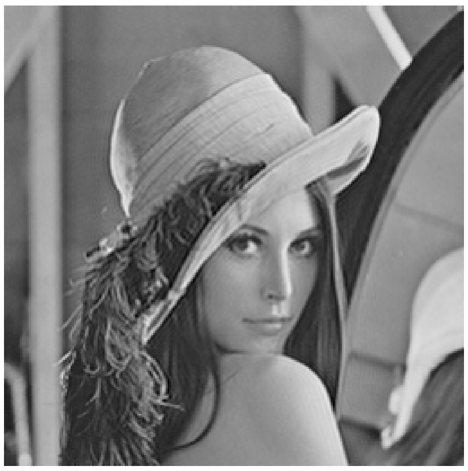
where A_{pq} and B_{pq} are Tchebichef moments of two equal sized images. We can use the Euclidean distance to gauge the effectiveness of the descriptive power of the moments.

(a) Original Image, $N = 256$


(b) Reconstruction $\forall \; p, q < 50$, MSE=3952.3


(c) Reconstruction $\forall \; p, q < 100$, MSE=2535.7


(d) Reconstruction $\forall \; p, q < 150$, MSE=1649.1


(e) Reconstruction $\forall \; p, q < 200$, MSE=923.6

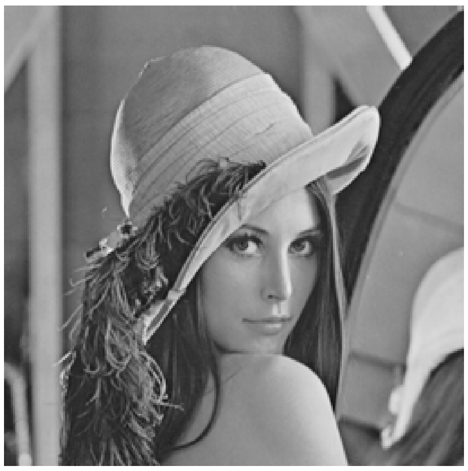

(f) Reconstruction $\forall \; p, q < 256$, MSE=186.6

Figure 4.8: Image reconstruction from ever increasing numbers of Tchebichef Moments.

Each image is represented as a point in an N^2-dimension Tchebichef moment domain. The shorter the Euclidean distance between two points indicates the greatest similarity between the images. A zero distance is the same point a result of the same input image.

Therefore it is important to understand the unit response of the Tchebichef moments. The linearity of the method may be proved as follows, by considering an image $f(x, y)$, the Tchebichef moments are defined as

$$T_{p,q} = \sum_x \sum_y \hat{t}_p(x)\hat{t}_q(y)f(x, y)$$

for $p, q, x, y = 0, 1, 2, \ldots, N - 1$ and all summations over x, y, p, q are from $0 \to (N - 1)$. Substituting $f(x, y) = I + g(x, y)$ where I is a fixed intensity over the image domain, equivalent to a unit impulse:

$$\begin{aligned}
T_{p,q} &= \sum_x \sum_y \hat{t}_p(x)\hat{t}_q(y) \left(I + g(x, y)\right) \\
&= I \sum_x \sum_y \hat{t}_p(x)\hat{t}_q(y) + \sum_x \sum_y \hat{t}_p(x)\hat{t}_q(y)g(x, y) \\
&= T_{pq}\left\{I\right\} + T_{pq}\left\{g(x, y)\right\}
\end{aligned}$$

Therefore given two images, $a(x, y)$ and $b(x, y)$ where $b(x, y) = a(x, y) + I$

$$T_{pq}\left\{I\right\} = T_{pq}\left\{b(x, y)\right\} - T_{pq}\left\{a(x, y)\right\}$$

and the Euclidean distance measure

$$\begin{aligned}
d &= \sqrt{\sum_p \sum_q \left(T_{pq}\left\{b(x,y)\right\} - T_{pq}\left\{a(x,y)\right\}\right)^2} \\
&= \sqrt{\sum_p \sum_q \left(T_{pq}\left\{I\right\}\right)^2} \\
&= \sqrt{\sum_p \sum_q \left(I \sum_x \sum_y \hat{t}_p(x)\hat{t}_q(y)\right)^2} \\
&= \sqrt{I^2}\sqrt{\sum_p \sum_q \left(\sum_x \sum_y \hat{t}_p(x)\hat{t}_q(y)\right)^2}
\end{aligned}$$

This may be reduced further using the orthonormal properties of the Tchebichef polynomial.

$$\sum_{n=1}^{N-1}\sum_{x=0}^{N-1} \hat{t}_n(x) = 0 \tag{4.36a}$$

$$\sum_{x=0}^{N-1} \hat{t}_0(x) = \sqrt{N} \tag{4.36b}$$

$$\begin{aligned}
d &= \sqrt{I^2}\sqrt{\sum_p \sum_q \left(\sum_x \sum_y \hat{t}_p(x)\hat{t}_q(y)\right)^2} \\
&= \sqrt{I^2}\sqrt{\left(\sum_x \hat{t}_0(x) \sum_y \hat{t}_0(y)\right)^2} \\
&= I\sqrt{\left(\sqrt{N}\sqrt{N}\right)^2} \\
&= IN
\end{aligned}$$

Therefore the distance between any two equal sized images is proportional to the image size. This can be verified by experiment using the Lenna image from Figure 4.8a. By successively increasing the zero point in the image and computing the Euclidean distance between each modified image and the original we can simulate the images $a(x, y)$ and $b(x, y)$ from the algebra above. The linear response is plotted in Figure 4.9.

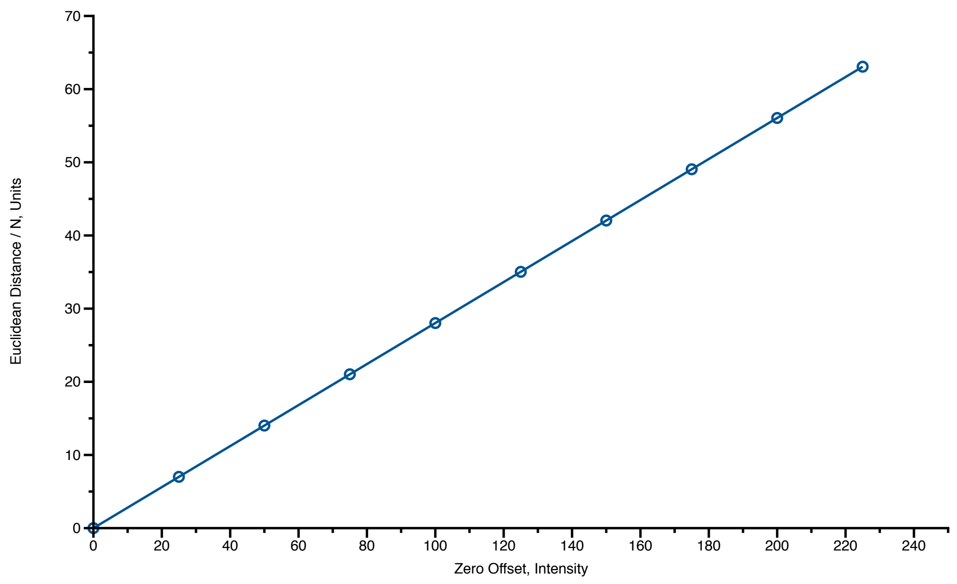


Figure 4.9: Demonstrating the linear response of Tchebichef Moments.

The linearity of the Tchebichef Moment is an important property which allows sets of moments of different images to be compared, after a common baseline threshold, T has been identified. This is discussed further and in greater detail in the appropriate context in Section 6.1.

Chapter 5

Covariate Interpolation

Covariates are variables that have an unknown influence on the phenomenon under study. The exact nature of such an influence is ill-defined and rarely known. Additionally the covariates that influence the phenomenon are generally unknown. Therefore the ability to explore a phenomenon's *covariate space*, by which we mean: determine the nature and extent of the relationships between covariates and the study phenomenon, is of increasing importance in a wide range of research areas. For example, it is accepted that cigarettes may cause lung cancer, but it is not possible to quantitatively determine how likely it is to get lung cancer through smoking. By considering the various covariates that might influence the nation-wide occurrence of lung cancer, such as price of cigarettes, traffic congestion, quality of life, population density and the extent of urban areas, it would be possible to measure the impact of a single covariate on the study phenomenon.

In general, phenomena of interest will be space-time varying, so it follows that likely covariates will also be space-time varying. Therefore any framework through which we might analyse the covariate space must be founded on spatiotemporal analysis techniques, such as shape description discussed in Chapter 4 and spatial statistics which is discussed in this chapter.

After introducing the idea of *implicit covariates* in the next section, the remainder of this chapter is devoted to examining the operations required to register the covariate data onto the spatial domain used by the Influenza data. These techniques may be applied to any spatial data sets and are not tuned to those used in this research. This is a necessary process to ensure that spatial measurements are grounded in the same coordinate system and pixel coordinates in one data set equate to same real world location in other data sets. The covariate data used are available as raw numbers and associated geographical locations. Therefore the registration process consists of two steps, a coordinate transform and then interpolation. The chapter is broadly organised into two parts which correspond to the these steps. Firstly, Section 5.2 provides further detail about the principle source of covariate data used in this research, the MIDAS weather observations first introduced in Section 3.2, with a thorough discussion of mapping geographical coordinates to Cartesian coordinates using a suitable *projection* and *datum*. Section 5.3 discusses some different interpolation methods and justifies the reasoning behind the use of Kriging in this research. Section 5.4 details the spatial modelling of the weather data using a Semivariogram which is a necessary component of Kriging interpolation. Section 5.5 develops the Kriging interpolation system, and finally Section 5.6 shows the result of its implementation. Additionally the specific implementation is included in Appendix B.4.

5.1 Implicit Covariates

The type of covariates that require further analysis often depends on the problem domain. In biometrics, covariate factors often relate to the subject's clothing, accessories, and the time lapse between the subject's registration and probe data. Bouchrika [6] identifies covariates of clothing, footwear, walking speed and carrying conditions[1] as factors of interest in the identification of people from their gait. These are discrete factors, which are treated as well defined attributes of the subject, and require a specific analysis method. For example, a subject is either wearing a hat or not; the covariate is true or false. This research does not focus on these types of covariates, as their analysis cannot be generalised because the a priori assumptions made about each factor is both significant and extensive. Instead we will focus on spatially sampled data-sets. This is primarily because we aim to analyse the re-

[1]The subject may be carrying any combination of briefcase or rucksack.

lationship between generic, seemingly unconnected phenomena, yet which is sampled over a large spatial domain.

There are examples of using implicit covariates to analyse a secondary phenomenon. In [67] Woodworth utilises annual mean atmospheric pressure at sea level as a covariate to improve an estimate of mean sea level (MSL) trends over time, and states that incorporating covariates is “a valuable tool in understanding part of the variability of MSL”, although “even if a complete meteorological modelling data set were available, it would not be able to account for all MSL variability”.

5.2 MIDAS Weather Observations

The principal source of covariate data used throughout this research, global weather observations, was introduced previously in Section 3.2. The remainder of this chapter discusses the necessary treatment of this data set for its use in the analysis of the Influenza data.

5.2.1 Geodesy: Datums & Projections

The weather stations are referenced by a unique WHO identifier which has an associated latitude, longitude and elevation. In generating the map shown in Figure 3.2, Google’s application converts the geographical coordinates of the stations into plane coordinates to position markers on the map. For this research, the same conversion is necessary as the Influenza surveillance is presented as a digital image map of France. Therefore the minimal area δA available is an image pixel, which is accessed using a Cartesian index.

This process is called *projection* and is the result of a simple concept, namely that the surface of a sphere (or any 3 dimensional object) cannot be perfectly transposed onto a 2 dimensional surface. The problem of mapping is complicated further in that the Earth is not an exact sphere or even an ellipsoid, hence a model representing the shape of the earth, called a *datum*, is used in conjunction with a specific map projection. There are generally three stages to converting such coordinates:

1. Select a datum. This is a model approximating the shape of the earth, which will either be a sphere or an ellipsoid. As the earth's shape is irregular, information is lost here.
2. Select a type of projection to transform geographic coordinates into plane coordinates. This is also a mathematical model which will have a number of parameters such as a false origin, standard parallels, and offsets all which vary depending on what area of the globe is being considered. Typically governments define a projection to provide minimal distortion over their territory.
3. Reduction of the scale.

Google Maps uses the Universal Transverse Mercator (UTM) system of projections, which offers less distortion for latitudes near the equator. Most counties choose a projection specifically tailored to minimises the distortion of their land mass for their national maps. The Institut Geographical National (IGN) of France uses the Lambert II conformal conic projection (CCP), which essentially superimposes a cone over the Earth with two reference parallels secant to the globe. Therefore distortion is minimal along the standard parallels, and increases further away from the parallels. Conformal map projections, preserve angles locally i.e. straight lines on such a map shows the true bearing between two points. Therefore in order to convert the geographical coordinates (latitude and longitude) of the weather stations into northing and easting coordinates we need to know the parameters of the datum and the projection, which are the given in Table 5.1.

Semi-major axis, a	6378249.145m
Semi-minor axis, b	6356514.967m
Flattening, $f = (a - b)/a$	1/293.46
Eccentricity, e	$\sqrt{2f - f^2}$

(a) Datum parameters

False Origin	46° 48'N latitude, 0° Paris longitude
1st Parallel, ϕ_1	45° 53'56,108"
2nd Parallel, ϕ_2	47° 41'45,652"
False Easting, E_F	600,000m
False Northing, N_F	2,200,000m

(b) Projection parameters

Table 5.1: Lambert II conformal conic projection parameters for France

Given the above we can derive Easting and Northing coordinates from geographical coordinates (ϕ, λ) using standard equations given in [49]. Appendix B.1 briefly explains the implementation of this process.

5.2.2 Transposing between co-ordinate systems

Once the weather stations are referenced by their northing and easting planar coordinates, all computation, such as interpolation discussed in Section 5.5, is performed using this coordinate system. The interpolated surface is rendered into the coordinate system of the phenomenon data by using an image mask which defines the area over which the phenomenon exists. For Influenza this is a square image with a 381×381 pixel map of France. The coordinates of the geographical extremities of the mask are measured from IGN's official maps and Google Earth, and converted into northing and easting coordinates. Finally, the covariate is evenly sampled at 381^2 locations in this coordinate system and the interpolated values create a new map. The values used for this mapping are outlined in Table 5.2.

	Pixel Coordinates	Geographical Planar Coordinates
North	61px	2700.0620224000709km
South	442px	1723.1173496146864km
West	46px	11.7837194579470km
East	427px	1086.6754783444192km

Table 5.2: Mask values for registration of covariate data with influenza data.

5.3 Interpolation Techniques

Interpolation is the procedure of estimating the value of a signal at unknown locations. Spatial interpolation is this procedure extended to 2 or potentially N dimensions. It is a well researched topic in a variety of fields such as mathematics, statistics, geography and image processing. From mathematics *spline interpolation* and its derivative methods such as bicubic and bilinear interpolation have been shown to provide far smoother results over simple nearest neighbour approaches. Indeed bicubic interpolation is widely used in computer graphics applications for scaling images and video. In general however all these approaches aim to achieve the same result, which is to convert point observations to continuous fields. Additionally, these methods are based on the same easily understandable principle, which is that observations close together in space are more likely to have similar values than observations far apart. This tends towards the limit where two observations at exactly the same location in space must have the same value.

Techniques such as bilinear and bicubic interpolation are referred to as deterministic methods as they do not employ any probability theory. They also do not impose any structure on the underlying signal. They effectively perform a polynomial interpolation in multiple directions, and if necessary in a piece-wise fashion. Given four known locations linear interpolation in two directions will provide a polynomial in the form

$$f(x, y) \approx a + bx + cy + dxy \tag{5.1}$$

where in general the number of coefficients will correspond to the number of known data points used in the interpolation process. These methods are used extensively in image processing to resample digital images.

The field of geostatistics focuses on interpolating real world phenomena and as such can apply a-priori knowledge about the attribute being interpolated. Geostatistic methods are stochastic, in that the surface is conceptually one of many that might have been observed and all of which produce the known data points. Another principle difference between geostatistical and deterministic methods is that often the latter are applied in a piece-wise fashion to avoid interpolating using inordinately high order polynomials, and therefore by definition perform local region interpolation. Geostatistical approaches are global interpolators which determine a single model which is used to calculated unknown values at all locations, and as a result will produce smoother surfaces.

A factor that requires consideration is measuring the quality of prediction. Deterministic methods such as polynomial interpolation would require the computation of a second surface using data points not used for the computation of first surface and consider the variance in interpolated values at the same locations. Another drawback of deterministic methods, mentioned earlier is that known properties of the attribute being interpolated cannot be incorporated. These failings motivated George Matheron [39] in the development of optimal techniques for spatial interpolation, known as Kriging.

While the mathematical background of Kriging is discussed in detail in Section 5.5, here the reasoning behind its use in this research is discussed. Geostatistics utilises the notion of a *regionalised variable*, which conceptually fall between random variables and fully deterministic variables. The distribution of geographical phenomena can be described using such a term, for they exhibit spatial continuity yet with small scale randomness. Additionally, it is not possible to sample every location, and data is observed at specific locations, such as weather stations. The size, shape, orientation and spatial arrangement, which is often random, of these locations is called the spatial support and will influence the ability to predict unknown samples.

Regionalised variables use a property called semivariance to model the relationship between any two points a surface. It is half the variance of the difference between all possible points separated by a given distance, usually referred to as h, and is given in Equation (5.2) in Section 5.4. As stated earlier, if two points are close together, i.e. $h \to 0$ it is expected that their values are similar, meaning that the semivariance tends toward zero. As any two points get further apart their values become less similar, and the semivariance will become larger. However, at a given separation distance the two observation feature no similarity, which is a consequence of the phenomenon's attribute having no spatial dependence beyond a particular separation distance, known as the *cut-off*. For this research the cut-off value used for air temperature and dew point covariates was 450km. A Semivariogram is a mathematical model which incorporates the properties of the semivariance; it is used to compute estimates of unknown locations using Kriging. The next section elaborates on the basic concept introduced here and discusses the different types of semivariogram and how they are computed.

Kriging is a method that estimates unknown values of a regionalised variable using a weighted average of the known data points. The weights are derived from the Semivariogram model and sum to one to ensure the estimate is unbiased. Additionally, the weights are chosen to minimise the estimation error.

One disadvantage of using Kriging is that the final interpolated surface is highly dependant on the Semivariogram. Although a smooth surface will always be generated due to the global interpolation approach, any minor change to the Semivariogram's properties will result in different numerical estimates.

In summary, Kriging is used for interpolation because it was specifically designed for the interpolation of regionalised variables such as weather observation which are used extensively in this research.

5.4 Semivariogram

The theoretical variogram, $2\gamma(x_1, x_2)$ is a function which measures the spatial dependence of a random field $Z(\mathbf{x})$. It is defined as the expected squared difference between locations x_1 and x_2. $\gamma(x_1, x_2)$ itself is referred to as the semivariogram.

$$2\gamma(x_1, x_2) = E\left(|Z(x_1) - Z(x_2)|^2\right) \tag{5.2}$$

From Eq (5.2) it is clear that the semivariogram is positively valued and $\gamma(x_1, x_1) = 0$. It is often modeled against a value of h, the separation distance between two points. In this way three parameters are often used to model a variogram, as illustrated in Figure 5.1:

- **nugget**

  n The height of any discontinuity at the origin.

- **range**

  r The separation distance at which spatial dependence is considered negligible (95% of the sill).

- **sill**

  s The value of the semivariogram as the separation distance tends towards infinity.

These properties are used to define various models for semivariograms which are defined in Equations (5.3). When implemented, a model is tailored to a data-set using regression. Throughout this research we employ an exponential semivariogram model, as it is the most simple and fits the weather observations well.

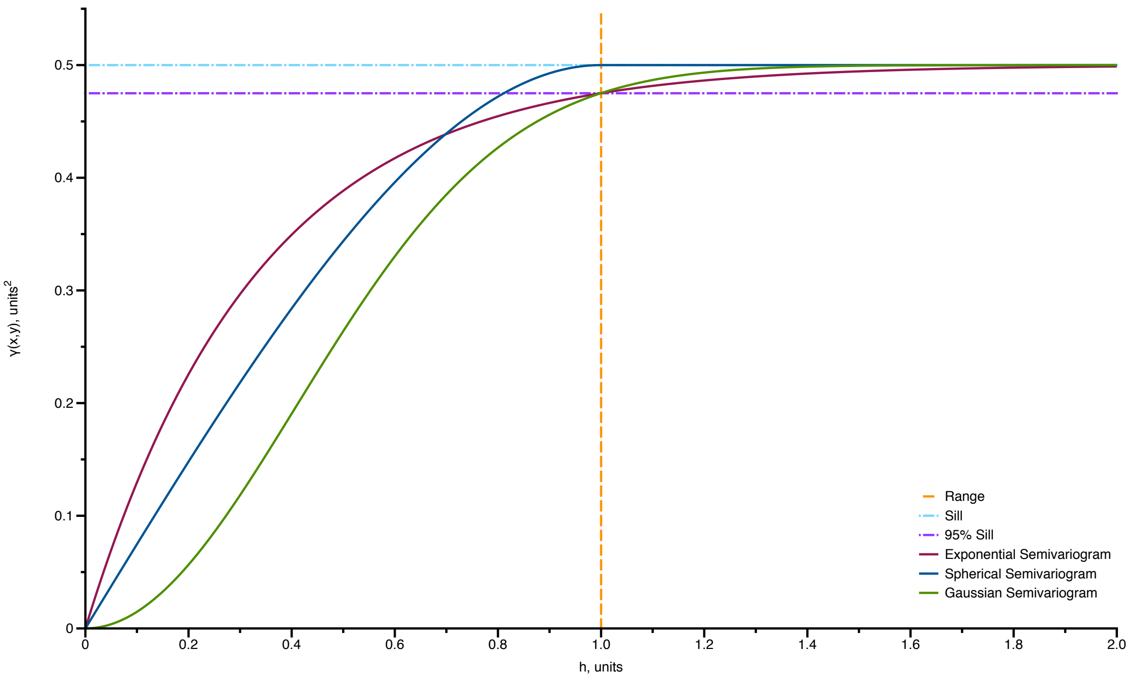


Figure 5.1: An example semivariogram, using the standard models.

$$\text{Spherical:} \qquad \gamma(h) = \begin{cases} s\left(\frac{3h}{2r} - \frac{1}{2}\left(\frac{h^3}{r^3}\right)\right) & h \leq r \\ s & h > r \end{cases} \tag{5.3a}$$

$$\text{Exponential:} \qquad \gamma(h) = n + s\left(1 - \exp\left(-\frac{3h}{r}\right)\right) \tag{5.3b}$$

$$\text{Gaussian:} \qquad \gamma(h) = n + s\left(1 - \exp\left(-\frac{3h^2}{r^2}\right)\right) \tag{5.3c}$$

For a set of geo-referenced values, a semivariogram model can be fitted to the *variogram cloud* using standard regression techniques. The variogram cloud is the term applied to the plot of the semivariance between two observations as a function of their separation distance, once any underlying trends have been removed from the data. Figure 5.2 shows a set of geo-referenced air temperature observations with the calculated trend. The trend is removed from the data before calculating the variogram cloud.

As stated previously in section 5.2, the MIDAS weather observations are recorded three-hourly and stored in conjunction with calculated daily averages. However the Influenza

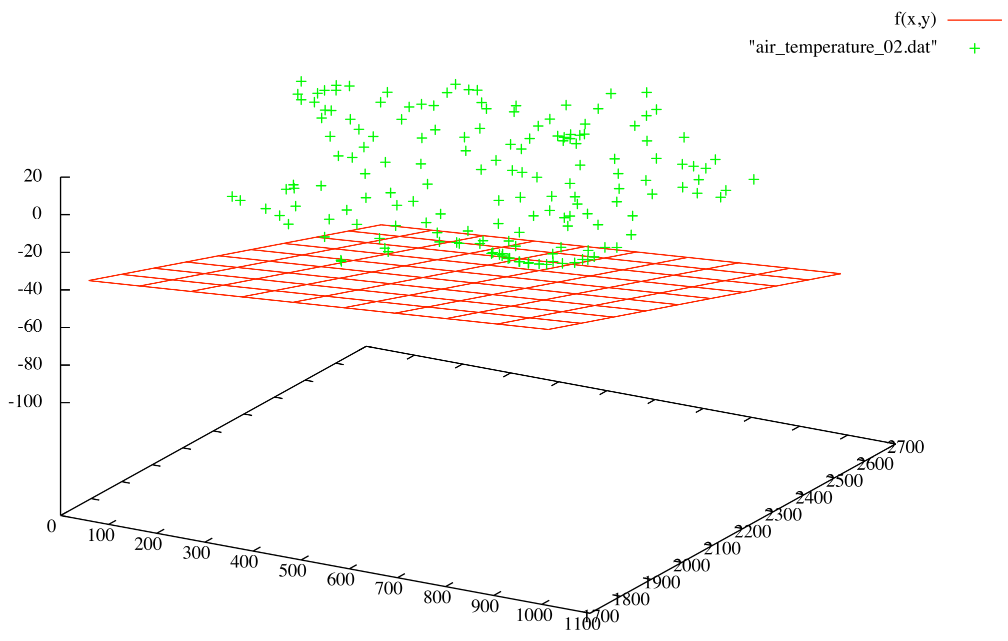


Figure 5.2: Average air temperature across France for 2nd calendar week of the year with trend.

dataset is recorded as weekly counts of reported ILI cases. The covariate data must be sampled using the same temporal support as the measured property of the phenomenon, therefore the weather observations at each location are averaged over weeks. Additionally because weather is seasonal, we need only to calculate models for each week of the calendar year. Hence for each weather covariate it is necessary to calculate 53 models. For every calendar week, the values for every year on record are averaged at each geographical location. This is represented by the points in Figure 5.2 for week two of Air Temperature.

Figure 5.3 shows the variogram cloud of week two Air Temperature once the trend has been removed. The *cutoff* value is the maximum separation distance at which spatial dependence is assumed. For the weather stations deployed in France this is empirically set at 450km. The exponential model is the result of fitting the nugget, sill and range parameters through least squares regression.

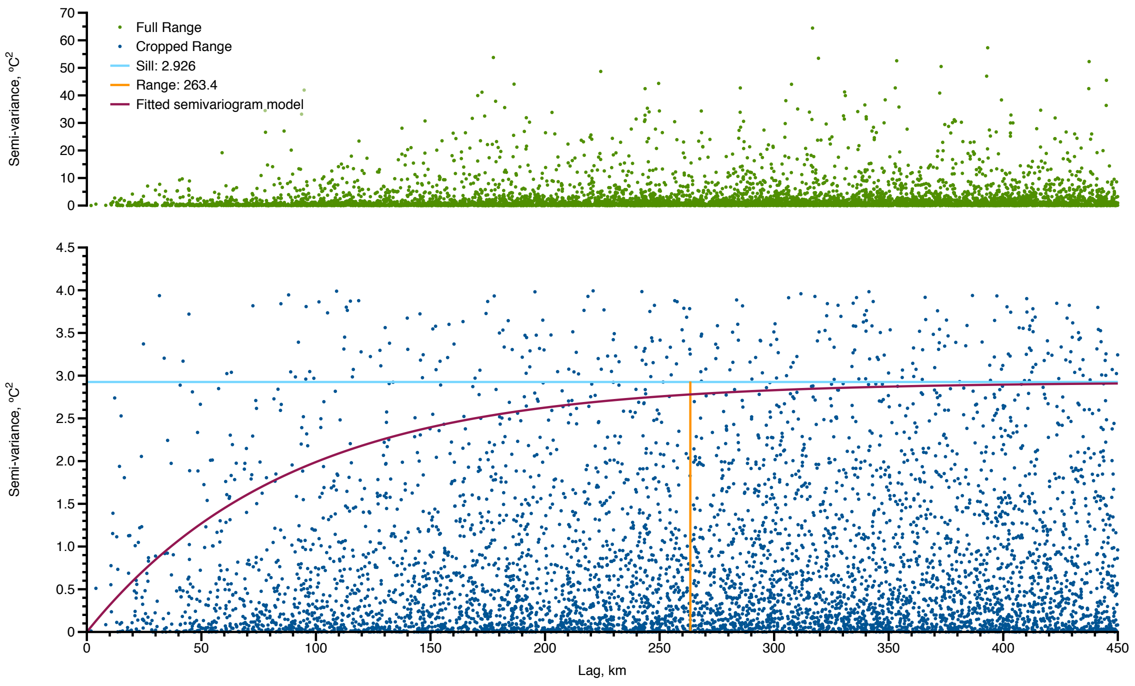


Figure 5.3: Variogram cloud of air temperature across France for 2nd calendar week of the year with fitted exponential semivariogram model.

5.5 Kriging

Kriging is a technique used to interpolate the value of a random field at an unobserved position. The estimated value is the weighted average of the sample observations, biased towards "nearby" observations, such that its error has the minimum variance. Hence it is sometimes referred to as the *Best Linear Unbiased Estimator (BLUE)*, as we define *best* to indicate minimum error variance. The technique was formalised [38] by the French mathematician Georges Matheron, who expounded on the work of Daniel Krige, a prospector on the Witwatersrand. The literature covers this technique in great detail [2, 15], and so will not be repeated. However we will remind the reader of the basic concepts.

There are three main forms of Kriging, as shown in Table 5.3, which transpire from different a-priori knowledge about the regionalised variable under study. Before considering which form of Kriging to use, it is prudent to consider the sampled data to select the most appropriate form. It is reasonable to assume that the data will exhibit a drift term, which indicates that the mean, $m(\mathbf{x})$, is unknown and varying over space. An obvious approach

Kriging Type	Mean
Simple	Constant, known
Ordinary	Constant, unknown
Universal	Varying, unknown

Table 5.3: Three main forms of Kriging

is to subtract an estimate of the mean, $\hat{m}(\mathbf{x})$, and reduce the data to a zero-mean case, suitable for Simple Kriging (SK). This technique is called *detrending*. The problem with such a solution, is that it mixes two estimation methods, firstly the detrending then Kriging. Hence it is very hard to distinguish where the uncertainty in the final estimate lies, as it is impossible to determine how any high frequency effects in the data corrutpted the estimate of the (low frequency) mean. We use Ordinary Kriging (OK) as we have already estimated a semivariogram.

Consider n data values, $z_i = Z(\mathbf{x}_i)$ where $i = 1, 2, \ldots, n$. $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ are the geographical locations of weather stations scattered across France. We intend to estimate the value $z_0 = Z(\mathbf{x}_0)$ where $\mathbf{x}_0$ is an unknown location using the linear estimator show in Eq (5.4)

$$Z^* = \sum_{i=1}^{n} w_i Z(\mathbf{x}_i) + w_0 \tag{5.4}$$

where Z^* denotes the *kriging estimator*, w_0 is a constant and w_i are the weighting factors. We choose them to minimize the error:

$$\mathrm{Var}\left(Z^*(\mathbf{x}_0) - Z(\mathbf{x}_0)\right) \tag{5.5}$$

subject to the condition

$$E\left[Z^*(\mathbf{x}) - Z(\mathbf{x})\right] = 0 \tag{5.6}$$

which stipulates that the estimator is unbiased, as the expected error is zero.

The kriging weights w_i, of OK have a unit sum to ensure unbiasedness,

$$\sum_{i=1}^{n} w_i = 1$$

and are calculated by the *ordinary kriging equation system* [20]:

$$\begin{bmatrix} w_1 \\ \vdots \\ w_n \\ \nu \end{bmatrix} = \begin{bmatrix} \gamma(\mathbf{x}_1, \mathbf{x}_1) & \cdots & \gamma(\mathbf{x}_1, \mathbf{x}_n) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ \gamma(\mathbf{x}_n, \mathbf{x}_1) & \cdots & \gamma(\mathbf{x}_n, \mathbf{x}_n) & 1 \\ 1 & \cdots & 1 & 0 \end{bmatrix}^{-1} \begin{bmatrix} \gamma(\mathbf{x}_1, \mathbf{x}_0) \\ \vdots \\ \gamma(\mathbf{x}_n, \mathbf{x}_0) \\ 1 \end{bmatrix} \tag{5.7}$$

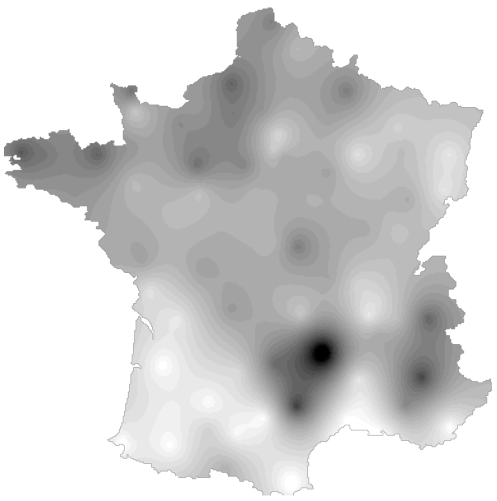
for each unknown location $\mathbf{x}_0$, where the additional parameter ν is a Lagrange multiplier. The interpolation of $Z^*(\mathbf{x}_0)$ may then be calculated as

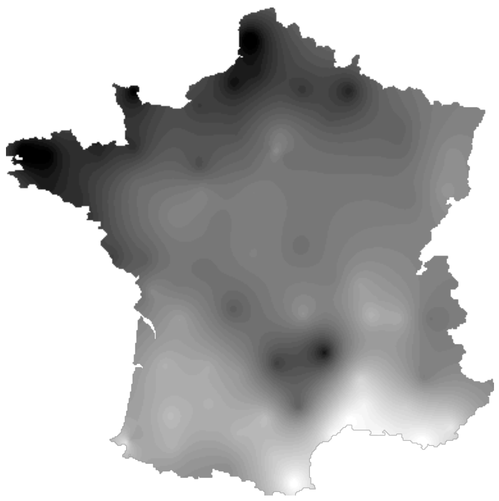
$$Z^*(\mathbf{x}_0) = \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix}' \begin{bmatrix} Z(\mathbf{x}_1) \\ \vdots \\ Z(\mathbf{x}_n) \end{bmatrix} \tag{5.8}$$

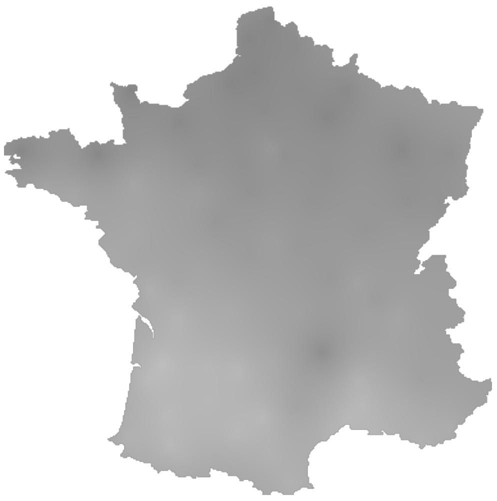
Equation (5.7) and (5.8) must be implemented in order to create smooth maps of weather observations at the same resolution as the study phenomenon. The Kriging matrix shown in Eq (5.7) need only be calculated once for each calendar week. Equation (5.8) must be calculated for every unknown pixel location. Potentially this is a computationally expensive operation, however calculating $\gamma(\mathbf{x}_{1\to n}, \mathbf{x}_0)$ as a vector enables the use of hardware acceleration which greatly improves the computation speed. The implementation developed to calculate semivariograms, and perform Kriging is detailed in Appendix B.4.

5.6 Interpolating Covariate Data

Figure 5.4 shows an evenly resampled covariate using Kriging interpolation. For display purposes the contrast in Figures 5.4a and 5.4b have been maximised, although the framework does not constrain the values to 8-bit integers, but actually stores the real covariate value as signed double precision values.


(a) Air Temperature for week 21, 1995


(b) Air Temperature for week 21, 1996


(c) Air Temperature for week 21, 1997

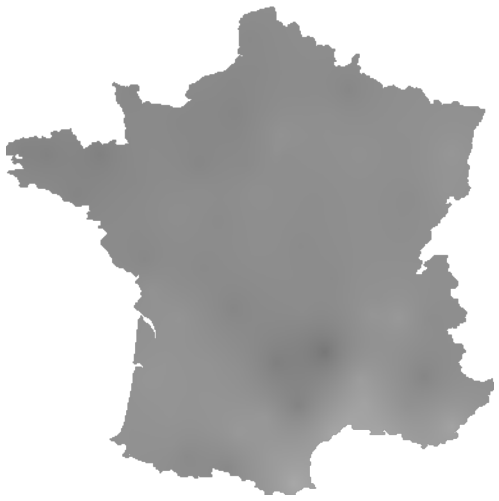

(d) Air Temperature for week 21, 1998

Figure 5.4: Covariate data evenly interpolated from weather stations using semivariogram model shown in Figure 5.3.

Figures 5.4a and 5.4b have been normalised locally, meaning that black pixels are the lowest values for that week, and respectively white pixels are the highest. This demonstrates the peaks and troughs of the covariate. For analysis purposes however, the data is processed as in Figures 5.4c and 5.4d which are normalised over global maxima and minima, and demonstrate that on a weekly scale the range of air temperature values is narrow.

Implementation details for calculating semivariograms and performing Kriging are available in Appendix B.4.

Chapter 6

An Object Based Framework

The subject of this research is phenomena that vary in both space and time, and which are measured as deforming contiguous objects. Additionally the measured response of the phenomenon might be zero, or a non-existent object. This behaviour can therefore be characterised as a *phenomenon event*: namely that an event is a chain of objects, which have been tracked over consecutive time samples from spontaneous appearance to eventual disappearance. Once the objects have been considered in such a form we can aggregate analysis over the whole data-set in order to draw generalised conclusions on the nature of the phenomenon. Before presenting the results of such analysis, we present the methodology used to create phenomenon events and define the nomenclature it introduces. This chapter represents a significant contribution of the research, as it defines an object based framework, and the process by which it is populated.

6.1 Defining Shapes And Objects

The digital images created by the Sentiweb organisation to show influenza activity use a colour key to represent the various reported levels of ILI occurrences. These images are the starting point for this research, which initially converts the colour images to grayscale images using a predefined mapping. While the algorithms perform analysis on grayscale images, this thesis will demonstrate the research using the original colour images. The extent of

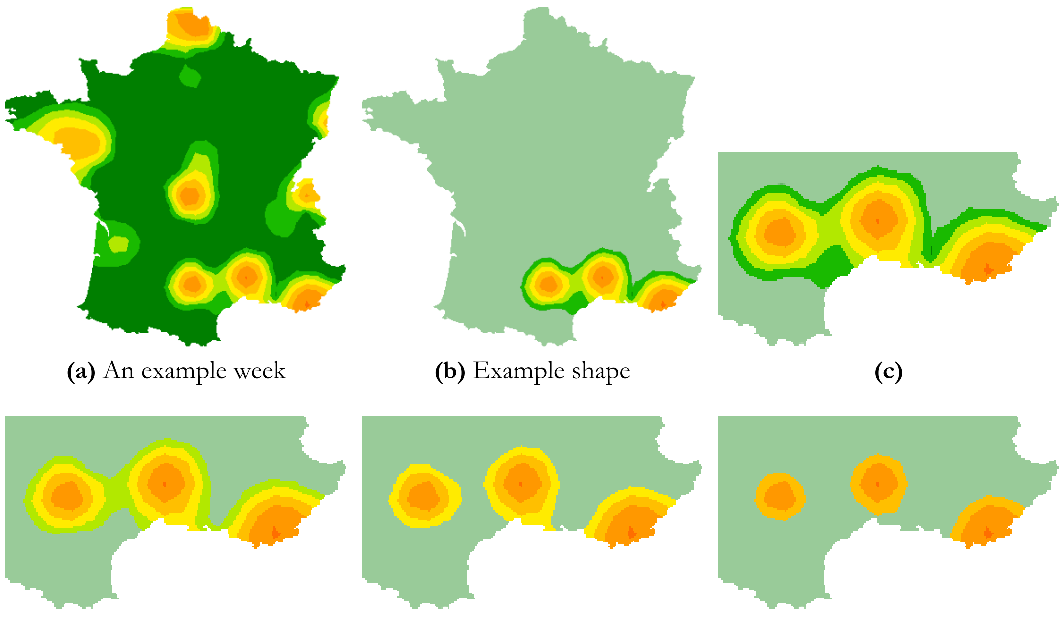

(a) An example week **(b)** Example shape **(c)** **(d)** **(e)** **(f)**

Figure 6.1: Defining a *Phenomenon Shape*.

localised disease outbreaks, or shapes, needs to be defined. This in effect means translating a continuously varying field into a set of spatial entities with well defined boundaries. We consider influenza to exhibit a degree of spatial correlation, and therefore should be analysed with respect to its locality. However this is in contrast to the notion of image moments, which consider the description of a discrete shape, i.e. an arrangement of pixel intensities, *removed* from its location.

Given the example of influenza activity in Figure 6.1, and in particular the activity around Marseilles. We define a *phenomenon shape* as the area of pixels of equal or higher intensity. For every image, we discover all the shapes, by performing a flood fill operation: following adjacent pixels if their intensity is greater than or equal to the current pixel. This is done for every intensity level. For each shape we also store the extent of the shape (its origin and size), in addition to its centre of mass with respect to the whole image, which is computed using centralised moments. This enables the simple construction of a hierarchy of shapes, as illustrated by Figures 6.1c, 6.1d, 6.1e and 6.1f. The shape in Figure 6.1c stores a reference to the shape in Figure 6.1d which in turn stores references to the 3 shapes in Figure 6.1e, and each one of those shapes references the appropriate shape in Figure 6.1f, and this continues

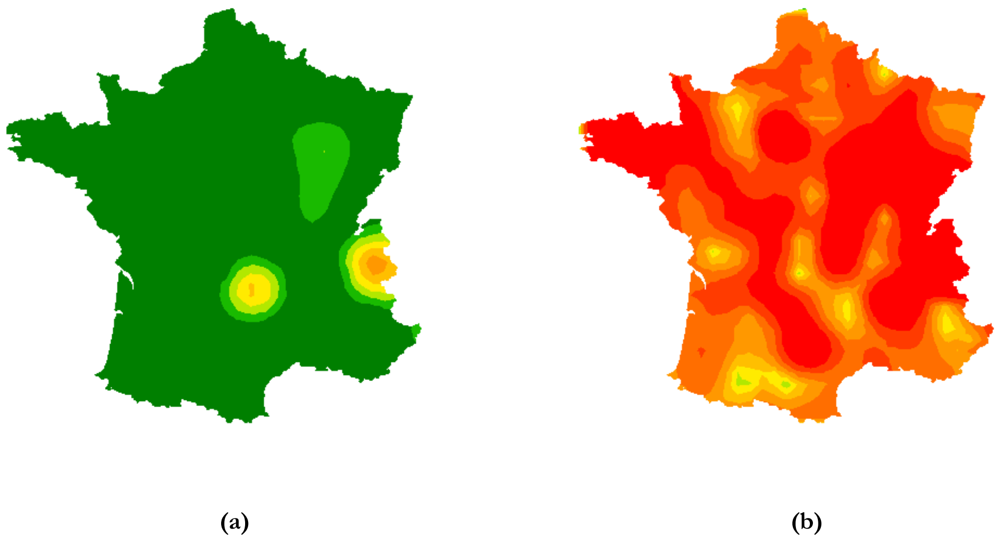


Figure 6.2: Examples of influenza activity

until the highest intensity level.

This process often results in numerous shapes, 39 for this example, however they are not all pertinent or require further processing. Those shapes which are analysed further are referred to as *phenomenon objects*, and are defined to be the shapes at the lowest intensity level which only contain a single branch hierarchy. Using this definition reduces the 39 shapes to ten objects for the example week above. The example shape of Figure 6.1b provides the objects shown in Figure 6.1e.

This simple definition is depicted in Figure 6.3 which depicts a hierarchy of shapes as a graph of connected objects. The motivation for this definition is based the cognitive process through which humans label the importance of the shapes. The Sentiweb images depict surfaces where the elevation indicates ILI activity. This surface features peaks, and humans instinctively rank the volume of each peak, with the largest volume indicating the most ILI activity. The definition used by the framework returns all the shapes in a single image that represent the peaks, which are those at the base of the peak.

There is an abundance of literature devoted to tracking shapes in video and image sequences. In general the problem of tracking objects consists of two stages, segmentation

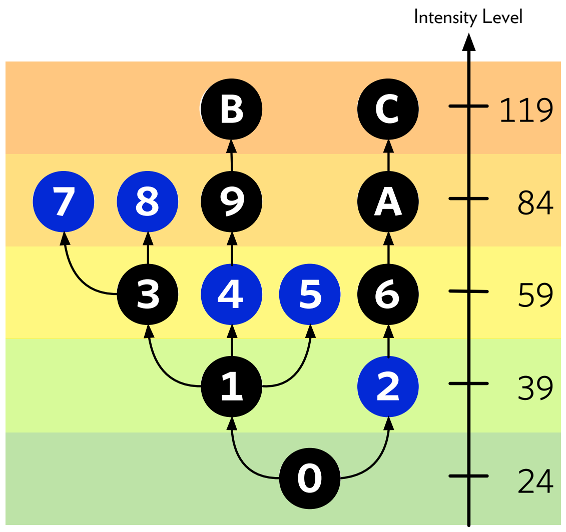


Figure 6.3: Example graph of connected shapes. Each shape is represented as a labelled circle. Blue circles are those at the the lowest intensity which only have a single branch. The coloured bands represent the quantised intensity levels present in the Influenza image data, which have an associated grey scale intensity, shown on the right.

and searching. Image segmentation, partly discussed in Chapter 4 extracts the desired object from its image background. Then subsequent images or video frames are searched for the same object. Model based image segmentation is a rich and diverse research topic, and may be used when the target shape is well defined, and able to be modelled, for example straight lines and basic shapes, faces, and even cars [32]. Shapes representing geographical phenomena, unfortunately, do not reside in this category. They have no guaranteed structure and cannot be imposed onto a model, they are amorphous blobs. While this precludes model based tracking methods from being utilised, simpler image processing methods may be used. A flood fill algorithm is used to locate the shapes, and because the images are computer generated from observed real-world data, their only noise is that introduced by the image format, which is negligible. Once the images are extracted and organised into a hierarchy, we can locate the peaks using graph theory.

This definition of an object allows for the wildly varying cases of influenza activity to be processed using the same methodology. Consider Figure 6.2, in Figure 6.2a it is very simple to see that there are are 3 distinct objects, as the influenza activity is highly localised. Figure 6.2b is the opposite extreme, and both are equally likely to occur throughout the data. In Figure 6.2b the worst areas of disease (the bright red areas) are easily recognisable as three separate shapes despite being elongated and convoluted. This is because the foreground information can be separated easily from the background. This problem is referred to as object segmentation in image processing literature, and techniques to solve it range from the simple: optimal thresholding [51], to the complex: geometric active contours [14]. The above definition of a phenomenon object is a form of optimal thresholding, except that the threshold, which becomes the object's lowest intensity is variable across the image, so that the most significant foreground objects are captured.

In order to compare objects from different thresholds we must remove the effect of this threshold difference. We can consider this mathematically, for if

$$I(x,y) = I_T + \Delta I(x,y)$$

where I_T represents the given threshold, and $\Delta I(x,y)$ is the increase in intensity at point (x,y), then for an image moment M_{pq}

$$\begin{aligned}
M_{pq} &= \sum_x \sum_y \Psi_{pq}(x,y)[I_T + \Delta I(x,y)] \\
M_{pq} &= \sum_x \sum_y \Psi_{pq}(x,y) I_T + \Psi_{pq}(x,y)\Delta I(x,y) \\
M_{pq} &= \underbrace{I_T \sum_x \sum_y \Psi_{pq}(x,y)}_{M_{pq}^T} + \underbrace{\sum_x \sum_y \Psi_{pq}(x,y)\Delta I(x,y)}_{\Delta M_{pq}} \\
\therefore \Delta M_{pq} &= M_{pq} - M_{pq}^T
\end{aligned}$$

Essentially, it is easier to realise that the term $M_{pq}^T = I_T \sum_x \sum_y \Psi_{pq}(x,y)$ is simply an offset. Given the centre of mass and threshold for each object, calculating this offset for all moment orders provides a vector which when applied to the origin of the moment set will render the moments for all objects, in a space, referenced from the same origin. This will allow us to make effective comparisons using the ΔM_{pq} term.

Objects are automatically assigned a label, which is used throughout the remainder of this thesis to specify a particular shape. In general the format of the label is a timestamp, followed by a quantisation level, followed by an shape index. For the influenza images, the timestamp is a four digit year and two digit week (of the year) number. The quantisation level corresponds to the pixel intensity of the shapes, which are 24, 39, 84, 119, 169, 235 and 255. Finally the shape index is the position of the shape in its quantisation level as they are numbered from left to right and top to bottom in the image. Therefore, the shape "1998 39 024 02" means the 2nd shape from level 24, in the 39th week of 1998.

6.2 Spatial Properties of Objects

The discovery of objects from surveillance data provides their initial spatial property: their position and extent. Other spatial properties are provided using the techniques of shape description described in Chapter 4. Computing centralised moments up to the 4th order provide two other basic properties, the object mass and its centre of mass. The centre of mass, a point referenced from the object's origin, the top left corner, may also be placed in the wider context of the sample by adding it to object's position, which places the object's origin within the sample's origin.

6.3 Creating Objects from Raw Data

Given access to raw spatial data that has been randomly sampled it may be evenly interpolated using Kriging as described in Chapter 5. To extract objects from this surface, which given its interpolation will be smooth, some basic image processing must be performed to render an image that is suitable for the object extraction descried earlier in this chapter.

The smooth evenly sampled data must be quantised, for example, the influenza data introduced in Chapter 3 is quantised into 9 bands. And for representation as a digital image, data is quantised into 256 bands. This has been performed to create the image shown in Figure 6.4a, and Figure 6.4b maximizes the contrast. The normalisation process stretches the image histogram, and as a result the average difference between pixel intensities increases thereby generating groups of contiguous pixels at the same intensity. Clearly by decreasing

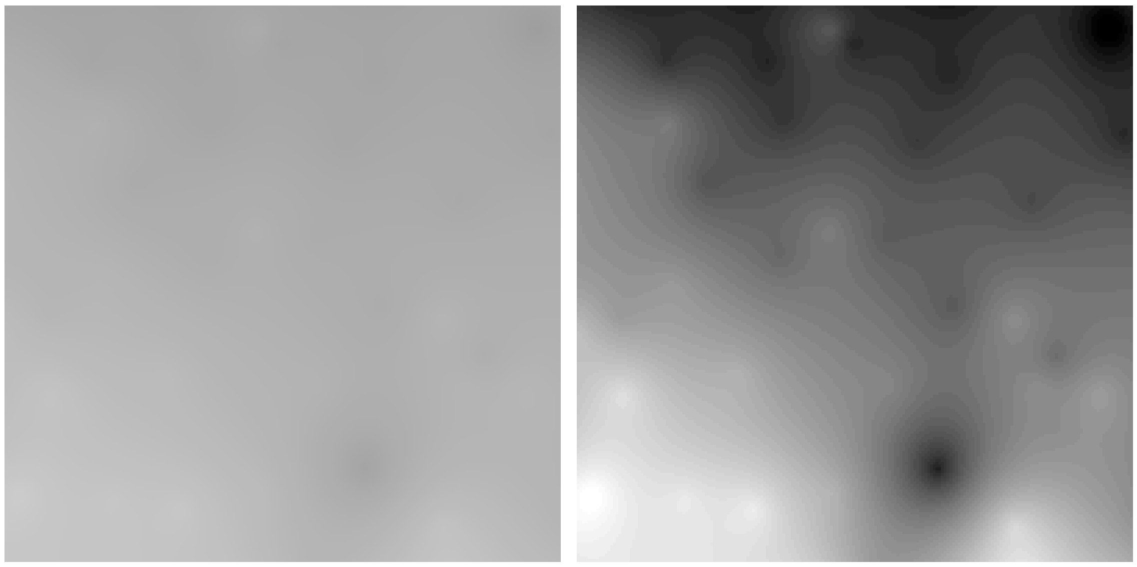

(a) Smooth evenly interpolated data **(b)** After normalization

Figure 6.4: Normalising evenly sampled spatial data

the number of quantisation levels the complexity of gradients within shapes will likewise decrease, however the number of shapes will additionally decrease, which is desirable as it simplifies the object graph, and ultimately reduces the complexity of further processing.

Ultimately, the aim of introducing object based processing, to data that is most commonly processed as a random field, is to encapsulate the spatial deformation using abstract entities. The benefits that this approach affords the researcher will be lost if the objects themselves are too granular and approach pixel size. However, if the quantisation is too course, the data encapsulated by the objects is not an accurate representation of the original raw data.

Maximally Stable Extremal Regions (MSER) is an region detector introduced by Matas *et. al.* [37] which finds the so called *extremal regions*, which is the set of image elements that is closed under continuous transformation of image coordinates and monotonic transformation of image intensities. As such it is suitable to use as a detector to locate image elements between scenes observed from different viewpoints, scales, out-of-plane rotations and with occlusion. The MSER detector searches for regions that are brighter or darker than their surroundings and therefore may be suitable to generate objects directly from the interpolated image. Additionally, an efficient algorithm to track maximum stable extremal regions has recently been developed [21] which features similar tracking techniques proposed in this research, notably forward and backwards tracking to increase stability.

Another method of object segmentation called *graph cuts* [7, 35] would be suitable for extracting objects from the interpolated images. This algorithm works by defining the image as a graph where pixels are connected vertices and the edges are functions of pixel intensity, typically $(I_a - I_b)^2$ where a and b are pixel locations. Additionally the graph may define two vertices to be terminals, and typically the aim is to cut the graph to divide the terminals. Therefore a graph cut is the set of edges which divide the terminals, and the minimum graph cut, is one which provides the maximum flow from one terminal to the other. Graph cuts have a disadvantage being that they only provide binary labelling such as foreground and background. This is a significant failing for this research context, as it is necessary to detect and track multiple generic objects.

6.4 Phenomenon Events

Given the above definitions of phenomenon shapes and objects, it is possible to define a *phenomenon event* as the chain of objects that have been tracked over consecutive time samples from spontaneous appearance to eventual disappearance. We construct events through a process of discovery. Given a sample of influenza, the objects are tracked through consecutive time samples in both directions using the process described in Section 6.5. Failure to find a new object when tracking backwards through time, indicates the start of the event, and failure to find a new sample while moving forward through time (relative to the starting week) indicates the end of the event. Representing the data as events is useful for we can abstract the properties of the phenomenon. Additionally the variation of the spatial distribution over time is placed into a wider temporal context which eases the analysis with respect to covariates.

6.5 Tracking Objects

Tracking objects is an iterative process, given the current phenomenon object, or probe, we aim to find an object from those of an adjacent time sample, which is referred to as the gallery. Each gallery object is measured by its attraction to the probe object, meaning the likelihood that it was the proceeding, or is the following object. This measure is the Euclidean distance between the centre of masses of the two shapes, which have been previously calculated using Centralised Moments. However, gallery objects are only considered if the area that intersects with the probe object is greater than zero, this mass is calculated as the number of pixels. In this manner, we only consider objects that overlap the probe object in some way, but of those, we select the one closest to the probe. This allows for the correct tracking of object displacement. Figures 6.6, 6.7 and 6.8 show the result of creating events, from week 50 of 1998 which is show in Figure 6.5a.

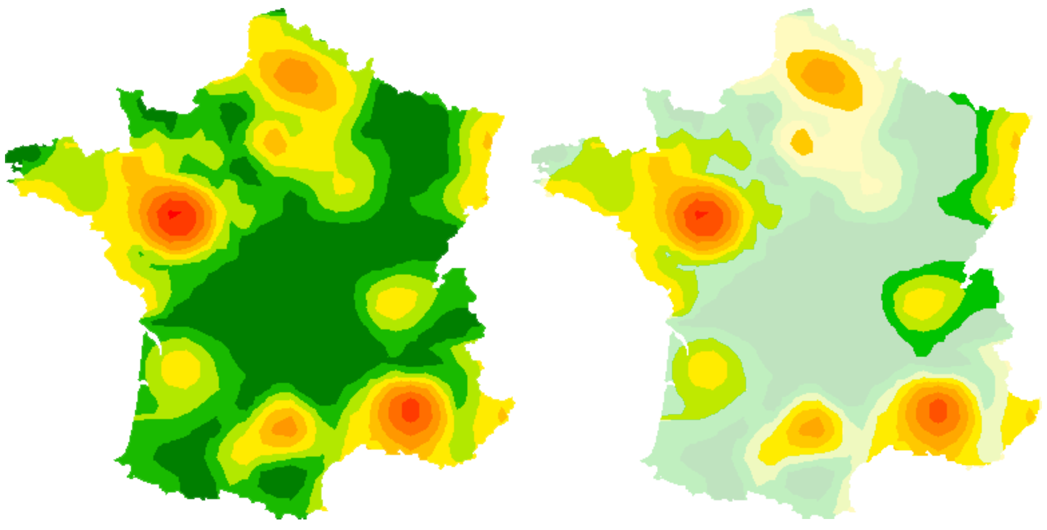

(a) Original **(b)** With extracted objects

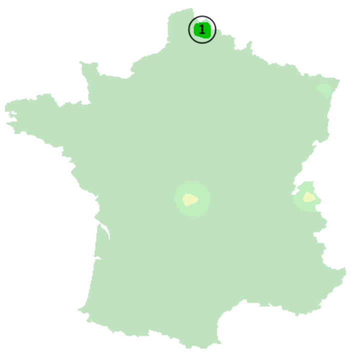
Figure 6.5: 1998 week 50. The starting time sample used to generate the Phenomenon Events shown in Figures 6.6, 6.7 and 6.8.

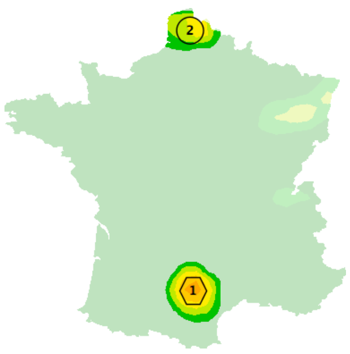
For each time sample of phenomenon, such as that shown in Figure 6.5, it is necessary to extract the phenomenon objects as described in Section 6.1. This process removes information about the phenomenon, which is considered to be noise, in order to focus on the significant objects. There is no panacea in engineering, and while the approach taken will sometimes ignore shapes that the human eye might consider significant, it is robust enough to correctly identify pertinent objects from wildly different data samples. The result of this process on week 50 from 1998 is shown in Figure 6.5b. Given these objects, a phenomenon event is created for each one in turn, tracking then proceeds going backwards in time, and then forwards in time. These nine events are depicted in Figures 6.6, 6.7 and 6.8 on pages 64, 65 and 66, and it should be noted that over the 35 week period these are not the only phenomenon events that exist, but to keep the figures as simple as possible they are the only events highlighted. Each object is labelled with a coloured shape, indicating the event it belongs to and a number, indicating its position in the event. Therefore there are three colours: black, white and blue; and three shapes: circle, square and hexagon. All annotated images are created automatically by the framework. In some cases an object is a member of more than one event, as branching and merging is quite likely, in such cases labels are placed at the same location so can be obscured. Branching and merging is discussed in Section 6.6.

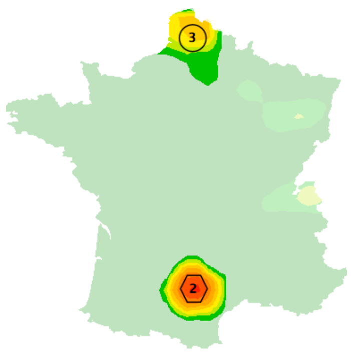
Quantitative measurement of the accuracy of the tracking algorithm poses a significant obstacle. There is no ground truth with which a comparison of the tracking result can be made. The input data is a quantised, interpolated image generated from approximately 1200 sample observations. Those observations are themselves an under-estimate of ILI activity. Essentially, the true ILI activity is unknown, therefore any measurement of how accurately an algorithm tracked amorphous shapes is fundamentally impossible. The only avenue is to qualitatively determine if the tracking of a single shape from one image to the next is reasonable given the two images themselves. Closely analysing the transitions depicted in Figures 6.6, 6.7 and 6.8 show that in the lifetime of an event the tracking is exceedingly accurate. Any failings, such as between shapes 12 and 13 in Figures 6.6l and 6.7a, are in fact a deficiency of the underlying object detection process. This deficiency is that an object containing more than one object is not considered a Phenomenon Object, and therefore not tracked; when the area of the contained objects approaches their parent, it might be considered more suitable to track the parent rather than the contained shapes. This minor deficiency is generally only noticeable at low ILI levels.

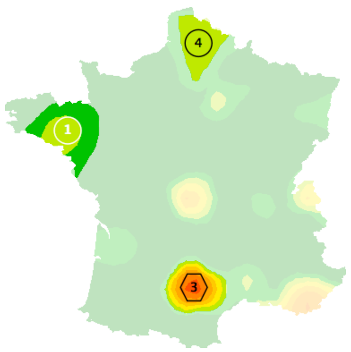
One area where it is possible to definitively gauge where the tracking algorithm fails is at

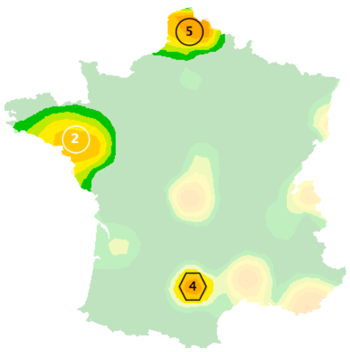
Phenomenon Event creation and termination. Of the nine events shown in the following figures, two events terminate prematurely, or 22%. Here failure means that tracking has not continued although a shape exists which could reasonably be the next iteration. These failures are the white hexagon event in Figure 6.7c, and the black hexagon in Figure 6.7d. In both cases it appears that there is a shape in the subsequent image which could conceivable be their next iteration. The failing is most likely due to the next images being just outside the current image's area, and the algorithm aims to choose shapes that are nearby, and therefore can possibly exclude the nearest shape entirely.

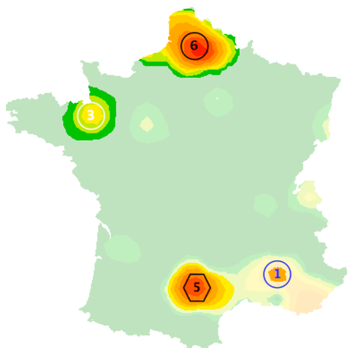

(a) 1998/36

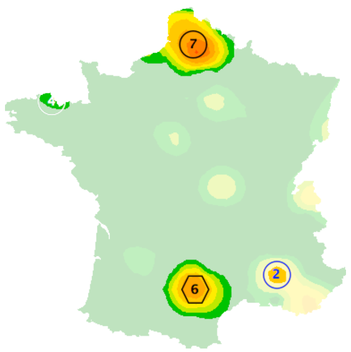

(b) 1998/37

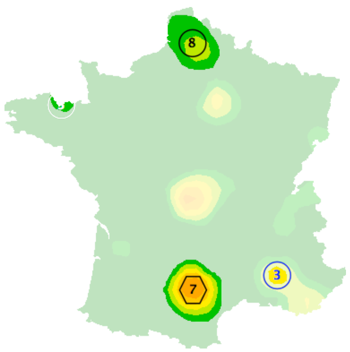

(c) 1998/38

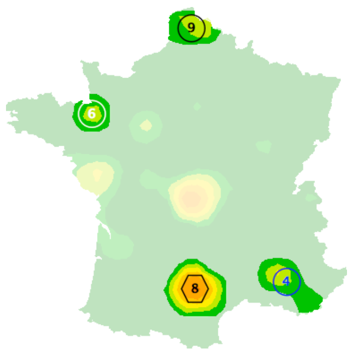

(d) 1998/39

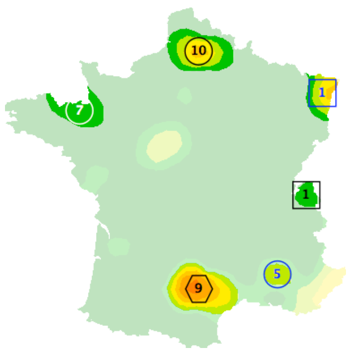

(e) 1998/40

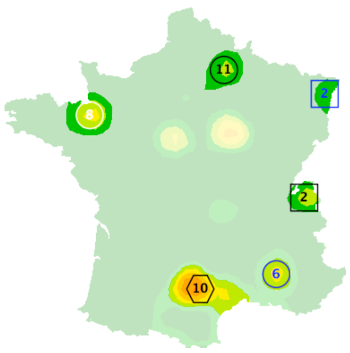

(f) 1998/41


(g) 1998/42


(h) 1998/43


(i) 1998/44


(j) 1998/45


(k) 1998/46

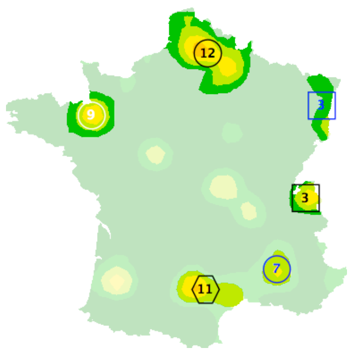
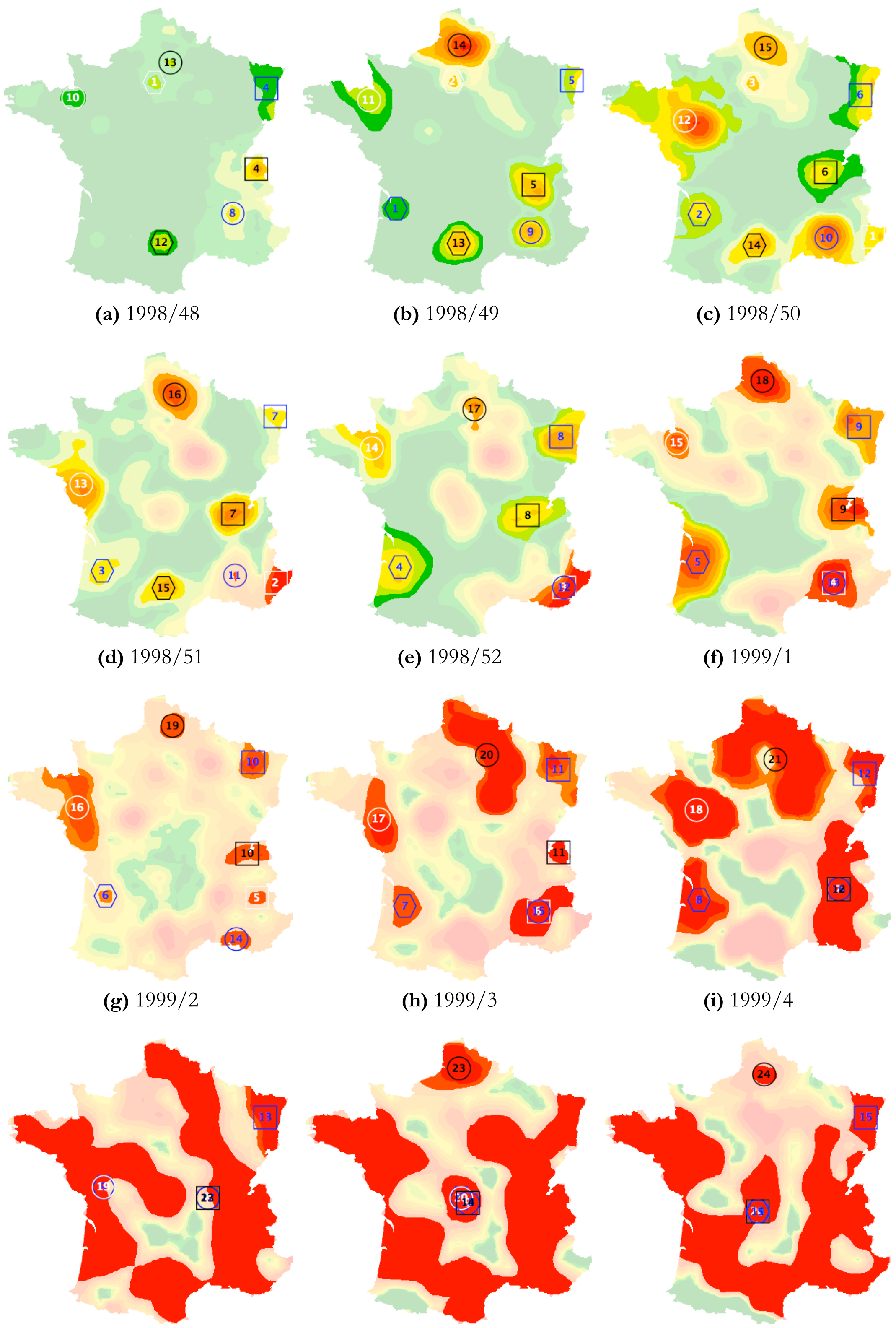

(l) 1998/47

Figure 6.6: Tracking disease outbreaks (year/week), part 1



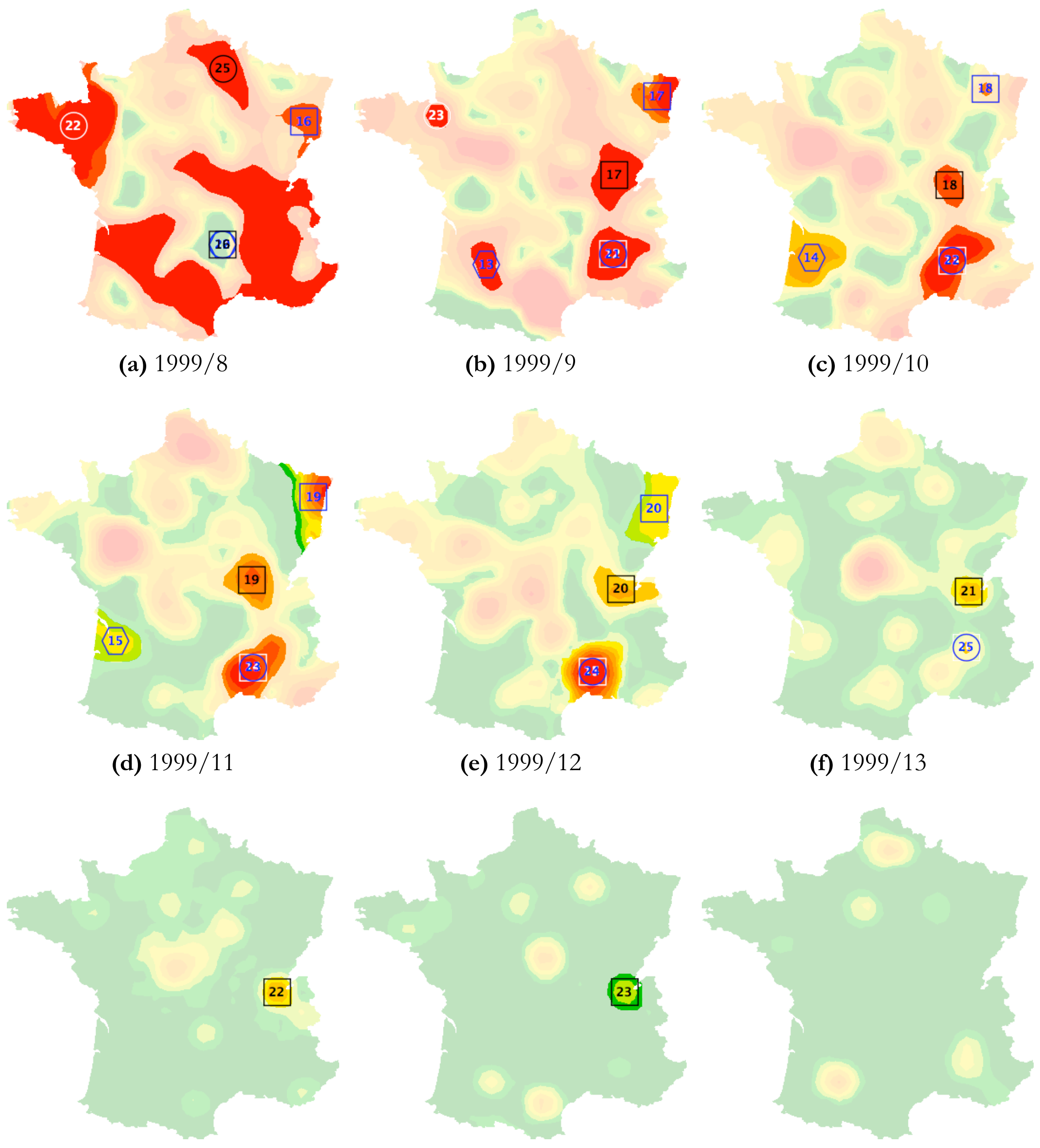
(a) 1998/48 **(b)** 1998/49 **(c)** 1998/50

(d) 1998/51 **(e)** 1998/52 **(f)** 1999/1

(g) 1999/2 **(h)** 1999/3 **(i)** 1999/4

(j) 1999/5 **(k)** 1999/6 **(l)** 1999/7

Figure 6.7: Tracking disease outbreaks (year/week), part 2



(a) 1999/8 **(b)** 1999/9 **(c)** 1999/10

(d) 1999/11 **(e)** 1999/12 **(f)** 1999/13

(g) 1999/14 **(h)** 1999/15 **(i)** 1999/15

Figure 6.8: Tracking disease outbreaks (year/week), part 3

6.6 Branching And Merging Objects

When tracking objects, it is possible that two or more objects from one sample track to the same object in an adjacent time sample, this is referred to as merging. It is effectively the intersect of two Phenomenon Events. Transitions of this type are present in the sample weeks shown throughout Figures 6.6, 6.7 and 6.8, and again in further detail in Figure 6.9. Multiple objects merging into a single object is not altogether problematic, the algorithms and framework maintains a many-to-many relationship between phenomenon objects and phenomenon events, so that a single object may be associated with multiple events, and of course a single event references many different objects. Therefore it is quite feasible for the large phenomenon object in week five in 1999 to be a member of four different events.

It is likely however that large objects will branch into multiple smaller objects. Figure 6.10 depicts the eventual branching, three weeks later, of the resultant large object from the merging of Figure 6.9. Maintaining the most likely object transitions with respect to each event is problematic. In Section 6.5 it was stated that the Euclidean distance is used to rank the attraction between the probe and each gallery object during object tracking. This approach is modified to measure the attraction between the centre of mass of the *event* rather than the probe object, thereby minimising the power of large objects whose centre of mass is invariably in the centre of the territory. The centre of mass of the event is simply the mean centre of mass of the objects associated with it prior to the object currently being tracked. Additionally, as tracking is performed in both temporal directions, "prior" is relative to the tracking direction. Therefore when tracking backwards through time, the event's centre of mass is calculated using those from the current object and any future objects associated with the event, as in equation

$$\{\tilde{x}, \tilde{y}\} = \left\{ \frac{1}{n} \sum_{i=t}^{n} S_i(x), \frac{1}{n} \sum_{i=t}^{n} S_i(y) \right\} \tag{6.1}$$

for object S_t followed by n other objects. This method ensures that as the tracking continues, the positions of previously tracked objects are just as significant as the end of the object chain. Work using a mean centre of mass weighted to favour different objects has been per-

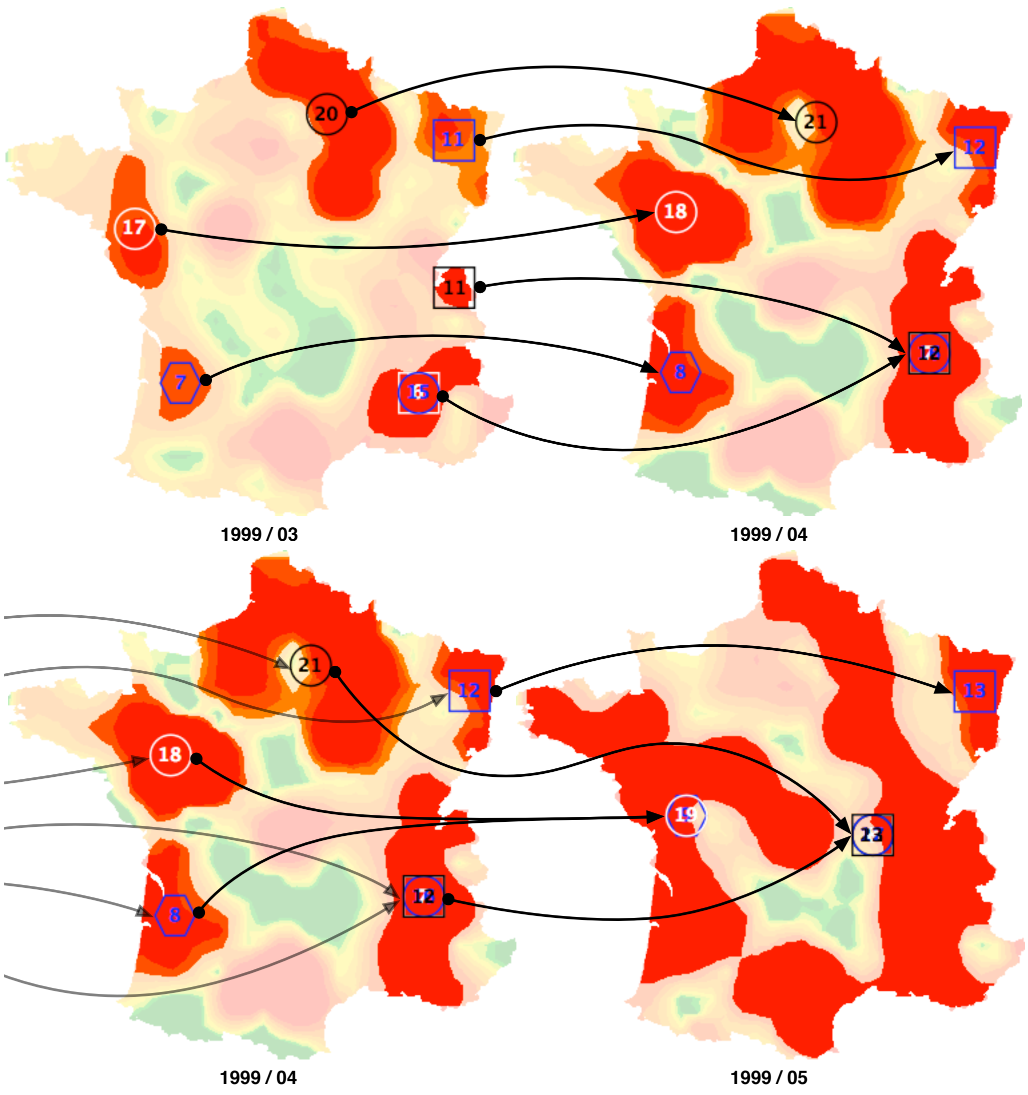


Figure 6.9: Merging objects

formed, and while the tracking results were in some cases unexpected due to the high degree in variation between event objects, the proposal is discussed further in Section 9.1.

Figure 6.10 also shows the termination of an event, indicated by the dotted arrow, which a human observer would dispute. In terms of the automatic object tracking this behaviour is correct, the problem is a result of the object segmentation which has extracted the two objects in the southern eastern area of France, rather than the larger underlying object, as being more significant. Had this been extracted as an object, then the northern and south eastern objects from 1999/08 would have merged in week nine, and the northern event continued.

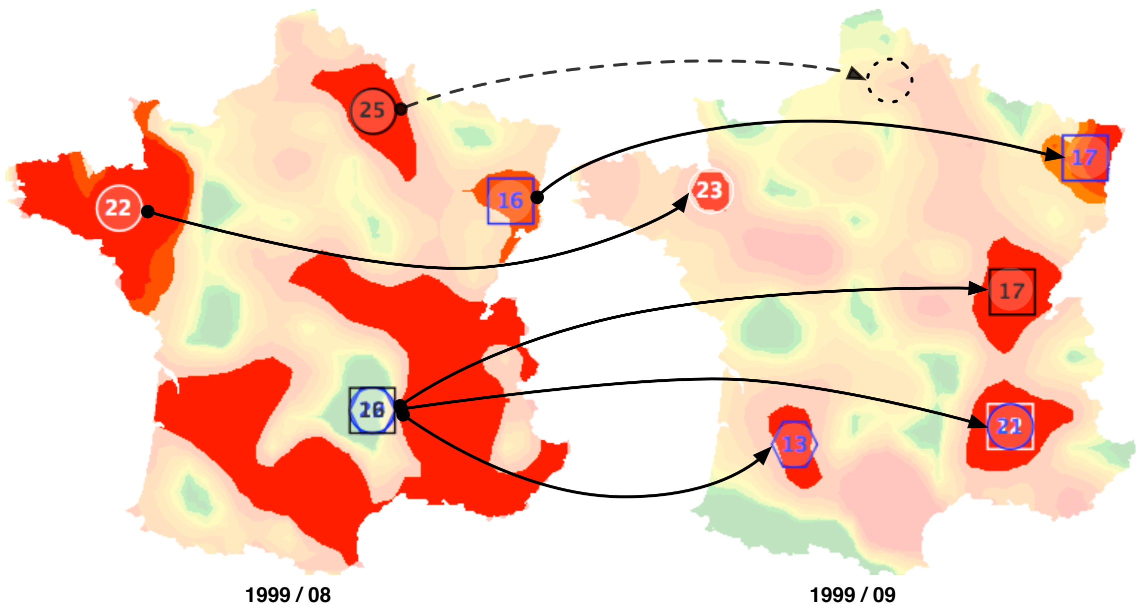


Figure 6.10: Branching objects

6.7 Forecasting Future Activity

This section summarises an avenue of research, which was pursued in the formative stages of the object-based framework's development, aimed at forecasting objects using information solely from the phenomenon event without any consideration of covariates. The method was developed before the covariate data was integrated into the framework.

The phenomenon event encapsulates the temporal deformation of the spatial attributes of the measured property of the study phenomenon. Therefore, it is postulated that after modelling the deformation of these attributes, the future spatial attributes may be estimated. The estimated attributes, which are Tchebichef moments, are capable of being reconstructed into an image, or rather an object, and hence provide an estimate of future activity. It is likely that the forecasting process will need to be performed on more than one phenomenon event to forecast the future activity over the complete spatial area under study[1], as each time sample of the measured phenomenon will have a number of objects, each of which will be members of different phenomenon events.

The Tchebichef moments of an object contain N^2 values, where N is the largest even

[1]In the framework this area is referred to as a *Sample* area

side of the square spatial domain. For the ILI surveillance data used throughout this research $N = 382$, although the surveillance images are 381px $\times$ 381px in size. N defines the size of the Tchebichef Polynomial, which requires an even size to ensure that it is orthogonal. To model the deformation in these changes N^2 new values must be estimated. Using a single N^2 dimensional model is not realistic so each combination of $\{p, q\}\ \forall\ 0 \leq p, q < N$ is modelled separately. Finally, from Figure 4.8, which illustrates the accuracy of image reconstruction from incomplete sets of moments, it is clear that reconstruction based on all moment orders up to N is not necessary, and empirically reconstruction from 75% carries acceptable errors.

6.7.1 Selecting a Model

Straight line, quadratic and cubic models, were fitted using least squares to the spatial attributes, and their mean squared error reviewed. Figure 6.11 plots three example moments: $(0, 0)$, $(5, 2)$ and $(9, 1)$ with cubic, quadratic and straight line models respectively.

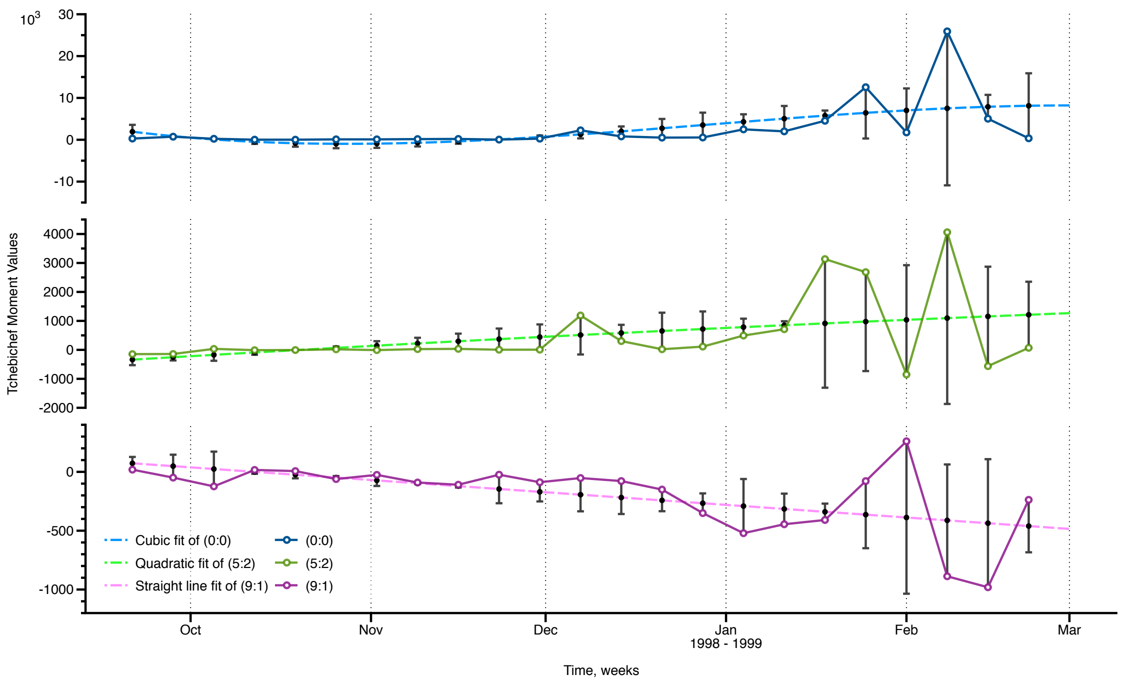
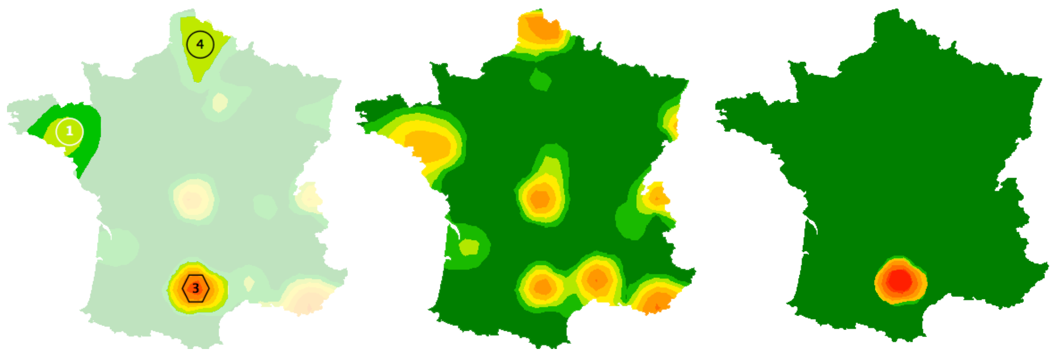


Figure 6.11: Fitting cubic, quadratic and straight line models to example Tchebichef Moments, labelled as $(p : q)$, for the Phenomenon Event that starts in week 39 of 1998, in Figure 6.6d on page 64, and denoted by the white circles thereafter.

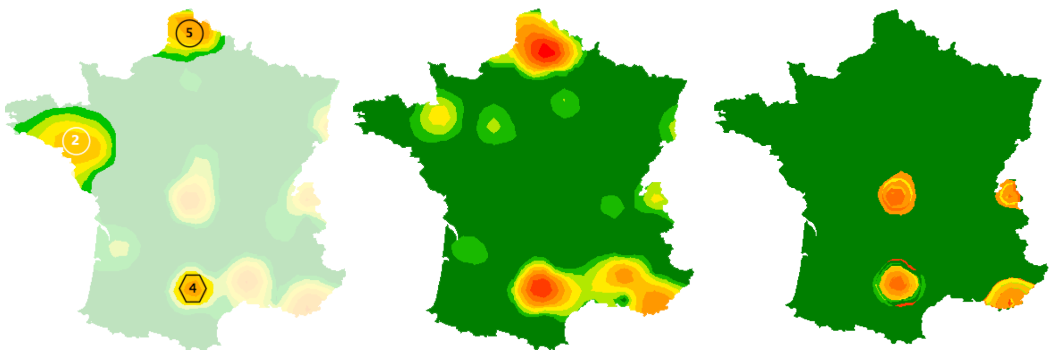
Clearly none of these models are an excellent fit, as the residuals increase significantly near the end of the event when the moment values vary wildly. These extremes changes in the moment values are not unexpected for it is the nature of the measured phenomenon. The Influenza event will most likely start small, as a small population is infected, then increase in size as more people become infected, and finally decrease in size as those infected recover.

In general however the forecast based on an arbitrary phenomenon sample will not be at the end of a phenomenon event. We only need use a small number of historical samples to fit a model; for an overdetermined cubic model, four weeks are required. It is also noted that a quadratic model provides minimal errors. The cubic model is too sensitive to noise, while a straight line is not sensitive enough, and the Tchebichef moment values vary erratically.



(a) 1998/39 **(b)** 1998/40 **(c)** Forecast of 1998/40

Figure 6.12: Estimate of week 40, 1998, only the black hexagon event had enough data to create a forecast.



(a) 1998/40 **(b)** 1998/41 **(c)** Forecast of 1998/41

Figure 6.13: Estimate of week 41, 1998

Figures 6.12 and 6.13 show the forecasts of weeks 40 and 41 from 1998 respectively. In both cases, forecasts of shapes are missing because sufficient history is not available to estimate the Tchebichef moments. The shapes that are forecasted seem reasonable, yet are also not completely accurate. This is because the objects of consecutive weeks vary considerably, the most important different being when new objects have spontaneously appeared, which is impossible to estimate from the history of shape attributes alone. This inadequacy, is not entirely due to the object based framework however, and could be mitigated to some degree with higher frequency sampling. The phenomenon data is discussed further in the conclusions in Section 8.4.

Chapter 7

Event Analysis

In this chapter, analysis techniques of phenomenon events are developed. The approach used for this research is based on correlation operations, however the framework is extremely flexible and exposes all attributes of the phenomenon events, meaning that other techniques may be incorporated into the analysis. In section 7.1, analysis of the covariate data is performed in conjunction with the phenomenon data, and experiments performed from then onwards use both data sets simultaneously. Section 7.2 performs a simple comparison between a phenomenon event and covariate data to confirm Influenza's seasonal behaviour. Sections 7.3 and 7.4 form the majority of the chapter and analysis. They describe the cross-correlation between shapes of the phenomenon data and covariate data, and between shapes of the phenomenon data and samples of the covariate data, respectively. Throughout different covariates are analysed and often presented together. Additionally in Section 7.3 we perform cross-correlation using a short temporal window to highlight the forecasting abilities of the framework.

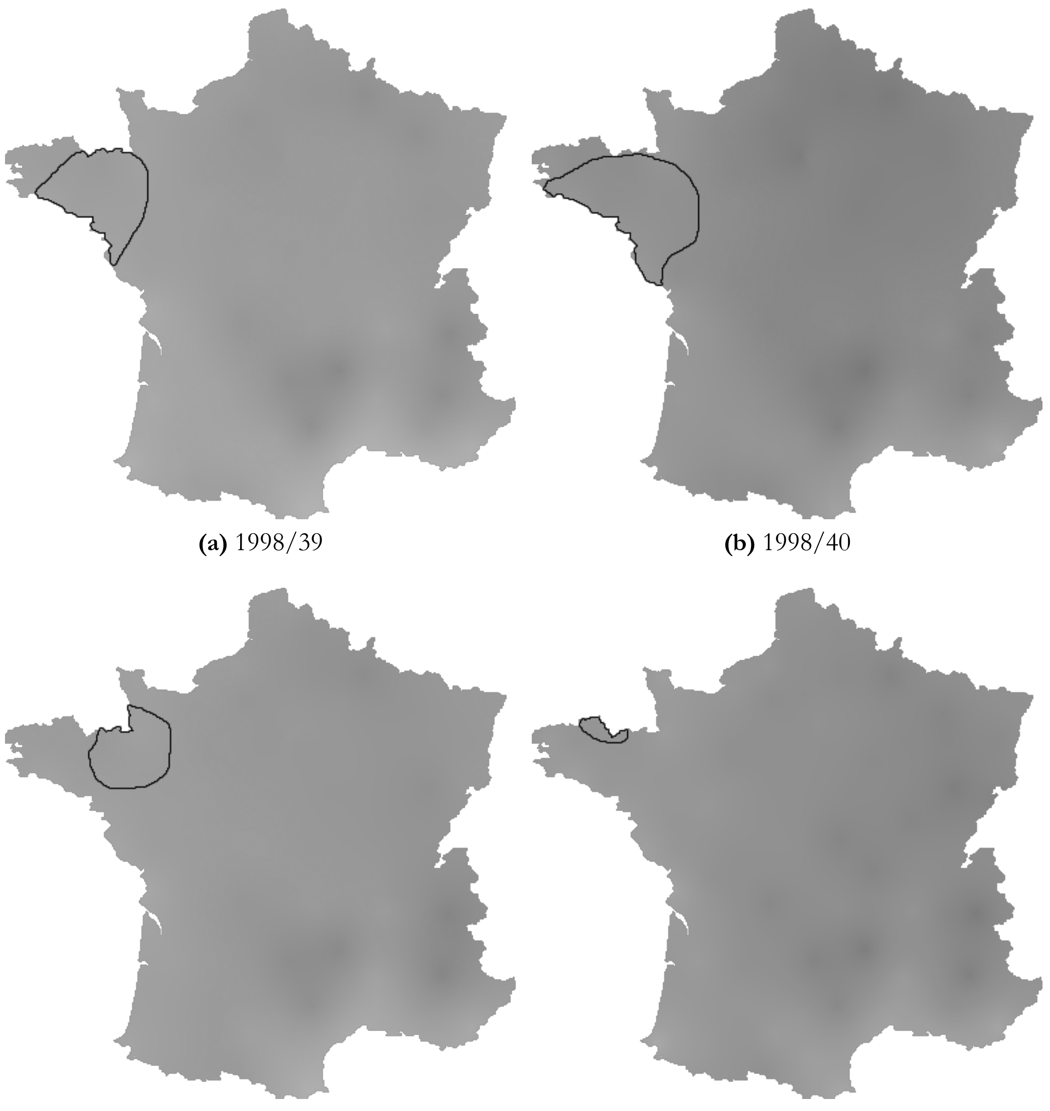
7.1 Analysis with Covariates

To analyse the phenomenon with respect to available covariate data both datasets must be viewed in the same spatial context. By implementing the spatial interpolation techniques described in Chapter 5, sample images of the covariate were created that are registered to the phenomenon data, examples of which were shown previously in Figure 5.4. Using the co-ordinates of the phenomenon objects, equivalent covariate objects were extracted from the interpolated image of the covariate for each time sample of the phenomenon event. These objects describe the spatial variation of the covariate in the same reference space as the phenomenon images. These *covariate objects* will each have the same number of non-zero pixels as their phenomenon counterparts, but a different mass and centre of mass. Additionally we can compute Tchebichef moments for these objects. Shapes of air temperature for the first four weeks of an event that starts in week 39 of 1998 are shown in Figure 7.1.

A cursory examination of the interpolated values might suggest that the result can be stored as simple grayscale images. Unfortunately, however, the variability between different covariate types is large, and a method to store floating point single band data with a large range is required. The average mean sea level pressure is 1017 hPa, and the average number of hours of rainfall is 8.9 hours per week. The framework must provide a storage method for these wildly different spatial data, and to that end a simple custom image type was developed that stores a single band image with double precision floating point values, yet will, if necessary, export to common image types for viewing. The interpolated images shown in Figures 7.1 and 5.4 were normalised into an 8 bit grayscale image after interpolation.

The length of a phenomenon event is measured as the number of time samples where objects of the phenomenon are successfully tracked. In the examples shown in Figures 6.6, 6.7 and 6.8 this ranges from three to 25 weeks. However, covariate objects were generated over a time period which starts on a given week depending on the study phenomenon, as shown by Figure 7.2. In the case of influenza, the season starts on week 32, and has a five week overlap, this indicates that covariate objects are created from week 27 through to week 37 of the following year. For the weeks proceeding and following the phenomenon event the object co-ordinates were taken from the first and last object respectively. This is done in order the variation of the covariate not only during the phenomenon period but also before and after, to discover behaviour that might indicate why the event started.

Tchebichef moments of phenomenon objects and covariate objects are computed. In

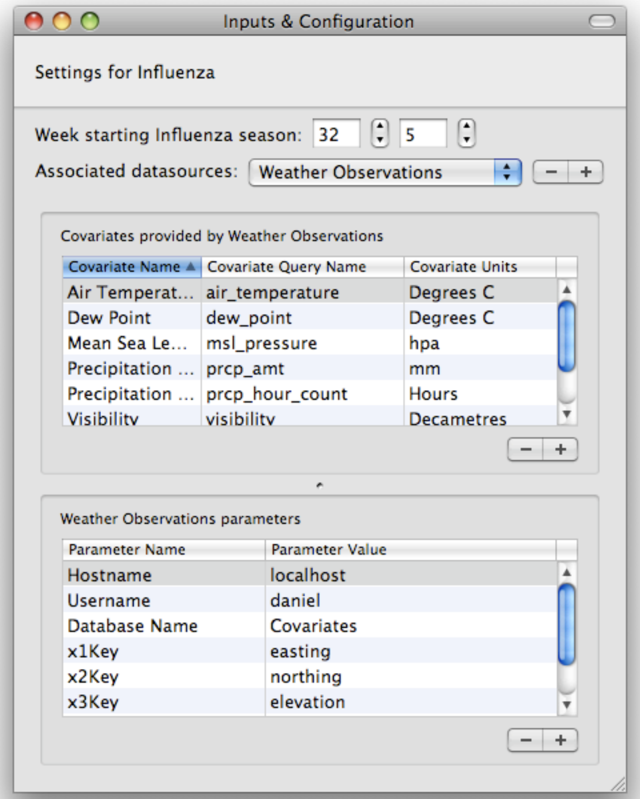


(a) 1998/39 **(b)** 1998/40

(c) 1998/41 **(d)** 1998/42

Figure 7.1: Shapes of air temperature for the first four weeks of event starting with in week 39 of 1998, see Figure 6.6d.

section 4.6.1 it was discussed that translation invariance was achieved as a consequence of computing the moments for objects isolated from their background. However, to compare the moment values of multiple objects it is imperative that those moments are computed over the same size. Hence before computing the moment, the largest side of any shape from the event is used as N, and each object is placed in a square of this size, while maintaining its correct position in the whole image. This does not impact the computation of the moments adversely, as where the pixel intensity is zero, no calculations are required.



Inputs & Configuration

Settings for Influenza

Week starting Influenza season: 32 5

Associated datasources: Weather Observations

Covariates provided by Weather Observations

Covariate Name	Covariate Query Name	Covariate Units
Air Temperat...	air_temperature	Degrees C
Dew Point	dew_point	Degrees C
Mean Sea Le...	msl_pressure	hpa
Precipitation ...	prcp_amt	mm
Precipitation ...	prcp_hour_count	Hours
Visibility	visibility	Decametres

Weather Observations parameters

Parameter Name	Parameter Value
Hostname	localhost
Username	daniel
Database Name	Covariates
x1Key	easting
x2Key	northing
x3Key	elevation

Figure 7.2: Phenomenon settings

The process of covariate object extraction described here is not the only method available. The discussion of object segmentation techniques in Section 6.3 is applicable here. A possible disadvantage of deriving the covariate objects from the phenomenon objects is that their position and boundaries are identical, which may overshadow any differences within the shapes during correlation. Alternative approaches are discussed further in Section 8.3.

7.2 Confirming Influenza's Seasonal Behaviour

Once the tracking of phenomenon objects was performed, the array of tracked objects was treated as a single entity in the framework: a *phenomenon event*. This abstraction allows for a myriad of further analysis techniques to be applied to the objects. This research focuses on correlation based experiments using several properties of the phenomenon event. However, other avenues of research afforded by the framework are discussed in Chapter 9.

The phenomenon event can illustrate how properties of the phenomenon vary over time, such as the position, area and "mass" of the object. The shape description techniques discussed in Chapter 4 may also be computed on each object and, therefore, illuminate how the shape of the phenomenon objects change over time. This level of analysis is not possible given a consideration of the phenomenon's measured property as a whole.

In order to analyse the Tchebichef moments of an event's objects it is necessary to compute the moments over a constant spatial domain, as the Tchebichef polynomial is discrete and fixed over the range 0 to $N - 1$. Therefore, to fully observe the variation in any single moment it must be calculated for all objects. As objects vary in size, it is necessary to select the largest object, and compute the moments for all objects based on this size. For objects smaller than N the extra pixels are zero and do not contribute to the moment values. Therefore, they do not need to be calculated so no added computation cost is incurred. Once these image moments have been computed, observations of how individual moments, or object properties vary over the lifetime of the event is possible. For example, Figure 7.3 shows the variation in object mass for an event, compared with the air temperature at the centre of mass of each object.

It is clear that as the air temperature decreases, the mass of the influenza objects increases. As the air temperature begins to rise, the event decreases and ends. It should be noted that the object mass is plotted on a logarithmic scale, as the variation in object area is significant. The mass of the object

$$m_{00} = \sum_x \sum_y f(x, y)$$

is a convenient attribute as it conveys both the intensity of the phenomenon and the area covered by the object.

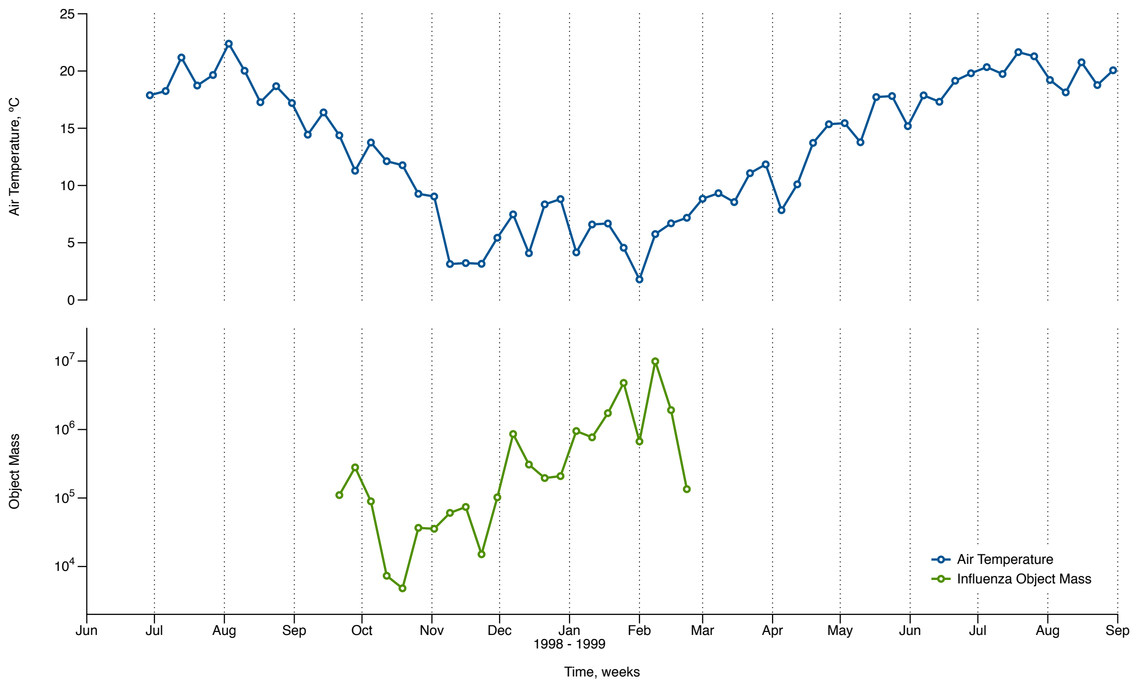


Figure 7.3: Variation in influenza object mass for a phenomenon event starting in September of 1998: labeled with white circles starting on Figure 6.6d, and air temperature at the centre of mass of objects over the entire phenomenon season.

7.3 Cross-correlation Between Phenomenon and Covariate Shapes

Cross-correlation measures the similarity of two waveforms as a function of a time-delay applied to one of them. In signal processing it is often used to locate a short signal within a larger signal. For a discrete function it is defined as:

$$(f \star g)\,[a] = \sum_{b=-\infty}^{\infty} f[b]g[a+b] \tag{7.1}$$

In these cross-correlation experiments $g[\cdot]$ is the array of covariate values, and the shorter $f[\cdot]$ is the phenomenon array. It follows from Equation (7.1) that the cross-correlation is only fully computed over the phenomenon season minus the length of the phenomenon event, and for the remainder of the season the cross-correlation drops to zero.

Equation (7.1) works on single dimension waveforms. However, these datasets are spatiotemporal. Therefore, the cross-correlation is computed in a $2D+t$ fashion. As the areas of the phenomenon objects vary over the course of the event, the values used in the cross-correlation must all be registered in the same spatial domain. This has already been effected for the Tchebichef moments. Cross-correlation maintains the dimension of its input data, which means that in these experiments the algorithm returns a $2D - t$ dataset. Along with the correlation at each sampled point, the mean average of each sample was calculated and plotted.

The results of various cross-correlation experiments are now presented. All experiments use the same phenomenon event: 1998 39 024 02[1], which starts with the shaped labeled 1 with a white circle in Figure 6.6d on page 64. In the following time series graphs, the 23 week phenomenon event is the shaded time period. The hatched time period is where the signal is not defined at all filter indexes plus the lag in the cross-correlation function, and therefore not strictly relevant.

The cross-correlation of moments of influenza objects and air temperature objects is shown in Figure 7.4. The cross-correlation function is performed at each T_{pq} for $p, q = 0, 1, 2, \ldots, N - 1$. Meaning that for $p, q = 0$, the signal array consists of the T_{00} value of each time sample of the covariate, and similarly the filter array consists of the T_{00} value for each time sample of the phenomenon event. The cross-correlation of these two signals is then computed for that spatial location $(0, 0)$.

In the following plots the cross-correlation results were also averaged over the spatial domain to provide a single dimension time series. In Figure 7.4 the large peak at the start of the event is expected, as the non-zero pixels of the phenomenon shape exactly match those of the covariate shapes. The cross-correlation of the same event with a different covariate, dew point, is shown in Figure 7.5. The pattern of the correlation is very similar to that of air temperature, which is not unexpected, as the shapes have the same position, size and boundary. However, the magnitude is significantly less than with air temperature.

In general the correlation structures will not always be so similar. Their cross-correlation with another event, starting two weeks later in week 41 is shown in Figure 7.6. Here both have peaks at the start of the event, but are otherwise completely different. In all cases

[1]As explained at the end of Section 6.1, this key refers to the 2nd shape from level 24, in the 39th week of 1998.

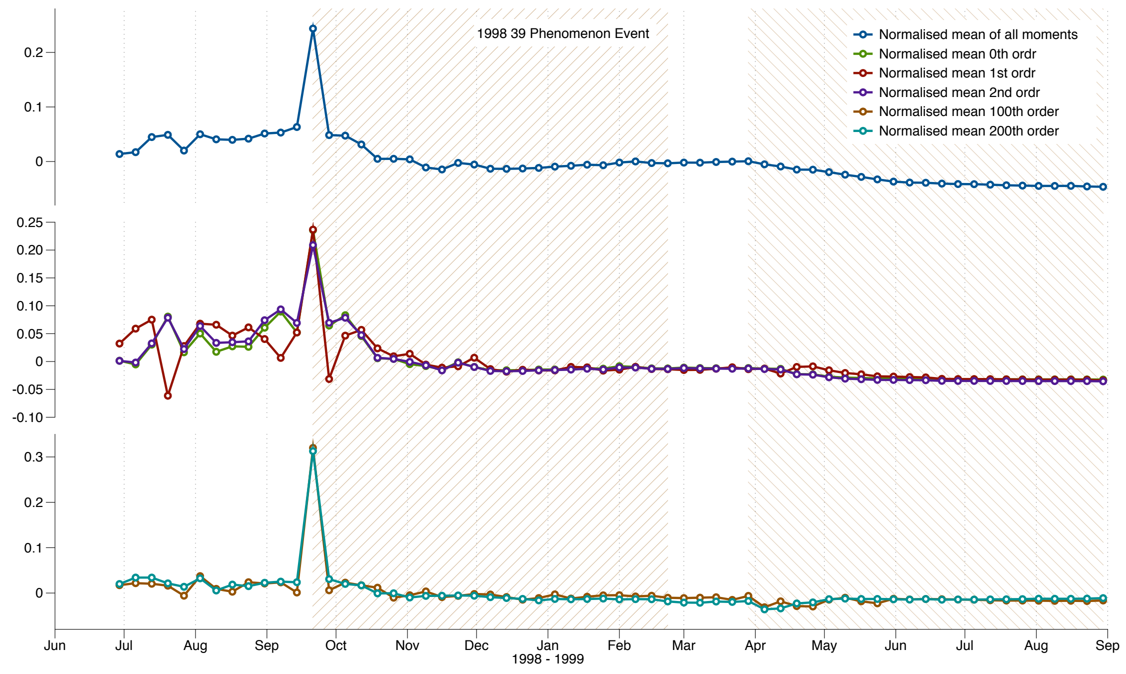


Figure 7.4: Correlation between Tchebichef moments of Influenza shapes with those of Air Temperature shapes for the 1998 39 event. The cross-correlations are normalised by the length of the event.

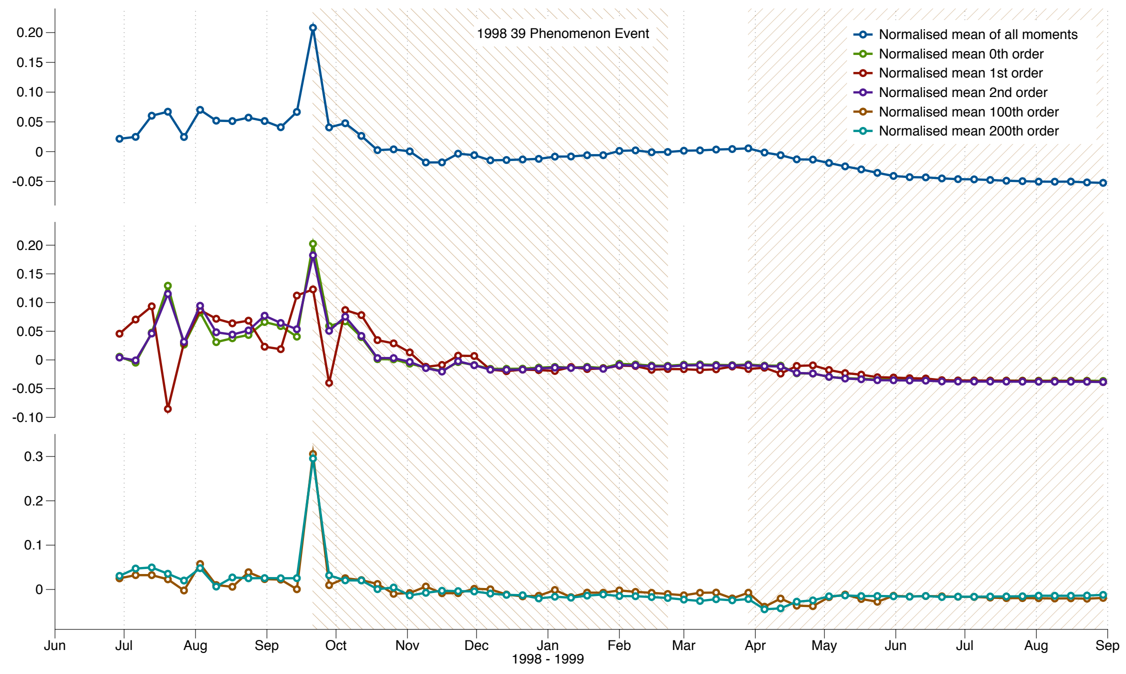


Figure 7.5: Correlation between Tchebichef moments of Influenza shapes with those of Dew Point shapes for the 1998 39 event. The cross-correlations are normalised by the length of the event.

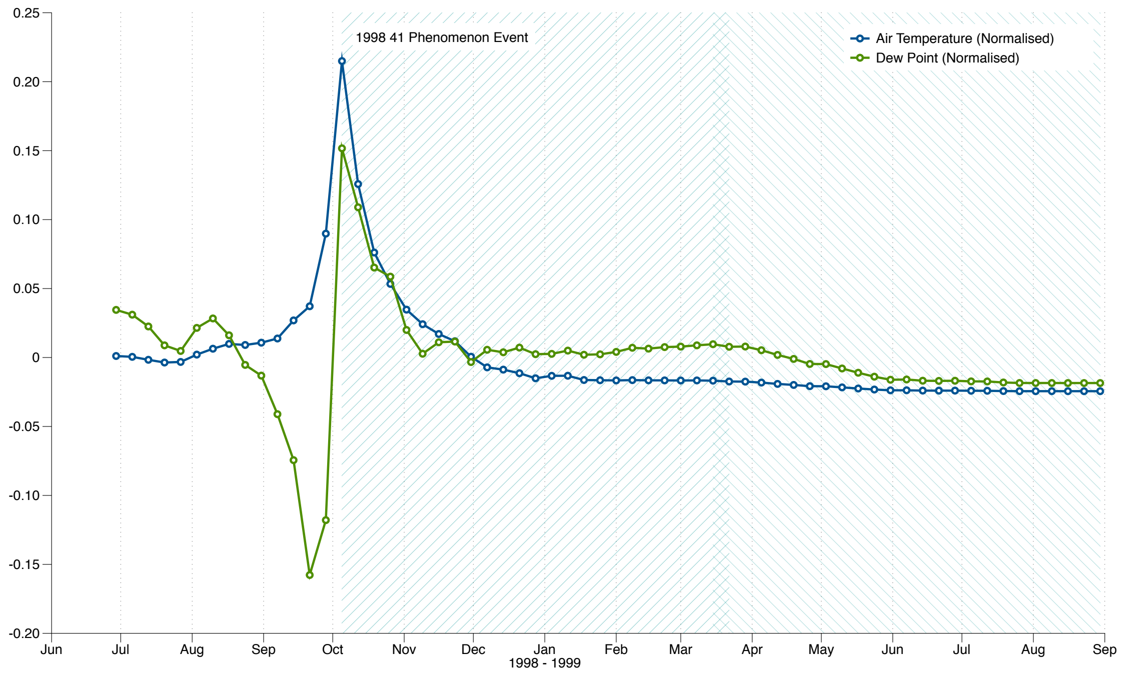


Figure 7.6: Average cross-correlation of Tchebichef moments of Influenza with those of Dew Point and Air Temperature shapes for the 1998 41 event, which starts in Figure 6.6f on page 64. The cross-correlations are normalised by the length of the event.

the magnitudes of the correlation peaks of dew point are less than those of air temperature which would indicate that its relationship with the phenomenon is more tenuous.

The large positive cross-correlation peaks indicate a strong similarity in the shapes of the influenza, and the shapes of the covariate. It should be noted that the value of Tchebichef moments is not directly proportional to the covariate value and, therefore, the cross-correlation is measuring how similar the *distribution* of the influenza is to the covariate over the area. For example, it is expected that a negative correlation between *values* of air temperature and *values* of ILI occurrences exist. These plots are comparing the *shapes* of objects of ILI occurrences with those of air temperature.

In the two experiments above, the entire phenomenon event was used as the filter in the cross-correlation, so only a single peak is expected where the complete event matches. However, the framework affords the ability to select a shorter time period of the event for cross-correlation. In this way it is possible to look for matches of part of the phenomenon event with "future" covariate activity. The cross-correlation of the first 10 time samples of the 199839 event with air temperature and dew point are shown in Figure 7.7, with an additional plot of the phenomenon object mass on a logarithmic scale. Here, subsequent peaks in December and February in the correlation plots correspond to when the influenza objects increase in size and intensity. This indicates that given accurate covariate forecasting, of which weather forecasting is routinely performed, forewarning of the measured property of a phenomenon, related to such a covariate is possible.

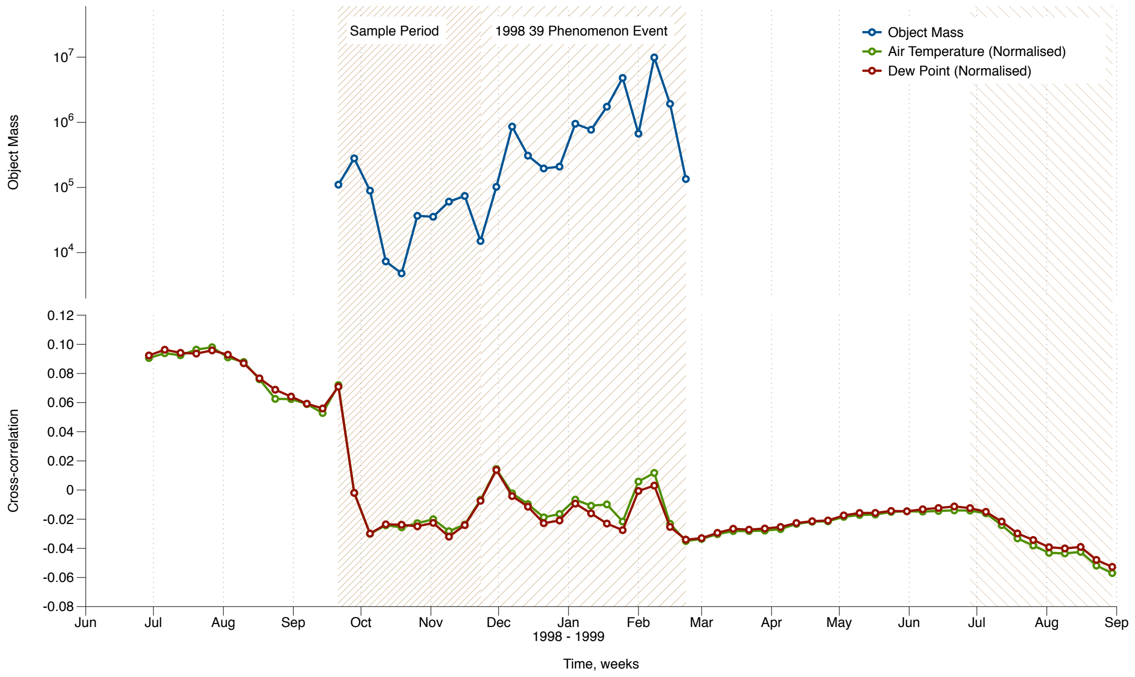


Figure 7.7: Average cross-correlation of Tchebichef moments of the first 10 weeks of Influenza with those of Air Temperature and Dew Point shapes for the 1998 39 event. Additional plot of phenomenon object mass on a logarithmic scale. The cross-correlations are normalised by the length of the event.

7.4 Cross-correlation between Shapes and Samples

The framework allows correlation to be performed using a variety of properties such as using the phenomenon and covariate values directly. Theoretically, performing the same correlation on the phenomenon and covariate value at each pixel position of the objects will return a similar distribution, as the covariate's objects are the same shape as the phenomenon objects. However, because the objects vary in size throughout the event and the correlation must be performed over a constant spatial domain, many points in this domain lie outside the area defined by all but the largest objects. Therefore, the value at these points is zero[2] which in turn results in the cross-correlation at those points being zero and, therefore, the average cross-correlation tends toward zero. This highlights one of the benefits of Tchebichef moments, which will be discussed further in Section 8.2, namely that they ensure a valid representation of spatial data over a fixed and complete size, which is ideal for comparing objects that vary in size.

The framework enables the cross-correlation of spatial areas at different scales, for example shapes against shapes, as illustrated in Section 7.3, but also shapes against samples and samples against samples. Figure 7.8 shows the average cross-correlation of *values* of influenza with those of air temperature and dew point using the whole image space, or samples vs samples. This simple experiment confirms that there is a negative correlation between air temperature and influenza, and between dew point and influenza. This plot is calculated averaging the cross-correlation of the values of influenza against those of air temperature and dew point at each pixel position in the phenomenon mask, which for the Influenza datasource is a 381^2 pixel map of France.

Figure 7.9 and 7.10 are screen captures of the application framework in use. Figure 7.9 depicts the objects from the phenomenon event that is selected using the control in the top left corner of the window. These images are all generated automatically by the framework using the result of the object tracking process combined with the original surveillance images. Indeed, the tracking process creates the events themselves and populates the selection list. All image processing within the framework is done using grayscale images, but images are rendered using the original colour mappings. Figure 7.9 is a table of data for the same event. Each row represents a single time sample (in the case of influenza the surveillance is weekly) and the table column labels are reasonably self explanatory. However, it should be noted that the phenomenon value (pixel intensity of the influenza image), air temperature (or whichever covariate is selected), average, 1st derivative and 2nd derivative of the air temperature are all taken from the centre of mass of the object. The two columns of moment

[2]Technically the value is simply not defined, as zero might be a valid covariate value.

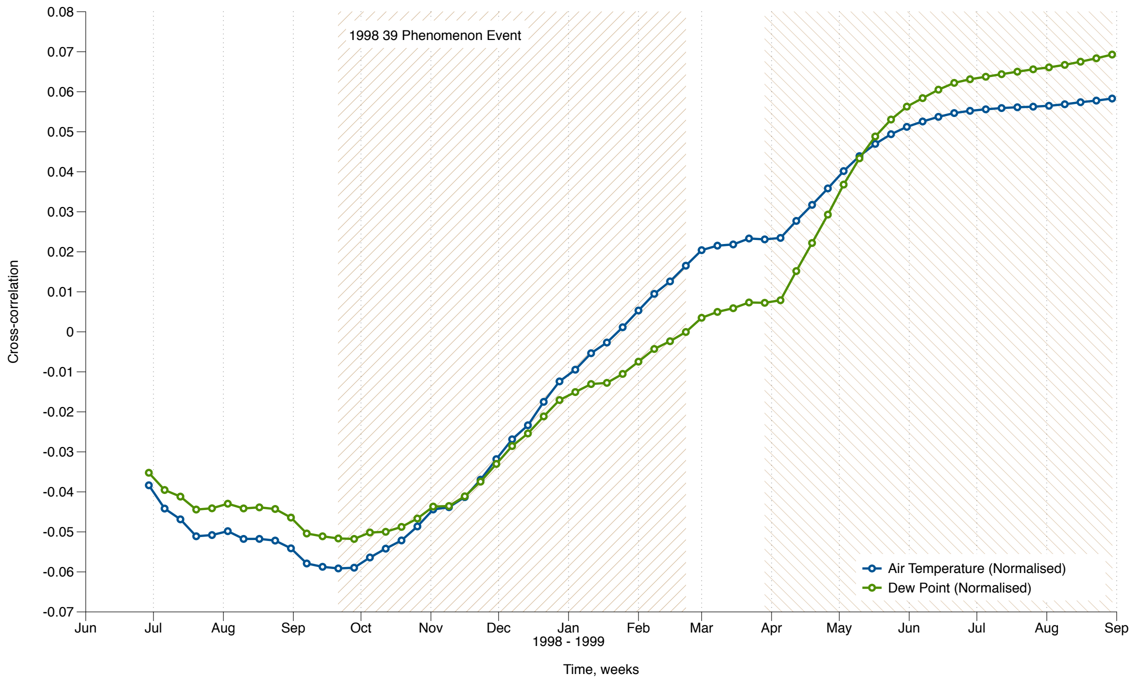


Figure 7.8: Average cross-correlation of Influenza values with those of Air Temperature and Dew Point for the 1998 39 event. The cross-correlations are normalised by the length of the event.

values, for T_{11} are to inform the user that the moments have been computed. Controls at the bottom right of the window provide analysis functionality.

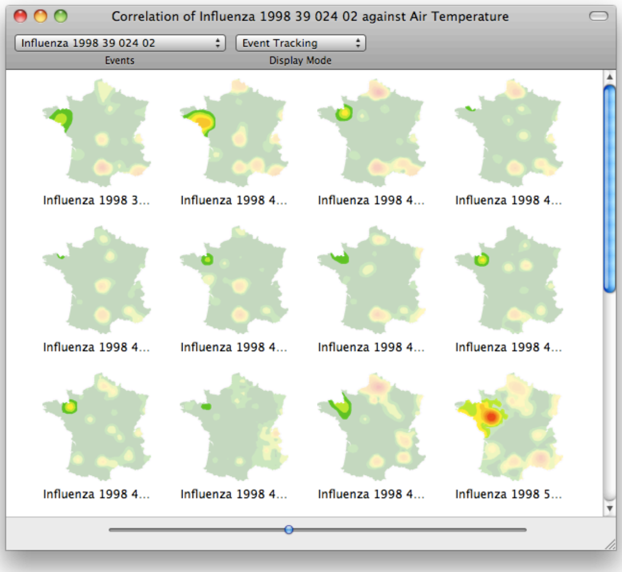
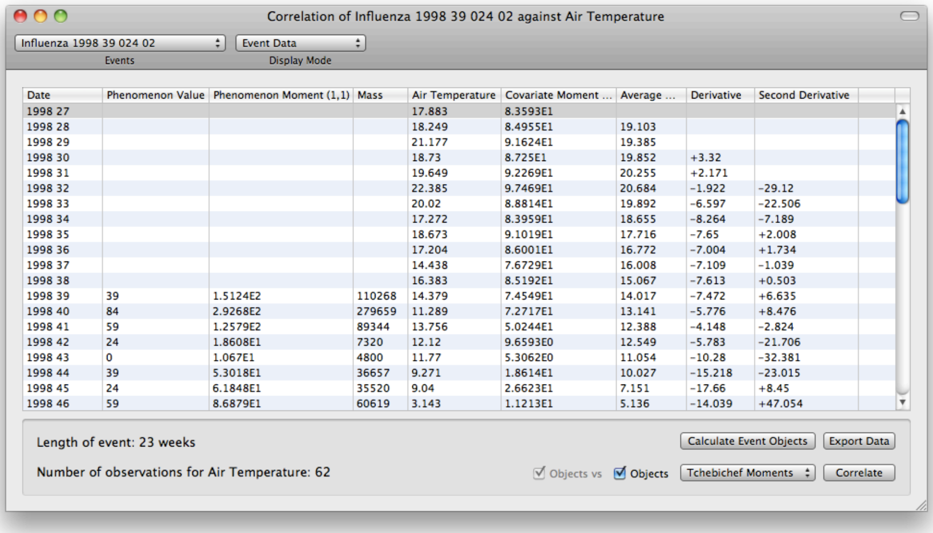


Figure 7.9: Viewing Phenomenon event images from within the framework.


Correlation of Influenza 1998 39 024 02 against Air Temperature

Events: Influenza 1998 39 024 02

Display Mode: Event Data

Date	Phenomenon Value	Phenomenon Moment (1,1)	Mass	Air Temperature	Covariate Moment ...	Average ...	Derivative	Second Derivative
1998 27				17.883	8.3593E1			
1998 28				18.249	8.4955E1	19.103		
1998 29				21.177	9.1624E1	19.385		
1998 30				18.73	8.725E1	19.852	+3.32	
1998 31				19.649	9.2269E1	20.255	+2.171	
1998 32				22.385	9.7469E1	20.684	-1.922	-29.12
1998 33				20.02	8.8814E1	19.892	-6.597	-22.506
1998 34				17.272	8.3959E1	18.655	-8.264	-7.189
1998 35				18.673	9.1019E1	17.716	-7.65	+2.008
1998 36				17.204	8.6001E1	16.772	-7.004	+1.734
1998 37				14.438	7.6729E1	16.008	-7.109	-1.039
1998 38				16.383	8.5192E1	15.067	-7.613	+0.503
1998 39	39	1.5124E2	110268	14.379	7.4549E1	14.017	-7.472	+6.635
1998 40	84	2.9268E2	279659	11.289	7.2717E1	13.141	-5.776	+8.476
1998 41	59	1.2579E2	89344	13.756	5.0244E1	12.388	-4.148	-2.824
1998 42	24	1.8608E1	7320	12.12	9.6593E0	12.549	-5.783	-21.706
1998 43	0	1.067E1	4800	11.77	5.3062E0	11.054	-10.28	-32.381
1998 44	39	5.3018E1	36657	9.271	1.8614E1	10.027	-15.218	-23.015
1998 45	24	6.1848E1	35520	9.04	2.6623E1	7.151	-17.66	+8.45
1998 46	59	8.6879E1	60619	3.143	1.1213E1	5.136	-14.039	+47.054

Length of event: 23 weeks

Number of observations for Air Temperature: 62

Figure 7.10: Performing correlation of a single event with air temperature from within the framework.

Chapter 8

Conclusions

The work described in this thesis draws on the techniques of traditional statistics, spatial statistics, epidemiology and image processing in order to provide a methodology to analyse spatiotemporal data.

The main thrust of this research has been to describe the spatial deformation of a study phenomenon over time, using a multi-stage process. It is assumed that data which samples a property of some phenomenon, in both space and time, is available. The contribution of this research is the development of an object based methodology to analyse such data. The various stages of this methodology were developed as follows. Initially, areas of activity within each time sample were isolated using a novel algorithm that extracts the largest and highest peaks as shape objects. This process, in effect, assimilates the data into an *object based framework*, which can subsequently be queried to provide further information. Every object provides spatial properties of itself using modern area based shape description techniques. Due to the hierarchical design of the framework, illustrated in Figure 8.1, every *area* of data, which includes the input samples of the phenomenon's measured property and detected objects within each sample, plus covariate data samples and objects, provide spatial attributes within the same spatial domain, so that they maybe be compared against each other.

The objects that exist for any given sample of phenomenon data were tracked both forwards and backwards in time until a complete phenomenon event is formed. The tracking

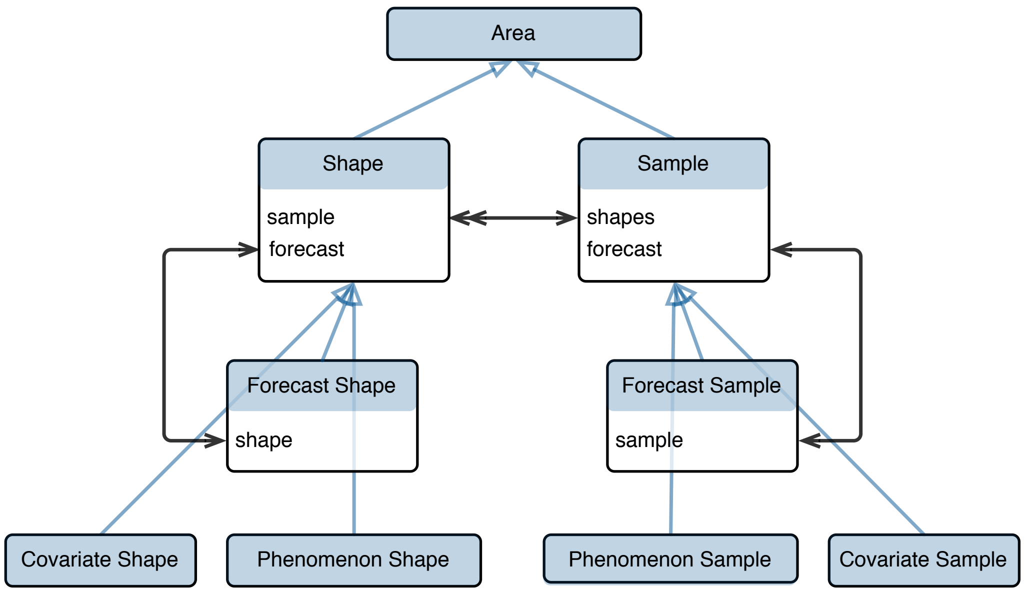


Figure 8.1: Hierarchy of area based objects.

algorithm is highly robust and easily able to track between complex branch and merge conditions as objects join together and break apart. The branching and merging is possible because the tracking is performed iteratively, and the event maintains an estimation of its own centre of mass, which is calculated using the centre of masses of the objects it has previously tracked. This introduces an element of negative feedback that ensures that events maintain their locality and do not meander across the territory.

Covariate data will likely come from different sources, and may be represented differently such as using values recorded by weather stations. Such datasets will likely be used to analyse more than one phenomenon; they are stored in a database so it may be queried to match the temporal support of the phenomenon dataset. This ensures that both phenomenon and covariate data may be indexed using the same time samples. If the covariate data are returned as a set of values distributed randomly across the territory, the framework will interpolate them automatically and register the distribution onto the same Cartesian coordinate system as the phenomenon images. Again, these *covariate samples* inherit from Area objects, and therefore have their own shape descriptions comparable to the phenomenon samples.

Covariate objects maybe be computed using two methods, either by "cookie-cutting" them based on the position and size of the equivalent phenomenon object, or by performing the same object detection strategy deployed on the input phenomenon data. Once covariate objects have been created this produces two sets of time series objects which may be analysed together. The covariate time series was extended to cover a phenomenon season, a time period predefined by each phenomenon type. This facilitates analysis of the event in a wider temoral context.

Analysis was performed using cross-correlation to search for similarities between the larger covariate time series and the phenomenon series. While the results have not provided an instant indication of why a phenomenon event starts, nor forecasting of its future activity, they have confirmed assumptions made about the relationships between covariates and phenomena, and additionally shown that by selecting a shorter time period of the phenomenon, forecasting will likely be possible. An indication of onset can be provided by considering the rate of change of the covariate, particularly with regard to air temperature.

Throughout this research, no assumptions about the nature of the study phenomenon, or available covariates were made. Similarly the assumptions made regarding the form of the datasets have been minimal, principally the size of the input images, and the weekly sampling rate are the only assumptions made. This has been to ensure that the framework developed is general and may be applied to any spatiotemporal datasets.

8.1 Shape Discovery

Figure 8.2 shows two significant processes that are executed frequently, creating samples from the input data, and tracking shapes. Samples are created from the input data and subsequently stored as objects on disc. Tracking shapes triggers either the retrieval or creation of samples. The red bordered step, Shape Discovery, is where shapes are created and marked for tracking, as discussed in Section 6.1. These marked shapes are the objects of interest and all further analysis uses them, therefore, it is imperative that the marked shapes represent the phenomenon's measured property faithfully.

In general the shape discovery works very well, shapes of interest are those that have

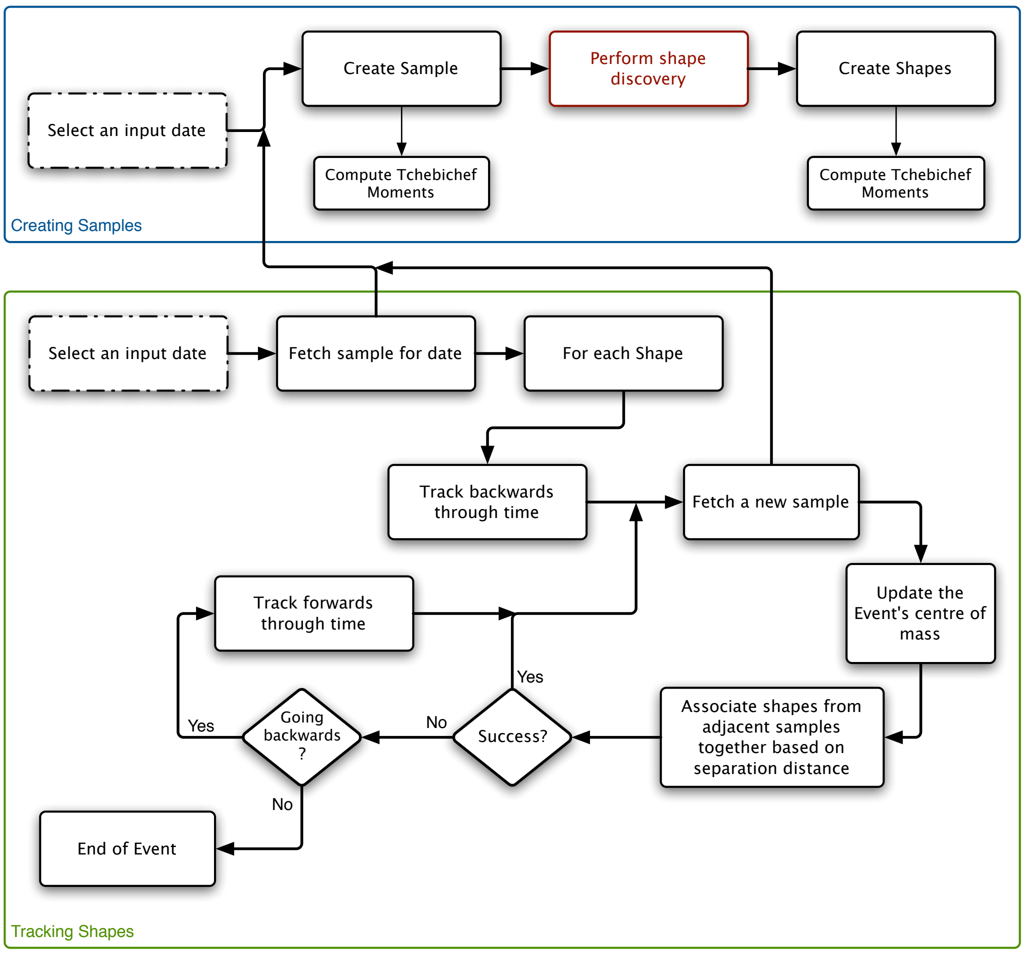


Figure 8.2: Tracking Shapes and Creating Samples.

the highest intensity and largest size. Additionally, their extraction is automatic without any human calibration or intervention, nor does the algorithm require any parameters or initialisation. However, there is a single case where the returned shapes are not ideal. If a shape contains more than one child shape (of a higher intensity) in its hierarchy, then it will not be marked for tracking, and instead one or more of the child shapes will be marked. Essentially, the algorithm will always choose intensity over area, and in some cases, where the higher intensity shapes are small, this fragments the data unnecessarily. An example of this, is the first shape from the 1998 41 event, shown in Figure 6.6f. This problem could likely be solved by considering the combined area of child shapes as a fraction of their parent shape. If it is below a particular threshold then the parent is marked for tracking. However, determining this threshold will require empirical testing, and will most likely vary between different datasets, resulting in a loss of generality.

8.2 Orthonormal Tchebichef Moments

A significant contribution of the research was to leverage the spatial distribution of the study phenomenon to forecast future activity and correlate with covariate data. Chapter 4 discussed in detail the various steps of this process, and it is evident that at this stage the final results show great promise they are not yet ideal. However, unacceptable errors are not a result of inadequacies in the general concept of shape description, but most likely a result of simplistic estimation models and unpredictable disease characteristics.

8.2.1 Shape Description

As discussed in Section 4.5, discrete orthogonal moments offer far superior shape description to Cartesian or continuos orthogonal moments. Orthogonal moments enable complete description using a finite set of moments, and a discrete formulation minimises the numerical errors introduced during implementation.

The size and position of the shapes is intrinsically important, when considering how a series of shapes change over time, within a larger spatial context. Hence, basica Tchebichef Moments were used that are not invariant to size, scale and rotation, despite invariant version being developed [42, 44, 73]. However, to be able to compare sets of moments from different shapes, there must be the same number of moments, which equates to the shapes having the same pixel dimensions. For the Sentiweb Influenza dataset, this often resulted in the Tchebichef Moments of all shapes and samples being computed over a 382×382 pixel area, wasting significant disc space, and consuming extra CPU cycles. In retrospect, using scale invariant Tchebichef Moments and computing all moments over a predefined size, would have removed the need for some phenomenon specific program logic, and ultimately provided a more elegant framework. The spatial properties: size, position, mass and centre of mass, of every shape was computed at creation time using standard geometric moments, and were analysed in tandem with the Tchebichef Moments. This means that the non-invariant properties of the Tchebichef Moments were not necessary.

8.2.2 Forecasting Shapes

Section 6.7 discussed the forecasting of the phenomenon's measured property based on the modelling, estimation and reconstruction of Tchebichef Moments. Again, the success of this process first relies on accurate shape discovery, for the Tchebichef Moments of these shapes form the basis of the estimation models. Each Tchebichef Moment, $T_{p,q}$ was modelled separately using simple linear models. The best model to use was chosen by computing the coefficient of determination, R^2, for each model and selecting that where $R^2 \rightarrow 1$. Clearly, as the higher order models have more degrees of freedom, they will always fit the data better than lower ordered models for an over-determined system. However, they also respond to noise more keenly, and consequently estimates using cubic or higher models fluctuate wildly. Moment estimation could have been improved by using more sophisticated models and regression techniques such as locally weighted scatterplot smoothing (LOESS), which would allow a greater range of time samples to be used during regression.

Additionally, estimated shapes feature artefacts of smaller shapes in them. This is a consequence of the ordinal nature of the input data, where pixel intensities are either 24, 39, 84, 119, 169, 235 or 255. Smoothing the estimated images would remove these artefacts, although at the expense of further losses in accuracy.

8.2.3 Boundaries

The Sentiweb Influenza dataset only covers the area of France. Consequently, shapes of disease are artificially limited by the border of the country. Given access to the raw data this would not be an issue, as generated images would not include any borders except for that of the land mass itself. The borders that France shares with Belgium, Germany, Switzerland and Italy can therefore corrupt the shape of the disease objects. Unfortunately due to the resolution of the data, it is not possible to focus on a sub-sample of France and extract enough Phenomenon Events to be worthwhile. Clearly, this problem will be resolved by using data over a larger territorial area, covering continental Europe, for example.

8.3 Covariate Shape Extraction

Chapter 7 discussed the cross-correlation of shapes of the phenomenon's measured property with shapes of covariates. The detection of shape objects from the phenomenon data has been discussed extensively. The extraction of shapes to represent the covariate data is achievable using two methods. The simplest method is to "cookie-cut" a shape from the smoothly interpolated covariate sample, which means using the area of the phenomenon shape as a mask:

$$\text{Covariate Shape}(\tilde{x},\tilde{y}) = \begin{cases} \text{Covariate Sample}(x,y) & \text{Phenomenon Shape}(x,y) > 0 \\ 0 & \text{otherwise} \end{cases}$$

The problem with this approach is that the Tchebichef Moments of the covariate shape are very similar to the equivalent Phenomenon Shape because they have the same position and boundary. In this case only the pixel intensity varies, and the variation might not be statistically significant, as is the case with air temperature. As a result of this the cross-correlation results are clouded by the overriding similarity between the sets of shapes, and the inter-shape variations are lost.

The other approach of shape extraction is to perform the same discovery process as that used for the phenomenon shapes, this is discussed in detail in Section 9.2.

8.4 Data

Ideally, access to the raw data collected by Senitweb would be available. This would provide a value, the number of ILI cases, with a timestamp and geolocation in latitude and longitude. Such data could be merged with similar surveillance operations across Europe to remove boundary discontinuities. Such a dataset would be in exactly the same form as the weather observations, and allow it to be rendered optimally for the framework. However, ideal datasources rarely exist, and the focus of this research has been the development of a general methodology which isn't tailored specifically to any dataset.

The shortcomings of the Senitweb images do impact negatively on the correlation results, and ultimately limit the results of any analysis. These shortcomings are most notably the spatial resolution of the images, which as highlighted in Section 3.1 as 1 pixel to 6.5km^2, meaning that any effective analysis of Paris is not possible. The Sentiweb organisation has recognised this deficiency recently, by providing Influenza images of the Paris regions at a higher resolution.

The normalisation of the data by population density is also detrimental to any analysis. Although computed at the départment administrative level, of which there are 98 in France, reporting the number of cases per 100,000 habitants disfigures the phenomenon's measured property. In areas such as Paris, with a population of over 10 million, the ILI response is effectively zero. However, in sparsely populated areas such as the Auvergne région the ILI response blooms, as seen by the large objects in the middle of images throughout Figures 6.6, 6.7 and 6.8.

Ultimately, the quality of any analysis is dependent on its source data. It is hoped that governments, universities and non-profit organisations that collect data, strive to provide open access to raw data wherever possible. Barak Obama's presidency promises a fresh outlook on science and government transparency, and it is with his Government that open access to raw data has become a reality. Data.gov aims to "increase public access to high value, machine readable datasets generated by the Executive Branch of the Federal Government" [63]. Hopefully other governments will follow President Obama's endeavours in transparency and provide similar solutions[1].

[1]As of writing the United Kingdom's http://data.gov.uk requires a password.

Chapter 9

Further Work

The research documented in this thesis has described a radical new approach for analysing spatiotemporal data, and as with all research that challenges conventions there are many avenues for further research that are yet to be explored. The following section discusses the further applications and refinements to this framework.

9.1 Phenomenon Event Centre Of Mass

As stated in Section 6.6, the centre of mass of the phenomenon event is calculated so that shapes are tracked correctly with respect to their parent events after a branch transition. This is a very important factor in the object tracking algorithm, as the majority of phenomenon events will merge into an object that represents the majority of the study territory. Once this very large shape branches into smaller shapes, the tracking algorithm must match the each shape to its correct event. Incorrect tracking after a branch will result in phenomenon events becoming corrupted.

Calculating the centre of mass of the event as the average of the event's shapes prior to the shape currently being tracked acts to always pull shapes back to the event like elastic. Using only the previous shape means that the anchored end of the elastic is moving just as frequently as the tracking, and is ineffectual, however using an average recalculates the anchor point, so that while its Euclidean displacement varies less significantly over time.

However, this solution is most likely not optimal, using a standard mean average of the centre of masses of each shape gives each shape equal significance. Yet, many events are numerous weeks long, so that the position of shape 15 or 20 weeks previous is just as significant as the most recently previous shape. A simple improvement therefore might be to use a weighted mean, such as

$$\{\tilde{x}, \tilde{y}\} = \left\{ \frac{1}{\sum\limits_{i=t}^{n} W_i} \sum_{i=t}^{n} W_i S_i(x), \frac{1}{\sum\limits_{i=t}^{n} W_i} \sum_{i=t}^{n} W_i S_i(y) \right\} \tag{9.1}$$

for object S_t followed by n other objects. The weight of each shape is determined by its proximity (in time) to the currently tracked week.

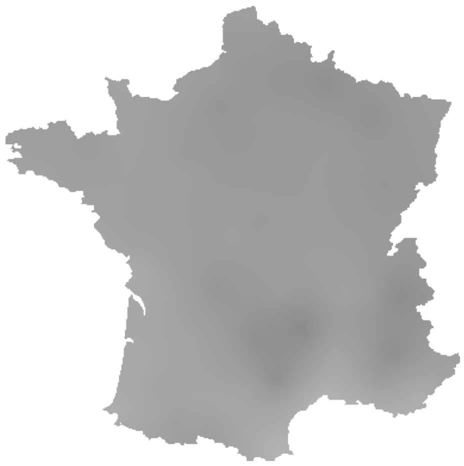
$$W_i = n - i + 1 \tag{9.2}$$

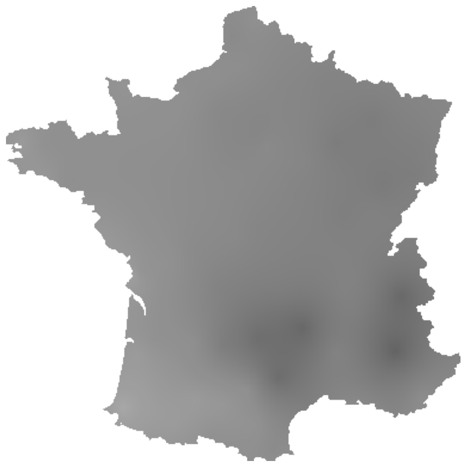
This alteration however actually degrades the tracking compared to a standard mean, because the position.

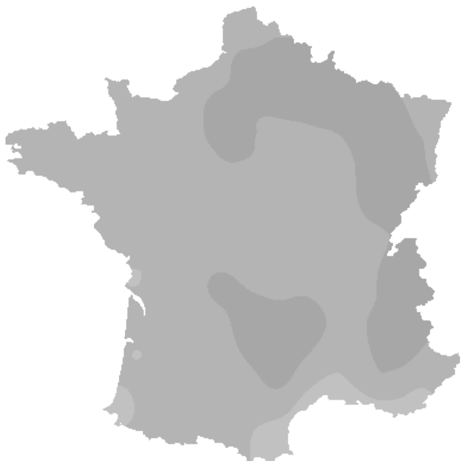
9.2 Covariate Shapes

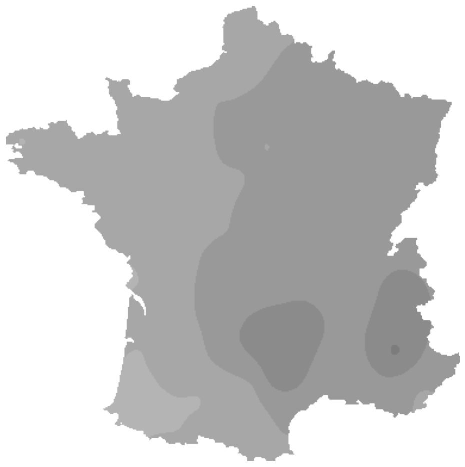
Covariate Shapes can be extracted directly from the interpolated covariate sample rather than cookie-cut from the sample using the spatial attributes of Phenomenon Shapes. However, before shape detection can be performed, the interpolated Covariate Sample must be quantised to reduce the number of intensity levels. L equal width intensity bands should be dependent on the expected maximum and minimum values of the covariate type over the entire dataset, rather than local maxima and minima. This will provide a consistent granularity of intensity levels. This is demonstrated in Figure 9.1

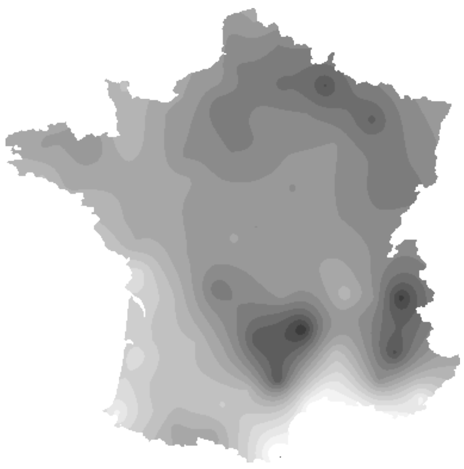
Once a Covariate Sample has been quantised, shapes may be detected using the same algorithms used for detecting Phenomenon Shapes. It is not necessary to track these covariate shapes, instead, during analysis, covariate shapes closest to phenomenon shapes will be used for cross-correlation.


(a) Interpolated Air Temperature 1998/39


(b) Interpolated Dew Point 1999/11


(c) Globally quantised Air Temperature 1998/39


(d) Globally quantised Dew Point 1999/11


(e) Locally quantised Air Temperature 1998/39

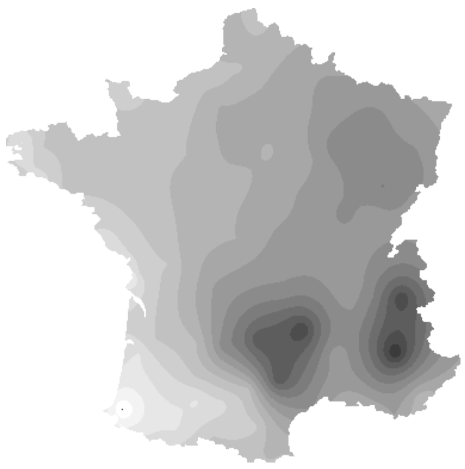

(f) Locally quantised Dew Point 1999/11

Figure 9.1: Quantising interpolated sampled spatial data

9.3 Population Density

The density of a population is incredibly important when analysing the dynamics of diseases. Simply put, the closer together people are, the higher the probability of a transmission from an infected individual to a susceptible individual. This notion is formalised in the Susceptible Infectious Removed (SIR) model as the β term. The SIR model [40] labels each member of the population, or compartment, as either susceptible (S), infectious (I) or removed (R). Generally it is assumed that the total number of people in the population remains constant, people move from the susceptible group, to the infected group and finally to the removed group. Birth and death rates are ignored because usually the lifetime of an epidemic is so short compared to the dynamics of the population. Therefore $S(t) + I(t) + R(t) =$ constant, C. The rates of change between the groups is straightforward, between S and I the transition rate is βI. Where β is the probability of a contact from an infected person leading to a disease infection. Between I and R the transition rate is ν, the rate of recovery, which is simply the reciprocal of the duration of infection. Given these transition rates, the models of each group are as follows:

$$\frac{\mathrm{d}S}{\mathrm{d}t} = -\beta IS \tag{9.3a}$$

$$\frac{\mathrm{d}I}{\mathrm{d}t} = \beta IS - \nu I \tag{9.3b}$$

$$\frac{\mathrm{d}R}{\mathrm{d}t} = \nu I \tag{9.3c}$$

From these equations, it follows that the dynamics of the infectious class depends on the ratio $R_0 = \beta/\nu$, known as the *basic reproduction number*, which itself is dependent on the population density of the compartment under study.

The images available from Sentiweb display the number of cases of ILI per 100,000 habitants. This division is performed using population density figures for the 98 administrative departments in France, and effectively normalises the ILI cases.

Population density would be a useful covariate for the analysis of Influenza as it directly influences β in the above model. Access to population density figures from census data is only available publicly at the very course région level, as dissemination of the population density of smaller administration levels poses a security risk.

9.4 Forest Fires: Further Phenomenons

The devastation caused by wildfires in California in recent years has garnered significant media attention. Similar wildfires have occurred in Australia and Europe in 2007, 2008 and 2009. Detection of such fires is the subject of significant research efforts, most notably by Roy *et. al.* [55, 56, 57], in collaboration with National Aeronautics and Space Administration (NASA) and United States Geological Survey (USGS). The major advances in satellite mapping of fire affected areas is due to the latest generation of moderate-resolution imaging spectroradiometer (MODIS) systems, which enable robustly calibrated, atmospherically corrected, cloud-screened geolocated data to be generated. In November 2009, the MODIS Burned Area Product (MCD45) is being generated for data from the year 2000 onwards [5].

The product asserts the existence of fire at a given location, identified by pixel position, and time, identified by the pixel intensity which indicates the day of the year. The globe is divided into tiles of approximately 10 degrees by 10 degrees in size. In addition to the burn date level, quality assurance of the detection is available in a separate data file. Samples of the product, processed by NASA are shown in Figure 9.2. An example of the GeoTiff images now available from the MCD45 product is shown in Figure 9.3.

Theoretically once a mask image is created this dataset may be used with the framework. However, the extremely large size of the images presents an implementation problem. Resizing the image returns stability to the framework's algorithms, but reduces the accuracy considerably. But this is not a suitable option the pixel values denote temporal position, therefore resizing corrupts the data. Additionally, the burnt areas are very small, in relation to the image size, the burned areas are, as seen by the tiny red pixels in Figure 9.2, so resizing the image will likely completely erase the burnt area pixels. Instead, analysis should be done in a patchwork fashion on small subsamples of the data.

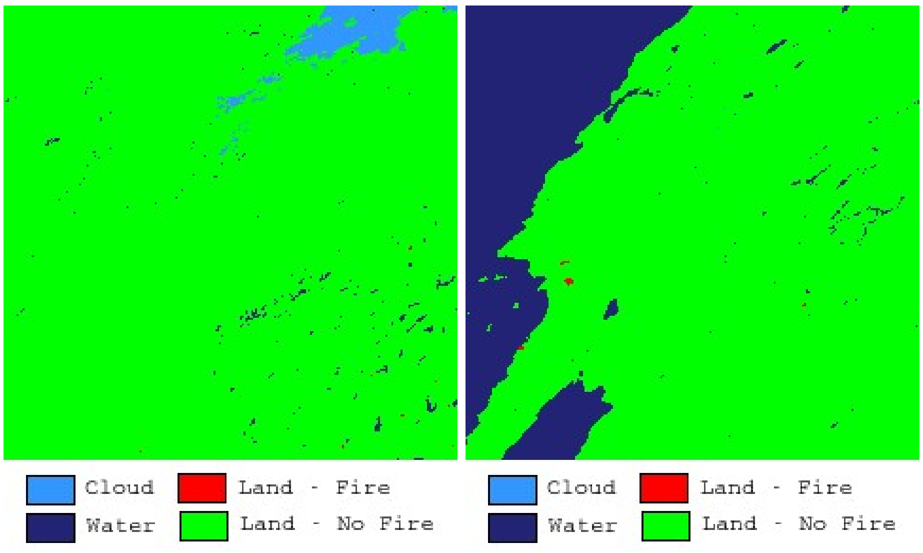


Figure 9.2: Examples of MODIS Burned Area Product

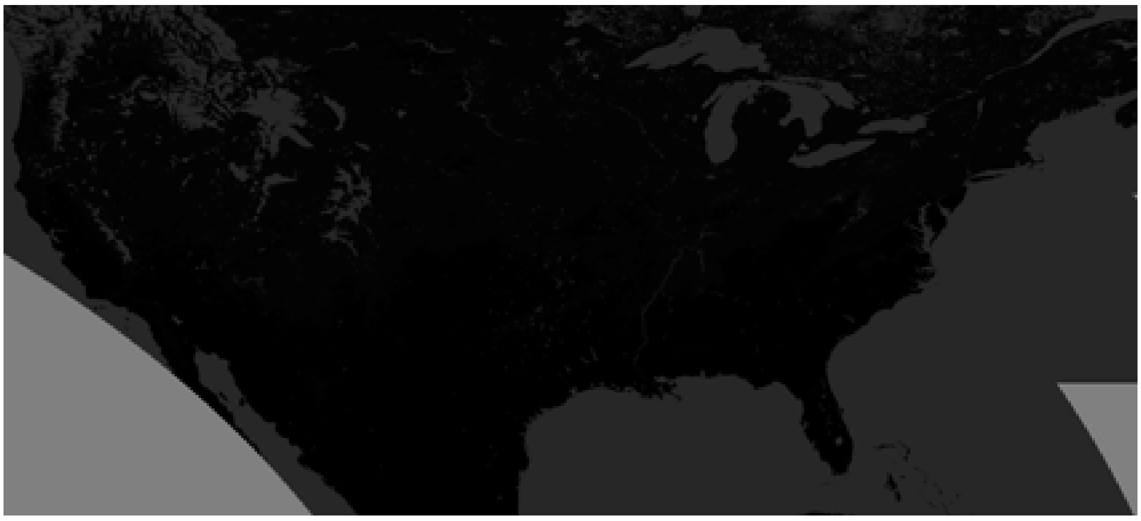


Figure 9.3: Sample GeoTiff MODIS Burned Area Product, from window 3, March 2008. The images are $13,654 \times 6144$ pixels in size and $\approx$ 160MB.

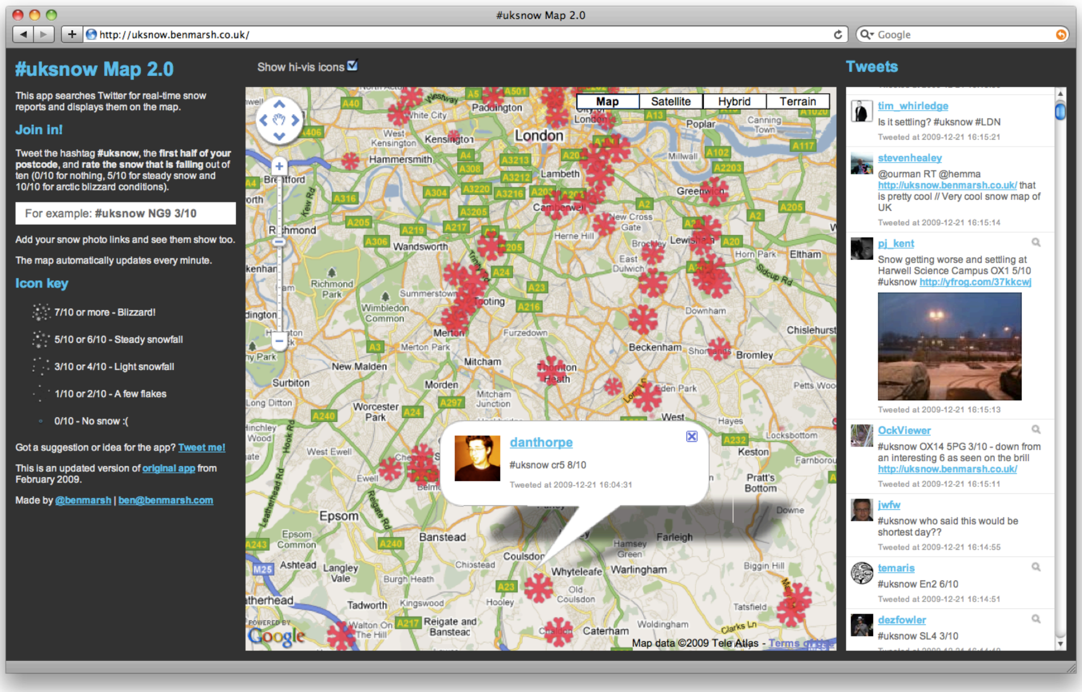
9.5 Real Time Surveillance

This research has developed an object-based framework for analysing spatiotemporal properties of phenomena using historial data. The primary source has been weekly surveillance data of Influenza Like Illness from volunteering physicians across France. This system is typical of surveillance networks around the globe, where data is collected on a weekly basis, and the published after a one or two week delay. It is most certainly not real-time, although might be considered "near real-time".

The Internet, and technologies founded on it, provides a network that is used for real-time reporting of numerous events. In November 2009, Twitter[1] enabled Global Positioning System (GPS) data, or a *Geotag*, to be attached to tweets[2][58], when issued by clients using its application programming interface (API). Another Twitter construct, the *hashtag*, allows for information pertaining to the same topic to be easily grouped together for searching. Hence, community driven surveillance, that is both spatially referenced and delivered in real-time is possible. In December 2009 this technology was used in a web-based application called #uksnow, to display snow fall in the UK on Google Maps using Twitter data. Users need only to post a tweet containing the hashtag "#uksnow", include their postcode and a rating of the snow fall in their area. The resultant application including map is shown in Figure 9.4. Google.org hosts a similar project for highlighting Influenza trends from their users' search terms called Google Flu Trends [22].

There are issues with crowd-sourcing information, such as ensuring a consistent measurement of the topic of surveillance and guaranteeing coverage in remote locations. A simple score out of 10 with a clear key is used for the #uksnow application. Although not all users are able to follow instructions and observations may be invalid, such as "V impressed with our bus driver and buses generally cruising through loads of snow in SE London #uksnow"[3], which does not include a rating of the snow. Ultimately however, this type of surveillance will not replace current weather surveillance that is performed by

[1]http://twitter.com A popular social network.
[2]A "tweet" is a single 140 character message posted by a Twitter user.
[3]http://twitter.com/Fundamentals/status/6899345276



#uksnow Map 2.0

This app searches Twitter for real-time snow reports and displays them on the map.

Join in!

Tweet the hashtag **#uksnow**, the **first half of your postcode**, and **rate the snow that is falling** out of ten (0/10 for nothing, 5/10 for steady snow and 10/10 for arctic blizzard conditions).

For example: **#uksnow NG9 3/10**

Add your snow photo links and see them show too.

The map automatically updates every minute.

Icon key

7/10 or more - Blizzard!
5/10 or 6/10 - Steady snowfall
3/10 or 4/10 - Light snowfall
1/10 or 2/10 - A few flakes
0/10 - No snow :(

Got a suggestion or idea for the app? Tweet me!

This is an updated version of original app from February 2009.

Made by @benmarsh | ben@benmarsh.com

danthorpe
#uksnow cr5 8/10
Tweeted at 2009-12-21 16:04:31

Tweets

tim_whirledge
Is it settling? #uksnow #LDN
Tweeted at 2009-12-21 16:15:21

stevenhealey
@ourman RT @hemma http://uksnow.benmarsh.co.uk/ that is pretty cool // Very cool snow map of UK
Tweeted at 2009-12-21 16:15:14

pj_kent
Snow getting worse and settling at Harwell Science Campus OX1 5/10 #uksnow http://yfrog.com/37kkcwj
Tweeted at 2009-12-21 16:15:13

OckViewer
#uksnow OX14 5PG 3/10 - down from an interesting 6 as seen on the brill http://uksnow.benmarsh.co.uk/
Tweeted at 2009-12-21 16:15:11

jwfw
#uksnow who said this would be shortest day??
Tweeted at 2009-12-21 16:14:55

temaris
#uksnow En2 6/10
Tweeted at 2009-12-21 16:14:51

dezfowler
#uksnow SL4 3/10

Figure 9.4: Twitter sourced real-time snow surveillance, showing my surveillance contribution at the time of writing.

specialist equipment to report quantitative measurements that are both reliable, and most importantly, consistent.

The framework developed by this research currently assumes data is either stored on disk as digital images, or in a MySQL database. Pertinent information, such as login credentials and field names, that are required to access data stored in a database must be provided, so that the framework can work with any MySQL database. This principle could be extended to augment the data retrieval capabilities of the framework. By specifying search terms, such as `http://twitter.com/#search?q=%23uksnow`, real-time data sources for covariates or phenomenon properties could be available. Data retrieved from such a source would require post-processing, but effectively would contain a value, a location and a time stamp, which is exactly the format of the weather data currently in use by the framework.

References

[1] D. Adalsteinsson and J. A. Sethian. A fast level set method for propagating interfaces. *Journal of Computational Physics*, 118(2):269–277, May 1995.

[2] M. Armstrong. *Basic Linear Geostatistics*. Springer, 1998.

[3] H. Bateman, A. Erdélyi, W. Magnus, F. Oberhettinger, and F. G. Tricomi. *Higher Transcendential Functions*, volume 3. New York, McGraw-Hill, 1955.

[4] S. O. Belkasim, M. Ahmadi, and M. Shridhar. Efficient algorithm for fast computation of zernike moments. *Jounal of the Franklin Institute*, 333(4):577–581, July 1996.

[5] L. Boschetti, D. Roy, and A. A. Hoffmann. *MODIS Collection 5 Burned Area Product (MCD45)*, November 2009.

[6] I. Bouchrika, M. Goffredo, J. Carter, and M. Nixon. Covariate analysis for view-point independent gait recognition. In *International Conference on Biometrics*, 2009.

[7] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, November 2001.

[8] British Broadcasting Corporation. *Many wards closed by vomiting bug*, 2007. `http://news.bbc.co.uk/1/hi/health/7170948.stm`.

[9] British Broadcasting Corporation. *Californian Forest Fires*, July 2009.

[10] British Broadcasting Corporation. *Scots swine flue cases confirmed*. British Broadcasting Corporation, April 2009.

[11] Center for Disease Control. *Outbreak of Swine-Origin Influenza A (H1N1) Virus Infection, Mexico, March - April 2009*, April 2009. `http://www.cdc.gov/mmwr/preview/mmwrhtml/mm58d0430a2.htm`.

[12] M. Chan. *Director-General Statement following the fifth meeting of the Emergency Committee*. World Health Organization, September 2009. `http://www.who.int/csr/disease/swineflu/5th$_$meeting$_$ihr/en/index.html`.

[13] M. Chan. *Swine Influenza*. World Health Organization, April 2009. `http://www.who.int/mediacentre/news/statements/2009/h1n1_20090425/en/index.html`.

[14] T. F. Chan and L. A. Vese. Active contours without edges. *IEEE Transactions on Image Processing*, 10(2):266–277, February 2001.

[15] J.-P. Chilés and P. Delfiner. *Geostatistics: Modelling Spatial Uncertainty*. Wiley InterScience John Wiley & Sons, Inc., 1999.

[16] C.-W. Chong, P. Raveendran, and R. Mukundan. A comparative analysis of algorithms for fast computation of zernike moments. *The Journal of the Pattern Recognition Society*, 39(3):731–742, March 2003.

[17] C.-W. Chong, P. Raveendran, and R. Mukundan. Translation invariants of zernike moments. *The Journal of the Pattern Recognition Society*, 36(8):1765–1773, August 2003.

[18] C.-W. Chong, P. Raveendran, and R. Mukundan. Translation and scale invariants of legendre moments. *The Journal of the Pattern Recognition Society*, 37(1):119–129, January 2004.

[19] Columbia University Press, editor. *[malaria] The Columbia Encyclopedia*. New York: Columbia University Press, 6th edition, 2001–04.

[20] N. Cressie. *Statistics For Spatial Data*. John Wiley, 1993.

[21] M. Donoser and H. Bischof. Efficient maximally stable extremal region (mser) tracking. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1:553–560, 2006.

[22] J. Ginsberg, M. H. Mohebbi, R. S. Patel, L. Brammer, M. S. Smolinski, and L. Brilliant. Detecting influenza epidemics using search engine query data. *Nature*, 457, February 2009.

[23] J. Gu, H. Shu, C. Toumoulin, and L. Luo. A novel algorithm for fast computation of zernike moments. *The Journal of the Pattern Recognition Society*, 35(12):2905 – 2911, December 2002.

[24] Health Protection Agency. *Health Protection Agency issues advice on controlling norovirus*, 2007. http://www.hpa.org.uk/eastofengland/press/071120_norovirus.htm.

[25] M.-K. Hu. Visual pattern recognition by moment invariants. *Information Theory, IEEE Transactions on*, 8(2):179–187, February 1962.

[26] Institut Geographique National. *Projection Systems*. http://www.ign.fr/telechargement/education/fiches/geodesie/projections%_EN.pdf.

[27] C. Kan and M. D. Srinath. Invariant character recognition with zernike and orthogonal fourier–mellin moments. *The Journal of the Pattern Recognition Society*, 35(1):143–154, January 2002.

[28] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. In *International Journal of Computer Vision*, volume 1, pages 321 – 331, 1988.

[29] A. Khotanzad and Y. H. Hong. Invariant image recognition by zernike moments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(5):489–497, May 1990.

[30] H. S. Kim and H.-K. Lee. Invariant image watermark using zernike moments. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(8):766–775, August 2003.

[31] W.-Y. Kim and Y.-S. Kim. A region-based shape descriptor using zernike moments. *Signal Processing: Image Communication*, 16(1-2):95–102, September 2000.

[32] D. Koller, K. Daniilidis, T. Thórhallson, and H. Nagel. Model-based object tracking in road traffic scenes. In *Lecture Notes in Computer Science*, volume 588, pages 437–452. Springer Berlin, Heidelberg, 1992.

[33] E. Korenromp, J. Miller, B. Nahlen, T. Wardlaw, and M. Young. *World Malaria Report 2005*. World Health Organisation (WHO) Roll Back Malaria (RBM) Department, and United Nations Children's Fund (UNICEF), 2005.

[34] S. X. Liao and M. Pawlak. On the accuracy of zernike moments for image analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1358–1364, December 1998.

[35] J. Malcolm, Y. Rathi, and A. Tannenbaum. Multi-object tracking through clutter using graph cuts. In *Non-Rigid Regsitration and Tracking Through Learning, The International Conference on Computer Vision*, 2007.

[36] D. W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Society for Industrial and Applied Mathematics*, 11(2):431–441, June 1963.

[37] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In P. L. Rosin and A. D. Marshall, editors, *BMVC*. British Machine Vision Association, 2002.

[38] G. Matheron. *Traité de géostatistique appliquée*. Editions Technip, France, 1962.

[39] G. Matheron. Principles of geostatistics. *Economic Geology*, 58(8):1246–1266, December 1963.

[40] A. G. McKendrick. Applications of mathematics to medical problems. In *Proceeding of the Edinburgh Mathematical Society*, volume 44, pages 1–34, 1925.

[41] R. Mukundan. Image analysis by tchebichef moments. *IEEE Transactions on Image Processing*, 10(9):1357–1364, September 2001.

[42] R. Mukundan. A new class of rotational invariants using discrete orthogonal moments. In *6th IASTED International Conference on Signal and Image Processing*. International Accociation of Science and Technology for Development, ACTA Press, 2004.

[43] R. Mukundan. Some computational aspects of discrete orthonormal moments. *IEEE Transactions on Image Processing*, 13(8):1055–1059, August 2004.

[44] R. Mukundan. Radial tchebichef invariants for pattern recognition. In *IEEE TENCON 2005*, pages 1–6, Nov 2005.

[45] R. Mukundan and K. R. Ramakrishnan. Fast computation of legendre and zernike moments. *Pattern Recognition Society*, 28(9):1433–1442, September 1995.

[46] R. Mukundan and K. R. Ramakrishnan. *Moment Functions In Image Analysis: Theory and Applications*. World Scientific, 1998.

[47] M. Nixon. *Biometric Gait Tunnel*. Electronics And Computer Science, January 2005.

[48] M. Nixon and A. Aguado. *Feature Extraction And Image Processing*. Newnes, 2002.

[49] OGP Surveying and Positioning Committee. *Coordinate Conversions and Transformations including Formulas*. International Association of Oil and Gas Producers, July 2007.

[50] S. Osher and N. Paragios, editors. *Geometric Level Set Methods in Imaging, Vision, and Graphics*. Springer-Verlag, New York, 2003.

[51] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9(1):62–66, January 1979.

[52] A. Padilla-Vivanco, A. Martinez-Ramírez, and F.-S. Granados-Agustín. Digital image reconstruction by using zernike moments. In J. D. Gonglewski and K. Stein, editors, *Optics in Atmospheric Propagation and Adaptive Systems VI*, volume 5237, pages 281–289. SPIE, 2004.

[53] N. Paragios and R. Deriche. Geodesic active contours and level sets for the detection and tracking of moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(3):266–280, March 2000.

[54] N. Paragios, O. Mellina-Gottardo, and V. Ramesh. Gradient vector flow fast geometric active contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:402 – 407, 2004.

[55] D. Roy, Y. Jin, P. Lewis, and C. O. Justice. Prototyping a global algorithm for systematic fire-affected area mapping using modis time series data. *Remote Sensing of Environment*, 97:137–162, 2005.

[56] D. Roy, J. Ju, P. Lewis, C. Schaaf, F. Gao, M. Hensen, and E. Lindquist. Multi-temporal modis–landsat data fusion for relative radiometric normalization, gap filling, and prediction of landsat data. *Remote Sensing of Environment*, 112:3112–3130, 2008.

[57] D. Roy, P. Lewis, and C. O. Justice. Burned area mapping using multi-temporal moderate spatial resolution data—a bi-directional reflectance model-based expectation approach. *Remote Sensing of Environment*, 83:263–286, 2002.

[58] R. Sarver. *Think Globally, Tweet Locally*. Twitter, November 2009.

[59] Sentinelles Network & Inserm, Institut national de la santé et de la recherche médicale. *Sentiweb: http://www.sentiweb.org*.

[60] J. A. Sethian. *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 2nd edition, 1999.

[61] M. R. Teague. Image analysis via the general theory of moments. *Journal of the Optical Society of America*, 70(8):920–930, August 1979.

[62] D. Thorpe. *On Shape Mediated Analysis of Spatiotemporal Phenomena*, September 2009. `http://bitbucket.org/danthorpe/phd/`.

[63] US Federal Government, http://data.gov. *About Data.gov*, January 2010.

[64] C. Viboud, P.-Y. Boelle, F. Carrat, A.-J. Valleron, and A. Flahault. Prediction of the spread of influenza epidemics by the method of analogues. *American Journal of Epidemiology*, 158(10):996–1006, 2003.

[65] L. Wang and G. Healey. Using zernike moments for the illumination and geometry invariant classification of multispectral texture. *IEEE Transactions on Image Processing*, 7(2):196–203, February 1998.

[66] R. Woodman. Doctors and politicians clash over size of flu problem. *British Medical Journal*, 320:138, 2000.

[67] P. L. Woodworth. Trends in U.K. mean sea level. *Marine Geodesy*, 11(1):57–87, 1987.

[68] World Health Organization. *Pandemic (H1N1) 2008 - update 67*, September 2009. `http://www.who.int/csr/don/2009_09_25/en/index.html`.

[69] World Health Organization. *The Weekly Epidemiological Record, No 21, 2009*, May 2009. `http://www.who.int/wer/2009/wer8421.pdf`.

[70] J. Wuttke. *Levenberg Marquardt Least Squares Fitting Algorithm*, 2004-2008.

[71] C. Xu and J. L. Prince. Gradient vector flow: a new external force for snakes. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 66 – 71, June 1997.

[72] C. Xu and J. L. Prince. Snakes, shapes, and gradient vector flow. *IEEE Transactions on Image Processing*, 7:359–369, 1998.

[73] H. Zhu, H. Shu, T. Xia, L. Luo, and J. L. Coatrieux. Translation and scale invariants of tchebichef moments. *Pattern Recognition*, 40(9):2530–2542, September 2007.

Appendix A

Application Framework Architecture

Figure A.1 describes the framework's entity hierarchy and relationships, it also lists the pertinent attributes of some entities.

The main elements of the framework are explained in further detail, and present a guide to the remainder of this thesis.

Phenomenon Type

The framework generates a Phenomenon Type object for each study phenomenon. It is primarily used as a type to select other phenomenon dependent objects, although it does directy store the phenomenon season and season overlap variables, as shown in the Phenomena Settings in Figure 7.2 on page 76.

Datasource

A Datasource entity encapsulates authentication credentials so that it may retrieve information from a variety of different sources. These sources may be of three different types: MySql database, files stored locally and files stored remotely on an ftp server. The Datasource object stores all the login and authentication information, and in the case of a MySql database is configurable to store field names for spatial dimensions. Each Phenomenon Type has at least one datasource for its own data, additional Datasources provide Covariate data. For the study phenomenon used in this research, there are two Datasource objects, one for the locally stored Sentiweb surveillance images, and one which is a MySql database which provides weather observations used as Covariates. These data-sets are described in further detail in Chapter 3 on page 11.

Covariate Type and *Semivariogram Model*

Similar to a Phenomenon Type, a Covariate Type is a reference to some spatialtemporal phenomenon which we intend to use as an explanatory variable with the study phenomenon. A Covariate Type is provided by a Datasource, which in turn can be linked to any Phenomenon Type, which means that the same covaraite can be used in experiments with any study phenomenon. Each Covariate Type calculates a number of Semivariogram Models using the same temporal support used in the phenomenon data-set, which is a time period provided by the Phenomenon Type. The framework performs automatic fitting of the Semivariogram Models, which are described in Section 5.4 on page 46, although manual fitting is also possible.

Area

An Area entity stores an arbitrary sized object, either on disc in a common image format, or using a simple custom format to allow for signed double precision values. This object also facilitates the calculation of object based properties such as object mass, centre of mass, size, and shape description objects which are described in Chapter 4 on page 15.

Observation

The Observation entity is used to store a single time sample of covariate data. It maintains a set of Point Values, which themselves are geographical Coordinate objects, and can be interpolated to generate an image map registed with the Phenomenon Type. This is achieved using Kriging and is discuess in detail in Section 5.5. The image map is stored using the capabilities of the Observation's parent entity Area.

Phenomenon Sample

The Phenomenon Sample stores a single time sample of the selected phenomenon type, while also providing functionality to fetch prior and future time samples. It performs object isolation and generates a hierarchy of Phenomenon Shape objects, described in detail in Section 6.1 on page 53.

Phenomenon Shape

Phenomenon Shape is also a child of Area, and is used to store all distinct areas of the study phenomenon. Some shapes are marked for tracking depending on the object isolation algorithm discussed in Section 6.1, and these are the shapes that used in by the Phenomenon Event entity.

Observation Shape

Each Observation Shape is created from an Observation object and a Phenomenon Shape object. Given the interpolated image of the covariate (by the Observation) the Phenomenon Shape acts as a mask to extract a shape of the Observation at the same locality. This is discussed further in Section 7.1.

Phenomenon Event

The Phenomenon Event generated by the framework as a result of tracking Phenomenon Shapes, which is discussed extensively in Section 6.5. It is also responsible for fetching and processing Observation objects for the selected Covariate Type, creating (if necessary) Observation Shape objects, and then performing analysis on the objects. This analysis is discussed in Section 7.1.

The entities described above represent the models of the framework. Numerous controller classes provide logic to bring these models together. Custom view objects provide capabilities to query the data, manipulate the resultant objects, and export the analysis results. Only a small portion of this work is described in this thesis, and for full details the reader should refer to the accompanying data archive [62].

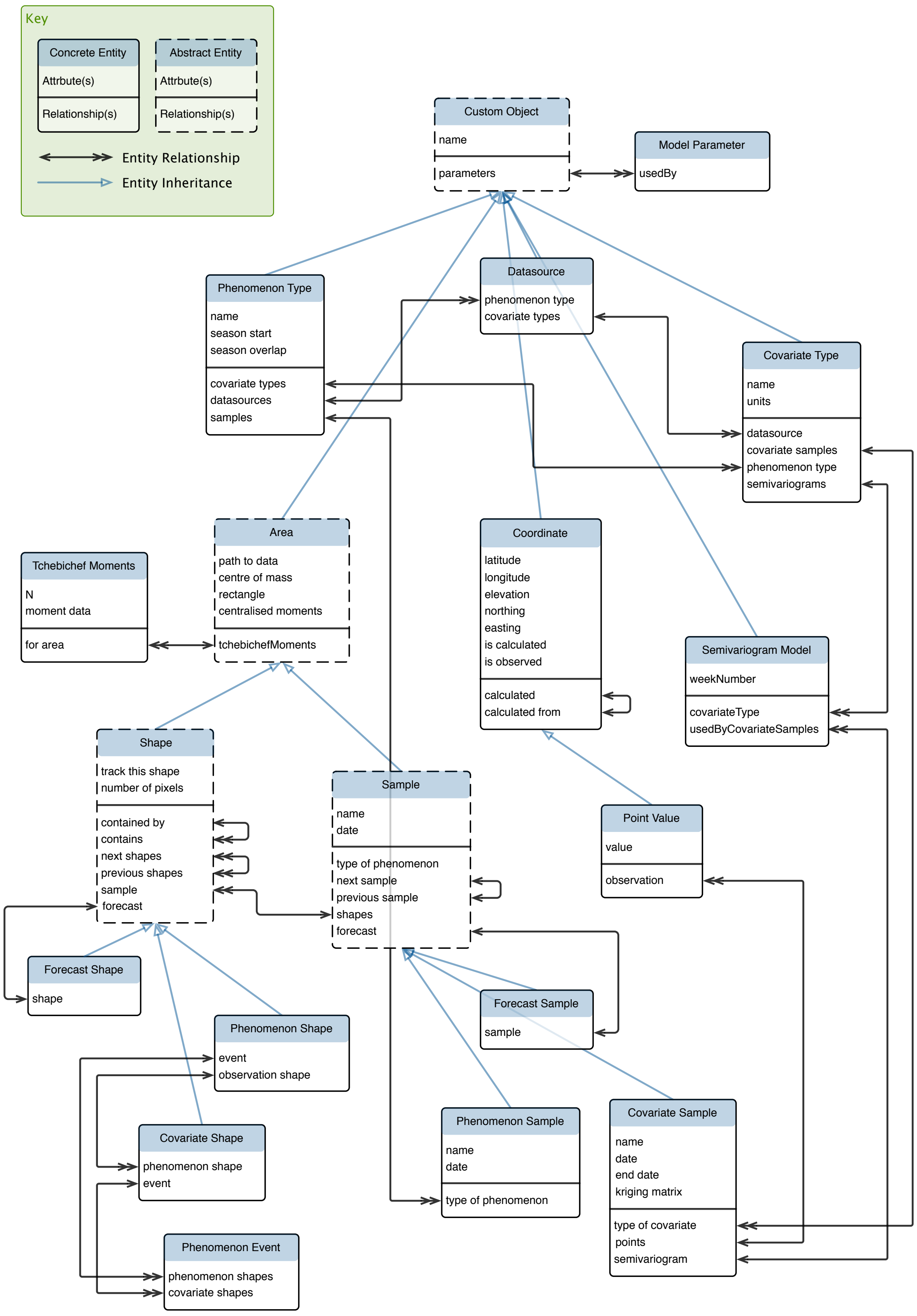


Figure A.1: Framework architecture

Appendix B

Implementation Details

This research features a substantial amount of work, some of which is not immediately apparent. Much of it has been necessary to *massage* the source data so that is can be analysed effectively. This appendix contains some specific implementation details which maybe of interest to the reader wishing to recreate some results or expound on the ideas explained throughout this report.

B.1 Co-ordinate Conversion

To derive Easting and Northing coordinates from geographical coordinates (ϕ, λ) we use the equations

$$\begin{aligned} \text{Easting,} \quad & E = E_F + r \sin\theta \\ \text{Northing,} \quad & N = N_F + r_F + r \cos\theta \end{aligned}$$

where

$$m_1 = \frac{\cos\phi_1}{\left(1 - e^2 \sin^2\phi_1\right)^{0.5}}$$

$$m_2 = \frac{\cos\phi_2}{\left(1 - e^2 \sin^2\phi_2\right)^{0.5}}$$

$$t = \tan\left(\frac{\pi}{4} - \frac{\phi}{2}\right) \Big/ \left[\frac{(1 - e\sin\phi)}{(1 + e\sin\phi)}\right]^{e/2}$$

for t_1, t_2, t_F, t using ϕ_1, ϕ_2, ϕ_F, ϕ respectfully.

$$n = \frac{\ln m_1 - \ln m_2}{\ln t_1 - \ln t_2}$$

$$F = \frac{m_1}{n t_1^n}$$

$$r = aFt^n$$

for r and r_F using t and t_F respectfully.

$$\theta = n(\lambda - \lambda_F)$$

given the previous definitions detailed in Table 5.1. Listing B.1 shows the implementation.

Code B.1: Calculating easting and northing coordinates from latitude and longitude using a LCC2SP projection.

```
- (void)calcEastingAndNorthingFromLat:(double)lat
                               andLon:(double)lon
                            givenDatum:(Datum *)datum
                         andProjection:(Projection *)projection {

  /* First we want to check that the projection type is Lambert Conformal Conic, with 2 standard
      parallels (LCC2SP) */
  if([[projection parameters] objectForKey:@"type"] == @"LCC2SP") {

    /* if lat and/or lon is nil, then we can use the observed values */
    double phi, lambda;

    if((lat == nil) || (lon == nil)) {
      phi = [[observed objectForKey:@"rad_lat"] doubleValue];
      lambda = [[observed objectForKey:@"rad_lon"] doubleValue];
    } else {
      phi = lat*(M_PI/180.0);
      lambda = lon*(M_PI/180.0);
    }

    /* For docs on the maths check out p20 of http://www.epsg.org/guides/docs/G7-2.pdf
    */
    double eBy2 = [datum e] / 2.0;
    double m1 = cos([projection phi1]) / sqrt(1 - ([datum e2]*[projection sin2phi1]));
    double m2 = cos([projection phi2]) / sqrt(1 - ([datum e2]*[projection sin2phi2]));
    double t1 = tan(M_PI_4-(0.5*[projection phi1]))
        / pow(((1-([datum e]*[projection sin_phi1]))
        / (1+([datum e]*[projection sin_phi1]))),eBy2);
    double t2 = tan(M_PI_4-(0.5*[projection phi2]))
        / pow(((1-([datum e]*[projection sin_phi2]))
        / (1+([datum e]*[projection sin_phi2]))),eBy2);
    double tF = tan(M_PI_4-(0.5*[projection phiF]))
        / pow(((1-([datum e]*[projection sin_phiF]))
        / (1+([datum e]*[projection sin_phiF]))),eBy2);
    double n = (log(m1) - log(m2)) / (log(t1) - log(t2));
    double F = m1 / (n*pow(t1,n));
    double rF = [datum a] * F * pow(tF,n);

    /* the following is code specific to calculating the Northings and Eastings */
    double sin_phi = sin(phi);
    double t = tan(M_PI_4-(0.5*phi))
        / pow(((1-([datum e]*sin_phi))
        / (1+([datum e]*sin_phi))),eBy2);
    double r = [datum a] * F * pow(t,n);
    double theta = n*(lambda - [projection lambdaF]);

    double north = [[[projection parameters] objectForKey:@"N0"] doubleValue] + rF - (r * cos(theta)
        );
    double  east = [[[projection parameters] objectForKey:@"E0"] doubleValue] + (r * sin(theta));

    /* in km */
    [calculated setObject:[NSNumber numberWithDouble:north] forKey:@"north"];
    [calculated setObject:[NSNumber numberWithDouble:east] forKey:@"east"];
  }
}
```

Note that it requires objects for the datum and projection, whose attributes store the model parameters. We also require an algorithm to convert easting and northing coordinates back to geographical coordinates. The reverse formulas are

$$\phi = \frac{\pi}{2} - 2\arctan\left\{t'\left[\frac{(1-e\sin\phi)}{(1+e\sin\phi)}\right]^{e/2}\right\}$$
$$\lambda = \frac{\theta'}{n} + \lambda_F$$

where

$$r' = \pm\left\{(E - E_F)^2 + [r_F - (N - N_F)]^2\right\}^{1/2},$$

taking the sign of n

$$t' = \left(\frac{r'}{aF}\right)^{1/n}$$

$$\theta' = \arctan\left\{\frac{E - E_F}{r_F - (N - NF)}\right\}$$

and n, F and r_F are derived as for the forward calculation. Note here that the formula for ϕ necessitates an interative implementation as shown in Listing B.2.

Code B.2: Calculating latitide and longitude coordinates from easting and northing using a LCC2SP projection.

```
- (double)funcPhiFrom:(double)tPhi
               andTPr:(double)tPr
              andEBy2:(double)eBy2
                 andE:(double)e {
  return M_PI_2 - 2*atan(tPr*pow(( (1-(e*sin(tPhi))) / (1+(e*sin(tPhi))) ),eBy2));
}

- (void)calcLatAndLonFromEasting:(double)east
                     andNorthing:(double)north
                      givenDatum:(Datum *)datum
                   andProjection:(Projection *)projection {

  /* First we want to check that the projection type is Lambert Conformal Conic,
    with 2 standard parallels (LCC2SP) */
  if([[projection parameters] objectForKey:@"type"] == @"LCC2SP") {

    /* if east and/or north is nil, then we can use the observed values */
    if((east == nil) || (north == nil)) {
      /* Need to put a try/catch statement in here in case we haven't got any calculated values or
           passed values */
      east = [[calculated objectForKey:@"east"] doubleValue]; /* x1000 to convert them back into
           meters, instead of km */
      north = [[calculated objectForKey:@"north"] doubleValue];
    }

    double eBy2 = [datum e] / 2.0;
    double m1 = cos([projection phi1]) / sqrt(1 - ([datum e2]*[projection sin2phi1]));
    double m2 = cos([projection phi2]) / sqrt(1 - ([datum e2]*[projection sin2phi2]));
    double t1 = tan(M_PI_4-(0.5*[projection phi1]))
        / pow(((1-([datum e]*[projection sin_phi1]))
        / (1+([datum e]*[projection sin_phi1]))),eBy2);
    double t2 = tan(M_PI_4-(0.5*[projection phi2]))
        / pow(((1-([datum e]*[projection sin_phi2]))
        / (1+([datum e]*[projection sin_phi2]))),eBy2);
    double tF = tan(M_PI_4-(0.5*[projection phiF]))
        / pow(((1-([datum e]*[projection sin_phiF]))
      / (1+([datum e]*[projection sin_phiF]))),eBy2);
    double n = (log(m1) - log(m2)) / (log(t1) - log(t2));
    double F = m1 / (n*pow(t1,n));
    double rF = [datum a] * F * pow(tF,n);

    // Now the below code is specific to reverse Easting and Northing
    double E0 = [[[projection parameters] objectForKey:@"E0"] doubleValue];
    double N0 = [[[projection parameters] objectForKey:@"N0"] doubleValue];
    double rPr = sqrt( pow((east-E0),2) + pow((rF-(north-N0)),2) );
    if(n < 0) { rPr *= -1; }
    double tPr = pow((rPr / ([datum a]*F)),(1/n));
    double thetaPr = atan( (east-E0) / (rF-(north-N0)) );
    double tPhiOld = M_PI_2 - 2*atan(tPr);

    double phi = [self funcPhiFrom:tPhiOld andTPr:tPr andEBy2:eBy2 andE:[datum e]];
    while(phi != tPhiOld) {
      tPhiOld = phi;
      phi = [self funcPhiFrom:tPhiOld andTPr:tPr andEBy2:eBy2 andE:[datum e]];
    }

    double lambda = thetaPr/n + [projection lambdaF];
    double lat = phi*(180/M_PI);
    double lon = lambda*(180/M_PI);
    [calculated setObject:[NSNumber numberWithDouble:lat] forKey:@"lat"];
    [calculated setObject:[NSNumber numberWithDouble:lon] forKey:@"lon"];
  }
}
```

B.2 Object Isolation

The algorithm developed to isolate areas of diease activity, so that they may be analysed removed from their surroundings is conceptually very simple. A human being can easily understand that there are 3 objects of significance in Figure B.1. But of course this spatial connectivity between adjactent pixels of the same intensity is an alien concept to a computer.

The method chosen to solve this problem is essentially a recursive neighbourhood search as shown in Figure B.2. The algorithm works by iterating through the input image, and inspecting each pixel. Every pixel tested is set to a non-zero integer in a result image of the same dimension. The number corresponds to either 1 indicating that the pixel has been tested and is not part of a shape, or an integer greater than 1 which corresponds to a shape. Upon encountering a pixel belonging to a new shape, the pixel is added to an otherwise empty queue. While there are still pixels in the queue, the first pixel is added to

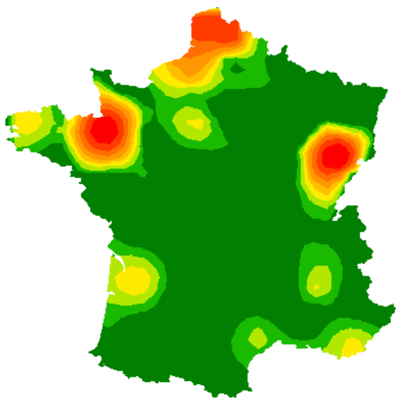


Figure B.1: Developing an algorithm to isolate these objects from their surroundings.

the current shape, and the 8 neighbourhood pixels are added to the queue if they belong to the shape. Then the pixel is removed from the queue. In this way, the algorithm explores the neighbourhood following connected pixels until it reaches the boundaries of the shape. Upon exiting this while loop, the algorithm continues to iterate through the image, the pixels which are part of the shape have now been tested and are ignored. Therefore, each pixel is only considered once. It is important to note that the recursion is done using a queue and in a systematic fashion. We add the 8 neighbourhood pixel rather than exploring in the 8 directions because otherwise it is very easy to cause a stack overflow.

Code B.3: Object isolation through exploratory search

```
// Get a pointer to the start of the actual bitmap data
unsigned char *img = [imgRep bitmapData];
// create a result image space
unsigned int *resultImage = (unsigned int *)malloc(sizeof(unsigned int)*dim.width*dim.height);
// Variable to store the shape number, we start at 2 (1 indicated tested but not in a shape)
unsigned int group = 2;
// Create a pixel queue
queue<long> pixelQueue;
int pivot = [[params objectForKey:@"pivot"] intValue];
unsigned int row, col, width, height, length;
width = dim.width;
height = dim.height;
length = width * height;

// Iterate over the image domain
for(row=0; row<height; row++) {
  for(col=0; col<width; col++) {

    unsigned int index = (row * width + col);
```

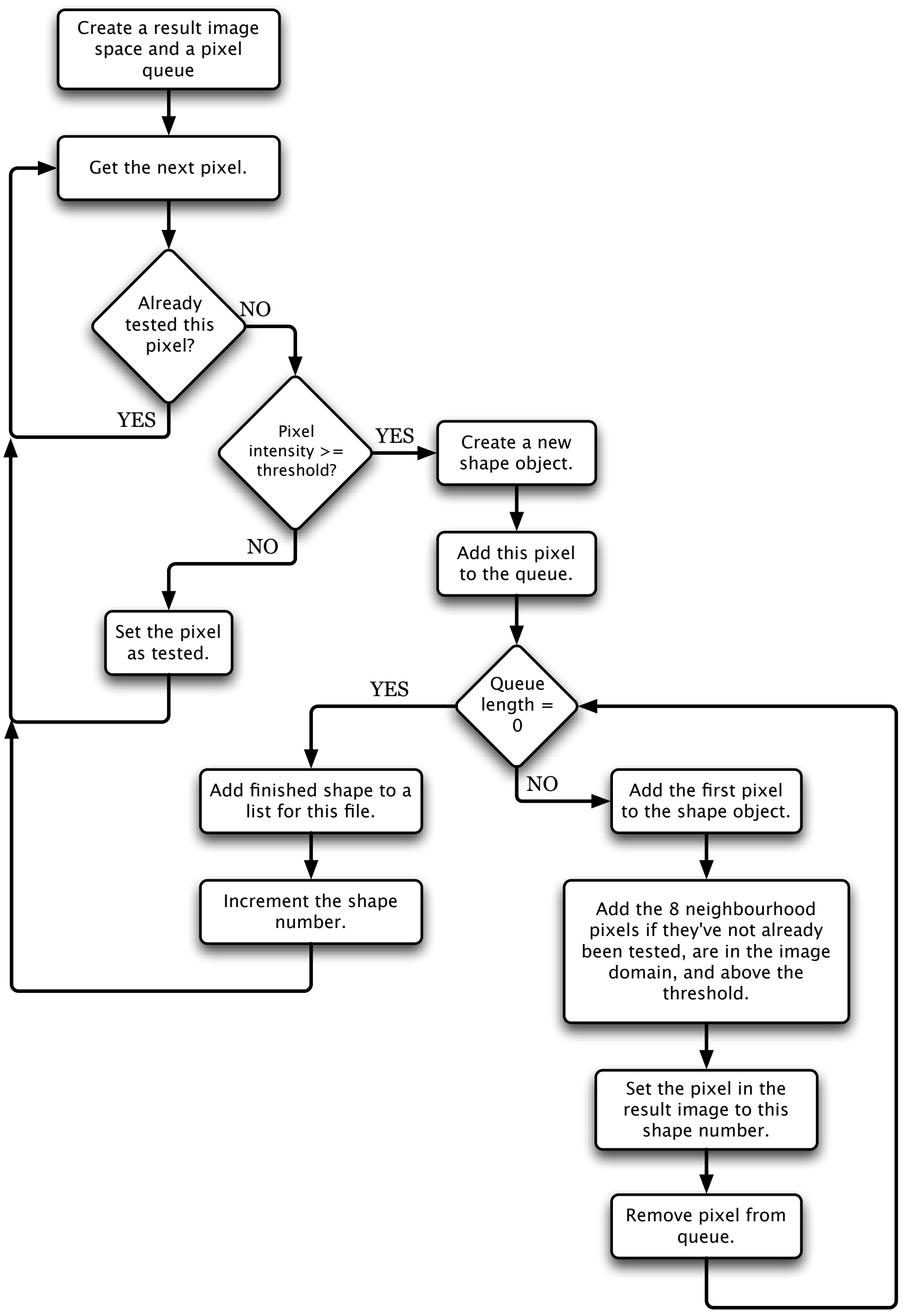

Create a result image space and a pixel queue
Get the next pixel.
Already tested this pixel?
NO
YES
Pixel intensity >= threshold?
YES
NO
Create a new shape object.
Add this pixel to the queue.
Set the pixel as tested.
Queue length = 0
YES
NO
Add finished shape to a list for this file.
Increment the shape number.
Add the first pixel to the shape object.
Add the 8 neighbourhood pixels if they've not already been tested, are in the image domain, and above the threshold.
Set the pixel in the result image to this shape number.
Remove pixel from queue.

Figure B.2: Flow diagram of object isolation algorithm.

```
// If the intensity of this pixel is greater than the pivot it's part
//  of a group. Also check that we haven't already tested this pixel
if( (img[index] >= pivot) && (resultImage[index] == 0) ) {

  // This is a new shape, so we'd better create a Shape object
  Shape *shp = [[Shape alloc] initWithSize:dim
                     withLabel:[NSString stringWithFormat:@"%4d%02d_%3@_%02d",
                       year, week, [params objectForKey:@"withKey"], group-2]];
```

```
// Push the pixel onto the queue
pixelQueue.push(index);
// Enter a loop to recursively (using the queue) search the
//  local neighbourhood
while(pixelQueue.size() != 0) {
  // Get the first pixel off the queue
  unsigned int pixel = pixelQueue.front();

  // Add the pixel to the shape object
  [shp addPixelAtIndex:pixel with:img[pixel]];

  unsigned int r, c;
  r = pixel/width;
  c = pixel%width;

  // Remove the first pixel from the queue
  pixelQueue.pop();
  // Set the group number
  resultImage[pixel] = group;

  // Work out the 8 neighbourhood pixels
  unsigned int north = pixel - width;
  unsigned int northwest = north - 1;
  unsigned int northeast = north + 1;
  unsigned int west = pixel - 1;
  unsigned int east = pixel + 1;
  unsigned int south = pixel + width;
  unsigned int southwest = south - 1;
  unsigned int southeast = south + 1;

  // Push the neighbourhood pixels onto the queue if, they are:
  //  - within image bounds
  //  - intensity of input is >= pivot
  //  - haven't been tested
  if( (resultImage[north] == 0) && (img[north] >= pivot) && (r > 0) ) {
    resultImage[north] = 1;
    pixelQueue.push(north);
  }
  if( (resultImage[northwest] == 0) && (img[northwest] >= pivot) && (r > 0) && (c > 0) ) {
    resultImage[northwest] = 1;
    pixelQueue.push(northwest);
  }
  if( (resultImage[northeast] == 0) && (img[northeast] >= pivot) && (r > 0) && (c < width) ) {
    resultImage[northeast] = 1;
    pixelQueue.push(northeast);
  }
  if( (resultImage[west] == 0) && (img[west] >= pivot) && (c > 0) ) {
    resultImage[west] = 1;
    pixelQueue.push(west);
  }
  if( (resultImage[east] == 0) && (img[east] >= pivot) && (c < width) ) {
    resultImage[east] = 1;
    pixelQueue.push(east);
  }
  if( (resultImage[southwest] == 0) && (img[southwest] >= pivot) && (r < height) && (c > 0) )
      {
    resultImage[southwest] = 1;
    pixelQueue.push(southwest);
  }
```

```
        if( (resultImage[south] == 0) && (img[south] >= pivot) && (r < height) ) {
          resultImage[south] = 1;
          pixelQueue.push(south);
        }
        if( (resultImage[southeast] == 0) && (img[southeast] >= pivot) && (r < height) && (c < width
            ) ) {
          resultImage[southeast] = 1;
          pixelQueue.push(southeast);
        }
      } // End of the while loop

      // We can now add this Shape to the list of shapes for this file
      [[shapesAtThreshold objectForKey:[params objectForKey:@"withKey"]] addObject:shp];
      // Release the shape object
      [shp release];
      // Increment the group number, to continue iterating through the image
      ++group;
    } else if(resultImage[index] == 0) {
      resultImage[index] = 1;
    } // End of if statements
  } // End of column iteration
} // End of row iteration
```

B.3 Orthonormal Tchebichef Moments

In order to compute orthonormal Tchebichef moments, first the calculation of the Tchebichef polynomial is required. Once this is done, the moments are computed by simply iterating over the Tchebichef moment space and calculating the moment function given in Equation (4.32) which is repeated below

$$T_{pq} = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \hat{t}_p(x) \hat{t}_q(y) f(x,y)$$

$$p, q = 0, 1, \ldots, N-1$$

B.3.1 Orthonormal Tchebichef Polynomial

The framework includes a `TchebichefPolynomial` class, which contains a convenience method to return a polynomial for a given size. The size of the polynomial, is referred to throughout the framework as `N`. Once a polynomial is calculated it is archived on disc in a location which the convenience method checks first. This ensures a polynomial is only calculated once. In calculation we primarily use recursion over x, the postion, rather than n, the order, as discussed in Section 4.6. We also take avantage of the symmetry condition present in the polynomials:

$$\hat{t}_n(N-1-x) = (-1)^n \hat{t}_n(x) \tag{B.1}$$

for computation, but not for storage. This is because the polynomial data will be accessed in a tight loop, show in Listing B.5, therefore it is quicker to store the full polynomial rather than invoke an `if else` statement inside the loop.

Code B.4: Calculating the Orthonormal Tchebichef Polynomial

```
- (void)populate {
  NSInteger index, prevNindex, prev2Nindex, prevXindex, prev2Xindex;
  double *ptr = (double *)calloc(N*half, sizeof(double));
  double value, n, x, NSqR, nSq, alpha1, alpha2, alpha3, gamma1, gamma2;
  // n recurrence scheme for all n, x = 0
  NSqR = sqrt(N);
  for(n=0.0; n<N; n++) {
    nSq = pow(n, 2);
    for(x=0.0; x<1.0; x++) {
      index = (n*half) + x;
      if(n == 0.0) {
        value = 1.0 / NSqR;
      } else if(n == 1.0) {
        value = ((2*x) + 1 - N)*sqrt( (3.0 / (N*(NSq - 1))) );
      } else if(n > 1.0) {

        alpha1 = (2.0/n)       * sqrt( ((4*nSq) - 1.0) / (NSq-nSq) );
        alpha2 = ((1.0 - N)/n) * sqrt( ((4*nSq) - 1.0) / (NSq-nSq) );
        alpha3 = ((n-1.0)/n)   * sqrt( (((2*n) + 1.0) / ((2*n) - 3)) ) * sqrt( ((NSq - pow(n-1,2)) /
              (NSq-nSq)) );

        prevNindex  = ((n-1)*half) + x;
        prev2Nindex = ((n-2)*half) + x;
        value = alpha1*x*ptr[prevNindex] + alpha2*ptr[prevNindex] + alpha3*ptr[prev2Nindex];

      }
      ptr[index] = value;
    }
  }

  // x recurrence scheme for n = 1..N-1, x = 0..half
  for(n=1.0; n<N; n++) {
    for(x=0.0; x<half; x++) {
      index = (n*half) + x;
      if(x == 0.0) {
        value = -sqrt( (N-n)/(N+n) ) * sqrt( ((2*n)+1)/((2*n)-1) ) * ptr[(NSInteger)((n-1)*half)];
      } else if(x == 1.0) {
        value = (1+((n*(1+n))/(1-N)))*ptr[(NSInteger)(n*half)];
      } else if (x > 1.0) {
        gamma1 = ((-n*(n+1)) - (((2*x) - 1)*(x-N-1)) - x) / (x*(N-x));
        gamma2 = ((x-1)*(x-N-1)) / (x*(N-x));;
        prevXindex = index - 1;
        prev2Xindex = index - 2;
        value = gamma1*ptr[prevXindex] + gamma2*ptr[prev2Xindex];
      }
      // Write the actual value
      ptr[index] = value;
    }
  }

  NSInteger ni, xi, i;
  for(xi=0; xi<half; xi++) {
    ptr[xi] = 1.0 / NSqR;
  }

  self.storage = [NSData dataWithBytes:ptr length:(N*half*sizeof(double))];

  // Double up the bytes
  double *large = (double *)calloc(NSq, sizeof(double));
  for(ni=0; ni<N; ni++) {
    i = ni*N;
    for(xi=0; xi<N; xi++) {
      large[i+xi] = [self valueForOrder:ni atPosition:xi];
    }
  }

  self.storage = [NSData dataWithBytes:large length:(NSq*sizeof(double))];

  free(ptr);
  free(large);
}
```

B.3.2 Computing Tchebichef Moments

Computing the actual moments is effectively four nested loops, as each point in the moment domain (iterations over p and q) requires an iteration over the image domain (iterations over x and y). The implementation in this framework introduces a third domain, called the `target rect` which is a spatial domain no larger than the image domain, and is used in its place. This allows use to compute moments over a sub rectangle of the image, but still registered in the same Tchebichef coordinate system.

In the listing below, the dictionary of properties passed as an argument to the compute function contains structures defining the image rectangle (which places the image in the N^2 Tchebichef domain) and the target rectangle. These are accessed in the prepareInputImageForComputation: method. The input image, and `N` have been previously set during the construction of the class.

Code B.5: Computing the Orthonormal Tchebichef Moments

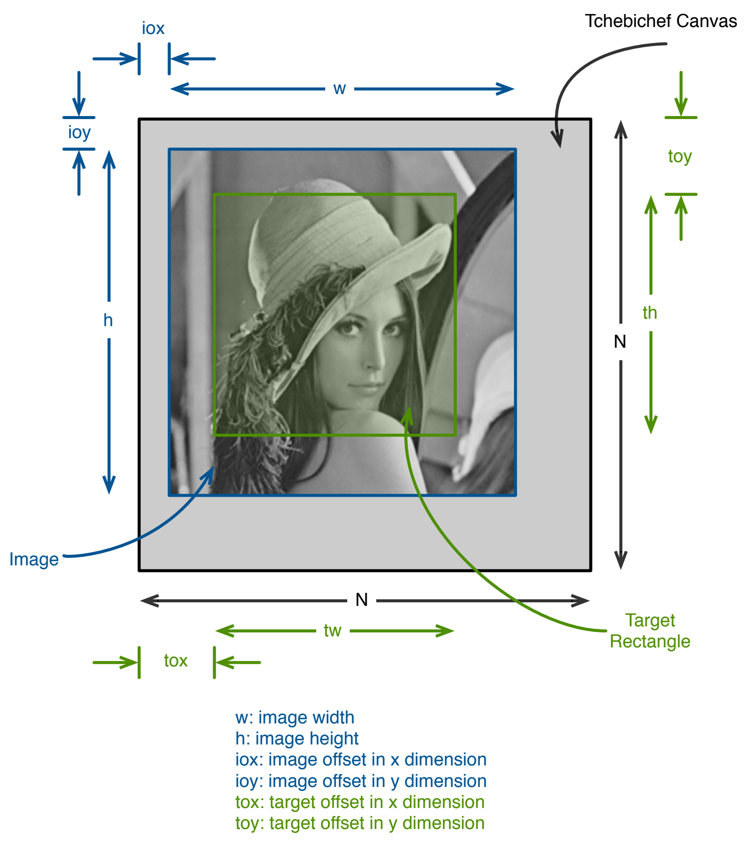

iox
w
Tchebichef Canvas
ioy
toy
h
th
N
Image
N
tw
Target Rectangle
tox
w: image width
h: image height
iox: image offset in x dimension
ioy: image offset in y dimension
tox: target offset in x dimension
toy: target offset in y dimension

Figure B.3: Rectangles used to determine iteration variables when computing Tchebichef moments.

```
- (void)computeWithoutSIMD:(NSDictionary *)props {

  NSNotificationCenter *nc = [NSNotificationCenter defaultCenter];
  [nc postNotificationName:kDTTchecbichefMomentsIsStartedComputation object:self];

  BOOL debug = [[props objectForKey:kDebug] boolValue];
  if(debug)
    [self debugWithString:@"Computing without SIMD"];

  // Prepare for computation
  if(![self prepareInputImageForComputation:props]) {
    // We've not a zero sized non-zero rect, therefore data is 0
    if(debug)
      [self debugWithString:@"Finished computing without SIMD"];
    [nc postNotificationName:kDTTchecbichefMomentsIsFinishedComputation object:self userInfo:nil];
    return nil;
  }

  NSUInteger targetWidth, targetHeight;
  targetWidth = (NSUInteger)self.targetRect.size.width;
  targetHeight = (NSUInteger)self.targetRect.size.height;

  NSUInteger targetColOffset, targetRowOffset;
  targetColOffset = (NSUInteger)targetRect.origin.x;
  targetRowOffset = (NSUInteger)targetRect.origin.y;
```

```
NSUInteger imageColOffset, imageRowOffset;
imageColOffset = targetColOffset - (NSUInteger)imageOffset.x;
imageRowOffset = targetRowOffset - (NSUInteger)imageOffset.y;

// Pointers to result memory and image memory
double *resultPtr = (double *)[self.momentData mutableBytes];
const double *polyPtr = (const double *)[self.poly.storage bytes];

const double *imgPtr = (const double *)[self.inputData bytes];
const CGFloat *coveragePtr = (const CGFloat *)[self.inputData coverageBytes];
BOOL hasCoverage = [self.inputData hasCoverage];

dispatch_apply(N, dispatch_get_global_queue(DISPATCH_QUEUE_PRIORITY_DEFAULT, 0), ^(size_t p) {

  // Row index for Tchebichef result buffer
  NSUInteger tRowIndex = p * N;

  dispatch_apply(N, dispatch_get_global_queue(DISPATCH_QUEUE_PRIORITY_DEFAULT, 0), ^(size_t q) {

    // local block variables
    NSUInteger row, col, tAdjRow, tAdjCol, iAdjRow, iAdjCol;
    double imgVal;
    double tp, tq, tmp, theMoment = 0.0; // Reset the moment variable

    // iterate over the shape domain
    for(row=0; row<targetHeight; row++) {

      // Adjusted Row
      tAdjRow = (row + targetRowOffset);
      iAdjRow = (row + imageRowOffset);

      // Reset the moment variable
      tmp = 0.0;

      // Row index for image
      NSUInteger iRowIndex = iAdjRow  * self.imgWidth;

      for(col=0; col<targetWidth; col++) {

        // Adjusted Col
        tAdjCol = (col + targetColOffset);
        iAdjCol = (col + imageColOffset);

        // If there is coverage data, check it
        if( !hasCoverage || (hasCoverage && (coveragePtr[iRowIndex+iAdjCol] == 1)) ) {
          // Calculate the inner loop
          tp = polyPtr[(p*N)+tAdjCol];
          imgVal = imgPtr[iRowIndex+iAdjCol];

          if(isnan(imgVal) || isinf(imgVal))
            imgVal = 0;

          tmp += (tp * imgVal);
        }
      } // End of column iteration

      // Calculate the outer loop
      tq = polyPtr[(q*N)+tAdjRow];
      theMoment += (tq * tmp);
      } // End of row iteration

      // Save the moment variable
      resultPtr[tRowIndex+q] = theMoment;

    }); // End of q iteration
  }); // End of p iteration

  self.isMomentsComputed = YES;

  if(debug) [self debugWithString:@"Finished computing without SIMD"];

  [nc postNotificationName:kDTTchecbichefMomentsIsFinishedComputation object:self userInfo:nil];

  return nil;
}
```

B.4 Covariate Interpolation

To interpolate point data as a smooth surface registered over the same spatial domain as the Sentiweb provided Influenza images, two main stages are performed: the estimation of a semivariogram for the given covariate type and calendar month, followed by Kriging interpolation at position in spatial domain, in this case France.

B.4.1 Estimating a Semivariogram

Semivariograms are stored as `SemivariogramModel` objects in the application framework, which is a subclass of `CustomObject` as shown in Figure A.1. This design allows numerous parameters to be defined dynamically, not only for the semivariogram properties sill, range and nugget, but also the cutoff value, and trend parameters.

1. Retrieve geo-referenced data.
2. Remove any trends. This is achieved by fitting a two-dimensional low-order polynomial to the data. Then, from each original value the trend estimate at the same position is subtracted.
3. Calculate the semivariogram cloud. This is the semivariance between each datum and every other datum, plotted against the Euclidean distance between the two points. In

this implementation this stored as two double precision vectors: `semivarianceBuff` and `lagBuff`

4. Fit the required model parameters sill, range and nugget to the semivariogram cloud using least squares regression.

Code B.6: Calculating the Semivariogram Cloud

```
/* Here we need to consider every point, and for each one, consider every other point.
 Then for each pair, we need to calculate the euclidean distance between them, and
 the semivariance between them, and stick those into an array. */

double cutoff = self.cutoffValue; // Cutoff in kilometers
numberOfSamples = [data count]; k = 0;
NSUInteger dataSizeLength = ceil(numberOfSamples*numberOfSamples*0.5);

// Get pointers to some storage space
double *semivarianceBuff = (double *)calloc(dataSizeLength, sizeof(double));
double *lagBuff = (double *)calloc(dataSizeLength, sizeof(double));
double i_easting, j_easting, i_northing, j_northing, i_value, j_value, eastingDiff, nothingDiff,
      lagValue, semivarianceValue;

// Loop over the points
for(i=0; i<numberOfSamples; i++) {
  // Loop over all the other points, and don't repeat
  for(j=i; j<numberOfSamples; j++) {
    if(i != j) {
      i_easting = [[[data objectAtIndex:i] objectForKey:@"easting"] doubleValue];
      j_easting = [[[data objectAtIndex:j] objectForKey:@"easting"] doubleValue];
      i_northing = [[[data objectAtIndex:i] objectForKey:@"northing"] doubleValue];
      j_northing = [[[data objectAtIndex:j] objectForKey:@"northing"] doubleValue];
      i_value =   [[[data objectAtIndex:i] objectForKey:@"residual"] doubleValue];
      j_value =   [[[data objectAtIndex:j] objectForKey:@"residual"] doubleValue];

      eastingDiff = i_easting - j_easting;
      nothingDiff = i_northing - j_northing;

      lagValue = sqrt( pow( eastingDiff, 2) + pow( nothingDiff, 2)); // Euclidean distance
          between the points

      if( (lagValue > 0) && (lagValue < cutoff) ) {
        semivarianceValue = 0.5 * pow(i_value - j_value, 2);
        semivarianceBuff[k] = semivarianceValue;
        lagBuff[k] = lagValue;
        k += 1;
      } // inside cutoff

    } // i != j
  }
} // End of loop over the points
```

Code B.7: Fitting the semivariogram model

```
// Range, Sill, Nugget
double *params = (double *)calloc(3, sizeof(double));
```

```
  params[0] = self.rangeValue;
  params[1] = self.sillValue;
  params[2] = self.nuggetValue;

  // Aux settings
  lm_control_type controlType;
  lm_data_type dataType;
  lm_initialize_control(&controlType);
  dataType.user_func = fitFunctionForExponential;
  dataType.user_t = (double *)[self.lag bytes];
  dataType.user_y = (double *)[self.semivariance bytes];
  NSInteger numberOfPoints = [self.lag length]/sizeof(double);
  self.numberOfDataPointsValue = numberOfPoints;

  // perform the fit
  lm_minimize(numberOfPoints, numberOfParams, params, lm_evaluate_default, lm_print_default, &
      dataType, &controlType);
```

Code B.7 utilizes a public implementation [70] of the Levenberg Marquardt [36] least squares regression algorithm.

B.4.2 Performing Kriging

There are two stages to performing Kriging, firstly the Kriging System must be computed, subsequently it must be solved for each interpolation operation, i.e. at every pixel position. As a result of this, the calculation of these two stages is separated, so that when new covariate data is retrieved from the database, the Kriging System matrix is created and saved. Later, if interpolation is required, the matrix is retrieved and solved.

Code B.8: Creating the Kriging Matrix

```
#pragma mark Kriging, pre-processing

// A function that ensures initialization tasks have
// been performed required for Kriging methods.
- (NSInteger)ableToProceedWithKriging {

  // Check we have a semivariogram model computed for this observation
  if(!self.semivariogramModel || (self.semivariogramModel && !self.semivariogramModel.isFittedValue)
      ) {
    BSLog(@"semivariogram model: %@", self.semivariogramModel);
    return -1;
  }

  // Get the pointer to array of NSPoints for the samples
  if(pointsAsNSPoints == NULL)
    if(![self makePointsAsNSPointsPointer])
      return -3;

  // We need the Kriging Matrix of these observation points
  if(!self.krigingMatrix)
    if(![self computeTheKrigingMatrix])
      return -2;

  return 0;
}

/* Compute the Kriging Matrix, required to perform Kriging interpolation.
 The results of this are stored with the object in Core Data,
 so this should only have to be done once. */
- (BOOL)computeTheKrigingMatrix {

  BSLog(@"Computing the Kriging matrix for %@", self.displayName);

  __CLPK_integer numOfRows, numOfColumns, length, rowIndex;
  __CLPK_integer i, j, N;
  N = [self.points count];
  numOfColumns = numOfRows = 1 + N;

  length = sizeof(float) * numOfColumns * numOfRows;

  // Create some space
  float *matrix = (float *)calloc(numOfColumns * numOfRows, sizeof(float));

  // Loop over the samples
  for(j=0; j<numOfRows-1; j++) { // loop down the rows
    rowIndex = j*numOfColumns;
    for(i=0; i<numOfColumns-1; i++) { // loop across the columns
      // Store in row order
      matrix[rowIndex+i] = [self.semivariogramModel valueBetweenThisPoint:pointsAsNSPoints[j]
           andThatPoint:pointsAsNSPoints[i]];
    }
    // At the end of each row, we want to add a 1
    matrix[rowIndex+i] = 1.0f;
  }

  // Then we add a final row of 1 except for last column is a 0
  rowIndex = N*numOfColumns;
  for(i=0; i<numOfColumns-1; i++) {
    matrix[rowIndex+i] = 1.0f;
  }
  matrix[rowIndex+i] = 0.0f;

  // We need to invert the matrix, we can do this with two LAPACK calls
  // First compute the LU factorization
  __CLPK_integer *pivotIndicies = (__CLPK_integer *)malloc(MIN(numOfRows,numOfColumns)*sizeof(
       __CLPK_integer));
  __CLPK_integer info;
  sgetrf_(&numOfRows, &numOfColumns, matrix, &numOfRows, pivotIndicies, &info);
  if(info != 0) {
    BSLog(@"We have a problem performing LU factorization.");
    return NO;
  }
  float *workspace = (float *)malloc(numOfColumns*sizeof(float));
  sgetri_(&numOfColumns, matrix, &numOfColumns, pivotIndicies, workspace, &numOfColumns, &info);
  if(info != 0) {
    BSLog(@"We have a problem inverting the matrix");
    return NO;
  }
  // Store this matrix
  self.krigingMatrix = [[NSData alloc] initWithBytes:matrix length:length];
  free(matrix);
  free(pivotIndicies);
  free(workspace);

  return YES;
}
```

The first function listed above simply checks that any processing that must be performed prior to Kriging has been done, and as a result will call the second function, if for some reason the Kriging Matrix hasn't already been created. To create the Kriging Matrix, a memory buffer is created and then filled by calling the Semivariogram Model methods that return the semivariance between any two points. The second half of the function simply inverts the matrix using LAPACK library calls.

Code B.9: Performing Kriging

```
// Performs Kriging to interpolate the value at a given point
- (float)interpolateValueAtPoint:(NSPoint)p {
  NSInteger err = 1;
  if( (err = [self ableToProceedWithKriging]) < 0) {
    BSLog(@"Unable to perform initialization for Kriging, error: %d", err);
    return 0.f;
  }

  // This is the Kriging Matrix
  float *matrix = (float *)[self.krigingMatrix bytes];

  // Sizes of the vectors
  __CLPK_integer N, numOfRows, numOfColumns;
  N = [self.points count];
  numOfColumns = numOfRows = 1 + N;

  // Buffers used in the interpolation
  xVector = (float *)calloc(numOfRows, sizeof(float));
  weights = (float *)calloc(numOfRows, sizeof(float));

  [self.semivariogramModel valueBetweenXs:xPointPositions andYs:yPointPositions andThisPoint:p
      inAnswer:xVector ofLength:N];

  xVector[N] = 1.0f; // For the Lagrange multiplier
  float ans, alpha, beta; alpha = beta = 1.0f;

  // kriging matrix multiplied by the vector of variogram values
  cblas_sgemv(CblasRowMajor, CblasNoTrans, numOfRows, numOfColumns, alpha, matrix, numOfColumns,
      xVector, 1, beta, weights, 1 );

  // Now we just need compute the inner product of the sample values and the weights
  vDSP_dotpr(weights, (vDSP_Stride)1, pointValues, (vDSP_Stride)1, &ans, (vDSP_Length)N);

  free(xVector);
  free(weights);
  return ans;
}

// Interpolate over the given rectangle, and return an NSImage
// containing an NSBitmapImageRep
- (NSDictionary *)interpolateOverRect:(NSRect)rect {

  // Get a pointer to the mask data
  if(!self.imageMask) {
//    self.imageMask = [[[NSImage imageNamed:@"FranceMask.png"] representations] objectAtIndex:0];
    self.imageMask = [[self.typeOfCovariate.typeOfPhenomenon.imageMask representations]
         objectAtIndex:0];
    self.imageMaskWidth = [imageMask pixelsWide];
    self.imageMaskHeight = [imageMask pixelsHigh];
  }

  unsigned char *maskData = (unsigned char *)[imageMask bitmapData];

  // Calculate the northing and easting coordinates of the area to interpolate
  NSRect realWorldCoords = [self realWorldCoordinatesForRect:rect];

  // Calculate the interpolation step size of the area
  NSSize stepSize = [self realWorldStepSizeForRect:rect];

  // If the rect is zero, set it to be the whole image mask
  if( NSEqualRects(rect, NSZeroRect) ) {
    rect.origin = NSZeroPoint;
    rect.size.width = self.imageMaskWidth;
    rect.size.height = self.imageMaskHeight;
  }

  // Process data
  NSUInteger numberOfBytes = sizeof(double)*(NSInteger)rect.size.width*(NSInteger)rect.size.height;
  BSSquareData *processData = [[[BSSquareData alloc] initWithCapacity:numberOfBytes andCoverage:YES]
      autorelease];
  processData.width = (NSInteger)rect.size.width;
  processData.height = (NSInteger)rect.size.height;
  double *processDataPtr = (double *)[processData mutableBytes];
  double *processDataCoveragePtr = (double *)[processData mutableCoverageBytes];

  NSBitmapImageRep *displayImgRep = [[[NSBitmapImageRep alloc] initWithBitmapDataPlanes:NULL
     pixelsWide:(NSInteger)rect.size.width pixelsHigh:(NSInteger)rect.size.height bitsPerSample:8
     samplesPerPixel:2 hasAlpha:YES isPlanar:NO colorSpaceName:NSCalibratedWhiteColorSpace
     bitmapFormat:NSAlphaFirstBitmapFormat bytesPerRow:2*(NSInteger)rect.size.width bitsPerPixel
     :2*8] autorelease];

  unsigned char *displayImgData = (unsigned char *)[displayImgRep bitmapData];


  // Some iteration variables
  CGFloat northing, easting, northingEnd, eastingEnd;
  NSUInteger i, j, dI, maskI, maskJ, processRowIndex, displayRowIndex, maskRowIndex, valueOffset,
     maskBytesPerRow, displayImgBytesPerRow;

  // Set the northing and easting end
  northingEnd = realWorldCoords.origin.y - realWorldCoords.size.height; // Remember we are
     minimising the northing
  eastingEnd = realWorldCoords.origin.x + realWorldCoords.size.width;

  maskBytesPerRow = [imageMask bytesPerRow];
```

```
displayImgBytesPerRow = [displayImgRep bytesPerRow];

// Set some of the iteration variables
j = 0;
maskJ = (NSUInteger)rect.origin.y;

float pixel, normalized, covMin, covMax, multiplier;

covMin = self.typeOfCovariate.minimumValue;
covMax = self.typeOfCovariate.maximumValue;
multiplier = 255.0/(covMax - covMin);

// Iterate over the geographical region
for(northing=realWorldCoords.origin.y; northing>northingEnd; northing -= stepSize.height) {

  // Update the easting iteration variables
  i = dI = 0;
  maskI = (NSUInteger)rect.origin.x;
  maskRowIndex = maskJ*maskBytesPerRow;
  processRowIndex = j*processData.width;
  displayRowIndex = j*displayImgBytesPerRow;


  for(easting=realWorldCoords.origin.x; easting<eastingEnd; easting += stepSize.width) {
    // Check if this postion is in the mask
    if( maskData[maskRowIndex+maskI] > 0 ) {
      pixel = [self interpolateValueAtPoint:NSMakePoint(easting, northing)];
      // Set the pixel in the process data
      processDataPtr[processRowIndex+i] = (double)pixel;
      processDataCoveragePtr[processRowIndex+i] = (CGFloat)1.0f;
      // Display image
      normalized = (pixel - covMin) * multiplier;
      displayImgData[displayRowIndex+dI] = 255;
      displayImgData[displayRowIndex+dI+1] = (unsigned char)normalized;

    } else {
      // Set the pixel in the process data to zero
      processDataPtr[processRowIndex+i] = (double)0.0;
      processDataCoveragePtr[processRowIndex+i] = (CGFloat)0.0f;
      // Display image
      displayImgData[displayRowIndex+dI] = 0;
      displayImgData[displayRowIndex+dI+1] = 0;
    }

    // Update the image column index
    i += 1;
    dI += 2; // Display image has an alpha component
    maskI += 1;

    if(i == imageMaskWidth)
      break;

  } // End of column iteration

  // Update the image row index
  j += 1;
  maskJ += 1;

  if(j == imageMaskHeight)
    break;
  } // End of row iteration

  // Create NSImage instance for the display data
  NSImage *displayImg = [[[NSImage alloc] initWithSize:rect.size] autorelease];
  [displayImg addRepresentation:displayImgRep];

  return [NSDictionary dictionaryWithObjectsAndKeys:processData,
       ObservationInterpolationProcessDataResult, displayImg,
       ObservationInterpolationDisplayDataResult, nil];
}
```

The above code details the two principle functions of this implementation of Kriging. The first function solves the Kriging System (interpolates the value) at the given point. This is called for each pixel position in the target image from the second function. However, the Kriging matrix, and associated Semivariogram are all using real world co-ordinates, so the point provided to the first function is a northing and easting coordinate. This is simplified by iterating over the target rectangle in real world northing and easting, rather than pixel coordinates. Separate iteration variables `i` and `j` are maintained to coordinate the pixel values after the interpolation has been performed. This algorithm is complicated further by the use of an image mask. Because the spatial domain is France in this application, many pixels in the square image are not defined, as they are outside the border. Therefore, an image mask is used to check if the tested location is valid, in which case interpolation proceeds.