Improving a stochastic rainfall forecasting model

Rémi Barillec

Master of Science by Research

ASTON UNIVERSITY

September 2003

ASTON UNIVERSITY

Improving a stochastic rainfall forecasting model

Rémi Barillec

Master of Science by Research, 2003

Thesis Summary

Short-term rainfall forecasting is a major subject of research in the domain of weather prediction. The development of radar imaging since the 50's and recent advances in data analysis methods such as neural networks made it possible to design radar-based statistical frameworks dedicated to the modelling of rainfall processes. A model of that kind has been developped last year at Aston University's Neural Research Group. This thesis describes the problems inherent to this model and the modifications attempted to correct them.

Keywords: Rainfall forecasting, variational Bayes, stochastic modelism

Acknowledgements

There are many persons I would like to thank, for their support, their advice, their interest in this thesis.

First, as the person most involved in this project, I want to thank Dan Cornford, my supervisor, for being someone both serious and nice, who helped me all along this year, and whose scientific rigour matches his ever good temper. Dan, this acknowledgement is too small to suit my need, so let me just make it short: it has been a real pleasure to work with you.

I would also like to thank all the NCRG staff, here, for their excellent lectures, their interest, their will to share a bit more than just the work. I will not list all your names, but I must write Manfred Opper's, to whom I owe most of the analytic tricks in my computations. It has been really nice to spend a year amongst you.

Thanks to my Master's fellows for the good time we shared, Benoit Fabre, Vagelis Sagianos, Benoit Maroillez, Jody Miotke, Hiroko Moriokah, and particularly to Frederic Le, for having been a close friend all along the year, wise in advice and deep in personnality.

Special thanks to Chiara Letardi, for her complicated friendship and so many reasons that wouldn't suit in here.

And last, but not least, thanks to my parents, who made this year in Aston possible. I owe you once again one of the richest and nicest years of my life so far.

Thanks to all of you, guys, this year, probably the last of the "young era", will let a bunch of good memories for ever, up there in the head.

Contents

1 Introduction **8**

1.1 Foreword . . . 8

1.1.1 General details . . . 8

1.2 Why this project? . . . 8

1.3 Organisation of the work . . . 9

1.3.1 Step 1: Localisation in the field . . . 9

1.3.2 Step 2: Learn the model . . . 9

1.3.3 Step 3: Apply the knowledge . . . 9

2 Statistical short-term rainfall forecasting **10**

2.1 Overview . . . 10

2.1.1 Numerical Weather Prediction . . . 10

2.1.2 Radar-based models . . . 11

2.1.3 Neural networks . . . 11

2.2 Radar images . . . 12

3 Bayesian theory **14**

3.1 Bayesian theory . . . 14

3.1.1 A statistical framework . . . 14

3.1.2 Bayes' rule . . . 15

3.1.3 Bayesian Learning: an example . . . 16

3.2 Radial Basis Functions (RBF) networks . . . 17

3.3 The state-space model (Kalman filter) . . . 18

3.3.1 Introduction . . . 18

3.3.2 Evolution step . . . 20

3.3.3 Assimilation step . . . 20

3.3.4 Extension . . . 21

4 The model **23**

4.1 Overview . . . 23

4.1.1 The rainfall rate R . . . 23

4.1.2 The advection field $\boldsymbol{u}$ . . . 24

4.1.3 The advection equation . . . 25

4.1.4 Priors . . . 26

4.2 Initialisation . . . 26

4.2.1 Initialisation of R . . . 27

4.2.2 Initialisation of u . . . 27

4.3 Dynamics . . . 28

4.3.1 Evolution of R . . . 28
4.3.2 Evolution of $\boldsymbol{u}$ . . . 29
4.3.3 Assimilation of R . . . 29
4.3.4 Assimilation of u . . . 30
4.3.5 Forecast . . . 30
4.4 Issues . . . 31

5 Speeding up the model with Variational Bayes **32**
5.1 Analysis of the speed issue . . . 32
5.2 Variational Bayes . . . 33
5.2.1 The Kullback-Leibler distance . . . 33
5.2.2 Distributions chosen . . . 34
5.2.3 Negative log-likelihood part (KL_1) . . . 35
5.2.4 Negative log-prior part (KL_2) . . . 39
5.2.5 Entropy of q (KL_3) . . . 41
5.2.6 Result . . . 42

6 The new model **43**
6.1 Initialisation . . . 43
6.1.1 Smoothing . . . 43
6.1.2 Local fitting . . . 43
6.2 Dynamics of the model . . . 44
6.3 Tests . . . 44
6.3.1 Simulated data . . . 44
6.3.2 Fitting a single cell . . . 45
6.3.3 Fitting two cells . . . 47
6.3.4 Fitting a large number of cells . . . 48
6.3.5 Further testing . . . 48
6.4 Conclusions . . . 49

7 Conclusion **50**
7.1 Summary of the work done . . . 50
7.2 Results . . . 50
7.3 Further work . . . 50
7.4 Afterword . . . 51

A Test parameter estimations **52**

List of Tables

5.1 Time involved by step 32

6.1 Single test fitting. Variables initial and final states. 46
6.2 Two cells test fitting. Variables initial and final states. 47

List of Figures

2.1 2D rainfall radar image 13
2.2 3D rainfall radar image 13

3.1 Probability distributions with different uncertainties 15
3.2 Bayesian learning: posterior distributions 17
3.3 An RBF network 18
3.4 Choice of RBF structure 19
3.5 The 2 steps of a Kalman filter 19

4.1 3D representation of R 24
4.2 2D representation of R 24
4.3 A 2D Gaussian process 25
4.4 Evolution of the rainfall rate through time 25
4.5 Initial radar image 27
4.6 Initialised rainfall rate 27
4.7 Dynamics of the model 28

5.1 Laplace approximation (good) 33
5.2 Laplace approximation (bad) 33

6.1 Smoothing the data 44
6.2 Simulated radar images 45
6.3 Single cell fitting 45
6.4 Evolution of the distributions 46
6.5 Two cells fitting 47
6.6 Fifteen cells fitting 48

A.1 Single cell fitting. Evolution of the parameters. 53
A.2 Two cells fitting. Evolution of the parameters (first cell). 54
A.3 Two cells fitting. Evolution of the parameters (second cell). 55

Chapter 1

Introduction

1.1 Foreword

1.1.1 General details

This thesis is the report of research work carried out from January to September 2003 at Aston University's Neural Computing Research Group, as part of a Master of Science by Research. The objective of this report is to give account of the work done, as well as to provide documentation about a project which offers further research opportunities.

D. Cornford and E. Batail developed last year (2002) a statistical model for short-term rainfall forecasting [1] [5], based on a Bayesian state-space framework similar to the Kalman filter (which will be introduced later). This model, though providing good results, lacked speed, which made it unsuitable for operational use. Several other minor features were to be improved, in order to make it more robust. The objective of the work carried out was to solve the speed issue and improve the model.

1.2 Why this project?

There are several reasons why I chose this project. Listing them all would make this introduction too personal for a scientific work, so I will only focus on the most important: the nature of the project.

Spending nine months on a project requires that one should be interested in the work. How can one keep working seriously for so long a time if one does not like what one does? There has to be a source of motivation, coming from interest, from which one can draw the impetus to work (especially in research, where you have no working constraints but the ones you give yourself). That is why I choose carefully a project which agreed with my interests.

Weather has always had a deep impact on the mind of man. We no longer worship the sun nor pray for the rain that waters vegetation, of course, but we are still charmed by a sunny spring morning or worried at the sight of heavy black clouds on the horizon. Formerly the field of sorcerers and psychics, weather forecast has evolved a lot since it became a subject of scientific interest. What was yesterday mysterious is almost understood today. What drove me to this project is not the magic of "telling the future", but rather the curiosity, the wonder, the will to understand the mechanisms behind the evening TV weather forecasts.

I felt this project would suit me quite well since it involved theoretical statistical knowledge, but wasn't yet a complete abstraction, and bore very practical, measurable applications. I thought it as a way to learn much, and I was right.

1.3 Organisation of the work

1.3.1 Step 1: Localisation in the field

The first step in a research project is to answer the questions one cannot help asking. What research field is this project related to? How does this project include itself in the ongoing developments in the field? Is it relevant, useful? What has already been done and what hasn't? What are the results? It is necessary to answer all those questions before starting the work, in order to get familiar with the domain we are to work in, learn the basic concepts, have a clearer view of where we start from and what we're heading to. This step basically included lots of reading from different sources, from general introductions to weather forecasting to research papers on similar projects, so as to get an overview of the research state in the domain and be able to understand what we are doing and why. Attending conferences such as the NCAF meeting in January 2003 helped me to learn more about neural networks applications (and even this particular project [6]), as well as attending a meeting at the Met. Office this spring, where different forecasting methods were presented and helped broaden my overview of research in this field. The results from this first approach are summarised in the following chapter so as to give the reader a rough description of where the project situates itself with respect to other works.

1.3.2 Step 2: Learn the model

Diving inside a project developed by somebody else is never an easy job. But it is a job from which one learns much in terms of the basic rules to apply when being part of a project in which one is involved for a time shorter than the project's life. One must not just think about his work, but keep in mind that others after him will have to use it, and thus try to build things on top of which it will be as easy as possible to add the following bricks.

Accurate documentation about one's work is vital. Some documentation existed about the model's theory (E. Batail's thesis [1] and D. Cornford's presentation draft [6]), though I found it not detailed enough for me to learn the model from scratch. I spent much time trying to understand things that were not particularly difficult to grasp, but were often too lightly explained or using pieces of knowledge not explicitly mentioned, so I had to do much research about the methods used. In a way, this is a good thing, for it helped me achieve strong knowledge about techniques that, had they been explained, I would perhaps have not taken the time to understand so deeply.

The Matlab code associated to the theory was much less documented (actually, no written documentation existed), and it has taken me a pretty long time to go through the code and understand it directly from the source files. Being myself from the computer science field, I experienced how difficult it is to understand somebody else's work when it is not sufficiently documented. This is why chapter 4 gives a detailed presentation of the model, in the hope this will fill the gaps in the existing documentation, and provide the reader with a clear understanding of the model.

1.3.3 Step 3: Apply the knowledge

Once familiar with the model, I was able to start thinking about its issues and some ways to solve them. I regret that it took me so long a time to master the model, for I have run a bit short on time on this part. The main issues are listed at the end of chapter 4 and discussed more in detail, as well as the solutions suggested, in chapter 5. Chapter 3 gives an introduction the theoretical concepts and methods used in the following chapters.

Chapter 2

Statistical short-term rainfall forecasting

This chapter gives a brief overview of what statistical short-term rainfall forecasting is about, as well as its constraints.

2.1 Overview

There has become a real need for reliable real-time forecasts, mainly as a means to forecast flash-floods and storms, which are amongst the most common and dangerous of natural hazards. To predict such phenomenon, high resolution (a few kilometres) and fast forecasts are required. The usual numerical weather prediction (NWP) systems cannot yet meet such requirements, so alternative approaches have to be found. Forecast models with simplified assumptions as to how rainfall evolves have been developed, amongst which are neural network based methods.

2.1.1 Numerical Weather Prediction

The classical NWP approach to rainfall forecasting consists of using the sets of multi-dimensional differential equations which govern the physics of rainfall. Such models require high computational resources, and are currently dedicated to large scale ($>$ 100 km grid resolution) long-term forecasts. Using those complicated models at the required scales for rainfall monitoring would currently be operationally infeasible. Meso-scale models were developed as a way to adapt NWP to lower scales, but nothing like the small scale required for accurate rainfall measurements.

Pedder *et al.* [15] have underlined the limitations of NWP models at small-scale resolution, and suggested a mesoscale model aimed at modelling the atmospheric moisture. Their model provides information that could be used in smaller-scale embedded models (typically real-time radar-based short-scale models), in order to increase their performance. In the other direction, small scale systems could also help improving larger scale models by providing more accurate local information, thus leading to a combination of different systems at different scales for optimal forecasts.

Alternative approaches to NWP were looked for, in order to cope with smaller scale problems, mostly by simplifying the dynamics of the atmosphere. Those methods were thought not to rely heavily on atmospheric physics, rather, they need more observations of the process to remain close to

reality. Such observations were formerly provided by ground rain-gauges measurements, until radar technology brought a more convenient way to get regular and fast observations.

2.1.2 Radar-based models

The benefits from the use of radar technology in weather measurements have been understood since the very beginning of radars in the 1940's, but the United Kingdom had to wait until the early 80's for the development of a national network covering England, Wales and Ireland [4]. Those radars could give high resolution (2 to 5 km grid size) images at rates down to 1 image every 5 or 15 minutes (radar images are introduced in the following section). This gave birth to a new generation of forecast models adapted to short-term rainfall prediction, first by combining weather radar data and mesoscale NWP models, then by using fully radar-based models.

Wheater *et al.* [16] review different stochastic methods to model rainfall cells. In the simplest model, rainfall cells are created according to a spatial-temporal Poisson process as random radius, velocity and duration circular objects, with constant rainfall intensity over the disc thus defined. All the cell contributions sum up to give the total rainfall rate. The model evolves in space and time, using linear approximations (Taylor's expansion). Since their model is too computationally expensive to use a maximum likelihood optimisation, they suggest interesting methods for fitting the model to the data, namely a generalised method of moments (minimise a weighted-sum of suitably chosen parameters) and a spectral method, based on the data's Fourier coefficients which permit the joint probabilities of small data subsets to be treated as normal, thus enabling the use of techniques similar to likelihood estimation.

Mathurin *et al.* [13] suggest an advection model to track rainfall cells. Rainfall cells are modelled by moving ellipses, the objective being to be able to track the cells, given the fact that new cells can appear or disappear, and existing cells can merge or split. Radar data is used, and the tracking of cells from an image to the following is achieved by minimising a clustering error function (fit both images, advect the first, and merge/split/create/remove its cells according to the option giving the results the closest to the new image). Though this model is not aimed at forecasting, it gives a clear description of an advection-based simple model.

Time-series techniques such as Kalman filter could be applied to sequences of radar images, giving data-driven models being able to keep only the most basic assumptions about rainfall physics. Different methods existed to cope with time-series, which gave different prediction models.

2.1.3 Neural networks

In parallel to radar technology, optimisation techniques improved, and in the 1990's, neural networks started to prove very efficient methods to tackle tricky non-linear problems. Their application to time-series gave good results, and soon they were applied to data-driven weather prediction. Many projects today are still focused on rainfall forecasting with neural networks.

Neural networks have multiple advantages. They are simpler to use than most of the other data analysis techniques, they converge very fastly, and they are stochastic. Their simplicity makes them easy to understand and adapt to specific problems, as well as giving them a robustness which makes them methods one can apply to lots of general regression and classification problems. In particular, they are very efficient at modelling complex non-linear mathematical processes. Their speed is another

obvious advantage in the domain of nowcasting, where models have to be fast to be useful. But their biggest advantage is certainly the fact that they are methods based on a statistical framework. As a matter of fact, they provide an efficient way to quantify the uncertainty inherent to the process, which is particularly useful in such an uncertain domain as weather forecasting.

Uncertainty lies in every step of a rainfall forecasting model. A perfect forecasting model would require perfect measurement and error free forecasting performance. Einfal and Denoeux [7] explain that in mathematical models adapted to natural processes, there are many factors of error: the transformation of information (relations between two physical variables), the impossibility of observing the process directly (rainfall's physics), the reliability of the data (errors in measurements, interpolations, even corrections), simplifications of the reality for working ease, all those factors bring their own amount of error to the forecast. Typically, radar data can be up to 50% corrupted. And as Grecu and Krajewski point out in a review of rainfall statistical forecasting procedures [14], even the dynamics of rainfall themselves are still not even clearly understood.

Thus, there is a large uncertainty in forecasting which it is very important to be able to measure in order to make relevant decisions when necessary (flood alert, for instance). This is a reason why a stochastic approach, with simplified assumptions about atmosphere's dynamics, appears as a relevant choice for short-term real-time forecasting, for it is able to quantify the model's uncertainty.

Multi-layer perceptrons have already been applied to short-term rainfall forecasts. French *et al.* [8] have developed a neural network based model for 1 hour rainfall forecasts using a multi-layer perceptron in a space-time framework, and shown that a neural network was capable of learning the complex relationships describing the space-time evolution of rainfall. They use a model similar to the one Wheater reviews (see 2.1.2). They emphasise the importance of the number of hidden nodes, that has to be chosen so as to enable sufficiently close approximation of the process modelled.

The idea of using a radial-basis functions (RBF) network to model the rainfall field is a step further in the effort to incorporate the power of neural networks into short-term rainfall forecasting.

2.2 Radar images

Radar images are obtained by measuring the reflectivity of a radar beam along a specific target of the atmosphere. The reflectivity, Z, is related to the rainfall rate, R, by equations that are beyond the scope of this thesis. This reflectivity is measured through space (by scanning over a given area) and time (by taking successive scans at a given frequency), thus producing a sequence of images. The inaccuracies in this process can occur at different steps [10], especially:

- change in rainfall intensity between the scanned area and the ground,
- errors inherent in the device itself (bad calibration of the radar),
- inaccuracies in the Z-R (radar-rainfall rate) relationship,
- noise added while the beam travels.

All these considerations make the final image uncertain, and one shouldn't assume the radar image gives a perfect representation of the rainfall as it could be measured on the ground.

Each image gives a spatial representation of R at a given time, as can be seen on figures 2.1 and 2.2. Figure 2.1 is a planar representation, where the rainfall rate is proportional to the brightness,

whereas Figure 2.2 gives a 3d representation, where the height is proportional to the intensity.

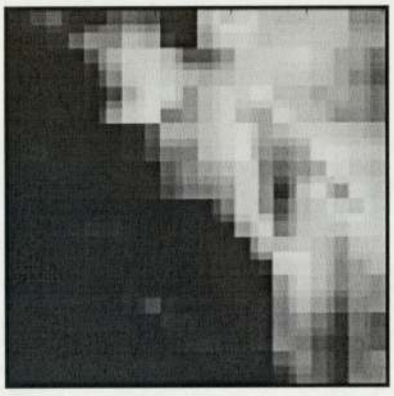


Figure 2.1: 2D representation of a radar image

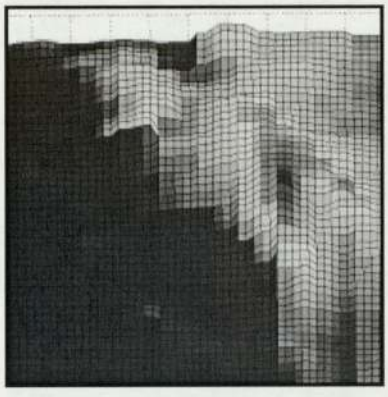


Figure 2.2: 3D representation of a radar image

The data (measurements) we use in the project are 200×200 km radar images at 2×2 km resolution from the NIMROD[1] radar imaging system (UK's Metoffice), issued at a rate of 1 image every 5 minutes. This gives us a sequence of images that can be used in a space-time framework, the space component corresponding to the area scanned for rainfall, the time coming from the sequential nature of the images series, which can be thought of as a discrete representation of a time-evolving process (each image being a "snap-shot" of the rainfall state at a given time).

We have briefly given an overview of the state of research in the field, to give the reader a better sight of how this project takes place in a chain of advances in weather prediction. Before discussing the model, we will now introduce the basics of theory necessary to understand it.

[1] NIMROD: Nowcasting and Initialisation for Modelling using Regional Observation Data

Chapter 3

Bayesian theory

In this section, we discuss the main methods used in our model and introduce the basics necessary to understand the discussion in the following sections. Readers may find this chapter a little too theoretical; they can then skip it and come back to those notions later, to fully understand the model.

3.1 Bayesian theory

3.1.1 A statistical framework

"In this world, nothing is certain but death and taxes." – Benjamin Franklin

Theories are exact and tractable representations for processes that are too complicated to be treated in their entire complexity. The main difference between a theory and the reality it models, is that starting from the same assumption, the theory will always provide the same result. But in the real world, it is simply impossible to carry out the same experiment twice, to make the same measurement twice, to obtain the same result twice. So many secondary processes interact with the main process we are interested in that it is impossible to imagine building a perfect system.

For example, in the domain of music, notes are assumed to be pure combination of sine waves, though we know there is no such thing in the real world. Imperfections lie everywhere, from the generation of the note (a finger never hits the piano keyboard twice in the same way) to its perception (a microphone never records the whole complexity of the acoustic process). Even if we imagine perfect devices for creating and observing our wave, it would immediately be corrupted from external emissions from close radiating objects, from reflections, distortions, etc. Yet, we are able to recognise when the same note is played twice on the piano, and to identify the note played to the note recorded. This comes from the fact that our ear is imperfect, and is able to extract the main information in what it hears. Building theoretical models is doing with our mind what our ear does with sound: focus on the important parts and filter the rest.

In order for our mind to be able to understand and act on the complex world around us, we have to focus on the main processes, neglecting external interactions then referred to as "noise", or, "randomness". This can be done because those interactions usually have an impact on the main process small enough not to change it in proportions which would make it unrecognisable.

Thus any model we build is valid to a certain extent only. It does not matter if the ear does not feel the difference between two similar notes, but it sometimes can be necessary to measure this difference. When forecasting a flash flood in some part of England, we need to know to what extent this forecast is valid, before asking people to evacuate the area. In such circumstances, the error

committed matters as much as the forecast.

Statistics is dedicated to such problems, where we are not only interested in the value of some random variable, but also in the uncertainty of this value's estimation. Let us assume we want to measure some device's temperature X which can take continuously distributed values x. Our measurement system isn't faultless, it cannot give us the true value of X. Instead, it provides us with an estimated value μ, which is only close to the exact value. This means that we assume the exact value of X lies somewhere around this estimate. It is interesting to know how certain our estimate is, since, for example, a small change in temperature could be critical to our system. The function p which associates to x the probability that X takes value x is called the probability distribution of X. Figure 3.1 shows two different distributions for X.

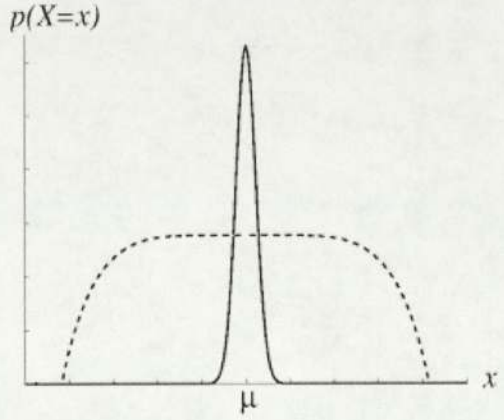


Figure 3.1: Probability distributions with different uncertainties

Those curves show the behaviour of the probability of X. The solid line illustrates a case where we are quite confident that $X = \mu$. On the contrary, the dashed line illustrates a case where there is a large uncertainty as to the value of X. Since both curves are centred around the most probable value μ, we can see that we need to know, in addition to this estimated value, how our uncertainty behaves, that is to say we need to know the shape of the probability distribution. In a critical system, an uncertainty such as the dashed line's would be hazardous, since there is a large chance for the true value to be far from the estimate. Hence the importance of using the variable's distribution, not just a single value estimate.

All the discussion that will follow takes place in a statistical framework, where all variables have a probability distribution (in fact, we will often forget the variable as such and only talk about it as a distribution).

3.1.2 Bayes' rule

Suppose we are interested in determining the probability distribution for a random variable θ. To do this, we have a dataset $\boldsymbol{D} = (d_1, ..., d_M)$ of random realisations of θ. Assuming we have set an initial distribution $P(\theta)$, called the *prior* distribution, according to our belief as to how X is distributed, Bayes' rule enables us to use the dataset to correct our initial distribution, as follows:

$$P(\theta|\boldsymbol{D}) = \frac{P(\boldsymbol{D}|\theta)}{P(\boldsymbol{D})} P(\theta)\,. \tag{3.1}$$

$P(\theta|D)$ is called the *posterior* distribution. The posterior distribution has incorporated the information contained in the data D, and is closer to the true distribution we are looking for.

$P(\boldsymbol{D}|\theta)$ is called the *likelihood*, since it tells how likely the sample $\boldsymbol{D}$ is to be picked assuming according to our parameters distribution. If the d_j in $\boldsymbol{D}$ are independent one another, we can write the likelihood as:

$$P(\boldsymbol{D}|\theta) = \prod_{j=1}^{M} P(d_j|\theta)\,,$$

$P(\boldsymbol{D})$ is called the *evidence* and is a normalising constant ensuring that $P(\theta|\boldsymbol{D})$ integrates to one. It sometimes plays a more important part than normalisation (it can be used to compare different models, for instance).

Note: We will always use the index j for data points or functions evaluated at those points.

3.1.3 Bayesian Learning: an example

Let us consider the following problem to introduce Bayes' rule. X is a random variable having an unknown probability distribution we try to find out. We assume this distribution is Gaussian with unknown mean μ (which we want to determine) and known variance σ^2. We have a dataset $\boldsymbol{D} = (x_1, ..., x_M)$ of M random realisations for X (which are equivalent to randomly sampling M points from the unknown distribution).

We will start by assuming $P_0(\mu) = \mathcal{N}(\mu_0, \sigma_0^2)$, with mean chosen according to our belief and variance big enough to account for our uncertainty with regard to the mean, and use the data set $\boldsymbol{D}$ to refine it. For this, we write the posterior distribution using Bayes' rule:

$$P_1(\mu) = P(\mu|\boldsymbol{D}) = \frac{P(\boldsymbol{D}|\mu)}{P(\boldsymbol{D})}\,P_0(\mu)\,. \tag{3.2}$$

We can write the likelihood as:

$$\begin{aligned}
P(\boldsymbol{D}|\mu) &= \prod_{j=1}^{M} P(x_j|\mu) \\
&= \prod_{j=1}^{M} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x_j-\mu)^2} \\
&= \frac{1}{(2\pi\sigma^2)^{M/2}} e^{-\frac{1}{2\sigma^2}(\sum_{j=1}^{M} x_j - M\mu)^2} \\
&= \frac{1}{(2\pi\sigma^2)^{M/2}} e^{-\frac{M}{2\sigma^2}(\bar{x}-\mu)^2}
\end{aligned}$$

And thus:

$$\begin{aligned}
P(\mu|\boldsymbol{D}) &= \frac{1}{Z} P(\boldsymbol{D}|\mu) P_0(\mu) \\
&= \frac{1}{Z'} e^{-\frac{M}{2\sigma^2}(\bar{x}-\mu)^2} e^{-\frac{1}{2\sigma_0^2}(\mu-\mu_0)^2} \\
&= \frac{1}{Z''} \exp\left[-\frac{M}{2\sigma^2}\left(\bar{x}^2 - 2\bar{x}\mu + \mu^2\right) - \frac{1}{2\sigma_0^2}\left(\mu_0^2 - 2\mu_0\mu + \mu^2\right)\right] \\
&= \frac{1}{Z'''} \exp\left[-\frac{\sigma^2 + M\sigma_0^2}{2\sigma^2\sigma_0^2}\left(\mu - \frac{\sigma^2\mu_0 + M\sigma_0^2\bar{x}}{\sigma^2 + M\sigma_0^2}\right)^2\right]
\end{aligned}$$

We know that this integrates to one and Z''' is constant with respect to μ, so Z''' is the integral of

the exponential part, that is to say the whole distribution is the Gaussian $\mathcal{N}(\mu_1, \sigma_1^2)$, where:

$$\mu_1 = \frac{\sigma^2}{\sigma^2 + M\sigma_0^2}\mu_0 + \frac{M\sigma_0^2}{\sigma^2 + M\sigma_0^2}\bar{x}$$
$$\frac{1}{\sigma_1} = \frac{1}{\sigma_0^2} + \frac{M}{\sigma^2}$$

$\bar{x}$ denotes the mean of the dataset:

$$\bar{x} = \frac{1}{M}\sum_{j=1}^{M} x_j$$

Figure 3.2 shows the posterior distribution thus obtained with different sample sizes.

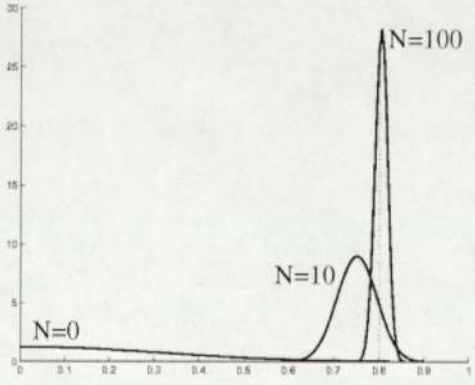


Figure 3.2: From a prior distribution (N=0) $\mathcal{N}(0, 0.1)$, we compute the posterior distribution using samples from the unknown distribution of size N=10 points and N=100 points. We observe that the more information we include from the unknown distribution, the more confident we are as to the distribution of μ. The unknown distribution had mean 0.8. As the sample size increases, the posterior focuses around the true mean, and the confidence in this value increases (ie the variance decreases).

3.2 Radial Basis Functions (RBF) networks

Radial basis functions are an efficient way to tackle non-linear regression problems. A radial basis function network is defined as a weighted sum of N functions (the radial basis functions) ϕ, as follows:

$$y_i(\boldsymbol{x}) = \sum_{k=1}^{N} w_{ik}\,\phi_k(\boldsymbol{x}) + \phi_0\,, \tag{3.3}$$

where $\boldsymbol{x} = (x_1, ..., x_M)$ is referred to as the input vector, M being the input space dimension. The output vector $\boldsymbol{y}$ is defined as $(y_1, ..., y_d)$, where d is the dimension of the output space. ϕ_0 is a constant additive term called the *bias*. Figure 3.3 shows a common way to conceptualise graphically an RBF network.

The basis functions can be of different types, but we will only focus on Gaussian basis functions

$$\phi_k(\boldsymbol{x}) = \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{\mu_k}\|^2}{2\sigma_k^2}\right).$$

w_{ik}, μ_k and σ_k^2 are the parameters for the k^{th} basis functions. It is those parameters we'll have to optimise so as to make our network as close to the function to be interpolated as possible.

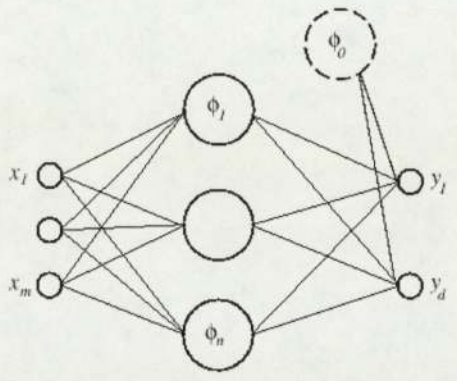


Figure 3.3: An RBF network with n=3 basis functions, with m=3 input nodes and d=2 output nodes. This RBF can be seen as a function from a 3d-space to a 2d-space.

The first step in a problem of interpolation with an RBF network is to choose the structure of the network, that is to say how many hidden units we think are relevant to tackle with our problem at best. There are rules of thumb to set that number efficiently. What one has to keep in mind is that the greater the number of hidden units, the more flexible the network. Setting this number to a too small value would make the function too rigid (Figure 3.4 (b)), whereas a too big value would make it too flexible, resulting in overfitting, i.e. fitting the imperfections in the data too well (Figure 3.4 (d)).

Once the structure is chosen, the parameters have to be optimised so that the network fits the sample best. We will not cover optimisation methods because it would require much theory that has furthermore been treated in depth on other works (see C. Bishop [3] for a good coverage of Bayesian theory and neural networks). Basically, the optimisation is achieved by optimising the model's parameters so as to minimise an error function (usually some distance between the model and the data, for instance the negative log-likelihood) with an optimisation algorithm (the Scaled Conjugate Gradient algorithm is used in this project).

Note: We will always use the index k for the network's basis functions.

3.3 The state-space model (Kalman filter)

The state-space model is a model in which an observed quantity I (a rainfall intensity, for example) depends on a state variable $\boldsymbol{\theta}$ (a rainfall model's parameters, for instance) which evolves through time. Both quantities evolve in parallel, linked together by a system of coupled equations. The Kalman filter is a convenient way to define such a framework when the observed quantities are linearly related to the state.

3.3.1 Introduction

To introduce the state-space model, we will follow D. Lowe's introduction to the Kalman filter [11]. We assume there is an unknown process evolving through time, from which we get observations (I_t) at regular time (a time-series). We want to model this process so that we can predict it. We assume the model is Gaussian and fully defined at time t with a mean parameter vector $\boldsymbol{\theta}_t$, usually called *state vector* since it gives account of the state of the model, and covariance matrix Σ_t^{θ}.

The Kalman filter is a two step algorithm. It basically involves the forecast of the state vector using a known dynamic rule (evolution step) and its update using the new observation (assimilation

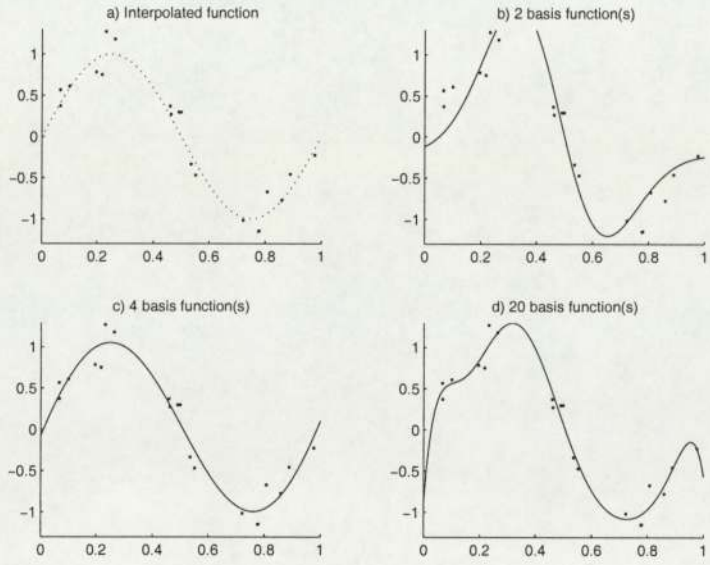


Figure 3.4: (a) shows the unknown function and the sample set drawn from it with added noise. The regression result is shown with (b) N=2, (c) N=5 and (d) N=20 hidden nodes. (b) shows our network is not flexible enough to fit the sample, whereas (d) is too flexible and fits the data too close (overfitting); (c) appears to be an appropriate choice for the network's structure.

step). We assume all the distributions treated are Gaussian.

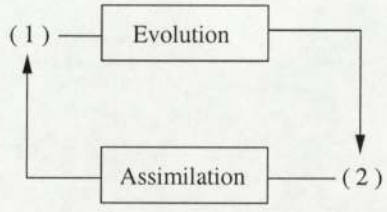


Figure 3.5: The 2 steps of a Kalman filter.

Observation equation

The observations come from noisy measurements of the process. We assume the process is a linear function F_t of the parameter vector $\boldsymbol{\theta}_t$ and the noise ϵ_t^I is Gaussian with mean zero (white noise) and covariance matrix Σ_t^ϵ. We then can write the observation in the matrix form as:

$$I_t = F_t\,\boldsymbol{\theta}_t + \epsilon_t^I$$

State equation

The state vector at time $t+1$, $\boldsymbol{\theta}_{t+1}$, is generated by applying the evolution function E_t (assumed known) to the previous state vector $\boldsymbol{\theta}_t$. This process includes white noise ν_t with covariance matrix Σ_t^ν. The noise is weighted with a weights matrix W_t for better accuracy. The evolution equation is thus:

$$\boldsymbol{\theta}_{t+1} = E_t\,\boldsymbol{\theta}_t + W_t\,\nu_t \tag{3.4}$$

3.3.2 Evolution step

Between two observations, the observed process evolves. We thus have to make the model evolve too, that is to say to we have to evolve $\boldsymbol{\theta}_t$. The evolution dynamics are assumed to be known (equation 3.4) and the model new state is conditioned on its previous state. The probability distribution for $\boldsymbol{\theta}_{t+1}$ given all previous observations $\mathcal{I}_t = (I_0, ..., I_t)$ is conditioned on $\boldsymbol{\theta}_t$. We thus need to integrate their joint probability over all possible values for $\boldsymbol{\theta}_t$:

$$p(\boldsymbol{\theta}_{t+1}|\mathcal{I}_t) = \int p(\boldsymbol{\theta}_{t+1}, \boldsymbol{\theta}_t|\,\mathcal{I}_t)\, d\boldsymbol{\theta}_t \tag{3.5}$$

$$= \int p(\boldsymbol{\theta}_{t+1}|\,\boldsymbol{\theta}_t)\, p(\boldsymbol{\theta}_t|\,\mathcal{I}_t)\, d\boldsymbol{\theta}_t \tag{3.6}$$

The distribution for $\boldsymbol{\theta}$ at time t given $\mathcal{I}_t$ is assumed Gaussian: $p(\boldsymbol{\theta}_t|\,\mathcal{I}_t) = \mathcal{N}(\hat{\boldsymbol{\theta}}_{t|t}, \hat{\Sigma}^{\theta}_{t|t})$, and $p(\boldsymbol{\theta}_{t+1}|\,\boldsymbol{\theta}_t)$ can be computed using the evolution equation (3.4)[1]:

$$p(\boldsymbol{\theta}_{t+1}|\,\boldsymbol{\theta}_t) = \mathcal{N}(E_t\,\boldsymbol{\theta}_t, W_t\,\Sigma^{\nu}_t\,W_t^T)\,.$$

From there, using the properties of Gaussian integration, we obtain the evolved state distribution ((2) on Figure 3.5):

$$p(\boldsymbol{\theta}_{t+1}|\,\mathcal{I}_t) = \mathcal{N}(\hat{\boldsymbol{\theta}}_{t+1|t}, \hat{\Sigma}^{\theta}_{t+1|t}) \tag{3.7}$$

with

$$\hat{\boldsymbol{\theta}}_{t+1|t} = E_t\,\hat{\boldsymbol{\theta}}_{t|t} \tag{3.8}$$

$$\hat{\Sigma}^{\theta}_{t+1|t} = E_t\,\hat{\Sigma}^{\theta}_{t|t}\,E_t + W_t\,\Sigma^{\nu}_t\,W_t^T\,. \tag{3.9}$$

3.3.3 Assimilation step

Suppose we have achieved the evolution using the state at time $t-1$, thus obtaining a forecast of the process at time t. Then, the real observation I_t is issued, and we have to use the information it bears to correct our model and ensure it remains close as close to the process as possible. We can write the posterior distribution:

$$p(\boldsymbol{\theta}_t|\mathcal{I}_t) = \frac{p(I_t|\boldsymbol{\theta}_t)\,p(\boldsymbol{\theta}_t|\mathcal{I}_{t-1})}{p(I_t|\mathcal{I}_{t-1})} \tag{3.10}$$

This is Bayes' rule, but written in a time-changing context. If one is not convinced, just write it with time $t = 0$ to find the usual formula. Since the variables are linked with their previous state, we have to keep trace of this constraint, hence the densities are conditioned on the whole sequence of observations up to time $t-1$.

We define the (Gaussian) distributions as follow:

- *Likelihood:* $p(I_t|\boldsymbol{\theta}_t) = \mathcal{N}(F_t\,\boldsymbol{\theta}_t, \Sigma^{\epsilon}_t)$, ie the observation is centred around the observed process's value with uncertainty due to the noise
- *Prior:* $p(\boldsymbol{\theta}_t|\mathcal{I}_{t-1}) = \mathcal{N}(\hat{\boldsymbol{\theta}}_{t|t-1}, \hat{\Sigma}^{\theta}_{t|t-1})$, $\hat{\Sigma}^{\theta}_{t|t-1}$ being the uncertainty in our estimation of θ_t
- *Evidence:* $p(I_t|\mathcal{I}_{t-1}) = \mathcal{N}(\hat{I}_{t|t-1}, \Sigma^{I}_t)$, Σ^{I}_t being the uncertainty in our prediction for I_t.

We can then write the posterior as a Gaussian (product of Gaussians) having estimated state vector and covariance matrix updated using the new data.

[1] When using double time indexes, the first denotes the step in the process and the second the image in the sequence associated. For example, $\boldsymbol{\theta}_{t+1|t}$ is the value for $\boldsymbol{\theta}$ at time $t+1$ using only the observations up to I_t.

- *Posterior:* $p(\boldsymbol{\theta}_t|\mathcal{I}_t) = \mathcal{N}(\hat{\boldsymbol{\theta}}_{t|t}, \hat{\Sigma}^{\theta}_{t|t})$

In order to find $\hat{\boldsymbol{\theta}}_{t|t}$ and $\hat{\Sigma}^{\theta}_{t|t}$, we will search for the maximum a posteriori (MAP) estimate, ie the value of $\hat{\boldsymbol{\theta}}_{t|t}$ that minimises the negative log of the posterior.

For this, we rewrite the posterior using Bayes rule and take the negative log, thus getting the equation (the constant terms have been dropped):

$$
\begin{aligned}
-\ln p(\theta_t|\mathcal{I}_t) &\propto (\boldsymbol{\theta_t} - \hat{\boldsymbol{\theta}}_{t|t})^T \, (\hat{\Sigma}^{\theta}_{t|t})^{-1} \, (\boldsymbol{\theta}_t - \hat{\boldsymbol{\theta}}_{t|t}) && (3.11)\\
&\propto (I_t - F_t\boldsymbol{\theta}_t)^T \, (\Sigma^{\epsilon}_t)^{-1} \, (I_t - F_t\boldsymbol{\theta}_t) && (3.12)\\
&\quad + (\boldsymbol{\theta}_t - \hat{\boldsymbol{\theta}}_{t|t-1})^T \, (\hat{\Sigma}^{\theta}_{t|t-1})^{-1} \, (\boldsymbol{\theta}_t - \hat{\boldsymbol{\theta}}_{t|t-1}) && (3.13)\\
&\quad - (I_t - \hat{I}_{t|t-1})^T \, (\Sigma^{I}_t)^{-1} \, (I_t - \hat{I}_{t|t-1}) && (3.14)
\end{aligned}
$$

We can expand this equation and equate the quadratic terms in $\boldsymbol{\theta}_t$, thus getting:

$$
(\hat{\Sigma}^{\theta}_{t|t})^{-1} = F_t^T \, (\Sigma^{\epsilon}_t)^{-1} \, F_t + (\hat{\Sigma}^{\theta}_{t|t-1})^{-1}, \tag{3.15}
$$

and, using the fact that its derivative with respect to $\boldsymbol{\theta}_t$ is null (optimum) and incorporating 3.15 in the result, we find:

$$
\hat{\boldsymbol{\theta}}_{t|t} = \hat{\boldsymbol{\theta}}_{t|t-1} + K_t \, \boldsymbol{e}_{t|t-1} \tag{3.16}
$$

where

$$
K_t = \hat{\Sigma}^{\theta}_{t|t} \, F_t^T \, (\Sigma^{\epsilon}_t)^{-1}
$$

is called the Kalman gain, and

$$
\boldsymbol{e}_{t|t-1} = \boldsymbol{\theta}_t - \hat{\boldsymbol{\theta}}_{t|t-1}
$$

is our estimate error.

Equation (3.16) states that our estimate evolves by being added a certain fraction of error at each step, this quantity being the filter's gain.

It can be shown that those expressions can be rewritten in a more computationally efficient form (with less matrix inversions in the expression of $\hat{\Sigma}^{\theta}_{t|t}$) using Σ^{I}_t:

$$
\begin{aligned}
\Sigma^{I}_t &= F_t \, \hat{\Sigma}^{\theta}_{t|t-1} \, F_t^T + \Sigma^{\epsilon}_t && (3.17)\\
K_t &= \hat{\Sigma}^{\theta}_{t|t-1} \, F_t \, (\Sigma^{I}_t)^{-1} && (3.18)\\
\hat{\Sigma}^{\theta}_{t|t} &= (Id - K_t \, F_t) \, \hat{\Sigma}^{\theta}_{t|t-1} && (3.19)
\end{aligned}
$$

where Id is the identity matrix (written as such not to be confused with the observation I_t).

3.3.4 Extension

In our case, the equations which govern the dynamics of rainfall fields are not linear (they are detailed in the following chapter). They involve the variables as well as their first order derivatives. To be able to use a "Kalman filter" like approach, approximations are made so that we can replace the derivatives by their first order Taylor expansions (making the assumption that the change in time is small enough).

Furthermore, the Gaussian assumption is dropped for the distributions, which we choose to have different, more general, shapes. Our model can thus be seen as a generalisation of a Kalman filter (relating it to the extended Kalman filter would be interesting, but knowing too little about this notion, we prefer not to talk about it).

All the details of Batail and Cornford's model will be introduced in the following chapter. They rely on the theoretical notions we have described in this chapter, but are often adaptations of those, so we will give enough details for the reader to be able to understand without wasting time trying to relate them to the general theory.

Chapter 4

The model

In this chapter, the existing model of Batail and Cornford is described and explained.

4.1 Overview

The way the model works can be thought of as a Kalman filter. The data observed is a sequence of radar images I_t, where t denotes the evolution in time. Our aim is, given a series of images $S_t = (I_1, ..., I_t)$ to be able to predict the state I_{t+1}. We will consider I as a set of observations associating to each coordinates couple (x_j, y_j) the intensity I_j.

We will often drop the time index for better readability. When not specified, the values are assumed to be taken at time t.

4.1.1 The rainfall rate R

Our model is a simplified representation of the reality of rainfall physics. It is mainly reduced to two variables linked together by the advection equation (explained later): the rainfall rate R, which models the rainfall intensity I, and the advection field $\boldsymbol{u}$, which models the spatial movement of I.

The rainfall rate R is modelled by an RBF network (see section 3.2) with 2D-Gaussian basis functions:

$$R = \sum_{k=1}^{N} \sum_{j=1}^{M} h_k \exp\left(-\frac{\lambda_k}{2} \left(\|cx_k - x_j\|^2 + \|cy_k - y_j\|^2 \right) \right) . \tag{4.1}$$

This means that R can be viewed as a mixture of N 2D-Gaussians having centre (cx_k, cy_k), width $\frac{1}{\lambda_k}$ and height h_k. Those Gaussians are spherical, meaning their width is the same in every direction (i.e. λ is a scalar, not a matrix). It could be interesting to use a full matrix for λ in order to have basis functions with elliptic support instead of circular. This would certainly add flexibility to R and improve the way we model I.

The easiest way to illustrate it is to consider a spatial visualisation. A single basis function is shown on Figure 4.1. By adding N similar functions with different parameters correctly tuned, we can approach the 3D radar image on Figure 2.2 (i.e. by constructing an RBF network and optimising it with respect to the image).

Those basis functions are assumed to have a physical equivalent, namely the rainfall cells. A rainfall field is not a solid object. It is usually made of different rain *cells* evolving together in a bigger ensemble. Assuming those cells produce a Gaussian-shaped rain rate (high rain in the centre and normal decay), we can think of the basis function as being 'rain cells' added to give the whole rain

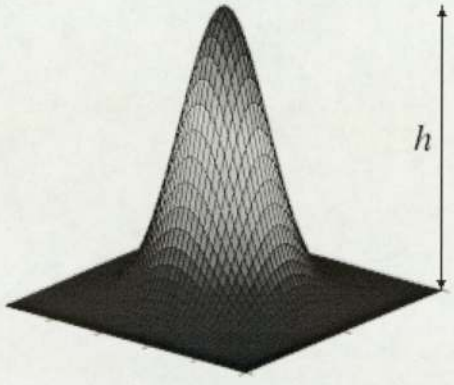


Figure 4.1: 3D representation of R

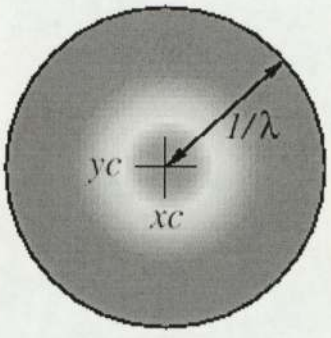


Figure 4.2: 2D representation of R

field. Even if this conception is a bit inaccurate, we will use this term to name the basis functions from now on (even though those are the rain rate associated to the cells, not the cells themselves).

We will use the following notation for the parameters of R :

$\boldsymbol{cx} = (cx_1, ..., cx_N)$ the centre x-coordinates for the model (same for $\boldsymbol{cy}$, $\boldsymbol{\lambda}$ and $\boldsymbol{h}$),

$\boldsymbol{\theta}_k = (cx_k, cy_k, \lambda_k, h_k)$ the parameter vector for the k^{th} cell,

$\boldsymbol{\theta} = (\boldsymbol{\theta_1}, ..., \boldsymbol{\theta}_N) = (\boldsymbol{cx}, \boldsymbol{cy}, \boldsymbol{\lambda}, \boldsymbol{h})$ the parameter vector for the whole rainfall rate (independent of their order in the vector).

Considering that the rainfall moves in time, we need another variable to model its movement before introducing how the model runs.

4.1.2 The advection field $\boldsymbol{u}$

The advection field can be thought of, in a very simple way, as the wind flow responsible for the rainfall movement. Actually, there are more complicated processes involved in this phenomenon (the attraction between cells, for instance), but since we deliberately want our model to remain simple, we define $\boldsymbol{u}$ as follows:

$$\boldsymbol{u}(x, y) = (u(x, y), v(x, y)),$$
$$\forall\{(x_1, y_1), ..., (x_s, y_s)\} \in \mathcal{F},\ P(\boldsymbol{u}(x_1, y_1), ..., \boldsymbol{u}(x_s, y_s))) = \mathcal{N}(\boldsymbol{\mu}_u, \Sigma_u)$$

where $\mathcal{F}$ is the radar image's coordinate space. That means $\boldsymbol{u}(x, y)$ is a 2d vector, and any finite subset from $span(\boldsymbol{u})$ has a Gaussian marginal density distribution, or in other words, $\boldsymbol{u}$ is a Gaussian process. The use of a Gaussian process ensures u varies through space in a continuous way (for example, we wouldn't want two close points to be advected in completely opposite directions). A sample from a 2D Gaussian process is illustrated in Figure 4.3.

In our model, the advection field definition space is restricted to the cells' (each cell has an associated movement) centres. An interesting evolution would be to have it sparsely defined all over the image, so that our rainfall cells really behave as if they were in a somewhat independent flow.

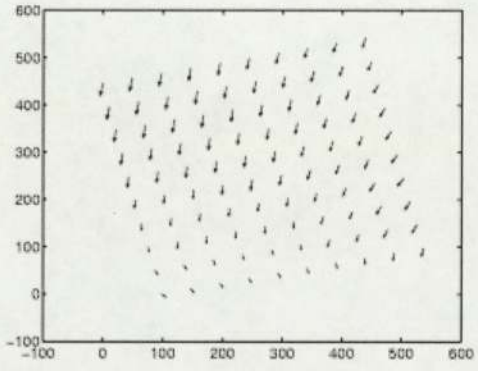


Figure 4.3: A 2D Gaussian process (courtesy of D. Cornford)

4.1.3 The advection equation

The advection equation links R and u together, and thus, time and space:

$$\frac{\partial R}{\partial t} = -\boldsymbol{u}.\nabla R \tag{4.2}$$

$$= -u\frac{\partial R}{\partial x} - v\frac{\partial R}{\partial y}\,. \tag{4.3}$$

This equation models the dynamics in our system. It states that the rainfall rate moves in space according to the advection field which 'advects' it. In this model, the advection is only applied to the cell centres, i.e. we consider the whole cell is advected as a block during the dynamic step. The other spatial modifications are ignored (the cell width and height are considered unchanged during its advection).

Let us explain where this equation comes from using a a simplified single dimension example. The rainfall rate R is advected through time according to the advection field u. If we consider a static point x in the spatial domain, the rainfall rate evolves with respect to its movement. This process is illustrated in Figure 4.4. The solid line represents R_t and the dashed line R_{t+1}, as functions of x. In

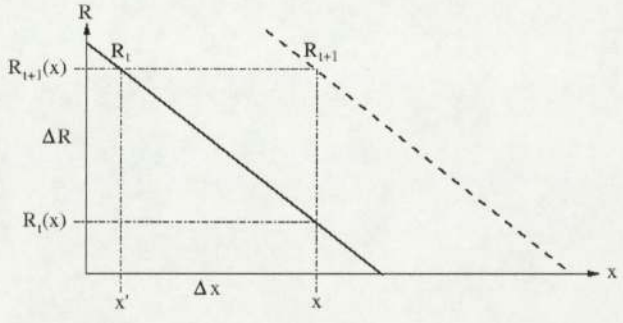


Figure 4.4: Evolution of the rainfall rate through time

a temporal approach, the model's movement is equivalent to a local change of intensity. We're thus interested in the amount ΔR of rainfall rate added in x. We can write:

$$R_{t+1}(x) = R_t(x) + \Delta R\,.$$

The difference in R is both proportional to its spatial derivative (the steeper the decrease, the bigger

ΔR) and its quantity of movement Δx :

$$\Delta R = -\frac{\partial R}{\partial x}\Delta x\,,$$

where $\Delta x = x - x'$. Note that if the spatial derivative of R is negative, it decreases, and

$$\begin{aligned} R_{t+1}(x) &= R_t(x' < x) \\ &> R_t(x)\,, \end{aligned}$$

hence the negative sign. Noting that Δx can be rewritten:

$$\Delta x = u\,\Delta t\,,$$

it comes:

$$\begin{aligned} R_{t+1}(x) &= R_t(x) - u\,\Delta t\,\frac{\partial R}{\partial x} \\ \frac{\Delta R}{\Delta t} &= -u\,\frac{\partial R}{\partial x} \end{aligned}$$

which becomes the advection equation when the differences tend to 0.

4.1.4 Priors

Our model is fully defined with the parameters of R: $(\boldsymbol{cx}, \boldsymbol{cy}, \boldsymbol{h}, \boldsymbol{\lambda})$ and those of $\boldsymbol{u}$: $(\boldsymbol{\mu}_u, \Sigma_u)$ (notice all those are vectors with length N where N is the number of cells). For those parameters, we need to specify a prior distribution, according to our knowledge or expectations for their probability density. Those priors are chosen as follow:

- The cell centres $\boldsymbol{c}$ are chosen to be Gaussian distributed, with mean vector $\bar{\boldsymbol{c}} = (\bar{cx}_1, ..., \bar{cx}_N, \bar{cy}_1, ..., \bar{cy}_N)$ and diagonal covariance matrix Σ_c (the centres are assumed to be uncorrelated).
- The heights $\boldsymbol{h}$ are chosen to be log-Gaussian with mean vector $\boldsymbol{\mu}_h$ and diagonal covariance matrix Σ_h (for we assume the heights are uncorrelated).
- The inverse widths $\boldsymbol{\lambda}$ are defined as the heights with mean vector $\boldsymbol{\mu}_\lambda$ and diagonal covariance matrix Σ_λ (they are also assumed to be uncorrelated).
- $\boldsymbol{u}$ has already been defined with as a Gaussian process with mean $\boldsymbol{\mu}_u$ and variance Σ_u.

The prior for R factorizes over the parameters as follows:

$$p(\boldsymbol{cx}, \boldsymbol{cy}, \boldsymbol{h}, \boldsymbol{\lambda}) = \prod_{k=1}^{N} p(cx_k, cy_k, h_k, \lambda_k)\,,$$

$$p(cx_k, cy_k, h_k, \lambda_k) = p(cx_k, cy_k)\,p(h_k)\,p(\lambda_k).$$

4.2 Initialisation

This part explains how the model is initialised before it runs.

4.2.1 Initialisation of R

The first thing to do is to build the RBF network R, using the first radar image in our sequence. For this, we have an iterative process that successively creates cells (i.e. basis functions) and optimise them. The following algorithm explains the process:

1. Create empty network R_0, initialise temporary radar image $I_{temp} = I_0$.
2. Find maximum intensity h_{max} of radar image I_0, it is located at (x_{max}, y_{max}).
3. Create a new cell having centre $(xc, yc) = (x_{max}, y_{max})$, width $\frac{1}{\lambda}$ (the initial value for λ is tuned by hand) and height h_{max}.
4. Proceed to optimisation.
5. Remove cell effect from I_{temp}.
6. Add cell with optimised (xc, yc, h, λ) to R_0.
7. Go back to 2. and repeat process on modified image I_0 until R_0 is sufficiently close to I_0.

The optimisation step is equivalent to minimising the negative log-prior for R. This step is a restriction to a single cell of the step detailed in section 4.3.3.

We thus end up with a network R_0 with optimised parameters giving a good representation of I_0, having N cells.

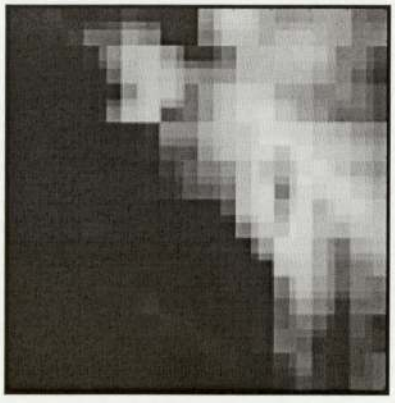


Figure 4.5: Initial radar image

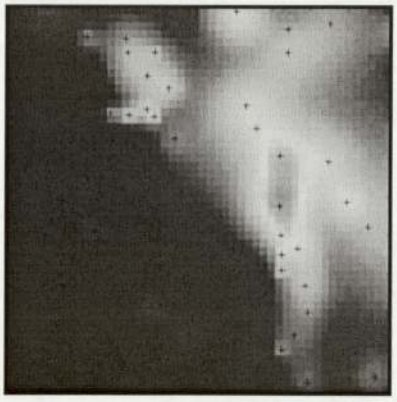


Figure 4.6: Initialised rainfall rate

Figure (4.6) shows the model initialised from the first image in a sequence (4.5). The crosses show the centres of the 38 cells used to model the rainfall intensity.

4.2.2 Initialisation of u

The advection field is initialised using the advection equation and the two first rainfall rates R_0 and R_1. R_1 is computed from the second image in the sequence, using the same procedure as for R_0. From the advection equation (4.2), it follows that:

$$\boldsymbol{u} = -\frac{\partial R}{\partial t} (\nabla R)^{-1} .$$

$\frac{\partial R}{\partial t}$ is estimated as the difference between the rainfall rate R_1 and R_0, while ∇R is computed from R_1 using the spatial differences between pixels as $\frac{\partial R}{\partial x}$ and $\frac{\partial R}{\partial y}$.

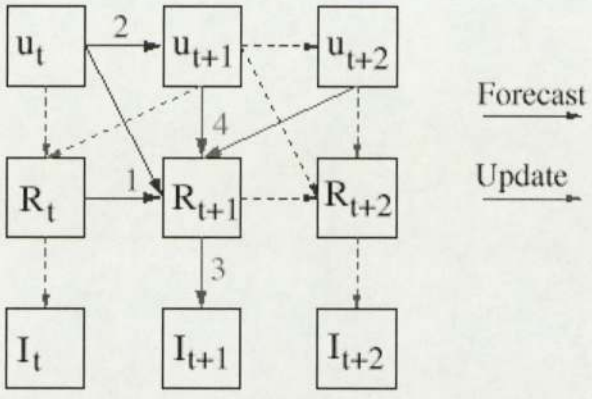


Figure 4.7: Dynamics of the model (courtesy of D. Cornford)

1. Evolution of R
2. Evolution of $\boldsymbol{u}$
3. Assimilation of R
4. Assimilation of $\boldsymbol{u}$

4.3 Dynamics

Once initialised, the model can run in an iterative way from a sequence of radar images. It has been shown in Section 3.3 that the Kalman filter involves evolution and update steps successively. The practical application to our model is detailed hereafter. Figure 4.7 illustrate the dynamics of the model.

4.3.1 Evolution of R

We use the advection equation as the evolution equation in the Kalman filter (section 3.3.2):

$$R_{t+1}(x, y) = R_t(x, y) - \boldsymbol{u}\,\nabla R\,,$$

which can be translated in terms of evolution of the cell's centre:

$$\begin{aligned} cx_{t+1|t} &= cx_t + u(cx_t, cy_t)\,\Delta t + \epsilon_c \\ cy_{t+1|t} &= cy_t + v(cx_t, cy_t)\,\Delta t + \epsilon_c \end{aligned}$$

In the implemented model, the mean of $\boldsymbol{u}$ has been used for simplicity.

The evolution's noise being supposed to have a zero mean Gaussian density with covariance Σ_{ϵ_c}, the estimated centres distribution becomes $p(cx_{t+1|t}, cy_{t+1|t}) = \mathcal{N}((\bar{cx}_{t+1|t}, \bar{cy}_{t+1|t}), \Sigma^c_{t+1|t})$, where:

$$\begin{aligned} \Sigma^c_{t+1|t} &= \Sigma^c_t + \Delta t^2\,\Sigma_{u_t} + \Sigma_{\epsilon_c} \\ \bar{cx}_{t+1|t} &= \bar{cx}_t + u_t(\bar{cx}_t, \bar{cy}_t)\,\Delta t \\ \bar{cy}_{t+1|t} &= \bar{cy}_t + v_t(\bar{cx}_t, \bar{cy}_t)\,\Delta t \end{aligned}$$

A small amount of noise is added to the variances of λ and h to reflect our uncertainty in the evolution process, thus giving estimated distributions $p(\lambda_{t+1|t})$ and $p(h_{t+1|t})$.

4.3.2 Evolution of u

u is assumed constant through time, and we only evolve it by adding a small quantity of white noise ϵ_u having same structure as u but a smaller variance Σ_{u_t} since the change is slow.

$$\boldsymbol{u}_{t+1|t} = \boldsymbol{u}_t + \epsilon_u \,,$$

so that the distribution for $\boldsymbol{u}$ becomes $p(\boldsymbol{u}_{t+1|t}) = \mathcal{N}(\boldsymbol{\mu}_{u_t}, \Sigma_{u_t} + \Sigma_{\epsilon_u})$. The assumption of a constant u is justified by the fact that in practice, the rainfall field R changes much faster than $\boldsymbol{u}$.

4.3.3 Assimilation of R

Once R has been moved according to the advection equation, and a new observation has been issued, we correct our estimate by assimilating the information in the new data into our model. As seen in the Kalman filter section (section 3.3.3), this step involves optimising the distribution for $(\boldsymbol{cx}, \boldsymbol{cy}, \boldsymbol{\lambda}, \boldsymbol{h})$ given the new image, which is computed using Bayes rule.

$$p(\boldsymbol{cx}_{t+1}, \boldsymbol{cy}_{t+1}, \boldsymbol{\lambda}_{t+1}, \boldsymbol{h}_{t+1}|I_{t+1}) = \frac{p(I_{t+1}|\boldsymbol{cx}_{t|t+1}, \boldsymbol{cy}_{t|t+1}, \boldsymbol{\lambda}_{t|t+1}, \boldsymbol{h}_{t|t+1})\, p(\boldsymbol{cx}_{t|t+1}, \boldsymbol{cy}_{t|t+1}, \boldsymbol{\lambda}_{t|t+1}, \boldsymbol{h}_{t|t+1})}{p(I_{t+1})} \tag{4.4}$$

Before developing the computation, we drop time indexes for better readability:

$$p(\boldsymbol{cx}, \boldsymbol{cy}, \boldsymbol{\lambda}, \boldsymbol{h}|I) = \frac{p(I|\boldsymbol{cx}, \boldsymbol{cy}, \boldsymbol{\lambda}, \boldsymbol{h})\, p(\boldsymbol{cx}, \boldsymbol{cy}, \boldsymbol{\lambda}, \boldsymbol{h})}{p(I)} \tag{4.5}$$

We then optimise the parameters (centres, inverse widths and heights) by minimising the error function defined by the negative log-posterior (equivalent to maximising the posterior, which means maximising the confidence in our model):

$$E \propto -\ln p(I|\boldsymbol{cx}, \boldsymbol{cy}, \boldsymbol{\lambda}, \boldsymbol{h}) - \ln p(\boldsymbol{cx}, \boldsymbol{cy}, \boldsymbol{\lambda}, \boldsymbol{h})$$

The prior over the parameters is Gaussian since the posterior (which will become the new prior) will be approximated by a Gaussian distribution using Laplace approximation (this step is detailed further).

Since the observations are uncorrelated, we can write the negative log-likelihood:

$$-\ln p(I|\boldsymbol{cx}, \boldsymbol{cy}, \boldsymbol{\lambda}, \boldsymbol{h}) = -\sum_{j=1}^{M} \ln p(I_j|\boldsymbol{cx}, \boldsymbol{cy}, \boldsymbol{\lambda}, \boldsymbol{h}).$$

Assuming a Gaussian distribution for the measurement process (radar) with noise σ^2, we can write the likelihood as a noisy realisation of the model:

$$p(I_j|\boldsymbol{cx}, \boldsymbol{cy}, \boldsymbol{\lambda}, \boldsymbol{h}) = \frac{1}{\sqrt{2\pi\sigma^2}}\, e^{-\frac{(I_j - R_j)^2}{2\sigma^2}}$$

where R_j denotes the model's evaluation at point (x_j, y_j),

$$R_j = R(x_j, y_j) = \sum_{k=1}^{N} h_k\, e^{-\frac{\lambda_k}{2}\left[(cx_k - x_j)^2 + (cy_k - y_j)^2\right]}.$$

Hence, dropping the constant term:

$$-\ln p(I|\boldsymbol{cx}, \boldsymbol{cy}, \boldsymbol{\lambda}, \boldsymbol{h}) \propto \frac{1}{2\sigma^2} \sum_{j=1}^{M} (R_j - I_j)^2 \,.$$

The negative log-prior is given by:

$$\begin{aligned} -\ln p(\boldsymbol{cx}, \boldsymbol{cy}, \boldsymbol{\lambda}, \boldsymbol{h}) &= -\ln p(\boldsymbol{cx}, \boldsymbol{cy}) - \ln p(\boldsymbol{h}) - \ln p(\boldsymbol{\lambda}) \\ &\propto \frac{1}{2}\boldsymbol{c}^T\Sigma_c^{-1}\boldsymbol{c} + \frac{1}{2}\boldsymbol{\lambda}^T\Sigma_\lambda^{-1}\boldsymbol{\lambda} + \frac{1}{2}\boldsymbol{h}^T\Sigma_h^{-1}\boldsymbol{h} \end{aligned}$$

Then, we find the parameters minimising the negative log-posterior, sometimes called *maximum a posteriori* parameters for they maximise the posterior distribution. Those parameters give a posterior distribution, often complex and difficult to integrate (we need to integrate it for the normalising constant). That is why we approximate it using Laplace approximation.

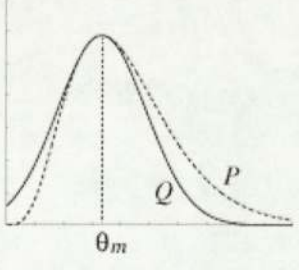
Laplace approximation

The Laplace approximation (see D. MacKay's introduction [12]) consists in approximating a complex probability distribution P about its maximum with a more tractable Gaussian distribution Q. It uses the fact that at the maximum $\boldsymbol{\theta}_m$, $\ln p(\theta)$ can be approximated using Taylor expansion [12], thus giving:

$$\ln p(\boldsymbol{\theta}) \approx \ln p(\boldsymbol{\theta}_m) - \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}_m)^T H(\boldsymbol{\theta} - \boldsymbol{\theta}_m)$$

where H is the Hessian of $\ln p$ at point $\boldsymbol{\theta}_m$, i.e.

$$H_{i,j} = \frac{\partial^2}{\partial\boldsymbol{\theta}_i\partial\boldsymbol{\theta}_j}\ln p(\boldsymbol{\theta})\Big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_m}$$



Then, if we note Z the normalising constant, p can be approximated at point $\boldsymbol{\theta}_m$ by

$$q(\boldsymbol{\theta}) = \frac{1}{Z}\, p(\boldsymbol{\theta}_m)\, \exp\left[-\frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta}_m)^T H(\boldsymbol{\theta} - \boldsymbol{\theta}_m)\right].$$

What we do in our case is compute the Hessian of the posterior distribution as a function of the whole optimised parameter vector $\boldsymbol{\theta}_m = (cx_1, ..., cx_N, cy_1, ..., cy_N, \lambda_1, ..., \lambda_N, h_1, ..., h_N)$. We thus can update the parameters posterior distributions using:

- the optimised values of $\boldsymbol{cx}$, $\boldsymbol{cy}$, $\boldsymbol{\lambda}$, $\boldsymbol{h}$ for the mean vectors,
- the appropriate parts of the Hessian diagonal for the covariance matrixes.

4.3.4 Assimilation of u

The advection field's assimilation is more simple since most of the difficulty has been put in the assimilation of R. $\boldsymbol{u}$ is then simply integrated over the uncertainty in R:

$$\begin{aligned} p(\boldsymbol{u}_{t+1}|I_t, I_{t+1}) &= \iint p(\boldsymbol{u}_{t+1}, \boldsymbol{c}_{t+1}, \boldsymbol{c}_t|I_t, I_{t+1})\, d\boldsymbol{c}_{t+1}\, d\boldsymbol{c}_t \\ &= \iint p(\boldsymbol{u}_{t+1}|\boldsymbol{c}_{t+1}, \boldsymbol{c}_t) p(\boldsymbol{c}_t|I_t) p(\boldsymbol{c}_{t+1}|I_{t+1})\, d\boldsymbol{c}_{t+1}\, d\boldsymbol{c}_t \end{aligned}$$

4.3.5 Forecast

Alternating evolution and assimilation steps, the model can run indefinitely as long as input data is provided. To forecast a time $t + n$, we run n evolution steps from time t, and then produce samples from the forecast distribution.

4.4 Issues

In its previous state (described in this chapter), the model ran and produced accurate forecasts. But there are several improvements one could think of to make it more robust:

- Though accurate, the model is very slow to run, which is a major issue in short-term forecasting, since we must be able to evolve and update the model in a time shorter than the time step between two radar images (if the model doesn't run faster than the process it models, then it is rather difficult to use it in a predictive way!).
- The observation noise isn't estimated from the data, which could be improved.
- u could be improved by using a sparse advection field, defined over the whole image not just at cells' centres. The dynamics of u could also be more realistic than the constant approximation we use.
- No real data test and comparison with existing techniques have been done so far.

We have introduced the model as it had been designed initially, describing how the rainfall rate and the advection field interact together in a state-space framework to forecast the rainfall field. We have also listed a number of issues affecting this model, the main being the speed of the model. The following chapters will discuss the work done to solve this issue.

Chapter 5

Speeding up the model with Variational Bayes

In the previous section, we have described the model and underlined several issues, amongst which the main being the update of R, very expensive in computation and thus needing a long time. Such a time makes the model simply not usable in concrete situations. Hence our choice to focus on solving this issue first. An analysis of the reasons for this issue has led us to consider applying variational methods in the hope of obtaining better results.

5.1 Analysis of the speed issue

From our theoretical work (previous chapter), we can infer that most of the computation time is dedicated to the update of R, where significant non-linear computation takes place. This feeling was confirmed by generating a Matlab report on a 1h 40min run of the model. The following parts of the process and the associated time are plotted in Table 5.1:

Step	*Time (s)*	*Time(%)*
Initialisation	1674.95	27.6
Evolution of the model	0.06	<1
Update of R	4385.65	72.4
Update of u	0.07	<1
Forecast	0.23	<1
Total time elapsed	6060.96	100

Table 5.1: Time involved by step

The most computationally intensive part is clearly the update of R, followed by the initialisation of the model. The former is not really important, since it is done once only, and before the model starts to run. Once the slow step was determined, we looked at possible reasons that could make it slow, and possible solutions to those.

We tried to reduce the accuracy of the model, by diminishing the training time and accepting larger errors in the fitting of the data. We ended up with correct mean values, but poor estimates of the distributions. This came from the fact that the optimisation is achieved by using the Laplace approximation. The Laplace approximation (cf. section 4.3.3) is a local approximation at the mode of the posterior distribution. Thus, to get a good estimate of the posterior, we need sufficient optimisation of the parameters to be close to the optimum (Figure 5.2). As a matter of fact, if we use the Laplace

approximation at some point a little too far from the optimum, the approximating Gaussian will give a poor estimate of the posterior. This compels us to carry out high optimisation to get very close to the maximum. Furthermore, with such an optimisation method, the approximate posterior is better than the prior only beyond a certain amount of optimisation. Carrying out the Laplace approximation too early would result in an estimated posterior distribution very far from the true posterior (Figure 5.2).

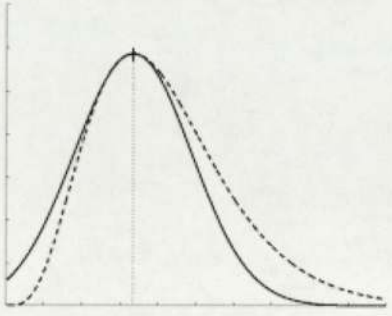


Figure 5.1: Laplace approximation (good). The Gaussian distribution (solid) gives a good local and correct global approximation of the true posterior (dashed)

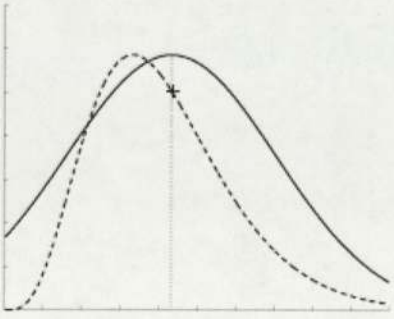


Figure 5.2: Laplace approximation (bad). The Gaussian distribution (solid) gives a very poor approximation of the true posterior (dashed)

In order to reduce the length of the optimisation process, we would like a better method to determine the posterior distribution. A global approximation, for instance, would be much better than a local approximation at the maximum probability. There are different ways to proceed to a global approximation. Sampling (such as in Monte-Carlo methods) is a common reliable way to get an approximation of a whole distribution, but it requires too much computation time in our case to be efficient. Thus, the framework we chose, in which the whole distributions are estimated during the optimisation, is the Variational Bayes framework. The following section discusses our work using Variational Bayes.

5.2 Variational Bayes

Variational Bayes was introduced in 1993 by Hinton and van Camp [9] in a paper where they showed that the true posterior distribution can be approximated by a Gaussian using a deterministic algorithm. The principle is similar to the Laplace approximation (see 4.3.3), except that instead of fitting the posterior distribution locally, the whole distribution is approximated by minimising the Kullback-Leibler (KL) distance between the posterior and a more tractable distribution. This method can be generalised to other approximating distributions than the Gaussian.

5.2.1 The Kullback-Leibler distance

Given two distributions p and q of a parameter vector $\boldsymbol{\theta}$, the KL distance between those two distributions is defined as follow:

$$\begin{aligned} KL(p||q) &= -\int q(\boldsymbol{\theta}) \ln \frac{p(\boldsymbol{\theta})}{q(\boldsymbol{\theta})} \, d\boldsymbol{\theta} \\ &= -\int q(\boldsymbol{\theta}) \ln p(\boldsymbol{\theta}) \, d\boldsymbol{\theta} + \int q(\boldsymbol{\theta}) \ln q(\boldsymbol{\theta}) \, d\boldsymbol{\theta} \end{aligned}$$

In our case, $\boldsymbol{\theta} = (\boldsymbol{cx}, \boldsymbol{cy}, \boldsymbol{\lambda}, \boldsymbol{h})$ and we want to approximate the posterior distribution. Using Bayes' rule, we can write its negative logarithm:

$$-\ln p(\boldsymbol{\theta}|I) = -\ln p(I|\boldsymbol{\theta}) - \ln p(\boldsymbol{\theta}) + \ln p(I)\,,$$

and, dropping the constant term, expand the KL-distance between q (which we want to determine) and the posterior:

$$KL(p||q) \propto -\int q(\boldsymbol{\theta})\ \ln p(I|\boldsymbol{\theta})\ d\boldsymbol{\theta} \tag{5.1}$$

$$-\int q(\boldsymbol{\theta})\ \ln p(\boldsymbol{\theta})\ d\boldsymbol{\theta} \tag{5.2}$$

$$+\int q(\boldsymbol{\theta})\ \ln q(\boldsymbol{\theta})\ d\boldsymbol{\theta}\,. \tag{5.3}$$

We will denote KL_1, KL_2 and KL_3 the integrals in the right part of this equation (the negative sign included), so that $KL(p||q) = KL_1 + KL_2 + KL_3$.

5.2.2 Distributions chosen

We chose to change the priors over the model for the following. Gamma distributions are chosen for the inverse widths and the heights, to ensure those remain positive, and a 2 dimensional Gaussian distribution conditioned on the widths is chosen for the centres.

$$\begin{aligned}
p(h_k) &= Ga(\gamma_k, \delta_k) = \frac{{\delta_k}^{\gamma_k}}{\Gamma(\gamma_k)}\, {h_k}^{\gamma_k - 1}\, e^{-\delta_k h_k} \\
p(\lambda_k) &= Ga(\alpha_k, \beta_k) = \frac{{\beta_k}^{\alpha_k}}{\Gamma(\alpha_k)}\, {\lambda_k}^{\alpha_k - 1}\, e^{-\beta_k \lambda_k} \\
p(cx_k, cy_k|\lambda_k) &= p(cx_k|\lambda_k)\, p(cy_k|\lambda_k) \\
&= \mathcal{N}\left(c\bar{x}_k, \frac{1}{\xi_k \lambda_k}\right) \mathcal{N}\left(c\bar{y}_k, \frac{1}{\xi_k \lambda_k}\right) \\
&= \frac{\xi_k \lambda_k}{2\pi}\, e^{-\frac{\xi_k \lambda_k}{2}\left[(c\bar{x}_k - cx_k)^2 + (c\bar{y}_k - cy_k)^2\right]}
\end{aligned}$$

α_k (γ_k for the heights) is called the *shape* parameter of the inverse width Gamma distribution, for it governs the shape the distribution takes. If it is smaller or equal to 1, the Gamma distribution is unbounded (it has no maximum). If it is greater than 1, the Gamma distribution decays asymmetrically around a maximum called the mode. This is this shape we are interested in, so we have to ensure the shape parameters remain strictly greater than 1 for our model to remain coherent.

β_k (δ_k for the heights) is called the *scale* parameter of the inverse width Gamma distribution, for it governs how high the mode (maximum probability) is. This will become obvious if we write the mode's value (when existing):

$$\lambda_k^{mode} = \frac{\alpha_k - 1}{\beta_K}$$

It is clear that we have to ensure β_k remains strictly positive for coherence of the model.

Note that we use a Gamma distribution for λ for computational ease, though in the model, it is more convenient to think in terms of widths, which then have a Inverse Gamma distribution. This doesn't matter since choosing a Gamma distribution for the inverse widths is equivalent to choose an Inverse Gamma for the widths, and their parameters are exactly the same.

The centres are now conditioned on the (inverse) widths, which is a better assumption than in the previous version of the model where both were uncorrelated. The joint distribution of the centres and

widths is called a Normal-Gamma (a study of such distributions can be found in [2]). $c\bar{x}_k, c\bar{y}_k$ and $\xi_k\lambda_k$ are the mean and inverse variance of the spherical Gaussian distribution for the centres. ξ is a parameter giving freedom to the shape of the Gaussian, so that the constraint on λ_k doesn't reduce flexibility too strongly. No domain constraint is set for the centre means, which can become negative. This makes sense since we work with finite radar images, where sometimes the effect of a rainfall cell which centre is outside the image (with possibly negative coordinates) must be taken in account.

For the approximating posterior distribution q, we choose the same structure as p. Thus, q will be the optimised version of p.

As in the previous version of the model, the distribution factorises independently over the cells as follows:

$$p(\theta) = \prod_{k=1}^{N} p_k(\theta_k)\,,\ \theta_k = (cx_k, cy_k, \lambda_k, h_k)\,, \tag{5.4}$$

$$q(\theta) = \prod_{k=1}^{N} q_k(\theta_k)\,. \tag{5.5}$$

It also factorises over the parameters, but now the centres are conditioned on the widths, so that:

$$p_k(\theta_k) = p_k(cx_k, cy_k|\lambda_k)\, p_k(\lambda_k)\, p_k(h_k)\,,$$

$$q_k(\theta_k) = q_k(cx_k, cy_k|\lambda_k)\, q_k(\lambda_k)\, q_k(h_k)\,.$$

5.2.3 Negative log-likelihood part (KL_1)

Let us first compute the first part of the distance:

$$KL_1(p||q) = -\int q(\theta)\ \ln p(I|\theta)\ d\theta\,. \tag{5.6}$$

This part of the distance is the correction the observation I brings to the model.

The observation I is a vector of all its data points, $I = (I_1, ..., I_M)$, $I_j = I(x_j, y_j)$. Those points are assumed to be picked up randomly, so we can write the negative log-likelihood:

$$-\ln p(I|\theta) = \sum_{j=1}^{M} -\ln p(I_j|\theta)\,.$$

Assuming a Gaussian distribution for the measurement process (radar) with noise σ^2, we can write the likelihood as a noisy realisation of the model:

$$\begin{aligned} -\ln p(I_j|\theta) &= -\ln\left(\frac{1}{\sqrt{2\pi\sigma^2}}\, e^{-\frac{(R_j - I_j)^2}{2\sigma^2}}\right) \\ &\propto \frac{1}{2\sigma^2}\ (R_j - I_j)^2 \end{aligned}$$

where R_j denotes the model's evaluation at point (x_j, y_j),

$$\begin{aligned} R_j &= R(\theta, x_j, y_j) \\ &= \sum_{k=1}^{N} R_j(\theta_k) \\ &= \sum_{k=1}^{N} h_k\, e^{-\frac{\lambda_k}{2}\left[(cx_k - x_j)^2 + (cy_k - y_j)^2\right]}\,. \end{aligned}$$

$R_j(\theta_k)$ denotes the effect of the k^{th} cell at point (x_j, y_j).

Hence:

$$\begin{aligned} -\ln p(I|\theta) &= \frac{1}{2\sigma^2} \sum_{j=1}^{M} (R_j(\theta) - I_j)^2 \\ &= \frac{1}{2\sigma^2} \sum_{j=1}^{M} (R_j^2(\theta) - 2I_j R_j(\theta) + I_j^2) \end{aligned}$$

(5.6) then becomes, after dropping the constant quadratic term in I_j:

$$KL_1 = \frac{1}{2\sigma^2} \sum_{j=1}^{M} \left[\int q(\theta)\, R_j^2(\theta)\; d\theta - 2I_j \int q(\theta) R_j(\theta)\; d\theta \right] \tag{5.7}$$

Let us denote $K_{11,j}$ and $K_{12,j}$ the first and second integrals in the previous equation, and compute $K_{12,j}$ first. Using 5.5, we can rewrite it:

$$\begin{aligned} K_{12,j} &= \int q(\theta)\, R_j(\theta)\; d\theta \\ &= \int q(\theta) \left(\sum_{k_1=1}^{N} R_j(\theta_{k_1}) \right) d\theta \\ &= \sum_{k_1=1}^{N} \int \left(\prod_{k=1}^{M} q_k(\theta_k) \right) R_j(\theta_{k_1})\; d\theta_1 ... d\theta_M \\ &= \sum_{k=1}^{N} \int q_k(\theta_k)\, R_j(\theta_k)\; d\theta_k \end{aligned}$$

In the last but one step, the product of q_k expands into a product of integrals, all but one (the one over θ_{k_1}) integrating out to one, hence the result.

We can see that this result relies on the q-average of R_j over θ_k. Let us compute this quantities. We drop the k index for better readability, but remind that θ, cx, cy, λ and h are in fact θ_k, cx_k, cy_k, λ_k and h_k.

$$\int q(\theta)\, R_j(\theta)\; d\theta \tag{5.8}$$

$$= \int q(cx, cy|\lambda)\, q(\lambda)\, q(h)\; h\, e^{-\frac{\lambda}{2}\left[(cx-x_j)^2+(cy-y_j)^2\right]} dcx\, dcy\, d\lambda\, dh \tag{5.9}$$

$$= \int q(h)\, h\, dh \;\times \int q(\lambda) \left[\int q(cx, cy|\lambda)\, e^{-\frac{\lambda}{2}\left[(cx-x_j)^2+(cy-y_j)^2\right]} dcx\, dcy \right] d\lambda \tag{5.10}$$

The integral in h is the first moment of a Gamma distribution:

$$\int q(h)\, h\, dh = \frac{\gamma}{\delta}, \tag{5.11}$$

$$\tag{5.12}$$

The integral over the centres in (5.10) (between brackets) is a bit more tricky. Let us compute it. Since the centres are normally distributed, we get the following expansion for the integral:

$$\frac{\xi\lambda}{2\pi} \int e^{-\frac{\xi\lambda}{2}\left[(\bar{cx}-cx)^2+(\bar{cy}-cy)^2\right]} e^{-\frac{\lambda}{2}\left[(cx-x_j)^2+(cy-y_j)^2\right]} dcx\, dcy \tag{5.13}$$

$$= \frac{\xi\lambda}{2\pi} \int e^{-\frac{\lambda}{2}\left[\xi(\bar{cx}-cx)^2+(cx-x_j)^2\right]} dcx \int e^{-\frac{\lambda}{2}\left[\xi(\bar{cy}-cy)^2+(cy-y_j)^2\right]} dcy \tag{5.14}$$

The integral in *dcy* is similar to the integral in *dcx*. Inside the integral is a product of Gaussians, which is also a Gaussian with parameters we are going to solve now.

Let us first notice that:

$$\xi(c\bar{x} - cx)^2 + (cx - x_j)^2 = [\xi + 1]\, cx^2 - 2\,[\xi c\bar{x} + x_j]\, cx + \left[\xi c\bar{x}^2 + x_j^2\right] \tag{5.15}$$

$$= A\, cx^2 - 2B\, cx + C \tag{5.16}$$

$$= A\left[cx - \frac{B}{A}\right]^2 + \left[C - \frac{B^2}{A}\right] \tag{5.17}$$

where:

$A = \xi + 1$,
$B = \xi c\bar{x} + x_j$,
$C = \xi c\bar{x}^2 + x_j^2$.

Given that:

$$\int e^{-\frac{\lambda}{2}A\left(cx - \frac{B}{A}\right)^2} dcx = \sqrt{\frac{2\pi}{\lambda A}}$$

and

$$C - \frac{B^2}{A} = \frac{\xi}{\xi + 1}\left(c\bar{x} - x_j\right)^2$$

The integral in *dcx* in (5.14) can then be rewritten:

$$\sqrt{\frac{2\pi}{\lambda(\xi + 1)}}\, e^{-\frac{\lambda}{2}\frac{\xi}{\xi+1}(c\bar{x} - x_j)^2}$$

and the whole integral (5.14) becomes:

$$\frac{\xi}{\xi + 1}\, e^{-\frac{\lambda}{2}\frac{\xi}{\xi+1}\left[(c\bar{x} - x_j)^2 + (c\bar{y} - y_j)^2\right]} \tag{5.18}$$

We then define $E = \frac{\xi}{\xi+1}\left[(c\bar{x} - x_j)^2 + (c\bar{y} - y_j)^2\right]$ and average (5.18) over $q(\lambda)$:

$$\begin{aligned}
&\frac{\xi}{\xi + 1}\int q(\lambda) e^{-\frac{\lambda}{2}E}\, d\lambda \\
&= \frac{\xi}{\xi + 1}\int \frac{\beta^\alpha}{\Gamma(\alpha)}\lambda^{\alpha-1}\, e^{-\beta\lambda} e^{-\frac{\lambda}{2}E}\, d\lambda \\
&= \frac{\xi}{\xi + 1}\,\frac{\beta^\alpha}{\Gamma(\alpha)}\int \lambda^{\alpha-1} e^{-\lambda\left(\beta + \frac{E}{2}\right)}\, d\lambda \\
&= \frac{\xi}{\xi + 1}\,\frac{\beta^\alpha}{\Gamma(\alpha)}\left(\beta + \frac{E}{2}\right)^{-\alpha}\int \Lambda^{\alpha-1} e^{-\Lambda}\, d\Lambda \\
&= \frac{\xi}{\xi + 1}\left(1 + \frac{E}{2\beta}\right)^{-\alpha} \\
&= \frac{\xi}{\xi + 1}\left(1 + \frac{1}{2\beta}\frac{\xi}{\xi + 1}\left[(c\bar{x} - x_j)^2 + (c\bar{y} - y_j)^2\right]\right)^{-\alpha}
\end{aligned}$$

using the change of variables $\Lambda = \left(\beta + \frac{E}{2}\right)\lambda$ to retrieve the Gamma integral evaluating to $\Gamma(\alpha)$.

Hence the preliminary result:

$$\int q(\theta)\, R_j(\theta)\, d\theta \;= \frac{\gamma_k}{\delta_k}\,\frac{\xi}{\xi + 1}\left(1 + \frac{1}{2\beta}\frac{\xi}{\xi + 1}\left[(c\bar{x} - x_j)^2 + (c\bar{y} - y_j)^2\right]\right)^{-\alpha}$$

Let us now compute $KL_{11,j}$. Like $KL_{12,j}$, we can rewrite it:

$$\begin{aligned}
K_{11,j} &= \int q(\theta) R_j^2(\theta)\ d\theta \\
&= \int q(\theta) \left(\sum_{k_1=1}^{N} R_j(\theta_{k_1}) \right) \left(\sum_{k_2=1}^{N} R_j(\theta_{k_2}) \right)\ d\theta \\
&= \sum_{k_1=1}^{N} \sum_{k_2=1}^{N} \int \left(\prod_{k=1}^{M} q_k(\theta_k) \right) R_j(\theta_{k_1}) R_j(\theta_{k_2})\ d\theta_1 ... d\theta_M \\
&= \sum_{k_1=1}^{N} \sum_{\substack{k_2=1 \\ k_2 \neq k_1}}^{N} \int q_{k_1}(\theta_{k_1})\, q_{k_2}(\theta_{k_2})\, R_j(\theta_{k_1}) R_j(\theta_{k_2})\, d\theta_{k_1}\, d\theta_{k_2} + \sum_{k=1}^{N} \int q_k(\theta_k)\, R_j(\theta_k)^2 d\theta_k \\
&= \sum_{k_1=1}^{N} \sum_{\substack{k_2=1 \\ k_2 \neq k_1}}^{N} \left[\int q_{k_1}(\theta_{k_1})\, R_j(\theta_{k_1})\, d\theta_{k_1} \times \int q_{k_2}(\theta_{k_2})\, R_j(\theta_{k_2})\, d\theta_{k_2} \right] + \sum_{k=1}^{N} \int q_k(\theta_k)\, R_j(\theta_k)^2 d\theta_k
\end{aligned}$$

This result relies on the q-averages of R_j and R_j^2 over θ_k. The first we have computed before, and we will compute the other now. As previously, we drop the k index for better readability.

$$\int q(\theta)\, R_j^2(\theta)\ d\theta \tag{5.19}$$

$$= \int q(cx, cy|\lambda)\, q(\lambda)\, q(h)\ h^2\, e^{-\frac{\mathbf{2}\lambda}{2}\left[(cx-x_j)^2+(cy-y_j)^2\right]}\, dcx\, dcy\, d\lambda\, dh \tag{5.20}$$

$$= \int q(h)\, h^2\, dh\ \times \int q(\lambda) \left[\int q(cx, cy|\lambda)\, e^{-\frac{\mathbf{2}\lambda}{2}\left[(cx-x_j)^2+(cy-y_j)^2\right]}\, dcx\, dcy \right] d\lambda \tag{5.21}$$

The integral in h is the second moment of a Gamma distribution:

$$\int q(h)\, h^2\, dh = \frac{\gamma(\gamma+1)}{\delta^2}, \tag{5.22}$$

$$\tag{5.23}$$

The integral over the centres is similar to the one in 5.10, except for a factor 2 due to the quadratic term in R_j. It is computed using the same method, but this time, we find:

$A' = \xi + 2,$

$B' = \xi c\bar{x} + 2x_j,$

$C' = \xi c\bar{x}^2 + 2x_j^2.$

and

$$C' - \frac{B'^2}{A} = 2\,\frac{\xi}{\xi+2}\,(c\bar{x} - x_j)^2$$

and for the whole integral averaged over $q(\lambda)$:

$$\frac{\xi}{\xi+2} \left(1 + \frac{1}{\beta}\frac{\xi}{\xi+2} \left[(c\bar{x} - x_j)^2 + (c\bar{y} - y_j)^2 \right] \right)^{-\alpha}$$

Hence the preliminary result:

$$\int q(\theta)\, R_j(\theta)^2\ d\theta\ = \frac{\gamma(\gamma+1)}{\delta^2}\,\frac{\xi}{\xi+2} \left(1 + \frac{1}{\beta}\frac{\xi}{\xi+2} \left[(c\bar{x} - x_j)^2 + (c\bar{y} - y_j)^2 \right] \right)^{-\alpha}$$

We can define the quantity $r_{k,j}^m$, $m \in \{1, 2\}$, reintroducing the k indexes, as:

$$r_{k,j}^m = \frac{\gamma_k}{\delta_k} \left(\frac{\gamma_k + 1}{\delta_k} \right)^{m-1} \frac{\xi_k}{\xi_k + m} \left(1 + \frac{1}{(3-m)\beta_k}\frac{\xi_k}{\xi_k + m} \left[(c\bar{x}_k - x_j)^2 + (c\bar{y}_k - y_j)^2 \right] \right)^{-\alpha_k}$$

$r^1_{k,j}$ is the q-average of R_j over θ_k, and $r^2_{k,j}$ the q-average of R_j^2.

And write the results:

$$\begin{aligned} KL_{12,j} &= \sum_{k=1}^{N} r^1_{k,j} \\ KL_{11,j} &= \sum_{k_1=1}^{N} \sum_{\substack{k_2=1 \\ k_2 \neq k1}}^{N} r^1_{k_1,j}\, r^1_{k_2,j} + \sum_{k=1}^{N} r^2_{k,j} \\ &= \left(\sum_{k=1}^{N} r^1_{k,j} \right)^2 - \sum_{k=1}^{N} \left(r^1_{k,j} \right)^2 + \sum_{k=1}^{N} r^2_{k,j} \end{aligned}$$

Hence:

$$KL_1 = \frac{1}{2\sigma^2} \sum_{j=1}^{M} \left[\left(\sum_{k=1}^{N} r^1_{k,j} \right)^2 - \sum_{k=1}^{N} \left(r^1_{k,j} \right)^2 + \sum_{k=1}^{N} r^2_{k,j} - 2I_j \sum_{k=1}^{N} r^1_{k,j} \right]$$

5.2.4 Negative log-prior part (KL_2)

In this section, we derive the computation of the second part of the KL distance:

$$KL_2(p||q) = -\int q(\theta)\ \ln p(\theta)\ d\theta\,.$$

Using the independence assumption in 5.4 we can rewrite KL_2 as:

$$KL_2(p||q) = \sum_{k=1}^{N} KL_{2,k}$$

where

$$\begin{aligned} KL_{2,k} &= -\int q(\theta)\ \ln p_k(\theta_k)\ d\theta && (5.24) \\ &= -\int q_k(\theta_k)\ \ln p_k(\theta_k)\ d\theta_k && (5.25) \end{aligned}$$

since the distributions are independent (all the $q_{k'}$, $k' \neq k$, in the factorisation of q separately integrate to 1). We already can see that the optimisation will be made on the whole parameters distributions, since the parameters $\theta_k = (cx_k, cy_k, \lambda_k, h_k)$ are integrated out. This corresponds to our wish to carry the whole distribution during the optimisation.

We see that we can restrict our discussion to a single cell for the computing and drop all the k indexes, and then sum up all the $KL_{12,k}$.

We will denote by $(\bar{cx}', \bar{cy}', \xi', \alpha', \beta', \gamma', \delta')$ the parameters of the distribution p and by $(\bar{cx}, \bar{cy}, \xi, \alpha, \beta, \gamma, \delta)$ the parameters of the distribution q.

Let us first expand $-KL_{2,k}$.

$$\begin{aligned} -KL_{2,k} &= \int q(cx, cy, \lambda, h)\ \ln p(cx, cy, \lambda, h)\ dcx\ dcy\, d\lambda\, dh \\ &= \int q(h) \ln p(h)\ dh \times \int q(\lambda)\ q(cx, cy|\lambda)\ \ln\left[p(cx, cy|\lambda) p(\lambda) \right]\ dcx\ dcy\, d\lambda\, dh. \end{aligned}$$

The integral in h gives:

$$\begin{aligned}\int q(h) \ln p(h)\ dh &= \int Ga(h,\gamma,\delta)\ \ln Ga(h,\gamma',\delta')\ dh \\ &= \int Ga(h,\gamma,\delta) \ln \left[\frac{\delta'^{\gamma'}}{\Gamma(\gamma')} h^{\gamma'-1} e^{-\delta' h}\right]\ dh \\ &= \ln \frac{\delta'^{\gamma'}}{\Gamma(\gamma')} + (\gamma'-1)\int Ga(h,\gamma,\delta) \ln h\ dh\ - \delta' \int Ga(h,\gamma,\delta)\ h\ dh\end{aligned}$$

The second integral is the first moment of the Gamma distribution, evaluating to $\frac{\gamma}{\delta}$. The first integral can be expanded into:

$$\begin{aligned}\int Ga(h,\gamma,\delta) \ln h\ dh &= \frac{\delta^\gamma}{\Gamma(\gamma)} \int e^{(\gamma-1)\ln h} \ln h\ e^{-\delta h}\ dh \\ &= \frac{\delta^\gamma}{\Gamma(\gamma)} \frac{\partial}{\partial \gamma}\left[\int e^{(\gamma-1)\ln h}\ e^{-\delta h}\ dh\right] \\ &= \frac{\delta^\gamma}{\Gamma(\gamma)} \frac{\partial}{\partial \gamma}\left[\frac{\Gamma(\gamma)}{\delta^\gamma}\right] \\ &= \frac{\delta^\gamma}{\Gamma(\gamma)} \left[\frac{\Gamma'(\gamma)\delta^\gamma - \Gamma(\gamma)\delta^\gamma \ln \delta}{\delta^{2\gamma}}\right] \\ &= \Psi(\gamma) - \ln \delta\end{aligned}$$

where $\Psi = \frac{\Gamma'}{\Gamma}$ is the di-gamma function.

Hence:

$$\int q(h) \ln p(h)\ dh = \ln \frac{\delta'^{\gamma'}}{\Gamma(\gamma')} + (\gamma'-1)\left(\Psi(\gamma) - \ln \delta\right) - \gamma \frac{\delta'}{\delta}$$

The integral in cx, cy, λ can be expanded into:

$$\begin{aligned}\int q(\lambda)\, q(cx,cy|\lambda)\ \ln\left[p(\lambda)\, p(cx,cy|\lambda)\right]\, dcx\, dcy\, d\lambda &= \int q(\lambda)\, q(cx,cy|\lambda)\ \ln p(\lambda)\ dcx\ dcy\, d\lambda \\ &\quad + \int q(\lambda)\, q(cx,cy|\lambda)\ \ln p(cx,cy|\lambda)\ dcx\ dcy\, d\lambda\end{aligned}$$

In the first integral, the normal distribution over the centres integrates to 1, leaving an integral similar to the one in h:

$$\int q(\lambda)\ \ln p(\lambda)\, d\lambda = \ln \frac{\beta'^{\alpha'}}{\Gamma(\alpha')} + (\alpha'-1)\left(\Psi(\alpha) - \ln \beta\right) - \alpha \frac{\beta'}{\beta}$$

The second integral needs to be integrated over cx, cy first and then over λ.

$$\int q(\lambda) \left[\int q(cx,cy|\lambda)\ \ln p(cx,cy|\lambda)\ dcx\ dcy\right] d\lambda$$

The integral over the centres gives:

$$\begin{aligned}&\int q(cx|\lambda)\, q(cy|\lambda)\ \left(\ln p(cx|\lambda) + \ln p(cy|\lambda)\right)\ dcx\ dcy \\ &= 2\int q(cx|\lambda)\ \ln p(cx|\lambda)\ dcx \\ &= \ln \frac{\xi'\lambda}{2\pi} - \xi'\lambda \int (cx - \bar{cx}')^2\, \mathcal{N}\left(\bar{cx}, \frac{1}{\xi\lambda}\right)\ dcx \\ &= \ln \frac{\xi'\lambda}{2\pi} - \frac{\xi'}{\xi}\end{aligned}$$

The first equality comes from the fact that the integrals in cx and cy are identical and sum up because of the logarithm. The integral in the second equality evaluates to the variance of the Gaussian, ie $\frac{1}{\xi\lambda}$, hence the result.

Averaging this result over $q(\lambda)$ gives:

$$\begin{aligned}\int q(\lambda)\left[\ln\frac{\xi'\lambda}{2\pi}-\frac{\xi'}{\xi}\right]d\lambda &= \int Ga(\lambda,\alpha,\beta)\ln\lambda\,d\lambda+\ln\frac{\xi'}{2\pi}-\frac{\xi'}{\xi}\\ &= \ln\frac{{\beta'}^{\alpha'}}{\Gamma(\alpha')}+(\alpha'-1)\left(\Psi(\alpha)-\ln\beta\right)-\alpha\frac{\beta'}{\beta}+\ln\frac{\xi'}{2\pi}-\frac{\xi'}{\xi}\end{aligned}$$

And summing the different preliminary results and reintroducing the k indexes, we find:

$$\begin{aligned}-KL_{2,k} = &\ln\frac{{\delta'_k}^{\gamma'_k}}{\Gamma(\gamma'_k)}+(\gamma'_k-1)\left(\Psi(\gamma_k)-\ln\delta_k\right)-\gamma\frac{\delta'_k}{\delta_k}\\ &+\ln\frac{{\beta'_k}^{\alpha'_k}}{\Gamma(\alpha'_k)}+(\alpha'_k-1)\left(\Psi(\alpha_k)-\ln\beta_k\right)-\alpha_k\frac{\beta'_k}{\beta_k}\\ &+\ln\frac{\xi'_k}{2\pi}-\frac{\xi'_k}{\xi_k}\end{aligned}$$

For a complete model, we have to sum the $KL_{2,k}$ over the cells, obtaining the final result for K_2:

$$\begin{aligned}KL_2 = -\sum_{k=1}^{N}\Bigg[&\ln\frac{{\delta'_k}^{\gamma'_k}}{\Gamma(\gamma'_k)}+(\gamma'_k-1)\left(\Psi(\gamma_k)-\ln\delta_k\right)-\gamma\frac{\delta'_k}{\delta_k}\\ &+\ln\frac{{\beta'_k}^{\alpha'_k}}{\Gamma(\alpha'_k)}+(\alpha'_k-1)\left(\Psi(\alpha_k)-\ln\beta_k\right)-\alpha_k\frac{\beta'_k}{\beta_k}\\ &+\ln\frac{\xi'_k}{2\pi}-\frac{\xi'_k}{\xi_k}\Bigg]\end{aligned}$$

5.2.5 Entropy of q (KL_3)

Let us now compute the third part of the distance:

$$KL_3(q) = -\int q(\theta)\ \ln q(\theta)\ d\theta\,.$$

This part is the entropy of the distribution q. Since p and q have the same definition, the result is exactly the same as for the computation of KL_2, since we integrate over the same variables, except that the parameters for p (with the prime) are replaced by the parameters for q (without the prime). We thus obtain the following result:

$$KL_3 = \sum_{k=1}^{N} \Bigg[\ln \frac{{\delta_k}^{\gamma_k}}{\Gamma(\gamma_k)} + (\gamma_k - 1)\left(\Psi(\gamma_k) - \ln \delta_k\right) - \gamma_k + \ln \frac{{\beta_k}^{\alpha_k}}{\Gamma(\alpha_k)} + (\alpha_k - 1)\left(\Psi(\alpha_k) - \ln \beta_k\right) - \alpha_k + \ln \frac{\xi_k}{2\pi} - 1 \Bigg]$$

5.2.6 Result

Summing KL_1, KL_2, and KL_3, we obtain the following result for the KL distance:

$$KL(p\|q) = \frac{1}{2\sigma^2} \sum_{j=1}^{M} \left[\left(\sum_{k=1}^{N} r_{k,j}^1 \right)^2 - \sum_{k=1}^{N} \left(r_{k,j}^1\right)^2 + \sum_{k=1}^{N} r_{k,j}^2 - 2I_j \sum_{k=1}^{N} r_{k,j}^1 \right] + \sum_{k=1}^{N} \Bigg[(\gamma_k - \gamma_k')\left(\Psi(\gamma_k) - \ln \delta\right) - \gamma_k \left(1 - \frac{\delta_k'}{\delta_k}\right) + \ln\left(\frac{\beta_k^{\alpha}\,\Gamma(\alpha_k')}{\beta'^{\alpha_k'}\,\Gamma(\alpha_k)}\right) + (\alpha_k - \alpha_k')\left(\Psi(\alpha_k) - \ln \beta_k\right) - \alpha\left(1 - \frac{\beta_k'}{\beta_k}\right) + \ln\left(\frac{\xi_k'}{\xi_k}\right) - \left(1 - \frac{\xi_k'}{\xi_k}\right) \Bigg]$$

where:

$$r_{k,j}^m = \frac{\gamma_k}{\delta_k} \left(\frac{\gamma_k + 1}{\delta_k}\right)^{m-1} \frac{\xi_k}{\xi_k + m} \left(1 + \frac{1}{(3-m)\beta_k} \frac{\xi_k}{\xi_k + m} \left[(c\bar{x}_k - x_j)^2 + (c\bar{y}_k - y_j)^2\right]\right)^{-\alpha_k}$$

As expected, the centres, inverse widths and heights have integrated out to leave only their distributions parameters. When we minimise the distance, the whole distribution will be updated, so we expect lower accuracy to be necessary for relevant results.

The steps in the optimisation of the KL-distance are listed in section 6.2 of the next chapter, which details the changes brought to the model.

Chapter 6

The new model

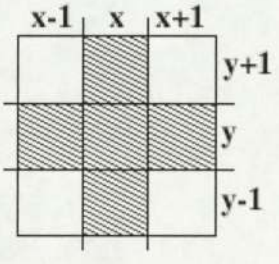
The new model is very similar to the former as far as its structure is concerned. Only the fact that the optimisation of R is now done using Variational Bayes as described in the previous chapter, and a few improvements such as the smoothing of the data before using it, make it different from its previous version. This chapter will discuss the modifications brought to the model and its tests on simulated data.

6.1 Initialisation

The initialisation step is similar to the one described in section 4.2.1, except it uses the minimisation of the KL-distance as the optimisation method instead of the minimisation of the negative log-posterior.

6.1.1 Smoothing

We have added one more step yet, which consists in smoothing the radar data before using it in order to make it more continuous and diminish the effect of noise. The smoothing algorithm is very simple and based on the fact that the noise is assumed to have mean zero. Thus, each pixel in the radar image is replaced by the mean of it and its closest neighbours (pixels highlighted on side picture). The averaging doesn't alter significantly the value of the intensity at the pixel, but makes the noise smaller, since it is averaged over a local subset of the data.



The results of the smoothing are plotted on Figure 6.1 below. On the left, the raw data, and on the right, the smoothed data. We can see that the smoothing has removed the angles in the data, making it closer to a noiseless continuous observation.

6.1.2 Local fitting

We have also implemented a local fitting feature, which consists in fitting each new cell on a portion of the whole radar image localised around the centre (determined by the maximum intensity). The reason for this is that a radar image usually contains many rain cells, with bounded width. Fitting the whole radar image would mean trying to approximate the whole set of cells with a single Gaussian, which then would no more represent a cell as we wish, but an poor approximation of the complete radar image. Thus, when creating a new cell, we fit it on a local part of radar image (typically of size 20×20 pixels, i.e. 40×40 km, centred at the Gaussian's centre). The size of the sub-image is

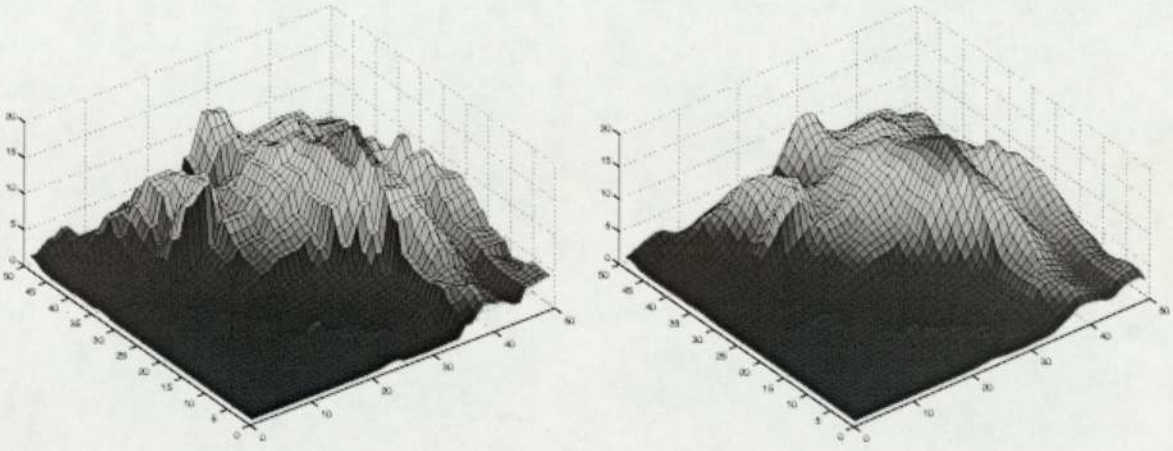


Figure 6.1: Effect of smoothing on the data

determined by experience, but would certainly require further analysis as to the usual sizes of rainfall cells, for more accurate tuning.

6.2 Dynamics of the model

The dynamics of the model have been left unchanged, except for the use of the KL-distance as an error function instead of the negative log-prior formerly used. For plots and forecast, we needed to set values for the parameters (cx, cy, λ, h). The modes have been used for the heights and inverse widths (Gamma distributions), and the means for the centres (Gaussian).

Optimisation of the parameters

The tip to ensure variables are optimised under the constraint that they remain positive is to optimise their logarithm instead. In the case of α and γ, we optimise the logarithm of $\alpha - 1$ and $\gamma - 1$:

1. Pack the parameters: $\boldsymbol{\theta} = (\bar{cx}, \bar{cy}, \ln(\alpha - 1), \ln(\beta), \ln(\gamma - 1), \ln(\delta), \ln(\xi))$
2. Optimise $\boldsymbol{\theta}$ using SCG. This step requires the KL distance and its gradient. Functions have been provided which compute those values, using sub-functions computing the $r^m_{k,j}$ and their gradient too.
3. Unpack $\boldsymbol{\theta}$: $(\exp(\bar{cx}), \exp(\bar{cy}), \alpha - 1, \beta, \gamma - 1, \delta, \xi) = \exp(\theta)$, necessarily positive because of the exponential, and greater than 1 for α and γ.

6.3 Tests

Before running the new model on real data, we needed to ensure the new optimisation method was working in the expected way. Thus, we developed a testing platform to train it on simulated data (in order to be able to compare the results with known parameters). We then carried out experiments, from the most simple tasks (fitting one single cell) to the most complex (fitting several cells, from two to a large number).

6.3.1 Simulated data

Testing a model on real data has a major inconvenience: there is no easy way to determine whether the model's parameters are accurate or not, for we don't know what they should be. Thus, it was

necessary to ensure the model worked the way we expected by training it on data where the parameters could be easily inferred and compared to the results. We designed a tool to simulate radar images, by generating noisy Gaussian mixtures with customisable parameters (image size, number of cells, heights and widths bounds, amount of noise,...). Different simulated images are shown on Figure(6.2).

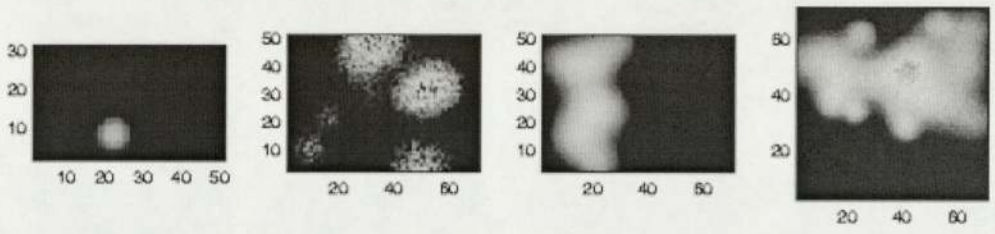


Figure 6.2: Simulated radar images with 1,10,10 and 20 cells. The second image from the left has large noise, the two images on the right are 'adherent', i.e. they're built as congregated rainfall fields.

6.3.2 Fitting a single cell

This first test was made to see how the model would fit a single rainfall cell. We simulated data, namely a Gaussian with added noise. Figure 6.3 illustrates this test. The picture on the left shows the simulated data after smoothing, the picture in the middle shows the prior for the rain model (the centre is determined by the maximum of the simulated data), and the picture on the right shows the rain model after fitting using Variational Bayes.

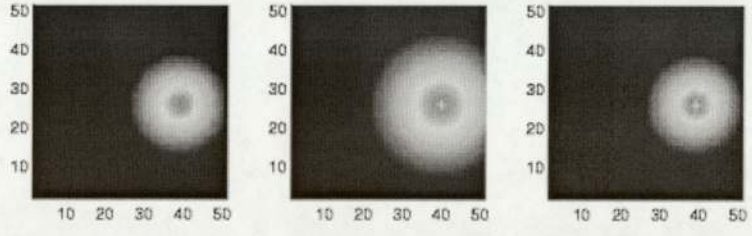


Figure 6.3: Single cell fitting. From left to right: simulated data, initialised model, optimised model.

The initial and final parameters means are listed in the table below, as well as the simulated data's:

We can observe that the KL distance decreases significantly (it doesn't tend to zero since all the constant terms in the computation have been dropped). The end parameters (means) correspond fairly well to the values of the simulated data. At the same time, the variances of the parameters decrease, which means we are getting more confident in our estimate of the parameters. The distributions are plotted on Figure 6.4, before (left) and after (right) fitting. The optimised posterior shows much higher confidence in the mean values than the prior. This gives us a good confirmation that the optimisation works as expected.

The parameter evolution is shown in appendix, Figure A.1. It is interesting to notice the decreasingly oscillating character of the optimisation for the means and modes, which reminds of acoustic or electronic oscillating fade-out. We must also notice that with 50 optimisation iterations, we reach the state of convergence. This looks promising, for it means we could be able to reach an accurate state with a small number of iterations.

State	Initial	Final	Data
KL distance	-65 950	-508 100	–
xc (mean)	40	39.13	39
yc(mean)	26	26.12	26
w (mode)	115.7	48.67	48
h (mode)	16.45	16.35	16
xi	1000	49 510	–
xc(variance)	0.1157	0.0010	–
yc (variance)	5	0.0106	–
w (variance)	0.1157	0.0010	–
h (variance)	25	0.0011	–

Table 6.1: Single test fitting. Variables initial and final states.

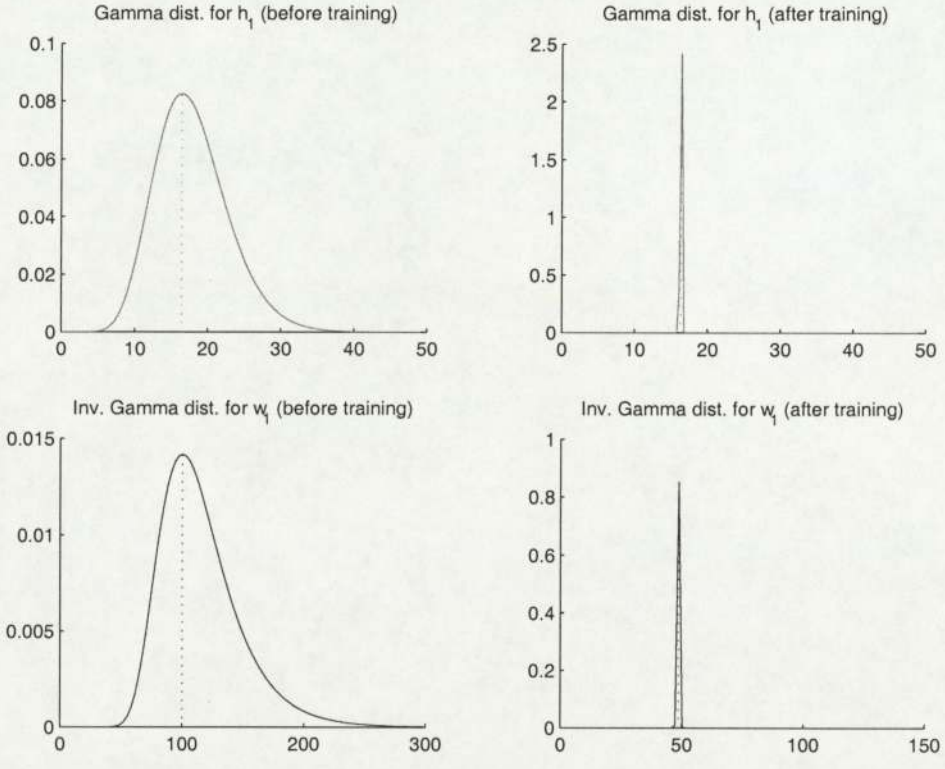


Figure 6.4: Evolution of the distributions. The means get closer to the simulated data values, while the variances decrease (beware the changes in the scale of axis).

Known issue: importance of the prior's choice

The choice of the prior has non negligible importance. Choosing a prior too far from the true distribution can cause the model to crash in the optimisation step. Though we are not absolutely certain about the reasons for this, it seems very likely to be that while using the Scaled Conjugate Gradient algorithm to minimise the KL-distance, a bad prior results in big optimisation steps at the beginning, and very large changes/oscillations in the parameter values, which sometimes get out of machine precision. It is thus important to tune the prior distribution for the parameters so that it remains "relatively" close to the expected true distribution.

We have given the values used in this test (Table 6.1), and we hope they are relevant enough to cope with most type of data. If preliminary tuning happened to be required each time the data is changed, this would make the model very impractical.

6.3.3 Fitting two cells

The results with the single cell fitting being satisfying, we set up a second test, with simulated data having two cells this time. The aim of this experiment is to test the ability of the model to carry out the simultaneous optimisation of several cells, which is a step further in testing the model's features.

The cells are two Gaussians, as shown on the left picture on Figure 6.5.

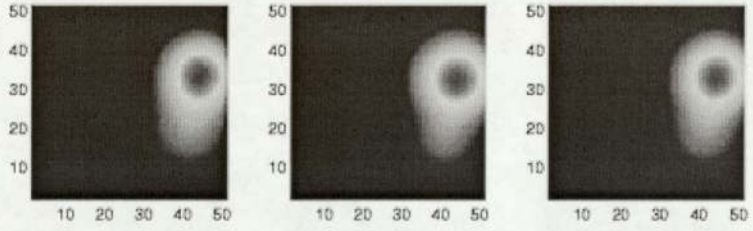


Figure 6.5: Two cells fitting. Fitting the whole rainfall. From left to right: simulated data, initialised model, optimised model.

We fit the data with a method similar to the initialisation step in the model, ie by fitting a first cell as in the previous test, removing its effect from the data, and fitting a second cell on the updated data. This gives us our initial image (middle picture in 6.5). We then run an optimisation of the whole 2-cells model, in order to fit the whole data, now. The results are shown in Table 6.2 below.

	First cell			*Second cell*		
State	Initial	Final	Data	Initial	Final	Data
KL distance	-352 488	-353 897	–	-352 488	-353 897	–
xc (mean)	44.14	44.27	44	41.10	41.30	41
yc(mean)	32.73	33.49	33	17.10	18.82	20
w (mode)	45.83	40.92	37	31.15	38.09	45
h (mode)	19.660	19.46	19	5.13	5.79	6
xi	589 940	599 738	–	129 519	131 264	–
xc(variance)	7.768e-5	6.823e-5	–	2.481e-4	2.901e-4	–
yc (variance)	7.768e-5	6.823e-5	–	2.481e-4	2.901e-4	–
w (variance)	7.197e-4	6.011e-4	–	3.264e-3	4.382e-3	–
h (variance)	1.577e-3	1.543e-3	–	1.122e-3	1.340e-3	–

Table 6.2: Two cells test fitting. Variables initial and final states.

We can notice that the first cell gains in confidence (the variances decrease) whereas the second loses. This comes from the fact that the initialisation is done cell by cell, on local areas, and then the single cells are summed to give the whole model. This shows it is necessary to optimise the whole model on the whole data, after optimising, to remove the imperfections due to the initialisation. Yet, too much importance should not be given to the initialisation, for it is a step run only once before running the state-space model, and most of the imperfections will be corrected once the model is running.

It is not really obvious on Figure 6.5 that the right picture (whole model optimised) is a better approximation of the left picture than the middle picture (initialised model, with no global optimisation). The graphs for the parameters are given in an appendix, where we can clearly see that the KL distance is minimised, which gives evidence that the whole optimised model, though locally less accurate, is globally closer to the data.

6.3.4 Fitting a large number of cells

Testing the optimisation method on a greater number of cells still gives satisfying results (see Figure 6.6). As far as time is concerned (remember that is our main concern), we can notice that the time taken to optimise the model increases significantly with the number of cells. We have not carried out experiments to find out which amount of optimisation was sufficient for good accuracy so far, but it is certainly something that should be done in the future.

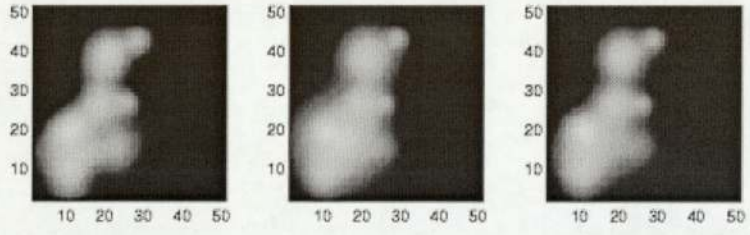


Figure 6.6: Fifteen cells fitting. Fitting the whole rainfall. From left to right: simulated data, initialised model, optimised model.

6.3.5 Further testing

The testing phase should be carried on. Here are a number of tests which could be relevant:

- Compare the old model with the new one using similar tests as those described in this chapter. Evaluate and compare both their accuracy (i.e. measure both parameter variances) and their efficiency (i.e. their computational time to reach equal accuracies).
- Test the new model on real radar images.
- Build and test the dynamics of the new model on real data (or simulated data first if wanted). Compare their accuracy and computation time.
- Compare both full models on real data.

Additional tests will certainly have to be performed, for it is likely that some of the tests listed go wrong, compelling to find more specific tests to track the possible errors or incoherences.

6.4 Conclusions

We did not have time in this project to go any further in testing the model, so only the tests concerning the optimisation method have been carried out. Those test show good results so far, but it is clear that further tests have to be done. First, the optimisation method must be tested on real data, and then the model must be tested in its dynamic form.

The issue with the prior's choice also has to be better understood and steps taken to solve it.

Chapter 7

Conclusion

7.1 Summary of the work done

This thesis has given an account of the work done during 9 months, as well as of the realities of research. Here are the main points we have covered in this report.

After having introduced the main theoretical concepts on which the model is built, namely the Kalman filter in a Bayesian state-space framework, we have explained the model itself, as well as its issues. We have underlined in the introduction the fact that working on someone else's work is not an easy thing, and explained why documenting one's work is essential when working on a long-term project that will certainly involve new developments from other persons in the future. Aware that this project would certainly see other modifications, we have tried to give a presentation of the model as clear as possible, which the reader may have found a little too detailed.

Once the model had been introduced, we analysed the issues and suggested solutions, namely the use of Variational Bayes. We have detailed the Variational Bayes theory as it is applied to this model, and detailed the tests to which we submitted the optimisation method. We didn't have time in this project to test the full dynamic model on real data, nor to compare it with its previous version.

7.2 Results

The preliminary results we discussed in this thesis are satisfying. We have seen that the new optimisation method had good performance fitting simulated data, with fast convergence and accurate results. This gives hope that the model, once fully implemented and tested, will represent a significant advance in the domain of stochastic real-time forecasting.

7.3 Further work

The time dedicated to this project was a little too short to bring the model to a fully working state, thus there is still work to carry on. Further directions are given here, which could be a basis for a future MSc project.

Amongst the issues met, we have mentioned the importance in the choice of the priors. When starting with an initial state of the model too far from the data, we often end up with the model crashing during the optimisation step. We suspect this comes from the sensitivity of the model to

its initial conditions, which, when too inaccurately chosen, cause the variables to take values out of machine precision. We think it is necessary to understand better the way the initialisation affects the optimisation, in order to set up an initialisation method which would prevent the model starting in too inaccurate an initial state. For the moment, hand tuning is required for the model to start, but an operational model would need this step to be automated. Understanding and solving this issue appears to us as a priority in further developments.

Once this issue is solved, testing the complete model in a dynamic framework, on real data, would confirm its robustness. For it to be operational, another step would be required, which would consist in optimising the code, which we believe is not as efficient as could be (but we didn't have time to improve it). The computation of the KL distance needs much mathematical computation, which if not optimised, can make the model very impractical.

Then, providing a comparison of the Variational Bayes model and Batail and Cornford's model, as well as comparing it with other existing models, would complete the work.

Other improvements can be thought of. We have discussed about the initialisation, and how we cope with it, and we believe it would be worth trying other initialisations methods, for even if the initialisation is only used once before running the model, it defines the model's structure once for all. We think a complete initialisation method aimed at analysing the data and finding the best priors would bring a lot more robustness to the model as it currently is.

We also want to mention the fact that so far, the advection field is cell-dependent, that is to say each cell is associated with its movement. It could be a good improvement to have a cell-independent advection field, defined on the whole image, and each rain cell would move according to the value of the advection field at its centre.

All those remarks show that there is still work to do on this project.

7.4 Afterword

This project has been interesting in many ways, from the abstract theory it involves to its practical hydrologic applications. It has given us the opportunity to learn new methods and understand better those learnt before, but also to work in a domain different to the one of our scholar background (computer engineering), and on all points, we consider it as a rich experience. It certainly is a little frustrating not to have had enough time to take the project to a complete operational state, which was our initial objective, but this is one of the aspects of a research work. It is an interesting piece of work, and to conclude this thesis, let us just express our hope that it will be carried on and will contribute to make research progress in stochastic weather forecasting.

Appendix A

Test parameter estimations

In this appendix are gathered the plots for the parameter evolutions in the tests carried out in chapter 6 (fitting one and two cells).

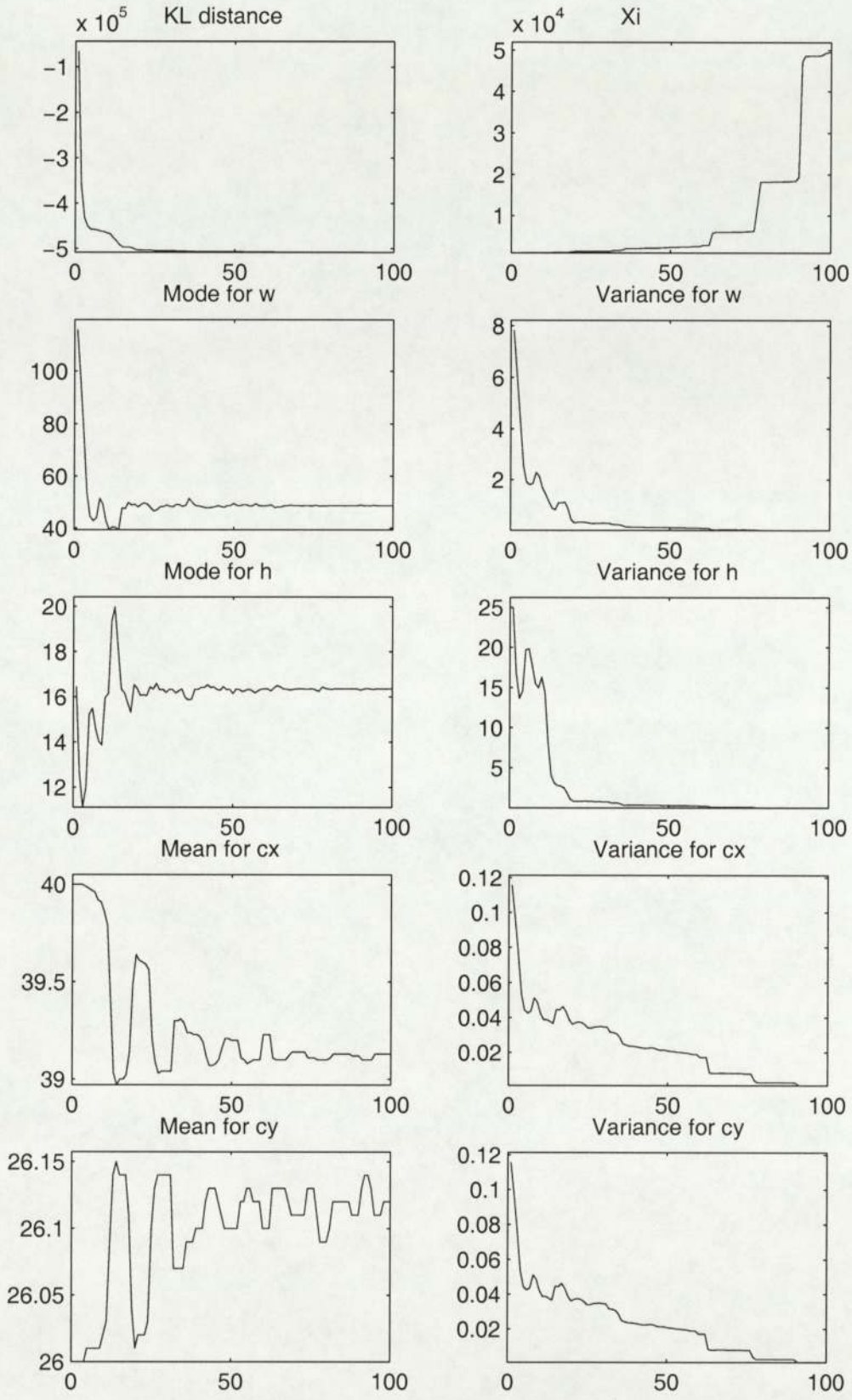


Figure A.1: Single cell fitting. Evolution of the parameters.

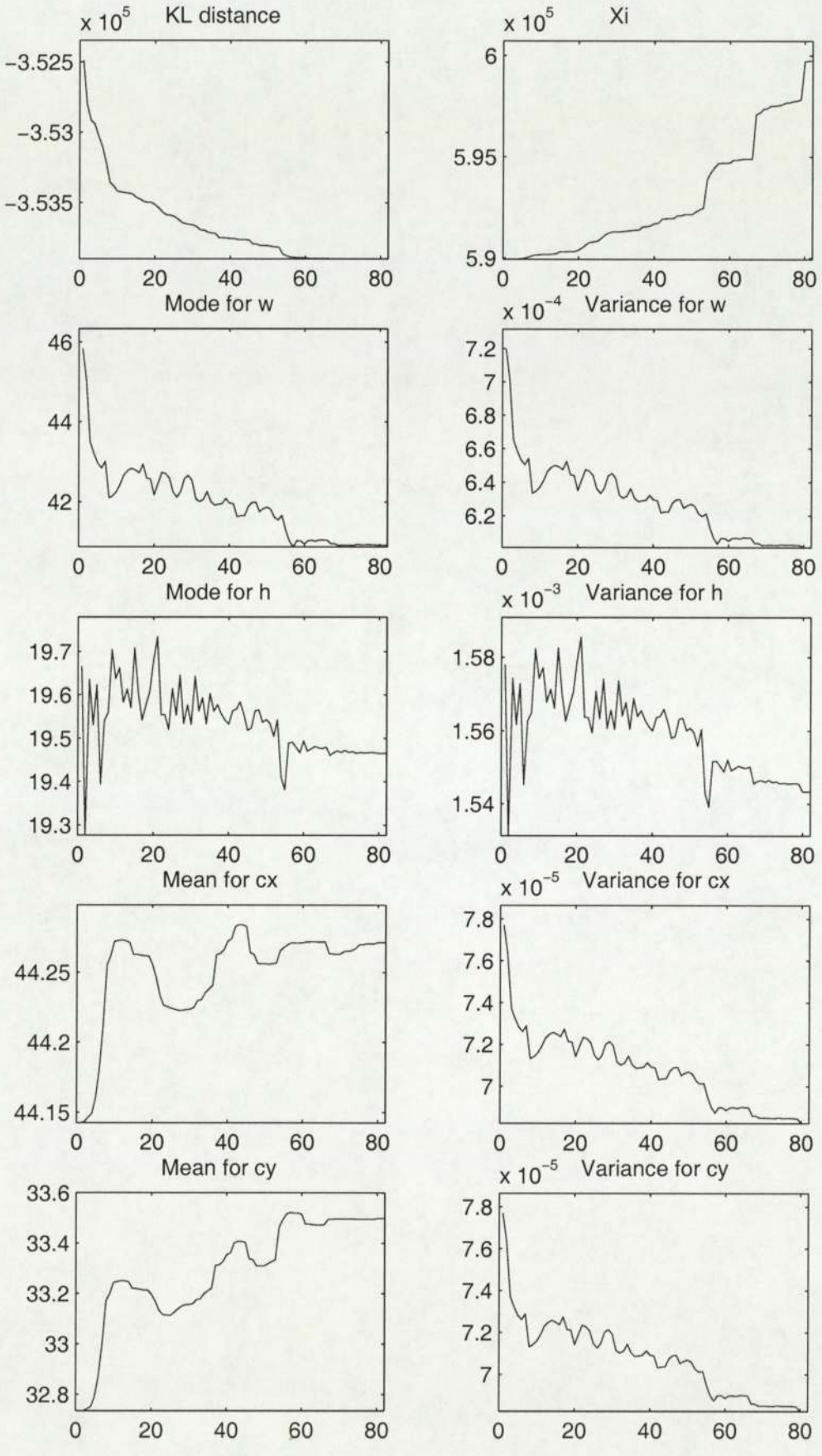


Figure A.2: Two cells fitting. Evolution of the parameters (first cell).

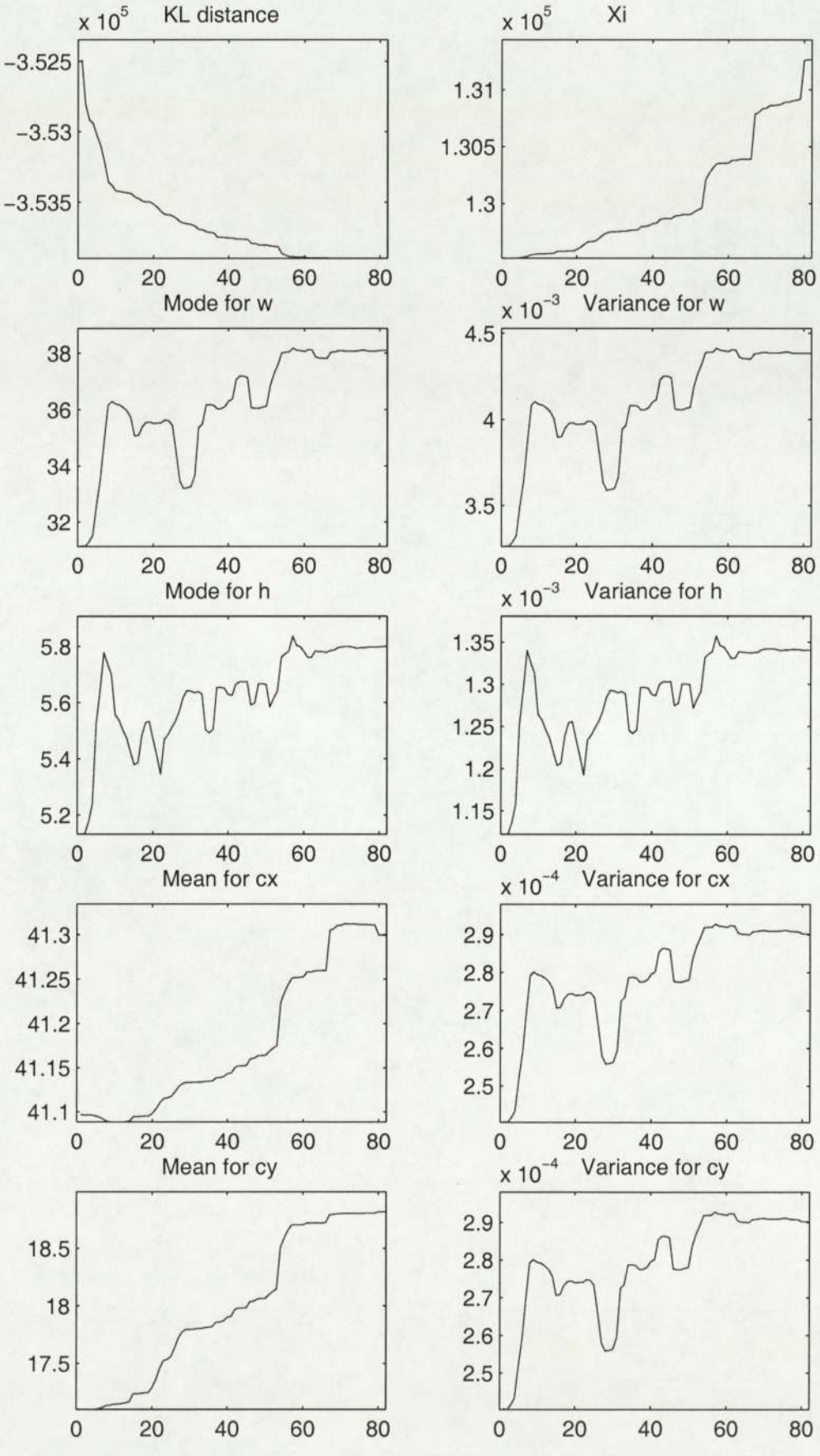


Figure A.3: Two cells fitting. Evolution of the parameters (second cell).

Index

λ, 22

Advection, 23
Advection equation, 24
Advection, assimilation, 29
Advection, evolution, 27
Advection, initialisation, 26
Assimilation (Kalman filter), 20
Assimilation, of the advection, 29
Assimilation, of the rainfall rate, 28

Bayes' rule, 15
Bayesian learning, 16

Centers, 22
cx, 22
cy, 22

Evidence, 16
Evolution step (Kalman filter), 19

h, 22
Heights, 22

Kalman filter, 18
Kullback-Leibler distance, 32

Laplace approximation, 29
Likelihood, 16

Model, 22
Model, dynamics, 27
Model, forecast, 29
Model, initialisation, 25
Model, KL distance, 41

Negative log-prior, 38
Numerical Weather Prediction, 10

Observation equation, 19

Posterior distribution, 15

Radar images, 12
Radial Basis Functions (RBF) network, 17
Rainfall cell, 22
Rainfall rate, 22
Rainfall rate, assimilation, 28
Rainfall rate, evolution, 27
Rainfall rate, initialisation, 26

Smoothing, 42
State equation, 19

Variational Bayes, 31

w, 22
Widths, 22

Bibliography

[1] E. Batail. Statistical models for rainfalls: caracterization and forecasting. Master's thesis, Aston University, 2002.

[2] J. Bernardo and A. Smith. *Bayesian theory.* Wiley series in probability and statistics. 1994.

[3] C. Bishop. *Neural Networks for Pattern Recognition.* Oxford, 1996.

[4] V. Collinge. *The development of weather radar in the UK.* Wiley, 1987.

[5] D. Cornford. Probabilistic precipitation prediction. 2002.

[6] D. Cornford. When to hang out the laundry? probabilistic precipitation forecasting using bayesian state space models. Natural Computing Application Forum, Jan. 2003.

[7] T. Einfal and T. Denoeux. Never expect a perfect forecast. *Hydrological applications of weather radar*, pages 452–458, 1991.

[8] M. French, W. Krajewski, and R. Cuykendall. Rainfall forecasting in space and time using a neural network. *Journal of Hydrology*, 137:1–31, 1992.

[9] G. Hinton and D. van Camp. Keeping neural networks simple by minimizing the description length of the weights. *Proceedings of the sixth annual conference on computational learning theory*, pages 5–13, 1993.

[10] P. Jordan, A. Seed, and E. Weinmann. Errors in radar measurements on rainfall - effect on flood forecasting. 2000.

[11] D. Lowe. The kalman filter. *Lecture notes*, 2003.

[12] D. MacKay. *Information Theory, Inference, and Learning Algorithms.* 2003.

[13] R. Mathurin and B. Rottembourg. A combinatorial approach for rain cell tracking. Twelfth International Conference and Workshops on Applied Geologic Remote Sensing, Nov. 1997.

[14] M.Grecu and W.F. Krajewski. A large-sample investigation of statistical procedures for radar-based short-term quantitative precipitation forecasting. *Journal of hydrology*, 239:69–84, 2000.

[15] M.A Pedder, M. Haile, and A.J. Thorpe. Short period forecasting of catchment-scale precipitation. part 1: the role of numerical weather prediction. *Hydrology & Earth System Sciences*, 4(4):627–233, 2000.

[16] H.S. Wheater, V.S. Isham, D.R. Cox, R.E. Chandler, A. Kakou, P.K. Northrop, L. Oh, C. Onof, and I. Rodriguez-Iturbe. Spatial temporal rainfall fields: modelling and statistical aspects. *Hydrology & Earth System Sciences*, 4(4):581–601, 2000.