Statistical models for rainfall: characterisation and forecasting

EMMANUEL BATAIL

MSc by Research in Pattern Analysis and Neural Networks

ASTON UNIVERSITY

March 2002

ASTON UNIVERSITY

Statistical models for rainfall: characterisation and forecasting

EMMANUEL BATAIL

MSc by Research in Pattern Analysis and Neural Networks, 2002

Thesis Summary

Weather forecasting is a prediction of what the weather to come will be like. This can be done in a deterministic or probabilistic way. We are interested in nowcasting, which is the production of short-range (0 to 6 hours) forecasts. The goal of this thesis is to create and develop a statistical model for precipitation forecast with radar data only. This forecast will be probabilistic, which means that we will provide a forecast mean and covariance for the precipitation. This study starts with the research of a space reduction for the data, as time required to run models is crucial in weather forecasting. Then, we build in a Bayesian framework a general model, which is developed in the latter part of the thesis. We need dynamics to represent the evolution of the rainfall field. The dynamics of this model come from the advection equation, which links the rainfall field to the advection field. The model has been tested with simulated and real data.

Keywords: Advection, Nowcasting, QPF, Bayesian, Gaussian

Acknowledgements

I would like to thank my supervisors, Dan Cornford and Christopher James, for their help, their patience, their availability, and especially Dan Cornford for the large amount of help he gave me to complete this thesis.

I am grateful to all my lecturers D. Lowe, D. Saad, M. Opper, I. Nabney, I. Stainvas, and P. Tino for the quality of their lecture. Many thanks also to Miss Vicky Bond.

I would also like to thank in general all the people from the Neural Computing Research Group department for their constant support and friendly presence through all this year and especially Nikos Skantzos, Lehel Csato, Wei Lee and the Msc students in PANN, Boremi, Mani, Nicolas, Oliver and Remi as well as Stephane a PhD student.

I also want to thank my friends for their support, Alexandre, Amos, Beatrice, Clement, Eric, Jimmy, Karl, Liz, Paul, Sebastien and Sergei from Aston with whom I've spent such a wonderful year as well as Aurelie, Bruno, Gilles, Matthieu, Richard, Stephanie, Thomas L, Thomas G and Vitali for their visits.

Thanks also to the UK Met office for the data they supplied.

To finish with my acknowledgements, I would like to thank my parents for their love and their financial support to make it possible.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Weather forecasting

Weather forecasts provide information about the weather to come. Many different techniques are used in weather forecasting, from relatively simple observation of the sky to highly complex mathematical models run on computers. Weather prediction can be for the next hour, next day, or next few days. However, the forecast's accuracy decreases significantly beyond about 10 days. Weather forecasting remains a very complex task although enormous improvements in the accuracy of weather forecasting have been realized. The improvements are mainly stimulated by the advances of modern technology, especially computers and weather satellites, and the availability of data provided by coordinated meteorological observing networks.

The works of Lorentz led to the idea of chaos in physical systems. In the case of meteorology, this implies that small differences in the initial conditions of the atmosphere can lead to big differences later in the weather. Sometimes these initial differences are too small to be detected by observations. The weather can therefore be chaotic and unpredictable, which is why it is really hard to do weather forecasting. Weather forecasting is one of the hardest challenge not only because of this but also because it demands a large amount of computational power.

Weather forecasting is a large field and we restrict our attention to precipitation forecasting. Meteorologists call precipitation all the forms of water that fall from the air to the Earth's surface. Air contains water vapour from the evaporation of liquid water from the Earth's surfaces (oceans, soil, lakes, rivers, ...). When air moves because of differences of temperatures and pressures, it may cool and may release vapour as condensation in form of cloud or eventually rain (or another form of precipitation).

Precipitation seems to be the most obvious of the weather elements in its effects on normal life. Indeed, too much precipitation (flash floods, floods, heavy rain storms) or drought can be dangerous (agriculture, transport, navigation,etc), that is why there is a need to forecast precipitation. Forecasting precipitation is not simple, rainfall can be convective, or frontal and a number of different forecast methods are possible : nowcasting, which is the production of short-range (0 to 6 hours) forecasts, and numerical weather prediction forecasts over longer periods.

We are interested in nowcasting of all types of rainfall. Nowcasting began nearly 50 years ago with the work of Ligda [11]. He showed that forecasts could be made from the persistence and the movement of radar echoes.

The aim of this project is to define and to develop a full probabilistic quantitative precipitation forecast (QPF) model based on the advection equation (Chapter 3). This model is a statistical one which produces a probabilistic nowcast of precipitation using radar data only.

1.2 Different Approaches to Forecasting

A lot of research on this subject, precipitation forecasting, has already been done. Within meteorology, there are two general forecasting approaches : deterministic or probabilistic methods.

1.2.1 Deterministic

Numerical Weather Precipitation (NWP) is the basic framework of deterministic forecasting. These models contain full representations of the physical equations governing large scale tropospheric behaviour. Actually, NWP numerically integrates a set of partial differential equations, in state variables that describe the atmosphere (physically based partial differential equations). These equations describe the time evolution (and coupling) of the state variables.

NWP consists of solving nonlinear differential equations. This is not possible by precise analytical methods; it is done by numerical approximation. To realize this, a huge amount of computation is required. NWP models have always been limited by the rate of evolution of computer power. Besides, because of the non linear nature of equations, the initialisation is again very important (small differences in the inital conditions can imply big differences later on).

These models do not obtain accurate results for nowcasting. In very short range forecasting, the key problem is data assimilation, spin up and the time required to run models. The spin up (0-3 hours) is the time it takes the model to stabilise. It ensures internal consistency in the variables. The spin up can seriously affect the quality of data assimilation.

1.2.2 Model used

There are many different modelling approaches, the two most commonly used in the UK are :

NIMROD is an automated precipitation nowcasting system using rainfall data from the UK weather radar network, weather satellite observations and Meteorology office Mesoscale Model outputs as the basis for prediction.

GANDOLF [2], acronym for Generating Advanced Nowcasts for Deployment in Operational Land-based Flood forecasts, is an automated convective rainfall nowcasting and early warning system. This model uses the previous one to forecast if there is no airmass convection. If airmass convection occurs, it uses an object-oriented model, which simulates the life cycles of individual showers (rain) independently.

1.3 Thesis overview

This thesis first develops a very general QPF model, and then implements it.

Chapter 2 presents the radar dataset used for the forecasting. This data has been provided by the UK Met office. The way data is retrieved from the radar is briefly overviewed. Chapter 3 deals with the advection equation, and the QPF in a very general definition. It introduces as well the Bayesian approach and the framework in which the model is defined. In Chapter 4, the first attempt to create a forecast precipitation model is described. This chapter also deals with the way to solve the advection equation, and problem encountered with the finite difference method. Chapter 5 copes with the chosen model for the rainfall field. The results of the fitting of this model on real data are presented. Chapter 6 presents the final fully probabilistic model developed, as well as the results obtained on simulated and real data. The development of this model is the result of a working collaboration with Dan Cornford.

Chapter 2

The data

This chapter presents radar data. It also deals with space reduction of our datasets.

2.1 Radar data

Radar data is widely used in precipitation forecasting, because radar can be used to measure rainfall. In 1948, Marshall and Palmer demonstrated the existence of drop size distribution which is a function of the rain rate, and therefore led to a corresponding relation between radar reflectivity Z and rain rate R [3]. Further work has been undertaken and now rainfall measurements can be retrieved from radar data. However, radar rainfall measurements are not perfect and ground clutter or bright beam must be taken into account. The former is simple to understand and to detect. When a pulse of energy leaves a radar antenna, it gradually spreads in all directions. When this pulse strikes a cloud or rain drops, it reflects a pulse in all directions and some of this reflected pulse reaches the antenna. The problem is that buildings and hills also reflect this pulse and are often plotted on radar pictures as precipitation. You can spot ground clutter by viewing a loop of individual radar pictures and looking for stationary radar returns embedded with moving radar returns. These stationary returns are usually ground clutter. Ground clutter often can be filtered out of the radar image due to the development of stronger computers and Doppler radar [14]. A radar bright band is observed in layers where the water phase in the air changes from solid to liquid (the height at which snow begins to melt into rain). The enhanced reflectivity can be caused by aggregation of wet snow crystals near this layer, or by relatively large single ice crystals developing a wet surface during melting. In either case, the wet crystals/aggregates appear to have a dielectric constant more nearly characteristic of water, while briefly maintaining a size that is larger than an equivalent-mass water droplet. As radar data used the drop size of the rainfall to estimate the rainfall, these effects imply error in the estimate of the rainfall. On the radar image, it gives a bright band layer. This bright band implies a error in the estimate of the rainfall [1].

Radar data provides rainfall measurement. This data is often plotted on an image where the colour map represents the different values of the rainfall rate. To colour the image is the same thing as using a plot in 3D (see Fig 2.1).

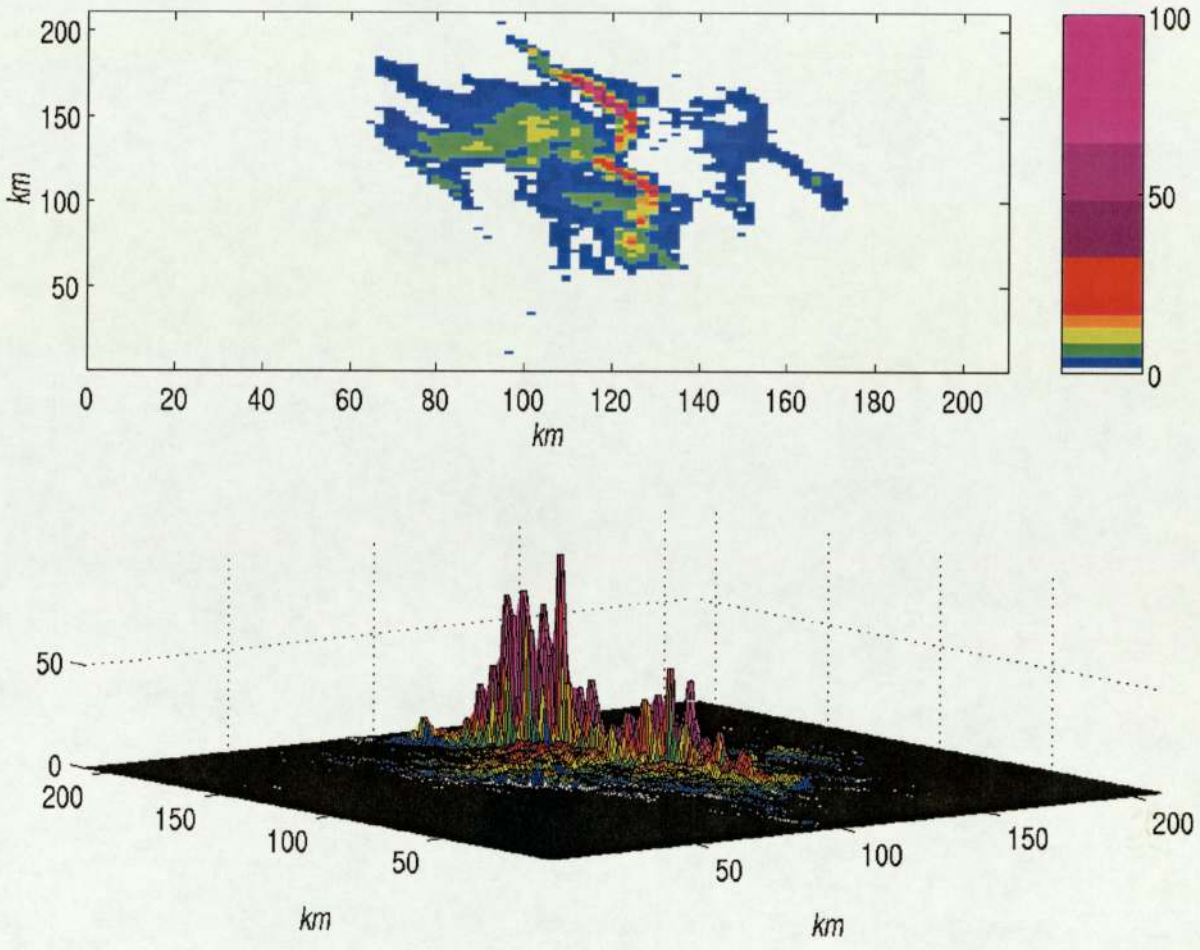


Figure 2.1: Image of the radar data in 2D and in 3D, 30^{th} October 2000 6:45 am. In 2D, the colormap determines the value of the rainfall at each point. The axis represents the size of the image in km.

2.2 Presentation of the data set

The data used in this project was supplied by the UK Met office. The data can be divided into two types : one covers the whole of the UK mapped to a 5-km grid, and the second covers part of England and Wales. We will concentrate on the data covering England and Wales (at 2km resolution). Each image represents rainfall intensity and is sampled at 5 minutes interval. Each set of data covers the same grid but over different dates and over different lengths of time. The data contains a matrix which gives the rainfall intensity.

Radar data is our primary and unique source of precipitation information. We obtained two differents types of data: one using only one radar, the other uses multiple radars to create a composite image, we have only used those with only one radar.

Actually radar echoes do not measure rain, but rather microwave radiation. The data we get is preprocessed to remove ground clutter and bright band. They are the best available estimates of spatially distributed precipitation intensity. But some errors still occurs.

2.3 Exploring the data

Weather forecasting is one of the most computationnally intensive activities, and today biggest computers are used to make weather predictions. The main problem is the high dimensionality of the data. In this chapter, each location will be treated as a separate dimension.

In this chapter, we study the possibility of reducing the dimensions of the provided data sets in order to gain time in data assimilation and in general to be as quick as possible. Being computionally efficient is especially important in nowcasting. The first part of the project was to deal with the data, try to understand its structure, and to reduce, if possible, the dimension of the state space in which to represent the rainfall field without compromising the forecast performance. A Principal Component Analysis (PCA) [6] was applied to the data in order to project it onto a subspace. PCA is a classical statistical method. PCA involves a mathematical procedure that transforms a number of (possibly) correlated variables into a smaller number of uncorrelated variables called principal components. The first principal component accounts for as much of the variability in the data as possible, and each succeeding component accounts for as much of the remaining variability as possible. Often, PCA is performed on the covariance matrix of the variables.

Let's define Σ_D the covariance matrix of the data matrix of dimension n. Σ_D is real symmetric positive definite matrix. So we can obtain an orthogonal basis by finding its eigenvalues and eigenvectors. We can rewrite the covariance matrix :

$$\Sigma_D = U\Lambda U' \tag{2.1}$$

where U is the basis change matrix from the old base to the new one, this matrix is orthonormal, Λ is a diagonal matrix with all the eigenvalues on the diagonal.

Let's note (λ_1,..,λ_n) the set of eigenvalues with $\lambda_n < .. < \lambda_1$ and (u_1,..,u_n) the corresponding eigenvectors.

$$Var_{exp}(k) = \frac{\sum_{j=1}^{k} \lambda j}{\sum_{j=1}^{n} \lambda j} * 100 \tag{2.2}$$

where $Var_{exp}(k)$ represents the variance explained by the k first eigenvalues, $k \leq n$.

The set of vectors $(u_1, .., u_k)$, $k \leq n$, explains Var_{exp} percent of the variance. The subspaces determined by the eigenvectors $(u_1, .., u_k)$ are all the projection space possible. A subspace then has to be chosen. Choosing the subspace is then equivalent to fixing k. The aim of PCA is to find k as small as possible, but large enough to explain the data. $Var_{exp}(k)$, which is a function of the number of components k, enables to know for each k how well this reduction fit the data. Therefore, the choice of k, which depends on how well we want to fit the data and how much we want to reduce space, can be made by comparing the reduction obtained and the variance explained.

In our case, the observation data is a matrix of 210 by 210 pixels. The data set used is from measurement made on the 3^{rd} April 2000. The observation starts at 9:15 am, finishes at 1:50 pm, and is sampled every 5 minutes. The number of observation, denoted T, is 34. In this chapter, we will denote $(D_t)_{1 \leq t \leq T}$ the observation matrix, where D_1 is the first observation at 9:15 am and D_T the last at 1:50 pm.

To perform PCA on the whole data matrix is impossible because it requires too much computational power. Therefore we divided the matrix into blocks and put each

block in a column vector. By convenience, these blocks are square. The size of the block is a parameter we have to choose. We have tried a lot of different size of block, which were computationally plausible. In the results presented below, the size of block is 20. We have choosen to show you the results with this particular choice of parameter because this choice gives interesting results.

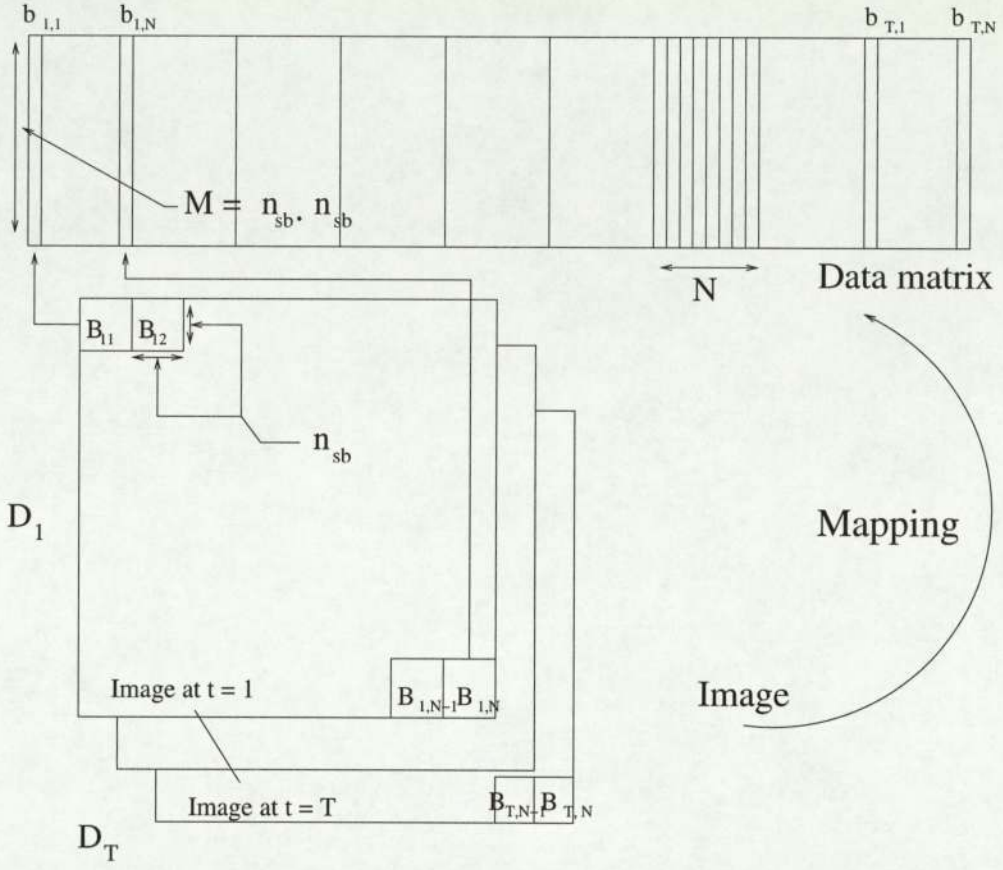


Figure 2.2: Division of the image into blocks. N is the number of block per image. $B_{t,n}$ represents the block number n at the time t. Each block $B_{t,n}$ composes a column of the data matrix. Image 1 is the radar data at time $t = 0$. T is the number of radar data we have in this dataset. Each time step between two observations is 5 minutes. The Image T represents the radar data at $t = T$. D_t is the radar data matrix at time t. $b_{1,1}$ is the first block of the first image ($t = 1$), $b_{1,N}$ is the last block of the first image, $b_{T,N}$ is the last block of the last image. The size of the mapped matrix is $M = n_{sb}^2$ lines, and $N * T$ rows.

PCA is applied to the transpose of the data matrix (shown in Figure 2.2).

This decomposition leads to a global basis for each block. We did this decomposition because we thought that all blocks could have some common features.

We have $N * T$ blocks of size $M * 1$. Then, we find a space basis such that :

$$\Sigma_b = U \Lambda U' \tag{2.3}$$

where Σ_b is the covariance matrix of the block, $U = [u_1, .., u_M]$ is the basis change matrix, Λ is a diagonal matrix with all the eigenvalues $(\lambda_i)_{1 \leq i \leq M}$ in the diagonal. In

PCA, we have to choose the number of eigenvectors we want to keep but we want to have $k << M$. This number depends on the Var_{exp} we want to have. Let us note k the size of the subspace chosen, reducing the dimension from M to k.

Let us define $A = [u_1, .., u_k]$. To each $b_{t,n}$ corresponds a unique $s_{t,n}$ such as :

$$b_{t,n} \approx A.s_{t,n} \tag{2.4}$$

where $s_{t,n}$ is a real row vector of dimension k.

The equation 2.4 is only an approximation because we have trimmed the matrix to the k first components.

Globally, D_t is represented by $s_{t,1}, ..., s_{t,N}$. Thus, for each observation data matrix we have a reduced size of $k * N$.

PCA enables us to change our basis to represent data. It computes all the eigenvalues as well as the eigenvectors. From the eigenvalues and eigenvectors, we have to decide if a reduction using PCA is worth while.

2.4 Results

We have explored the evolution of the eigenspectrum (sorted by decreasing value) and the eigenvectors. The ideal would be to find a big difference between the n first eigenvalues and the others with n as small as possible. Then, these first n eigenvalues would account for most of the variance and we could do a projection over these eigenvalues and neglect the others. The purpose is to find n really small in order to have a useful projection. In the test presented, we choose 20 as the size of each block. The Figure 2.3 shows the eigenvalues on a linear and a logarithmic scale. The logarithmic scaled graph is interesting because it shows some step between the eigenvalues.

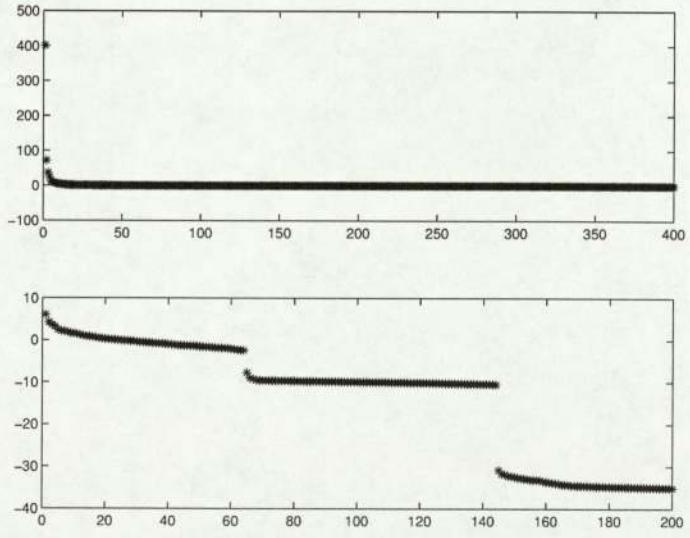


Figure 2.3: Plot of the eigenvalues on a linear scale (top) and on logarithmic scale (botttom). The size of each image block is 20. Data measurements from the 3^{rd} of April 2000. On the log-plot, we have only plotted the first 200 eigenvalues, because of computational issues.

Number of principal components	Variance explained
10	93.14
20	96.93
30	98.43
40	99.23
50	99.67
60	99.94

Table 2.1: Variance explained in accordance with the number of components

In this test, the size of each block $b_{t,n}$ is 20, so the number of blocks per observation data matrix is 100. The dimension of the eigenspectrum is in this case 400 (20^2).

So it seems possible to remove some components. More than 99,9% of the variance is explained with the first 60 eigenvalues (see Table 2.1). We can project the data on a subspace of dimension 60. The 60 eigenvectors corresponding to these eigenvalues will be a basis of this subspace. We have to keep so many basis because we want to fit the data the best we can. The size of the space when the size of the block is 20 is 400. The reduced dimension is still not small enough.

We want to plot the eigenvectors corresponding to these eigenvalues (Figure 2.3), and to look at these basis vectors. We have to invert the mapping to plot them. Eigenvectors are column vectors of size M, where M is the square of the size of each block. The size of each block has been chosen before mapping. We are reshaping the eigenvectors into a square matrix whose dimension is the size of the block. Now, we are able to plot them.

The 16 eigenvectors corresponding to the 16 biggest eigenvalues are plotted as well in order to visualise the resulting projections. Those eigenvectors are the most interesting one because they are explaining most of the variance. Plotting the corresponding eigenvectors seems to encourage us to believe suitable basis functions could be found (Figure 2.4). Indeed, the global shape of the eigenvector is quite similar to radar data. Some local structures can be seen on the image of each eigenvector.

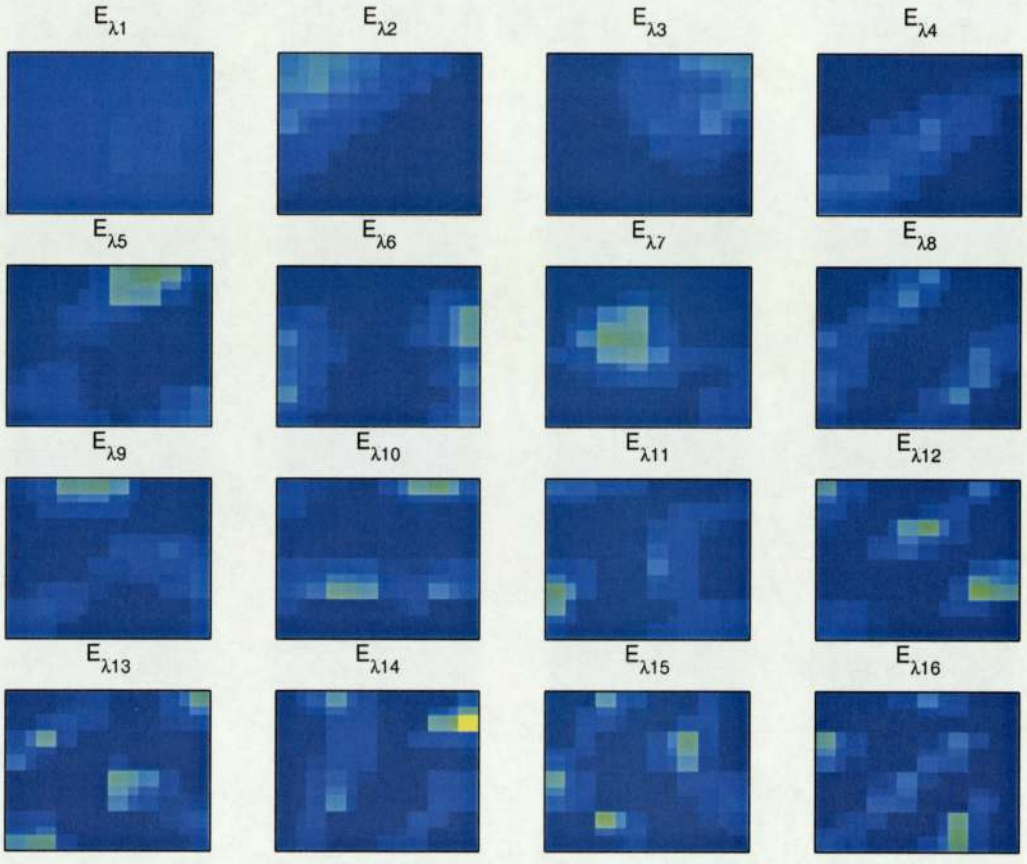


Figure 2.4: The eigenvector images E_λ corresponding to the specific eigenvalues λ in decreasing order. Localised structures appear on the images. Data from the 3^{rd} of April 2000.

We then increase the size of each block, but now blocks can overlap each other, as shown in Figure 2.5.

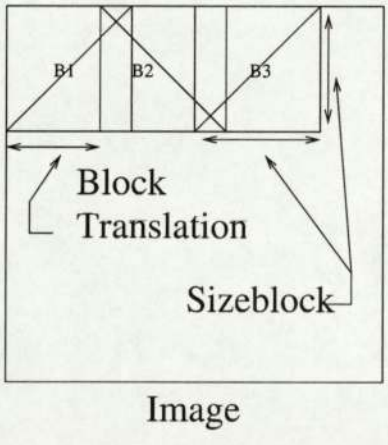
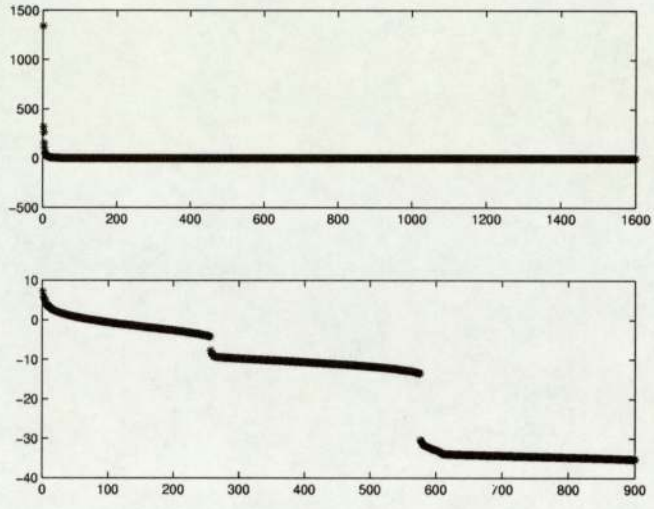


Figure 2.5: Division of the data image with overlapping block. The division in block is different than in Figure 2.2 but afterwards the process done on each block is the same. Surrounding blocks have now a common place.

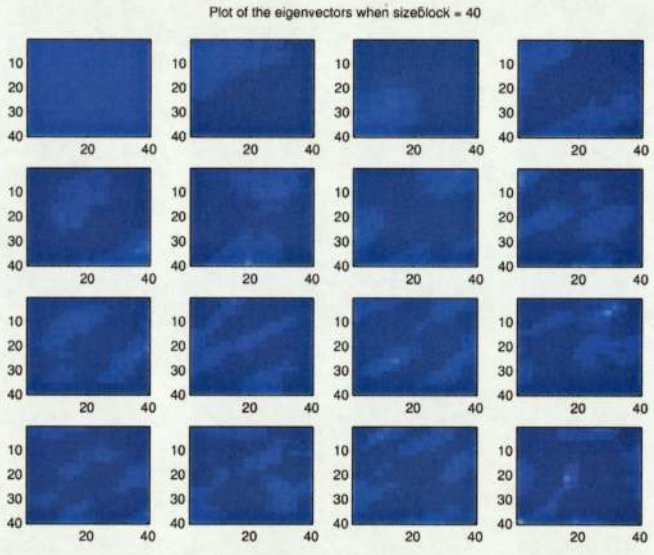
This overlapping block decomposition has been done, because we thought that it could give more general basis. Since, two successive blocks have a common part. This methods increases the size of the data, but we hope to catch more characteristics of the radar images. Moreover, the method without overlapping was presenting a drawback: if a precipitation cell is divided between blocks, we are not able to track all the characteristics of this cell. Now with this method this problem is solved.

The size of the block is again a parameter we have to choose as well as the size of the overlapping. We have tried a lot of different size of block and overlapping, which were computationally plausible. In the results presented below, the size of block is 40 and we translate each block by 25 (translation along the rows). We have choosen to show these results because this choice gives interesting results. More general eigenvectors which now require approximately 150 eigenvalues to explain more than 99.9% of the variance (Figure 2.6(a) and 2.6(b)) are now obtained.

These eigenvectors which represent the basis are more general because a part of the data appears two times as there is an overlapping block decomposition. Moreover, it is clear that to account for the same amount of variance than in the previous experimence (without any overlapping) we need to keep more basis function as surrounding blocks have a common part. The graph of the eigenvalues is similar to the one obtained without this overlapping method and presents the same kind of discontinuity.



(a) Eigenvalues



(b) Eigenvectors

Figure 2.6: Plot of eigenvalues on the top and eigenvectors on the bottom of the PCA. The image block $b_{t,n}$, t fixed, are overlapping. On the bottom plot, the eigenvector images corresponding to the specific eigenvalues in decreasing order (must be read from the top left to the right as in Figure 2.4). Data measurements made from the 3^{rd} of April 2000

2.5 Summary

To conclude this chapter, it can be seen that it is possible to reduce the dimension of the state space, and here we have quite promising results. But the reduction we find is not worth while. In this project, we need to put dynamics in the model. With this projection, to cope with prediction is a hard task.

To deal with dynamics, we would have to put dynamics in $s_{t,1}, .., s_{t,N}$ as $\bar{A}$ is fixed (see equation 2.4).

Let us define $S_t = (s_{t,1}, .., s_{t,N})'$ as a real row matrix whose size is $k * N, 1$, k is the number of components chosen. Having dynamics is to be able to infer S_{t+1} from S_t.

A typical linear dynamical system can be defined by :

$$S_{t+1} = B.S_t + \eta \tag{2.5}$$

where B is a real matrix of size $(k * N, k * N)$ and η is observational noise. We would have to learn B for the observation data sequence. In practice, this would not be computationally feasible because of the size of B. For instance, if the size of each block $b_{t,n}$ is 20, then N is 100 and we choose 50 components, so B is a matrix whose size is $5000 * 5000$.

The learning of B will be too long. In this project, we need to put dynamics in the model. If we kept this space reduction, we would have computation problem for the dynamics. To have a matrix B whose dimension is not too big, we would need to keep only 1 or 2 components (then the size of B would be $100 * 100$ or $200 * 200$). But we would not have enough information if we kept only 1 or 2 components.

The idea of using PCA to reduce the size of the data is not incorporated into our analysis. This reduction will be done differently. Although this space reduction method was not chosen, this study gave us some knowledge about the rainfall data structure.

Chapter 3

Quantitative Precipitation Forecast Model

Precipitation forecasting can be made over different time lengths. For this work, we are interested only in short range prediction which are usually up to 24 hours forecast. Nowcasting is a production of short range prediction where time lengths are limited to 6 hours.

The basic difference from the previous chapter, where the model for the rainfall was a discrete pixelised model with uncorrelated error, is that in this chapter the rainfall field model is a continuous process. The method on which the model is based is the advection and the growth/decay of the current precipitation field. The advection is supposed to be a continuous process in space and time. We will assume only the advection term is responsible for the dynamics.

There are several pieces of knowledge we would like to incorporate concerning precipitation and its short term evolution. The rainfall field is supposed to be positive all the time. The rainfall field and the advection are continous processes in space and time. We will also try to incorporate that the space and time rate of change of the advection is much slower than that of the precipitation field.

This chapter deals first with advection, then, using the Bayesian framework, the model is developed. In the final section, we give the description of a general model.

3.1 Advection

3.1.1 Definition of advection

Air movements are sparked by differences of temperature or pressure, and are felt as the wind. In the atmosphere convection and advection transfer heat energy from the hottest regions to the coldest ones. We can observe air movements of this type during the formation of the sea or coastal breeze, as consequences of the difference of temperatures between the sea and the soil. On a larger scale, differences of temperature are at the origin of the principal wind zone through the earth [10].

One way that heat is transferred through air is by convection. In the atmosphere, convection occurs when a layer of air in contact with a hot surface warms by conduction, acquires buoyancy because warmer air is less dense than colder air, and rises, taking with it the energy that it stores. As the Earth is heated by the Sun, bubbles of hot air

called thermals rise upward from the warm surface. In meteorology, convection refers primarily to atmospheric motions in the vertical direction.

Advection is defined as the process of transport of an atmospheric element just from the mass motion of the atmosphere. As a consequence the properties of the atmosphere element are also transferred. Here advection refers to the horizontal transport of the variable [5]. So the advection can be represented by a vector of dimension 2. Let us note it (u,v). Two features of information about advection are especially important in meteorology : the vorticity and the divergence. The vorticity is defined by $(\frac{\partial v}{\partial x} - \frac{\partial u}{\partial y})$ with x,y planar coordinates and the divergence is defined by $(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y})$. Vorticity is defined as twice the angular speed of rotation, whereas divergence as the relative rate of increase of the area enclosed by a material surface [7]. We want to set these features of the advection and try to stick to the reality as far as possible. In practice, the divergence is very small.

In this thesis, we are only assuming advection, because our dynamics will be only in 2 dimensions.

3.1.2 Advection equation

The dynamics in the rainfall field we will use are actually from the optic flow theory. The optic flow theory is used for tracking objects in an image. Finding optic flow is finding image displacements [12]. To find optic flow, different methods have been developed. The gradient based approach is one of them. Gradient-based methods use spatial and temporal partial derivatives to estimate image flow at every position in the image.

Horn and Schunk use space-time derivatives of the evolving image brightness function to give a single equation which partially determines the optic flow. The assumption is made that the brightness of any part of the imaged world changes very slowly, so that the total derivative of the brightness is zero. When differentiated using the chain rule, this gives the equation 3.1 :

$$\frac{\partial I}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial I}{\partial y}\frac{\partial y}{\partial t} + \frac{\partial I}{\partial t} = 0 \tag{3.1}$$

where $I(x, y, t)$ is the image brightness function.

In radar data, we can use as well as in optic flow the equation 3.1 to track movements of cells of rainfall. Our starting point is the work from the Met office based on this equation [9]. Our goal is to introduce probabilistic methods to their current advection based approaches. We will use optic flow methods, and these will be used in order to compute the advection field.

Nevertheless, the equation 3.1 makes the assumption that features within an image sequence only change shape, and do not change in size or intensity. This is not the case with rainfall, that is why we add a growth/decay term G in this equation to take into account the change in our image sequence. The advection equation is then defined by equation (3.2).

The advection equation is then :

$$\frac{\partial R}{\partial t} = -u\frac{\partial R}{\partial x} - v\frac{\partial R}{\partial y} + G \tag{3.2}$$

where (u, v) is the velocity at each point (x, y), in the images and $R(x, y)$ is the rainfall intensity at a given time.

The rainfall and the advection field are continuous processes in time and in space. G is, in principle, a function of space and time but due to its complexity we will proceed with certain approximations. Namely it will encompass the model error and will not be a function of space. Judging from the model's output and by comparison with the data the idea is to adjust G by trial and error.

In comparison with the model used in Chapter 2, the main difference is that R is now a continuous process. In Chapter 2, R_t was represented by D_t, which is a simple pixel model.

The main advantage of this continuity in the rainfall model is that this is more realistic. Indeed, the rainfall process is a continuous one. Since the equation 3.2 describes a differential equation our objective will be to construct a differentiable model to enable partial derivatives with respect to space and time to be computed. The model will be necessarily continuous, as differentiability implies continuity. With the advection equation, we decompose R using the advection field and an initial value of R. This point is essential because it is our dynamics. It enables us to do our update along the time axis. We assume only advection which means that advection is the only dynamics we will take into acount.

3.2 The Bayesian approach

3.2.1 Bayesian point of view

Bayesian inference is another way of using probability. In a frequentist paradigm, the probability of an event is considered as the fraction of times that the event occurs in the limit of an infinite number of trials. But in a Bayesian framework, a probability can be interpreted as a subjective degree of belief about the result of the match. It is an expression of belief rather than an expectation of a number of trials.

Cox in 1946 showed that the Bayesian interpretation of probability leads to a consistent Bayesian formalism where degrees of belief can be updated in the light of new information, or data, using Bayes' rule.

3.2.2 The principle

The purpose of a Bayesian framework [8] is to use the Bayes' rule to compute the probability of parameters given the observations (data). For instance, the probability of the parameter w given the observations D :

$$p(w|D) = \frac{p(D|w)}{p(D)}p(w) \tag{3.3}$$

The distribution $p(w)$ is called the prior distribution and quantifies the knowledge available about the parameter values before the data is observed. The prior reflects

our initial belief in the range of values that w takes. In the absence of any serious beliefs, the prior distribution will normally be rather flat or uninformative. The prior distribution is very broad to reflect the fact that we have little idea of what values the paramters should take. It is better to give little information on the prior than to give wrong one. The distribution $p(D|w)$ is the dataset likelihood.

The distribution $p(w|D)$ is called the posterior distribution and is determined by using Bayes' rule (Equation 3.3) (see Figure 3.1).

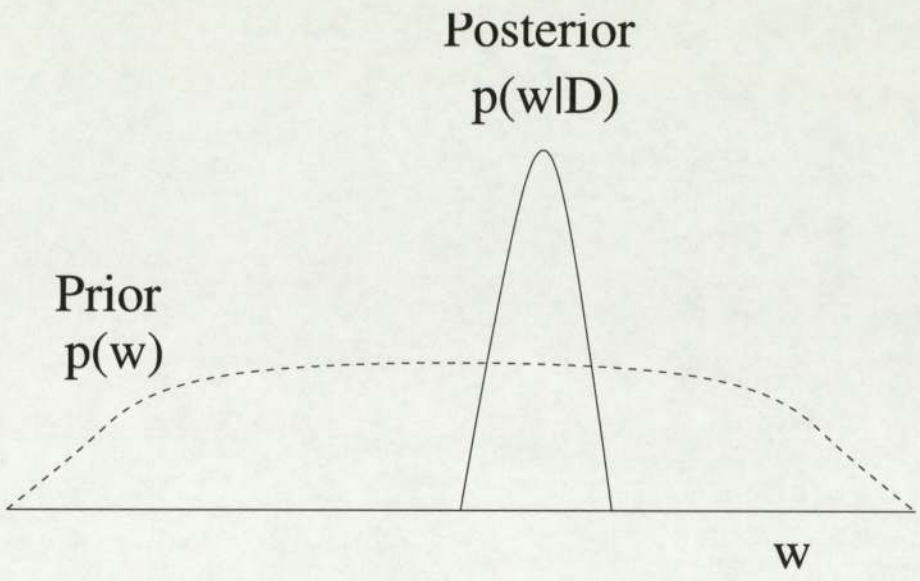


Figure 3.1: Schematic illustration of Bayesian inference for a parameter w. The prior distribution reflects our initial belief in the range of values of w. Once the data are observed, we can calculate the posterior distribution using Bayes' theorem. Since some values of the parameter will be more consistent with the data than others, the posterior distribution will be narrower than the prior distribution.

$p(D)$, known as the evidence, is a normalisation factor that ensures that the posterior distribution integrates to 1. It is given by an integral over the parameter space.

$$p(D) = \int p(D|w')p(w')dw' \tag{3.4}$$

$p(D)$ is constant as the observations does not change and is unknown, so :

$$p(w|D) \propto p(D|w)p(w) \tag{3.5}$$

Then, if we assume that the noise on each component D_i is independent and identically distributed (iid), we have :

$$p(w|D) \propto [\prod_{i=1}^{n} p(D_i|w_i)]p(w) \tag{3.6}$$

The distribution $p(w|D)$ is rarely tractable, so we need to do some approximations. There are two differents types of approximation. You can either approximate the posterior with a tractable distribution keeping probability nature or you can use a point

estimate: the maximum a posteriori (MAP) solution. The MAP solution consists of finding the w which maximises probability $p(w|D)$. The MAP solution is the standard regularization, provided that the noise is gaussian and additive, and the prior is a gaussian distribution. We need to put some smoothness constraints to do approximation, and the techniques that exploit these constraints are called generalization. The prior in the case of regularization framework represents the smoothness constraints of the model.

We can then write the error on the parameter, which is the negative log likelihood of the parameter :

$$\begin{aligned} E &= -\log(p(w|D)) \\ &= -\log\left([\prod_{i=1}^{n} p(D_i|w_i)]p(w)\right) + constant \\ &= -\sum_{i=1}^{n} \log(p(D_i|w_i)) - \log(p(w)) + constant \end{aligned} \tag{3.7}$$

Then, an estimate of the parameter can be retrieved using an optimisation method like gradient descent, or scaled conjugate gradient (SCG). In simulation, we will use a SCG optimisation method which is much more faster than a gradient method.

The goal is to retrieve the parameter using a MAP solution. Roughly, we are looking for the parameter w which minimises $-\log(p(w/D))$ the negative log-likelihood. The Bayesian process starts from an uniform field at the expected mean to converge as quick as possible.

3.3 Full Model

Our purpose is to create a statistical model which enables us to forecast precipitation. In this section, I will first introduce time series, statistical model, and describe the model that inspired us in the creation of our own model. Then, we will define in a very general way the model we developed.

3.3.1 Kalman filter method

The final objective is to predict, given a sequence of observations, the rainfall field, which can be seen like time series. A multivariate time series is a sequence of continuous d-dimensional random vectors X indexed by a time variable t. Suppose that at time t, we have seen a sequence of observations $X^T = [x_1, .., x_t, ..x_T]$ and that we wish to build a model from the sequence X^1 that enables us to predict the value of X at time $T+1$. Unfortunately, in most realistic cases, the observations are not deterministically related. Besides, because of the finite and limited size of the data, there will be undoubtedly a mismatch between our model and the true process.

Probability theory is a powerful tool for expressing uncertainty and randomness in our model. A statistical model is based on certain probabilistic assumptions that attempt to capture the essential characteristics of the data generation process. We do not believe that a model will represent exactly the true process but we hope we will

be able to develop tools that enable us to make decision for new data points. Given a model and a sequence of observations, the main purpose of time series is to estimate the parameters of the model by a statistical procedure, such as maximum likelihood, and make predictions.

We are now going to describe the probability theory we used in our model : state space and Kalman filter algorithm. The Kalman filter algorithm is largely explained through an example.

The state space representation is a compact description of ARMA (Auto Regressive Moving Average) models and is based on the result that any finite order difference or differential equation can be rewritten as a first order vector difference or differential equation. For instance, an AR (Auto Regressive) model of order 2 is written:

$$x_t + a_1 x_{t-1} + a_2 x_{t-2} = \epsilon_t \tag{3.8}$$

where x_t is a time series and ϵ_t is a noise term We can write :

$$\begin{bmatrix} y_t(1) \\ y_t(2) \end{bmatrix} = \begin{bmatrix} 0 & -a_2 \\ 1 & -a_1 \end{bmatrix} \begin{bmatrix} y_{t-1}(1) \\ y_{t-1}(2) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} \epsilon t \end{bmatrix}.$$

where $y_t(2) = x_t$ and $y_t(1) = -a_2 x_{t-1}$

In state space modelling, the observations are taken to be a combination of a set of variables called state variables which constitute the state vector at time t. This vector describes the state of the underlying system at time t. In the state formalism, the observation vector D_t is described by a state vector w_t, which contains information gained from previous measurements.

The state evolution for a linear discrete model is such that :

$$w_t = A_t w_t + B_t \mu_t \tag{3.9}$$

where μ_t is the noise term

A Kalman Filter is a sequential procedure for estimating the state vector at each time step for a linear model with Gaussian noise processes. Actually, the said procedure is composed of an evolution step and an update step. To give a concrete example, the all Kalman filter process, we give the example of one iteration in the filter with linear model. In this case, we wish to find the MAP estimate of the state time at time t given new data D_t and our estimate of the state at the previous time step $t-1$, i.e. $\hat{w}_{t|t-1}$. The Kalman filter operates in a Bayesian framework, the posterior distribution can be written :

$$p(w_t|D_t) \propto p(D_t|w_t)p(w_t|D_{t-1}) \tag{3.10}$$

We assume that each of these densities is Gaussian. As the model is linear in this case, we can define S_t such as:

$$D_t = S_t w_t + \epsilon_t \tag{3.11}$$

where ϵ_t is the noise. We assume the noise to be zero mean.

The likelihood contains all the information from the new data.

$$p(D_t|w_t) = N(S_t w_t, R_t) \tag{3.12}$$

The prior (see Equation 3.12) contains all the information obtained from the previous data. Assuming a Gaussian, this distribution will be centred on our estimate of the state vector at time t given all the data at time $t-1$, that is $\hat{w}_{t|t-1}$. The covariance matrix of the distribution will also be an estimate, denoted $\hat{P}_{t|t-1}$, which will be continually updating along with the state. The prior is then:

$$p(w_t|D_{t-1}) = N(\hat{w}_{t|t-1}, \hat{P}_{t|t-1}) \tag{3.13}$$

Finally, given the Gaussian assumptions above, the posterior distribution will also be a Gaussian centered on the best estimate given all the data up to now and including time t. The covariance matrix will our updated estimate of P_t. So we have :

$$p(w_t|D_t) = N(\hat{w}_{t|t}, \hat{P}_{t|t}) \tag{3.14}$$

In order to determine $\hat{w}_{t|t}$, we minimise the log posterior with respect to w_t. It can be shown that:

$$\hat{P}_{t|t} = S_t^T R_t^{-1} S_t + \hat{P}_{t|t-1}^{-1} \tag{3.15}$$

and that:

$$\hat{w}_{t|t} = \hat{w}_{t|t-1} + K_t \epsilon t|t-1 \tag{3.16}$$

where $K_t = \hat{P}_{t|t} S_t^T R_t^{-1}$ is called the Kalman gain

The equation (3.16) gives our new state estimate. This is the update step.

Now, the next step is to evolve the estimates of w while waiting for new data. In the update step, we have obtained the posterior distribution $p(w_{t+1}|D_t)$ (see Equation 3.12). We can write this probability in the following form:

$$\begin{aligned} p(w_{t+1}|D_t) &= \int p(w_{t+1}, w_t|D_t) dw_t \\ &= \int p(w_{t+1}|w_t, D_t) p(w_t|D_t) dw_t \\ &= \int p(w_{t+1}|w_t) p(w_t|D_t) dw_t \end{aligned} \tag{3.17}$$

where $p(w_{t+1}, w_t)$ is the joint distribution of the random variables w_{t+1}

The second step comes from the Bayes' rule and the last one follows from the fact that w_{t+1} depends on the data history omly through w_t.

We notice that the posterior from the previous time step has appeared within the integrand. It has already assumed to be $N(\hat{w}_{t|t}, \hat{P}_{t|t})$ distribution. $p(w_{t+1}|D_t)$ represents the prior distribution of the next time step (see Equation 3.12). It is assumed to be a $N(\hat{w}_{t+1|t}, \hat{P}_{t+1|t})$. The expression for $p(w_{t+1}|w_t)$ is the state evolution equation 3.9. The noise is assumed to be zero mean with covariance M_t. So we can show that

$$p(w_{t+1}|w_t) = N(A_t w_t, B_t M_t B_t^T) \tag{3.18}$$

By substituting these expressions in 3.17, we obtain the following equations which complete the Kalman filter algorithm.

$$\hat{w}_{t+1|t} = A_t \hat{w}_{t|t} \tag{3.19}$$

$$\hat{P}_{t+1|t} = A_t \hat{P}_{t|t} A_t^T + B_t M_t B_t^T \tag{3.20}$$

3.3.2 State space and observations

Data is available from radar which gives us the rainfall intensity that we will denote I_t, which are the observations. We are using the following variable presented in Table 3.1 :

I_t	Observation
R_t	Rainfall field
u_t	First component of the advection vector
v_t	Second component of the advection vector
G_t	Growth/decay term

Table 3.1: Observation and state space

The radar data, I, provides the observations which are a noisy realisation of R. The error in the observation is assumed to be uncorrelated. We will assume a Gaussian noise over the data.

All these variables are supposed to be random variables and compose the state space. The precipitation field and the advection fields are stochastic processes at each space location.

G is also in theory a stochastic process at each space location. But due to its complexity, it won't be considered as a state variable in the development of the model. G will disappear from the state variable to be included in the model noise . But in this section and the next one which presents our model in a theoretic point of view, G is treated as a state variable.

3.3.3 Statistical model

Our state space is composed by the rainfall field R_t, the advection field $(u, v)_t$ and G_t.

We will pose this general state space model :
Forecast step :

$$p(\hat{R}_{t+1} \mid R_t, u_t, v_t, G_t) \tag{3.21}$$

where $\hat{R}_{t+1}$ is the estimate of R at time $t+1$
Update step :

$$p(R_{t+1}, u_{t+1}, v_{t+1}, G_{t+1} \mid \hat{R}_{t+1}, I_{t+1}, R_t) \, \alpha \, p(u_{t+1}, v_{t+1}, G_{t+1} \mid R_{t+1}, R_t) \, p(R_{t+1} \mid \hat{R}_{t+1}, I_{t+1}) \tag{3.22}$$

The only assumption on this basic model is that the advection equation holds and that the observations from time to time are not independent. This model is general, and we have a lot of choice over methods used to model the state variables. The intention is to use statistical model with a partial differential equation based approach. Within a Bayesian framework (see Section 3.2), the equation of the update step is defined as the posterior distribution of the state variables given the forecast (prior) and the observation (radar).

We can represent this model by this graphical model (see figure 3.2).

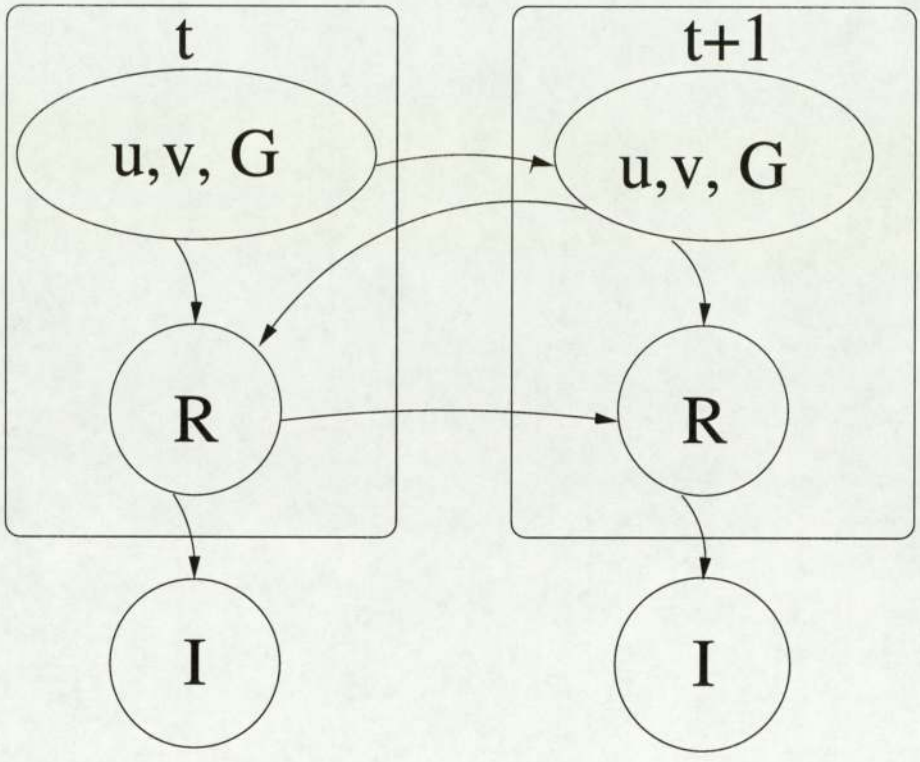


Figure 3.2: General graphical model representing links between state space variables.

The model we posed is very general and it will function in a similar way to a Kalman filter (see Section 3.3.1). The same method will be applied to evolve and update the states.

The forecast step corresponds in Kalman filter term to the evolution state. In the forecast step, we will integrate the advection equation with respect to time. Given the advection field and the growth/decay term at time t, and the rainfall at time t, as the advection equation decompose R_{t+1} by a combination of R_t, $(u, v)_t$,G_t, we will first compute the evolution of the rainfall field and then the advection field and the growth/decay term. This introduces a hierarchy into the steps defined in Kalman filter. The computation of the forecast step is not easy because we need to estimate probability of the current state R_t by using our differential equation, which is also not trivial to integrate. Here we will need to again use some approximations.

In the update step, we will condition our state space model to the observation I. We can say much about this step because we need to define model over our state space. We specify model classes on R,u,v, and G through the definition of appropriate prior (we are in a Bayesian framework). In these specifications of priors, we try to incorporate the properties listed in the previous section.

Besides, there are several pieces of prior knowledge we would like to incorporate to this model concerning precipitation and its short term evolution, these include:

- $R \geq 0$
- The rainfall advection fields are continous processes in space and time
- The space and time rate of change of the advection is much slower than that of

the rainfall

- The observations I are a noisy realisation of R

Finally, the model posed is very general and can lead to lot of different developments. It depends on how the variables of the state space will be modelled.

To develop our model, we have to specify a model over our state space. This is clearly needed by both of the model steps described in this section. The growth/decay term will not be modelled in first instance and will encompass the error model. The advection field has not been modelled yet and nor has the rainfall. This last one is the variable which has the most of constraints. We have several pieces of physical information we want to represent, and this modelling will be the hardest one.

But before trying to model our state space variable, I will introduce the first work using the advection equation and using the 2 steps described in this section (Equation 3.21 and 3.22. This work makes a lot of simplifications on the nature of all the variables and makes use of techniques to approximate partial derivatives.

Chapter 4

Preliminary work

The final model has been introduced in Chapter 3. The variables introduced previously, i.e. the rainfall R and the advection field (u, v), and the growth/decay term G are not modelled, but have to be continuous. However, in this chapter, G will not be taken into account, and R will still be represented by the pixel model with uncorrelated errors. Actually the rainfall field R and the observation I are equivalent. In this chapter, we build a first model of quantitative forecastprecipitation. This model is not a probabilistic model. But it is a first step in the process of creation of the final fully probabilistic model. There is no forecast step (Equation 3.21). The forecast uses the current value of the advection field and the advection equation to compute the rainfall field at the next time step.

We will first deal with the initialisation of the advection field, then with the simplifications made in the model, and finally we will give some results.

4.1 Initialisation of the advection field

The initialisation of the advection field consists of giving an estimate of u and v at time $t = 0$. Given 2 successive observations, and by using the advection equation (3.2), we can compute a estimate of the advection vector. The initialisation is a crucial part of a model as we want to find the best estimate possible.

In this chapter, the growth/decay term is completely neglected because of its complexity. We do not have enough information about this term to incorporate it in the model. So we can simplify the expression of the advection equation:

$$\frac{\partial R}{\partial t} = -u \frac{\partial R}{\partial x} - v \frac{\partial R}{\partial y} \tag{4.1}$$

The equation (4.1) can be seen as linear equation with 2 unknowns u and v, if $\frac{\partial R}{\partial t}$, $\frac{\partial R}{\partial x}$, $\frac{\partial R}{\partial y}$ are fixed. This is the case at a given time. In our case, we deal with the initialisation which takes place at time $t = 0$.

The problem here is to compute an estimate of the partial derivatives of the rainfall field with respect to space and time given two observations. The solution presented below makes use of the finite difference technique [4]. The finite difference techniques

consist of using Taylor series at the second order (Equation 4.2) and subtracting them to compute derivatives.

$$f(x \pm \Delta x) = f(x) \pm \Delta x f'(x) + \frac{\Delta x^2}{2!}.f''(x) + O(x^3). \tag{4.2}$$

Equations (4.3), (4.4), (4.5) are obtained from the equation (4.2) :

$$\frac{\partial R}{\partial x} = \frac{R(x + \Delta x) - R(x - \Delta x)}{2\Delta x} \tag{4.3}$$

$$\frac{\partial R}{\partial y} = \frac{R(y + \Delta y) - R(y - \Delta y)}{2\Delta y} \tag{4.4}$$

$$\frac{\partial R}{\partial t} = \frac{R(t + \Delta t) - R(t)}{\Delta t} \tag{4.5}$$

This enables the computation of an estimate of the rainfall field at the following time step, if we know (u, v) and R at the current time. This is the decomposition of R. Using the finite difference, we transform the advection differential equation into the following linear equation :

$$A.u + B.v = C \tag{4.6}$$

where R_0 and R_1 are the rainfall field at the respective time $t = 0$, $t = 1$, $A = \frac{R_0^{x+1,y} - R_0^{x-1,y}}{2\Delta x}$, $B = \frac{R_0^{x,y+1} - R_0^{x,y-1}}{2\Delta y}$, and $C = \frac{R_1^{x,y} - R_0^{x,y}}{\Delta t}$

This linear equation has 2 unknowns, and we haven't got any other equation to create a equation system. This is underdetermined. So we assume that (u, v) is locally constant on an area using again this decomposition by block of the data matrix (see Figure 2.2).

We do not want either the size of each block to be too small to avoid inaccurate result either to be too big to avoid having one single value for a big part of the map. We have tried different size of block among the plausible values: 5,10,15,20,30,40. The results are quite similar with these values, so from now the size of each block chosen is 20. With this approximation, we have now 20*20 equations and always 2 unknowns. So we are able to find the least mean square error solution of the overdetermined system.

To avoid solving the advection equation in block where there is no rain, a threshold has been fixed and the initial advection vector is computed only if the sum of the precipitation in this block is more than this threshold. The resolution of the system gives an estimate of the advection vector on each block. Then we average this vector over all the blocks to have one single value as estimate of this vector on the whole map.

Once we have solved this system which computes an estimate of the initial value of the advection vector (u, v), we are able to start a forecast by using the decomposition of R through the advection equation.

4.2 Process

This section presents the first model. In this first model, a lot of simplifications have been done. First, the forecast step (3.21), which integrates the advection equation and which corresponds to the state evolution in Kalman filter term (see Section 3.3.1), does not exist in this model. As the model for R is a simple pixel model, R and I are the

same. The update step is only an update of the estimate of the advection vector. The forecast used in its first sense predicts quantitative precipitation.

4.2.1 Gaussian process

The goal of the update in this case is to update the advection. As we are working in a Bayesian framework (see Section 3.2), we need to define a prior over the advection vector (u, v). A Gaussian Process has been chosen to model the advection field. A Gaussian process is a family of random variables $y(x)$, such that for any finite collection the joint distribution of $y(x_1)...y(x_n)$ is Gaussian.

A Gaussian Process prior seems to be a good model for the advection vector because it allows to control the vorticity and divergence in the advection field and to control the length scales and prior variance of the processes. We can add some constraints to encourage advection fields with very small divergences. Indeed, the rotational component of the flow is much larger than the divergent component. This is convenient because the vorticity and the divergence are two parameters of this model we can set up.

4.2.2 Update of the advection

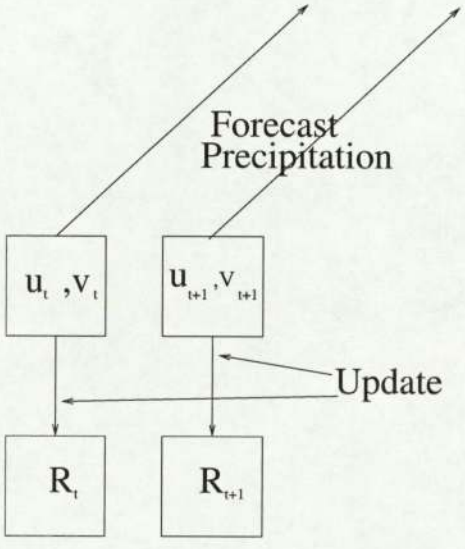


Figure 4.1: Simplified model. R and I are the same as we assume a simple pixel model. The forecast step has disappeared from the figure 3.2 Forecast is used here in its first sense.

Given that the growth/decay term is neglected and that R is equivalent to I, we have only one variable the advection field in our state space (see Section 3.3.2. Because of this simplification, the expression of the update step (Equation (3.22)) can be rewritten. It remains only the posterior distribution $p(u_{t+1}, v_{t+1} \mid R_{t+1}, R_t)$ of the advection given the rainfall field at time t and $t+1$. The equation obtained using Baye's rule then for the update is:

$$p(u_{t+1}, v_{t+1} \mid R_{t+1}, R_t) \, \alpha \, p(R_{t+1} \mid u_{t+1}, v_{t+1}, R_t) \, p(u_{t+1}, v_{t+1}) \tag{4.7}$$

The idea is to find the MAP solution to update the advection equation. We have already defined the prior, and we need to be able to compute the likelihood of the rainfall a time $t + 1$ given the advection at the same time and the rainfall at the previous time. As the error in the rainfall model is assumed to be uncorrelated, we can rewrite the expression of the likelihood term:

$$p(R_{t+1} \mid u_{t+1}, v_{t+1}, R_t) = \Pi_{i,j} \, p_{i,j}(R_{t+1} \mid u_{t+1}, v_{t+1}, R_t) \tag{4.8}$$

We assume that this likelihood term is a Gaussian over all the locations.
We can write the error on the parameter (equation 3.7):

$$E = -\sum_{i,j} \log(p_{i,j}(R_{t+1} \mid u_{t+1}, v_{t+1}, R_t)) - \log p(u_{t+1}, v_{t+1}) + K \tag{4.9}$$

where K is a constant

The equation 4.9 is the basis for the update of our model and will be optimized by minimising (see Chapter 3 Section 3.2) using a scaled conjugate gradient method to the find the MAP solution. .

4.2.3 Forecasting

In this first forecasting model, the forecast step used is not probabilistic but deterministic. The advection equation is used again to compute from the advection vector and the previous value of R the next forecast values of R. The finite difference approximation is reused.

$$\hat{R}_{t+1}^{i,j} = R_t^{i,j} - u_t^{i,j} \frac{\delta t}{2\delta x}(R_t^{i+1,j} - R_t^{i-1,j}) - v_t^{i,j} \frac{\delta t}{2\delta y}(R_t^{i,j+1} - R_t^{i,j-1}) \tag{4.10}$$

where the advection field is supposed to be locally constant.

We are aware that this model is too simple to be efficient. This model uses the finite difference technique approximation for the initialisation of the advection vector, but also for the forecast. This approximation is unstable. As we are using it often, we are aware that the result will probably be inaccurate. Besides, the forecast is deterministic. The evolution of the rainfall field can not be predicted with such a simple model.

In this chapter, the rainfall field has not got any model yet, whereas the advection field is assumed to be a Gaussian process.

4.3 Results

4.3.1 Test on simulated data

The initialisation has been tested on simulated data and the method coded in this model has been compared with a cross correlation method.

We have implemented a method which computes the cross correlation between two images. The idea is to translate the image of $\pm n$ pixels through the x axis, and $\pm m$ pixels through the y axis. The value of the advection vector is proportional to the number of pixels whose the image translated gives the biggest correlation. In the case of simulated data, the true parameter value is known, which enables us to know how accurate is the result of this initialisation. The table (4.1) gives the results obtained when we simulate different values for the advection vector (u, v), using the solution of the advection equation when using the finite diffence techniques and the technique called cross-correlation comparing two imagesS.

We have to be clear about thoses values. We are simulating two images. The second one is just a translation of the first one. The couple of values in the table in the column called Simulated represents in the coordinates system the translation, expressed in pixel, along the x and y axis. The function cross-correlation enables us to check we have created without any mistakes the simulated data.

The advection vector is expressed in $m.s^{-1}$. The value given in the table 4.1 are actually the result of the operation $(u, v) * \Delta t/(2 * 1000)$, where Δt is the time step between two images, 1000 is to pass in km, and 2 because one pixel is worth 2 km.

The results of the computation of the advection vector using this advection equation give really inaccurate values. We notice in this table that globally when the translation between those images increases, the accuracy decreases significantly.

These results are not really usable and we are not expecting accurate results on real data. This issue can be explained by the fact the partial derivatives are approximated by finite difference, and these techniques are very unstable. When the translation between two images is important, as to initialise the advection vector we are using this block decomposition, the image at time t and at time $t + 1$ are completely different.

The kind of problem occurs as well when the time step is too big, since the translation is determined by the move speed of the object in the image, and the time step between those 2 images.

4.3.2 Test on real data

This section presents the results obtained when running the model on our real data. The first part in this model is the initialisation of the advection vector at time $t = 0$ (section 4.1).

The figures (4.2) and (4.3) show the results of this initialisation:

This computed initial advection vector gives the general direction of the advection, but some arrows are clearly wrong. Some arrows are much longer tha the others, these arrows are wrong. The advection is supposed to be a continuous process, so we can not have such differences between surroundings arrows. We have an idea of this general direction of the advection because we have the observations at the following time step, and we have built from all the observations a movie (displaying in real time

Advection		
Simulated	Advection equation	Cross-correlation
1,0	1.03,0.02	1,0
2,0	1.14,0.24	2,0
3,0	1.17,0.22	3,0
4,0	1.2,-0.02	4,0
0,1	0,0.99	0,1
0,2	0.12,1.16	0,2
0,3	0.19,1.32	0,3
0,4	0.22,1.39	0,4
1,1	0.95,0.73	1,1
2,1	1.06,0.57	2,1
2,2	1.01,0.65	2,2
Each value is expressed in pixel		

Table 4.1: Results obtained on simulated data. The column "Simulated" represents in pixels along the x and y axis the true translation which we have performed on the image. The column "Advection equation" gives the result in pixels (i.e. 2km) obtained when using the advection equation to find the translation and, finally the last column gives the results obtained when using the cross-correlation method. For instance (4^{th} line of the table), when we have translated the image of 4 pixels along the x axis and 0 along the y axis, the cross-correlation method finds the true translation whereas when using the advection equation, we find 1.2 pixel along the x axis and almost nothing along the y axis. These results show that the initialisation of the advection does not give accurate results. These values represent pixel on the image (i.e. 2 km each pixel).

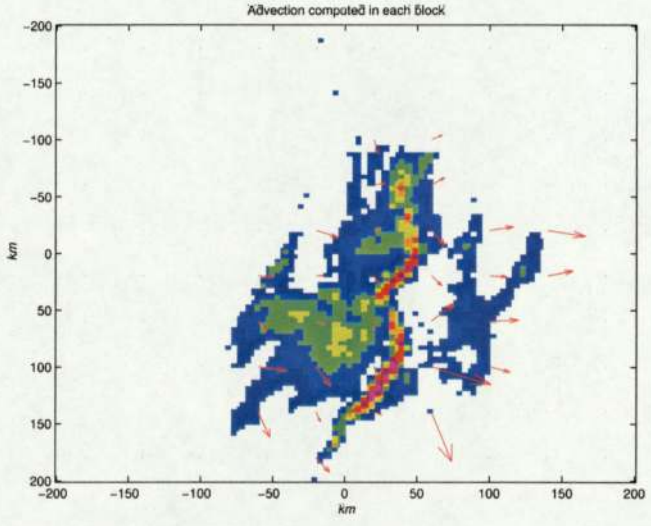


Figure 4.2: Computed advection on real data. Arrows represent the advection vector. Data from 30^{th} October 2000 6h45am and 6h50am.

our sequence of images) which gave us an idea of the advection.

We were expecting inaccurate results with real data as we did not get good ones

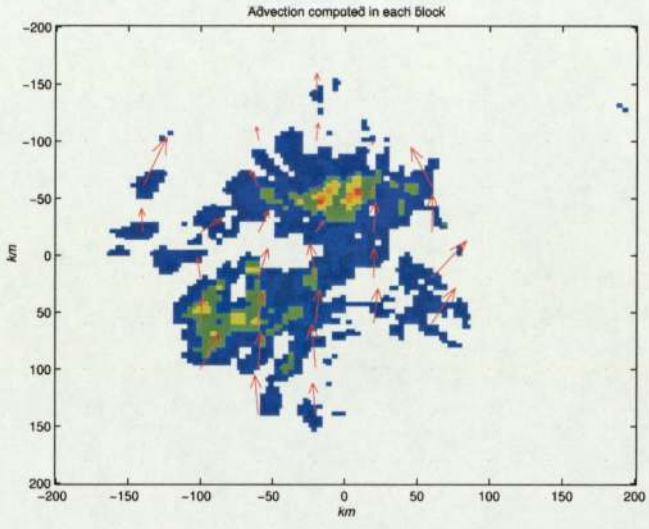


Figure 4.3: Computed advection on real data. Arrows represent the advection vector. Data from 3^{rd} April 2000 9h30am and 9h35am.

with simulated data. The results obtained when running this model are especially inaccurate. Results of the forecast from this model are not presented here as there are too inaccurate. The forecast part gives worse results than the initialisation. The forecast uses the equation (4.10) at each time step. This equation makes use again of the finite difference techniques to compute the partial derivatives.

The finite difference technique are something to use with care. The way of solving of the advection equation (4.1) we used is certainly not the most clever one.

The use of finite difference to compute derivatives is banned in our case. The finite difference techniques are really instable, especially in weather radar data, where the value from one pixel to another can change drastically.

In this chapter, we have defined the model for the advection field. The advection field is going to be a Gaussian process.

As this technique can not be used, we have to find a model for the rainfall field which will enable us to compute the derivatives of the rain with respect to the space and the time. In the building of our model introduced in Chapter 3, the next step is thus to find a continuous model for the rainfall field which takes into account the constraint concerning the partial derivatives.

Chapter 5

Rainfall Model

The rainfall field R and the growth/decay term G introduced in Chapter 3 are not modelled. The advection field (u, v) will be modelled by a Gaussian process.

R is the variable we are most interested in as the goal of our model is to forecast precipitation. We are going to propose a working model for the rainfall field R.

The discrete pixelised model used initially is not good enough and the main point is that since Chapter 3 the rainfall field is constrained to be a continuous model in space and in time. In this Chapter, we cope with the space continuity of this field.

The rainfall model R is to represent the data, which are our noisy radar observation I.

In this chapter, we will first introduce the model chosen, then its development and finally we will test the model on real data.

5.1 Theoretic Support

5.1.1 RBF network

Radial Basis Functions (RBF) are related to kernel methods for density estimation, regression and to normal mixture models. RBF models [8] measure the distance between an input vector x and a parameter μ_i (i is the hidden node index) with some weighting coefficients w.

The idea of a RBF model is to approximate a given function f using a set of basis function Φ. Φ is a non-linear function to be chosen. The output is then taken to be a linear combination of the basis functions :

$$f(x) = \sum_i w_i \Phi_i(\| x - x^i \|) + w_0 \tag{5.1}$$

where w_i is the weight associated to the i^{th} basis function, w_0 the bias.

A radial basis function network uses several RBFs as hidden units (See Figure 5.1).

Several forms of basis function can be used, the most commonly used are the Gaussian and the thin-plate spline, the thin-plate spline basis function is :

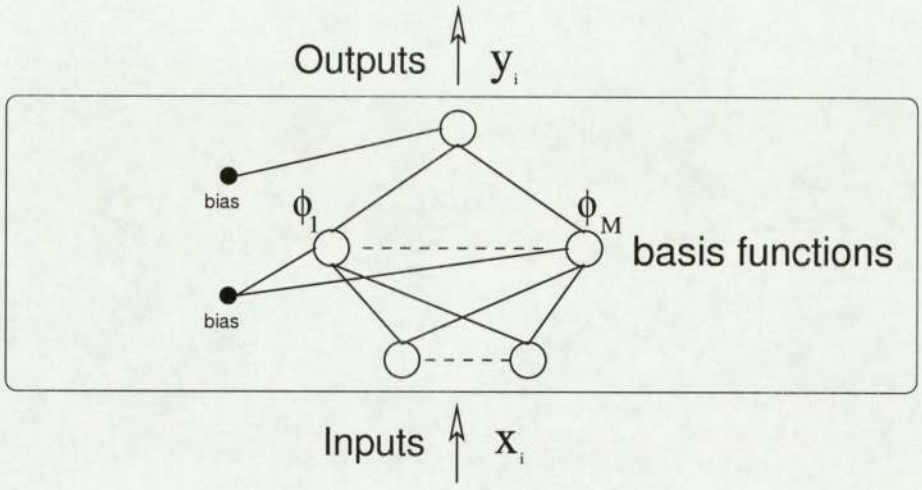


Figure 5.1: Radial basis function network

$$\Phi(z) = z^2 \ln(z) \tag{5.2}$$

The Gaussian basis function can be expressed by:

$$\Phi_j(x) = \exp(-\frac{\| x - \mu_j \|^2}{2\sigma_j^2}) \tag{5.3}$$

where x is the input vector and μ_j is the vector determining the centres of the basis function Φ_j. The interpolation formula (5.1) is then:

$$y = f(x) = \sum_{j=1}^{M} w_j \Phi_j(x) + w_0 \tag{5.4}$$

where x is the input vector, y is the output vector, σ_j the width and M is the number of basis function.The number of basis functions needs to be less than the number N of data points.

Each basis function has its own width σ_j, which controls the smoothness properties of the interpolating function. The Gaussian is a localised basis function with the property that $\lim_{x\to\infty} \Phi(x) = 0$. This property is interesting in our case as we have a compact working space.

For a large class of basis functions, RBF networks are universal approximators [13]. Besides, they posess the property of best approximation, which means that the set of functions corresponding to all possible choices of the adjustable parameters includes the optimal approximation.

The main advantage of this network's family is that RBF models are very fast in comparison to networks with sigmoidal units. Once the basis functions have been

chosen, we have a simple model whose parameters can be found by a least squares procedures, or any other optimisation procedure. The main drawback of this network is that it becomes impractical with input vectors of large dimension (the curse of dimensionality).

5.1.2 The chosen model

There are a lot of possible parameterisations of the precipitation fields. In the first instance, we were using a simple model with pixel descriptions, and uncorrelated errors (Chap 4), which actually comes to use the raw radar observation.

After having observed the structure of the rainfield in Chapter 2, we have thought that to represent this type of data, we could use a RBF network to fit the data.

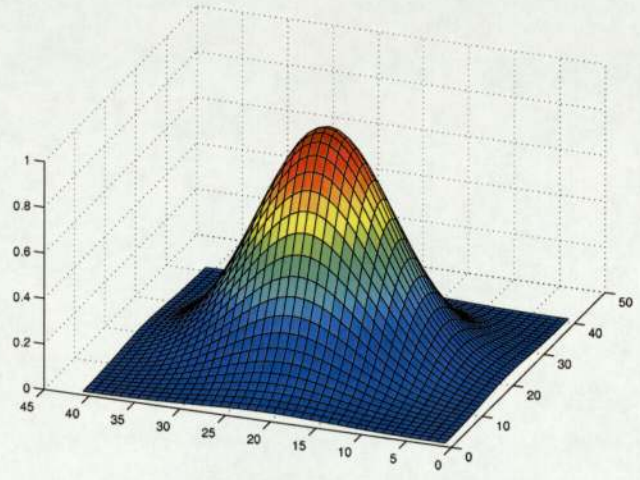


Figure 5.2: Gaussian basis function of the chosen RBF

Actually, we are using this kind of basis function to fit the rainfield (Figure 5.3). This basis function is a gaussian, the usual basis function employed in RBF models. It enables us to fit gaussian on precipitation field distribution. The observation of image in 2D and in 3D of the observation data in Chapter 2 gave us the idea to use RBF network. It allows to smooth the rainfall field and to have a continuous model. Each bump can be thought of as representing a precipitation cell. The main advantage cited in the previous section of the use of an RBF is important because in Nowcasting, the time required to fit the data is crucial and we want to be as fast as possible. The main drawback cited in the previous chapter is not concerning as the dimension of the dataset is only 2. In our case, use of RBF network seems to be relevant.

The model rainfall field R presented here is differentiable with respect to x and y, as it is a sum of differentiable functions. We reach here with this model another advantage. We will have to compute derivatives with respect to space to solve the advection equation (Equation 3.2). We will not have to make use of any approximation like the finite difference to compute the partial derivatives. The error will then come from the differences between the dataset and the model used for the rainfall field.

With this model, the rainfall field is determined by the following parameters:

- c the centers of the Gaussians.
- h the height of the Gaussians.
- w the width of the Gaussians.

These three parameters are vectors whose size is given by the number of cells (i.e. Gaussians) used in the RBF network to determine the rainfall.

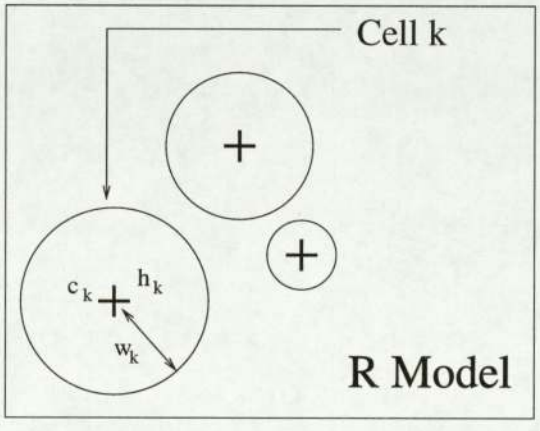


Figure 5.3: Graphical representation of the rainfall field model. c_k is the center, h_k is the height, and w_k is the width of the cell k

In the case of the rainfall field, the weight of each basis function represents the rainrate on the center of this cell.

The initialisation of the center, of the widths and of the weights is really important to fit well the model as well as to have the quickest computation possible. Unlike a proper RBF network where only the heights are updated, here the center and the width will change as well during the iteration in the filter (see Figure 3.2). A scaled conjugate gradient is applied to the data to fit the model. As defined in Chapter 3, the QPF model we want to develop is within a Bayesian framework. We need to define priors over all our state space composed by R, (u, v), and G. That means with the model adopted for R that we have prior on the rainfall parameters : centers(x_c,y_c), the width of each cell h, and the height of each cell h.

These priors constrain our initial estimates of the state. We assume diffuse Gaussian process priors over the centers c of the form $N(0, \Sigma_c)$, where $N(\mu, \Sigma)$ stands for the Gaussian distribution with mean μ and covariance Σ. Σ_c is a full covariance matrix. It ensures that the centers can be correlated. The priors over the heights and the widths are log-Gaussian. This ensures $R \geq 0$, this matches with one of our desires expressed in Chapter 3 and constrains the value to be realistic.

Gaussians are convenient, and widely used in this project because to define a Gaussian, we only need to define a mean and a covariance matrix and also because of their properties (multiplication, addition).

5.2 Development

In this section we explain the different steps of the development of the rainfall model: the initialisation and the optimisation of the parameters to fit the data. This development corresponds to the training of an RBF network. We are using the noisy radar data observation I to train this network.

5.2.1 Initialisation

The initialisation consists of setting initial values for the parameters of R. We want to choose good starting points for the centers, the widths, and the heights of the cells. Find good starting means to set values for the parameters which are the closest possible to the true value. This process is quite important as we want to have to learn the parameters determining R as fast as possible. The better the initialisation, the faster the optimisation will be. We implemented and compared two different initialisations. The starting point to initialise the model for the rainfall field is an observation. In the first initialisation, the number of cells (i.e. number of centers) is fixed and has to be hand chosen. The observation I is first smoothed. A function to find maxima in a matrix data is then applied. The basic idea is to find the biggest value in the matrix parameter of this function and then to set this maxima and the pixels surrounding this maxima to zero which implicitly means that the initial value given to the width is 1 pixel. With this method, only one maxima is found in this region. This is done iteratively and stops when the number of cells chosen is reached. Then the index of the maxima found in the observation is used to give a starting point to the centers. The index corresponds to the index of the observation matrix.

We have to define a grid over the observation to make more easily understandable what represents the index on the observations and to show how we place centers on the map (see Figure 5.4).

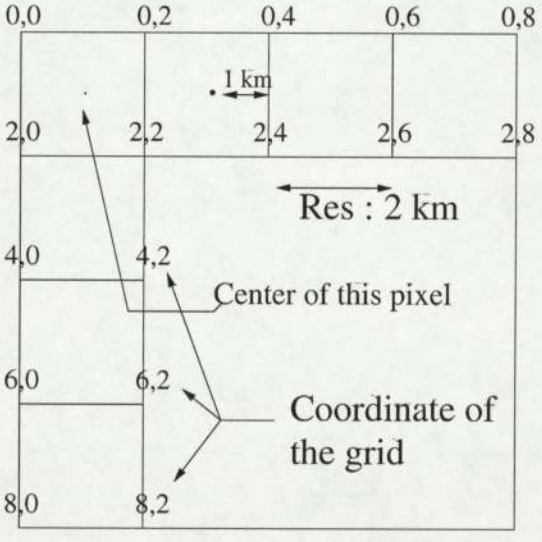


Figure 5.4: Grid on the map. This figure shows the coordinates system defined over the map obtained with radar data.

The centers are not put exactly on the point where the maxima has been found. Random numbers r_x and r_y are added because we don't want to have the center right in the middle as it won't be realistic. Each value of the observation matrix I represents a 2 km by 2km square in the data sets we used in this study (see Chapter 2). There is no reason to put this straight in the middle of the pixel, and as you have not any knowledge on where to put we add these random numbers. The heights are chosen close to the value of the maxima and the weights worth 1, due to the method used to find the center. The principal problem with this initialisation is that the number of cells is fixed.

$$x_c^{init} = xgrid(maxi, maxj) + \frac{res}{2} + r_x \tag{5.5}$$

$$y_c^{init} = ygrid(maxi, maxj) + \frac{res}{2} + r_y \tag{5.6}$$

where x_c^{init} and y_c^{init} are the center's coordinates, xgrid is the grid in accordance with the x abscisse applied to the map, ygrid is the one in accordance to y coordinate, (maxi,maxj) is the index of the maxima and r_x and r_y are random numbers between 0 and 1.

The second way of initialising is quite different. In the previous initialisation process, we were setting all the parameters first without any optimisation method and the network was ready for the training. In this case, we are searching for the biggest value of the observation matrix. The first center is then found. The height and weight are chosen in the same way as in the previous initialisation, but then a quick first optimisation is done. So a cell is fitted to the observation and then this first result is subtracted from the observation data and the same process is done again iteratively using as observation the subtracted resulting we call updated observation. This iterative process continues until all the remaining value in the updated observation matrix are under a threshold (0.5 $mm.h^{-1}$ in practice).

5.2.2 Optimisation

Once the process has been initialised, an optimisation of the centers, the widths and the heights is done. This optimisation consists of trainning our network with our observation I.

Using Bayes' rule, we obtain :

$$p(R|I) \propto p(I|R).p(R) \tag{5.7}$$

We have defined in the Subsection 5.1.2 a prior for the rainfall field. All the parameters which determine R have a prior (Gaussian for the centers, and log-Gaussian for the heights and the widths). The optimisation is done using the negative logarithm of this equation (see 3.7).

$$-\log(p(R|I)) = -\log(p(I|R)) - \log(p(R)) + K \tag{5.8}$$

where K is a constant.

We assume a gaussian noise over the radar data I. So,

$$p(I|R) = \Sigma_{i,j} \frac{(I_{i,j} - R_{i,j})^2}{\sigma_{noise}} \tag{5.9}$$

where σ_{noise} is the variance of the noise

This optimisation used again a SCG to determine the MAP solution of all the parameters which determine R.

5.3 Test

The goal here is to test the efficiency of the model at fitting real data. This real data is our observation I provided by the Met office (see Chapter 2). We will do a comparison between our different initialisations, and then choose the one which is the more interesting for our QPF model. The most important constraint we have is the time required by the model to fit the data.

5.3.1 Error measure

To have an idea of this efficency, we define an error function. The error measure used to analyse the results of this RBF model is the root mean squared error:

$$E = \sqrt{\frac{\sum (R_{i,j} - I_{i,j})^2}{N^2}} \tag{5.10}$$

where $R_{i,j}$ is the network output, $I_{i,j}$ the target, and N is the number of elements. This error measures the average error in $mm.h^{-1}$ on each pixel.

5.3.2 Test on real data using first initialisation

The model we developed has been tested on real data. It consists of training our network on the radar observation provided I. The Table 5.1 shows the errors obtained when using the first initialisation described in the previous section of this chapter in function of the number of cells chosen. Tests have been realised on two different sizes of data. The figures (5.5) and (5.6) are the images of the real radar data and of the result given by the model. The difference between this 2 couples of figures is the size of the data used.

The results show that this initialisation give interesting accuracy. But the RMSE shows that the results do not really vary with the number of cells but depends more on the data. As the number of cells has to be fixed at the beginning of the process, it matters because we do not really know which value to give to this parameter. Really different results happen when the number of cells is changed. Besides, we can fix the number of cells but as soon as the data will change, the number of cells will have to be changed. Moreover, time required to fit this model to the observation is very long and in nowcasting, we need to be as quick as possible.

Map size	60*60 km^2	100*100km^2
Number of cells	RMSE	
5	3.69	3.14
10	2.53	4.77
15	2.14	3.78
20	1.87	4.25
40	1.39	4.15
60	3.43	2.13
80	1.08	1.12
100	1.06	4.19
120	4.96	2.66
140	4.36	3.35
160	4.73	0.89
180	3.14	0.86
200	4.24	0.93
220	4.26	4.41
240	1.08	4.37
260	1.17	0.86

Table 5.1: RMSE in accordance with the number of cells. Two different size have been used : 60km by 60km and 100km by 100km. Data from the 30^{th} of October 2000 at 6:45 am.

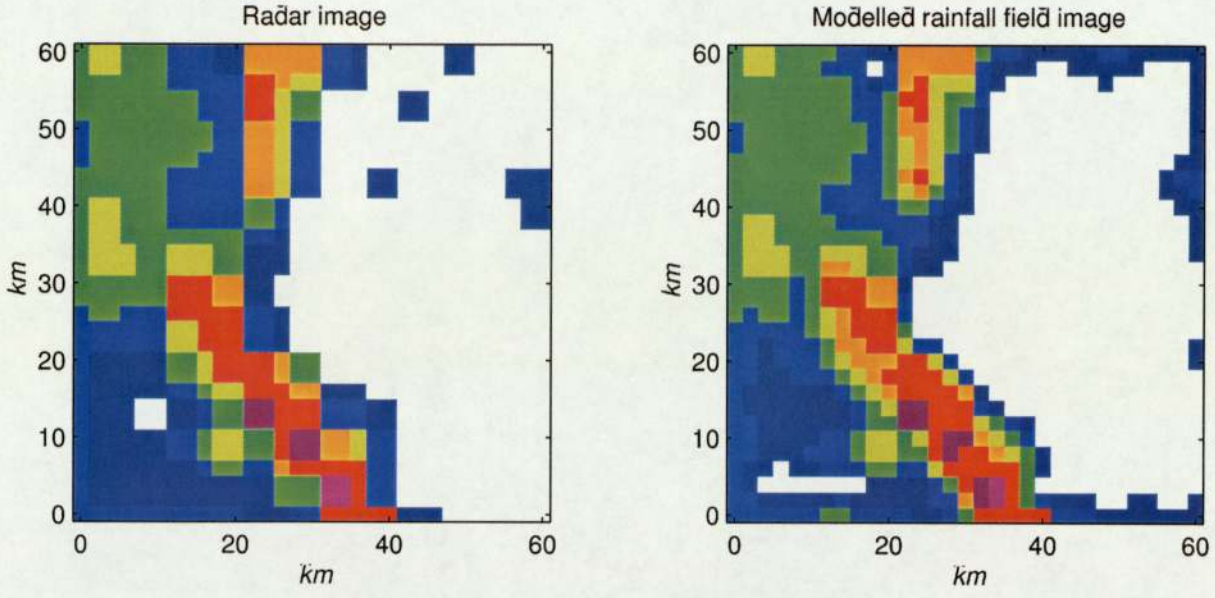


Figure 5.5: Image of the observation over 60km by 60km and of the rainfall field modelled from this observation with 100 cells. Realised with the first initialisation (data from measurement made the 30^{th} of October 2000 at 6:45 am).

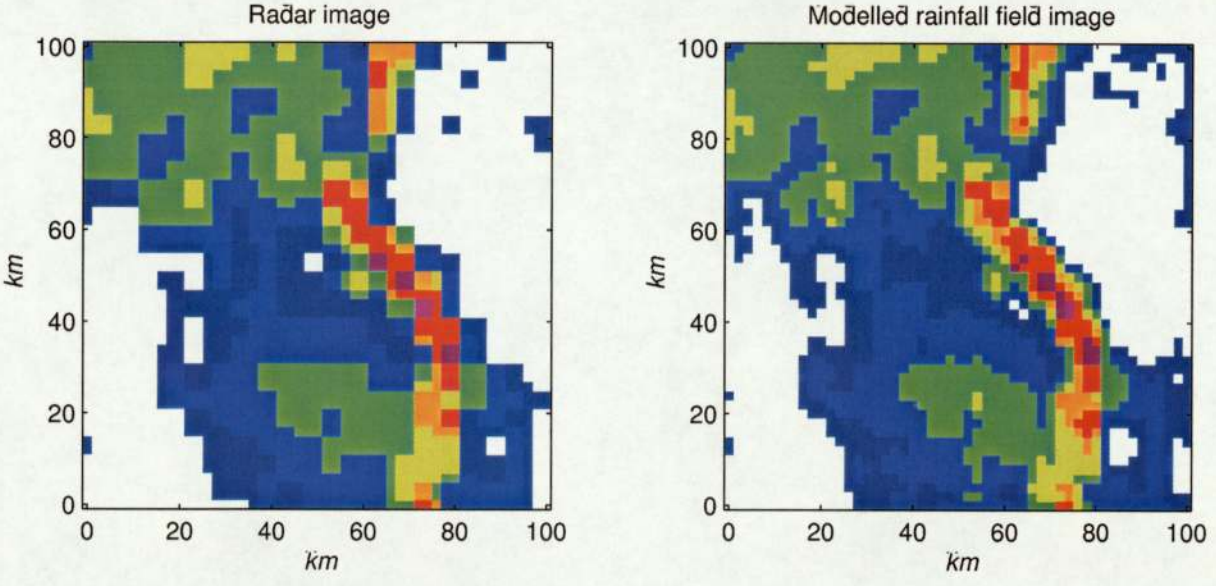


Figure 5.6: Image of the observation over 100km by 100km and of the rainfall field modelled from this observation with 260 cells. Realised with the first initialisation (data from measurement made the 30^{th} of October 2000 at 6:45 am).

5.3.3 Test on real data using second initialisation

The results are not as promising as the one for the other initialisation (see Table 5.2).

Map size (km^2)	Number of cells	RMSE
40*40	9	1.79
60*60	13	2.40
80*80	17	2.37
100*100	20	2.17
120*120	21	1.96

Table 5.2: RMSE and number of cells in accordance with the size of the map used. This test is realised using the second initialisation (data from the 30^{th} of October 2000 at 6:45 am).

This initialisation presents a lot of advantages over the previous one :

- The second initialisation is quicker than the first initialisation, especially when the size of the data is not too big (matrix is 40 by 40 or less).
- The number of cells is not fixed and is determined by the algorithm. This problem in the first initialisation is really critical as the accuracy of the results will change drastically as soon as we test new data. The number of cells which fits best the model depends on the data and should not be fixed.
- Test realised on real data showed that this algorithm does not create too much center. Using the first initialisation with the same number of cells as in the second will give really inaccurate results. For instance,with a map of 60km by 60km, the second initialisation determines 13 cells and the RMSE is 2.4. With 15 cells in the first one, the RMSE is 3.78.

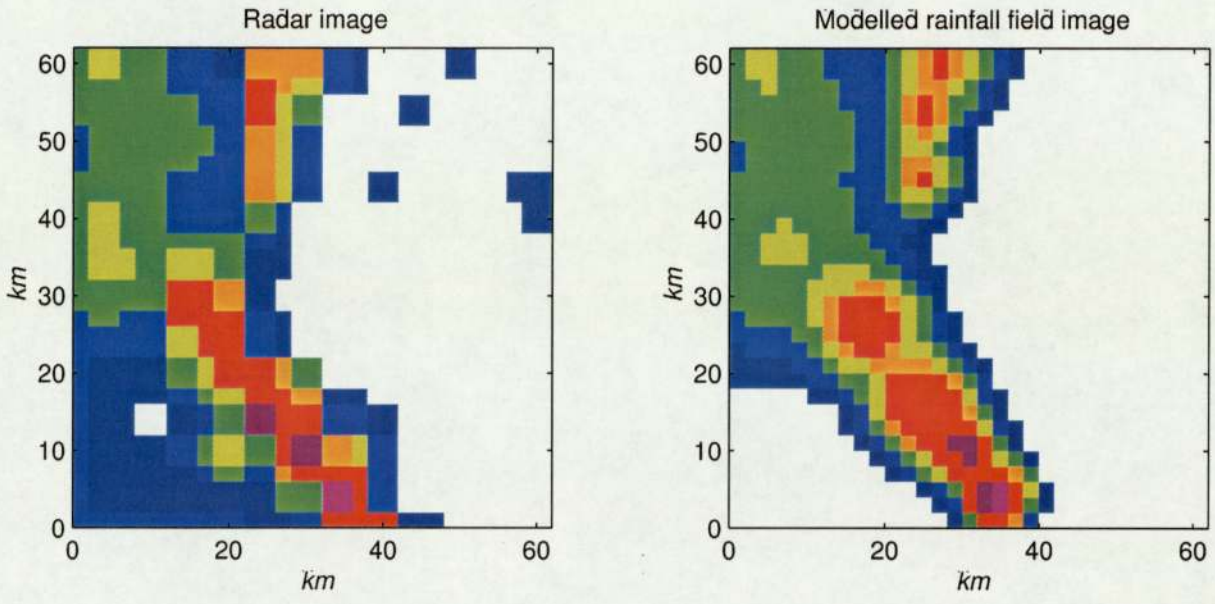


Figure 5.7: Image of the observation over 60km by 60km and of the rainfall field modelled from this observation. Realised with the second initialisation (data from measurement made the 30^{th} of October 2000 at 6:45 am).

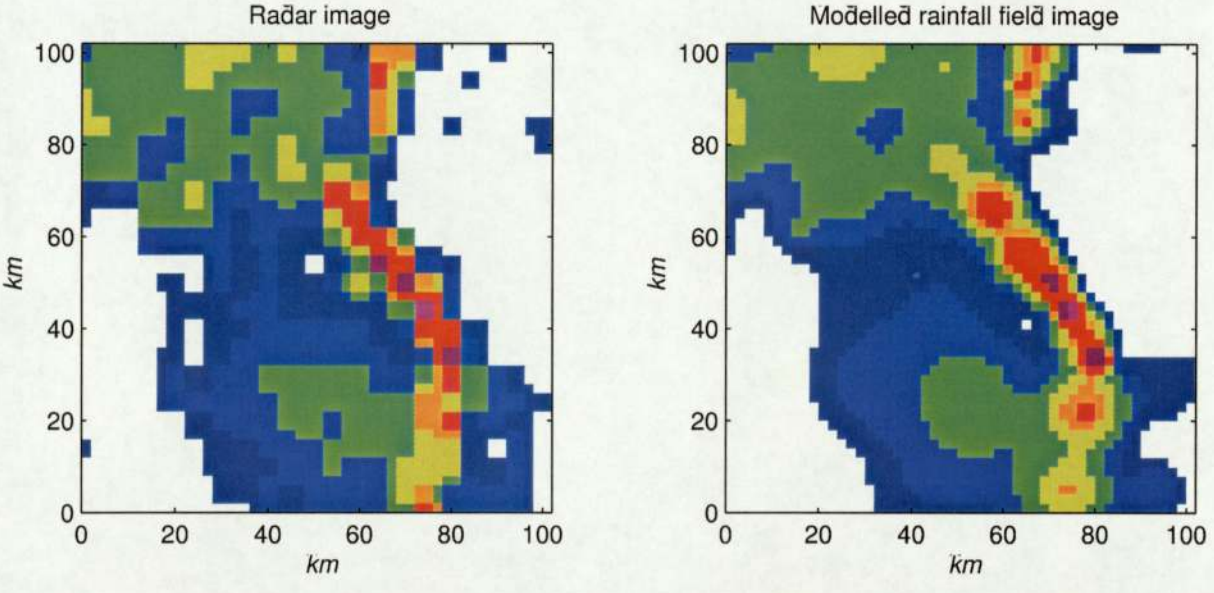


Figure 5.8: Image of the observation over 100km by 100km and of the rainfall field modelled from this observation. Realised with the second initialisation (Data from measurement made the 30^{th} of October 2000 at 6:45 am).

With the first initialisation, interesting results can be obtained, but it needs a lot of centers, which implies a very long optimisation. So as the time required to run the model is crucial in nowcasting, and because the second initialisation gives also accurate enough results, the second initialisation has been chosen as the starting process of the training of our RBF model on the observation I. Moreover, because the number of cells has to be hand set, this initialisation is not usable in our case.

The rainfall model is now determined and the way to fit it to the data as well. The rainfall model is a linear combination of Gaussian functions, and is determined by three parameters: the centers, the widths, and the height.

Now, we have modelled all our state space variables, and the next and last step in the construction of our model is the development of the state evolution and the update step in our filter introduced in Chapter 3. We assume that the advection is the only dynamics.

Chapter 6

Fully probabilistic model

In the previous chapter, we define our state space, the model, and the model used for the advection field (u, v) (Chapter 4) and for the rainfall field R (Chapter 5).

We are now going to present the development of the model introduced in Chapter 3.

This model is a statistical model and enables probabilistic nowcast of precipitation. In this chapter u will denote the advection vector which was denoted (u, v) until now.

In this chapter, we will first briefly cope with the model introduced in Chapter 3, then we will deal with the process which leads to forecasting, finally we will give some results on simulated and real data.

6.1 Model

6.1.1 Framework

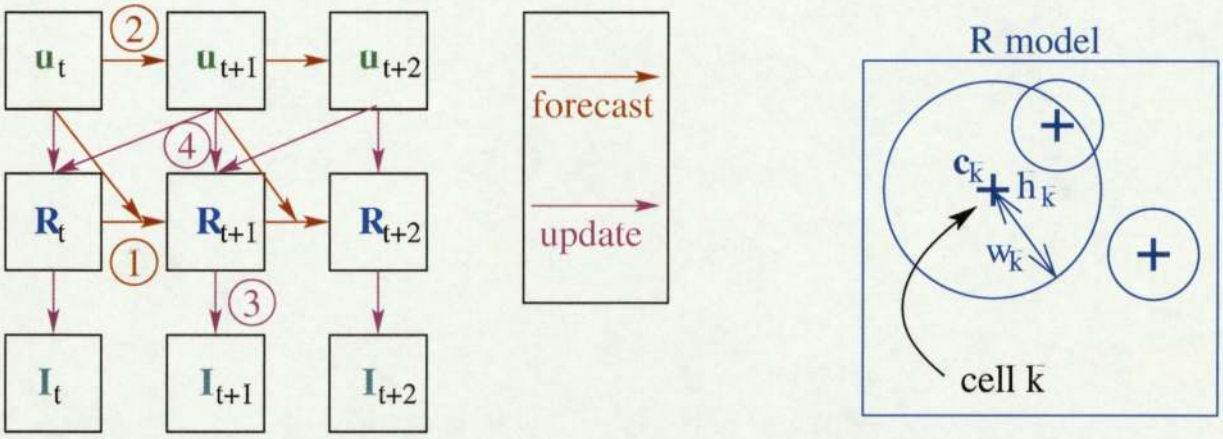


Figure 6.1: Advanced graphical model. This model is hierchichal. The numbers 1, 2, 3, 4 give the order of the operation to be made. In the graphical model, arrows in red represent the forecast, and the one in yellow the update. On the right is a scheme of the model used in this model for R. Each cell is determined by a center, a height and a width.

The model is a state space model with the state space R, u and observations I. The term G is omitted because it is encompassed in the model error. R and u are continuous function in time and space.

To ensure correct inference of u, a hierarchical structure is imposed as shown in Figure 6.1. We need to evolve and to update the rainfall field first because these update and evolution are based on the advection equation and so we need the current value of the advection field to compute these values.Thus conditional on R and I becomes independent over time and is found using two R fields. The model functions in a similar way to a Kalman Filter (see Section 3.3.1) but the updates are non-linear as the the relation between R and I is not linear.

6.1.2 Priors

As we are using a probabilistic Bayesian framework, we have defined prior distribution over all the state variables. The prior over the advection model has been introduced in Chapter 4 and the precipitation one in Chapter 5. The Table (6.1) reminds all the prior defined :

State variable	Prior
c	Gaussian $N(0, \Sigma_c)$
w	log-Gaussian
h	log-Gaussian
u	Gaussian process $N(0, \Sigma_u)$

Table 6.1: Priors for state variables

Figures 6.2 and 6.3 represent samples from the priors over R and u:

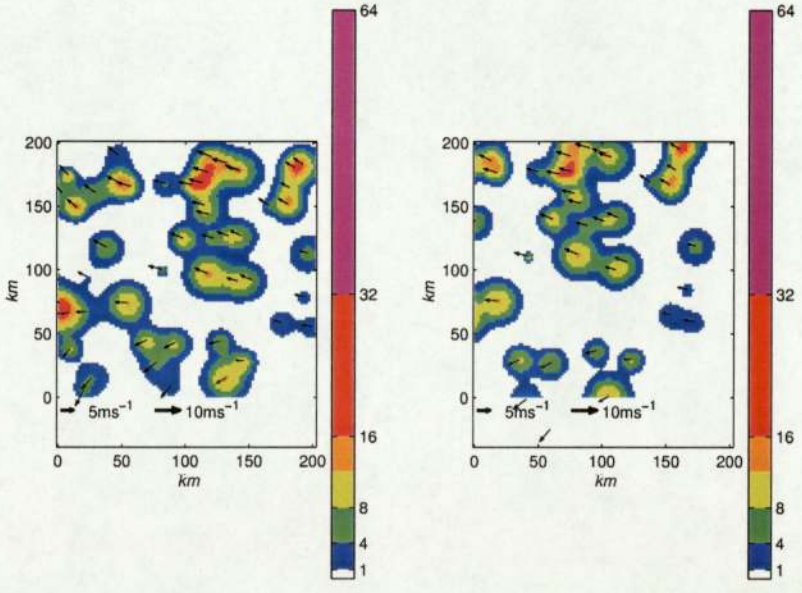


Figure 6.2: Two different samples from the prior over the rainfall field R.

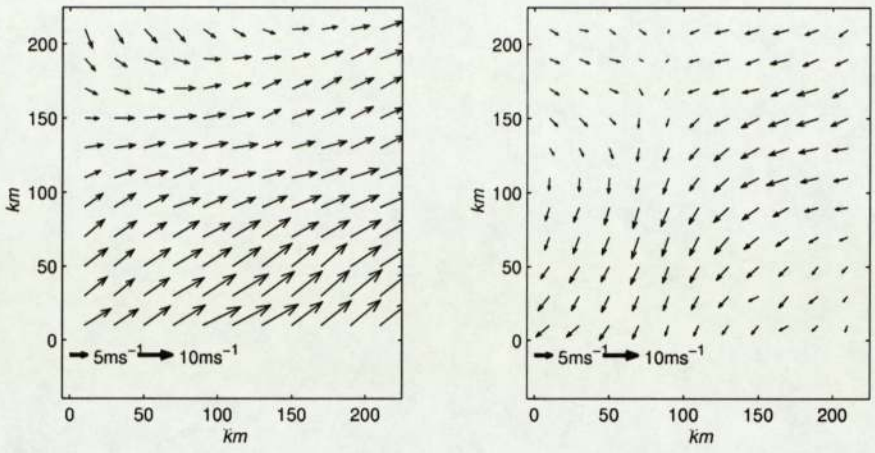


Figure 6.3: Two different samples from the prior over the advection vector u.

Once the priors are well defined over all our variables, we can deal with probabilistic tools.

6.2 Process

In this section, we are presenting the different steps in our filter (state evolution and update), and how the precipitation forecast is done. Before iterating in the filter, we need to initialise R and then u. The rainfall field is initialised by following the procedure initialisation, optimisation presented in Section 5.2 with the observation at time $t = 0$ and $t = 1$ as we need to solve the advection equation (3.2) to find the initial value of u. We are able to compute exactly the partial derivatives of R with respect to space. The partial derivatives with respect to time are computed using finite difference techniques (see Chapter 4). Then, we have a linear system over all space location (Section 4.1), and we are assuming u locally constant to solve this system with the same method than in Chapter 4.

6.2.1 State evolution

We need first to define the state evolution for R and u, which corresponds to step 1 and 2 in the Figure 6.1.

Step 1 : Since the model is hierarchical, we will first update R using the advection equation (see Equation 3.2). R is a function of the centers c, of the height h and of the widths w and is a linear superposition of RBFs We assume that the advection vector is locally constant. The forecast step R for is given by :

$$c_{t+1} = c_t + \delta t.u_t + \epsilon_c \tag{6.1}$$

where δt is the length of the forecast step and ϵ_c is the error in the forecast due to the simplifications of the model and that not all apparent cell motion is due to the

advection. As c and u are both Gaussian, this error is also assumed to be Gaussian. Since c, u and ϵ_c are both gaussian, this forecast is also a Gaussian with :

$$\mu_{c_{t+1}} = \mu_{c_t} + \delta t * \bar{u}_t \tag{6.2}$$

$$\Sigma_{c_{t+1}} = \Sigma_{c_t} + \delta t^2 * \Sigma_{u_t} + \Sigma\epsilon_c \tag{6.3}$$

where the $\bar{u}$ over-bar denotes the mean of the Gaussian.

So the dynamics are taken into account in the update of the center. For the update of the height and the widths of the cells it is easier as there is no dynamics. A small amount of log-Gaussian is added at each time step. We have not got any knowledge about the change of h and w, so this small amount stands for our growing uncertainty about these parameters.

Step 2 : This step deals with the update of the advection field. There is no explicit dynamics for this field, but the belief that it has much more longer time scale than the movement of the cells. The update chosen for u is :

$$u_{t+1} = u_t + \epsilon_u \tag{6.4}$$

where ϵ_u has the same characteristics as u but with a smaller variance to reflect that u changes slowly. This update of u is a sum of two Gaussians, so it is still a Gaussian with :

$$\mu_{u_{t+1}} = \mu_{u_t} \tag{6.5}$$

$$\Sigma_{u_{t+1}} = \Sigma_{u_t} + \Sigma\epsilon_u \tag{6.6}$$

The definition of this system noise reflects our uncertainty in our predictions. Now we have defined a method by which the states can be updated. Besides, priors have also been defined, so the model can be run in a generative way. We can sample from the prior distribution and then use the state updates to generate example data. Using this model in this way is a good way to check for a sensible model specification.

On the next figure (6.4), the model is run in a generative way:

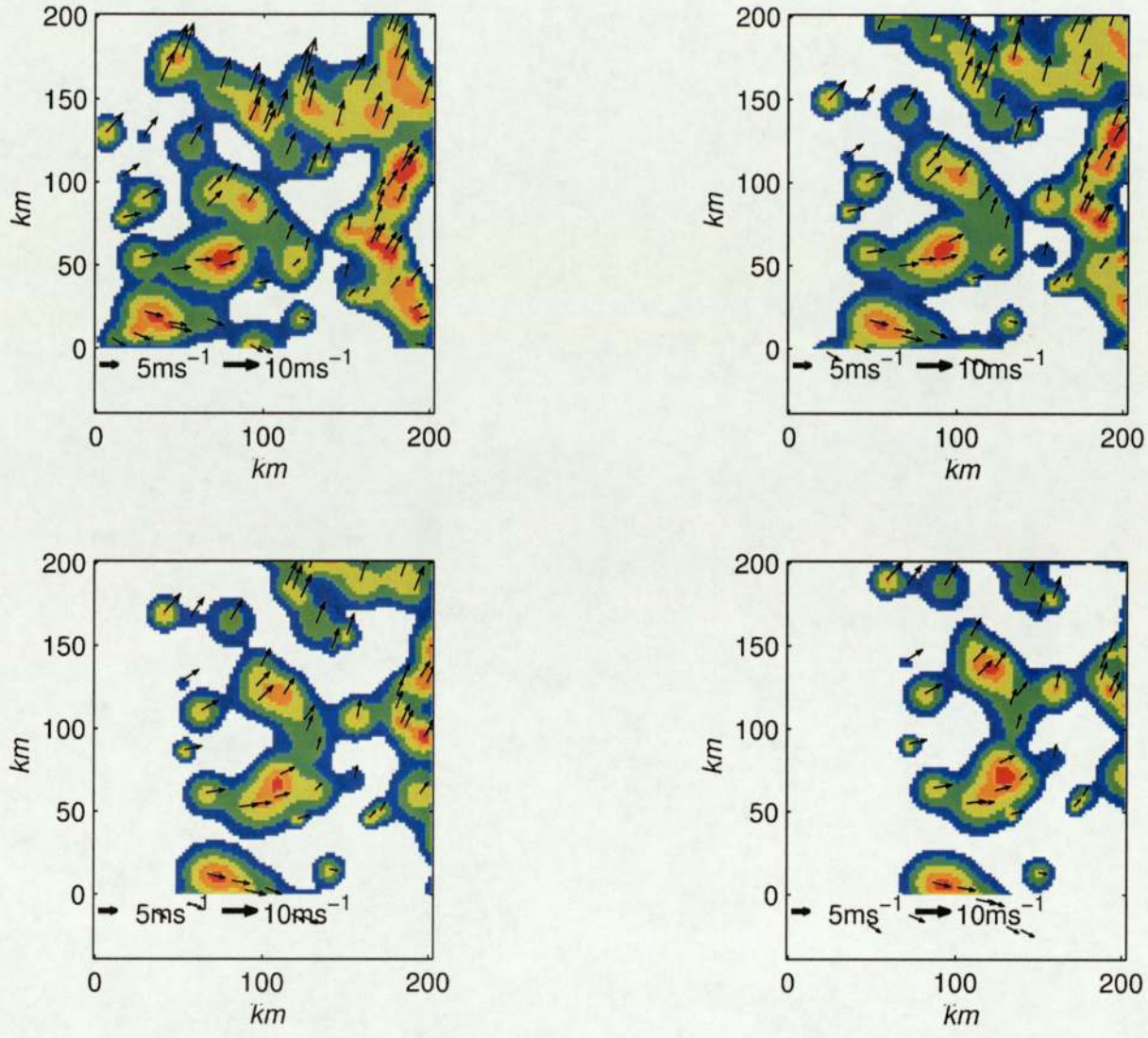


Figure 6.4: Model run in generative way. The time step between each image is 1 hour. These figures must be read from the top left to the right.

6.2.2 Data assimilation

We have to condition the model on the observations to make the model useful. In Numerical Weather Prediction (NWP), this is called data assimilation. Using Kalman Filter (Chapter 3) vocabulary, we are going to update the state given observations. First we update R and then u. These steps correspond to the step 3 and 4 on the Figure 6.1.

Step 3 :

$\hat{R}_{t+1}$ must be updated first. But this update is not trivial at all because the relation between I and R is not linear.

Using Bayes' rule, we can write :

$$p(c_{t+1}, w_{t+1}, h_{t+1} \mid \mathcal{I}_{t+1}) = \frac{p(\mathcal{I}_{t+1} \mid c_{t+1}, w_{t+1}, h_{t+1}) p(c_{t+1}, w_{t+1}, h_{t+1})}{p(\mathcal{I}_{t+1})} \tag{6.7}$$

The prior distributon for the parameter (c, w, h) is written:

$$p(c_{t+1}, w_{t+1}, h_{t+1}) = p(\hat{c}_{t+1}) p(\hat{w}_{t+1}) p(\hat{h}_{t+1}) \tag{6.8}$$

This writing of the prior comes from the step 1 in the previous section and c,w and h are uncorrelated. The likelihood term $p(\mathcal{I}_{t+1} \mid c_{t+1}, w_{t+1}, h_{t+1})$ (6.7) is defined by the errors on the observations and $p(\mathcal{I}_{t+1})$ is constant and unknown.

This step which corresponds to have an estimate of the posterior distribution has been developed in two ways. First, sampling using Markov Chain Monte-Carlo was used. But sampling, although giving accurate results, needs again a lot of computationnal power and it is quite long, so it is not appropriate for nowcasting. Then, we used a Laplace approximation about the MAP probability solution. This method is not as good as sampling but it is so much quicker. Actually, we are using the Hessian at the most probable parameter values to give the inverse of the covariance of the approximating Gaussian posterior distribution for the parameters. Because when the posterior is a Gaussian, then we have the following property : $\nabla^2 E = \Sigma^{-1}$ where $\nabla^2 E$ is the Laplacian of the error function at the MAP (using Taylor's series).

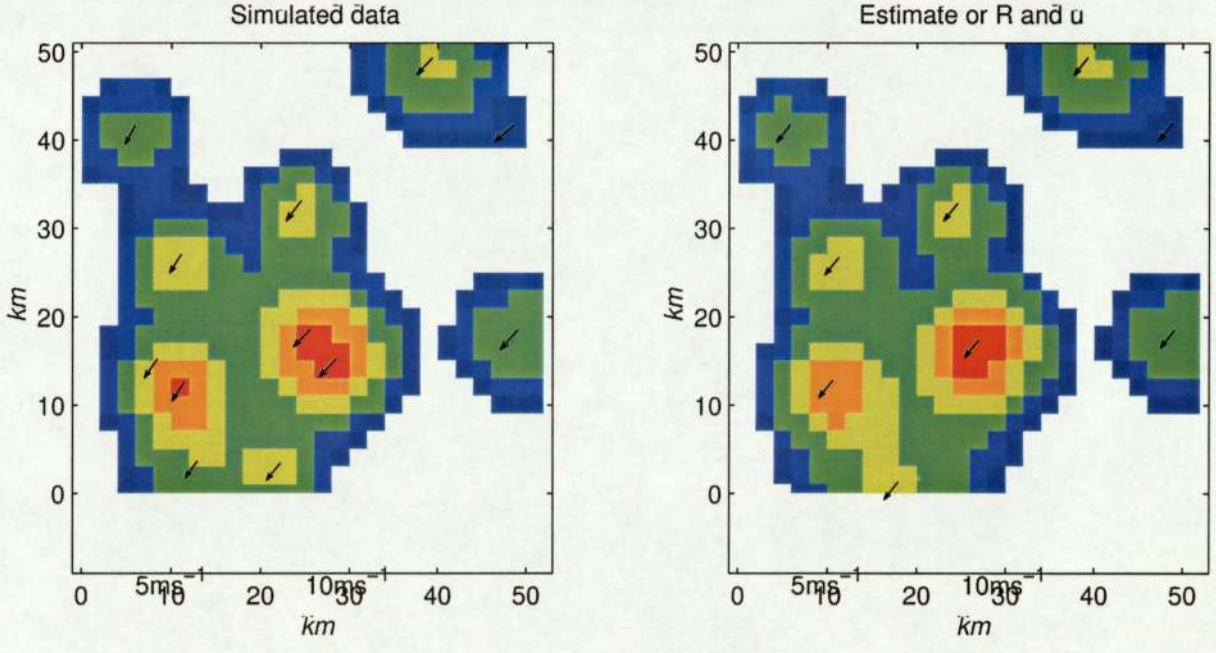


Figure 6.5: Left plot, simulated data. Right plot, the estimate of R and u.

Step 4 : Now u has to be assimilated. This is simpler for the advection field as the complex part is in the model for R. However, the model is hierarchical and we have to integrate over our uncertainty in R.

$$p(u_{t+1} \mid \mathcal{I}_{t+1}, \mathcal{I}_t) = \int \int p(c_{t+1} \mid c_t, u_{t+1}) dc_t p(\hat{u}_{t+1}) p(c_{t+1} \mid \mathcal{I}_{t+1}) dc_{t+1} \tag{6.9}$$

Here again the approximation that u is locally constant is made.

The updated u is a Gaussian. The mean and covariance are computable and we find (based on equation (6.9)) :

$$\mu_{u_{t+1}} = \Sigma_{u_{t+1}}[\Sigma^{-1}_{\hat{u}_{t+1}}\bar{u}_t + \delta t[\Sigma c_t + \Sigma c_{t+1} + \Sigma\epsilon_c]^{-1}(\bar{c}_{t+1} - \bar{c}_t)] \tag{6.10}$$

$$\Sigma^{-1}_{u_{t+1}} = \Sigma^{-1}_{\hat{u}_{t+1}} + \delta t^2[\Sigma_{c_t} + \Sigma_{c_{t+1}} + \Sigma_{\epsilon_c}]^{-1} \tag{6.11}$$

The resulting advection vector is a mix of the advection vectors inferred at the previous time step and those computed from the relative movements of the cells. This mix is actually a linear combination where the weight assigned to each information source source depends on our beliefs about their relative errors.

6.2.3 Forecasting

All the non-linearity is in the state update, when you condition the model on the observations. Given the approximations, the forecast step is linear and fast. After a few iterations in the filter, we can start forecasting rainfall. This step consists only of propagating the distribution of the rainfall and of the advection field, in which we have incorporated our incertainty. Actually, the forecast step repeats the state evolution of the system using the same propagation of the distribution as it was done in step 1 and 2 (described in this Chapter Section 6.2.1). The difference between the forecasting and the state evolution is in the definition of the error covariances definition. On the forecast side, the noise added is lower than in the state evolution. These parameters are set using expert judgement, and so are specified as deterministic variables and are hand tuned.

This forecast is a probabilistic one, so we have a forecast distribution, and not only a prediction. That means we have a forecast mean and covariance for the rainfall and the advection field. Sampling can produce realisations from this distribution.

The specification of the forecast error covariances is especially hard, and for the moment these values are hand tuned. This model could be improved by transforming these hyper-parameters into random variables and then estimating their posterior distribution given a large set of data. The errors on w and h are supposed to be indepedent. That means that cells close to each other do not have any link except that they have similar advection. The errors on the centers are uncorrelated as well. The covariance matrix $\Sigma\epsilon_c$ is diagonal given the errors in the advection field.

6.3 Test

6.3.1 Simulated data

Before testing it on real data, the model was run on simulated data. First, it is useful to debug when coding, then once the development is finished, it enables us to have an idea of the accuracy of the model. In the case of simulated data, the true parameter values

are known, so we can compare our results with the true parameters. Tests on simulated data show that the model can retrieve well the rainfall as well as the advection field in the filter (Figure 6.6).

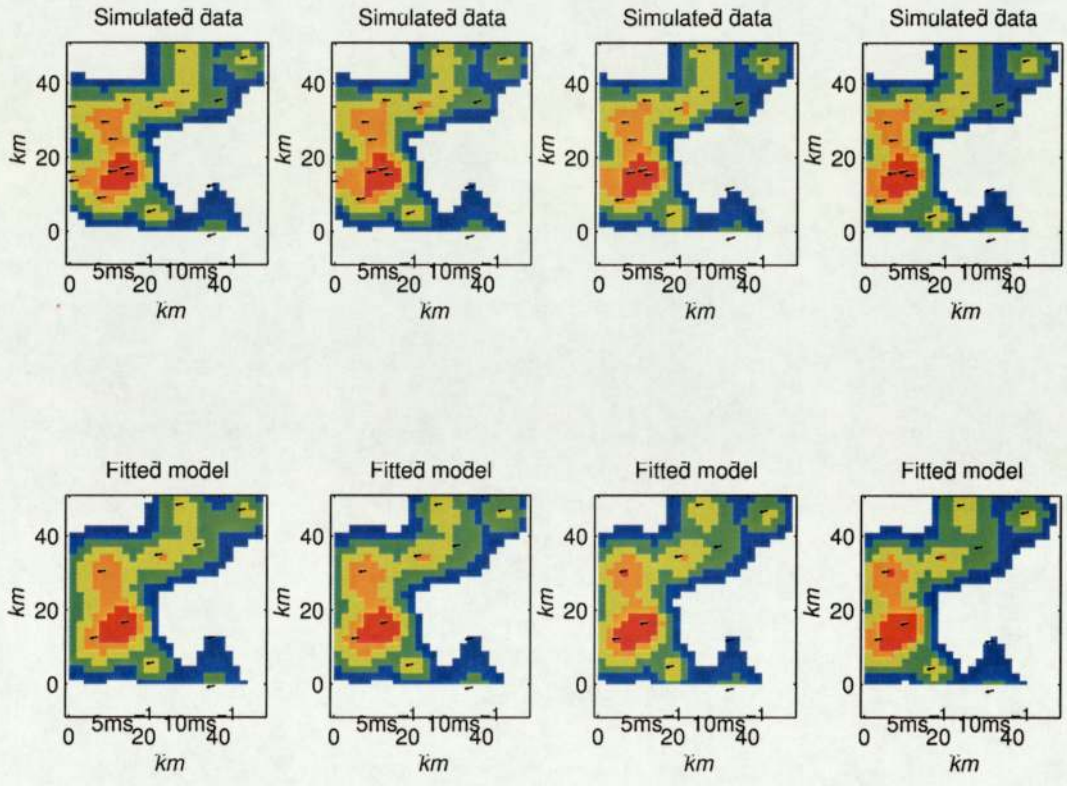


Figure 6.6: Simulated data on the top and fitted model on the bottom. This is the result obtained at each iteration of the filter.

This model provides a probabilistic nowcast, which means that we don't obtain only one forecast rainfall matrix, but a forecast distribution.

Figure (6.7) shows the mean prediction of a one hour forecast of the distribution, whereas Figure (6.8) shows 6 realisations from the forecast rainfall distribution.

Test on simulated data are promising, and the model is able to estimate the rainfall field and the adevction vector in the filter. To conclude about the forecasting part, we need to test it on real data in order to compare the mean forecast and some realisations from the distribution to the true rainfall field.

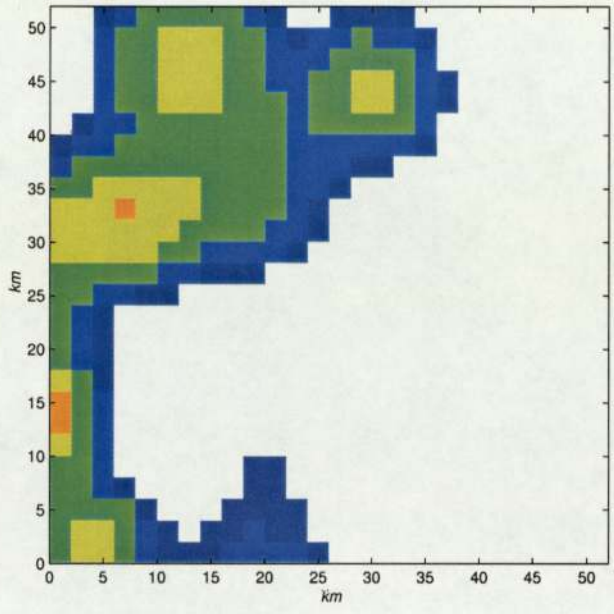


Figure 6.7: A mean 1 hour forecast from the forecast rainfall distribution.

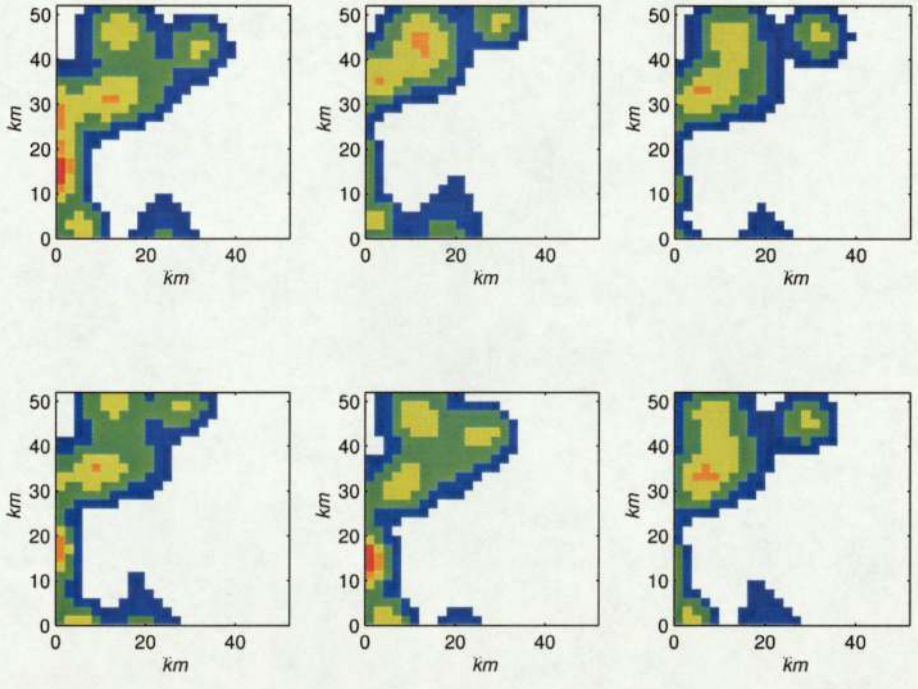


Figure 6.8: 6 random realisations from the forecast rainfall distribution.

6.3.2 Real data

Results on simulated data are promising, but testing a model on real data is often not trivial. Applying the model to real data is more challenging, but the goal of a precipitation forecasting model is to be operational to be run on real data. The figure 6.9 shows that the model can retrieve quite accuratly the rainfall and the advection field with real data. The result is not as accurate as with real data, but in the filter, the model fits well the data.

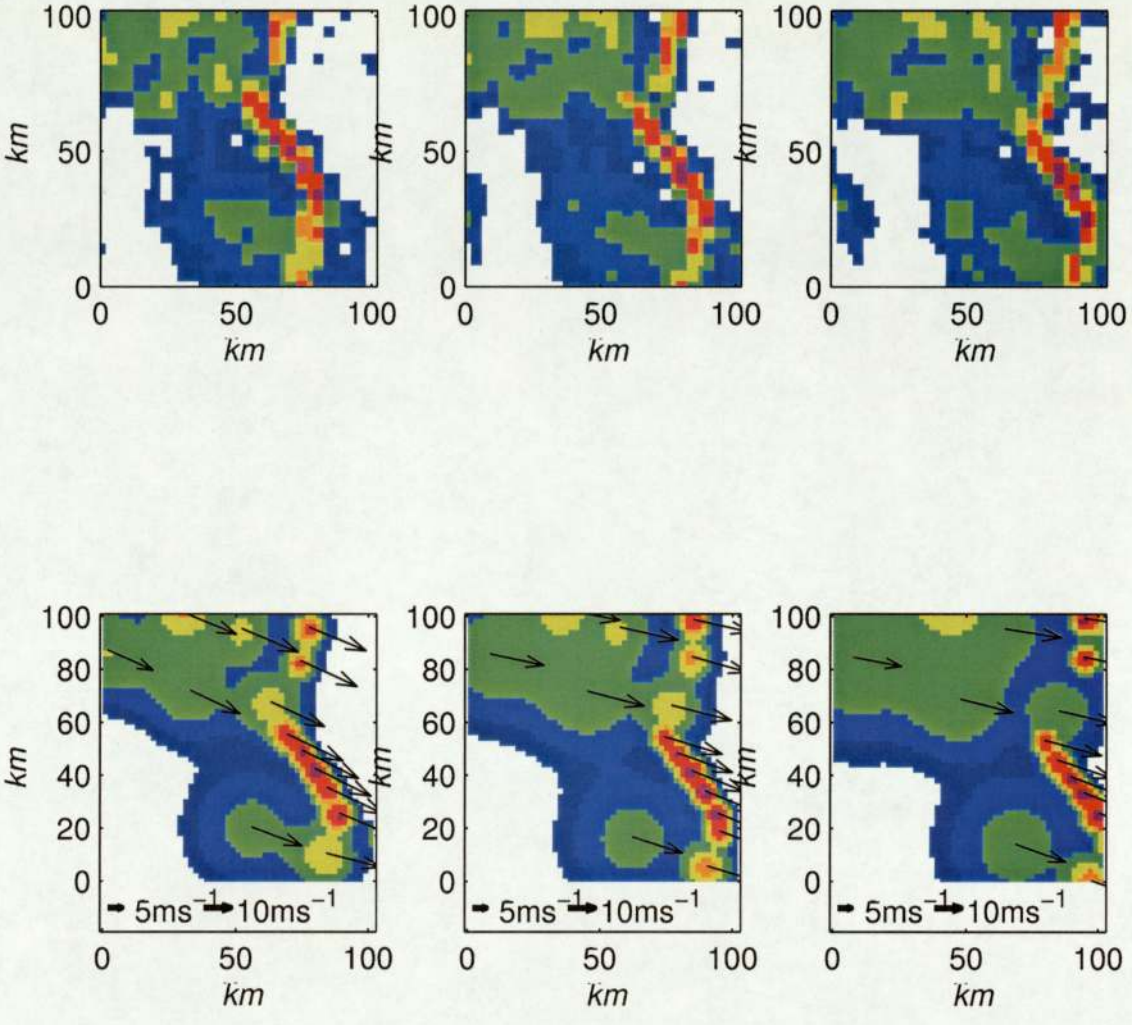


Figure 6.9: Radar images on the top and fitted model on the bottom. Data from measurements made on the 30^{th} of October 2000.

Figure 6.10 shows a one hour mean forecast and the corresponding real radar image. The result is a complete over estimates of the precipitation. This over-estimates is in part accounted by the absence of model for the growth/decay term.

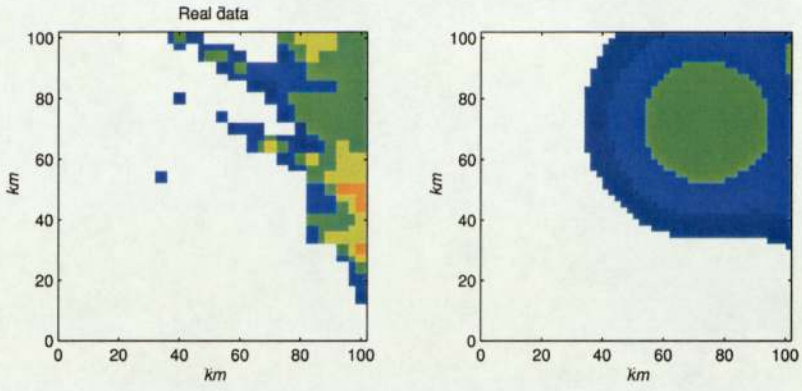


Figure 6.10: Radar image on the left and a one hour mean forecast on the right.

The figure 6.11 shows 6 realisations from the forecast distribution associated to the mean forecast in the figure 6.10: The definition of the forecast error covariances on

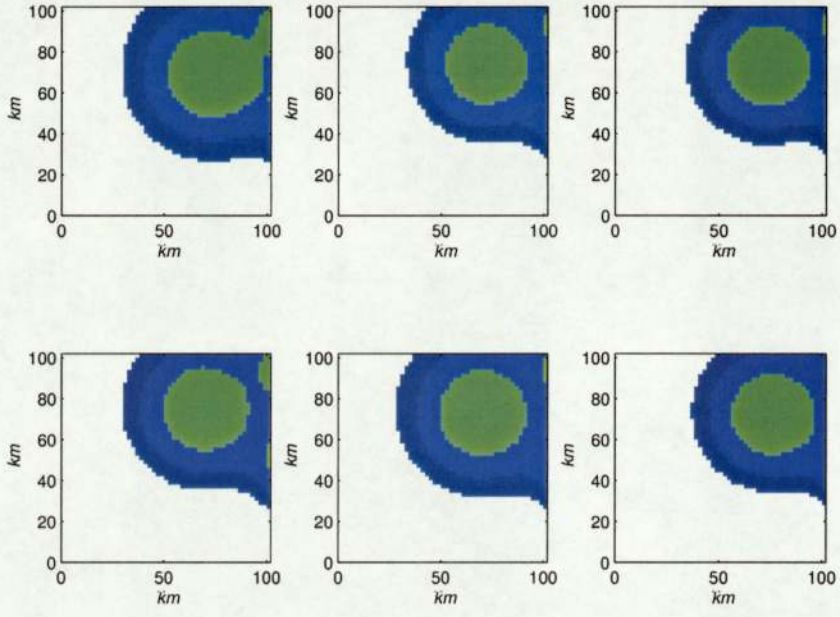


Figure 6.11: 6 random realisations from the forecast rainfall distribution.

the rainfall and advection field needs a more data driven approach to be specified. The growth/decay term G encompass the model error, thus it lacks of dynamics. On this point, a lot of improvements on this QPF model can still be done.

This model is a fully probabilistic one for precipitation nowcasting using only radar data, which produces probabilistic predictions. It provides a mean and covariance for the rainfall field R and the advection vector u. This model incorporates uncertainty about our variables in the observations and the model assumptions.

Chapter 7

Conclusions

The aim of this project was to create, and develop a fully probabilistic model for precipitation nowcasting using radar data only. The main reason for using a probabilistic framework is that at scales we can hope to represent there is an inherent stochastic nature to the evolution of the precipitation field. To develop this model, we used statistical theory and a Bayesian framework.

As preliminary work, we studied if a space reduction of the dataset provided was feasible. Work on the reduction space has not been incorporated in the development of this model, but some promising results have been obtained. The size of the data set was a constant problem in this project as we wanted to have an operational system.

After this work of space reduction, we have constructed the basis of the quantitative forecast model. We made the assumption that advection is the only dynamics, and we decided to incorporate the model into a Bayesian framework. The model introduced in this chapter is very general and is the basis of all the further developments. The growth/decay term G due to its complexity was included in the model error in practical development.

Once the theoretical model was defined, a first attempt to create a model which is not probabilistic was done. The results are not accurate, but this attempt gave some information about the problems. Especially, the instability of the finite differences makes us look for a rainfall model whose space partial derivatives are computable as finite difference techniques give inaccurate results for huge advection vector. Although this first model was not completely probabilistic, the update of the advection vector was treated in a Bayesian framework, and we had to define our prior over this state variables. We decided to represent this field by a Gaussian Process, as we wanted to be able to control the vorticity and the divergence of this one.

After assimilation of the structure of the data in Chapter 2 and 4, considering the constraints posed in Chapter 3, we have modelled the rainfall by an RBF network. We obtained accurate and promising results with this model for the precipitation field. The training of this network on real data is fast enough for nowcasting, and the fitting model smoothes the rainfall. The dynamics of this field were again related to the advection equation (3.2). Then, as we had defined all our state space variables, we implemented our model using a filter working in the same way as a Kalman filter excepting the non linearity of the update space. This non linearity is due to the fact than the rainfall model is not linear. We incorporated our uncertainty in the model through an noise addition in our evolution state. The model developed provides a forecast mean and

covariance for the rainfall field and the advection field. By sampling, we can produce realisations from this forecast distribution. Results on simulated data show that the model can retrieve well the rainfall field and the advection vector as well. In the light of the results, the difference between simulated and real data can be observed once more. The model still needs some improvements to be able to do accurate predictions. But preliminary tests are promising, and we can really expect better results if some further work is undertaken.

First, it will be nice to see results with other real tests and to compare this model with other operational models. Then, it is really important to improve the representation of the growth/decay term G. This term is, in principle, a variable of the state space but due to its complexity has been included in the model error. To be more+ realistic, it should be, as the rainfall field R and the advection field u, a function of space and time. It could be represented by a stochastic process at each space location. Then, inserted in the model, it will need to cope with dynamics. One idea is to find some information about this term by trial and error. It would also be nice to have a more data driven approach in order to specify more carefully the forecast error covariances (rainfall and advection field). Now, these error covariances are set by expert judgement which means they are hand tuned deterministic variables. One improvement to this model would be to make these parameters random variables, and estimate their posterior distribution given a set of data. Finally, explicit dynamics for the advection vector should be found.

This project, which provides a probabilistic QPF model, can serve as a basis for further experiments.

Bibliography

[1] Rilling B. The radar bright band. Available from http://www.doc.mmu.ac.uk/aric/eae/french/french.html.

[2] C. G. Collier C. M. Haggett C. E. Pierce, P. J. Hardaker. Gandolf : a system for generating automated nowcasts of convective precipitation. *Meteorl. Appl. 7*, 2000.

[3] D. Rosenfeld D. Atlas and A. R. Jameson. Evolution of radar rainfall measurements. Available from http://www.atd.ucar.edu.

[4] R. T. Williams G. J. Haltiner. *Numerical prediction and dynamic meteorology, second edition.* Wiley, 1971.

[5] R. E. Huschke. *Glossary of Meteorology.* Ed. 1959.

[6] G. Saporta J. M. Bouroche. *L'analyse de donnees.* Presse Universitaire Francaise, 2000.

[7] FSU meteorological department. Kinematics of the wind field. Available from http://www.met.fsu.edu/Classes/Met3502/lec_13.pdf.

[8] I. T. Nabney. *Netlab Algorithms for Pattern Recognition.* Springer, 2002.

[9] Met Office. Nowcasting using optic flow equation. Not published, 2002.

[10] J. Buchdahl S. Mller. Encyclopedia of the atmospheric environment. Available from http://www.doc.mmu.ac.uk/aric/eae/french/french.html.

[11] K. T. Smith and G. L. Austin. Nowcasting precipitation - a proposal for a way forward. *Journal of Hydrology*, 2000.

[12] S. Smith. Finding optic flow. Available from http://www.fmrib.ox.ac.uk/ steve/review/review/node1.html.

[13] A. Webb. *Statistical Pattern Recognition.* Arnold, 1999.

[14] J. West. How ground clutter affects radar? Available from http://www.usatoday.com/weather/wclutter.htm.

Index