New Genetic Programming Methods for Rainfall Prediction and Rainfall Derivatives Pricing

Sam Cramer

School of Computing
University of Kent

This dissertation is submitted for the degree of
Doctor of Philosophy by Research

October 2017

-

Acknowledgement

Firstly, I would like to extend my gratitude to my supervisors Michael Kampouridis and Alex Freitas. They have both given so much of their time and have provided invaluable feedback over the duration of the PhD. Additionally, I would like to extend my thanks to Antonis Alexandridis who was always there to support and offer advice throughout the years. Without this combined help, the completion of this thesis would never have been possible. I am forever grateful for their help.

My appreciation extends to James Brookhouse and Ayah Helal who I had the pleasure of sharing an office with, making the journey very enjoyable. Additionally, all the staff at Medway that has created a lovely working environment and closeness within the department, extending for those that I have had the pleasure of working alongside. I would like to express my thanks for friends made in Canterbury, including An Kang and James Smith for lots of fun times. A special thanks to Claire Carter for treating me like I was one of her students. Finally, I would also like to extend my thanks to my Mum and Dad who continue to support and believe in me throughout my life.

Last but by no means least, I would like to extend a very special thank you to my girlfriend Dr Su Wang. She has been an inspiration to myself since the day I met her, by being there since the beginning of Masters until the end of the PhD and beyond. I have made so many amazing memories at Kent, all because of you. I could never have imagined doing this without you, thank you for always being there for me. I love you.

Abstract

Rainfall derivatives is a part of an umbrella concept of weather derivatives, whereby the underlying weather variable determines the value of derivative, in our case the rainfall. These financial contracts are currently in their infancy as they have started trading on the Chicago Mercantile Exchange (CME) since 2011. Such contracts are very useful for investors or trading firms who wish to hedge against the direct or indirect adverse effects of the rainfall.

The first crucial problem to focus on in this thesis is the prediction of the level of rainfall. In order to predict this, two techniques are routinely used. The first most commonly used approach is Markov chain extended with rainfall prediction. The second approach is Poisson-cluster model. Both techniques have some weakness in their predictive powers for rainfall data. More specifically, a large number of rainfall pathways obtained from these techniques are not representative of future rainfall levels. Additionally, the predictions are heavily influenced by the prior information, leading to future rainfall levels being the average of previously observed values. This motivates us to develop a new algorithm to the problem domain, based on Genetic Programming (GP), to improve the prediction of the underlying variable rainfall. GP is capable of producing white box (interpretable, as opposed to black box) models, which allows us to probe the models produced. Moreover, we can capture nonlinear and unexpected patterns in the data without making any strict assumptions regarding the data.

The daily rainfall data represents some difficulties for GP. The difficulties include the data value being non-negative and discontinuous on the real time line. Moreover, the rainfall data consists of high volatilities and low seasonal time series. This makes the rainfall derivatives much more challenging to deal with than other weather contracts such as temperature or wind. However, GP does not perform well when it is applied directly on the daily rainfall data. We thus propose a data transformation method that improves GP's predictive power. The transformation works by accumulating the daily rainfall amounts into accumulated amounts with a sliding window. To evaluate the performance, we compare the prediction accuracy obtained by GP against the most currently used approach in rainfall derivatives, and six other machine learning algorithms. They are compared on 42 different data sets collected from different cities

across the USA and Europe. We discover that GP is able to predict rainfall more accurately than the most currently used approaches in the literature and comparably to other machine learning methods.

However, we find that the equations generated by GP are not able to take into account the volatilities and extreme periods of wet and dry rainfall. Thus, we propose decomposing the problem of rainfall into 'sub problems' for GP to solve. We decompose the time series of rainfall by creating a partition to represent a selected range of the total rainfall amounts, where each partition is modelled by a separate equation from GP. We use a Genetic Algorithm to assist with the partitioning of data. We find that through the decomposition of the data, we are able to predict the underlying data better than all machine learning benchmark methods. Moreover, GP is able to provide a better representation of the extreme periods in the rainfall time series.

The natural progression is to price rainfall futures contracts from rainfall prediction. Unlike other pricing domains in the trading market, there is no generally recognised pricing framework used within the literature. Much of this is due to weather derivatives (including rainfall derivatives) existing in an incomplete market, where the existing and well-studied pricing methods cannot be directly applied. There are two well-known techniques for pricing, the first is through indifference pricing and the second is through arbitrage free pricing. One of the requirements for pricing is knowing the level of risk or uncertainty that exists within the market. This allows for a contract price free of arbitrage. GP can be used to price derivatives, but the risk cannot be directly estimated. To estimate the risk, we must calculate a density of proposed rainfall values from a single GP equation, in order to calculate the most probable outcome. We propose three methods to achieve the required results. The first is through the procedure of sampling many different equations and extrapolating a density from the best of each generation over multiple runs. The second proposal builds on the first considering contract-specific equations, rather than a single equation explaining all contracts before extrapolating a density. The third method is the proposition of GP evolving and creating a collection of stochastic equations for pricing rainfall derivatives. We find that GP is a suitable method for pricing and both proposed methods are able to produce good pricing results. Our first and second methods are capable of pricing closer to the rainfall futures prices given by the CME. Moreover, we find that our third method reproduces the actual rainfall for the specified period of interest more accurately.

List of Publications Derived from This Thesis

Journal papers

- Cramer, S., Kampouridis, M., Freitas, A. A., Alexandridis, A. K., "Providing Stochastic Solutions for Pricing Rainfall Derivatives", 2017. (working paper)
- Cramer, S., Kampouridis, M., Freitas, A. A., "Decomposition Genetic Programming: An Extensive Evaluation on Rainfall Weather Derivatives", Applied Soft Computing, 2017. (under review)
- Cramer, S., Kampouridis, M., Freitas, A. A., Alexandridis, A. K., "An Extensive Evaluation of Seven Machine Learning Methods for Rainfall Prediction in Weather Derivatives", Expert Systems with Applications, Elsevier, Vol. 85, pp. 169 – 181 (2017).
- Alexandridis, A., Kampouridis, M., Cramer, S., "A Comparison between Wavelet Networks and Genetic Programming in the Context of Temperature Derivatives", International Journal of Forecasting, Elsevier, Vol. 33 (1), pp. 21 – 47 (2017).

Conference papers

- Cramer, S., Kampouridis, M., Freitas, A.A., Alexandridis, A.K., "Pricing Rainfall Based Futures Using Genetic Programming", EvoBafin, EvoStar, Amsterdam, Springer, pp. 17 – 33 (2017).
- Cramer, S., Kampouridis, M., Freitas, A.A., "A Genetic Decomposition Algorithm for Predicting Rainfall within Financial Weather Derivatives", Genetic and Evolutionary Computation Conference (GECCO), Denver, Colorado, USA, ACM, pp. 885 – 892 (2016).
- Cramer, S., Kampouridis, M., Freitas, A.A., Feature Engineering for Improving Financial Derivatives-based Rainfall Prediction, IEEE World Congress on Computational Intelligence, Vancouver, Canada, IEEE, pp. 3483 – 3490 (2016).

- Cramer, S., Kampouridis, M., Freitas, A.A., Alexandridis, A., "Predicting Rainfall in the Context of Rainfall Derivatives Using Genetic Programming", IEEE Computational Intelligence for Financial Engineering & Economics, Symposium Series on Computational Intelligence, Cape Town, South Africa, IEEE, pp. 711 – 718 (2015).

Table of contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Rainfall Derivatives

In this thesis, the main research focus is on the pricing of rainfall derivatives (Benth and Benth, 2012). In order to achieve this, the research is mainly concentrated on rainfall prediction, which directly underpins a derivatives contract's value in the financial market. Once the best possible prediction model for the rainfall can be accurately established, we then are able to price the futures contracts for the interests of both trading individuals as well as trading institutes.

In the research area of weather applications, both estimations and predictions are difficult to measure due to the vast amount of uncertainties. Hence, the uncertainties in the weather contribute towards a considerable amount of risk factor to various individuals, with businesses profit being greatly affected by the state of the weather. There is a lot of exposure to these elements and until 1996 there was no suitable financial protection available to businesses. Previously, one would have to rely on insurance for protection, but it is hard to prove if a business had been adversely affected by the weather. The insurance market can cover extreme weather damages such as hurricanes, but it cannot cover unfavourable fluctuations. This is due to its difficulty in the evaluation of business effect due to damage. Nowadays, individuals or institutions are able to seek for financial protections against the weather risk elements by weather derivatives.

Weather derivatives are a type of financial contract, which can be held between two or more parties. The value of the derivative depends upon the underlying weather variable. Various different types of weather derivatives exist in the market and are commonly traded on temperature, rainfall and wind. Due to the incompleteness of the weather derivative market, trading can be problematic. This incompleteness implies that we are not able to physically hold or store the weather variables, which is required

to price the contracts according to risk-neutral conditions. In the current weather derivatives market, there is no generally accepted pricing framework to value the financial contracts. Our interest lies in financial rainfall derivatives, which only began trading in 2011 on the Chicago Mercantile Exchange (Cabrera et al., 2013).

Rainfall derivatives are less commonly traded compared to other types of weather derivatives in the market, but are just as important, especially for those seeking for protections in agriculture. Rainfall derivatives are a recent addition to the weather derivatives, because of the difficulties in modelling and pricing. If the modelling measure is insufficient or inaccurate, it can lead to large pricing errors. In other words, future rainfall forecasts do not reflect future events of rainfall, which causes a derivative's price to be far away from the true values. The inaccuracy in pricing derivatives increases the volatilities and the uncertainties within the financial market. As a result, this could reduce the prospect of attracting new investors to the market.

In order to price accurately, it is key to predict the underlying variable of rainfall as accurately as possible. The difficulties in future rainfall level predictions lie in very large fluctuations observed in the daily data. Unlike other weather domains, the rainfall time series is highly discontinuous providing with no trends or seasonality. However, there is some relevant literature in the areas of statistics and machine learning; the prediction of rainfall can be addressed through methods in the aforementioned research areas.

Therefore, the two key problems that exist and will be addressed in this thesis are:

- The difficulties in predicting long run daily rainfall.
- The pricing of derivatives in an incomplete market.

1.2 Motivation

The main motivation behind the practice of purchasing rainfall derivatives is the importance and necessity to protect individuals, who may be exposed to the adverse affects of rainfall (Carmona and Diko, 2005). The abnormal rainfall conditions affect individuals as well as businesses globally, in particular those invested in the area of agriculture and natural resources. It is estimated around 30% of the US gross domestic product is directly or indirectly affected by the weather and climate (Allianz, 2013). There are two main motivation branches for this thesis. The first is the academic perspective and the second is the practical perspective of rainfall prediction and pricing rainfall derivatives.

The former is concerned with the lack of available literature in the problem domain of daily rainfall prediction. Without a sufficient modelling technique, the market will

remain illiquid and will deter investors, due to the large uncertainties surrounding rainfall. There exists unique challenges that are present within rainfall data, which should inspire the creation of new algorithms and methodologies. By creating new algorithms and methodologies, the unique patterns in rainfall as a time series problem can be discovered. Hence, they can be used to improve the long run predictive accuracy of rainfall prediction. This then translate to better pricing accuracy and the development of new pricing methodologies.

From a practical perspective, research should encourage the growth of rainfall derivatives. The use of such contracts has real-life benefits. Using agriculture as an example, too much rainfall may cause crops to be destroyed and vice versa. The practical problem is farmers' revenue is heavily dependent on favourable weather conditions. Their needs in terms of rainfall may differ between each commodity to maximise their revenue. Therefore, a flexible contract based on rainfall to protect individuals is very important. The damages caused by the irregularities of rainfall can have a financial impact on the earnings of businesses and individuals, which may filter down to the cost and availability of goods and services provided. This causes a social and economic problem to society from the unfavourable fluctuations of rainfall. Therefore, rainfall derivatives are a key financial instrument to protect individuals from unfavourable circumstances.

1.3 Genetic Programming

Regarding the types of algorithms investigated in this thesis, the core method is called Genetic Programming (GP). GP is a machine learning method inspired by natural evolution, where computer programs act as the individuals of a population. It evolves randomly generated individuals to solve the problem of rainfall prediction, through the Darwinian principal of survival of the fittest. Each individual (model) is assessed on its capability in solving the problem of rainfall prediction, based on the residuals between the predicted and actual outcome of rainfall. Over a given number of generations, GP aims to converge on the 'best' possible individual through the recombination of individuals' genetic material during the process of natural selection. Upon convergence, this 'best' individual or model is returned.

Through the use of GP, we aim to be able to provide a suitable solution to the problem of maximising the accuracy of rainfall predictions, so that it increases the accuracy in pricing weather derivatives. This can be validated against a series of different data sets across a large variety of climates. Within this thesis, we explore the capability GP has for the time series of rainfall, based on its predictive accuracy in

both rainfall amount as well as derivative prices. Additionally, GP has a significant advantage over other popular machine learning algorithms, such as Neural Networks and Support Vector Regression, due to its potential for producing interpretable models, instead of classical black-box models.

Another key aspect is that the framework of GP allows for a very flexible platform for novel extensions to overcome the challenges presented by the time series of rainfall. Additionally, this flexible framework allows for GP to be adapted to the probabilistic nature of the pricing environment.

1.4 Genetic Algorithm

Another type of evolutionary algorithm similar to GP, namely a Genetic Algorithm (GA), is also used within the thesis. It shares many of the same characteristics in its ability to solve a problem through the Darwinian principle. However, an important difference is in the individual representation. In general GA's individuals represent only data (variable values). On the other hand, in general GP's individuals represent both data and functions (e.g., operators). Hence, GA can be seen as evolving a solution to a problem, rather than a computer program. This particular type of evolutionary algorithm is used to assist GP in the prediction of rainfall. More precisely, in this thesis a GA can be used to indicate when to expect varying levels of rainfall, in order to help GP make the most accurate prediction.

1.5 Scope of Thesis

In this thesis, we cover a range of different literatures from different academic fields, namely: Computer Science, Statistics, Finance and broadly within a Meteorological application. This thesis focuses on popular machine learning methods from Computer Science and put emphasis on the generation of new algorithms for the problem of rainfall. Within Statistics, this thesis covers the two main methodologies used for rainfall prediction, which have been applied to a vast number of hydrological applications. Also, we consider Bayesian inference and Monte Carlo simulations for the estimation of futures prices. From Finance, we cover the pricing methodologies used within rainfall derivatives and focus our development of new algorithms to fit within the landscape of finance.

1.6 Goals and Original Contributions of Thesis

The main goal of this thesis is to develop a GP method that is capable of providing accurate futures prices of rainfall. To achieve this goal, it is crucial to create an algorithm which specifically focuses on predicting rainfall. We aim to provide a method using the framework of GP to maximise the predictive performance and outperform other machine learning methods, including the state-of-the-art used within the area of rainfall derivatives. In order to validate the performance of the proposed GP, we require rigorous testing by analysing model performance across a variety of performance measures.

Within this thesis we outline eight original contributions for the prediction and pricing of rainfall derivatives through new GP methods, which are summarised below:

- The superior predictive performance of machine learning methods over the currently used methodologies in rainfall derivatives. (Chapter 4)
- A new GP tailored for the problem of rainfall derivatives. (Chapter 4)
- A data transformation technique to address the issues of the underlying data. (Chapter 4)
- A new GP method called Decomposition GP (DGP) using a GA to create subproblems for the problem of rainfall prediction. (Chapter 5)
- Three new algorithms for the problem of pricing rainfall derivatives. The first two use Monte Carlo Markov chain to assist DGP to produce a probabilistic output, with one focussing on contract-specific equations and the other focussing on a single equation for all contracts. The third is a new GP method for producing stochastic equations (SMGP). (Chapter 6)
- Provided more accurate rainfall futures prices than listed by the CME using SMGP. (Chapter 6)
- A thorough comparison of all proposed GPs' predictive errors against six other machine learning methods on daily and transformed data. (Chapters 4-6)
- A thorough analysis through alternative model performance measures of all proposed GP algorithms and the other machine learning benchmarks, based on climatic indicators and the coverage of all algorithms. (Chapters 4-6)

1.7 Thesis Overview

The thesis is laid out as follows. In Chapter 2, we provide an overview of the rainfall predictions methods used within rainfall derivatives, namely Markov chain extended with rainfall prediction and Poisson-cluster. Also, an overview of popular machine learning methods: Neural Networks and Support Vector Regression for rainfall prediction. Finally, we give a brief background in rainfall derivatives pricing and highlight the two pricing methods, namely indifference pricing and arbitrage free pricing.

In Chapter 3, we focus on our chosen machine learning method of Genetic Programming (GP). We begin by giving a thorough overview of the algorithm and outlining the motivation for using GP. We present a selection of popular interpretations of GP that have been used for time series problems. We also provide a literature review where each has been applied within the finance and climatic problem domains. Finally, we present the similarities and differences between the GP and Genetic Algorithms (GA).

In Chapter 4, we propose a tailored GP to the problem of rainfall derivatives. We begin by presenting the data to be used throughout all experimentations. Through the analysis of the data, we propose a data transformation to assist GP. We then present our proposed GP for this chapter, which is tailored for the problem of rainfall derivatives, and introduce the six other benchmark methods used in this thesis. The results of all algorithms are compared based on their predictive error and other model measures.

In Chapter 5, we propose a new GP based on decomposing the problem of rainfall prediction. We begin by outlining how we plan to decompose the rainfall time series into subproblems, along with the modifications required for the GP. As part of the GP, we require a classification algorithm to assist in the decomposition of rainfall prediction. We propose a GA for this task and discuss how to integrate the GA and GP together to form Decomposition GP (DGP) in a hybrid relationship. We present the results of the DGP against the benchmark methods from Chapter 4 on the same evaluation measures, and evaluate the use of other classification algorithms.

In Chapter 6, we propose three novel ways of calculating rainfall derivative prices. We begin by reintroducing rainfall derivatives pricing using the arbitrage free pricing approach. We then propose the use of Monte Carlo Markov chain (MCMC) to allow DGP to price according to the existing pricing methods. We propose two variations for pricing, DGP with MCMC and DGP focussing on contract-specific equations with MCMC. Our last proposal is a new GP that is capable of producing and evolving stochastic equations without the need for MCMC, called Stochastic Model GP (SMGP). We present the results of our three proposed methods based on the predictive error for rainfall prediction and for rainfall derivatives pricing.

Finally in Chapter 7, we conclude and provide future research opportunities.

Chapter 2

Background of Pricing Within Rainfall Derivatives

2.1 Introduction

The ability to price rainfall derivatives relies heavily on predicting the level of rainfall as accurately as possible to minimise problems of mispricing (Alexandridis and Zapranis, 2013; Jewson et al., 2010). In this chapter we outline the relevant literature surrounding predicting the level of rainfall used within rainfall derivatives. Since the techniques outlined have not come directly from the rainfall derivatives field, we provide a brief literature review of the relevant approaches used for the daily prediction of rainfall. They are Markov chain extended with rainfall prediction and Poisson-cluster model (Section 2.2). For completeness we also provide a short review on some machine learning methods that have been previously used for rainfall prediction in other problem domains (Section 2.3).

After establishing the rainfall methods, we then give a brief introduction to the problem of pricing within rainfall derivatives and outline how the rainfall amounts are used to calculate the derivative price (Section 2.4). As this is a very recent field, there exists only two approaches for calculating the market value of a derivative. The first is difference pricing (Section 2.5) and the second is the arbitrage free pricing (Section 2.6). In both sections we provide a short overview of the approach with a short literature review of its use within rainfall derivatives. We finally summarise the key points in the conclusions (Section 2.7).

2.2 Rainfall Prediction Within Rainfall Derivatives

Within rainfall derivatives, and similar to other derivatives pricing, the index modelling technique is often adapted. This way of pricing requires predicting the underlying variable of the derivative to generate a futures contract price at the risk-neutral level. Within rainfall derivatives this requires an algorithm to predict future levels of rainfall. There exists two main types of models, the first is Markov chain extended with rainfall prediction (MCRP) and the second is the Poisson-cluster model. Other methods do exist for rainfall prediction, but we focus on the two methods that have been used within rainfall derivatives.

2.2.1 Markov Chain Extended With Rainfall Prediction

We present an overview of the method, before we consider the literature that is specific to rainfall derivatives for Markov chain extended with rainfall prediction.

Overview

The Markov-chain extended with rainfall prediction (MCRP) is a commonly used method in explaining the process of rainfall since its proposal in Gabriel and Neumann (1962). In their paper, a simple Markov chain is used to model rainfall as a stochastic process, by calculating the occurrence as a probability whether rainfall will occur at Tel Aviv. A simple or first-order Markov chain is the probability of rain on a given day based on whether the previous day rainfall occurred or not. It was not until Todorovic and Woolhiser (1975) that added the rainfall prediction part to MCRP, where the rainfall amount is modelled by some distribution. The process of rainfall using MCRP can be seen as two separate parts.

The occurrence process has not changed over the years, except for considering more rainfall states (Haan et al., 1976) or a higher order Markov chain (Chin, 1977; Stern, 1980; Wilks, 1999). A higher order Markov chain uses more previous days to calculate the occurrence pattern. Additionally, Stern and Coe (1984); Wilks and Wilby (1999) noted that a hybrid approach by considering a higher order chain for wet or dry chains is more appropriate. More recently Schoof and Pryor (2008) proposed that for each month a different order of Markov chain would be more appropriate. It is well documented throughout these papers (amongst others work) that each climate (data set) requires its own order of Markov chain. For example, Wilks (1999) shows that in drier areas with a greater level of seasonality a higher order is preferred. However, in wetter low seasonal areas a lower order is preferred. His findings were supported based on a large sample of weather stations across the U.S.A.

The proposed level of rainfall is achieved through a random sampling via a distribution capturing the rainfall amounts for nonzero (wet) days. The exponential distribution is the first distribution to be noted (Richardson, 1981; Taewichit et al., 2013; Todorovic and Woolhiser, 1975), because it is a distribution controlled by one parameter. Additionally, the two main distributions used are gamma distribution (Buishand, 1977) and mixed exponential distribution (Roldán and Woolhiser, 1982). Both are extensively applied for the simulation of rainfall amounts (Suhaila et al., 2011; Taewichit et al., 2013; Wan et al., 2005; Wilks, 1999) and have shown to match the different climates more effectively, given that the gamma and mixed-exponential have two and three parameters controlling them respectively. More information will be provided in Chapter 4.

Application in Rainfall Derivatives

The first work to propose a model for rainfall derivatives was by Cao et al. (2004) using a first order Markov chain for the occurrence of rain. The amount process was modelled by the mixed exponential, gamma distribution and kernel distribution. The kernel distribution provided the best fit after the simulations of rainfall based on the data from Chicago. Further applications include Odening et al. (2007), which also applied a first order Markov chain with the amount modelled by a mixed exponential distribution; this model had time varying parameters for the occurrence and amount. Their approach was modelled on rainfall data based in north-east Germany at Berlin-Tempelhof. Stowasser (2011) also applied a Markov chain to the data of Colorado, a first order Markov chain was used as a benchmark for the occurrence of rainfall, whilst the rainfall amount was modelled via the gamma distribution. They also considered the approach by Schoof and Pryor (2008) to determine which order of Markov chain is the most appropriate for a given month in the year. They additionally tested a second order and third order Markov chain for comparison as well. By using the Bayesian information criterion, one can calculate the optimal order of Markov chain to use in a given month. The results found that over the year the order varied between a first order, second order and third order. The results showed that the varying order Markov chain throughout the year performed the best, with the third order performing the worst. Ritter et al. (2014) applied a one state Markov chain to two areas in Germany, Koßdorf and Nordhausen. They considered a multi-site approach, as they built their model with nearby weather stations to improve the modelling accuracy. Lastly, the work by Cabrera et al. (2013) applied a first order Markov chain, but modelled their approach on daily data from New York City, Detroit and Jacksonville. They used the mixed exponential distribution to model the amount.

2.2.2 Poisson-Cluster Model

We present an overview of the method, before we consider the literature that is specific to rainfall derivatives for the Poisson-cluster model.

Overview

The Poisson-cluster model is seen to be a closer representation to the meteorological approach by taking into account both time and space, instead of time typical in other models. This process is called a spatial-temporal model. It was first suggested by Le Cam (1961), however, it was not until the work of Waymire et al. (1984) that the model started to gain popularity. The model is based on explaining a storm at a specific point in space and time, which consists of many different smaller storms, with each storm consisting of different levels of intensities. Additionally, each storm has a decaying nature, where it appears and dissipates over a random period of time. Waymire and Gupta (981a,b,c) identified that storms arrive according to a Poisson distribution and that rainfall is represented by smaller clusters of rainfall cells scattered around the storm's central point.

Rodriguez-Iturbe et al. (1987, 1988, 1984) studied various different approaches, mainly the Poisson based model, Neyman-Scott white-noise model and the Bartlett-Lewis rectangular pulse model. One problem outlined by Cox and Isham (1988) is that the Poisson based model struggles to see effects on a range of timescales. Even though there exists a strong theory developed by Cox and Isham (1988, 1994), these types of models are very difficult to apply to real life data, because of the parameter estimation problem and the availability of necessary data (Sanso and Guenni, 1999). One issue noted by many literature is that the Poisson-cluster is not effective for extreme weather scenarios, and a separate model should be used to account for such scenarios. Verhoest et al. (2010) identifies that the Bartlett-Lewis models exhibit irregular behaviour and produce unrealistic rainfall cells. However, it is noted that the Barlett-Lewis process shows promise for generating long-run series rainfall simulations.

Application in Rainfall Derivatives

The first application of the Poisson-cluster model is by Carmona and Diko (2005), which is built on the model by Rodriguez-Iturbe et al. (1988). Their model includes the maximum likelihood estimation (MLE) to find optimal values for their parameters. The disadvantage of the model proposed by Rodriguez-Iturbe et al. (1988) is not robust when fitting its parameters. This model is extended to a Markovian process by implementing a random jump function that increases at the cell arrival rate and

decreases at the cell extinction rate. Additionally, a smoothing feature is implemented on the rate of extinction based on the rainfall intensities rather than the number of active cells. This model with the modifications is referred as a Markov jump model, which is fitted to Norwegian data. One of the issues with this approach is that the model is unable to adequately capture and take into account seasonal variation of occurrence and amounts of rainfall. In some climates where the climate exhibits significant seasonality then the model may be insufficient. This is noted by Leobacher and Ngare (2011), who measures rainfall with a Markovian gamma model. Their approach splits a year into 12 different sections, each representing one month and directly model the amount based on the gamma distribution for each individual month using MLE. This approach is technically incorrect as one property of MLE is that the data is independent, which is not the case for their model. In order to overcome this, a correlation value between each month is required. They model the rainfall in Kenya. Benth and Benth (2012) use a slight variation using the independent increment model. Noven et al. (2015) further develop the work by Carmona and Diko (2005) by incorporating a generalisation of an integrated Ornstein-Uhlenbeck process to a continuous-time autoregressive moving average process. This particular type of model allows for additional flexibility of the autocorrelation structure, increasing the fitting accuracy. Hung et al. (2009a) extends this model to incorporate weather forecasts through a filtration method, to increase accuracy nearer windows. No numeric example was given, only theoretical.

2.2.3 Limitations of Existing Methods

From outlining the literature for both methods MCRP and Poisson-cluster, there is a common drawback. In both cases, the formulation of the model is heavily dependent on historical data reflecting future rainfall values, due to the methods being simulation rather than predictive. In all the literature, there is no mention of the predictive performance of the rainfall process, but only considers how well the model fits and its capability in replicating past information. One key assumption in all the work described, is the existence of an annual seasonal pattern for rainfall. This leads to a common problem where the expected rainfall amount is similar to the historical mean for each time point. This type of behaviour is acceptable if rainfall is highly seasonal with a clear reoccurring pattern. However, this is not the case, which causes the parameters explaining the model to be prone to issues of underfitting. Moreover, through the simulation of the stochastic process, there exists a large number of unrealistic rainfall pathways, suggesting the model to be predictively weak. This

disadvantage raised is key, as potential mispricing is more likely with a predictively weak model (Alexandridis and Zapranis, 2013; Jewson et al., 2010).

When comparing the two approaches of MCRP and Poisson-cluster, the model fitting capability of the Poisson-cluster is superior, since the model is much higher in complexity. However, this is not necessarily equivalent to a better predictive accuracy and we find that both perform similarly to each other. MCRP carries an advantage in this respect as the complexity is minimal while being just as effective.

In this thesis, we shift the focus away from simulation based methods, towards predictive techniques in the field of machine learning. We opt for Genetic Programming, because it is able to provide a white-box nonlinear model that helps explain rainfall without settling on the historical mean. Moreover, we are able to overcome the disadvantage of the literature by using a predictive method for predicting rainfall, which should avoid issues of mispricing. For this thesis we opt for MCRP as our chosen benchmark, not only because it is the most commonly used approach, but also because the performance is similar to Poisson-cluster whilst being a more simple and comprehensive approach.

2.3 Machine Learning Methods for Rainfall Prediction

Machine learning methods have not been used within the context of rainfall derivatives, but have been widely used for various applications of rainfall prediction. The literature on the long-run predictive accuracy on daily rainfall is quite light and is mostly done on monthly amounts or short forecasts of only a few days. We present a range of works that have applied Artificial Neural Network (ANN) and Support Vector Regression (SVR) in hourly and daily predictions. These two methods are included as they are considered the state-of-the-art in the machine learning field and are the most reoccurring algorithms other than the statistical methods of MCRP and Poisson-cluster model, in the area of rainfall prediction.

2.3.1 Neural Networks

Hall et al. (1999), applies a neural network to the prediction of rainfall by proposing a two step model for short run predictions. The first step is to predict the probability that rain would occur in a given season and then the second step is to predict the level of rainfall of a given day. Yuval and Hsieh (2003) applies a neural network with the use of model output statistics, which is a statistical relationship between the output of a numerical weather prediction model and observations. Similar to Hall et al. (1999), this is another short run prediction for precipitation, which has a benefit for flood control.

Essentially, the neural network is involved to help make the statistical connection and thus improve the performance. Hung et al. (2009b) applies a backpropagation ANN (BPANN) to the rainfall prediction in Bangkok. The findings from this paper show that just using previous days' rainfall is not sufficient enough and is highly inaccurate. However, predictive performance can be improved by including weather variables other than rainfall, such as: humidity, wet bulb temperature, air pressure, cloud cover, average rainfall across all stations and rainfall at surrounding station. This approach is again for the short-term rainfall, given their forecasts were only for 1-3 hours ahead. Mislan et al. (2015) applies a BPANN for the monthly amounts in Indonesia providing sufficient forecasts for the future monthly amounts. However, it appears their model was misspecified as negative rainfall values are generated.

Weerasinghe et al. (2010) applies a feed-forward back-propagation neural network for the daily prediction of rainfall in Sri Lanka. Their approach is inspired by the statistical models previously discussed, by classifying the results first. They construct a second model that splits the occurrence into four types of rainfall (trace, light, moderate and heavy). Additionally, their approach includes a multi-site approach as well as three nearby stations. Moustris et al. (2011) applies a neural network for forecasting monthly mean amounts of rainfall in Greece and find that their model is able to forecast reasonably accurately for four months.

2.3.2 Support Vector Regression

Support vector regression (SVR) is also used across rainfall prediction, but the majority of the literature is focused on the monthly prediction since Tripathi et al. (2006) first applies SVR to the problem domain. Lin et al. (2009) applies SVR for the prediction of typhoons (extreme rainfall events) based on hourly data. The results show that SVR can successfully outperform BPANN in providing longer range forecasting of up to 6 hours and provide a more robust framework.

Wu et al. (2010) applies SVR to the problem of daily rainfall in China, comparing against a set of different neural network set-ups. Findings suggest that SVR leads to positive results, using a particle swarm optimisation algorithm for parameter searching. Kisi and Cimen (2012) applies SVR to the daily rainfall to the same data sets in Kisi and Shiri (2011) (Turkey). However, one issue that underpins both works is that minimal experimentation was conducted with only 1 and 2 data sets used respectively in the daily SVR works. Similar to Kisi and Shiri (2011), daily rainfall does increase in accuracy from the use of wavelet-transformation. However, it should be noted that the use of wavelet-transform appears to increase predictive accuracy, the results are for two weather stations, and they are unable to validate whether it actually contributes to

an average gain. This is a common problem that exists across the machine learning literature for rainfall prediction, since some papers merely apply a technique without a rigorous experimental procedure.

From the above literature review, rainfall prediction methods are relatively weak for the prediction of daily rainfall. The models often provide an unrealistic solution, although this is not noted within the work. The approaches from the statistical side provide a better solution, which is more realistic. However, these methods are predictively weak.

2.4 Background to Pricing

The general method for pricing a derivative contract for the rainfall amount is given by:

$$F(t;\tau_1,\tau_2) = E^Q[I(\tau_1,\tau_2)|f_t] = E^Q\left[\sum_{\tau=\tau_1}^{\tau_2} R_T|f_t\right] \tag{2.1}$$

where $F(t;,\tau_1,\tau_2)$ represents a futures contract priced at time point t for a contract length from the beginning of the contract period τ_1 till the last day of the contract period τ_2. For this, t does not have to equal to τ_1, because contracts are priced for a future date. $E^Q[I(\tau_1,\tau_2)|f_t]$ represents the index I of the rainfall amount over the contract period τ_1 till τ_2, given the available data at time point f_t (f_t can be referred to as information set (finance literature) or likelihood (statistics literature)). This index level is calculated at the risk-neutral expectation denoted by E^Q. This gives us the final part of the equation that is the sum of the total rainfall (R_T) over the contract period given the available historical data that we have under risk-neutral conditions. As the rainfall index is explicitly used in the formulation of a derivatives price, the prediction of the underlying variable of rainfall is required. Please note that Q (risk-neutral measure) does not have anything to do with the objective probability of occurrence of scenarios, i.e. the probability of a certain rainfall prediction pathway from happening. Q in our case is a probability measure on the set of scenarios, which is a *bet* on the occurrence of this event. In other words, we are trying to measure the probability of us betting on the occurrence of this outcome, rather than the probability of the outcome.

Rainfall derivatives is an incomplete market, as rainfall amounts do not have a price, nor can they be held or traded. Therefore, one cannot assume *arbitrage-free* pricing (there exists the opportunity for risk-free profit), as a result pricing directly on the accumulated amount of rainfall is considered risky. Because of this, additional methods (e.g., Esscher transformation and indifference pricing) are required to transform rainfall amounts from the real world to the risk-neutral world. Therefore, the rainfall amount

is directed towards the more likely scenario in order to achieve neutrality. Another way of looking at it, is finding the expectation of the index that has been calculated and then what is the probability for the index to take that value. Arbitrage pricing and risk-neutrality mentioned above are key concepts, which need to be addressed within derivative pricing. The absence of arbitrage imposes constraints on the way derivatives are priced within a market. Risk-neutrality allows the price of any derivative within an arbitrage-free market to discount the expected payoff under an approximated probability measure called "risk-neutral" measure.

Our rainfall estimates $I(\tau_1, \tau_2)$ are considered the expected price under the canonical measure P (i.e., the probability space (Ω, f, P)), but are within the 'risky' world. Therefore, we require $Q \sim P$, such that all tradable assets in the market are martingales after discounting taking into account investors' exposure to risk. A martingale is a sequence of a random variable, such as a stochastic process (in our case, MCRP), where at a particular time in the realised sequence (i.e. a given day in one of our rainfall pathways), the expectation of the future value (i.e. the following day) is equal to the present observed value. The expectation is also conditioning on the given knowledge of all prior observed values. To establish the risk preferences of investors require the market price of risk (MPR), which is the additional return or risk premium expected by investors for being exposed to undertaking the futures contract. Within complete markets, where the modelled quantity is tradable, the MPR does not explicitly feature in the formulation of the price. This is because, investors are able to hedge away the risk in any position by dynamically buying and selling the underlying asset, allowing the equivalent martingale measure of Q to be calculated. It is crucial to derive the equivalent martingale measure, which verifies that there is arbitrage-free pricing. Therefore, we must specify the risk-neutral probability of Q. The weather derivatives are traded in an incomplete market, there will exist many different martingales (Q). Hence, it is not possible to find a unique risk-neutral measure Q (Benth and Benth, 2012; Jenson and Nielsen, 1996), such that Q is equivalent to the physical measure P. As mentioned previously, Q is the betting on the outcome of P. Therefore, the derivative price is arbitrage-free, if and only if there exists a probability measure $Q \sim P$, such that the derivative payoffs are martingales with respect to Q. For this reason Q is an equivalent martingale measure. The Black-Scholes model, the first and most well known pricing models achieves its equivalent martingale measure by modifying the drift in the Brownian motion.

Another issue with rainfall derivatives is that rainfall is not a tradable asset, which cannot held, traded or stored. In other words, it is not possible to construct a portfolio that can be perfectly hedged (has a replication strategy), i.e. a way to completely protect an individual financially containing a rainfall derivative, where the individual

neither gains nor loses money. Subsequently, it is not possible to find a unique risk-measure, or unique equivalent martingale $Q \sim P$. Instead, many different martingales exist and only prices can be derived directly based on the basis of no-arbitrage. Due to this reason, we are looking to estimate Q_θ, where theta is the MPR, a parameter for finding the unique equivalent martingale.

There are two main approaches for approximating the unique (a generalisation of many) equivalent martingale and to find the MPR, which is the indifference pricing and arbitrage free pricing. We cannot use Brownian motion like in the Black-Scholes pricing model for three reasons. Firstly, rainfall is a binary event with extremely volatile peaks making the data non smooth. Secondly, there is no mean-reverting value, i.e. there is no seasonal mean. Thirdly, rainfall is strictly non-negative and does allow for an unbounded random walk.

2.5 Indifference Pricing

We give a general overview of indifference pricing before referring to the relevant literature surrounding indifference pricing applied to rainfall derivatives.

2.5.1 Overview

We aim to find the MPR to calculate the risk-neutral probability based on a buyer and seller within a market. The method consists of finding a price of a derivative so that the buyer and the seller are indifferent and a fair price (market value of the derivative) are calculated. Thus, we are looking for a unique equivalent martingale. This leads to two sets of equations to be calculated and estimated. The first equation, to maximise the price at which the seller will be willing to sell at and the second equation, to minimise the price that the buyer will be willing to buy at, given the risk aversion parameter of MPR. Thus, we are looking to maximise the utility of both parties. The indifference pricing technique (Carmona, 2009) is based on the standard approach or is extended to utility indifference pricing. This framework is popular before the rainfall derivatives started trading on the exchange, hence is used for pricing options and forwards (over-the-counter derivatives) rather than futures.

The idea behind indifference pricing as mentioned above is that two individuals exists, one buyer and one seller. Both of these individuals are trying to maximise their wealth at some time point in the future. It is also assumed that no trading is allowed between the two individuals. The market itself consists of 2 different assets, one riskless, which is typically a savings account and one risky assets, which is in a related market (one associated with rainfall risk). At the same time contracts

in rainfall derivatives can be sold at a given price. The wealth criteria which is trying to be maximised is typically based on a utility function, usually the negative exponential utility function. There are two scenarios that we need to consider, when the buyer/seller has or does not have any rainfall derivative contracts within their strategy. The indifference price is obtained when the seller/buyer is indifferent between which of the strategies to consider, whether it is the one with rainfall derivatives or one without. Monte Carlo simulations are required to estimate the optimal point based on the rainfall predictions.

For a full description we refer the interested reader to Carmona (2009).

2.5.2 Application in Rainfall Derivatives

In rainfall derivatives, this method is first proposed by Carmona and Diko (2005), who prices a call option based on the data of Bergen, Norway. They assume that the buyer and seller both have an exponential utility function. The pricing is achieved by using electricity forward contracts traded on NordPool (Nordic power exchange). Odening et al. (2007) applies the indifference pricing for two put options for wheat in Brandenburg, which is located forty kilometers away from the measuring station. For the geographic risk present, they capture this by the normal distribution. They find that their prices remained fairly constant, given that there is a small amount of correlation between wheat yield and rainfall sum caused by having high geographic risk. Leobacher and Ngare (2011) apply their model in Dagoretti, Kenya. Their findings show that there is a relative decrease in the price of the call option, due to hedging with an electricity contract.

2.6 Arbitrage Free Pricing

In this section we provide an overview to the pricing technique of arbitrage free pricing, before outlining the relevant literature within rainfall derivatives that have applied the approach. This approach is the current pricing method used within rainfall derivatives (Benth and Benth, 2012; Cabrera et al., 2013; Noven et al., 2015) and will be the chosen method for pricing within this thesis.

2.6.1 Overview

The arbitrage free pricing approach uses the Esscher transform (Esscher, 1932) (synonymous to exponential tilting), which is a generalisation of the Girsanov transform for Brownian processes. The Esscher transform can be seen as a method to change the

index value, whilst in most cases retaining the original probability density function. Numerous distribution functions can be used to achieve this shift as part of the Esscher transform, as long as they are within the exponential distribution family (Esscher, 1932). Therefore, there is a greater choice available and can fit a distribution that is more suitable to the problem. The use of the Esscher transform changes the jump intensity and jump size under P to the new probability Q_θ. Therefore, achieving risk-neutral and arbitrage free pricing from the predicted rainfall amounts. Gerber and Shiu (1995) generalise the transformation to a stochastic process driven by a Lévy process and is applied across a variety of different pricing applications (Bühlmann et al., 1998; Gerber and Shiu, 1995; Kremer, 1982).

The Esscher transform works by changing the probability density $f(x)$ of a random variable X (in our case a probability density of all rainfall pathways based on the accumulated rainfall amount for a given period) to a new probability density $f(x;\theta)$ with parameter θ denoting the MPR, given by:

$$f(x;\theta) = \frac{\exp(\theta x) f(x)}{\int_{-\infty}^{\infty} \exp(\theta x) f(x) dx}. \tag{2.2}$$

Here we see the Radon-Nikodym derivative with θ being the level of risk exposed to investors from the jumps of the driving process of rainfall. The Esscher transform reflects the corresponding risk by exponentially tilting the jump measure shown by Equation 2.2 through θ. Many distributions can be used as mentioned before, those applied within the literature come from the exponential family. They are: Bernoulli, Binomial, Normal, Poisson and Normal Inverse Gamma (NIG) distributions. All of these distributions can take the θ into consideration. The next step is to fit one of the chosen distributions to the $f(x)$, the most common one is the NIG, which has 4 parameters to tune μ for the location, β the skewness, σ the scaling and α for the steepness. Other than the good fit, from having 4 parameters, using the NIG$(\alpha,\beta,\mu,\sigma)$ benefits from the distribution maintaining its shape (Esche and Schweizer, 2005) under the Esscher transform with parameter θ becoming NIG$(\alpha,\beta+\theta,\mu,\sigma)$.

Theoretical prices under Q_θ can be estimated by taking the mean value of the sampled index (MPR = 0) or of the transformed outcome (MPR $\neq$ 0) with a given MPR, which can be negative or positive. This can be assumed and picked arbitrarily at first (constant over time), but it would be wise to consider the value changing over time to deal with different time periods. Having the MPR calibrated with the real market data would go towards finding the most appropriate MPR and hence calculate over time the risk present to investors. Once the MPR has been chosen accordingly, then the unique equivalent martingale is found and prices can then be derived using the formula given at the beginning.

2.6.2 Application in Rainfall Derivatives

Benth and Benth (2012) propose the arbitrage free approach for the problem of rainfall derivatives, but does not apply it to generate any prices. This is closely linked with Cabrera et al. (2013), who uses the framework of Benth and Benth (2012) to apply a range of distributions to the output from MCRP. Based on maximising the Kolmogorov-Smirnov test, they find that the NIG distribution is the most suitable distribution. They apply the arbitrage free pricing method to price rainfall futures at the Chicago Mercantile Exchange for three cities in the United States of America, namely Detroit, Jacksonville and New York. This is the first work of pricing real futures prices. Noven et al. (2015) follows up the work using the Poisson-cluster model to apply to the rainfall futures prices at Detroit. The findings suggest that both models are suitable for pricing at Detroit, but results indicate that the Poisson-cluster fitted the data better. In terms of pricing performance, both are very similar.

2.7 Conclusion

In this chapter, we present the relevant literature surrounding rainfall derivatives for the prediction of rainfall and literature surrounding daily rainfall prediction with machine learning methods. The literature surrounding daily rainfall is quite light with Markov chain extended with rainfall prediction (MCRP) being the most prevailing within rainfall derivatives. MCRP will be used as the benchmark within this thesis. For completeness we also included a short review on the literature of machine learning methods for daily rainfall and use these methods throughout the thesis as a benchmark as well. Finally, we moved on to provide an overview of the pricing procedures that have been used within rainfall derivatives, after producing future rainfall amounts. Within this thesis, we focus our attention on the second approach of arbitrage free pricing, since this is the method that has been used since derivatives started trading. The next chapter focuses on our proposed machine learning method of Genetic Programming as an alternative to the statistical (simulation based) methods outlined in this chapter, which will be used to predict rainfall and price derivatives.

Chapter 3

Background of Genetic Programming

3.1 Introduction

Genetic Programming (GP) is our chosen methodology for this thesis and we provide a literature review of the relevant areas, including a comprehensive overview of GP itself. Within this chapter we give an overview of GP (Section 3.2) by introducing the components that make up the algorithm, then provide our motivation for using this algorithm. As GP can take many forms through different interpretations, we provide an overview of a handful of popular and relevant types of GP within the literature (Section 3.3). Following this discussion on the different interpretations of GP, we provide the reasoning for deciding upon our chosen type of GP (Section 3.5). We also analyse the differences between GP and Genetic Algorithms (GA), which will be used to assist GP in the prediction of rainfall (Section 3.6). This analysis is necessary since we use a GA later in this thesis. Finally, we bring together all the information to conclude (Section 3.7).

3.2 Overview of Genetic Programming

Genetic Programming (GP) is a type of evolutionary algorithm that is inspired by Darwin's theory of evolution, (survival of the fittest) which is used to evolve programs to perform a certain task (Koza, 1992). Essentially, GP tries to find the best solution to a given problem by selecting the best performing individuals from a population of programs based on its fitness landscape. After selecting the best performing individuals, they are combined with each other to produce better solutions to the problem. Over time, solutions should get better and better until an optimal solution is found. In some cases, it is not possible or is unknown if an optimum solution is reached. In this case GP returns a near-optimal solution.

The methodology behind GP has several steps. The first step is to create a random population of individuals (candidate solutions) according to the terminals and functions that are appropriate to the problem domain. Terminals relate to the variables and constants that do not take arguments as inputs, whilst functions are used for processing the values that they receive as input. Examples of what we mean by terminals and constants are given later. Once the population has been created, the fitness of each individual is measured against a pre-defined fitness function. This fitness value is how we can objectively compare each individual against one another at how well they are able to solve the problem. For example, if modelling a time series-based problem, then using Mean Squared Error might be an appropriate metric to use. After a fitness is applied to all individuals, a given number of individuals are chosen to produce new offspring, using a selection procedure based on each individual's fitness. Different selection methods exist and will be discussed later. The individuals chosen from the population are manipulated by genetic operators to produce new offspring, such as crossover and mutation. The new offspring then becomes a part of the next generation's population, where each individual in this new population is assigned a fitness value based on the pre-defined fitness function. The procedure then continues until a termination point has been achieved, which tends to be after a pre-defined number of generations. Once this stopping criterion is met then the best performing individual, i.e. the fittest individual is returned as the result.

There are four main phases for GP, these are:

- Initialise the population.
- Evaluate each individual within the population against a pre-defined fitness function.
- Select the most fit individuals to create new offspring.
- Use genetic operators to form the next generation (new population).

For the last three steps, this procedure is then repeated a number of times until a stopping criterion has been met.

3.2.1 Representation of Individuals

The individual representation is how an individual encodes a candidate solution. The most common way to express the genotype of an individual is to use a tree-based representation made up of a set of functions and terminals (Koza, 1992). This is just one type of representation and others do exist, but are much less common in the literature. Such alternative representations can be a linear representation (Banzhaf,

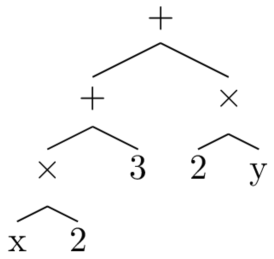


Fig. 3.1 An example individual as represented via a tree-based GP for the formula $2x+3+2y$.

1993; Cramer, 1985), a graph representation (Teller and Veloso, 1995) or an extension of graph, or a Cartesian representation (Miller, 1999; Miller and Thomson, 2000). Figure 3.1 shows an example individual using a tree-based GP, for the mathematical formula $2x+3+2y$.

3.2.2 Initialisation of the Population

Each individual's function to solving the problem specified is represented by a choice of terminals and functions. A terminal set consists of the inputs and constants, which are specific to solving the problem. For example, terminals may consist of numbers (e.g., 1, 2, 3) and letters (e.g., x, y, z) for arithmetic problems, or monthly inflation values for financial time series problems. The function set is also dependent upon the problem domain to connect the terminals together to meet the problems output. For example, for an arithmetic or mathematic problem the function set may consist of arithmetic operators (e.g., $\div$, $\times$, +, -) and functions (e.g., log, exp, pow). Other types of function sets could include that of boolean operators (e.g., AND, OR, NOT), conditional operators (e.g., $<$, $>$, ==) or any other specific functions for a given problem. A random choice of terminals and functions for each individual are used to create the initial population. The importance of such approach is to represent the search space sufficiently. By producing individuals in this manner, there is a substantial level of diversity across all individuals. Thus, encouraging the evolutionary process to generate increasingly better solutions.

In order to produce an individual there must some structure to them. Three approaches that are widely adopted exist for the generation of individuals: Full, Grow and Ramped Half-and-Half (Koza, 1992).

The Full approach first selects a root node randomly from the given set of functions. The other internal nodes of the tree are also made up of other functions. This process continues until one level before the maximum depth of the trees, where the terminals are then randomly selected to form the leaves of the tree. This approach gives a

balanced and symmetrical tree, whilst guaranteeing that each tree will have a full depth. An example is shown in Figure 3.2.

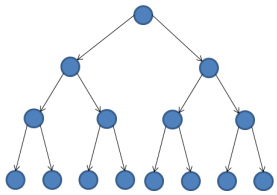


Fig. 3.2 An example tree showing the Full initialisation of an individual.

The Grow approach first randomly selects a root node from either the function set or terminal set. If the root is a function, then the child nodes are filled with random functions or terminals. This then recursively happens for each depth in the tree structure. This process happens until either all branches end with terminals or the maximum depth has been reached. This approach will produce trees with a wide variety of shapes. As shown in Figure 3.3.

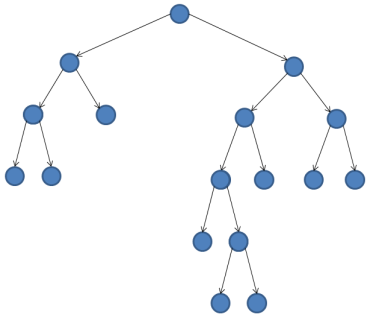


Fig. 3.3 An example tree showing the Grow initialisation of an individual.

The Ramped Half-and-Half approach combines the Full and Grow approaches to produce trees of different depths. For every depth from two to the maximum initial depth, half of the individuals of the population will use the full approach and have balanced trees with all the paths having the same length. The remaining individuals of the population will use the grow approach and have unbalanced trees with paths of different sizes. This method allows for a diverse population and will achieve a balanced distribution of individuals with irregular and balanced trees.

3.2.3 Genetic Operators

Once individuals have been chosen they need to be processed to produce offspring, or individuals for the next generation's population. There are two main genetic operators

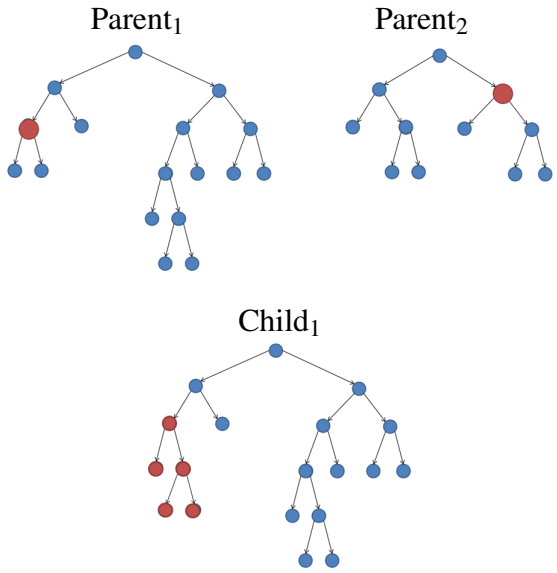


Fig. 3.4 This figure shows two parents with a randomly chosen crossover point (shown in red on both parents) and the resulting offspring from combining parent$_2$ with parent$_1$ to produce child$_1$. The red subtree on child$_1$ shows parent$_2$'s genetic material.

for doing so: crossover and mutation (Koza, 1992). Crossover creates an offspring by mixing parts from two individuals (parents) together. Once two parents have been chosen (based on fitness value) a node from each parent is selected at random. The point randomly selected is called the cross-point. Therefore, each parent is broken up into two parts, the first is from the root node to the cross-point and the second is the subtree whose root node is the cross-point. Performing this on both parents returns four different parts and so two offspring can be created. The first offspring consists of the root node up to the cross-point from the first parent and the cross-point with the remaining subtree from the second parent. The second offspring consists of the root node up to the cross-point from the second parent and the cross-point with the remaining subtree from the first parent. As is shown in Figure 3.4.

The mutation approach is different to the crossover approach, because only one parent is required to create a new offspring. The process begins by randomly selecting a node within the parent. Once a node has been selected the node and the subtree rooted at that node is removed. Once removed, a random subtree is generated from the available function and terminal sets and is inserted at the position of the randomly chosen node. Once the new subtree has been inserted then a new offspring is created. As is shown in Figure 3.5.

There are many ways of performing crossover and mutation, but we choose to focus on the most well-known variations within this section. For a review of other

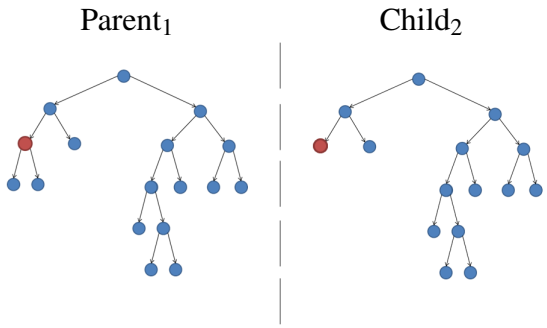


Fig. 3.5 This figure shows a mutation occurring on parent$_1$ to produce an offspring (child$_2$), where the red point shows a randomly chosen mutation point.

types of crossover and mutation, please see Poli et al. (2008). There also exists the genetic operator of reproduction that puts an identical copy into the next generation.

3.2.4 Breeding Methods

These are strategies that involve which individuals are to be chosen to be mated and, determine the genetic operator that will be applied to a pair. The two main criteria to distinguish between different breeding methods are:

- Mating restrictions between the individuals.
- Types of genetic operators that can be used to create offspring.
- The probability of a certain genetic operator being used.

For the first point, the breeding can either be generational or steady state (Rogers and Prügel-Bennett, 1999). For generational breeding, the individuals mate and produce offspring from the selected parents and are put into an intermediate population, this will avoid an overlapping between the generations. Afterwards, the new offspring that are in the intermediate population will then become the new population. Steady state is an alternative approach, which differs by continuously adding new offspring to the current population. Whenever a new individual is added to the population, a member already in the population is removed to avoid the population from forever increasing.

Additionally, the mating can be restricted in some way. Examples of breeding strategies include panmictic (Goldberg, 1989), demic sub-populations (Langdon, 1995) and pygmy (Ryan, 1994). Panmictic is the most common throughout the literature and this allows all the individuals to mate with one another. In demic sub-populations, the population is divided into smaller demes and then mating is only allowed to occur

between individuals within the same deme. Pygmy divides the population into two main groups and mating is only allowed from two individuals in a different group.

The last two points are determining which genetic operators to use, which are determined by a given probability. The main objective is to determine whether one parent is used, i.e. for mutation and reproduction, or whether two parents are used for crossover. The main breeding method that is present in the literature is Koza's crossover combined with mutation. Koza (1992) did not use mutation in his experiments. In later experiments, mutation has been shown to be beneficial as it helps diversify the population. The probability assigned to these genetic operators is an important aspect and needs to be chosen according to the problem. Typically, it would be unwise to have too high a mutation rate, because this could make the GP too random. Furthermore, a crossover percentage too low will be dominated by the reproduction of individuals, thus, decreasing the diversity of the population.

3.2.5 Selection Methods

An important requirement is selecting fit individuals and applying genetic operators to evolve the population into the next generation. The main determining factor is the individual's fitness value and how it compares with another individual in the population, where fit individuals have comparably higher fitness levels. The roulette wheel is one strategy that could be used to select the best individuals. Under the roulette wheel selection method, a probability of selection is assigned to an individual, which is directly proportional to its fitness value. Another type of selection strategy is tournament selection, where a specified number of individuals are sampled at random from the population. The sampled individuals then compete against one another based on their fitness level. The individual with the highest fitness is chosen to act as a parent. For a complete review on selection methods, please see Shukla et al. (2015).

3.2.6 Reasons for Using Genetic Programming

GP is just one method that can be used for the pricing of rainfall derivatives. There are various applications of GP across different classification and regression problems; with several extensions to the traditional framework outlined above. Regardless of the extensions and the applications, GP has key advantages and strengths that make this an appropriate and well suited method for the pricing of rainfall derivatives. Firstly, GP is able to produce white-box models, which allows users to easily examine and understand the models produced. This allows users to generate a selection of models, which can be compared; with the most appropriate model selected for the problem.

Within this thesis, we use the benefit of white-box models to assist in the development of new techniques to maximise the predictive performance of GP. Secondly, rainfall and derivative pricing is a nonlinear data series, which can be modelled via GP, given that GP is a nonlinear modelling technique. Finally, the key advantage about using GP over other methods such as Auto Regressive Moving Average (ARMA) is that GP assumes nothing about the structure of the problem. Therefore, GP provides a flexible framework to address regression based problems.

3.3 Genetic Programming Approaches

3.3.1 Traditional Genetic Programming

Traditional Genetic Programming (TGP) is the approach propelled by Koza (1992), previously discussed. We outline some of the weaknesses of TGP. Firstly, with TGP there is the issue with the combination of trees, when two parents produce offspring. When two parents create an offspring there is a chance that the recombination is not meaningful, thus does not aid search performance. Luke and Spector (1998) showed through experimental evidence that in TGP the conventional sub-tree crossover is only slightly better than a mutation approach. The problem is less from the way sub-tree crossover works, but more from the way the trees are represented. By exchanging randomly chosen sub-trees from the parents there is no real structure and no logical testing for the crossover to be recombined. A potential situation is combining two parents that have different shape, size and functionality, which will produce a child carrying little similarity to either of its parents. The parents are assumed to have a good fitness, but by drastically changing the functionality of the new child, the child is likely to have a low fitness.

GP is a tool that has to be able to both express appropriate programs and evolve appropriate programs. Both of these issues are related to each other and both need to be addressed. For GP to be successful, it must sufficiently explore the search space to find an appropriate region and then generate a suitable population that can sufficiently evolve programs in the promising region. As TGP by itself does not restrict parts of the problem's search space from being explored, it suffers greatly from the lack of expressiveness. This creates a problem, if the problem's search space is too large, TGP may spend a long time exploring a non-promising region, thus hindering the evolutionary process. Additionally, the lack of expressiveness can lead to illegal candidate solutions being generated, because no attempt to limit the problem's search space has been performed. Preventions can be included, but for complicated and hard problems where different data types are required to formulate an answer (e.g. vectors,

matrices and boolean operators). It is much harder to deter TGP from generating illegal candidate solutions. Moreover, additional computation time is required to check whether candidate solutions are valid. Furthermore, the illegal candidate solutions with lack of expressiveness can cause a problem. As a result, it does not satisfy closure.

GP requires a desirable situation of closure, which is essential for GP to work. Closure is required so that any function can handle any valid inputs and handle them accordingly without any data type or evaluation conflicts. In general, a data type error will only occur if the problem has more than one data type, for example numeric and boolean. The types must be kept consistent, otherwise illegal candidate solutions can be formulated. For example, combining numeric and boolean operators may return an inconsistent type. Similarly, evaluation conflicts may occur, for example when dividing by zero, or the square root of a negative number. Exceptions can be included, such as returning 1 if dividing by zero. These exceptions need to be specified in order for closure to be met. Depending upon the problem, it may be hard for TGP to achieve complete closure, or it may be computationally inefficient for checking validity by specifying too many rules.

Applications in Financial and Climatic Time Series

Many different applications have used TGP, which will be briefly discussed here. We are interested in the symbolic regression applications from both the financial area and also in the prediction of weather variables. These two areas are of particular interest, given the topic of this thesis explores both topic areas. A range of applications exist in the weather domain such as rainfall-runoff modelling. Khu et al. (2001) applied a TGP using the standard genetic operators with tournament selection, whilst using a maximum depth to prevent individuals from growing excessively large. The fitness function that was used was RMSE. Results show that the approach suitably predicted the arrival of storms, slightly better than a specific rainfall-runoff model. Nasseri et al. (2011) applied TGP to forecasting water demand. A similar procedure was followed, but the selection method was different. Lexictour was used instead, which is similar to tournament except if two individuals share the same fitness level then the one with fewer nodes is selected. One addition was the use of sensitive analysis to simplify the inputs to the model.

Falco et al. (2005) successfully applied TGP to the prediction of El Niño. Their approach was similar to other TGP applications, but instead measured their accuracy over a validation set instead of just a testing set to avoid the potential problem of over-fitting. Similarly, Arganis et al. (2009) applied TGP to water temperature forecasting by the modelling of hourly, daily and weekly climatic data. TGP derived a suitable

equation for the quick calculation for future water temperature. They standardised their data first, before applying it to GP. Additionally, they restricted the number of nodes on a particular individual to a maximum of 30 nodes, instead of fixing a depth.

In other meteorological applications, GP is applied to wind power forecasting. Arshad et al. (2014) and Zameer et al. (2017) used predictions from Artificial Neural Networks (ANN) along with the basic data to provide ensemble forecasts for wind power and found TGP to provide positive results. Within temperature forecasting, Seo et al. (2013) proposed TGP for short-range predictions of temperature of up to three days and found their GP to provide superior results to the Unified Model. Ramesh et al. (2015) predicted the surface air temperature in India comparing against ANN and found that TGP's predictive accuracy was inferior to ANN across all data sets.

Agapitos et al. (2012) and Alexandridis and Kampouridis (2013) both applied TGP to the forecasting of temperature to be used for the pricing of weather derivatives. Their approaches differed, but ultimately were calculating the total heating degree days (HDD's). Agapitos et al. (2012) considered the monthly average temperatures and the HDD values as inputs to their various different models. Whereas, Alexandridis and Kampouridis (2013) modelled the daily average temperature, before calculating the HDD's. Both approaches claimed that TGP performs well for the modelling of temperature. Alexandridis et al. (2017) provided a more thorough analysis of their previous paper, but only performed the analysis on the daily temperatures.

TGP is also applied in different financial settings including the work of Wagner and Michalewicz (2001). Their approach looked at modelling U.S. Stock Exchange, but they used a slightly different population control approach than the one normally used in TGP. Their approach involved a two tier soft node limit and hard node limit; this was to relax and for population control to be a bit more flexible. This is in contrast to a static population limit. If the addition of a new individual to a population exceeded the soft node limit, then that individual would be added to the population, but no more individuals after this. The hard node limit is the absolute limit of total nodes in a population. If an individual breaks this, then the tree must be trimmed to fit within the limit. Wagner et al. (2007) later proposed a tailored GP named Dynamic Forecasting (DyFor) GP that was specifically aimed at non-static environments, with a dynamically changing time series. The application field was for forecasting the gross domestic product and the consumer price index of inflation. Therefore, the model can be updated when different past-learned information is present and can build on already learned knowledge to increase convergence. Their approach was compared against an AR(1) model and a well established method by Kitchen and Monaco (2003).

More recent applications of TGP for regression including Hsu et al. (2015), who forecasted the prices of Taiwan Stock Exchange Capitalization Weighted Stock Index

(TAIEX) options. They applied a GP and compared against options prices generated from Support Vector Regression (SVR) and Black-Scholes model. The results indicated that GP was able to price options at TAIEX better than both benchmarks. Yin et al. (2015) applied GP to predict the implied volatility comparing against a heterogeneous autoregressive model. Results indicated that TGP was able to use a variety of different predictors to more accurately forecast future levels of volatility. Dabhi and Chaudhary (2016) proposed a hybrid wavelet and TGP approach and applied it on the stock prices of Intel and Microsoft. Results indicated that the combined usage of TGP and wavelets was able to provide satisfactory results, they compared their method against the GP given in the ECJ toolbox.

The literature shows that for certain problems, TGP is adequate and performs well. All of the above applications are relatively easy to fit a GP to and it may be possible to see better improvements to the models proposed. Regardless of the issues surrounding TGP, TGP is able to give a benchmark in terms of assessing the performance against other GPs.

3.3.2 Strongly-Typed Genetic Programming

Overview

Strongly-typed Genetic Programming (STGP) was the first major extension to the TGP, which was pioneered by Montana (1995). This approach incorporates types and their constraints into the GP. Moreover, in this approach every terminal has a type and every function has a type for its arguments and for the return value as well. When the programs are generated either by being initialised or after genetic operators have been applied, the new programs are evaluated without violating the type constraints specified by the system. This restricts what can be evolved as only some symbols can match the type required in a given part of the tree. For example, having an if statement that will return an either true or false. If the crossover were to occur on the if statement, then the only type that can take its place is another relational operator, such as greater than. This ensures that the correct type has been used.

Advances to Strongly-typed Genetic Programming

Since Montana (1995) first introduced STGP, there are some extensions to the original work. Haynes et al. (1996) introduced the concept of type inheritance allowing for polymorphism. STGP only allows one abstract class, but in this extension multiple abstract classes were allowed. Once an argument is instantiated to a specific type then the remaining arguments are also typed to that instantiation. However, their

approach could only support subtyping for a limited single inheritance type hierarchy. This was further extended by Yu (2001), by creating PolyGP, allowing for infinite number of types and polymorphism was extended to function type. Castle and Johnson (2012) more recently proposed Strongly Formed Genetic Programming (SFGP), which extends Montana (1995) by adding in additional structural constraints. The extension requires that all non-terminals must also define both a data type and a node type, whereas STGP only imposed a data type rule. The node-type constrains to what terminal or non-terminal the child node should take, which when evaluated will return a value of the specified data type. This method allows the trees to grow in a more constrained shape and structured way, which is a more simplified idea than Montana's (1995). One issue though with this approach is that through the genetic operators the resulting children may be very similar, given the excessive constraints on the trees.

Applications in Financial Time Series

Zumbach et al. (2002) applied STGP for the discovery of new volatility models based on foreign exchange rates. In foreign exchange rate applications, there is an induced exact symmetry by the exchange of two currencies. The method that is used is an extension on the original STGP by imposing an exact symmetry on the generated programs, thus imposing a well defined representation of the solution. Therefore, STGP keeps track of the parity of the GP trees and reduces the search space from all candidate solutions to those having exact symmetry. Specific genetic operators were used and implemented in order to obey the syntactic restrictions. For a full coverage of the specific operators we refer the reader to Zumbach et al. (2002). Additionally, a local optimiser technique was used to optimise the constants, since a random search is insufficient in finding the best constant to swap with. The performance gain from implementing these was significant; the added constraints improves GP convergence, whilst only correct symmetry property individuals are generated. The GP was also very robust in terms of its predictive performance, whilst producing trees with little complexity.

Manahov et al. (2014) used a similar approach with a dynamic fitness function, where STGP was compared against K nearest neighbour and AR-GARCH (Autoregressive with Generalized Autoregressive Conditional Heteroskedasticity). The method outperformed the traditional and most commonly used econometric forecasting models. The approach was also able to make a profit on its trading even after taking into account transaction costs. Manahov et al. (2015) then applied their STGP across multiple stocks to produce return strategies for investors, showing significant gains over a number of short-term runs. Their results identified that there is an emergent

behaviour of "low intelligence traders", whereby they set their STGP to incorporate various levels of intelligence to simulate an investor.

Despite STGP having issues with structural constraints, the applications outlined go a long way to solving those problems. The downside is the added complexity required in forming and handling the constraints. However, as shown the improvements are significant and STGP can be applied to time series problems.

3.4 Other Types of Genetic Programming Used For Time Series

Many other types of GP exist in practice and those previously presented highlight two of the main ones currently used. However, we will mention and discuss some of the other types of GP that have been used in both the climatic application and financial application for time series. We include these for completeness, as a lot of work has been undertaken. This is by no means an exhaustive list, as there exist many other types not outlined here.

3.4.1 Grammar-based Genetic Programming

Overview

Grammar-based Genetic Programming comes in a variety of forms, but its initial development was by Whigham (1995). The constraints of the problem are given via grammars to satisfy the property of closure. Grammar in this context is a way of expressing the kinds of constraints to specify the syntax of the language. The grammar also plays a part in helping to guide the genetic operators, but as to what is evolved depends upon the particular type of grammar rules. Whigham (1996) developed a Context Free Grammar GP (CFGGP), which evolves derivation trees. Derivation trees is a hierarchical representation to the rules that are to be applied and in which order to obtain a given program. Closure is achieved by only swapping sub-trees with respect to the grammar. Whigham extended his previous GP (Whigham, 1996) allowing a user to include bias into the search, so that certain parts of the grammar are more likely to be evolved; allowing for better evolved programs.

Other popular types of Grammar-based GP include Grammatical Evolution (GE) proposed by Ryan et al. (1998), which is based on Backus Naur Form (BNF). GE is a linear representation instead of the tree based representation as previously mentioned. In GE, the evolution does not occur on the program, but instead on a variable-length binary string (usually integers). This approach uses a robust mapping technique called

wrapping, inspired by the overlapping genes phenomenon allowing the re-use of the same genetic material in the expression of different genes (O'Neill and Ryan, 2003). A comprehensive overview of grammar-based GPs can be found in Brabazon et al. (2015); Mckay et al. (2010).

Applications in Climatic Time Series

Grammar-based GP are applied across various different problem domains. Whigham and Crapper (2001) used CFGGP to model rainfall-runoff. The approach uses the traditional genetic operators, with crossover producing two children, whilst a hill climber is used for mutation of real numbers. The grammar for this model was biased towards forming equations composed of a linear component and a nonlinear component. The results were compared against a model used for rainfall-runoff that required previous knowledge relating to the area, both performed equally well. However, when the area was unknown, CFGGP was able to reap the benefits of not requiring any prior knowledge to the data or situation. Babovic and Keijzer (2000) applied the dimensionally aware GP to a stream flow model. The results were compared against a standard GP and their GP was found to be significantly better at handling the task, by successfully restricting the space. They found that by restricting the space and making the GP aware of the dimensions, results tended to be more compact, which is preferred for understanding. The method also takes a much shorter time to converge compared to TGP. This can be explained by TGP blindly trying to meet their goodness-of-fit criteria. Donne et al. (2014) proposed a grammar GP to the problem domain of wave height, which can be very important to climatic studies. The results were compared against an ANN and did perform well on the respective problem.

3.4.2 Semantic Genetic Programming

Overview

Semantic Genetic Programming (SGP) is a more recent advance in GP by the use of semantic-aware techniques. Semantics essentially means which symbols should be used at the syntax level to match the inputs to the outputs (Krawiec and Pawlak, 2013). The use of semantics can create a unique set of individuals that can suitably explore the search space. Moreover, it is able to make better use of genetic operators and create children that are representative of the parents to aid exploration of the search space. There are numerous ways that can be used to incorporate semantics into GP, the three main areas (Vanneschi et al., 2014a) are diversity methods (Beadle and Johnson,

2009), indirect semantic methods and direct semantic methods. Diversity methods are concerned with increasing the semantic diversity of individuals to represent the search area over time, instead of stagnating. Indirect semantic methods are concerned with acting on the syntax of the individuals to indirectly promote a semantic behaviour. Direct semantic methods are concerned with the direct effect on the semantics by using semantic aware genetic operators.

Time Series Applications of Geometric Semantic Genetic Programming

Vanneschi et al. (2014b) applied their approach of Geometric Semantic GP (GSGP) (Moraglio et al., 2012) to four areas, three in pharmacokinetics and one regression problem of electrical power demand. Results were compared against TGP and found that GSGP, using geometric semantic crossover and mutation outperformed TGP in all instances. Castelli et al. (2015) applied GSGP to the same problem of energy consumption including the same data set as Vanneschi et al. (2014b). Their approach included a local search technique to greatly speed up the search process, whilst minimising the error compared to other set-ups of GP.

3.4.3 Multi-Objective Genetic Programming

Overview

Multi-Objective Genetic Programming (MOGP) looks to optimise more than one fitness function, by finding an optimal solution for each fitness function simultaneously, which can be handled in two ways (Konak et al., 2006). Either by defining a composite function or using a Pareto dominance-based approach (Zhou et al., 2011). To achieve the former, methods such as weighted sum are used to weight the composite function. This approach can yield various different favourable solutions by changing the weights, which could be better for decision making. However, this method is less suited to real-world problems, because it can be very difficult to accurately select the weights. Additionally, this approach has a major disadvantage of being computationally expensive, requiring one run of GP for each setting of the weights.

The latter is to determine an entire Pareto optimal solution set, a non-dominated set of solutions with respect to each other. It achieves the goal by finding trade-offs between the fitness functions, which are more appropriate to real-life scenarios as there is usually a trade-off. One issue is the scalability and that the Pareto set can become excessively large with an increasing number of fitness functions (Vamplew et al., 2008). Additionally, this approach has the problem of selecting the best solution to be used, out of all non-dominated solutions returned by MOGP.

Applications in Financial and Climatic Time Series

Vazquez (2001), used residual variance and long-term prediction error as their two fitness functions using the GP-NARMAX model for meteorological data. Their model was both compact and predictively strong. Babovic and Keijzer (2002) applied a MOGP for rainfall-runoff using a dimensionally aware GP including a coefficient of determination as a secondary objective. They were interested in real-time forecasting to keep a constant forecasting horizon, a form of data assimilation. GP is then able to recalculate errors as it progresses through time. Borrelli et al. (2006) tested a MOGP for time series analysis for the Milan stock exchange with the objective functions based on statistical features (mean, standard deviation, skewness, kurtosis). More recently, Zerenner et al. (2016) applied MOGP for the downscaling of near-surface temperature fields to provide more realistic high-resolution accounts of the atmospherical data. Their MOGP consisted of three criteria which were measuring different types of accuracies along with a fourth criterion of measuring the size of solutions. Results indicated their MOGP was able to find a sufficient balance of complexity and performance, whilst maintaining a competitive performance to the relevant benchmark.

3.4.4 Gene Expression Programming

Overview

Gene Expression Programming (GEP), proposed by Ferreira (2001) is encoded as a linear representation, which is decoded into expression trees instead of the more common tree representation of GP. GEP allows syntactically correct programs regardless of any modifications by having a functional genotype/phenotype relationship. The genotype refers to a fixed length linear chromosome, which is composed of one or more genes of equal length. The phenotype is the expression tree representing a candidate solution. This type of genotype-phenotype system allows for complex tree structures that can vary in size, shape and composition despite being encoded as a fixed length gene. The genetic operators are more similar to those in Genetic Algorithms than those in GP, because of the linear representation. GEP has three different types of crossover: one-point, two-point and gene recombination. One-point recombination is similar to TGP, whereas in two-point recombination, the parents are spliced in two sections before recombining. Gene recombination involves swapping a randomly chosen gene between two parents, but they must have occupied the same position in the parent genotype. Mutation is the same, but is subject to the head and

tail constraint. Transposition is a different genetic operator that is sometimes used, which involves swapping material around within the same chromosome.

Applications in Financial and Climatic Time Series

Azamathulla et al. (2011) applied GEP for modelling the relationship between the total discharge of water at a given surface elevation on a stream (referred to as stage-discharge). Results were compared against GP, artificial neural network and stage rating curve. GP and GEP outperformed the others in terms of correlation, predicted against observed. However, GEP was able to predict significantly better than GP when the discharges were higher than normal. These higher than normal discharges are a result of extreme weather conditions. Kisi and Shiri (2011) applied GEP to daily rainfall and found that GEP was not capable of predicting the level of rainfall very accurately on their data set. To improve the predictive accuracy, they used wavelets for GEP to build equations. Garg et al. (2013) proposed a multigene GP approach to select the best GP model and data transformation methods on the stock indexed on the New York Stock Exchange. Although this approach is not explicitly GEP, it is very similar in the sense that it uses multiple genes; the major difference is that multigene uses the traditional genetic operators and is a tree representation.

3.5 Chosen Method of Genetic Programming

For this thesis, Strongly-Typed GP is chosen as the main type of GP. This type of GP is chosen because of the structural advantages that can be derived from the use of data typing. This allows a clear structure to be manipulated in order to derive accurate pricing of rainfall derivatives. More importantly, STGP allows for future extensions that can handle different numeric variables, weather variables and weather stations. This allows us to meet the objective of obtaining closure for our problem domain. Moreover, STGP is successfully implemented in financial areas and is shown to be a very powerful technique, outperforming other state-of-the-art approaches.

We choose STGP over grammar-based GPs due to the possibility of over constraining the search space. Therefore, potentially better solutions may never be explored due to the limitations of the grammar. This requires a specific grammar for each problem, whereby the problem becomes more about creating a suitable grammar. Semantic GP (SGP) is critiqued by Pawlak (2016) for being overly complex, with better results coming from much simpler algorithms. The issue highlighted is that fundamentally SGP is still the same as the syntactic types of GP through a linear combination of random parts. Therefore, the process of finding a solution may be superior to TGP, but

it does so at a large computational, interpretability and complexity cost. Although the results are based on toy examples, it does show that the SGP is not quite semantically aware yet.

MOGP is not our preferred choice, because our problem domain only has a single objective function. Moreover, the framework requires a sacrifice in an objective function in order to meet the criteria of other objectives. Therefore, an optimal value of the fitness function is not usually found. Finally, despite already applied to the problem of rainfall, the main problem with GEP is that the genetic operators are highly destructive and cause unnecessary complexity. Thus, causing very erratic changes to the phenotype, without keeping the structural information during the evolution process.

3.6 Comparison Between Genetic Algorithms and Genetic Programming

So far we have discussed GP, but there is a similar approach of Genetic Algorithms (GA) within evolutionary algorithms that should also be mentioned, especially as we use this approach later in Chapter 5. They share a lot of the similarities in terms of the process for evolution based on the Darwinian principle of survival of the fittest. The similarities include the fixed or variable-length representation of an individual, the representation of an individual and the genetic operators.

Typically GA algorithms are considered to have fixed-length individuals, and GP is considered to have variable length individuals, but this is not necessarily true. Variable length GAs are used (Srikanth et al., 1995). In some sense, fixed length GPs are common by imposing maximum depths or maximum number of nodes, also considering maximum memory for computation. Therefore, a GP is not completely variable length, as a maximum is imposed and is only variable up to a point. The representation is essentially the same between a GA and GP, as a GP can be expressed as a bit string similar to a GA. Although, genetic operators modify the structure of a tree, this can be interpreted at the bit level, as a modification of the bit string. Linear GP and GEP are cases where typical genetic operators that are used within GA are constructed for a GP. Finally, the effect from crossover can be equivalent when material is exchanged between two parents. Within GA, the material usually is kept at the same location of a bit string when creating a new offspring and with GP the material is usually placed in a random location. As long as the structure is maintained, there is no reason why this has to be the case and genetic operators can be constructed to allow for this similar behaviour. An important difference in individual representation, is that

in general GA individuals represent only data (variable values). On the other hand, in general GP individuals represent both data and functions (e.g., operators).

Despite the similarities, there is a conceptual difference between GA and GP. The difference is on how the representation of an individual is interpreted (Woodward, 2003). The differences are outlined in greater detail within Woodward (2003), but are summarised here. The difference stems from how the problem can be represented. GP is concerned with evolving computer programs to carry out a task, rather than a predefined solution. Optimising a function requires representing the numbers from input to output, typically evolving numeric numbers within a GA. It entails a one-to-one mapping, whereby a known function can easily be optimised by a single numeric value. Within GP, this is not usually the case. Since with a given function and terminal set, many solutions can be given to represent the problem, hence many-to-one mapping. However in both cases, GA and GP are finding a solution to a problem which optimises a common fitness function over a set of test cases. In certain problem domains, there may exist many possible solutions, hence a many-to-one mapping.

GA have been widely studied and are used in a variety of different applications, including rainfall (Şen and Öztopal, 2001; Haidar and Verma, 2016; Nhita and Adiwijaya, 2013). Moreover, it is applied extensively in finance as reviewed within Aguilar-Rivera et al. (2015).

3.7 Conclusion

Within this chapter, we provide an overview of Genetic Programming (GP) along with a discussion regarding the different interpretations of GP, along with the relevant literature that exists. This analysis allows us to decide on Strongly-typed GP (STGP) as our chosen GP methodology. This ensures the ability to provide a suitable syntactic structure allowing for legal trees to be generated, which contain multiple different data types. Finally, as we also use a Genetic Algorithm later in this thesis, we discuss the similarities and differences that exist between itself and GP. In the next chapter we begin to develop our novel STGP for the problem of rainfall derivatives.

Chapter 4

Predicting Rainfall Using Genetic Programming

4.1 Introduction

In our previous two chapters, we have presented a range of literature from our chosen methodology of Genetic Programming (GP) and have discussed the typical rainfall prediction methods. As previously mentioned, the goal of pricing can be achieved through predicting rainfall. There are two main models given within the literature. The first, which is more commonly used across all application fields is the Markov chain extended with rainfall prediction (MCRP) (Wilks, 1998) and the second is the Poisson-cluster model (Rodriguez-Iturbe et al., 1987). Within our review we identified that there are issues with both approaches, where both are predictively weak, caused by both methods being overly reliant on previous information being reflective on future predictions. This stems from the notion of simulation rather than prediction, leading to the mean of past years being more prominent, rather than learning and predicting based on more recent rainfall information.

In order to overcome the issues with simulation, we proposed the use of GP as an algorithm for predicting future rainfall levels. Through the benefits of GP outlined in the previous chapter, we aim to provide a prediction driven rainfall equation to forecast future rainfall amounts. Using GP, we have the advantage of not having to specify a model or assume any prior information in our construction of how rainfall behaves. The use of GP allows the problem of rainfall to be tackled by the long term and recent changes in rainfall behaviour.

Our target is to price rainfall derivatives, which requires the underlying variable of rainfall to be predicted as accurately as possible. Using algorithms that cannot minimise predictive error leads to potential mispricing and unnecessary risk. This not

only deters investors, but affects the markets' ability to price a derivative accurately, and free from arbitrage opportunities. As raised in the previous chapter, Cao and Wei (2004) notes that the chaotic nature of rainfall is one of the reasons why rainfall derivatives are unattractive compared to temperature. Daily rainfall carries some unique characteristics unlike other time series data sets. The key issues surrounding rainfall data is that the time series is highly volatile and highly discontinuous. Through the use of GP, our general goal is to minimise the error in rainfall prediction, whilst tackling the fundamental issues that persist within the data. The consideration of both aspects will aim to provide satisfactory predictive equations that can deal with the structure of rainfall time series.

Within this chapter we aim to create a new GP that is tailored towards the problem of rainfall. In order to predict rainfall, we consider GP predicting based on the daily rainfall data. In the consideration of previous work by Kisi and Shiri (2011), there is a suggestion that GP may not be able to predict the daily rainfall data based on the results provided. However, we will investigate why GP did not perform well on the daily data of rainfall. We are interested in daily data because derivatives for weather are updated on a daily basis to reflect the extra days worth of information. Upon our analysis, we can examine why daily prediction leads to poor results as suggested within Kisi and Shiri (2011).

As an alternative to daily prediction, we also consider a data transformation technique to help cope with the key issues surrounding the rainfall time series. As previously discussed, daily rainfall is a very hard time series to model and exhibits characteristics that may be hard to predict. Therefore, by transforming the data we aim to create a new landscape for our GP to predict on. The transformation proposed has to keep the same pricing behaviour (capable of daily updating) as if using daily data to price. We explore the use of a sliding window as a suitable technique to accumulated the rainfall amounts. Therefore, our new proposed landscape is based directly on the accumulated rainfall amounts that make up a contract. Through using the sliding window, we can specify the number of days to accumulate and then shift the transformation across time, thus capable of updating on a daily basis. Moreover, by directly predicting the accumulated amounts we are more intuitively solving the problem, by working with data that is consistent with the final step (accumulation of daily rainfall for a given period) for pricing.

This chapter is laid out as follows. In Section 4.2 we outline the data to be used for all of our experimentation within this thesis, including an in depth analysis of the data and climate. In Section 4.3 we propose a new data transformation for the problem of rainfall derivatives, by considering the accumulation of daily values through a sliding window technique. In Section 4.4 we outline the proposed GP for this chapter as our

initial experimentation and introduce the most commonly used approach of MCRP along with other machine learning benchmarks. In Section 4.5 we outline the method for tuning which will be consistently used throughout the whole thesis whenever we propose a new algorithmic change. In Section 4.6 we outline the experimental set-up for this chapter and outline the goals and objectives we will address in the results. In Section 4.7 we show the results for all algorithms and evaluate the effectiveness of our proposed data transformation and our proposed GP. Additionally, we also take into account any considerations for climate. Finally in Section 4.8 we summarise the chapter.

4.2 Data Used in the Experiments

Within this section, we will introduce the cities used as part of the experimentation for all chapters within this thesis. We will first identify the chosen cities along with the reasons for choosing them. Secondly, we will analyse the time series of each data set. Through the analysis we hope to better understand and identify potential problems that we may face.

4.2.1 Choice of Data Sets

Throughout this thesis, we use the same daily rainfall data sets, which are from specific weather stations across multiple cities in Europe and the USA. In total, we choose 42 different cities, and access the data from NOAA NCNC[1]. We choose these two distinct areas based on prior knowledge: the USA and Europe are well known for having contrasting climatic patterns. Additionally, the USA is specifically chosen as the rainfall derivatives are traded for ten cities, Chicago, Dallas, Des Moines, Detroit, Jacksonville, Kansas, Los Angeles, New York, Portland and Raleigh.

Other than the ten cities outlined, we choose twelve other cities in the USA and twenty within Europe. The remaining cities are chosen based on two criteria. The first criterion is having different types of climates, or climatic features such as very dry to very wet. Secondly, the cities are chosen based on being geographically different. The first point is key to maintain a sense of generalisation to our GP's performance, as the ten cities listed have very different climates. Therefore, by exposing GP to many different climates, we have more information from analysing results whether certain climates are more challenging. If true, then future extensions to our GP are required to cope with the difficulties discovered.

[1] https://www.ncdc.noaa.gov/

For the second criterion, cities within the same area are likely to have a similar climatic effect, unless there is a geographic reason, for example a mountainous region. By considering a spread of cities across the USA that spans around 3,000 miles, instead of focusing in one area, we can capture a range of different climates by choosing vast distances apart. The selection is even more crucial since the USA is generally more exposed as it is mainly flat except for the Rocky Mountains, which runs up from New Mexico through to British Columbia in Canada. Typically, there are three different types of climates across the USA, from the very dry areas in the south-west, to the more persistent areas in the north-east. The central areas have a mix of both, caused by the formation and location of the Rocky Mountains.

In contrast, Europe has more prominent mountainous areas, including the Pyrenees, the Alps and the Balkans, which has a large impact on the climate of cities. This creates a barrier for weather systems, creating more microclimates within smaller geographic regions. For Europe, by considering cities from Spain across Europe through to Slovenia, we can be confident that a good range of climates can be observed.

By having a large variation of cities in our experimentation, this will not bias our results to one particular type of climate, or geographic region. This will allow for our GP to be evaluated not just in terms of its traditional error measurement, but according to its ability to predict under various different climatic effects. As previously mentioned, we will be able to create an algorithm that is capable of working well on numerous different climates. With regards to pricing this type of analysis is critical, as more cities can be adopted with the reassurance that pricing accuracy will not be affected, due to climatic considerations.

The daily rainfall data used throughout the thesis is summarised in Table 4.1. We include a total of twenty cities from around Europe and twenty two from the USA. We include the city name and the country or state that the city is located within to give a sense of geographic location.

4.2.2 Analysis of Daily Rainfall

We perform analysis of our daily data to understand our data and provide potential solutions to further improve the predictive accuracy of our GP. To analyse, we propose a set of descriptive statistics to try and understand the key aspects relating to the discontinuity, volatility and irregularity of rainfall. Therefore, we consider the balance between wet and dry days, the spell lengths for both wet and dry periods, the volatility of daily rainfall over the year, the average rainfall on an annual scale, the average level of rainfall intensities and the variation around the average intensities. We describe the nature of the data given in Tables 4.2 and 4.3.

Table 4.1 The list of all cities whose daily rainfall amounts will be used for experiments.

City	State	City	Country
Akron	Ohio	Amsterdam	Netherlands
Atlanta	Georgia	Arkona	Germany
Boston	Massachusetts	Basel	Switzerland
Cape Hatteras	North Carolina	Bilbao	Spain
Cheyenne	Wyoming	Bourges	Germany
Chicago	Illinois	Caceres	Spain
Cleveland	Ohio	Delft	Netherlands
Dallas	Texas	Gorlitz	Germany
Des Moines	Iowa	Hamburg	Germany
Detroit	Michigan	Ljubljana	Slovenia
Jacksonville	Florida	Luxembourg	Luxembourg
Kansas City	Kansas	Marseille	France
Las Vegas	Nevada	Oberstdorf	Germany
Los Angeles	California	Paris	France
Louisville	Kentucky	Perpignan	France
Nashville	Tennessee	Potsdam	Germany
New York City	New York	Regensburg	Germany
Phoenix	Arizona	Santiago	Portugal
Portland	Oregon	Strijen	Netherlands
Raleigh	North Carolina	Texel	Netherlands
St Louis	Missouri		
Tampa	Florida		

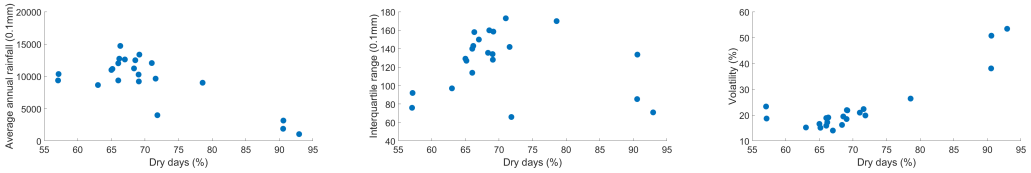
One of the most important aspects about these tables is the climatic difference between cities in the USA and Europe. We can observe the percentage of dry days for cities in the USA varying from 56.67% to 92.93%, compared to 40.82% to 77.03% for cities in Europe. Despite the USA appearing to have drier climates, the total amount of rainfall on an annual basis is much higher than in Europe. The observed average annual rainfall in the USA is 981.08mm, compared to 849.25mm for Europe. This is an interesting difference between the two continents, the USA has short wet spell lengths, but experiences far more extreme rainfall in a single day. Whereas, Europe has longer wet spell lengths, but experiences a more stable amount of rainfall. The annual rainfall amount for the USA is very volatile each year, with the volatility of annual rainfall amounts at 22.13%, compared to 17.86% for Europe. Therefore, each year the USA can expect a larger deviation away from the average rainfall amount. This can also be seen when considering the interquartile range of daily intensities, with the USA having a larger spread compared to Europe.

Table 4.2 A statistical description of our data for the USA.

City	Dry days (%)	Longest dry spell (days)	Longest wet spell (days)	Mean dry spell (days)	Mean wet spell (days)	Average daily rainfall (0.1mm)	Average annual rainfall (0.1mm)	Daily volatility (%)	Highest intensity (0.1mm)	Median intensity (0.1mm)	Inter-quartile range of intensity (0.1mm)
Akron	56.67	21	11	2.89	2.21	29.13	10659.68	19.85	1229	76	97
Atlanta	68.56	28	14	3.94	1.93	34.78	12498.20	19.49	1697	112	160
Boston	65.17	37	13	3.44	1.85	30.32	11158.28	15.14	1552	94	127
Cape Hatteras	66.33	26	12	3.77	1.93	40.66	14727.36	19.01	2207	107	158
Cheyenne	71.84	33	12	4.43	1.85	10.90	3969.20	19.83	617	56	66
Chicago	66.01	24	9	3.64	1.89	25.69	9368.60	18.87	1742	81	114
Cleveland	57.13	21	13	2.98	2.22	28.25	10338.44	18.65	1166	74	92
Dallas	78.59	85	9	5.05	1.88	25.35	9006.72	26.42	1331	117	170
Des Moines	69.09	42	8	3.90	1.86	25.36	9195.44	21.92	1151	89	128
Detroit	62.98	23	10	3.26	1.92	23.57	8654.08	15.22	1161	76	97
Jacksonville	69.17	34	17	4.25	1.96	36.40	13378.32	21.84	1989	102	159
Kansas City	71.58	41	8	4.23	1.77	26.60	9648.08	22.29	1115	99	142
Las Vegas	92.93	146	6	7.94	1.69	2.92	1061.40	53.38	419	64	71
Los Angeles	90.58	225	7	7.75	1.80	8.46	3131.12	50.78	1151	91	134
Louisville	66.03	39	11	3.61	1.86	33.23	12008.68	15.88	1834	102	140
Nashville	67.00	25	10	3.82	1.90	34.49	12596.36	13.99	1842	102	150
New York City	66.17	26	12	3.54	1.82	34.67	12742.40	17.15	1923	102	143
Phoenix	57.07	51	25	4.16	2.79	25.89	9377.64	23.36	683	69	76
Portland	90.54	143	9	7.18	1.74	5.18	1888.40	38.11	838	71	85
Raleigh	68.38	38	12	3.98	1.86	31.10	11245.64	16.23	1433	102	136
St Louis	69.05	23	8	3.97	1.80	28.61	10262.08	18.46	1420	99	134
Tampa	71.01	40	15	4.52	1.91	33.56	12070.28	20.92	2106	109	173

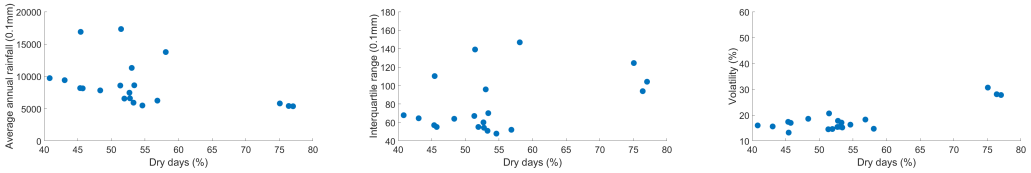
Table 4.3 A statistical description of our data for Europe.

City	Dry days (%)	Longest dry spell (days)	Longest wet spell (days)	Mean dry spell (days)	Mean wet spell (days)	Average daily rainfall (0.1mm)	Average annual rainfall (0.1mm)	Daily volatility (%)	Highest intensity (0.1mm)	Median intensity (0.1mm)	Inter-quartile range of intensity (0.1mm)
Amsterdam	40.82	38	35	3.03	4.39	26.68	9724.00	15.98	581	64	68
Arkona	54.61	49	24	3.2	2.66	15.12	5500.96	16.26	656	53	48
Basel	53.39	24	17	3.36	2.94	23.37	8597.68	15.22	850	65	70
Bilbao	53.03	24	28	3.32	2.96	31.06	11317.28	15.47	1081	81	96
Bourges	52.68	28	22	3.25	2.93	20.42	7472.96	15.37	790	62	60
Caceres	76.39	125	17	5.8	2.62	14.6	5403.08	28.02	1285	76	94
Delft	43.04	29	36	3.03	3.88	26.01	9418.60	15.64	763	62	65
Gorlitz	52.75	29	18	3.16	2.86	17.89	6587.36	17.77	737	55	54
Hamburg	48.33	32	22	3.08	3.28	21.4	7813.00	18.61	682	59	64
Ljubljana	58.09	33	21	3.66	2.75	37.39	13767.00	14.74	1396	103	147
Luxembourg	51.33	28	23	3.32	3.17	23.19	8556.24	14.46	643	62	67
Marseille	77.03	53	12	5.28	2.14	14.61	5347.36	27.76	1460	80	104
Oberstdorf	45.43	36	20	2.92	3.48	45.96	16874.24	13.17	1217	92	110
Paris	56.83	31	19	3.47	2.69	16.94	6229.12	18.24	1042	56	52
Perpignan	75.06	50	9	4.93	1.95	15.65	5793.56	30.65	2220	76	125
Potsdam	53.29	29	19	3.13	2.77	16.15	5908.32	17.05	841	53	51
Regensburg	51.92	25	19	3.09	2.89	17.87	6567.64	14.63	566	56	55
Santiago	51.45	41	53	3.92	3.69	47.23	17335.52	20.69	1186	102	139
Strijen	45.72	37	32	3.15	3.6	22.55	8156.48	16.96	705	59	55
Texel	45.35	27	45	3.03	3.53	22.55	8191.92	17.39	461	60	57



(a) Average annual rainfall (b) Interquartile range (c) Volatility

Fig. 4.1 Identifying any relationships against the percentage of dry days (x axis), using a selection of the descriptive statistics given in Table 4.2 for all USA cities.



(a) Average annual rainfall (b) Interquartile range (c) Volatility

Fig. 4.2 Identifying any relationships against the percentage of dry days (x axis), using a selection of the descriptive statistics given in Table 4.3 for all European cities.

Looking at Europe, there is a general downward trend, shown in Figure 4.2a with the frequency of annual rainfall and percentage of dry days. We would expect this behaviour given Europe's climate is less dominated by extreme stormy conditions, unlike the USA in Figure 4.1a, where the rainfall is generally more uniformly spread. The interquartile range of rainfall intensities for Europe is fairly stable and consistent across the different percentages of dry days, shown in Figure 4.2b. Similarly, this is the case for the USA, as shown in Figure 4.1b. The difference in interquartile range is that the USA has a general downward trend against the percentage of dry days, whereas Europe has a upward trend against the percentage of dry days. However, we do see a possible pattern with intensities where the driest and wettest climates have the lowest interquartile range. This reflects the clear difference in climates, showing that Europe exhibits a more consistent level of rainfall over a year. Those European climates that exhibit large annual rainfall amounts, e.g. Santiago, have a climate more similar to those in the USA. We can identify the three cities that exhibit this climate by the outliers in Figures 4.2a and 4.2b, where three outliers can be identified between the 45% to 60% of dry days. Lastly, the volatility reflects a fairly similar upward pattern for both Europe and the USA, shown in Figures 4.2c and 4.1c. It is worth noting that the volatility within Europe is lower and is far more consistent in comparison against the USA. Therefore, we can clearly see a difference in climates between the two geographic locations and it will be interesting to note if the results of our experimentation are affected.

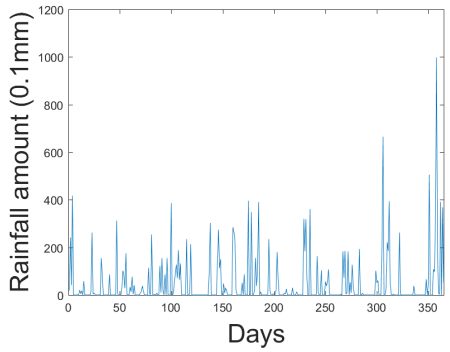
Looking at Figure 4.3, we can observe daily rainfall data from eight different cities, four within the USA and four within Europe. To aid in the comparison of the various climatic differences we have fixed the y-axis to see the sheer difference in intensities. The cities were chosen arbitrarily with some regard to the geographic location attempting to demonstrate the differing climates and we use these examples throughout this chapter to demonstrate various aspects of our data. This gives a picture to the information displayed in Tables 4.2 and 4.3. One of the striking aspects is the sheer volatility on a day by day basis, another is the randomness between the dry to wet days. Although the latter is hard to identify from this snapshot of the data, Tables 4.2 and 4.3 do give some insight into this phenomenon by looking at the various statistics given for the dry/wet spell lengths. One of the key differences is the difference between the USA and Europe, where we observe significantly higher rainfall intensities within the USA.

Observing the data does give some potential difficulties of trying to predict rainfall amounts. Firstly, there is the issue of when a certain day should be dry or wet. Secondly, how to calculate when to make the transition from wet to dry and vice versa. Thirdly, how to predict what the rainfall amount should be, given that the rainfall amount can be from 1mm up to and above the maximum observed rainfall intensity, shown in Tables 4.2 and 4.3. This is made increasingly more challenging, given that the previous day makes little impact to what the current day's intensity is. Therefore, we propose a technique to transform the data to allow algorithms to cope better with the three challenges, whilst making it more intuitive to the end goal of pricing based on accumulated rainfall amounts.

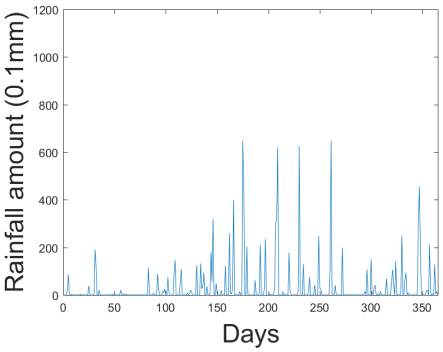
By analysing the data, we have an impression of the difficulties that exist with regression methods. From the statistics describing the data, we can clearly identify that the time series is very volatile, given the large fluctuations around the intensities and overall volatility of the annual amounts. Moreover, when we consider the time series plots of daily rainfall, we can clearly see a highly nonlinear relationship with abrupt spikes and periods of no activity. From a regression perspective, this particular time series would be incredibly hard to fit a model on satisfactorily. Ultimately for this type of data set, it most likely cannot be explained by a single regression equation.

The key findings that will be interesting to examine from our experiments are whether:

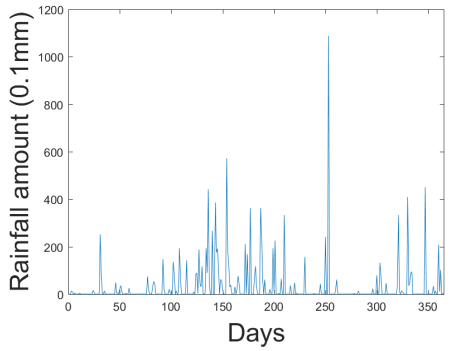
- The predictive error is similar between Europe and the USA.
- Drier or wetter climates are associated with a lower predictive error.
- More volatile cities are associated with higher predictive error.



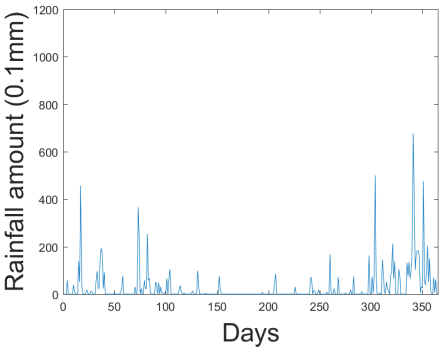
(a) Atlanta



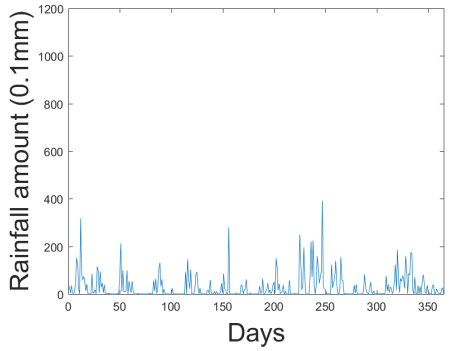
(b) Des Moines



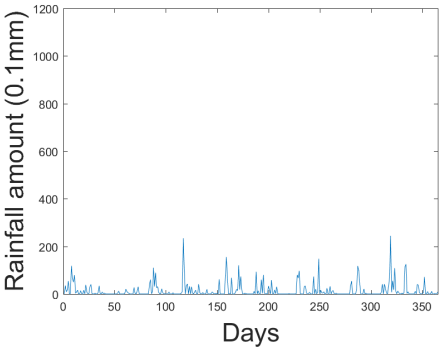
(c) Kansas



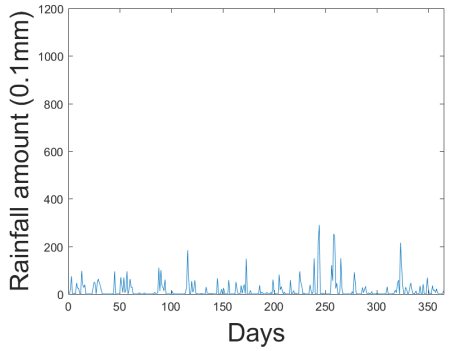
(d) Portland



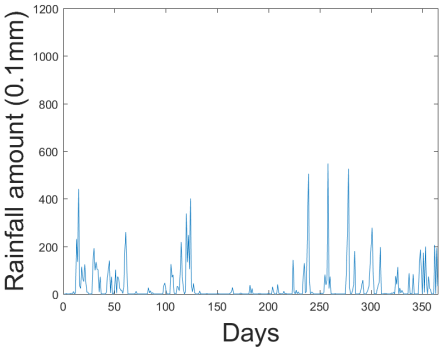
(e) Delft



(f) Gorlitz



(g) Luxembourg



(h) Santiago

Fig. 4.3 The daily rainfall (0.1mm) time series for the period 01/01/2015 - 30/01/2016.

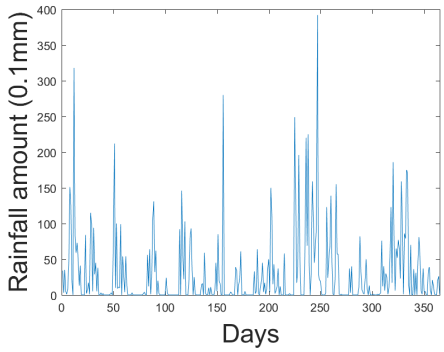
- High rainfall intensities are associated with higher predictive error.

4.3 Data Transformation Using a Sliding Window

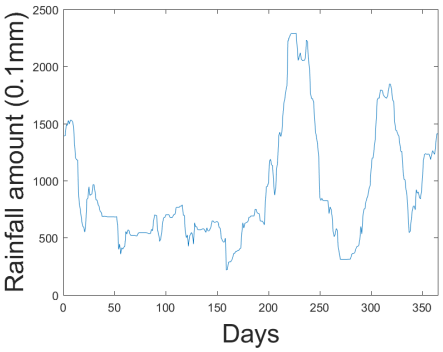
As highlighted in the previous section, we have many different data sets showcasing very different and somewhat difficult climates to predict. The unique characteristics of rainfall data that are shown in Tables 4.2 and 4.3 begin to describe just how chaotic and interesting the data series is on a daily basis. In such cases, the usual approach in time series would be to take the moving average to help smooth the data. However, this would be inappropriate for our problem domain, given we want accumulated amounts of rainfall over a period of time to price rainfall derivatives. By averaging out the values, we would be unable to achieve the final goal despite smoothing out the problem of rainfall prediction. In this case a sliding window is preferred, which is similar to the moving average whereby the daily rainfall series is summed, but the rainfall amounts are not averaged out. The sliding window for a given day is given by:

$$r_{t_s} = \sum_{t=t_s}^{t_e} r_t, \tag{4.1}$$

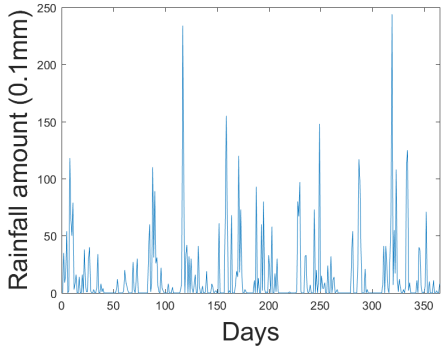
where, r_t is the accumulated amount of rainfall for a given day, with the day varying over a contract period from t_s till t_e. As mentioned earlier, the idea of the sliding window should help smooth out the characteristics of rainfall. This is consistent with pricing a contract, whereby the price of a contract is the total amount of rainfall within a specified period of time, otherwise known as the contract period. The most common contract traded is monthly and contracts are only available for the months of March through October. Given we are interested in pricing monthly contracts, we use a sliding window length that covers the majority of contracts that are traded between March and October. In this paper, the modal length of month within that period is 31 days. We do not look for an optimum period to accumulate to help with prediction, because our problem domain is set out as the accumulated rainfall amounts over the contracts that are currently traded. Figure 4.4 shows the effect of the transformation for four cities, we noticed a similar effect in all cases of our data sets. What we can see from the figure is that a pattern has definitely been uncovered visually from the daily values after transformation. We still have the extreme values within the data with the sudden peaks, but as expected with accumulation, there is a degrading effect from those sudden peaks. Whereas, on the daily values, the spikes are met with low levels of rainfall the following day. This behaviour would be much harder to capture, but the degrading effect should be easier.



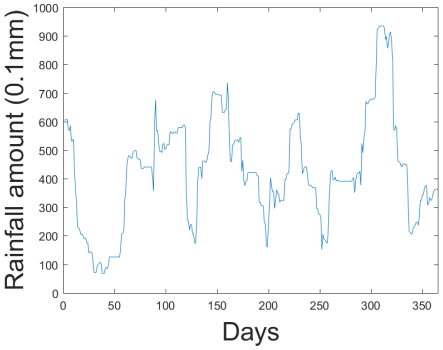
(a) Daily data of Delft



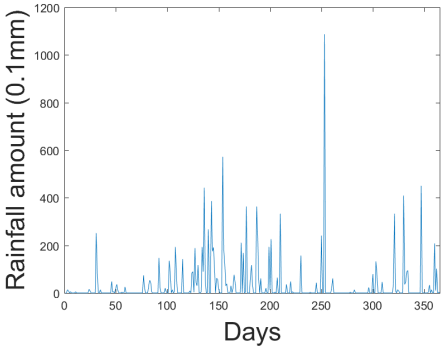
(b) Delft data after transformation



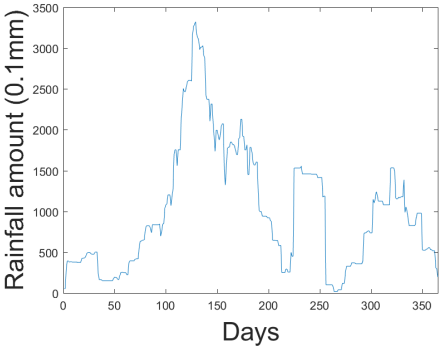
(c) Daily data of Gorlitz



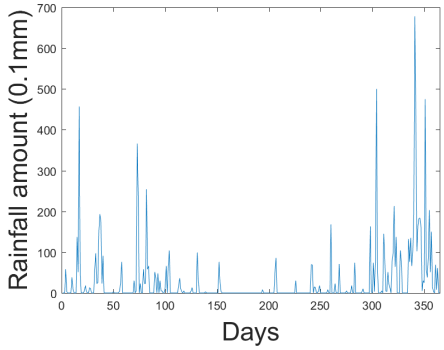
(d) Gorlitz data after transformation



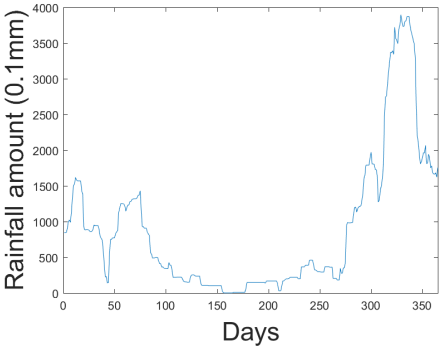
(e) Daily data of Kansas



(f) Kansas data after transformation

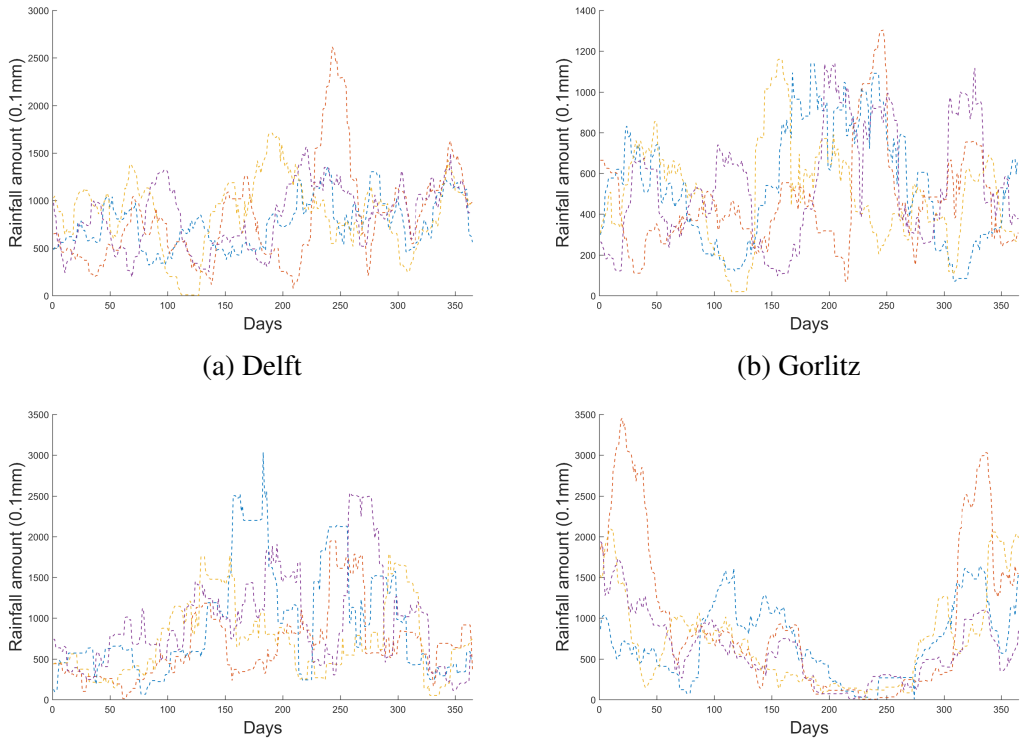


(g) Daily data of Portland



(h) Portland data after transformation

Fig. 4.4 A comparison between the daily rainfall time series and the respective rainfall using the proposed data transformation (0.1mm).



(a) Delft (b) Gorlitz

(c) Kansas (d) Portland

Fig. 4.5 Transformation of the data for various cities (0.1mm) over a four year period, showing the lack of reoccurring pattern.

With patterns more visually accessible, we can further inspect the data, to see if there is any notion of seasonality across multiple years. The initial look at Figure 4.4 hints towards there being some notion of seasonality in a year, cycling through wet and dry periods. However, once we consider how the same period looks on across multiple years we do not witness the cyclical pattern. Figure 4.5 shows the same period but over the last 4 years for the same four cities in earlier analysis. This shows the complexity that exists within the rainfall prediction process; it exhibits little annual seasonal effect, but could have an irregular pattern of seasonal effects. However, unlike temperature, there is a large variation amongst the pattern that exists. On the analysis of all cities we were unable to identify any strong evidence of seasonality.

4.4 Proposed Genetic Programming and Overview of Machine Learning Methods

Our main focus is the proposal of GP to the problem of rainfall derivatives, which we will outline within this section for the initial process of rainfall prediction. However,

along with our proposed GP we also need to compare against a series of benchmarks to fully evaluate the effectiveness of GP. Along with our GP for the problem domain we apply the commonly used approach within rainfall derivatives and across other rainfall related fields, namely Markov chain extended with rainfall prediction (MCRP). Moreover, we propose the application of five other machine learning algorithms for the problem of rainfall prediction for further comparison against GP.

4.4.1 Genetic Programming

Using GP allows the final form of the solution (equation) to be discovered without forcing a predefined relationship. The advantages of GP allow the creation of a generalised approach to rainfall prediction, without the need of a specific model for each city. Within this section, we propose our GP specifically aimed at tackling the rainfall prediction problem. For this thesis, we opt for an extension over the original Koza type of GP (Koza, 1992), and use a Strongly-typed GP (STGP) (Montana, 1995), because we can include different types to avoid illegal trees being generated. Several modifications are made to the STGP, which is covered briefly here.

Data Variables

For our GP to predict the rainfall data, we need to provide a set of variables that GP can use when constructing an equation for rainfall. For our experimentation, our proposal is a set of variables based on the level of rainfall in the past days and years. We construct a series of variables under both daily rainfall setting and under the proposed data transformation setting. As previously mentioned within Section 4.3, the proposed data transformation setting is performed under the modal contract length of 31 days. Therefore, when constructing the variables we accumulate the rainfall over 31 days.

Under the daily rainfall setting, we define a set of variables r_t, which is the level of rainfall based on the previous t days. For example, r_{t-1} represents the level of rainfall yesterday. Additionally, we define a set of variables r_y, which is the level of rainfall for the day GP is trying to predict one year ago. For example, r_{y-1} is the rainfall value for today's date, but one year ago.

Under the data transformation setting, we define a set of variables r_t, which is the accumulated level of rainfall based on the length of the sliding window t periods ago. For example, for a sliding window length of 31, r_{t-1} would refer to the accumulation of the previous 31 days of rainfall. Unlike daily rainfall, r_{t-2} would be the day before r_{t-1}, within our data transformation r_{t-2} refers to the accumulated level of rainfall immediately prior to last day in r_{t-1} with no overlapping days. Assuming a sliding window value of ten, r_{t-2} refers to the accumulation period between 62 and 32 days

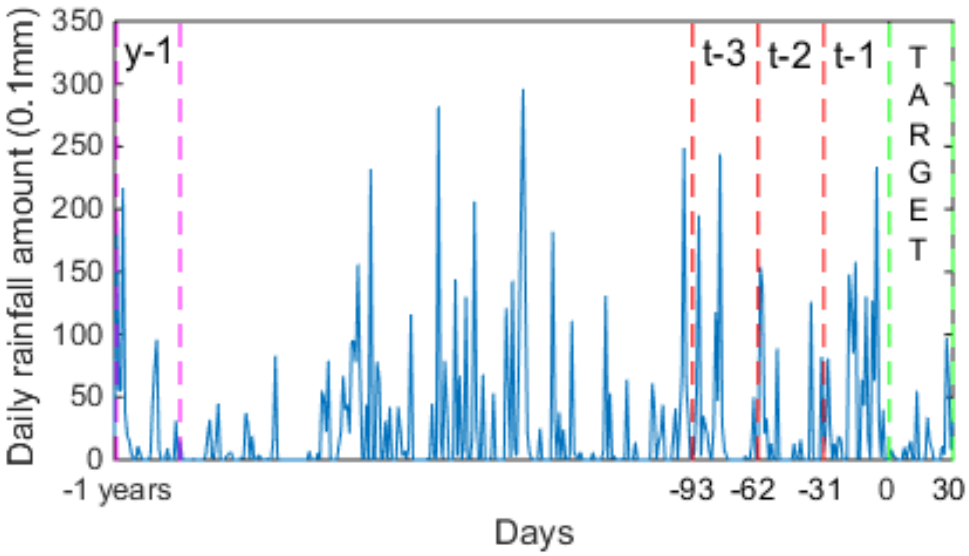


Fig. 4.6 A visualisation for the generation of data variables of r_t and r_y based on the daily data for use on the data transformation data.

prior to today. The variables r_{y-1} refers to the accumulation of rainfall in the next sliding window period one year ago. Therefore, r_{y-1} would consist of the accumulated rainfall values between today's date and the following 30 days, assuming a sliding window length of ten.

We show the accumulation procedure in Figure 4.6, for a sliding window period of 31 days.

Terminals

The first type of terminal is a set of variables that are defined by the r_t's and r_y's calculated based on the data from the previous section, for either daily or transformed data sets.

The second type of terminal is an ephemeral random constant (ERC), which will pick a uniformly distributed random number within a given range. The third type of terminal is a set of constants from -4 to 4, at 0.25 intervals excluding 0, which will take a separate type from the terminals already discussed. These are constants that are specific to the power function. Due to using STGP, we can ensure that the second argument of the power function is always one of these constants and does not create an illegal tree.

Functions

The function set includes: Add (ADD), Subtract (SUB), Multiply (MUL), Divide (DIV), power (POW), square root (SQRT), and log (LOG). The functions LOG, SQRT and DIV are protected. Our LOG is protected by:

$$LOG(x) = \begin{cases} LOG(x) & \text{if } x > 0 \\ LOG(abs(x)) * -1 & \text{if } x < 0 \\ LOG(\delta) & \text{otherwise,} \end{cases} \tag{4.2}$$

where x is the input to the function and δ is the ratio based on the previous two x's used for evaluation of this function. Through calculating the ratio we can determine whether to increase or decrease and can ensure $\delta \in (0,1]$. If this occurs in the first two data points we define $\delta = 1$, so that $\log(\delta)$ will evaluate to 0. Our SQRT is protected by:

$$SQRT(x) = \begin{cases} SQRT(x) & \text{if } x > 0 \\ SQRT(abs(x)) * -1 & \text{if } x < 0 \\ 0 & \text{otherwise.} \end{cases} \tag{4.3}$$

Finally we define our DIV by:

$$DIV(x) = \begin{cases} DIV(x,y) & \text{if } x \neq 0 \\ 0 & \text{otherwise,} \end{cases} \tag{4.4}$$

where x is our numerator and y is our denominator. In both LOG and SQRT we opt to keep the negative sign to influence a downward movement in our rainfall series.

Additionally, the second argument for POW is a constant in a specified range as mentioned in Section 4.4.1. Since we allow for fractional powers, we enforce the same rules as SQRT to deal with issues of square rooting negative values. We summarise the terminals and functions in Table 4.4.

Table 4.4 Genetic Programming functions and terminal sets.

Set	Value
Functions	ADD, SUB, MUL, DIV, POW, SQRT, LOG
Terminals	r_l, r_y, ERC, Constants in the range [-4,4]

Management of Trees

Due to rainfall being a strictly non-negative variable, a wrapper around each individual is included to modify the prediction to zero if the tree evaluates to a negative amount. The final adjustment is to ensure a balance between variables and random numbers in an individual (candidate solution). Thus, the first child of each node is either a function or a variable. Whereas, the second child of each node can be a variable, ERC or a function. We initialise the population using the ramped-half-and-half method. We present the fitness function for the evolution of GP later in Section 4.6.1.

4.4.2 Markov Chain Extended With Rainfall Prediction

The most commonly used methodology in the rainfall prediction literature is Markov chain extended with rainfall prediction (MCRP), based on the single-site version of Wilks (1998). The process is implemented in the rainfall derivative literature by Cabrera et al. (2013); Cao and Wei (2004); Odening et al. (2007); Ritter et al. (2014). MCRP is a daily rainfall model that is split into two processes. Firstly, the occurrence pattern (wet or dry X_t) is calculated and secondly the intensity of rainfall r_t given that the day was wet is computed. The estimated amount of rainfall is given by:

$$R_t = r_t \cdot X_t. \tag{4.5}$$

We first discuss how we calculate the occurrence process X_t, before moving on to discuss r_t, the rainfall intensity for a wet day.

Occurrence Process

The daily occurrence process X_t is defined as a binary event, being either wet (1) or dry (0). The process can be best described as a chain dependent process (Katz, 1977) using a Markov chain. Previous research indicates that the process of rainfall follows a simple (first-order), two-state Markov process. Thus, the rainfall occurrence probability only depends on the previous day. In other words, the best way to describe the chance of rainfall today only depends on whether it is wet or dry yesterday. The daily rainfall transition probabilities p_t^{01} and p_t^{11} determine the probability of rainfall, 01 refers to the probability of rainfall given the previous day was dry and 11 refers to the probability of rainfall given the previous day is wet.

Following on from previous works, the transition probabilities are calculated for each day of the year. Due to the volatility of doing so, a fourier series is used to smooth the transition probabilities across a year. For this set-up, the transitional probabilities

are assumed to stay constant across years. The fourier series is estimated via Maximum Likelihood Estimation (MLE) (Roldán and Woolhiser, 1982).

Whether a day is wet or dry is based on drawing a uniform random variable $u_{1,t} \sim U(0,1)$:

$$X_t = \begin{cases} 1 & \text{if } u_{1,t} \leq p_t^{01} \\ 0 & \text{otherwise.} \end{cases} \tag{4.6}$$

The above is only true when the previous day is dry. On the other hand, if the previous day was wet, then p_t^{01} is replaced by p_t^{11} in Equation 4.6.

Intensity

The daily rainfall amount process r_t can be described either by the mixed exponential distribution (Roldán and Woolhiser, 1982; Wilks, 1999) or by the gamma distribution (Buishand, 1977; Wilks, 1999). The mixed exponential is given by:

$$f(r_t) = \frac{\alpha_t}{\beta_t} \exp\left[\frac{-r_t}{\beta_t}\right] + \frac{1-\alpha_t}{\gamma_t} \exp\left[\frac{-r_t}{\gamma_t}\right], \tag{4.7}$$

where β is the mean of the first exponential distribution and γ is the mean of the second exponential distribution, with a mixing parameter α to control whether a random number is chosen from an exponential distribution with higher mean (β) or lower mean (γ). These parameters are subjected to the following constraints: $\beta_t \geq \gamma > 0$ and $0 < \alpha_t < 1$ for all t. Whereas, the gamma distribution is given by:

$$f(r_t) = \frac{(r_t/\beta)^{\alpha-1} \exp[-r_t/\beta]}{\beta\Gamma(\alpha)}, \tag{4.8}$$

where α and β are the respective shape and scale parameters for the gamma distribution and Γ is the normalising constant evaluated at α.

Similar to the occurrence pattern, the parameters for each distribution are calculated and optimised on a daily basis via historical data. Furthermore, the parameters are smoothed via a truncated fourier series using MLE.

4.4.3 Other Non-Linear Approaches

In addition to GP and MCRP, we also apply other well known techniques to the problem of daily rainfall and rainfall after the data transformation. We apply Support Vector Regression (SVR), Radial Basis Function (RBF), *k*-Nearest Neighbour and M5, both model trees (M5P) and rules (M5R). None of these techniques have been applied to the problem of rainfall derivatives before. The implementation of these algorithms

are those used in Chang and Lin (2011); Hall et al. (2009) and we refer the interested reader to Broomhead and Lowe (1988); Holmes et al. (1999); Quinlan (1992); Vapnik et al. (1996) for a full description of each approach.

4.5 Tuning Method for All Experimentations

In this section, we outline the general procedure used for tuning for GP and our benchmark algorithms of MCRP and other proposed machine learning methods. Two methods will be used for the parameter tuning, due to the difference in set-up between MCRP and the other techniques we are unable to have a common tuning method. The first method will be used for tuning MCRP parameters to each city according to the literature. The second parameter tuning method will be used for the other techniques applied within this paper. As highlighted earlier, one of the disadvantages of MCRP is that it is purely driven off the historical data of each city and needs to be tuned according to the specific data set.

4.5.1 Parameter Tuning - GP and Machine Learning Methods

In order to tune our GP and the other machine learning benchmarks, we use iRace; a popular tool for optimising parameters of algorithms (López-Ibáñez et al., 2011). It is an iterative process, which samples and evaluates many parameter configurations across multiple problems in order to find an optimal configuration for the problem. The advantage of using such a tool is that no prior knowledge is required and even for experienced users of a certain algorithm, iRace considers combinations that a user may never have considered. Additionally, the process of finding the best configuration is more efficient than blindly guessing or by using the best configuration for a previous problem. A configuration that works well on a previous problem may not necessarily work for a previously solved but very different problem. Across each iteration, iRace resamples configurations that perform well. Therefore, allowing iRace to search the space of the problem, and focus on promising areas.

Using such a tool is crucial as it reduces the chance of bias through the tuning procedure. By performing the process manually, we may be more likely to settle on optimal configurations for some algorithms instead of others. For example, GP and RBF, there are only three parameters to tune for RBF, two continuous (c) and one discrete (d) (minimum standard deviation (c), ridge (c) and number of clusters (d)); whereas, GP has eight with three discrete and five continuous (population size (d), generations (d), tree depth (d), mutation rate (c), crossover rate (c), elitism percentage(c), primitive probability (c) and terminal node bias (c)). To decide on

Table 4.5 List of cities used for tuning purposes only, located around Europe and the USA.

Cities used for tuning			
City	Country	City	State
Berg	Austria	Akron	Ohio
De Kooy	Netherlands	Charlotte	North Carolina
Falsterbo	Sweden	Little Rock	Arkansas
Nancy	France	Minneapolis	Minnesota
Valencia	Spain	Philadelphia	Pennsylvania

an optimal tuning configuration is much easier for RBF than for GP due to less number of parameters. By using iRace, we have the advantage of being able to account for these difficulties.

In order to generalise our configurations as best as possible, we propose a set of steps required to perform tuning to avoid bias within the data. Firstly, to assist in the generalisation, we tune based on a small set of cities which are not part of our data sets given in Section 4.2. Secondly, we do not use the same period as the testing set for our original cities to avoid any potential bias. We include the set of cities used for tuning purposes only in Table 4.5, along with its location.

To keep the process as fair as possible, these cities are chosen based on having a similar type of climate to another data set. Although, we do not want to bias towards certain types of climate, our tuning set must be representative of those we are looking to compare our algorithms on. Therefore, we have chosen ten cities that have climates similar to our data sets outlined in Section 4.2, based on the statistical properties.

From our ten data sets (five in each continent), we create multiple different training sets to use for tuning from each city and test on a validation set, which constitutes the final year in each new subset of data. More precisely, for each city's data set we create 9 smaller data subsets, each consisting of 10 years of rainfall data with a preserved temporal order, and a 5 year overlap between each data set. For the testing year 2015, our first training set would be the years spanning from 2005 until 2014, the second would be 2000 until 2009 and so on. Using such an overlap allows for the generation of a higher number of data subsets, making the parameter tuning procedure more robust. The reason for this specific overlap and this number of smaller subsets of data is based on the availability of daily rainfall data that is consistent across all 10 cities chosen for tuning, where we are able to go back to 1957 for all cities having a complete data set.

Even though we are using different cities for tuning, we still use the last year in each data subset and not the testing set period. The reason is that climatic effects must remain unseen at all times. For example, a drought period developing in Europe may

affect most cities, if this occurs during the testing periods then all algorithms have been biased towards this behaviour. Hence, within the 10 years of each data subset, the first 9 years are used for model building and the tenth year for validation. The procedure we take is summarised by Figure 4.7. The validation set length is chosen such that it is consistent with the testing set length. In total, we have 90 training sets to be used by iRace, where each city has 9 different data folds.

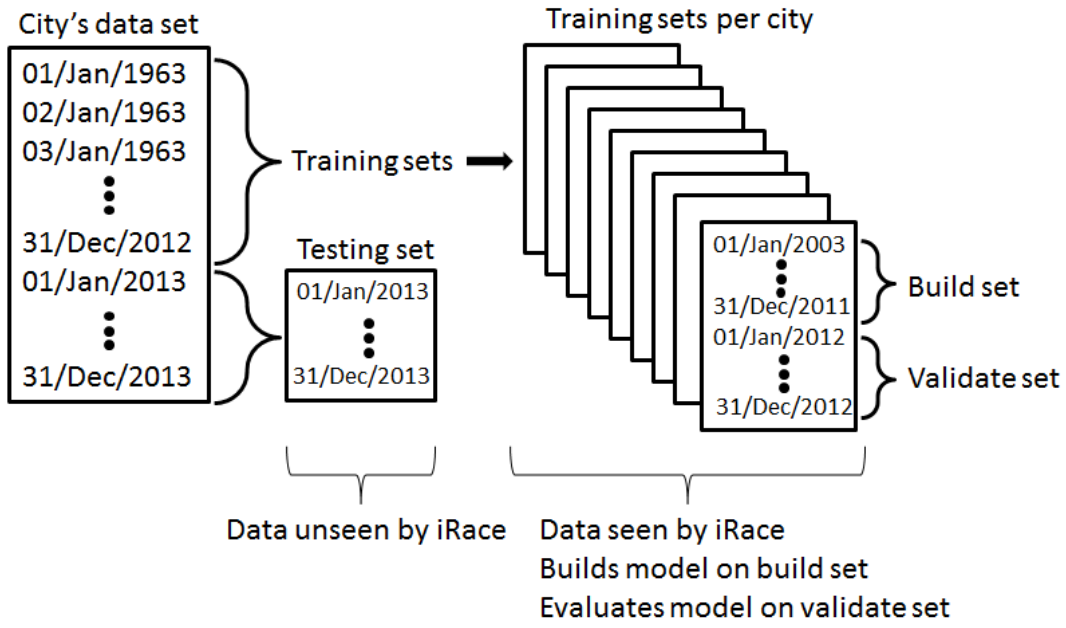


Fig. 4.7 The set-up of each city's data set and how iRace interacts with the training set.

All of our algorithms are tuned based on the same 90 different tuning sets and are examined on each one the same number of times to maintain consistency across all algorithms. After running iRace, the top performing configuration is obtained for each algorithm.

4.5.2 Parameter Tuning - MCRP

It should be noted that iRace is not used for the configuration of MCRP, because MCRP does not have a configuration set that controls the behaviour of itself (compared to GP). Furthermore, there are only two components (occurrence and amount) that make up MCRP, which have no alternatives. The occurrence and amount are tailored specifically for each data set, based on the daily rainfall values. Firstly, the occurrence process is controlled via a Markov-chain, which has its transitional probabilities calculated deterministically. Secondly, the amount process is controlled by a single distribution (in our case, gamma or mixed-exponential), which is estimated based on the data.

Thus, both of these aspects fall outside the scope of iRace, as neither component requires parameters to be optimised. However, if any estimation is required (e.g. the fourier series or distribution parameters) we use Maximum-Likelihood Estimation (MLE), which is a standard technique used within statistics in estimating parameters for statistical models (Woolhiser and Pegram, 1979). Therefore, we keep our benchmark consistent with the literature.

The other key aspect with MCRP is the different configurations, which are chosen for each city based on the Akaike Information Criterion (AIC). It is a standard model fitting measure based on the likelihood function. The configurations refers to the order of the Markov chain and whether gamma or mixed-exponential is the preferred choice to model the intensity. Similar to the literature, we choose the best one for each city based on the lowest AIC score.

4.6 Experimental Set-Ups

In this section, we outline the data used for experimentation and parameter tuning for MCRP, GP, SVR, RBF, KNN and M5 (both M5R and M5P) outlined in Section 4.4. We investigate three main research questions in this chapter. Firstly, whether the use of the data transformation method has a positive effect on the performance of our algorithms. Secondly, whether GP is the best performing algorithm and specifically whether it outperforms the most commonly used approach of MCRP. Thirdly, whether the algorithms are biased towards certain types of climate.

4.6.1 Error Measurement and GP Fitness Function

The fitness function used for evaluating an individual (candidate solution) of the GP and the overall performance of each technique is the root-mean-squared error (RMSE):

$$RMSE = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(r_t - \bar{r}_t)^2}, \tag{4.9}$$

where T is the length of the training set, r_t represents the predicted rainfall amount and $\bar{r}_t$ represents the actual rainfall amount for the t^{th} data point (time index). As we have two prediction landscapes, the RMSE refers to the daily data and to the accumulated level of rainfall.

4.6.2 Training and Testing Set-Ups

All algorithms will have their predictive error compared on all 42 different data sets across the USA and Europe and will all use exactly the same length of data. In total, we will be using 20 years of data in order to construct all of the variables in our data sets, meaning that all algorithms, except MCRP, will be trained on 10 years of data and be tested on one year of data. Whereas, MCRP will use the full 20 years of data without the need for variable creation. The 20 year period spans from 01/01/1995 to 31/12/2014, with a 10 year training set from 01/01/2005 to 01/12/2014. All techniques will be compared on the same unseen testing data set of 01/01/2015 to 31/12/2015. 10 years of training will be chosen based on previous experimentation, training on shorter or longer periods did appear to affect performance.

4.6.3 Parameter Tuning

As previously mentioned in Section 4.5, we outlined the two methods required for tuning. To reiterate, the tuning data sets are different to our pool of 42 cities listed in Section 4.2 to aid the generalisation of algorithms. Moreover, we developed a set of steps in order to avoid bias while using iRace, and most importantly keeping the testing set unseen and untouched throughout the whole process. The results from tuning of the bench-marking (algorithms) parameters are shown in Table 4.6, and the results from tuning the GP are shown in Table 4.7.

We include two sets of tuning results, because we have two very different landscapes to test on, as shown visually earlier in Figures 4.4. The first using daily data to predict daily rainfall amounts. The second using the data transformation to accumulate the daily rainfall to predict future levels of accumulated rainfall.

4.6.4 Experimentation

We will compare the results of each algorithm on data with and without the data transformation and discover what the effect is and how beneficial it is. The exception is that MCRP is strictly a daily prediction technique and cannot be adapted to use the sliding window transformation. Therefore, we report the error after accumulating the daily predictions, to act as a benchmark. Due to applying a wide variety of regression techniques, different data set-ups will be required. Firstly, MCRP is heavily reliant on a large number of Monte Carlo simulations (10,000 runs) before we are able to take the median result. We require 10,000 runs to allow the technique to converge. Based on how MCRP works, we will not be able to predict the rainfall amounts using the sliding window technique and have to predict daily before accumulating.

Table 4.6 Optimal parameters using iRace for the four benchmark non-linear models: SVR, RBF, M5R and M5P for daily (top) and accumulated (bottom) rainfall.

SVR		RBF		M5R and M5P		
SVM Type	epsilon-SVR	Minimum SD	17.5624	minInstance	32	49
Cost	9.6974	NumClusters	71	Regression tree	no	yes
Gamma	6.8399	Ridge	3.2138	Unpruned	no	no
Kernel Type	RBF			Unsmoothed	yes	no
Epsilon	0.3017					
SVM Type	epsilon-SVR	Minimum SD	25.3373	minInstance	9	1
Cost	8.4114	NumClusters	2	Regression tree	yes	yes
Gamma	0.5382	Ridge	3.2443	Unpruned	yes	no
Kernel Type	RBF			Unsmoothed	no	no
Epsilon	0.4719					

Table 4.7 The optimal configuration of GP using iRace for daily prediction and transformed prediction.

GP Parameters	GP daily	GP transformed
Max depth of tree	6	5
Population size	1100	1400
Crossover	55%	81%
Mutation	33%	38%
Primitive	34%	32%
Terminal/Node bias	31%	48%
Elitism	1%	6%
Number of generations	70	50
ERC negative low	-465.36	-366.29
ERC negative high	-47.41	-120.17
ERC positive low	191.95	131.85
ERC positive high	407.01	366.01

GP is a stochastic algorithm producing many different models, so we run GP for 50 times before taking the median result from the set of the best individuals on the training set from each run. RBF will also be run 50 times, but neural networks are known for getting stuck in local optima and so we take the best result on the training

set across all 50 runs. SVR, KNN, M5P and M5R are all deterministic and only one run will be required.

4.7 Results

In this section, we look at the predictive error between algorithms with and without the data transformation. In order for a fair comparison across different approaches and data sets, the errors will be normalised based on the mean of the data set. We perform the following normalisation:

$$\text{CV(RMSE)} = \frac{\text{RMSE}}{\hat{x}}, \tag{4.10}$$

where $\hat{x}$ is the mean of the target variable on the testing set, with the new objective measure for comparison being the coefficient of variation of the RMSE (CV(RMSE)). We opt for the mean instead of the range to act as our normalisation constant, to avoid biasing the perceived CV(RMSE), given that our data exhibits large volatile spikes and skews the range. The median is unsuitable as zero's are possible if the dry days percentage is above 50%.

Our results will follow, by first establishing whether the transformation has an impact on the model's performance before analysing how the different algorithms performed on the problem. We will tackle this in a general sense amongst all algorithms, before moving on to discuss the predictive accuracy with emphasis on our GP. Finally, within the analysis we will also consider the effect of analysing cities in the USA and Europe separately, since we have shown that the two climates are completely different.

4.7.1 Effects of the Data Transformation Technique

Tables 4.8 and 4.9 show the CV(RMSE) for each algorithm before and after the data transformation, for Europe and the USA. Figures 4.8, 4.9, 4.10 and 4.11 shows the predictions against actual rainfall for GP, RBF and MCRP before and after the data transformation, for select cities across Europe and the USA.

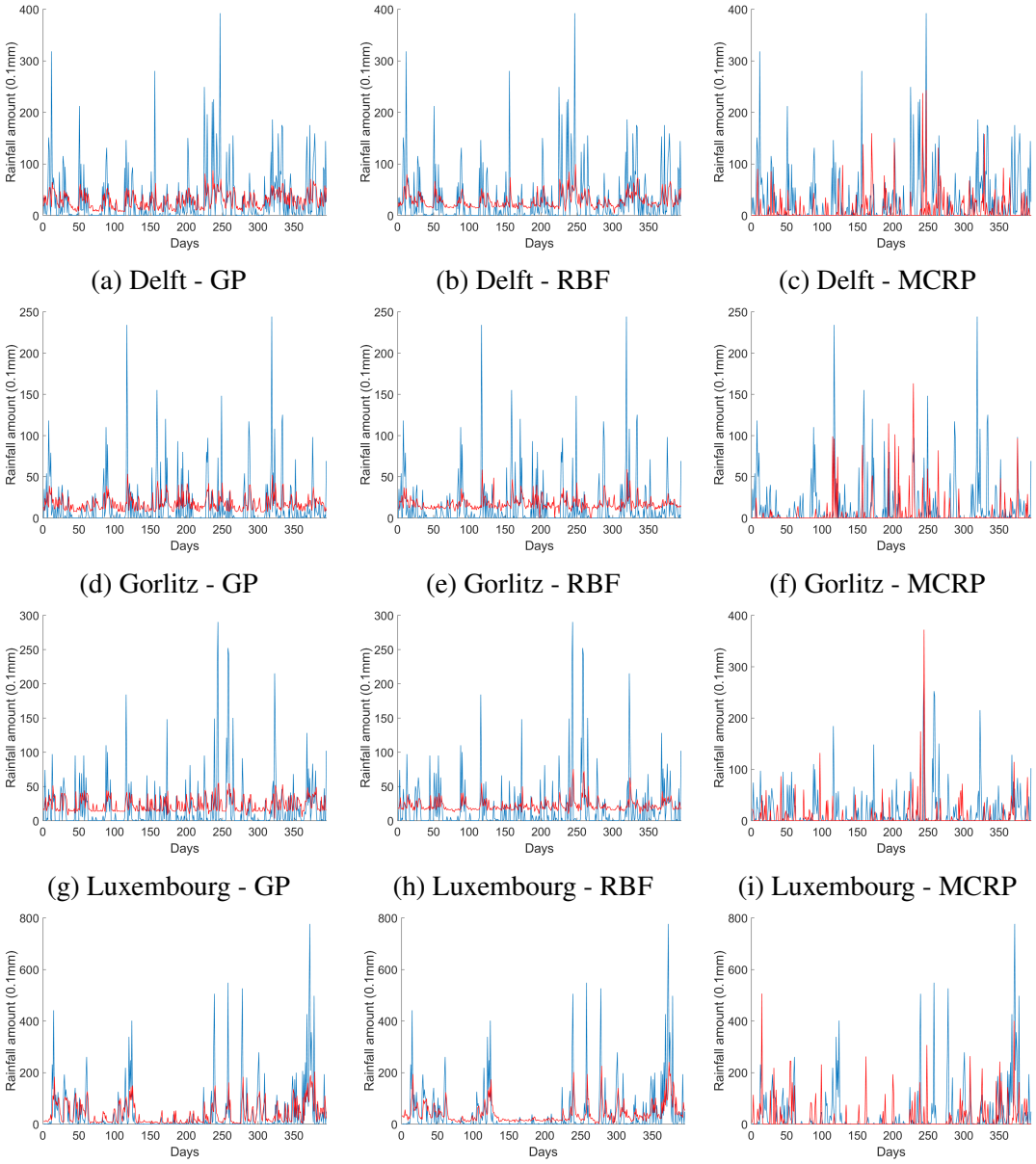
Tables 4.8 and 4.9 show that the use of the transformation decreases the predictive error in all cases. One of the main reasons for the increase in model performance is the issue of problem difficulty that was presented earlier in Figure 4.4, when we compared the daily rainfall amounts before and after applying the sliding window transformation. Considering the effect of the transformation, we can observe that the difficulty of the problem has decreased, which is reflected by the respective smaller CV(RMSE). We observed a error reduction of around 70% on average, with the most noticeable

Table 4.8 The standardised results for European cities with the daily prediction's CV(RMSE) shown on the left side and the data transformation's CV(RMSE) shown on the right side. Values in bold represent the best performance in CV(RMSE) both before and after the data transformation.

Data	Daily prediction							Transformed prediction						
	SVR	RBF	M5R	M5P	KNN	GP	MCRP	SVR	RBF	M5R	M5P	KNN	GP	MCRP
Amsterdam	2.894	2.857	2.796	2.780	2.920	**2.760**	2.840	0.855	**0.834**	0.973	0.923	0.935	0.851	1.219
Arkona	1.899	1.908	1.869	1.880	1.912	1.763	**1.761**	0.423	**0.399**	0.551	0.575	0.510	0.490	1.111
Basel	1.531	1.531	1.836	1.535	1.533	**1.370**	1.377	**0.264**	0.303	0.379	0.367	0.271	0.287	0.604
Bilbao	6.226	**5.628**	5.749	5.870	6.463	6.667	6.511	2.624	2.431	2.928	2.953	3.166	2.612	**2.058**
Bourges	1.587	1.563	1.573	1.565	1.600	1.700	**1.470**	0.325	**0.318**	0.437	0.425	0.358	0.355	0.679
Caceres	3.186	3.067	3.053	3.087	3.160	2.363	**2.054**	**0.708**	0.712	0.823	0.977	1.050	0.827	1.373
Delft	4.190	4.117	4.110	**4.063**	4.202	4.773	4.850	1.471	1.389	1.614	1.445	1.489	**1.367**	1.773
Gorlitz	1.913	1.929	1.921	1.870	1.914	1.104	**1.103**	**0.315**	0.410	0.451	0.506	0.339	0.411	0.768
Hamburg	1.551	1.496	1.502	**1.494**	1.573	1.527	1.509	0.421	**0.395**	0.462	0.471	0.430	0.437	0.817
Ljubljana	10.660	10.652	10.516	10.548	10.700	7.053	**6.123**	**1.187**	1.259	1.736	1.795	3.082	1.300	1.609
Luxembourg	1.274	1.242	1.254	1.251	1.283	**0.901**	0.993	0.275	0.279	0.349	0.383	**0.226**	0.294	0.465
Marseille	2.104	2.015	1.982	1.989	2.046	1.667	**1.586**	**0.347**	0.350	0.536	0.449	0.746	0.411	0.641
Oberstdorf	2.740	2.718	2.722	2.663	2.767	2.361	**2.320**	**0.462**	0.469	0.554	0.547	0.671	0.557	0.609
Paris	0.923	0.899	0.917	0.913	0.928	**0.840**	1.002	**0.200**	0.204	0.233	0.243	0.200	0.221	0.473
Perpignan	16.219	14.641	15.028	15.385	16.132	12.040	**7.405**	**2.403**	2.494	3.093	2.940	9.344	2.548	3.865
Potsdam	1.362	**1.303**	1.368	1.353	1.369	1.352	1.493	**0.231**	0.254	0.333	0.324	0.394	0.278	0.705
Regensburg	1.268	1.255	1.260	1.248	1.269	**1.194**	1.239	0.273	0.273	0.338	0.338	**0.243**	0.280	0.622
Santiago	17.405	16.207	16.165	**15.810**	17.986	15.933	16.071	5.777	5.342	6.585	7.410	8.057	6.039	**3.606**
Strijen	1.026	1.010	1.028	**1.006**	1.031	1.413	1.378	0.432	0.430	0.437	0.470	0.453	**0.419**	0.509
Texel	0.987	0.986	0.990	0.966	0.996	0.965	**0.931**	**0.280**	0.282	0.294	0.302	0.350	0.294	0.440
Mean rank	12.25	10.60	11.05	10.10	13.20	**9.85**	9.90	**1.90**	2.05	4.90	5.10	4.75	3.25	6.10

Table 4.9 The standardised results of the USA with the daily prediction's CV(RMSE) shown on the left side and the data transformation's CV(RMSE) shown on the right side. Values in bold represent the best performance in CV(RMSE) both before and after the data transformation.

Data	Daily prediction							Transformed prediction						
	SVR	RBF	M5R	M5P	KNN	GP	MCRP	SVR	RBF	M5R	M5P	KNN	GP	MCRP
Atlanta	4.955	4.905	4.893	**4.889**	4.942	5.597	5.161	1.450	1.353	1.469	1.439	1.554	1.352	**1.044**
Boston	3.611	3.586	3.645	3.594	3.611	2.859	**2.476**	**0.552**	0.570	0.889	0.761	1.375	0.594	0.879
Cape Hatteras	7.739	7.653	7.637	**7.615**	7.706	8.999	8.177	2.409	2.279	2.501	2.616	2.759	2.209	**1.453**
Cheyenne	1.248	1.256	1.486	1.218	1.234	**1.184**	1.407	0.356	0.365	0.365	**0.307**	0.458	0.351	0.638
Chicago	4.531	4.498	4.757	4.461	4.525	4.180	**3.452**	0.835	**0.816**	1.037	1.054	1.350	0.938	1.163
Cleveland	5.756	5.756	5.748	5.758	5.758	4.967	**4.459**	**1.207**	1.232	1.286	1.344	1.219	1.224	1.413
Dallas	1.603	1.649	1.659	1.604	**1.590**	3.188	2.729	1.069	1.040	1.107	1.061	1.198	1.019	**0.514**
Des Moines	6.235	6.252	6.227	6.209	6.213	5.697	**5.041**	1.200	**1.061**	1.443	1.491	2.170	1.239	1.313
Detroit	5.530	5.538	5.534	5.549	5.529	3.375	**2.993**	0.847	**0.807**	0.953	0.971	1.001	0.860	1.371
Indianapolis	2.644	2.661	2.684	2.660	**2.643**	3.522	2.847	1.024	1.032	1.031	1.057	1.090	1.022	**0.709**
Jacksonville	2.191	2.217	2.191	2.149	2.170	1.670	**1.500**	0.395	0.396	0.418	**0.385**	0.503	0.434	0.425
Kansas	3.936	3.938	3.949	3.937	3.906	3.758	**3.061**	0.916	0.929	0.986	1.108	1.377	0.929	**0.819**
Las Vegas	**0.175**	0.182	0.175	0.175	0.175	0.333	0.814	0.066	**0.057**	0.075	0.101	0.067	0.065	0.411
Los Angeles	1.409	1.397	1.405	1.405	1.405	1.420	**1.316**	**0.251**	0.276	0.331	0.371	0.475	0.363	0.726
Louisville	**3.051**	3.067	3.137	3.079	3.056	4.195	3.754	1.070	1.066	1.089	1.167	1.240	1.064	**0.735**
Nashville	3.037	3.089	3.059	3.037	3.031	2.563	**2.344**	0.412	**0.406**	0.539	0.517	0.455	0.454	0.598
New York	3.568	3.597	3.575	3.526	3.560	2.724	**2.505**	**0.406**	0.445	0.794	0.552	0.710	0.525	0.642
Phoenix	2.217	2.344	2.205	2.205	2.205	**0.959**	0.980	0.206	**0.164**	0.279	0.225	0.171	0.191	0.792
Portland	1.729	1.701	1.742	**1.675**	1.750	1.989	2.075	0.717	0.673	0.756	0.776	0.976	0.725	**0.569**
Raleigh	2.949	2.946	2.948	2.915	2.930	2.363	**2.206**	0.468	0.413	0.527	0.539	0.483	**0.403**	0.531
St Louis	2.710	2.677	2.677	**2.657**	2.707	3.835	3.266	1.122	1.093	1.090	0.979	1.212	1.038	**0.686**
Tampa	3.025	3.004	3.007	2.998	2.986	3.089	**2.693**	0.901	0.847	0.973	1.031	1.072	0.928	**0.470**
Mean rank	11.41	11.50	11.68	**10.05**	10.41	11.05	10.18	2.82	**2.50**	5.00	4.82	5.95	2.86	4.32



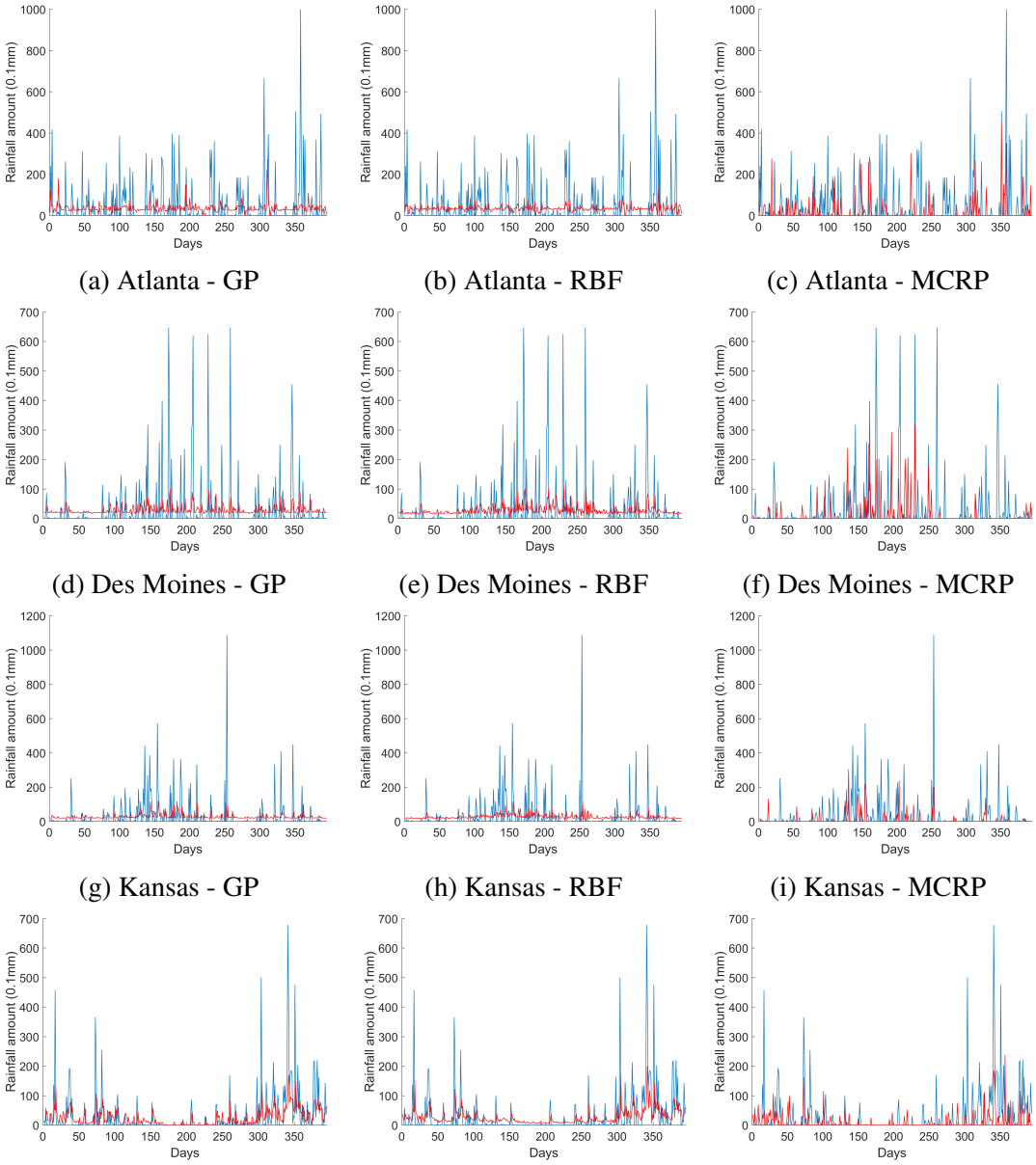
(a) Delft - GP (b) Delft - RBF (c) Delft - MCRP

(d) Gorlitz - GP (e) Gorlitz - RBF (f) Gorlitz - MCRP

(g) Luxembourg - GP (h) Luxembourg - RBF (i) Luxembourg - MCRP

(j) Santiago - GP (k) Santiago - RBF (l) Santiago - MCRP

Fig. 4.8 Comparison of GP's, RBF's and MCRP's predictive performance on four cities across Europe using the daily data. The blue line being the actual daily rainfall amount (0.1mm) and the red line being the predicted daily rainfall amount (0.1mm).



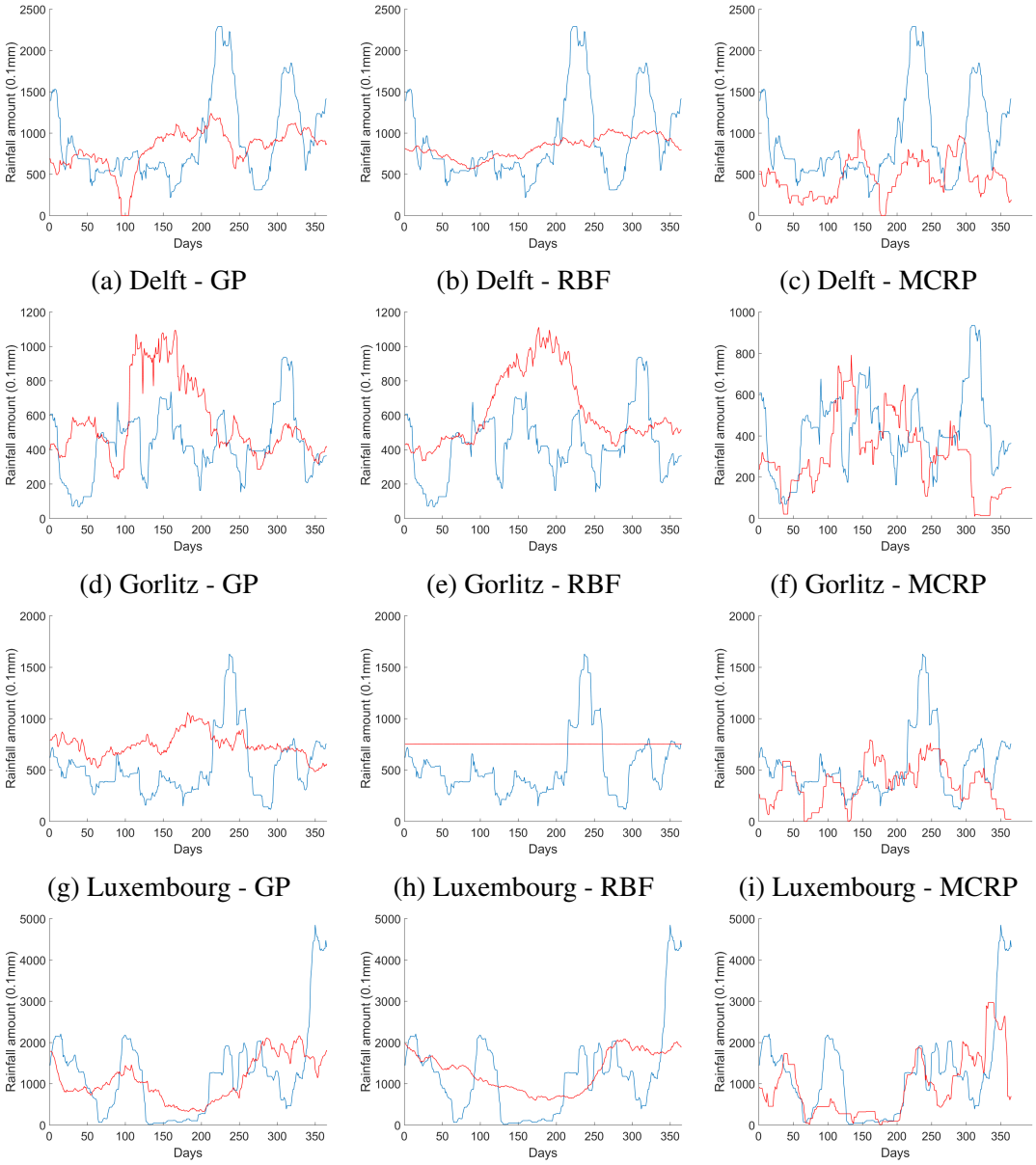
(a) Atlanta - GP (b) Atlanta - RBF (c) Atlanta - MCRP

(d) Des Moines - GP (e) Des Moines - RBF (f) Des Moines - MCRP

(g) Kansas - GP (h) Kansas - RBF (i) Kansas - MCRP

(j) Portland - GP (k) Portland - RBF (l) Portland - MCRP

Fig. 4.9 Comparison of GP's, RBF's and MCRP's predictive performance on four cities across the USA using the daily data. The blue line being the actual daily rainfall amount (0.1mm) and the red line being the predicted daily rainfall amount (0.1mm).



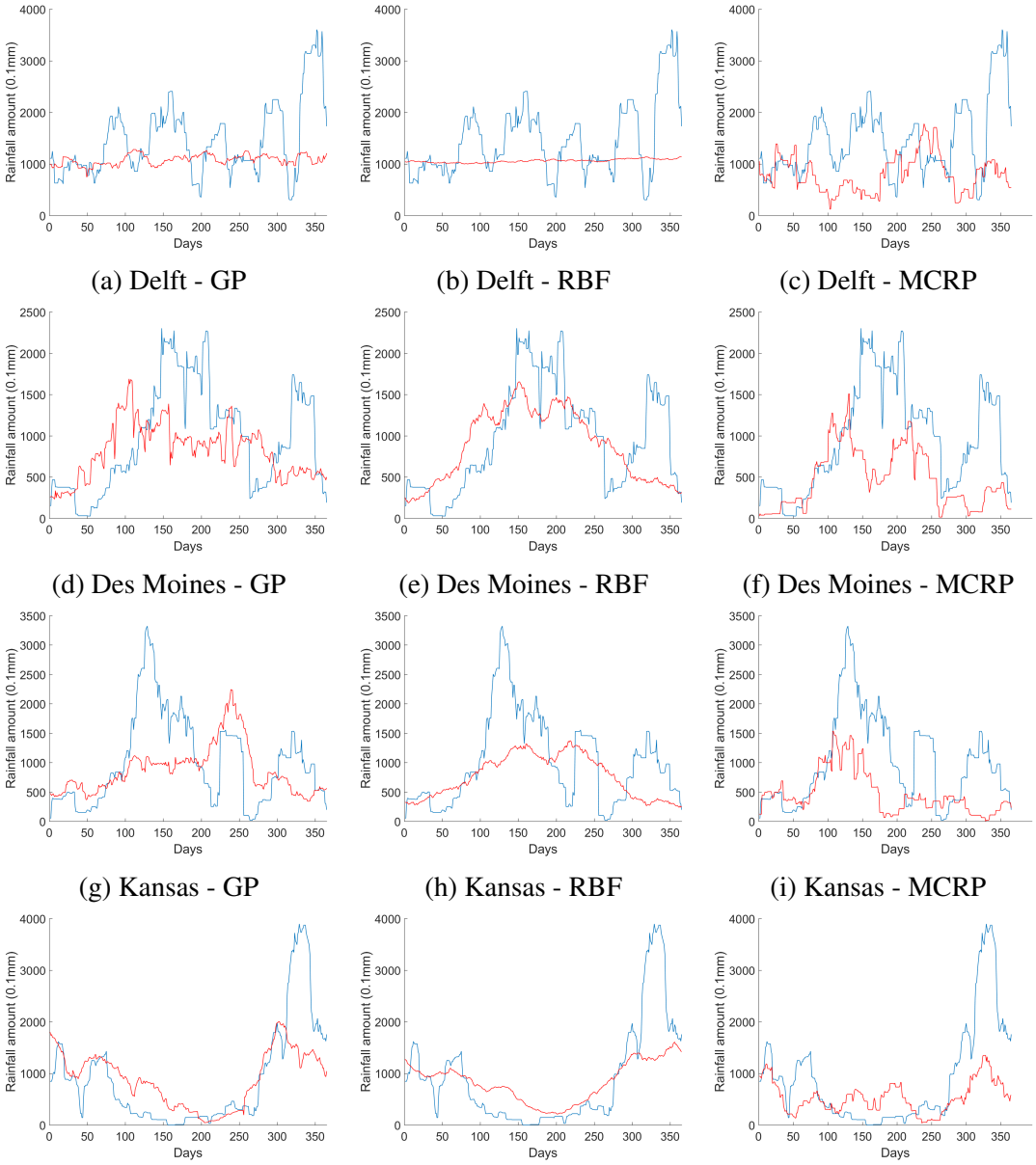
(a) Delft - GP (b) Delft - RBF (c) Delft - MCRP

(d) Gorlitz - GP (e) Gorlitz - RBF (f) Gorlitz - MCRP

(g) Luxembourg - GP (h) Luxembourg - RBF (i) Luxembourg - MCRP

(j) Santiago - GP (k) Santiago - RBF (l) Santiago - MCRP

Fig. 4.10 Comparison of GP's, RBF's and MCRP's predictive error (RMSE) on four cities across Europe after applying the data transformation. The blue line being the actual rainfall amount (0.1mm) after the data transformation and the red line being the predicted rainfall amount (0.1mm) after the data transformation.



(a) Delft - GP (b) Delft - RBF (c) Delft - MCRP

(d) Des Moines - GP (e) Des Moines - RBF (f) Des Moines - MCRP

(g) Kansas - GP (h) Kansas - RBF (i) Kansas - MCRP

(j) Portland - GP (k) Portland - RBF (l) Portland - MCRP

Fig. 4.11 Comparison of GP's, RBF's and MCRP's predictive error (RMSE) on four cities across the USA after applying the data transformation. The blue line being the actual rainfall amount (0.1mm) after the data transformation and the red line being the predicted rainfall amount (0.1mm) after the data transformation.

improvements for Phoenix (92% for RBF and M5P), Ljubljana (88% for RBF and SVR) and New York city (88% for SVR). Moreover, comparing Figures 4.8 and 4.9 with 4.10 and 4.11, we can visually see the improvements in predictive performance after applying the data transformation. After the data transformation, we notice that SVR, RBF and GP are the top performing algorithms, whereas the best algorithms prior to the transformation were M5P, GP and MCRP. We can observe that the algorithms

are able to fit the transformed data much better than the original daily values, but there is still a concern with the predictive error for all methods.

The nonlinear techniques find it very challenging to predict daily rainfall, let alone even fit the data. What we observe from Figures 4.8 and 4.9 is that the techniques try to find an average level of rainfall for each day. This means that when we try to predict on a daily basis, we predict that every day rains, indicating that the model is underfitting by not considering the dry and wet day pattern. However, we know this to be false based on the information presented earlier in Tables 4.2 and 4.3, where we expect (depending on the data set) somewhere between 10% to 60% of days to be wet over a year. One of the reasons for this behaviour is that all algorithms (except MCRP) are misspecified for the fitting of daily rainfall (regardless of tuning). In this particular problem domain, the highly discontinuous and irregular pattern restricts the algorithms from focussing on the classification and regression aspect simultaneously. Each algorithm appears to only focus on the regression side without considering the discontinuous nature of the data.

The exception to this is MCRP, which does exceptionally well at capturing the structure of rainfall on a daily basis, where we find that the rainfall pathways generated are similar to that of our prior information. However, the approach itself is predictively very weak, which we expect, as the model is fully representative of the historical data and does not take into account annual variations or extreme values.

Even though we are able to reduce the complexity of the problem, the algorithms could not maximise the transformed landscape. We can observe this by the lack of coverage for all possible rainfall amounts, shown in Table 4.10 and visually in Figures 4.10 and 4.11. The coverage is defined by the percentage between the range of each algorithm's predictions and the range of rainfall in the data set, given by:

$$Coverage = \frac{r_{max} - r_{min}}{\hat{r}_{max} - \hat{r}_{min}}, \tag{4.11}$$

where r represents the predicted rainfall amounts and $\hat{r}$ represents the rainfall amounts observed in the data set. If $r_{min} < \hat{r}_{min}$, then we set $r_{min} = \hat{r}_{min}$. Similarly, if $r_{max} > \hat{r}_{max}$, then we set $r_{max} = \hat{r}_{max}$.

In some cases, we cover the full 100% of possible rainfall amounts for GP and M5P. However, in a small number of cases we only cover as low as 0% for RBF. A coverage of 0% means the algorithm predicted a single value. This highlights an important aspect of possible model misspecification, most likely occurring from our tuning process, because we wanted to have a robust parameter setting for all data sets, instead of trying to find the best parameter setting for each data set (city) separately.

Table 4.10 The rainfall prediction coverage (given by Equation 4.11) for all algorithms, presented as the median coverage and the range of minimum and maximum coverage across all data sets.

Algorithm	Europe		the USA	
	Median	Range	Median	Range
GP	57%	22%-100%	41%	12%-92%
SVR	26%	17%-67%	22%	7%-46%
RBF	34%	0%-73%	21%	1%-67%
M5P	77%	40%-100%	65%	39%-99%
M5R	75%	45%-98%	56%	31%-99%
KNN	0%	0%	0%	0%

The reasons for having a single parameter set are twofold: for efficiency purposes (it would be very computationally expensive to try to optimise parameters for each data set), and for improving an algorithm's robustness across a series of different data sets. One downside is that this parameter tuning approach may affect the coverage of the algorithm and hence lead to underfitting.

Within the domain of rainfall derivatives, prices of contracts are updated daily, which also requires the rainfall model to be updated daily, from the arrival of new information. To avoid the unnecessary computational cost of tuning every day, a single parameter set should be sufficient for capturing daily changes. Moreover, there may exist an ad-hoc scenario where a contract may need to be priced, which is not part of the traded cities. Thus, having a robust parameter set is then important for allowing for a more flexible framework. This is a key advantage over MCRP, which requires constant calibration to take into account the change in daily rainfall amounts. For our problem domain we may be pricing for contracts up to 10 months in advance, thus requiring effective long run predictions of rainfall.

Despite the observed issues with the algorithms, we have clearly shown the improvement that can be made by using the proposed data transformation to increase model performance. In general, all algorithms' predictive errors were reduced following the data transformation, and the algorithms are capable of producing models more reflective of the rainfall time series. These results of using the data transformation are a key step forward towards more accurate derivative pricing and will help overcome issues of mispricing.

For GP we can be very encouraged on how our proposed algorithm is able to produce equations more reflective of the rainfall time series, shown in GP's coverage results in full in Table 4.11. We were able to provide coverage results better than SVR and RBF whilst maintaining predictive accuracy. Unlike M5P and M5R that

Table 4.11 The coverage in terms of the number of data points covered by GP for all cities.

Europe				the USA			
Amsterdam	37%	Luxembourg	34%	Atlanta	16%	Kansas	38%
Arkona	85%	Marseille	80%	Boston	31%	Las Vegas	81%
Basel	66%	Oberstdorf	76%	Capehatteras	23%	Los Angeles	92%
Bilbao	35%	Paris	71%	Cheyenne	46%	Louisville	21%
Bourges	35%	Perpignan	57%	Chicago	58%	Nashville	59%
Caceres	100%	Potsdam	75%	Cleveland	16%	New York	50%
Delft	41%	Regensburg	78%	Dallas	12%	Phoenix	81%
Gorlitz	84%	Santiago	36%	Des Moines	59%	Portland	44%
Hamburg	47%	Strijen	22%	Detroit	28%	Raleigh	37%
Ljubljana	57%	Texel	45%	Indianapolis	13%	St Louis	26%
				Jacksonville	46%	Tampa	44%

had excellent coverage, but the resulting model was a poor fit. This gives a better indication where the performance of our algorithm can be built upon. Currently the data sets of the USA are causing a problem, which makes sense as the climate is far more erratic and is prone to much higher extreme rainfall amounts. We can clearly see that the data transformation had a positive effect on our rainfall prediction accuracy and the GP is ranked third behind the two top performing black-box algorithms of RBF and SVR. We will examine the predictive accuracy of GP in more detail in the following section.

4.7.2 Best Algorithm for Rainfall Prediction

One of the goals outlined at the beginning of Section 4.6, is to find out how good our GP is at predicting the level of rainfall. Ideally, we are looking for GP to outperform all techniques, but most importantly it should outperform the currently used approach in rainfall derivatives, i.e. MCRP.

We will now evaluate how well the algorithms performed against each other. We have already presented the predictive accuracy of all algorithms, but we now explore in more detail the performance of GP in comparison to the other benchmarks and determine whether GP statistically outperforms other algorithms. In order to check for any statistical difference, we use the nonparametric test of Friedman (Demšar, 2006) to determine the difference between several related samples. Our null hypothesis is that there are no significant differences between each algorithm for predicting the rainfall. We perform the Friedman test at the 95% confidence level using the results from Tables 4.8 and 4.9 and present the findings in Table 4.12, with significant differences highlighted in bold.

Table 4.12 The Friedman test statistic and the Holm post-hoc results for daily prediction on the left and for the transformed predictions on the right hand side. Values in bold represent a significant difference to the top ranked algorithm.

Daily rainfall prediction Friedman statistic **2.4723x10^{-5}**				Transformed (accumulated) rainfall prediction **1.9351x10^{-19}**			
Algorithm	Mean rank	p-value	Critical value	Algorithm	Mean rank	p-value	Critical value
MCRP (c)	3.0714	-	-	RBF (c)	2.2857	-	-
M5P	3.1786	0.8202	0.0500	SVR	2.3810	0.8399	0.0500
GP	3.5000	0.3633	0.0250	GP	3.0476	0.1060	0.0250
RBF	4.0952	0.0299	0.0167	M5R	4.9524	**0.0000**	0.0167
M5R	4.4762	**0.0029**	0.0125	M5P	4.9524	**0.0000**	0.0125
SVR	4.8333	**0.0002**	0.0100	MCRP	5.0238	**0.0000**	0.0100
KNN	4.8452	**0.0002**	0.0083	KNN	5.3571	**0.0000**	0.0083

From the left-hand side of Table 4.12, we can see that in the daily predictions MCRP ranked first, with KNN coming in last place. Interestingly, RBF and SVR did not perform very well on a daily basis, ranking 4$^{\text{th}}$ and 6$^{\text{th}}$ but this may be explained by the daily predictions requiring very specific parameters for RBF and SVR. We can see from Table 4.12 that there was a significant difference between algorithms, shown by a p-value of 2.4723x10^{-5}, thus at least 1 or more algorithms performed better than the others. In order to determine which one(s) performed better, we use the Holm post-hoc test, using MCRP as the control algorithm. The Holm results show that MCRP was the best algorithm with a mean rank of 3.07 and statistically outperformed M5R, SVR and KNN, as shown by the bold p-values located in Table 4.12. It is encouraging to see the generalisation that GP is able to show against the respective benchmarks by performing equally as well as MCRP, M5P and RBF, where MCRP is the most commonly used method in the rainfall derivatives literature. This result is favourable, because GP does not explicitly deal with wet and dry days unlike the commonly used approach of MCRP. Therefore, demonstrating that even on a hard problem like rainfall, GP can be a successful technique compared to other popular, well established methods such as SVR and RBF.

Looking at the right-hand side of Table 4.12 (for transformed data), shows a very different picture to the algorithms behaviour in terms of overall performance compared to daily predictions. We observe that RBF performs the best with a mean rank of 2.28, with SVR and GP performing very similarly, with mean ranks of 2.38 and 3.05 respectively. Noticeably, we see that in terms of the transformed (accumulated) rainfall prediction, MCRP is ranked 6$^{\text{th}}$ with a mean rank of 5.02. This is in contrast to the daily accuracy where we saw that MCRP performed the best, which highlights the

potential from using a transformation for predicting rainfall. We do observe that KNN performed the worst in both model set-ups, indicating that this algorithm is not suitable for this problem domain. We also see M5P drop from the 2nd in the rank (for daily prediction) to 5th in the rank (for accumulated prediction). We can see from Table 4.12 that there is a significant difference between one or more algorithms, shown by the Friedman test statistic of $1.9351 \text{x} 10^{-19}$. Similarly, we apply the Holm post-hoc test to identify which algorithms outperformed each other, with the results being shown in Table 4.12. We can identify that GP performed equally as well as RBF and SVR, with neither algorithm statistically outperforming GP. Therefore, showing that our GP is able to predict equally with two of the most powerful machine learning algorithms in the literature.

This is a promising result for our GP, because it is able to perform equally well with two of the most respected black-box methods of SVR and RBF. Furthermore, we are able to show that GP is capable of predicting a lower RMSE on average when compared to MCRP, as indicated by the mean ranks. The last important aspect to consider is whether the performance of GP and our other algorithms are affected by climatic effects, which will be discussed next.

4.7.3 Algorithm's Performances According to Climates

The final aspect that we evaluate is whether there is any connection between algorithms and climates. Therefore, we consider how the performance of rainfall predictions from Tables 4.8 and 4.9 compares with the descriptive statistics explaining the climate from Tables 4.2 and 4.3 based on the results from using the data transformation. This will be an interesting comparison, which may help future research by identifying certain patterns that need to be incorporated into future models.

We do not consider looking into the daily predictions, because none of the algorithms (except MCRP) predicted the daily values well and our main research goal is to study the accumulated rainfall amounts. We link this back to our areas of interest indicated in Section 4.2, and we want to investigate to what extent the following statements are true:

- The predictive error is similar between Europe and the USA.
- Drier or wetter climates are associated with a lower predictive error.
- More volatile cities are associated with higher predictive error.
- High rainfall intensities are associated with higher predictive error.

Table 4.13 Mann-Whitney U-test results (p-values) to determine whether each algorithm statistically performed better in one continent.

Set-up	SVR	RBF	M5R	M5P	KNN	GP	MCRP
Daily prediction	0.1779	0.1441	0.1548	0.1779	0.1946	0.1441	0.1779
Transformation prediction	0.2845	0.3198	0.3780	0.3849	0.1861	0.2959	0.4575

This analysis does give an insight into the effectiveness of the sliding window data transformation and whether any other data considerations are necessary. Moreover, if we can identify that certain algorithms are less affected by climate, we can further investigate those algorithms unaffected.

When we consider the performance of algorithms in Europe against the USA, we use the Mann-Whitney U-test to determine whether the predictive error for both daily and after applying the data transformation is affected by the continent. We test at the 5% significance level to establish whether any of our algorithms performs better in one continent. Table 4.13 shows the p-values for the Mann-Whitney U-test. We can clearly see that there is no significant difference between Europe and the USA, with p-values ranging from 0.1476 to 0.1946 for daily predictions and from 0.1861 to 0.4575 for using the transformation. Therefore, we are confident that our results of our previous hypothesis tests are not biased by the performance of one continent over another. Due to there being no significant different, we do not perform further Friedman tests on the USA and Europe separately.

Table 4.14 The linear correlation coefficient (r) and p-value for European cities, in order to determine whether there is sufficient evidence that a relationship exists between a data set property and an algorithm's predictive error, measured by CV(RMSE). The p-value is shown in brackets below the correlation coefficient. Significant relationships ($p < 0.05$) are shown in bold.

Data set property	SVR	RBF	M5R	M5P	KNN	GP	MCRP
% of dry days	0.05	0.08	0.10	0.08	0.33	0.08	0.30
	(0.8197)	(0.7421)	(0.6823)	(0.7262)	(0.1493)	(0.7454)	(0.2049)
Average dry spell	0.21	0.23	0.24	0.23	0.40	0.23	0.39
	(0.3763)	(0.3350)	(0.3106)	(0.3261)	(0.0822)	(0.3315)	(0.0907)
Average wet spell	0.15	0.13	0.10	0.12	-0.17	0.13	-0.10
	(0.5187)	(0.5900)	(0.6617)	(0.6110)	(0.4606)	(0.5811)	(0.6744)
Annual rainfall	0.07	0.10	0.11	0.12	0.30	0.10	0.22
	(0.776)	(0.6898)	(0.6402)	(0.6137)	(0.1965)	(0.6805)	(0.3434)
Volatility	0.24	0.26	0.27	0.25	**0.50**	0.26	**0.47**
	(0.3091)	(0.2652)	(0.2572)	(0.2904)	**(0.0256)**	(0.2745)	**(0.0349)**
Highest intensity	0.40	0.44	**0.45**	0.42	**0.72**	0.42	**0.65**
	(0.0776)	(0.0536)	**(0.0442)**	(0.0670)	**(0.0003)**	(0.0641)	**(0.0019)**
Interquartile range	**-0.57**	**-0.59**	**-0.61**	**-0.58**	**-0.74**	**-0.58**	**-0.70**
	(0.0089)	**(0.0057)**	**(0.0042)**	**(0.0069)**	**(0.0002)**	**(0.0071)**	**(0.0006)**

Table 4.15 The linear correlation coefficient (r) and p-value for the USA cities, in order to determine whether there is sufficient evidence that a relationship exists between a data set property and an algorithm's predictive error, measured by CV(RMSE). The p-value is shown in brackets below the correlation coefficient. Significant relationships ($p < 0.05$) are shown in bold.

Data set property	SVR	RBF	M5R	M5P	KNN	GP	MCRP
% of dry days	-0.3	-0.31	-0.35	-0.31	-0.26	-0.30	**-0.48**
	(0.1818)	(0.1663)	(0.1124)	(0.1628)	(0.2461)	(0.1683)	**(0.0230)**
Average dry spell	-0.40	-0.41	**-0.46**	-0.41	-0.39	-0.42	**-0.47**
	(0.0644)	(0.0550)	**(0.0304)**	(0.0557)	(0.0692)	(0.0539)	**(0.0276)**
Average wet spell	-0.06	-0.07	-0.06	-0.07	-0.16	-0.08	0.22
	(0.8009)	(0.7645)	(0.7759)	(0.7616)	(0.4729)	(0.7362)	(0.3151)
Annual rainfall	**-0.42**	-0.40	-0.33	-0.33	-0.27	-0.37	0.14
	(0.0496)	(0.0656)	(0.1323)	(0.1310)	(0.2216)	(0.0926)	(0.5288)
Volatility	-0.36	-0.37	-0.42	-0.36	-0.35	-0.37	-0.32
	(0.0955)	(0.0861)	(0.0546)	(0.0964)	(0.1107)	(0.0894)	(0.1486)
Highest intensity	**0.46**	**0.48**	**0.51**	**0.48**	**0.43**	**0.49**	0.08
	(0.0298)	**(0.0247)**	**(0.0151)**	**(0.0247)**	**(0.0463)**	**(0.0199)**	(0.7085)
Interquartile range	-0.40	-0.41	**-0.43**	**-0.42**	**-0.45**	**-0.44**	-0.03
	(0.0676)	(0.0606)	**(0.0450)**	**(0.0489)**	**(0.0361)**	**(0.0396)**	(0.9100)

We compare the percentage of wet/dry days, the average spell length, average daily rainfall, volatility, highest intensity and interquartile range for each city against the predictive error for each algorithm, establishing whether a strong correlation exists. The results for both Europe and the USA are presented in Tables 4.14 and 4.15, showing the Pearson product-moment linear correlation coefficient for each pair of algorithm and data set property. Additionally, we include the p-values, in order to determine whether there is a statistically significant relationship between the predictive error and the descriptive statistics. The values highlighted in bold indicate a statistically significant relationship at the 5% level.

Looking at both Table 4.14 and 4.15, we can see a mixed picture of relationships and it appears that there are some strong correlations between climatic aspects and predictive error. In order to assist the comparison against our points of interest, we discuss separately the relationships for European and the USA cities, because we have already determined the significant differences in climate between these two sets of cities in Section 4.2.

When considering if the percentage of dry days is significantly correlated with predictive error, we observe that across Europe, little relationship exists with the correlation coefficient r values ranging between 0.05 to 0.33 looking at Table 4.14. It indicates there is some positive relationship, but not enough for a statistically significant result. On the other hand, in Table 4.15, we observe r values between -0.26 to -0.48 within the USA, indicating that for MCRP there is some correlation at the

5% significance level. Interestingly, with all algorithms having a negative correlation, the drier the climate the smaller the CV(RMSE). This difference in the findings for Europe and the USA makes sense, given that the USA has longer dry spell lengths compared to Europe, which is far more changeable. Therefore, it is easier to predict constant behaviour over time than more sporadic behaviour.

When we consider the volatility we observe a completely different picture. Across Europe, there is a positive correlation with r values ranging between 0.24 and 0.50. In the case of KNN and MCRP there is sufficient evidence to suggest a relationship exists at the 5% significance level, indicating the CV(RMSE) increases with higher levels of volatility. However, we observe the opposite relationship looking at the correlation values from the USA, with r values between -0.32 and -0.42. Unlike Europe, we do not see any algorithm exhibiting a significant relationship, but we see a slight improvement in CV(RMSE) the higher the volatility. One reason for this negative relationship is because the volatility is closely linked with the balance of dry and wet days, which links back to the spell lengths previously discussed. This underlying relationship means that the higher the volatility, the drier the climate is, and we have shown that drier climates are negatively correlated to the CV(RMSE) for cities in the USA.

Finally, when we consider the rainfall intensities, we notice a broadly similar pattern across Europe and the USA. Across Europe, we observe r values in the range of 0.40 to 0.72 with MCRP, KNN and M5R having a significant correlation. Similarly, within the USA we observe r values in the range of 0.08 to 0.51, with all algorithms having a significant correlation except for MCRP. These findings are expected for both geographic areas since we noticed when analysing the data earlier that achieving the highest peaks is very difficult. The relationships noticed with the intensity and volatility are reflected by the interquartile range, which is heavily dependent on both aspects. We witness a reduction in CV(RMSE) from data exhibiting a larger interquartile range, which makes sense for rainfall data series as the cities with larger interquartile ranges have a more consistent climate. We witness this alongside the improvement in rainfall prediction coverage as previously shown in Table 4.10.

Our GP has shown its potential very well at being able to generalise across continents and across different climatic data sets. Only in three descriptive measures was a relationship discovered, where the interquartile range was flagged up in both Europe and the USA and the highest intensity also within the USA. Referring back to our original four bullet points, we found that the predictive error is similar between the USA and Europe, no association between wet and dry days and the volatility. However, we did identify that the intensity does correlate with GP's predictive error. This is a promising result and helps to guide the future improvements that can be made using GP. Going forward in our research, this area to increase coverage should lead to a

non statistical significant relationship between predictive error and our descriptive statistics.

From examining the different climates based on our four areas of interest, we notice that the patterns of results for all algorithms in general are different when we compare results for Europe and the USA. There are two benefits we can see from the analysis. The first one is the benefit the data transformation has across both geographic areas, with no significant difference in the predictive error of all algorithms between Europe and the USA. Secondly, the binary problem of dry and wet days appears to have been satisfactorily solved, since the predictive error is not significantly affected in general regardless of the balance between dry and wet days. However, we do witness that climate does have some impact on predictive error for some algorithms, as shown by the volatility, maximum potential rainfall intensities and the interquartile range. Future research should look into capturing these shocks of the model, which fall outside of the interquartile range, including the known range of the training set. Capturing this behaviour should result in the algorithms performing equally in predictive power across all climates.

The results from this chapter have indicated that the methodology of GP is a positive direction. GP is capable of providing very competitive predictive results. The ability to reconstruct our model is very positive, from a research perspective, and allows us to further tailor our new GP to take into account certain behaviour through the semantic or syntactic structure of a tree. This gives us more flexibility for dealing with the underlying issues that surrounds the time series of rainfall. Currently we have a very good platform to work from, where the equations produced better reflect the rainfall time series, than RBF and SVR. Therefore, we have strong evidence from our results that GP should be used for the remainder of the thesis as our proposed methodology.

4.8 Conclusion

This chapter demonstrates the effectiveness of a data transformation within the problem of rainfall derivatives, dealing with complex data sets which exhibit extreme rainfall values and volatility. This data transformation is based on the use of a sliding window by accumulating the rainfall amount for a long period (e.g. one year), in order to predict the accumulated rainfall amounts (rather than predicting daily rainfall) for such applications as financial securities (e.g. rainfall derivatives). The data transformation allows for patterns that were previously unrecognised and helps deal with the problem of extreme values.

We proposed the use of GP and started to tailor a new GP specifically for the problem of rainfall. GP was proposed as an alternative method to overcome the weaknesses of the commonly used methods within the literature of rainfall derivatives. Strongly-typed Genetic Programming (STGP) was our chosen methodology, because we can influence types to avoid illegal trees being created.

Moreover we applied a range of well established machine learning algorithms and the most commonly used technique within rainfall derivatives, to predict rainfall with and without the proposed data transformation. These were used to act as a benchmark to compare against our proposed GP and our proposed data transformation for the problem of rainfall. We set out three goals to this chapter: (i) extensively testing the data transformation technique on several machine learning techniques, (ii) evaluating the predictive performance of GP against the benchmark machine learning methods, and (iii) evaluating whether the combination of the data transformation and algorithms is affected across varying climates.

We evaluated the predictive error on cities from around Europe and the USA with and without the sliding window data transformation. Our results show that there is sufficient evidence that the data transformation has superior predictive power than predicting the daily amounts. Furthermore, when applied to the transformed data, Radial Basis Functions (RBF), Support Vector Regression (SVR) and GP were the best algorithms in general.

Our proposed GP performed equally as well as RBF and SVR and was able to predict a lower (better) mean rank than MCRP when using the transformed data, showing that GP is a very suitable method. We were also able to identify that GP was capable of producing models more similar to that of the underlying data than RBF and SVR as reported by the level of coverage of our equations.

We did notice that some climatic features do have an affect on predictive error in general across all algorithms, namely the volatility of rainfall, maximum rainfall intensities and the interquartile range of rainfall. Most importantly, we do not witness any geographic difference across all algorithms in predictive error across Europe and the USA. Additionally, the problem of discontinuity within the rainfall time series is satisfactorily solved using the data transformation, given no significant effect from the balance of wet and dry days.

Within the next chapter, we will keep with the methodology of GP which has shown to be very effective and consider a way to increase the predictive accuracy taking into account the coverage of our algorithm.

Chapter 5

The Decomposition Genetic Programming Algorithm

5.1 Introduction

In the last chapter, we introduced our GP to the problem of rainfall prediction and outlined a proposed data transformation to assist the prediction of rainfall within rainfall derivatives. For our chosen methodology of GP, we observed that it was able to perform similarly to other well established algorithms after the data transformation and from the creation of additional features. Moreover, we showed that it outperformed MCRP, the most commonly used approach for rainfall derivatives.

Leading on from the previous chapter, the data transformation helped to avoid potential issues that exist within daily rainfall time series. The two key characteristics of extreme volatility and discontinuity of daily data were overcome by the use of a sliding window to accumulate rainfall. This enabled GP and other algorithms to work with the daily data to predict accumulated levels of rainfall, which contract prices are based on. However, from our previous experimentations, GP was unable to predict the full dynamic range of rainfall for the majority of cities. Two aspects were observed, first it was hard to capture the volatile and irregular periods, where all techniques tried to flatten the predictions. Secondly, the extreme dry and wet periods of rainfall were not adequately covered by our equations.

These two aspects resulted in GP only covering around 50% of data points, which is unacceptable for an algorithm that is required to price. As the concept of pricing is based on the chance of an event occurring in the future, by only covering a relatively narrow band of rainfall points, we are limiting our pricing potential. The limiting factor is that we are observing that the very wet and the dry periods are unrepresented,

meaning our algorithm predicts that such events carry a probability close to zero. Therefore, leading to potential mispricing.

In this chapter, the goal is to produce more accurate rainfall amount predictions by proposing a solution to the two observations from the previous chapter to avoid mispricing. Thus, we propose a new methodology for predicting rainfall amounts, consisting of two components through the evolutionary process. The motivation allows us to break the problem of rainfall prediction into smaller partitions and to tackle the problem of rainfall in a divide and conquer manner. As a consequence, this reduces the level of difficulty when dealing with data sets with high volatility and extreme values. Therefore, rather than attempting to tackle the problem as a whole, the decomposed GP (DGP) will evolve multiple equations. Our decomposition approach divides the problem of rainfall prediction into subproblems, where each subproblem is represented by producing rainfall equations that predict within a specified range of the accumulated level of rainfall. Thus, each regression equation is tailored to a specific partition of the data, rather than 'one-size-fits-all'. We construct our algorithm to be suitable within the application field of rainfall derivatives, but it could be easily adapted to suit other application fields as well.

By producing multiple equations to predict within its specific partition, GP should only evaluate one equation on a given day. In order to the evaluate the equations, we first need to decide which partition the day is within. Therefore, we require a decision based on classifying which state we are in on a given day. We propose within this chapter to extend the GP individual representation and operators (based on trees), with a Genetic Algorithm (GA) linear representation and operators. Therefore, creating a hybrid GP with an additional GA component to assist the predictions and simplify the regression problem. We restrict the problem to one of three classes (low, medium, high rainfall), in order to facilitate the final prediction to be performed by the GP component of the hybrid DGP. We opt for a GA as our chosen classification method, because both algorithms are from the same paradigm, with both consisting of the same evolutionary procedure. This allows for more cohesion between the methods allowing for both methods to learn off each over the evolution.

By the proposal of a new DGP hybrid algorithm for the rainfall prediction process, we aim to explore whether the new algorithm will have a positive effect on the predictive accuracy.

This chapter is laid out as follows. In Section 5.2, we outline how we decompose our rainfall time series. In Section 5.3, we outline the GA proposed in this chapter to assist our GP for decomposition. In Section 5.4, we outline how our GP and GA exchange information to form their hybrid relationship within DGP. In Section 5.5, we outline the experimental set-up for our proposed hybrid DGP algorithm. In Section

5.6, we discuss the results of our decomposition algorithm DGP and the effectiveness of our GA component. In Section 5.7, we discuss the effect that DGP has had on our coverage including any climatic impacts. Finally, in Section 5.8, we conclude this chapter.

5.2 Decomposed Genetic Programming

Within this section, we outline how we achieve the initial decomposition and how we break the problem down into smaller subproblems. The initial structure of the proposed DGP is built on the GP presented in the previous chapter along with the same function and terminal set, and using the same data variables as already outlined.

Our proposed DGP consists of a number of individuals split into two separate populations, a GP part and a GA part. The GP part consists of b expression trees, where nodes represent functions or terminals as usual in our GP. For our implementation we define b to equal 3, such that we have 3 GP equations to predict low, medium and high rainfall amounts. The GA part consists of a linear chromosome with a string of n rules, each with g genes.

5.2.1 Decomposing Rainfall Amounts

In order to decompose rainfall, we propose partitioning the data into three different partitions (low, medium and high rainfall amounts), thus simplifying the prediction process. More could be considered, but we anticipate that three partitions are sufficient based on the analysis of previous experimentations. We identify the low and high levels of rainfall received little coverage by a single regression equation. We discuss the process of splitting the data in Section 5.2.2. Then, in Section 5.2.3, we will discuss how GP will be adapted to create multiple regression equations one for each partition.

5.2.2 Splitting the Data

As we are creating a separate equation for low, medium and high levels of rainfall, we require two constants to split the data into three partitions. We refer to these two constants as a lower criterion LC and upper criterion UC, as shown by Figure 5.1. Thus, anything below LC is considered low rainfall, anything between LC and UC is considered medium rainfall and above UC is considered high rainfall. We allow for each individual of DGP to have its own LC and UC, instead of having a fixed constant applied to all individuals within the population. By assuming a fixed constant we cannot determine whether the value of LC and UC is optimal and would need a way of

estimating prior to running our DGP. Therefore, we allow the LC and UC to evolve along with the GP and GA part of DGP, by encoding the LC and UC values within the linear representation of a GA individual. The values of LC and UC are considered based on the training data of each individual city. One aspect that would be open to future research is considering a dynamically changing LC and UC, taking into account the uncertainty around certain periods of time.

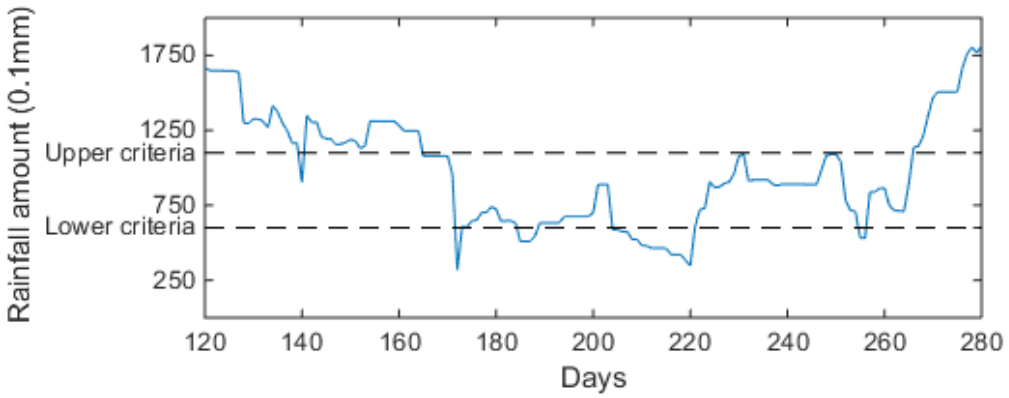


Fig. 5.1 Rainfall data split into three partitions according to a lower criterion and upper criterion.

5.2.3 Genetic Programming Trees

By using the information from a given LC and UC, the rainfall time series can be split into three partitions. As shown by Figure 5.1, we require an equation to predict within the boundaries specified, thus we map each partition to a particular GP branch (b_n). It is controlled by a classification method (decision criteria), shown by Figure 5.2. The concept is that a rainfall equation should be capable of predicting all points within its specified range and is evolved based on its ability to do so, whilst restricting behaviour outside of this range. Thus, having independent equations allow the GP to evolve each branch to maximise the predictive performance within each partition. Keeping the branches independent is required, given the patterns and information will differ within each partition. To ensure that b_1 does not consider information from b_2 or b_3, we keep each branch independent and separate throughout the evolutionary process. To achieve this behaviour, we create a crossover and mutation operator that can only evolve the same branch amongst individuals. The procedure is similar to the standard genetic operators, but is performed branch wise, rather than once per individual. Using tournament selection to randomly select two parents based on the performance of the complete problem, DGP chooses a random node/leaf from one branch and combine with the same branch from the other parent. This process is repeated for all branches.

We choose to keep the same parents for the three crossovers associated with the three branches, rather than select a new parent for each branch, to avoid too much disruption and randomness during the evolutionary process. Mutation follows the same procedure, a parent is chosen and one node/leaf on each branch undergoes single-node mutation.

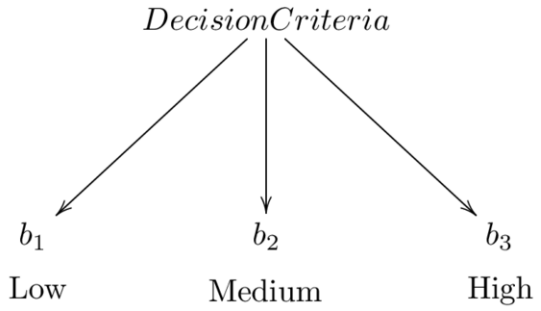


Fig. 5.2 The representation of the decision criteria and the three branches for regression. Upon evaluation of the decision criteria, this leads to one of the three branches; each branch is a different GP tree, representing a different rainfall prediction equation.

Elitism places into the next generation a new individual formed by a combination of branches b_1, b_2 and b_3 based on the predictive performance of each branch. In order to create the elite individual, we merge the best from b_1, b_2 and b_3 across the entire population, creating a new individual consisting of the three best branches from the previous generation. Within this framework we use b_1 to represent low rainfall, b_2 to represent medium rainfall and b_3 to represent high rainfall, as shown by Equation 5.1.

$$\text{GP individual} \begin{cases} b_1 & \text{if } r_t \leq LC \\ b_3 & \text{if } r_t \geq UC \\ b_2 & \text{otherwise.} \end{cases} \tag{5.1}$$

The general algorithm of DGP can be found in Algorithm 1. One variable that is unknown from Algorithm 1 and Equation 5.1 is r_t, which is the actual level of rainfall. Within our framework of DGP, this is the crucial variable to compare against the proposed LC and UC. To do so, we propose the use of decision criteria, which is a classification technique to determine the branch to evaluate, discussed in the next section.

5.3 The Genetic Algorithm Component of the DGP

In this section, we outline the GA to classify each data point into the correct partition of rainfall amount. Firstly, we introduce the representation of our GA in Section 5.3.1.

Algorithm 1 Decomposing rainfall amounts

```
1: P ← Number of individuals in population.
2: B ← Number of partitions.
3: for Individual i = 1,...,P do
4:     for Branch b = 1,...,B do
5:         initialise(branch_b^i).
6:     end for
7:     Set LC_i.
8:     Set UC_i.
9: end for
10: for Generation g = 1,...,G do
11:     for Individual i = 1,...,P ∀ t do
12:         if r_t ≤ LC_i then
13:             Evaluate b_1.
14:         else if r_t ≥ UC_i then
15:             Evaluate b_3.
16:         else
17:             Evaluate b_2.
18:         end if
19:         Calculate fitness.
20:     end for
21:     Breed.
22: end for
```

Then, we discuss the fitness criteria to be used in Section 5.3.2. Finally, the breeding of our GA is described in Section 5.3.3.

5.3.1 Decomposing the Problem With the GA Component

Levels of rainfall prediction requires rebuilding the decomposition back into the original problem. Within our framework, DGP needs to choose which branch to evaluate on a given day. In order to do so, we propose using a GA with a linear representation, as part of a hybrid DGP individual, to classify. Figure 5.1 shows the importance of classifying correctly, especially when considering the impact of misclassifying by more than one class. For example, if the actual rainfall amount is within the high rainfall partition (amounts $>$ 110mm) and a classifier predicts low rainfall, then this points to the wrong branch (tree) in the GP-part representation of the DGP individual, leading to an equation predicting much lower rainfall amounts, possibly in the range of less than 50mm, thus causing an error of at least 50%.

The GA-part of the DGP individual representation consists of 5 genes; predictor, period, lower criterion, upper criterion and order. Our GA linear representation is essentially a rule list for a given period of time within a year. Each rule has the

same number of outcomes as the number of specified partitions. Keeping the rules consistent keeps the understanding of the rules very intuitive and comprehensive. The rules consists of making decisions based on the same attributes used within the GP's terminal set, given in the previous chapter. The rules are kept very simple and are based on a single attribute along with a $>$ or $<$ operator and a constant. For each period of time only one rule is present with three outcomes. For efficiency reasons, we do not consider chaining rules involving the logical operators such as AND, OR and NOT. Based on the outcome of the rule, the GA decides the respective branch to evaluate.

Table 5.1 All the possible values for each gene, except for order. As we have a rule for each month, only the total number of days per month is given.

Genes of the GA-part of an individual	
Predictor	$\{r_{t-1}, r_{t-2} \dots r_{t-11}\}$, $\{r_{y-1}, r_{y-2} \dots r_{y-10}\}$
Period	31, 30 and 28
Lower Criterion (*predLC*)	0.05 - 0.65
Upper Criterion (*predUC*)	0.35 - 0.95

The predictor refers to one of the attributes used within the GP's terminal set, e.g. r_{t-1}, r_{t-2} and so on. Period refers to the number of days covered by a rule e.g., a value of 31 would cover the next 31 days. Within our methodology we keep the period consistent and apply a rule for each month of the year, however, variable period lengths can also be considered. The lower and upper criteria are the decisions threshold for choosing which class to predict, *predLC* and *predUC* respectively based on the predictor value. For our experimentations, we define the *predLC* and *predUC* in terms of percentiles of the training set, but this can be modified accordingly to any real number or function. The complete list (excluding order) of values of the genes in the GA is specified in Table 5.1. The order is one of the unique permutations of the three branches, given below:

Order reference

$$\begin{array}{cccccc} \begin{bmatrix} 1 \end{bmatrix} & \begin{bmatrix} 2 \end{bmatrix} & \begin{bmatrix} 3 \end{bmatrix} & \begin{bmatrix} 4 \end{bmatrix} & \begin{bmatrix} 5 \end{bmatrix} & \begin{bmatrix} 6 \end{bmatrix} \\ \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} & \begin{bmatrix} b_1 \\ b_3 \\ b_2 \end{bmatrix} & \begin{bmatrix} b_2 \\ b_1 \\ b_3 \end{bmatrix} & \begin{bmatrix} b_2 \\ b_3 \\ b_1 \end{bmatrix} & \begin{bmatrix} b_3 \\ b_1 \\ b_2 \end{bmatrix} & \begin{bmatrix} b_3 \\ b_2 \\ b_1 \end{bmatrix} \end{array} \tag{5.2}$$

where each permutation corresponds to the following criteria:

$$\begin{bmatrix} predictor < predLC \\ predLC < predictor < predUC \\ predictor > predUC \end{bmatrix}.$$

For example order 3, whenever the predictor is less than *predLC*, we classify medium rainfall (b_2). If it is greater than *predUC*, we classify high rainfall (b_3), otherwise low rainfall (b_1). For order 5, whenever the predictor is less than *predLC*, we classify high rainfall (b_3). If it is greater than *predUC*, we classify medium rainfall (b_2), otherwise low rainfall (b_1).

Due to rainfall features exhibiting very complex and chaotic processes, it is highly unlikely that a single predictor can classify accurately. Such low probability in classification motivates us to allow larger number of rules to be created throughout the year, which is able to reduce complexity in rainfall prediction, hence the period criteria. To best describe the characteristics of each month throughout each year, we set 12 rules, one for each corresponding month. However, the number of rules can be adjusted according to the user's or model's preferences. Furthermore, the orders of *predLC* and *predUC* are important aspects within the classification process, because the same predictor could be used in a different month under different criteria. Figure 5.3, shows a sample representation of the above description, where we demonstrate the rules for January, February and December.

$$\overbrace{\left[\mathrm{r}_{t-1}, 31, 37, 91, 2, \mathrm{r}_{y-3}, 28, 22, 77, 2\right.}^{\text{January}}\overbrace{\ldots}^{\text{February}}\overbrace{\left.\mathrm{r}_{t-1}, 31, 11, 64, 6\right]}^{\text{December}}$$

Fig. 5.3 An example of a GA for 3 out of 12 months.

The classification rules for January, February and December are shown in Equation 5.3, Equation 5.4 and Equation 5.5 respectively, showing the impact of a different order (by cross-referencing Equation 5.2 with Figure 5.3) and the different criteria to split the predictor. The period refers to the number of days the rules cover and is expressed in each equation as the days covered during a year. Therefore, the rules shown below are the same for every day in the respective months.

$$\text{January (Days 1-31)} \begin{cases} b_1 & \text{if } r_{t-1} \leq 37^{th}\text{percentile}, \\ b_2 & \text{if } r_{t-1} \geq 91^{st}\text{percentile}, \\ b_3 & \text{otherwise}, \end{cases} \tag{5.3}$$

$$\text{February (Days 32-60)} \begin{cases} b_1 & \text{if } r_{y-3} \leq 22^{nd}\text{percentile}, \\ b_2 & \text{if } r_{y-3} \geq 77^{st}\text{percentile}, \\ b_3 & \text{otherwise}, \end{cases} \tag{5.4}$$

$$\text{December (Days 335-365)} \begin{cases} b_3 & \text{if } r_{t-1} \leq 11^{th}\text{percentile}, \\ b_1 & \text{if } r_{t-1} \geq 64^{th}\text{percentile}, \\ b_2 & \text{otherwise.} \end{cases} \tag{5.5}$$

After the inclusion of our GA component into our DGP, we modify our general DGP algorithm as shown in Algorithm 2:

Algorithm 2 Adding our decision criteria into DGP

```
P ← Number of individuals in population.
B ← Number of partitions.
for Individual i = 1,...,P do
    for Branch b = 1,...,B do
        initialise(branch_b^i).
    end for
    Set LC_i.
    Set UC_i.
    Initialise GA.
end for
for Generation g = 1,...,G do
    for Individual i = 1,...,P ∀ t do
        Evaluate individual i of the GA.
        Choose branch.
        Evaluate branch.
        Calculate fitness.
    end for
    Breed.
end for
```

5.3.2 Fitness Criteria

Each individual of the hybrid DGP has the output of its GP component (which is partly determined by the values of the GA-component genes) evaluated using RMSE. However, we also need to compute the fitness of the GA-part of an individual separately. To compute the GA-part's fitness we use Kendalls tau (τ) correlation coefficient, which is used to measure the rank correlation between two variables taking into account the natural ordering of nominal classes. This measure helps deter from misclassifying by more than one class. Kendalls tau is given by:

$$\tau_B = \frac{n_c - n_d}{\sqrt{(n_0 - n_1)(n_0 - n_2)}}, \text{where}$$
$$n_0 = \frac{n(n-1)}{2}, n_1 = \sum_i \frac{t_i(t_i-1)}{2}, n_2 = \sum_j \frac{u_j(u_j-1)}{2},$$

where n_c = Number of concordant pairs, n_d = Number of discordant pairs. t_i = Number of tied values in the i^{th} group of ties for the first quantity and u_j = Number of tied values in the j^{th} group of ties for the second quantity. Let $(p_1, a_1), (p_2, a_2), \ldots, (p_n, a_n)$ be a set of observations, in our case the predicted class and the actual class, where n refers to the number of training instances. A pair is concordant if the ranks for (p_i, a_i) and (p_j, a_j) both agree, such that $p_i > p_j$ and $a_i > a_j$ or $p_i < p_j$ and $a_i < a_j$ and vice versa if discordant.

5.3.3 Individual Evaluation and Breeding of the GA

Each individual of the GA is evaluated based on the Kendalls correlation mentioned above, which returns a value in the range of $[-1, 1]$. A value of 1 represents a perfect agreement between rankings of predicted and actual classes. Once the population is evaluated, the selected individuals undergo genetic operators. The GA-part of the individuals can undergo point mutation and a variety of crossover techniques. The mutation procedure chooses a random point within the individual and replace it with a random variable or value that is of the same type. Therefore, one cannot replace a predictor (e.g. r_{t-4}) with *predLC*, only with another predictor (e.g. r_{y-5}). We cover the process of elitism in Section 5.4, because it requires the interaction between the GP and GA components of the hybrid DGP. We opt for tournament selection to select the parents for breeding and discuss the variety of crossover methods below. All these methods are used in our DGP and are chosen at random to promote a good diverse balance of individuals.

Multiple Split Points

We apply multiple split point methods, similar to the one-point crossover, where we choose a random point and take one section from the first parent and the other section from the second. However, given our chromosome is 60 genes (12 sets of 5 genes) in length and to increase the mixing of individuals, we choose a random number s in the range [1,12] and create s splits in random locations in our chromosome. Therefore, creating individuals with a mix from two parents through random split points.

Multiple Rule Split

We create a technique where we perform crossover on rules i.e., choosing a crossover point located at the boundary between two adjacent rules, rather than arbitrary split points (which could be inside rules). One possible advantage is that we keep the rules intact and do not cause too much destruction of each GA individual. Therefore, we consider crossover on our 12 rules. We choose which rules to crossover through assigning a probability to the crossover process. The first step is to choose the number of rules s randomly in the range [1,11] to select from each parent and from that we assign the probability. For example, if s is 6, then the probability is 50% of selecting a rule from either parent and if s is 3 then the probability is 25% of choosing a rule from the first parent. We then sequentially move along each rule and sample a uniform value to decide which parent to choose from based on the probability identified.

Single Split Within Rule

An alternative is the combination of the two methods above. Sequentially moving along each rule, we choose at random a gene in the range [0,5]. A value of 0 means that no split is required and to use all of the material from the first parent. A value of 1 indicates that the first 4 genes are from the first parent and the 5^{th} gene would be from the second parent. We repeat this process for all rules.

Uniform Crossover

The final alternative for crossover is adapting a uniform crossover procedure, where we apply a probability (0.5) for each gene within each rule. Then, for each gene, we choose at random whether to pick from the first or second parent for the new offspring, when creating each child.

5.4 Integrating the GP and GA Components

In this section, we outline three aspects of the integration of the GP-part and GA-part of the individual representation of the hybrid DGP, namely: penalising the regression trees, elitism, and the evolution of the *LC* and *UC* criteria to partition the data for classification.

5.4.1 Penalising GP Regression Trees

Following the decomposition approach, it is key that each regression equation (a GP tree) predicts values within its respective partition. For example, it makes little sense

for an equation to be responsible for the low rainfall class, predicting values in medium and high rainfall class. Therefore, we implement a penalty function based on the distance away from the correct partition, as shown in Figure 5.4. To integrate the GP and the GA components and to maximise the usefulness of this idea, we implement a simple check before choosing whether to penalise or not. The GP-related penalty only applies to situations where the GA has correctly classified. Therefore, we are not penalising the GP for making a wrong prediction given that the GA was at fault. This modification should influence the GP to predict within a range similar to that of the specified partition. From Figure 5.4 any deviation denoted by the dashed vertical lines is penalised by Equations 5.6.

$$
\begin{aligned}
&\text{Actual class is low,}\\
p^{new} &= \begin{cases} p^{old} + m(p^{old} - LC) & \text{if } \; c_p = c_a \text{ AND } p^{old} > LC \\ 0 & \text{otherwise.} \end{cases}\\
&\text{Actual class is medium,}\\
p^{new} &= \begin{cases} p^{old} - m(LC - p^{old}) & \text{if } \; c_p = c_a \text{ AND } p^{old} < LC \\ p^{old} + m(p^{old} - UC) & \text{if } \; c_p = c_a \text{ AND } p^{old} > UC \\ 0 & \text{otherwise.} \end{cases}\\
&\text{Actual class is high,}\\
p^{new} &= \begin{cases} p^{old} - m(UC - p^{old}) & \text{if } \; c_p = c_a \text{ AND } p^{old} < UC \\ 0 & \text{otherwise,} \end{cases}
\end{aligned}
\tag{5.6}
$$

where p^{new} represents the predicted rainfall amount by GP after penalising and p^{old} represents the predicted rainfall amount originally predicted by GP. Parameter m represents a scaling function on the penalty, c_p is the predicted class and c_a is the actual class (i.e. the classified rainfall amount). UC and LC are the upper and lower criteria for splitting the data into its respective classes. For example, let us assume that $c_p = c_a$. If GP predicts 1000 tenths of mm (p^{old}), where the UC is 1100 and m is 2, but the true class is high rainfall. We would then update p^{new} by $1000 - 2 \times (1100 - 1000)$, hence p^{new} is penalised to 800. The idea is for GP to deter from this individual given the large penalty effect.

An alternative method for handling with the prediction in the wrong partition is to have a wrapper to round the equation up or down to the nearest partition. However, compared to the idea of the penalisation, this may encourage poor performers to be selected for future generations by forcefully rounding poor performers. An example is an equation for the medium partition, predicting values excessively large or low. By

penalising the DGP individual, they are further deterred, but through rounding they are comparable to equations predicting within the same range. Within Algorithm 2, this step would be inserted before calculating the predictive accuracy.

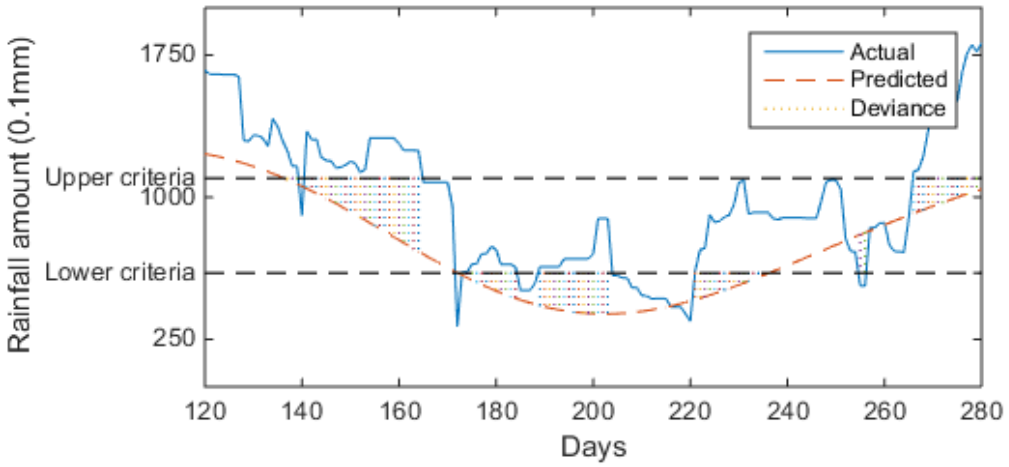


Fig. 5.4 The distance from the predicted amount to either the lower bound or upper bound when GP predicts a rainfall amount in the wrong partition. The deviance is then used to calculate a penalty.

5.4.2 Elitism Merging Different Individuals

The use of elitism in our evolution process relies on exchanging information to create the best individual to carry forward to the next generation. Typically, elitism takes the best GP trees and the best GA genes separately and put them into the next generation. However, due to the close integration between the GP and the GA components of an individual, we create our own elitism strategy.

The first consideration is mentioned in Section 5.2.3, where we merge the best performing branches b^{rank}_{number} together in ranking order. The elitism strategy proposed perceives the DGP as combination of three separate populations of individuals and the GA-part as a separate population as well. Each population of branches gets its fitness evaluated based on how it is able to solve its respective subproblem via the RMSE. Additionally, each GA individual has its fitness evaluated based on the Kendalls tau correlation rank. Through this procedure, we aim to promote the best branches to create an elite individual. Thus, the best branch b^1_1 merges with b^1_2 and b^1_3. Note that, the GA component and the GP branches are jointly responsible for achieving a better RMSE. For instance, b^1_1, b^1_2 and b^1_3 may not come from the same parent using the same GA-based partition rules. Potentially, we may have 3 different GA-based rule lists influencing the performance. Thus, we need an intermediate step to decide

which of the GA-based rule lists is responsible for the best overall individual using all 3 branches. Therefore, we evaluate in turn each GA-based rule list (i.e., each GA individual) associated with the best branches merged together. We also evaluate the best GA individual overall based on its Kendalls tau correlation rank, which may not be attached to any branch. After re-evaluating the newly merged offspring, the partition rule list that is responsible for returning the best fitness in RMSE is moved into the next generation as part of the offspring.

This helps to evolve the partition rules that can perform the best classification across the training period, assisting the GP to solve the regression problem.

5.4.3 Evolution of *LC* and *UC*

The last aspect of the hybrid DGP is the process of evolving *LC* and *UC* (our decomposition approach). This criteria is required for the GP component to construct regression equations (trees) to predict within each data partition and for the GA-based rule lists to classify into the relevant classes. Recall that each individual consists of three GP regression trees and a GA-based rule list.

The use of *LC* and *UC* is to split the initial data into the three partitions, such that GP creates an equation to predict within each partition and the GA assists by selecting the corresponding branch to evaluate on a given day. By evolving the criteria that binds the two hybrid parts together, we hope to find an optimal point where both the GP and GA part can minimise the RMSE on the whole problem. We do not directly influence the behaviour of the *LC* and *UC* and leave it up to the GA through the evolutionary process to modify as necessary. To ensure the split points for decomposition are evolved, during crossover the two parents' *LC* and *UC* values undergo uniform crossover to create the future offspring. With uniform crossover on two points there is a $\frac{1}{2}$ chance of both *LC* and *UC* coming from the same parent and $\frac{1}{2}$ chance of a mixture, as shown in Figure 5.5. Moreover, we do allow these points to be mutable as well, but instead of mutating using a uniform selection of values, we opt for the number to be normally distributed around the old value with a variance of 0.1. The motivation is that we want to modify the split point by a small amount, otherwise, mutation can be too disruptive by changing a *LC* value. For example, changing a value from 0.02 to 0.53 has a massive effect on our performance. Unlike the previous two aspects, this aspect is more subtle and directly affects the performance of both the GP and the GA, and helps guide the evolution process of both in turn.

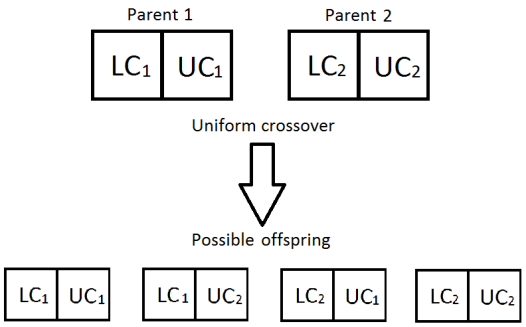


Fig. 5.5 An example showing the breeding of the *LC* and *UC* from two parents using uniform crossover.

5.4.4 Alternative Classification Techniques

An extension to test the effectiveness of the combination of the GA and the GP is to consider the use of other classification techniques to act as the decision criteria. The GA part is modified to replace the rule list with a different classification method. Therefore, our GA is simplified by containing an *LC* and *UC* and a classification method to perform the selection for which branch to evaluate for our DGP. We use the following classification techniques: Support Vector Machines (SVM), Radial Basis Function (RBF), Repeated Incremental Pruning to Produce Error Reduction (RIPPER), Discriminant Analysis (DA) and Naive Bayes (NB). We provide the predictive accuracy along with the performance after applying GP in Section 5.6. We anticipate the higher the classification accuracy the lower the RMSE, but we may observe the GA and GP intertwined throughout evolution may lead to better results.

5.5 Experimental Set-Ups

The goal of our experimentation is to establish whether the use of DGP outperforms a standard GP and whether we are able to overcome issues raised in Chapter 4. These issues are the lack of coverage on average across all data sets and correlation existed between predictive accuracy and climatic features. The motivation for improving the rainfall predictive accuracy is based on the final application pricing rainfall derivatives. As it is emphasised previously, producing more accurate rainfall predictions should lead to more accurate pricing.

We have identified four key aspects to investigate for the DGP within this chapter. The first is the performance against the benchmarks and GP presented in Chapter 4. The second is the performance of different classification techniques and the GA proposed within this chapter, based on how accurately they are able to classify into one

of the three classes of rainfall. The third is how each classification helps the overall problem of rainfall prediction. We anticipate that the higher the correlation score the lower the RMSE is, as more classes are accurately predicted. Moreover, we anticipate that GA carries a significant advantage based on the sharing of information throughout the evolutionary process to help evolve both the GA and GP part. We may observe that the GA performs worse than other classification techniques, but makes up with a lower RMSE. The fourth is whether or not DGP's predictive performance is correlated with the climatic aspects outlined in Chapter 4.

Testing for this chapter is performed in the same manner for the previous chapter, with the algorithms tested on the testing set of Jan-01-2015 until Dec-31-2015, by training on the period Jan-01-2005 until Dec-01-2014. Furthermore, as we are proposing a new algorithm, we perform the tuning of its parameters using iRace using the same method as proposed in Chapter 4, by tuning on the same subset of cities not included in our test cases. We present the best tuning parameter set found by iRace in Table 5.2.

Table 5.2 The optimal configuration of DGP found by iRace. Parameters with a * represent parameters used by both the GP-part and GA-part of DGP.

GP Parameters	DGP
Max depth of tree	8
Population size	1000*
Crossover	99%*
Mutation	30%*
Primitive	32%
Terminal/Node bias	64%
Elitism	3%*
Number of generations	70*
ERC negative low	-288.42
ERC negative high	-224.31
ERC positive low	210.43
ERC positive high	432.23

We will run our DGP and compare its predictive performance against the original GP proposed in the previous chapter, the machine learning benchmark methods used in the previous chapter and MCRP. The same 42 data sets will be used as per the previous chapter and the average predictive accuracy on the test set over 50 runs for DGP will be given.

We will then consider the impact of changing the underlying classification technique from the GA to one of the techniques given in Section 5.4.4. We will first consider the classification performance and will observe how the DGP performs when

Table 5.3 Optimal parameters using iRace for the three benchmark classification algorithms: SVM, RBF and RIPPER.

SVM		RBF		RIPPER	
SVM Type	C-SVC	Minimum SD	28.3	Folds	4
Cost	0.85	Clusters	2	Weight	7.01
Gamma	0.34	Ridge	0.541	Optimisations	3
Kernel Type	RBF			Prune tree	False

the decision process is controlled via a different algorithm. The classification accuracy of our GA and benchmarks will be based on a predefined set of upper and lower criteria. To avoid bias in comparisons, we will use the same set for all classification techniques. Our results will be based on randomly selecting 100 upper and lower criteria to partition our data and we will report the average results. If the algorithm is non-deterministic (which is the case for the GA and the RBF) then we will run the technique 50 times on the same split points. Afterwards, we will show the performance of the DGP with the new decision techniques and will compare against the DGP with the GA as the decision criteria. Table 5.3 shows the optimal configurations for our classification techniques found by iRace based on the same validation set of Jan-01-2014 until Dec-31-2014.

5.6 Results

Within this section, we outline the results of how our proposed DGP performs against the benchmarks from the previous chapter. Moreover, we test the classification ability of our GA against other well known techniques and how this impacts our DGP's predictive accuracy.

5.6.1 Predictive Accuracy of DGP

We present the findings for all algorithms in Tables 5.4 and 5.5. Note that the results for all other techniques are the same from the previous chapters, as the same testing set is used for comparison, however we have expressed the results in terms of the RMSE measure.

From looking at Tables 5.4 and 5.5, we can observe that the DGP is able to outperform the original GP proposed in the last chapter fairly consistently, as shown by the underlined values. The percentage improvement is approximately 8% on average over the 42 cities, which is a positive result. Some noticeable results from the cities

Table 5.4 RMSE results for Europe of the proposed DGP against the predecessor GP and other benchmarks. Values in bold represent the best algorithm for each city. Underlined values indicate the lowest predictive error between DGP and our originally proposed GP.

City	DGP	GP	SVR	RBF	M5R	M5P	KNN	MCRP
Amsterdam	430.28	430.88	432.94	**422.24**	492.97	467.45	473.41	625.15
Arkona	296.66	272.16	235.08	**221.70**	306.28	319.63	283.35	414.26
Basel	303.90	293.26	**269.35**	309.50	387.07	374.88	277.00	373.18
Bilbao	774.16	783.58	787.14	**729.30**	878.28	885.94	949.61	1020.70
Bourges	304.95	322.63	295.80	**289.09**	397.57	386.09	325.89	425.89
Caceres	357.46	371.71	**318.10**	320.09	370.20	439.19	472.00	385.82
Delft	**455.86**	476.01	512.31	483.94	562.26	503.30	518.83	732.90
Gorlitz	257.82	330.30	**253.04**	329.80	363.04	406.87	272.46	304.21
Hamburg	332.21	330.08	318.09	**298.62**	349.39	355.81	325.05	476.22
Ljubljana	483.81	499.10	**455.43**	483.10	666.23	689.07	1183.07	642.49
Luxembourg	331.67	390.91	364.88	370.43	463.58	509.14	**300.47**	384.44
Marseille	372.13	395.81	**334.03**	337.08	516.69	432.31	718.98	429.98
Oberstdorf	**436.68**	563.98	468.04	475.31	561.20	554.66	679.59	682.52
Paris	268.95	287.83	**260.68**	265.59	303.47	316.97	260.76	356.38
Perpignan	396.12	407.00	**383.94**	398.48	494.05	469.72	1492.69	445.26
Potsdam	231.30	243.18	**202.30**	222.87	291.61	283.94	344.93	362.87
Regensburg	269.36	277.66	271.41	270.83	335.37	335.78	**240.96**	334.62
Santiago	**860.67**	1034.02	989.13	914.70	1127.44	1268.65	1379.51	1068.89
Strijen	**458.05**	507.86	523.21	520.90	529.29	569.76	548.57	715.82
Texel	399.90	412.91	**393.25**	396.05	412.54	423.88	491.18	611.57

Table 5.5 RMSE results for the USA of the proposed DGP against the predecessor GP and other benchmarks. Values in bold represent the best algorithm for each city. Underlined values indicate the lowest predictive error between DGP and our originally proposed GP.

City	DGP	GP	SVR	RBF	M5R	M5P	KNN	MCRP
Atlanta	**764.76**	799.73	857.53	800.03	868.86	851.39	919.44	1159.81
Boston	**380.26**	417.14	388.20	400.56	624.63	535.08	966.33	492.04
Cape Hatteras	**866.71**	938.51	1023.58	968.16	1062.44	1111.34	1171.89	1304.73
Cheyenne	342.81	339.62	344.60	353.29	353.58	**297.42**	443.19	455.40
Chicago	453.91	498.05	443.42	**433.04**	550.44	559.64	716.73	655.31
Cleveland	**474.60**	534.76	527.59	538.55	562.07	587.37	532.82	676.48
Dallas	**1070.64**	1223.09	1283.32	1248.00	1328.54	1273.65	1437.85	1415.62
Des Moines	**553.35**	582.78	564.35	498.90	678.43	701.12	1020.52	805.94
Detroit	**358.96**	387.56	381.69	363.39	429.08	437.55	450.84	486.93
Indianapolis	**834.96**	889.52	891.26	898.76	897.16	919.82	948.64	1047.53
Jacksonville	663.42	630.29	573.48	574.81	607.41	**560.05**	731.03	793.05
Kansas	**667.69**	700.58	691.15	701.07	743.62	835.84	1038.90	917.71
Las Vegas	104.68	97.71	99.57	**86.26**	112.69	151.49	101.03	120.97
Los Angeles	323.20	308.52	**213.25**	235.13	281.23	315.29	403.75	276.18
Louisville	**784.55**	894.04	899.69	895.82	915.28	981.09	1042.56	1172.18
Nashville	431.00	467.94	424.96	**418.49**	556.37	533.51	469.66	698.28
New York	454.65	505.07	**390.83**	427.85	764.03	531.26	683.14	551.05
Phoenix	175.79	148.53	160.45	**128.06**	217.63	175.75	133.11	182.97
Portland	**661.44**	787.17	777.84	729.94	819.96	841.69	1059.11	969.16
Raleigh	485.30	**469.13**	543.97	480.19	613.22	626.52	561.72	846.81
St Louis	**838.33**	933.92	1010.01	984.12	981.52	881.00	1091.14	1241.65
Tampa	1125.76	1219.45	1184.48	**1112.94**	1278.94	1355.24	1408.39	1491.53

are Oberstdorf and Gorlitz, where the predictive error reduced by 22%. We also note that comparing the DGP against the GP, the former performs better in 33 data sets. Moreover, the DGP is able to predict the best out of all approaches 16 times, which again shows the real performance gains that can be realised by breaking the process of rainfall down and solving subproblems. By comparison, the second best algorithm regarding the number of victories overall the SVR, which achieved the lowest RMSE in 11 cities.

In order to determine the effectiveness of our proposed DGP and to test whether the above results are statistically significant, we compare the eight algorithms by using the Friedman test. This is a non-parametric test based on the mean rank of all algorithms across all data sets (cities) (Demšar, 2006). Similar to the previous chapter, we use the same null hypothesis, that all algorithms should perform similarly across the testing set at the 95% confidence level. The results of the Friedman hypothesis test can be found in Table 5.6, where we also include the mean ranks based on the results from Tables 5.4 and 5.5. As our Friedman test statistic is significant at the 5% level, we use the Holm post-hoc test to compare the control (best) algorithm against each of the others.

Recall from the previous chapter we were able to show that the GP is able to keep up with the two most powerful black-box techniques of the SVR and the RBF, by not being statistically outperformed. From looking at the mean rank within Table 5.6, the DGP is now ranked top, achieving the lowest RMSE on average against all other algorithms. It shows that the use of decomposition helps to reduce the average predictive error. For our hypothesis test our control algorithm is the DGP.

Table 5.6 The mean rankings of all algorithms, and the Friedman test statistic with the best performing algorithm (DGP) being the control measure. Values in bold represent a significant difference.

Friedman statistic	**1.11x10^{-32}**		
Algorithm	Mean ranking	p-value	Critical value
DGP	2.52	-	-
RBF	2.55	0.96	0.050
SVR	2.67	0.79	0.025
GP	3.62	0.04	0.017
M5P	5.81	**7.90x10^{-10}**	**0.013**
M5R	5.83	**5.96x10^{-10}**	**0.010**
KNN	6.00	**7.85x10^{-11}**	**0.008**
MCRP	7.00	**5.56x10^{-17}**	**0.007**

Table 5.6 shows the DGP as the control measure statistically outperforming all algorithms except for the SVR, the RBF and the GP at the 95% confidence level. The original GP is not significantly outperformed, even though the DGP predicts more accurately in 33 (out of 42) cities against GP. Here, we can see the DGP performs better in terms of mean rank than the top black-box methods (the RBF and the SVR) and the GP. It also statistically outperforms all other algorithms. Therefore, we are able to show that the predictive errors is reduced by the use of decomposition by comparison of the mean rank. Additionally, the runtime of DGP is only 4% greater than our original GP.

5.6.2 Classification Accuracy of the GA

We now investigate how effective the GA is at classifying for the Decomposition GP (DGP) algorithm. In order to determine the GA's effectiveness, we compare it against other well-established techniques as a benchmark. The results can be found in Tables 5.7 and 5.8 based on the same randomly chosen set of LC and UC.

Table 5.7 Classification accuracy for Europe shown as a percentage of correctness on the test set. Values in bold show the best algorithm for each city.

Data	GA	SVM	RBF	RIPPER	DA	NB
Amsterdam	**51.10**	47.87	48.92	44.01	39.63	48.82
Arkona	46.04	**50.55**	43.81	44.32	39.60	45.35
Basel	53.56	43.45	**63.12**	55.86	36.67	41.65
Bilbao	49.03	51.25	46.59	**52.81**	41.27	51.51
Bourges	45.85	47.17	**51.18**	44.42	23.03	46.53
Caceres	52.27	40.79	63.69	58.40	52.77	**68.54**
Delft	**53.63**	50.33	48.28	45.79	27.63	37.28
Gorlitz	36.02	40.00	46.77	**47.45**	25.58	40.26
Hamburg	46.19	**51.02**	44.45	47.52	32.24	49.18
Ljubljana	45.46	49.51	44.88	**50.87**	41.55	49.19
Luxembourg	**46.27**	35.60	44.43	38.87	29.61	38.86
Marseille	32.25	46.15	**49.33**	48.55	39.10	40.78
Oberstdorf	47.69	55.88	**59.81**	50.42	33.70	41.35
Paris	50.74	**51.41**	49.45	47.90	28.43	40.79
Perpignan	53.35	**57.35**	55.00	50.93	30.41	45.50
Potsdam	57.33	47.99	**60.41**	47.65	53.98	56.26
Regensburg	39.54	47.05	**53.28**	46.24	38.85	49.59
Santiago	46.80	46.47	**52.18**	44.21	39.17	49.11
Strijen	**49.88**	48.36	42.52	47.36	27.46	30.45
Texel	**59.32**	53.85	59.05	53.29	45.41	47.74

Table 5.8 Classification accuracy for the USA shown as a percentage of correctness on the test set. Values in bold show the best algorithm for each city.

Data	GA	SVM	RBF	RIPPER	DA	NB
Atlanta	49.64	**49.68**	46.55	42.21	25.44	27.98
Boston	**45.63**	36.20	42.25	40.43	28.19	41.32
Cape Hatteras	**64.77**	61.06	60.53	47.13	52.12	56.26
Cheyenne	43.46	**66.50**	57.11	57.28	48.10	46.65
Chicago	30.50	43.45	42.64	40.31	32.33	**45.93**
Cleveland	43.45	37.65	**45.90**	44.24	37.38	35.15
Dallas	42.39	**43.21**	41.79	31.70	39.05	29.36
Des Moines	40.95	53.16	56.32	44.84	48.35	**57.21**
Detroit	39.21	37.10	35.76	37.58	39.65	**40.87**
Indianapolis	36.40	39.79	**50.89**	39.71	41.74	40.88
Jacksonville	48.28	56.36	**58.86**	53.13	46.12	45.03
Kansas	45.61	**58.40**	50.76	49.00	48.91	46.83
Las Vegas	**74.71**	68.74	62.50	57.58	54.55	57.90
Los Angeles	75.70	65.08	74.21	72.14	73.54	**77.83**
Louisville	37.18	**43.08**	33.23	32.95	36.65	34.77
Nashville	41.04	**52.28**	49.45	50.56	25.28	38.65
New York	50.05	35.90	**58.08**	52.97	30.81	47.37
Phoenix	**63.03**	60.92	58.47	58.28	48.50	49.38
Portland	59.38	56.87	54.39	53.27	63.89	**72.47**
Raleigh	47.26	**61.10**	56.48	45.44	42.68	54.48
St Louis	46.14	47.04	45.81	40.79	46.76	**48.82**
Tampa	**69.78**	63.80	65.87	53.23	58.90	51.05

In Tables 5.7 and 5.8, we can observe that our GA performs well, just behind the best algorithms of the RBF and the SVM. More precisely, the GA, the RBF and the SVM are the winners in 10, 11 and 11 cities, respectively. The experimental set-up of this is to test the robustness of each algorithm, which is why the average percentage of correctness for most algorithms appears to be near 50% accuracy. The results here are not directly the same as they are inside the DGP algorithm. The class boundaries specified by LC and UC are randomly selected and are not optimised. One issue with choosing random LC and UC for decomposition is that the chance of it being optimal is slim and does impact performance. This shows the importance of the step outlined in Section 5.4.3 of evolving the LC and UC criteria. Ideally, the algorithm has ability to perform well with a non optimal splitting of data. Considering the range of all classification techniques, in most cases our GA is very competitive, which is a positive sign. The random selection of the criteria is necessary to avoid bias and to allow for a fair comparison across all classification techniques.

Table 5.9 The Friedman test statistic along with the results of the Holm post-hoc test at the 95% confidence level, with the best performing algorithm (RBF) being the control value. Values shown in bold represent a significant difference in classification accuracy against the control algorithm.

Friedman statistic	**1.7925x10^{-10}**		
Algorithm	Mean rank	p-value	Critical value
RBF	2.64	-	-
SVM	2.76	0.771	0.050
GA	3.14	0.221	0.025
NB	3.57	0.023	0.017
RIPPER	3.81	**0.004**	**0.013**
DA	5.07	**2.702x10^{-9}**	**0.010**

In order to determine whether there are any significant differences between classification techniques, we perform the Friedman test at the 95% confidence level and show the results in Table 5.9. We observe a statistical difference, as can be seen by the Friedman statistic of 1.7925x10^{-10}, which is much less than the 5% significance level. Therefore, one or more classification algorithms significantly outperforms at least one other algorithm.

From the perspective of our GA, we observe that it is not significantly outperformed by the best performing classification algorithm of the RBF. We believe that the better the classification accuracy is, the better the performance of the DGP gives, given by classifying more data points correctly. Therefore, based on the mean rank, we expect under this assumption of the RBF performing the best compared to our DGP with the GA. However, a key difference is that the GA rules evolve alongside the GP equations, whereas the other classification algorithms are fixed throughout the GP's evolution. We may observe a substantial number of misclassifications throughout the evolutionary process, which may hinder the generalising ability of the DGP.

5.6.3 Performance Under Different Decision Criteria

We now examine the predictive performance of the DGP when we use an alternative classification algorithm. We look to examine two aspects. Firstly, if using a technique that improves the classification accuracy has a greater effect on lowering the RMSE of the DGP. Secondly, whether in the final generation of the DGP the decision criteria that maximised the classification accuracy is used by the best performing individual (lowest RMSE) of the DGP.

Table 5.10 The average RMSE for Europe after applying different classification algorithms. The best results for each city are shown in bold.

Data	GP + RBF	GP + SVM	GP + GA	GP + NB	GP + RIPPER	GP + DA
Amsterdam	458.08	454.12	**430.28**	454.72	458.38	448.52
Arkona	**274.71**	300.34	296.66	310.63	318.23	304.76
Basel	296.91	309.92	303.90	307.70	306.36	**283.30**
Bilbao	775.86	765.18	774.16	**716.64**	777.88	813.80
Bourges	**297.57**	298.79	304.95	324.28	325.02	313.73
Caceres	381.91	366.68	**357.46**	380.95	368.83	357.60
Delft	449.07	458.60	455.86	**438.45**	455.54	472.91
Gorlitz	258.90	241.11	257.82	256.12	**249.78**	254.75
Hamburg	343.14	342.77	**332.21**	342.11	344.04	349.72
Ljubljana	480.23	517.53	483.81	483.71	461.51	**454.73**
Luxembourg	320.13	319.20	331.67	329.25	**315.32**	335.52
Marseille	372.39	349.76	372.13	**344.93**	345.45	368.97
Oberstdorf	408.51	456.07	436.68	446.42	439.39	**404.85**
Paris	287.88	274.95	**268.95**	283.77	288.58	278.23
Perpignan	382.34	373.74	396.12	**366.41**	384.24	410.38
Potsdam	240.27	243.10	231.30	232.27	242.19	**228.13**
Regensburg	254.41	258.34	269.36	266.96	264.46	**254.09**
Santiago	**800.94**	823.75	860.67	925.48	880.12	890.02
Strijen	**428.41**	440.96	458.05	449.67	439.73	436.80
Texel	**380.10**	389.62	399.90	384.54	383.62	428.45
Mean rank	**3.10**	3.40	3.45	3.65	3.90	3.50

Table 5.11 The average RMSE for the USA after applying the different classification algorithms. The best results for each city are shown in bold.

Data	GP + RBF	GP + SVM	GP + GA	GP + NB	GP + RIPPER	GP + DA
Atlanta	747.63	756.04	764.76	725.30	**711.84**	740.13
Boston	373.80	**360.33**	380.26	398.70	387.64	390.79
Cape Hatteras	866.19	861.94	866.71	914.99	918.71	**839.84**
Cheyenne	**327.90**	348.16	342.81	351.11	346.03	357.28
Chicago	475.15	472.52	**453.91**	473.70	456.91	482.14
Cleveland	497.05	483.95	**474.60**	492.87	485.28	483.90
Dallas	1021.28	**992.06**	1070.64	1022.03	1022.25	1048.16
Des Moines	515.94	542.17	553.35	526.90	**512.57**	566.19
Detroit	385.02	**348.05**	358.96	356.91	359.10	373.28
Indianapolis	783.53	783.61	834.96	**772.34**	797.47	887.73
Jacksonville	710.52	668.00	**663.42**	702.10	678.75	688.30
Kansas	631.03	685.32	667.69	**622.69**	625.69	695.07
Las Vegas	107.07	104.76	**104.68**	107.59	106.54	105.84
Los Angeles	339.04	324.53	323.20	313.83	345.95	**300.48**
Louisville	790.51	802.83	784.55	793.42	**762.19**	811.07
Nashville	426.17	438.93	431.00	427.25	**401.39**	436.82
New York	442.15	450.15	454.65	439.65	**421.19**	463.42
Phoenix	186.23	182.08	175.79	168.83	165.35	**164.72**
Portland	**629.23**	705.43	661.44	693.06	691.20	658.93
Raleigh	490.54	**450.50**	485.30	491.41	508.69	497.72
St Louis	869.01	891.23	**838.33**	845.20	874.63	881.25
Tampa	**1112.70**	1151.77	1125.76	1139.83	1153.00	1133.19
Mean rank	3.36	3.55	**3.09**	3.50	3.33	4.18

Tables 5.10 and 5.11 show the average RMSE of the DGP using each classification algorithm, along with the mean ranks located at the bottom of the tables. Similar to our previous experimentation, we run the DGP for 50 times and initialise 1000 randomly generated *LC* and *UC* combinations (population size) pairing them to a DGP individual throughout evolution. We present the order of algorithms according to the classification accuracy from Tables 5.7 and 5.8, with the RBF performing the best and the DA performing the worst. Interestingly, the respective RMSE of each algorithm is not too dissimilar between first and last place and considering the mean ranks. One aspect we notice is that there does appear to be a negative correlation that exists across the table looking at the mean ranks, where the higher the classification accuracy, the lower the RMSE error, which is exactly as we anticipated. Taking the combined mean rank across both tables, we notice that the RBF ranks first (3.23), the GA ranks second (3.26), the SVM ranks third (3.48) and the remaining algorithms ranked in the same order as per the classification accuracy. The GA is the only algorithm to increase its rank on its predictive error relative to its rank on the classification accuracy (from third to second).

In order to determine whether this relationship exists between the classification accuracy and the predictive error, we calculate the Pearson product-moment linear correlation coefficient to measure the strength of the relationship. We observe based on the results provided in Tables 5.7, 5.8, 5.10 and 5.11, that we obtain a coefficient value of -0.8924. This indicates a strong negative linear relationship between classification accuracy and predictive error. We obtain a p-value of 0.0167, which is less than the 5% level and is concluded that a relationship does exist.

We do notice that the GA has an irregular effect on the RMSE. It is an anomaly that does not fit the trend. The GA's average predicted error is similar to the classification technique ranked first (RBF), despite classifying third.

We perform the Friedman hypothesis test to determine whether there is a significant effect on the RMSE from the use of different decision criteria. We discover the p-value is 0.6675, which is greater than the 5% significance level and accept the null hypothesis. Although we observe a trend that is consistent with our previous analysis of the classification accuracy, there is not enough evidence to suggest that one decision criteria leads to a significant change in RMSE.

Our DGP algorithm leads to a reduction in RMSE by having a more accurate classification technique. For further analysis, we also consider the effect of each classification technique has on the standard deviation of our DGP predictions, which are shown in Tables 5.12 and 5.13.

Table 5.12 The standard deviation for Europe of different set-ups of DGP using different classification algorithms.

Data	GP + RBF	GP + SVM	GP + GA	GP + NB	GP + RIPPER	GP + DA
Amsterdam	4.63	5.68	10.55	5.53	5.32	4.80
Arkona	6.82	4.30	6.34	6.42	4.37	5.16
Basel	4.12	3.89	11.33	6.79	5.48	6.53
Bilbao	5.74	5.57	8.38	6.05	4.12	4.47
Bourges	3.87	6.56	11.28	3.16	4.75	5.21
Caceres	4.66	3.52	12.55	5.39	5.24	6.34
Delft	4.91	4.42	11.25	4.67	5.36	7.30
Gorlitz	3.84	5.55	8.75	6.47	6.46	4.42
Hamburg	4.36	4.05	8.39	5.14	4.81	7.25
Ljubljana	5.65	3.18	12.44	5.00	5.40	3.26
Luxembourg	6.73	6.87	7.61	6.91	4.09	7.08
Marseille	3.29	6.43	6.30	7.05	6.38	7.32
Oberstdorf	4.82	6.94	7.85	4.05	4.89	4.13
Paris	5.19	5.54	5.99	7.10	6.40	3.70
Perpignan	4.55	4.04	8.76	5.01	7.01	5.26
Potsdam	3.13	3.30	6.22	6.06	3.92	3.44
Regensburg	5.58	3.52	9.78	5.30	5.18	5.34
Santiago	3.93	5.15	9.20	3.39	4.64	5.93
Strijen	6.53	6.21	6.81	4.49	4.45	5.74
Texel	5.98	4.92	6.31	5.55	5.69	5.17

Table 5.13 The standard deviation for the USA of different set-ups of DGP using different classification algorithms.

Data	GP + RBF	GP + SVM	GP + GA	GP + NB	GP + RIPPER	GP + DA
Atlanta	3.49	4.47	11.22	3.42	6.54	4.17
Boston	4.67	5.82	6.03	5.43	6.05	6.70
Cape Hatteras	3.56	4.56	13.59	3.54	6.18	3.31
Cheyenne	5.15	6.38	6.71	5.97	3.86	4.24
Chicago	6.91	4.99	12.09	7.36	5.97	6.20
Cleveland	4.07	5.72	6.00	4.50	7.38	6.67
Dallas	5.21	3.92	11.80	3.37	5.54	4.29
Des Moines	4.30	4.56	8.89	3.39	5.99	4.21
Detroit	5.77	4.53	7.92	7.04	5.61	3.37
Indianapolis	6.42	4.38	10.14	6.81	7.29	6.77
Jacksonville	6.93	3.46	8.83	3.89	6.02	3.51
Kansas	4.05	3.24	5.66	4.88	6.30	3.17
Las Vegas	3.40	4.84	10.46	5.89	4.00	4.04
Los Angeles	3.66	4.96	10.01	4.52	6.00	7.25
Louisville	6.48	3.66	5.75	3.61	7.25	4.68
Nashville	3.46	4.96	9.27	5.47	4.41	7.05
New York	6.22	5.83	13.47	3.84	3.17	6.01
Phoenix	4.95	5.42	13.01	3.62	4.19	5.08
Portland	5.62	6.07	8.44	3.71	6.60	6.60
Raleigh	3.42	5.48	7.80	4.38	5.11	5.04
St Louis	3.68	4.33	7.67	4.63	3.84	6.62
Tampa	3.14	4.97	11.73	5.34	4.29	3.65

From Tables 5.12 and 5.13, we can identify why the performance generally fitted the negative correlation between RMSE and classification accuracy. Here we witness that all classification techniques, except the GA, tend to increase the robustness of the GP, indicated by the lower RMSE. However, we see that the standard deviation does increases when using our GA respective to the other algorithms. This is quite an interesting discovery for our DGP, where we observe that keeping consistent decision criteria helps to improve the stability of our DGP's performance, since the same model is used for all algorithms except for our GA.

In the special case of our GA, we can have many rules sets explaining the same LC and UC class threshold combination. It adds more randomness into our model and hence reflects with a larger spread of results. Whereas, under all other classification algorithms the outcome of using a certain LC and UC combination is fixed across all DGP generations. We discover the best combination of LC and UC is evolved much more efficiently with the final generation of the DGP by having a more similar LC and UC values. In comparison to our GA, we observe a more mixed set of LC and UC values. In both cases, the mutation from the previous generation is excluded.

For interest, we consider whether the LC and UC that returned the highest classification accuracy from the final generation of the DGP is responsible for the lowest RMSE of our final DGP individual. Tables 5.14 and 5.15 show the best overall classification accuracy on average from the final generation of DGP and in brackets, the classification accuracy of the individual that minimised the RMSE of DGP. This analysis helps to understand classification part of DGP, which may indicate why the individual with the best classification accuracy does not always lead to a lower RMSE.

Tables 5.14 and 5.15 show in almost all cases, the DGP tend to choose the individual with the best classification accuracy, except for our GA. This is interesting as it appears that one of the benefits is the relationship of our GA evolving alongside that of the GP. This indicates the potential for the GP part to be overfitting on the incorrect predictions from the classification algorithm, given that there is only a single model for each LC and UC combination. Alternatively, there may exist a problem of early convergence, as we notice little diversity in the LC and UC of each individual in the final generation. On the other hand, in the final generation the GA has many different classification outcomes with more diverse combinations of LC and UC. This analysis indicates that through the evolution of our GA-part (outlined earlier in Section 5.4), provides the DGP to learn from more frequently changing information to avoid early convergence and to explore different classification rules.

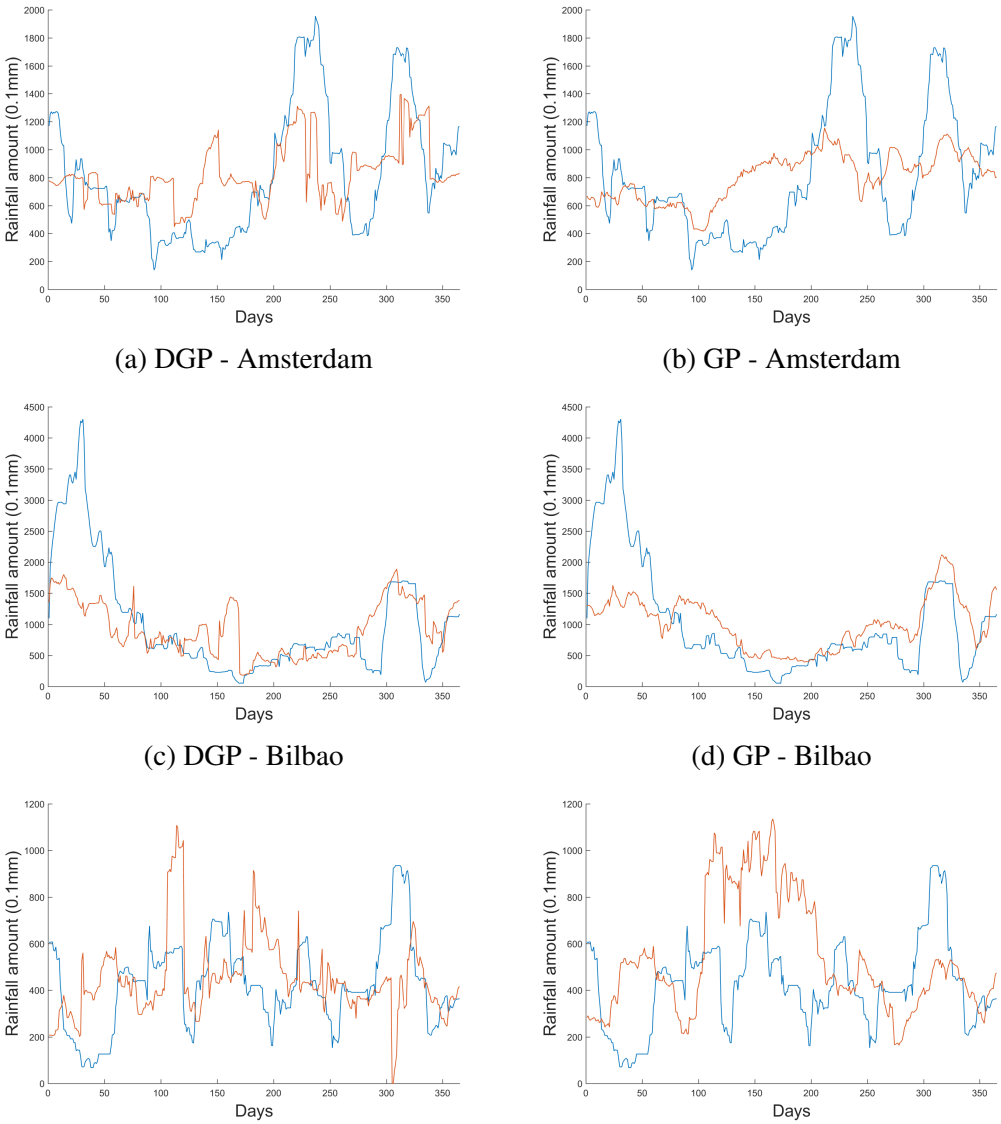
The results show that the GA is competitive with the SVM and the RBF (Table 5.9), and we find that the GA is computationally much more efficient than all classification algorithms. Therefore, we continue with GA being our classification part of DGP.

Table 5.14 The average classification accuracy in the final generation of DGP for Europe, with the average classification accuracy that provided the lowest RMSE for DGP in brackets.

Data	GP + RBF	GP + SVM	GP + GA	GP + NB	GP + RIPPER	GP + DA
Amsterdam	0.717 (0.713)	0.797 (0.797)	0.756 (0.716)	0.735 (0.731)	0.750 (0.749)	0.831 (0.830)
Arkona	0.807 (0.803)	0.766 (0.765)	0.704 (0.643)	0.715 (0.715)	0.849 (0.845)	0.759 (0.753)
Basel	0.828 (0.821)	0.865 (0.862)	0.707 (0.652)	0.713 (0.713)	0.845 (0.845)	0.775 (0.768)
Bilbao	0.701 (0.700)	0.825 (0.817)	0.781 (0.732)	0.797 (0.796)	0.773 (0.773)	0.678 (0.676)
Bourges	0.761 (0.756)	0.763 (0.762)	0.832 (0.756)	0.836 (0.828)	0.817 (0.816)	0.735 (0.732)
Caceres	0.798 (0.791)	0.843 (0.838)	0.853 (0.806)	0.847 (0.841)	0.800 (0.800)	0.835 (0.833)
Delft	0.818 (0.817)	0.775 (0.775)	0.822 (0.771)	0.792 (0.784)	0.784 (0.784)	0.716 (0.715)
Gorlitz	0.682 (0.680)	0.691 (0.687)	0.679 (0.633)	0.646 (0.644)	0.649 (0.647)	0.672 (0.670)
Hamburg	0.865 (0.858)	0.781 (0.777)	0.846 (0.786)	0.704 (0.700)	0.810 (0.808)	0.794 (0.791)
Ljubljana	0.810 (0.805)	0.834 (0.834)	0.764 (0.701)	0.718 (0.712)	0.733 (0.733)	0.660 (0.654)
Luxembourg	0.845 (0.837)	0.743 (0.743)	0.701 (0.631)	0.848 (0.841)	0.861 (0.854)	0.761 (0.757)
Marseille	0.682 (0.680)	0.703 (0.699)	0.648 (0.592)	0.660 (0.653)	0.610 (0.604)	0.615 (0.614)
Oberstdorf	0.797 (0.792)	0.727 (0.722)	0.788 (0.747)	0.821 (0.817)	0.786 (0.786)	0.733 (0.726)
Paris	0.820 (0.812)	0.835 (0.834)	0.770 (0.693)	0.735 (0.734)	0.745 (0.739)	0.839 (0.838)
Perpignan	0.800 (0.798)	0.674 (0.669)	0.798 (0.750)	0.838 (0.829)	0.774 (0.769)	0.734 (0.730)
Potsdam	0.794 (0.794)	0.712 (0.712)	0.701 (0.666)	0.721 (0.714)	0.845 (0.837)	0.799 (0.798)
Regensburg	0.746 (0.746)	0.652 (0.647)	0.696 (0.651)	0.734 (0.732)	0.660 (0.654)	0.627 (0.621)
Santiago	0.815 (0.809)	0.854 (0.854)	0.742 (0.700)	0.739 (0.735)	0.709 (0.704)	0.807 (0.799)
Strijen	0.724 (0.720)	0.775 (0.774)	0.768 (0.705)	0.799 (0.792)	0.734 (0.734)	0.652 (0.650)
Texel	0.897 (0.892)	0.871 (0.864)	0.811 (0.762)	0.824 (0.822)	0.793 (0.787)	0.703 (0.699)

Table 5.15 The average classification accuracy in the final generation of DGP for the USA, with the average classification accuracy that provided the lowest RMSE for DGP in brackets.

Data	GP + RBF	GP + SVM	GP + GA	GP + NB	GP + RIPPER	GP + DA
Atlanta	0.752 (0.750)	0.830 (0.824)	0.737 (0.665)	0.734 (0.734)	0.721 (0.720)	0.709 (0.702)
Boston	0.769 (0.763)	0.803 (0.800)	0.808 (0.758)	0.787 (0.784)	0.769 (0.766)	0.724 (0.722)
Cape Hatteras	0.661 (0.661)	0.707 (0.707)	0.785 (0.735)	0.806 (0.804)	0.753 (0.749)	0.692 (0.689)
Cheyenne	0.720 (0.713)	0.745 (0.738)	0.805 (0.750)	0.773 (0.767)	0.761 (0.757)	0.797 (0.793)
Chicago	0.732 (0.730)	0.735 (0.727)	0.672 (0.625)	0.714 (0.707)	0.667 (0.663)	0.616 (0.615)
Cleveland	0.653 (0.648)	0.720 (0.720)	0.771 (0.704)	0.765 (0.764)	0.739 (0.738)	0.711 (0.711)
Dallas	0.806 (0.806)	0.850 (0.842)	0.733 (0.670)	0.703 (0.699)	0.720 (0.715)	0.629 (0.625)
Des Moines	0.776 (0.772)	0.680 (0.674)	0.696 (0.637)	0.720 (0.718)	0.689 (0.688)	0.617 (0.612)
Detroit	0.551 (0.548)	0.553 (0.550)	0.617 (0.577)	0.583 (0.579)	0.604 (0.598)	0.568 (0.567)
Indianapolis	0.699 (0.694)	0.761 (0.757)	0.675 (0.632)	0.703 (0.702)	0.661 (0.655)	0.649 (0.647)
Jacksonville	0.841 (0.836)	0.836 (0.828)	0.726 (0.665)	0.739 (0.739)	0.869 (0.861)	0.824 (0.818)
Kansas	0.790 (0.788)	0.794 (0.787)	0.856 (0.795)	0.845 (0.845)	0.843 (0.838)	0.809 (0.808)
Las Vegas	0.872 (0.867)	0.958 (0.948)	0.827 (0.766)	0.848 (0.841)	0.800 (0.793)	0.796 (0.790)
Los Angeles	0.878 (0.871)	0.997 (0.990)	0.866 (0.796)	0.897 (0.894)	0.847 (0.842)	0.845 (0.837)
Louisville	0.629 (0.627)	0.638 (0.634)	0.641 (0.593)	0.678 (0.674)	0.622 (0.615)	0.639 (0.634)
Nashville	0.727 (0.722)	0.803 (0.802)	0.785 (0.714)	0.799 (0.799)	0.761 (0.760)	0.659 (0.657)
New York	0.820 (0.815)	0.786 (0.782)	0.853 (0.780)	0.818 (0.813)	0.822 (0.815)	0.848 (0.848)
Phoenix	0.702 (0.701)	0.867 (0.862)	0.816 (0.740)	0.773 (0.765)	0.804 (0.801)	0.805 (0.805)
Portland	0.813 (0.807)	0.746 (0.745)	0.843 (0.774)	0.821 (0.815)	0.830 (0.825)	0.835 (0.829)
Raleigh	0.728 (0.721)	0.732 (0.729)	0.819 (0.777)	0.838 (0.832)	0.776 (0.773)	0.700 (0.696)
St Louis	0.795 (0.791)	0.765 (0.760)	0.696 (0.646)	0.712 (0.706)	0.670 (0.668)	0.588 (0.583)
Tampa	0.755 (0.754)	0.804 (0.800)	0.836 (0.760)	0.864 (0.861)	0.786 (0.780)	0.778 (0.772)



(a) DGP - Amsterdam

(b) GP - Amsterdam

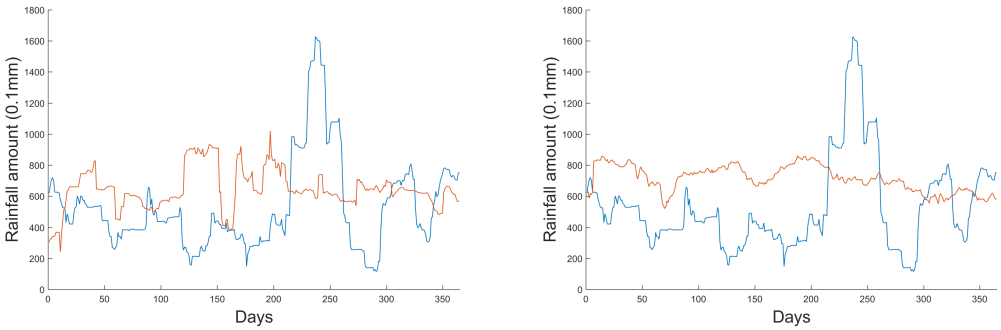
(c) DGP - Bilbao

(d) GP - Bilbao

(e) DGP - Gorlitz

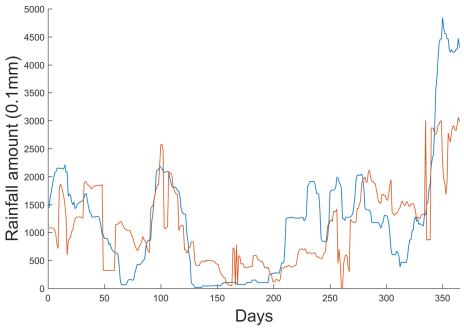
(f) GP - Gorlitz

Fig. 5.6 Rainfall time series for Amsterdam, Bilbao and Gorlitz on the testing set Jan-01-2015 until Dec-31-2015 for DGP (left) and GP (right). The blue line is the actual accumulated level of rainfall and the red line is the rainfall level predicted by the best individual from training over 50 runs.

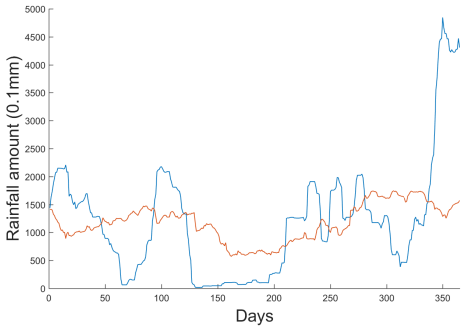


(a) DGP - Luxembourg

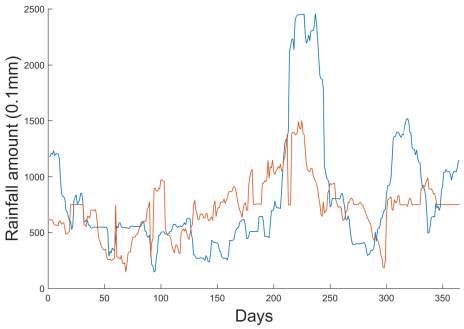
(b) GP - Luxembourg



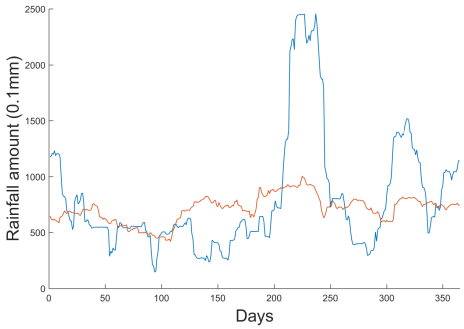
(c) DGP - Santiago



(d) GP - Santiago

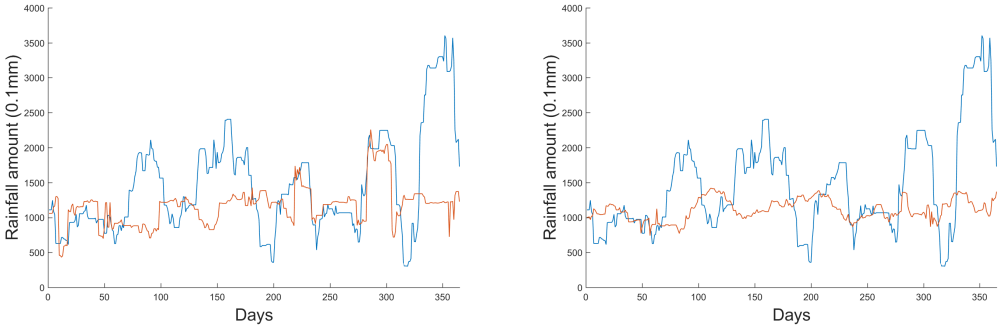


(e) DGP - Strijen



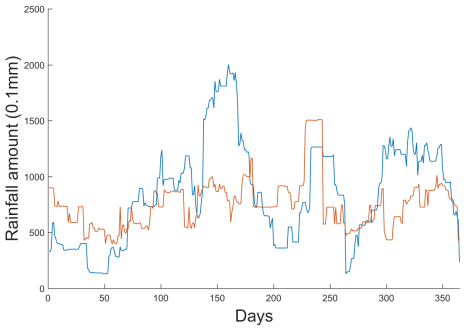
(f) GP - Strijen

Fig. 5.7 Rainfall time series for Luxembourg, Santiago and Strijen on the testing set Jan-01-2015 until Dec-31-2015 for DGP (left) and GP (right). The blue line is the actual accumulated level of rainfall and the red line is the rainfall level predicted by the best individual from training over 50 runs.

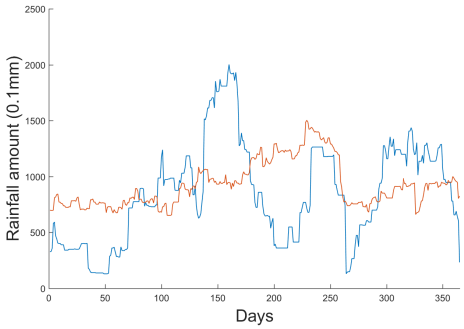


(a) DGP - Atlanta

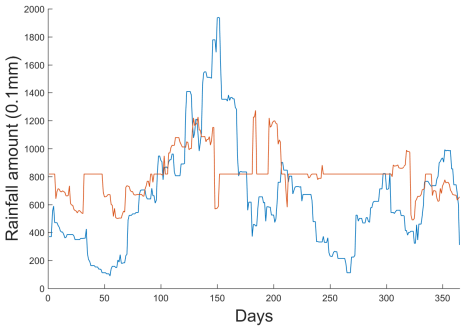
(b) GP - Atlanta



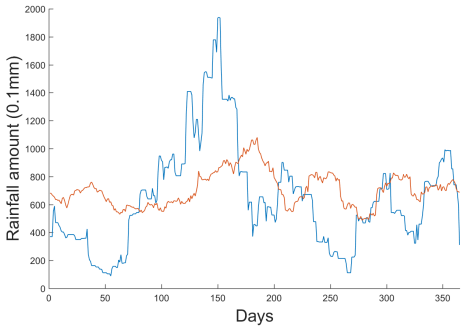
(c) DGP - Chicago



(d) GP - Chicago

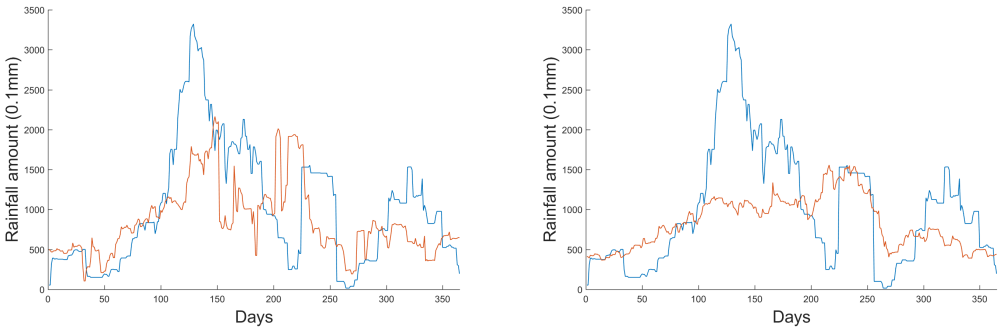


(e) DGP - Detroit



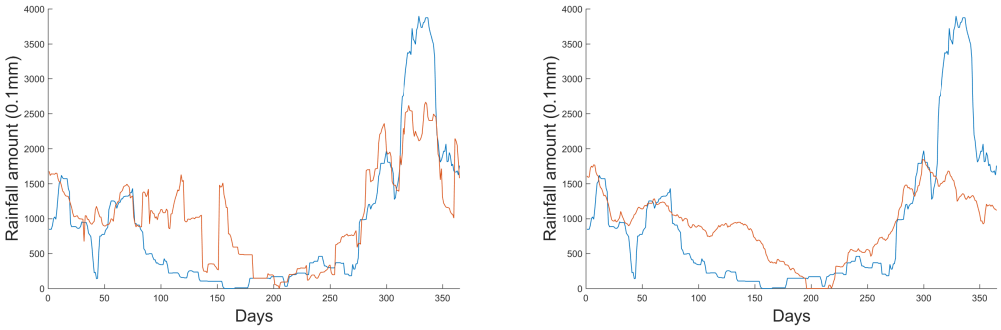
(f) GP - Detroit

Fig. 5.8 Rainfall time series for Atlanta, Chicago and Detroit on the testing set Jan-01-2015 until Dec-31-2015 for DGP (left) and GP (right). The blue line is the actual accumulated level of rainfall and the red line is the rainfall level predicted by the best individual from training over 50 runs.



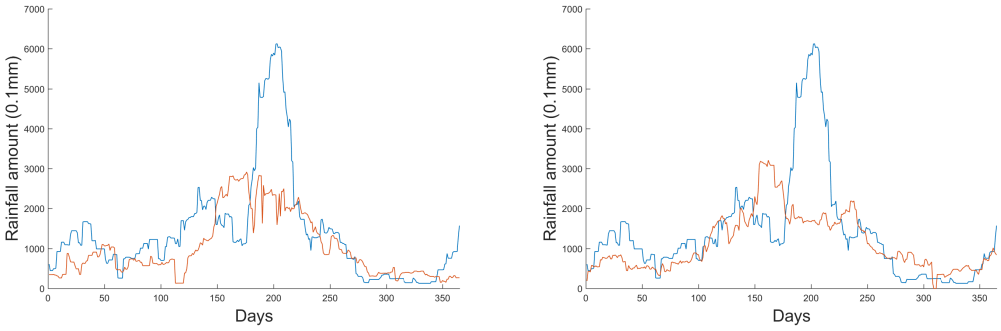
(a) DGP - Kansas

(b) GP - Kansas



(c) DGP - Portland

(d) GP - Portland



(e) DGP - Tampa

(f) GP - Tampa

Fig. 5.9 Rainfall time series for Kansas, Portland and Tampa on the testing set Jan-01-2015 until Dec-31-2015 for DGP (left) and GP (right). The blue line is the actual accumulated level of rainfall and the red line is the rainfall level predicted by the best individual from training over 50 runs.

5.7 Effect of Decomposing the Problem

Within this section, we consider the benefit of decomposing the problems (i.e., evolving a separate equation for each rainfall class) has on the ability to predict more similarly to the underlying data. We previously noted that the GP tends to produce equations with flat predictions and is unable to meet the oscillations of the time series. To consider

this we analyse the effect that the DGP has on the coverage of the predictions and whether the DGP is able to overcome any climatic issues.

5.7.1 Effect on Increasing the Coverage

One of the observations noted from the analysis of the last chapter is that the GP was unable to predict the full spread of rainfall amounts. One of the motivations of the DGP is to improve the behaviour by the use of decision criteria, to choose an equation that specialises in the wetter and drier periods.

We show in Figures 5.6, 5.7, 5.8 and 5.9 examples of a DGP and a GP individual for twelve cities on the testing set. Those individuals that produce the lowest RMSE error on training over all 50 runs are chosen. What we observe from this, is that DGP does appear to predict the highs and lows more consistently. Moreover, the predictions are similar to the underlying data where we observe more volatile periods. We generally witness the problem with coverage, where visually it appears that the DGP cover more points, and the GP tends to provide flatter predictions in some examples. We provide the full coverage results in Table 5.16 to compare the DGP and the GP across all data sets over 50 runs.

From Table 5.16, we can observe in every city that the DGP is able to cover more rainfall values than GP, when the coverage for the GP is less than 100%. *There were no occurrences where the DGP covered less rainfall values than the GP*. Therefore, we can take away the advantage that the DGP has over its predecessor and its ability to increase the coverage. It is shown from the figures, the DGP creates equations that predict rainfall amounts more similar to that of the underlying data of accumulated rainfall.

5.7.2 Effect on the Climate

Lastly, we consider the performance of the DGP against the same climatic features outlined in the previous chapter. We investigate the same research issues within the previous chapter, given below:

- The predictive error is similar between Europe and the USA.
- Drier or wetter climates are associated with a lower predictive error.
- More volatile cities are associated with higher predictive error.
- High rainfall intensities are associated with higher predictive error.

Table 5.16 The coverage (in %) in terms of the number of observed rainfall values covered of each algorithm on all cities for GP and DGP.

Data	GP	DGP	Data	GP	DGP
Amsterdam	37%	65%	Boston	31%	44%
Arkona	85%	100%	Capehatteras	23%	61%
Basel	66%	93%	Cheyenne	46%	72%
Bilbao	35%	54%	Chicago	58%	77%
Bourges	35%	61%	Cleveland	16%	57%
Caceres	100%	100%	Dallas	12%	49%
Delft	41%	72%	Des Moines	59%	100%
Gorlitz	84%	99%	Detroit	28%	53%
Hamburg	47%	62%	Indianapolis	13%	71%
Ljubljana	57%	59%	Jacksonville	46%	89%
Luxembourg	34%	62%	Kansas	38%	83%
Marseille	80%	85%	Las Vegas	81%	100%
Oberstdorf	76%	86%	Los Angeles	92%	100%
Paris	71%	83%	Louisville	21%	37%
Perpignan	57%	100%	Nashville	59%	59%
Potsdam	75%	81%	New York	50%	58%
Regensburg	78%	100%	Phoenix	81%	100%
Santiago	36%	66%	Portland	44%	71%
Strijen	22%	61%	Raleigh	37%	49%
Texel	45%	72%	St Louis	26%	64%
Atlanta	16%	41%	Tampa	44%	78%

By evaluating the points above, we hope that the DGP is able to overcome the problem of higher predictive error associated with the above issues, unlike the GP from the previous chapter.

The first consideration is the effect across the two distinct geographic regions of Europe and the USA. We apply the Mann-Whitney test to determine if the predictive error is consistent across both continents. For our proposed DGP, we obtain a p-value of 0.7721. Therefore, the predictive error is similar between Europe and the USA, since $p > 0.05$.

In order to investigate the next three issues, we consider the correlation between the descriptive statistical points and the predictive error of our DGP. We present the findings in Table 5.17, by using the Pearson's product-moment linear correlation coefficient (π) to measure the strength of the relationship. Additionally, we include

Table 5.17 The linear correlation coefficient (r) and p-value for European and cities from the USA, in order to determine whether there is sufficient evidence that a relationship exists between a data set property and an algorithm's predictive error. The p-value is shown in brackets below the correlation coefficient. Significant relationships ($p < 0.05$) are shown in bold.

Data set property	DGP		GP	
	USA	Europe	USA	Europe
% of dry days	0.02 (0.9446)	0.27 (0.2571)	-0.30 (0.1683)	0.08 (0.7454)
Average dry spell	0.21 (0.3539)	0.08 (0.7256)	-0.42 (0.0539)	0.23 (0.3315)
Average wet spell	0.24 (0.3410)	-0.23 (0.3577)	-0.08 (0.7362)	0.13 (0.5811)
Annual rainfall	0.06 (0.7940)	0.27 (0.2528)	-0.37 (0.0926)	0.10 (0.6805)
Volatility of annual rainfall	0.28 (0.2140)	0.07 (0.7547)	-0.37 (0.0894)	0.26 (0.2745)
Highest intensity	-0.30 (0.1719)	-0.07 (0.7557)	**0.49** **(0.0199)**	0.42 (0.0641)
Interquartile range of intensity	0.35 (0.1105)	-0.35 (0.1351)	**-0.44** **(0.0396)**	**-0.58** **(0.0071)**

the p-value computed by the Student's t-distribution, in order to determine whether there is a statistically significant relationship between the predictive error and the descriptive statistics. The null hypothesis for the test is that $\pi = 0$. We only include our original GP as a comparison, because we are only considering whether the DGP leads to an improvement over the GP. The values highlighted in bold indicate a statistically significant relationship at the 5% significance level.

Looking at Table 5.17, considering the dryness and volatility of cities, these city properties are not significantly correlated with the DGP's and the GP's predictive error. The p-value is higher than our significance level in both cases of Europe and the USA, for both algorithms.

Finally, considering the high rainfall intensities we initially see that this factor is significantly correlated with the GP's predictive error in the USA, observed by a p-value of 0.038, but not for Europe. The DGP's predictive error shows no significant correlation with rainfall intensity within the USA and Europe, in both cases with a p-value greater than 0.05.

A general comment for the table, the relationships provided for the DGP shows that our proposed the DGP algorithm is more robust against different climates, across different geographical regions.

5.8 Conclusion

Within this chapter, we outlined a new algorithm for the problem of rainfall called Decomposed GP (DGP). The DGP was a way to overcome the potential issues outlined in our previous chapter, where we observed that the GP was unable to consistently provide equations suitable for the underlying problem of rainfall. Therefore, we aimed to address this issue by creating DGP to influence this behaviour in our final equations.

DGP is a novel algorithm based on decomposing the problem of rainfall prediction. The idea revolved around breaking the problem of rainfall prediction into subproblems (partitions) for GP to solve, before combining the subproblems into the solution for the original problem. DGP was a hybrid approach, which incorporated a GP and a Genetic Algorithm (GA), in order to predict the accumulated rainfall amounts. The motivation for this chapter was to make the process of rainfall prediction a simpler problem space. Therefore, DGP created a separate regression equation (tree) to predict each partition more accurately. Our GP proposed in Chapter 4 was extended with the proposed GA component for classification (before performing regression). A classification technique was used, because the DGP needed to choose which regression equation was evaluated. We additionally proposed the use of other classification algorithms as the decision process to substitute for the GA.

The results showed that our DGP predicted more accurately when compared to GP and managed to perform similarly to Radial Basis Function and better than Support Vector Regression. We also considered the impact of changing the underlying GA to a different classification algorithm. At first, we compared the classification accuracy and found that our proposed GA was one of the best classification techniques, and the RBF the best classification algorithm out of our tested algorithms. Our hypothesis was that the better the classification algorithm the lower the predictive error of the DGP. To test this, we ran each classification algorithm as the decision criteria within the GP and discovered that this hypothesis was mostly true. One exception was from the GA, which was an anomaly in terms of its effect on the predictive accuracy, since the DGP with the GA outperformed the DGP with classification techniques with a higher classification accuracy.

Finally, we observed the effect that our proposed DGP with the GA as the decision criterion had on its ability to predict rainfall amounts. We found that the DGP consistently produced equations that were representative of the underlying variable of rainfall and the DGP's predictive error was not significantly correlation with variations in most climatic aspects.

In the next chapter, we plan on using our DGP for pricing derivatives at the Chicago Mercantile Exchange.

Chapter 6

Pricing Rainfall Derivatives

6.1 Introduction

In the previous chapter, we outline a new methodology, called Decomposition GP (DGP) for predicting the accumulated level of rainfall using a decomposition approach. We observe that this methodology predicted better results than the previous GP implementation and the current approach of MCRP. We continue to build on this success for better predictions in rainfall futures prices at the Chicago Mercantile Exchange (CME). Throughout this thesis, we strive to decrease the error from rainfall prediction, because of the negative correlation that exists between rainfall future pricing accuracy and predictive error. We observe the lower the predictive error, the better accuracy there is for pricing derivatives (Alexandridis and Zapranis, 2013).

To price derivatives there are two techniques that are formulated for rainfall derivatives, which are discussed earlier in Chapter 2: indifference pricing (Carmona and Diko, 2005) and the arbitrage free approach (Cabrera et al., 2013). Since contracts began trading on the CME, the latter became the standard pricing technique. The technique works by probabilistically transforming the predictions from the risky world to the risk-neutral world. Thus, the method requires a series of possible future rainfall values for each contract in order to calculate the probability of an event occurring. Synonymous to general derivative pricing, a stochastic process (e.g. a random walk) is required to generate the possible rainfall values. MCRP is a valid approach, given it is a stochastic process. However, we already identify that it is predictively very weak. As shown in previous chapters, a large number of potential rainfall pathways generated by MCRP are not reflective of future rainfall events. This has implications on pricing accuracy.

In order to price using DGP, which requires model extension to account for the uncertainty that surrounds the future forecasting of rainfall. Currently, DGP produces

a deterministic solution for the accumulated level of rainfall within the risky world, without taking into account any uncertainty that exists. Hence, only a single rainfall value representing the future can be generated, which is a probability that cannot be assigned. Since no probability is obtained, we are unable to calculate the price of a derivative under risk-neutral conditions by using the arbitrage free approach or indifference pricing methods.

In this chapter, we aim to extend our DGP to account for future uncertainty by producing a probabilistic landscape of our future predictions of accumulated rainfall. Therefore, we are able to price by using the most adopted method for pricing (the arbitrage free approach). To achieve the desired behaviour, we begin by extrapolating a series of future predictions. It samples multiple individuals from the population in the final generation across multiple runs. To avoid the unnecessary computational overhead of the GP and to select individuals that perform well, we restrict the number of runs and samples taken. Then, using the Monte Carlo Markov Chain (MCMC) method of Gibbs sampler, we can estimate the population based on the samples provided.

Additionally, we consider producing contract-specific equations, which should help to reduce the predictive error. This is more favourable, because the DGP becomes more focused on a certain period of activity. This helps MCMC to provide a better representation of the more likely outcomes. One downside is the methodology is no longer one size fits all, but with the predictive gains and the effect it has on pricing, it is a positive way forward.

Finally, we propose a new algorithm for rainfall prediction by generating and evolving a stochastic equation of rainfall. This concept incorporates both of the prior approaches within this chapter. Firstly, we no longer require MCMC to estimate the population, as we are generating our own stochastic equations. This can be used for generating many future rainfall indices based on a single model with minimal overhead. Secondly, we do not need to produce separate models for each contract, because we incorporate that behaviour within our process.

All three methods outlined are compared on their predictive accuracy for rainfall and their respective pricing accuracy. To evaluate our pricing performance, we use the initial prices quoted by the CME on three cities for eight monthly contracts[1]. These prices reflect the initial estimates up to eleven months ahead of the contract period and are not those from actual trading or those based on the final price for each contract. Therefore, base the assumption that the contract prices available to us will eventually converge to the real price (Hull, 2006). We additionally compare the actual rainfall amount for each contract period. We perform this comparison, as the accumulated

[1]These were publicly available from Cabrera et al. (2013) and are accessible via Bloomberg.

rainfall amounts within each contract form the price for each contract (with some adjustment for risk) (Prigent, 1999).

We begin with Section 6.2, with a short overview of the pricing procedure, which is discussed in more detail in Chapter 2. In Section 6.3, we outline the first methodology of transforming our deterministic solutions into a range of possible outcomes by using MCMC. In Section 6.4, we outline the need for contract specific equations as a substitute to the first approach, in order to further improve the predictive accuracy of rainfall and hence prices. In Section 6.5, we outline our new algorithm based on producing a stochastic model, which should offer an advantage over the previous two approaches. In Section 6.6, we outline the experimental set-up and tuning for all methodologies. In Section 6.7, we show the experimental results for all three methods on the rainfall prediction problem and also how they relate to the pricing of rainfall derivatives. Finally, we conclude in Section 6.8.

6.2 Pricing Within Rainfall Derivatives

One of the key characteristics of the weather derivative market is the nature of the incomplete market, whereby the underlying weather indices are non-tradable assets and cannot be replicated by other risk factors. In other words, it is not possible to construct a riskless hedge portfolio containing the weather derivative. The standard approach is to price a futures contract $F(t;\tau_1,\tau_2)$ at time t with accumulation period $[\tau_1,\tau_2]$. It calculates the risk-neutral expectation Q of the rainfall index $I(\tau_1,\tau_2)$ with accumulation period $[\tau_1,\tau_2]$ based on the information set F_t available at time t. Therefore, the underlying variable is required to calculate the index over an accumulation period. We can express the price of a futures contract as the following:

$$F(t;\tau_1,\tau_2) = \exp^{Q}[I(\tau_1,\tau_2)|F_t] = E^{Q}\left[\sum_{\tau=\tau_2}^{\tau_2} R_\tau|F_t\right]. \tag{6.1}$$

Our rainfall estimates $I(\tau_1,\tau 2)$ is considered as the expected price under the canonical measure P, but is within the 'risky' world. Therefore, we require $Q \sim P$ such that all tradable assets in the market are martingales after discounting taking into account investors' exposure to risk. To establish the risk preferences of investors requires the market price of risk (MPR). It is the additional return or the risk premium expected by investors for being exposed to undertaking the futures contract. Within complete markets, where the modelled quantity is tradable, investors are able to hedge away the risk in any position by dynamically buying and selling the underlying asset. This allows the calculation of the equivalent martingale measure of Q, hence the MPR

is not required. Under the Black-Scholes and other similar pricing models, the unique equivalent martingale measure is obtained by changing the drift in the Brownian motion. Within the incomplete markets, the MPR plays an important role in estimating the equivalent martingale measure. Because an investor cannot hedge away associated risks, a premium (MPR) is expected to compensate for taking an unhedgeable risk.

Therefore, we must specify the risk-neutral probability of Q, but given that weather derivatives in general are incomplete, there will exist many different martingales (Q). Hence, it is not possible to find a unique risk-neutral measure Q (Benth and Benth, 2012; Jenson and Nielsen, 1996), such that Q is equivalent to the physical measure P. Benth and Benth (2012) propose the use of the Esscher transform, a generalisation of the Girsanov transform for Brownian processes, parameterised by the MPR (θ). The Esscher transform has long been used across financial applications (Bühlmann et al., 1998; Gerber and Shiu, 1995; Kremer, 1982) and is valid for Lévy processes.

To use the Esscher transform, we require estimating the type of distribution for our predictions. We can then apply a constant MPR to transform our distribution to find the expected price under the risk-neutral measure Q_θ, where θ is calibrated to the market data. The transformation of probability density $f(x)$ of a random variable X to a new probability density $f(x;\theta)$ with parameter θ is the Esscher transform, given by:

$$f(x;\theta) = \frac{\exp(\theta x) f(x)}{\int_{-\infty}^{\infty} \exp(\theta x) f(x) dx}. \tag{6.2}$$

Here we see the Radon-Nikodym derivative with θ being the level of risk exposed to investors from the jumps of the driving process of rainfall. The Esscher transform reflects the corresponding risk by exponentially tilting the jump measure shown by Equation 6.2. Prior work by Cabrera et al. (2013) noted that approximating the index distribution of rainfall by using the Normal-Inverse Gaussian (NIG) (Barndorff-Nielsen, 1997) distribution is the most appropriate approach. The NIG distribution is a class of Lévy processes that can capture the skewness of the observed distribution and the heavy tail nature of rainfall, which we use within this chapter. Thus, the use of the Esscher transform is suitable.

The NIG distribution has four parameters and belongs to the generalised hyperbolic distributions. It is used for several applications of risk-neutral modelling across a variety of financial problems, with a PDF in the closed form of:

$$f(x|\alpha,\beta,\mu,\delta) = \frac{\alpha\delta \exp(\delta\sqrt{\alpha^2-\beta^2} + \beta(x-\mu))}{\pi\sqrt{\delta^2+(x-\mu)^2}} K_1\left(\alpha\sqrt{\delta^2+(x-\mu)^2}\right), \tag{6.3}$$

where K_1 denotes the modified Bessel function of the second kind. Parameters α, μ, β and δ control the steepness, the location, the skewness and the scaling of the

distribution respectively. The NIG distribution is infinitely divisible and creates a Lévy process $L_t, t \geq 0$, making it valid for the Esscher transform instead of the Girsanov transform (Brownian motion). The parametric shape of the distribution is also kept under the Esscher transform, with the MPR (θ) becoming NIG($\alpha, \beta + \theta, \mu, \delta$).

6.3 Adapting the DGP Algorithm to the Esscher Transform

From the introduction of the pricing framework, our first method is to allow the GP to provide rainfall futures prices at the CME. It is based on taking our deterministic equation and extrapolating to fit and price via the NIG distribution. Before pricing, we need to perform an intermediate step in order for the GP to calculate risk-neutral prices using the Esscher transform. One of the key aspects of the Esscher transform is the probabilistic shift under $P \sim Q$ to find a unique equivalent martingale close to the predicted level of rainfall. This requires constructing a probability density function (PDF) out of the predictions generated. The current approaches in rainfall derivatives are stochastic processes that simulate unique rainfall pathways on each iteration. Despite GP being a stochastic algorithm, the output is a deterministic model and cannot be used to estimate the expected index of rainfall. However, the GP generates many different equations to describe the rainfall process over the evolution process.

In order to recreate the probabilistic density of a stochastic process (e.g. MCRP), we choose a subset of rainfall equations generated from the GP over many runs to get multiple predictions for every day. By building a large enough sample size by using many subsets of different rainfall equations, we can form a PDF of the expected level of rainfall for each day. The PDF generated can be manipulated to price under the risk-neutral density using the Esscher transform. Based on the nature of rainfall, we expect to generate a non-gaussian distribution similar to the underlying data, which is assumed to follow either a gamma distribution or a mixed-exponential distribution (Wilks, 1998).

6.3.1 Strategy for Prediction Selection

To generate a PDF it requires to have several different observations for the same time point. We also require a sufficient number in order to determine the form of the distribution. By using MCRP, one would typically run the chain for 10,000 times in order to generate sufficient samples. This is unfeasible given the computational cost

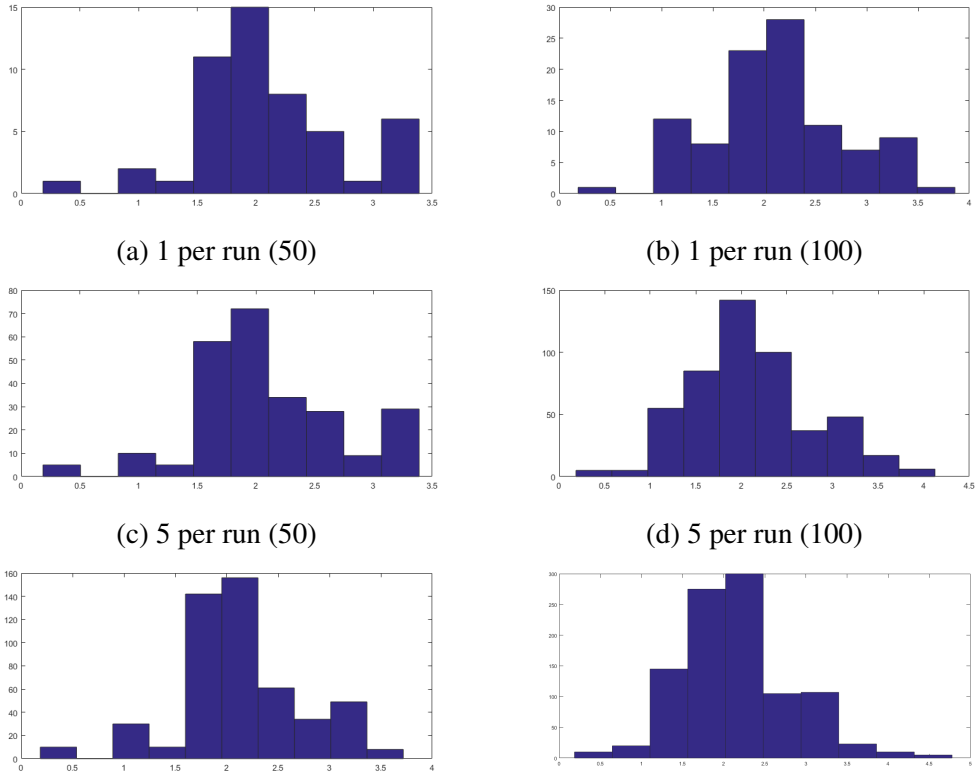
of running the GP 10,000 times, if we were to take the final prediction from each run. To reduce the overhead, we propose taking a sample of the best solutions from the final generation of each run. Not only this reduces the computational cost, but is a simple method to extract the required information. One concern is that taking too many samples from the final generation may reduce the fit of a distribution. If poor predictions are selected, it produces heavily skewed results. Thus, we aim to find the best possible balance.

We present sample results for a contract of March by using various strategies in Figure 6.1. Figures 6.1a, 6.1c and 6.1e show the PDF of choosing between 1, 5 and 10 best individuals per GP run over 50 runs and Figures 6.1b, 6.1d and 6.1f show the outcome over 100 runs instead. Therefore, Figure 6.1a contains a total of 50 samples, whereas, Figure 6.1f contains a total of 1000 samples. We choose the sample sizes to avoid longer runs of the GP and to reduce the risk of selecting too many extreme values that may exist from poor fitting solutions. We noticed that samples of 25 or larger pose a risk of selecting extreme values. We can see, from the figures the nonguassianity of the predictions, which we would expect and not to exceed 100 runs of GP, due to computational overhead. In some cases, we witness that the GP seems to find a modal value for the prediction with a fairly narrow distribution. From a pricing perspective, this is a positive sign as the GP is able to determine what it believes to be the expected outcome.

Based on the PDF's generated, we notice that in several cases no clear distribution can be easily identified shown in Figure 6.2. This indicates the sample size is not large enough and we would anticipate that generating more samples would lead to a clear distribution. However, we attempt to reduce the computational overhead and the necessity of running 10,000 separate GP runs. Hence, we employ Markov Chain Monte Carlo (MCMC) to estimate the true value of parameters for the distribution.

6.3.2 Markov Chain Monte Carlo (MCMC) With Gibbs Sampling

The first key ingredient in Bayesian inference is the observation whose values are initially uncertain and described through a PDF. Another critical aspect is the previous belief about values of the parameter of interest before observing the data. Bayesian theory is based on Bayes' Theorem, which allows new evidence to be used for updating beliefs through probabilities. Consider a random sample $x = (x_1, \ldots, x_n)$ and the parameter of interest $\theta \in \Theta$ with Θ being the parameter space. The likelihood function of θ is defined as: $f(x_1, \ldots, x_n|\theta)$, the prior distribution $p(\theta)$ is the PDF before the observation of the value x. The inference is then based on the probability distribution of θ after observing the value of x, upon which information becomes available. We



(a) 1 per run (50)

(b) 1 per run (100)

(c) 5 per run (50)

(d) 5 per run (100)

(e) 10 per run (50)

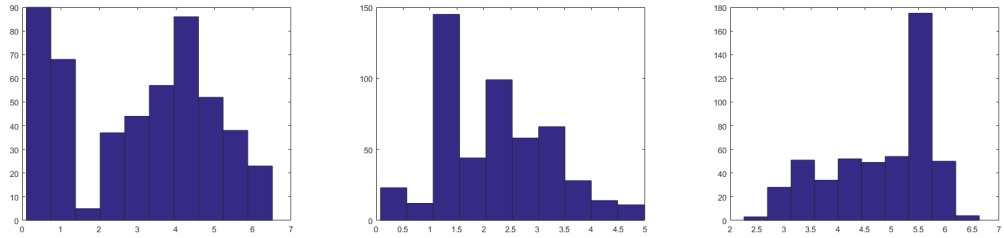
(f) 10 per run (100)

Fig. 6.1 The probability densities generated from GP for different strategies are shown for Detroit for the contract period of March (01/03/2011 - 31/03/2011). Values in brackets represents the number of GP runs, with the number per run showing how many best individuals are chosen to form the PDF at each run.

can then obtain the posterior distribution:

$$p(\theta|x_1,\ldots,x_n) = \frac{\prod_{i=1}^{n} f(x_i|\theta)p(\theta)}{\int \prod_{i=1}^{n} f(x_i|\theta)p(\theta)d\theta} \propto \prod_{i=1}^{n} f(x_i|\theta)p(\theta). \tag{6.4}$$

In order to estimate the posterior distribution stated in Equation 6.4, we can use MCMC simulation. We draw new samples of parameter $\theta = (\theta_1,\ldots,\theta_p)$ directly from the joint posterior $p(\theta|x_1,\ldots,x_n)$. We estimate the joint posterior by using a Gibbs sampler, which is one type of MCMC algorithm. Gibbs sampling begins with an initialised vector of $\theta^0 = (\theta_1^0,\ldots,\theta_p^0)$. At each iteration t, each component θ_j^t is sampled from the conditional posterior distribution given all the other components of θ to generate a new vector of $\theta^t = (\theta_1^t,\ldots,\theta_p^t)$. The sampling step of θ follows as:



(a) 10 per run (100) (b) 10 per run (100) (c) 10 per run (100)

Fig. 6.2 Situations where a clear density cannot be identified for contracts traded for Detroit in June (a), Jacksonville in June(b) and New York in April (c). Values in brackets represents the number of GP runs, with the number per run showing how many best individuals are chosen to form the PDF at each run.

$$\begin{aligned}
\theta_1^t &\sim p(\theta_1|\theta_2^{t-1},\theta_3^{t-1},\ldots,\theta_p^{t-1},x_1,\ldots,x_n)\\
\theta_2^t &\sim p(\theta_2|\theta_1^{t-1},\theta_3^{t-1},\ldots,\theta_p^{t-1},x_1,\ldots,x_n)\\
&\vdots
\end{aligned}$$

The sampling steps end once the last iteration is reached (with sufficient iterations in the so called "burn in period" to achieve convergence). The predictive rainfall r_t of days of interest t follows an independent reparameterised gamma distribution in the form of the mean and the standard deviation of the initial rainfall predictions:

$$f(r_t|\alpha,\beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} r_t^{\alpha-1} e^{-\beta r_t}, \quad \alpha = \frac{\mu^2}{\sigma^2}, \quad \beta = \frac{\mu}{\sigma^2}. \tag{6.5}$$

Hence, the parameters of interest for the likelihood distribution in Equation 6.5 are the mean and the standard deviation parameters. The prior probability distributions are the same for both parameters of interest. They both have vague priors, that is the Uniform priors $U(0,1)$. In order to estimate the posterior of the parameters of interest, we use JAGS (Plummer, 2003), which is an iterative MCMC simulation method, using the Gibbs sampler described previously. We run a total of 50,000 iterations including 10,000 iterations being the burn-in period.

Figure 6.3 shows the density plots of the Markov chains for both the shape and the rate parameters of the gamma distribution obtained by using JAGS. Note that for each posterior density, all simulated Markov chains converge to stationary, shown by the clear peak. Hence, the number of iterations and burn-in period used are sufficient to achieve convergence of the Markov chains.

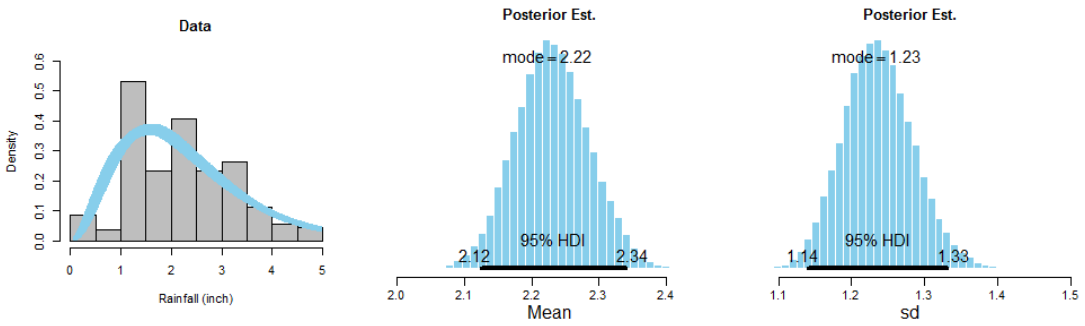


Fig. 6.3 The estimated kernel density of the predicted rainfall amounts and the density plots of our posterior estimates for the shape and rate parameter.

Figure 6.3 also shows the estimated kernel density of the predicted rainfall amounts by using the posterior means of the shape and the rate parameters in the Gamma distribution. We can see that the estimated density is representable of the target density of the rainfall prediction. This shows that using MCMC has assisted GP in creating a PDF that can be used for pricing, whilst minimising the overhead. To obtain the same 10,000 samples by just running GP, would have been up to ten times slower and without any guarantee of convergence.

6.4 Producing Contract-Specific Models

Our next approach builds on using a Gibbs sampler to estimate a rainfall density where we were able to minimise the computational overhead by running the DGP fewer times. We propose a second methodology to the rainfall process by maintaining the same methodology, but by considering contract-specific equations. We hope to increase the ability of predicting the highs and lows of the rainfall time series, which we refer to as the dynamics of rainfall.

One of the issues present with the GP is the lack of dynamics (covering the full range of possible values) in our final equations produced, despite encouraging this behaviour from the use of decomposition discussed in the previous chapter. One of the underlying trends that we notice and analysed across all experimentations so far is that the GP is more likely to produce an equation with flatter predictions to avoid being heavily penalised by extreme values. We previously hypothesised that this is due to irregular behaviour in the rainfall time series, whereby there exists few reoccurring patterns across years.

In an attempt to address this issue, one possibility that gives flexibility to achieve this, is by considering shorter training and testing time frames. The DGP focuses on shorter periods and produces forecasting horizons of a shorter length. We should

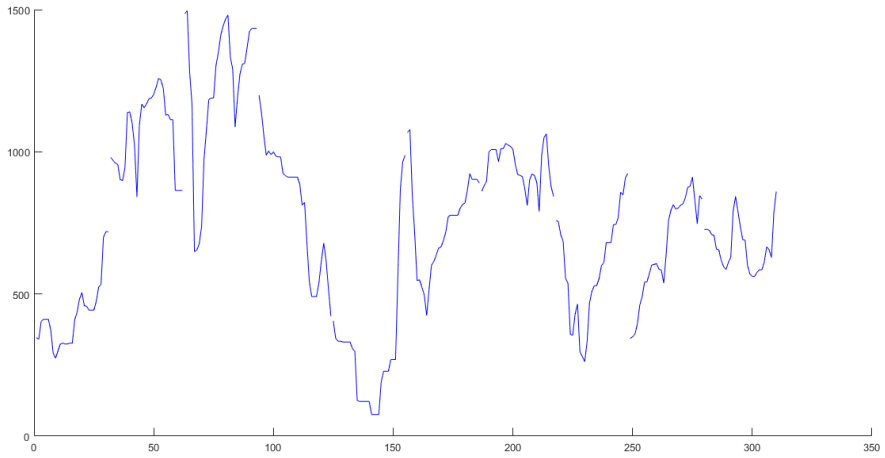


Fig. 6.4 One Contract length.

expect to see more dynamic behaviour. Periods of fluctuation in the time series are usually fitted by a smooth line through the points, rather than the DGP attempting to capture the fluctuations. Thus, by considering periods of training of shorter lengths, we would expect contract-specific equations to be less affected.

To achieve this behaviour we set out four alternatives in how we create contract-specific equations. We use the data set of Portland to show how each alternative is formed.

6.4.1 One Contract Length

In the shortest of the training and the testing set-ups, we consider the number of data points prior to the contract date, which makes the last day in our observation the target contract period (e.g. daily rainfall between March-01 and March-31). For example, when we consider the contract of March, we are interested in the period 30-Jan until 01-March for both the training and the testing. We show the filtered data set for March across the whole length of our rainfall training set in Figure 6.4. Note that for all figures showing the training period, we do not connect up the partitions. In other words, when one partition ends and another begins across the length (in years) of our training set.

This shows the nonexistence of a reoccurring pattern in the rainfall time series. As we can see from Figure 6.4, the rainfall level either increases, decreases or stays approximately the same, along with some observable extremes of wet or dry periods. By isolating each period, we aim to extract more meaningful patterns that represent future rainfall patterns.

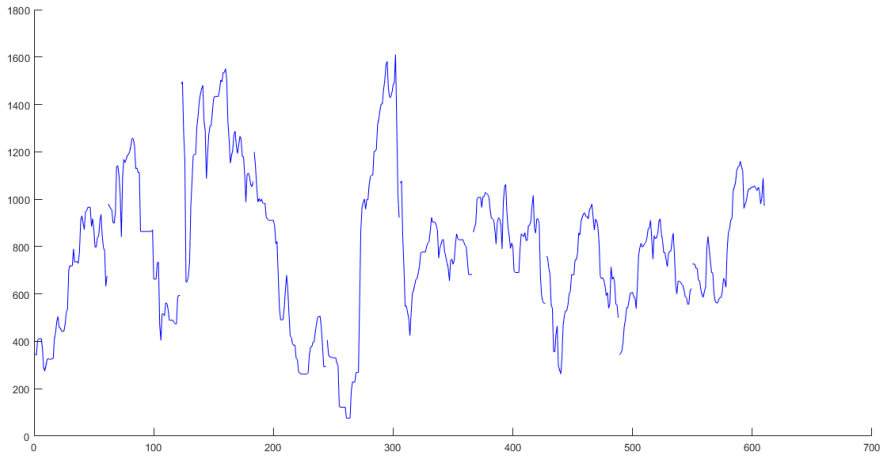


Fig. 6.5 Two Contract lengths.

6.4.2 Two Contract Lengths

Secondly, we consider the effects of two contract length periods. Here we consider only using the data points corresponding to the contract length prior to the contract and the month concerned. Since we are using the sliding window data transformation, this becomes the minimum period length that can be considered to cover all possible values contained in the contract period. We are interested in any sliding window values that contain one or more days from the contract period. For example, consider the contract of March, which is the accumulated level of rainfall from 01-March until 31-March. The sliding window values of 30-Jan until 31-March for both the training and the testing contain a value of rainfall during the period of 01-March until 31-March. We show the proposed data in Figure 6.5.

We show the length of two contracts, such that the point of interest is the mid point for each part. It represents the accumulated rainfall amount for the contract of March.

6.4.3 Three and Four Contract Lengths

Thirdly, we consider three and four contract lengths, whereby we consider an extended length prior to and after the two contract lengths' set-up. For the three contract length version, we consider a contract length of rainfall prior to the two contract lengths. We try to identify any information that may lead up to the contract window. For March, the training and the testing period would be from 30-Dec until 31-March.

The four contract length version takes into account the same information as three contract lengths, but considers the contract length following the two contract lengths. For March, the training and the testing period would be from 30-Dec until 01-May.

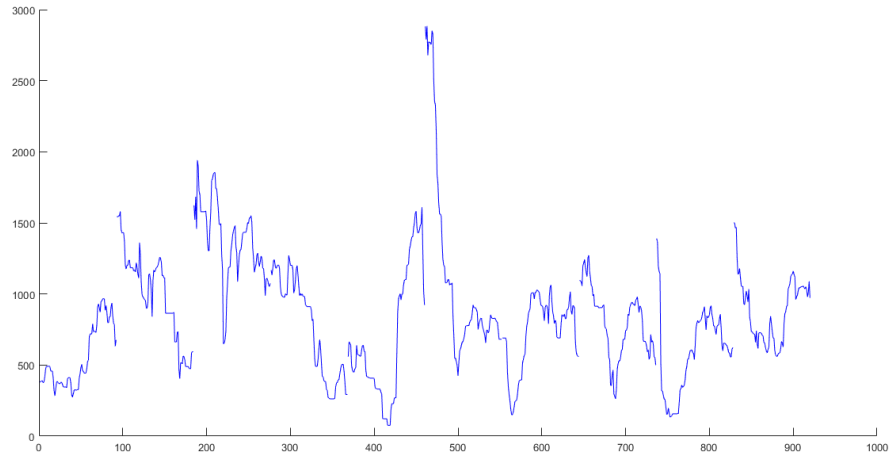


Fig. 6.6 Three Contract lengths.

Both of these versions examine if the extra information is useful, without causing the problems of flat predictions. The rainfall time series for three and four contract lengths can be found in Figures 6.6 and 6.7 respectively.

Considering Figure 6.6, we can observe some clear trends that occur in the run up to the contract period of interest. However, we see the existence of extremes and irregular rainfall amounts that can occur, which may hinder the predictive performance. Figure 6.7 paints a similar picture, but at this point there are several patterns occurring where the rainfall increases and shortly decreases and vice versa. This behaviour is the cause for flatter predictions from the DGP that we wish to avoid. Visually looking at the time series, we believe that this should be the maximum length. In some instances it appears to add value by strengthening the pattern that exists. But in others, it adds more instances of irregular patterns.

Across all data sets from visualising the time series, two and three contract lengths appears to give better indications of the observable trend and pattern. One contract length may not give sufficient information to build reliable predictions, whilst four contract lengths may cause the DGP to flatten out the predictions. We show the decision on the optimal length in Section 6.6.1.

We could consider longer periods, but we see some evidences that when the period is greater than or is equal to four contract lengths most probably causes flatter predictions. We observe that five contract lengths leads to too much information. In the vast majority of cases, leads to the inclusion of multiple periods of extremes and high fluctuations. We already identify this is an issue for the DGP. Additionally, one of the motivations is to learn from information on a shorter time scale. By considering

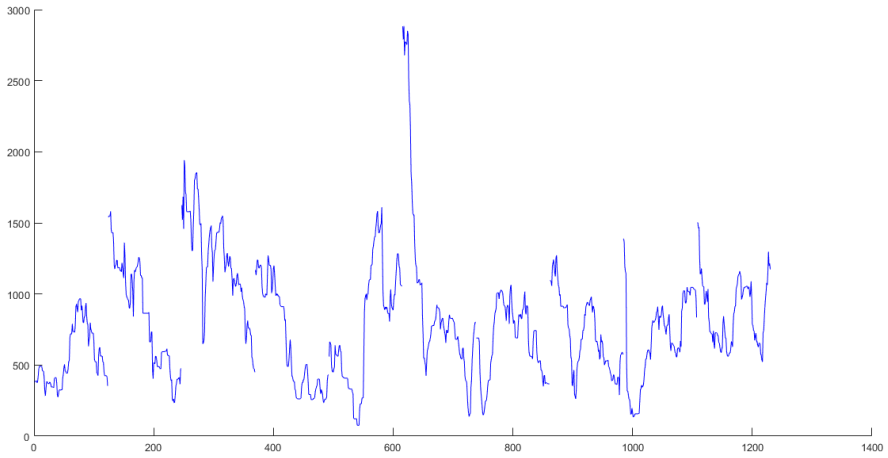


Fig. 6.7 Four Contract lengths.

periods longer than four contradicts our motivation of shorter training periods, as we would be training on just under half a year for five periods.

6.5 Generating Stochastic Equations Through GP

Our third method, which we refer to as Stochastic Model GP (SMGP). It is a new algorithm for the process of rainfall. The reason for developing a new algorithm for the process of rainfall is to have the stochastic equations produced by the GP, which avoids having to estimate using MCMC. Moreover, we want to avoid the overhead of evaluating 100 runs to extrapolate a density and to avoid running a model for each contract. Therefore, our aim in this section is to outline a process, which can evolve a single stochastic equation to be evaluated. The considerations to take into account within the algorithm are the dynamic nature of the time series and to avoid having a number of eight models [2] to run for each city.

6.5.1 General Model

Unlike previous experimentation, a general framework for each GP individual is given by Equation 6.6

$$y_t = \phi_t + \kappa_t + \varepsilon_t, \tag{6.6}$$

where t denotes each day, ϕ a seasonal component, κ an autoregressive component and a noise component ε. The motivation for having this extra ϕ component is that it

[2] one for each monthly derivatives contract traded in a year.

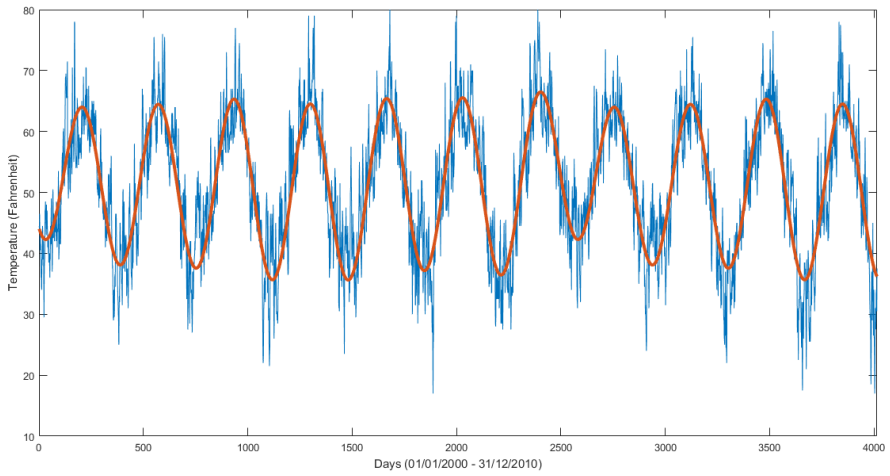


Fig. 6.8 The annual seasonality that exists within temperature as modelled via a truncated Fourier series.

allows us to extend each individual into the construction of a stochastic equation (will be described later). Despite no reoccurring seasonality in rainfall on an annual basis like temperature (Figure 6.8), there is some element of a reoccurring pattern on an irregular time scale when examining the time series (Figure 6.10).

6.5.2 GP Representation

We represent the GP to reflect the general model given above by Figure 6.9. Here we

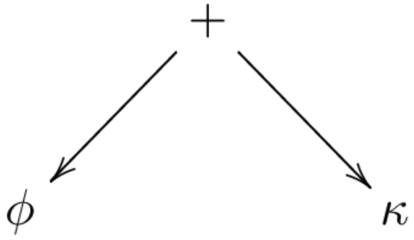


Fig. 6.9 The high-level representation of each individual in the population, consisting of a seasonal and a autoregressive component.

have a GP whose root node takes a "plus" symbol, which combines parameters κ and ϕ. We have a single population of individuals, which must consist of two branches (Equation 6.6). One for the parameter of ϕ and the other for the parameter of κ; they evolve to minimise the RMSE. The procedure of breeding is done on the pairing of ϕ with κ based on how well they solve the overall problem of rainfall predicting. We choose this procedure to reduce the randomness and to encourage more emphasis on

solving the combined problem. Usually, solving the subproblem for the seasonal and the autoregressive separately is more beneficial, but the GP needs to learn how much to offset the seasonal effect. This would be very difficult to generalise considering different seasonal patterns.

Within this framework, parameter κ is part of our individual that produces the same tree as the DGP, solving the autoregressive part. The only modification required is the wrapper that protects trees producing negative values, since the DGP part can now produce negative values. Instead of checking if the DGP (κ) tree's prediction is less than zero, we check if $\phi_t + \kappa_t$ is less than zero. If so, the output of κ_t is then modified to satisfy the equation $\phi_t + \kappa_t + d = 0$, where d is the value to offset the output of a GP individual at time t producing a nonnegative output.

6.5.3 Measuring Seasonality ϕ

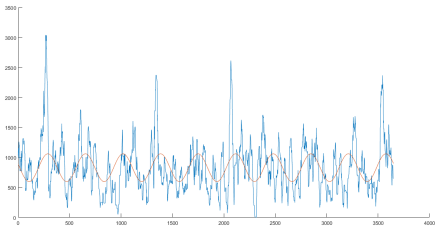
One aspect of the experimentations so far focuses on the absence of measuring seasonality, which is commonly done within time series analysis. We have discouraged the use of seasonality till now, because there is a lack of a reoccurring pattern [3]. However, within our algorithm, the seasonal component of ϕ is required to create a stochastic equation. It allows our algorithm to decide whether the predicted value of SMGP lies above or below the seasonal effect. Within this section, we outline the methods used to estimate a seasonal pattern. The most common procedure for analysing any seasonal effect is through fitting a truncated Fourier series, given by:

$$\phi(t) = \frac{a_0}{2} + \sum_{n=0}^{N} a_n \cos\left(\frac{2\pi nt}{T}\right) + b_n \sin\left(\frac{2\pi nt}{T}\right), \tag{6.7}$$

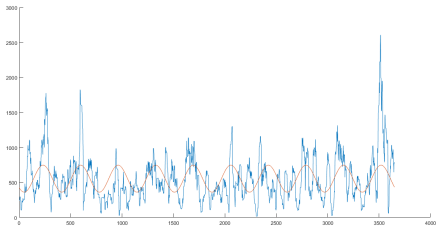
where a and b are constants fitted for the data, n is the order of the fourier series and T is the time period of the seasonal effect. Ideally, we expect a seasonal pattern for $T = 365$, which represents seasonality on an annual basis. For our problem, the effects of seasonality after the data transformation is not consistently the same over a year, which can be observed in Figures 6.10a, 6.10b, 6.10c and 6.10d. We observe no clear seasonal pattern, that is similar to that of Figure 6.8. The truncated fourier series overestimates and underestimates significant periods over the years.

This shows the problem with detecting and removing seasonality from our time series. We witness some level of seasonality, but not on a consistent scale depending on the data set. For example, we see the same spikes for all time series, but the lags between the spikes varies between 9 months to 15 months every year. Therefore, there exists some level of seasonality following an irregular pattern, which is difficult

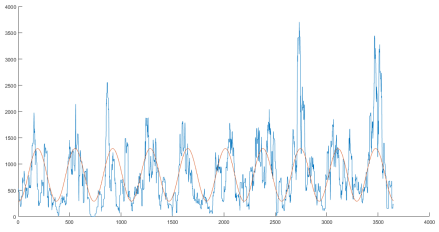
[3] We show this most recently in Section 6.4.



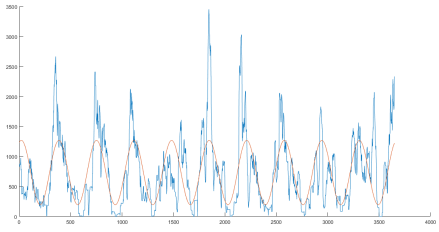
(a) Rainfall data for Delft fitted with an annual seasonal effect.



(b) Rainfall data for Gorlitz fitted with an annual seasonal effect.



(c) Rainfall data for Des Moines fitted with an annual seasonal effect.



(d) Rainfall data for Portland fitted with an annual seasonal effect.

Fig. 6.10 An attempt at fitting a truncated fourier series for an annual frequency.

to capture correctly. Fourier series unfortunately does not allow for this behaviour as the frequency of sine and cosine waves must be consistent. In order to have the desired behaviour to look for irregular patterns, we design a GP to perform the fitting of seasonality. This uses the fundamental behaviour of a Fourier series as a guidance for our model.

Structure of GP for Seasonality

For our GP to include a seasonality feature, we enforce a syntactic structure similar to that of a truncated Fourier series. However, the components within the sine and cosine allow for seasonal patterns of variable length. The main components of the proposed GP are as follows:

- All individuals consist of a root node (addition) with the first argument being an intercept and the second being any function.
- If a sine or cosine node is chosen, the first argument is the amplitude of that wave.
- The amplitude and intercept terms are strictly constants.
- Only within sine or cosine environments are terminals possible that affect the frequency of the curve.

By enforcing these syntactic structures, we are able to control the seasonality, which allows the GP plenty of flexibility to evolve solutions for varying seasonality. In order to enforce the structure of valid solutions and to maintain it throughout evolution, we use a Strongly-Typed GP, the same as the DGP.

Terminals

The terminals we use for ϕ are specifically designed for the seasonal part. The first terminal is the intercept, which is the equivalent to a_0 from the Fourier series. The second terminal is an amplitude, that is similar to the terms a and b from the Fourier series prior to the sine and cosine. It multiplies the output from the sine or cosine function. The third terminal is a dynamic terminal that reflects time index t of the function, which increments with each day till it reaches its seasonal length before repeating from 0. Finally, we have the frequency of the wave. The final can only exist within a sine or cosine environment.

Similar to the DGP, we have a set of constants specifically for the power function, which are in the same range -4 to 4, with 0.25 increments excluding 0.

Functions

The function set includes the same functions as the DGP, also includes sine, cosine and a root node, which must be addition. The list of terminals and functions is summarised in Table 6.1.

Table 6.1 Genetic Programming functions and terminal sets.

Set	Value
Functions	ADD, SUB, MUL, DIV, POW, SQRT, LOG SIN, COS, ROOT
Terminals	Amplitude, Frequency, Intercept, Dynamic, ERC, Constants in the range [-4,4]

Management of Trees

Additionally, to ensure that only the terminals *frequency* and *dynamic* could be chosen inside the sine and cosine environment, we modify the type system to include two types of add, subtract, multiply and divide. One set of types accepts only functions

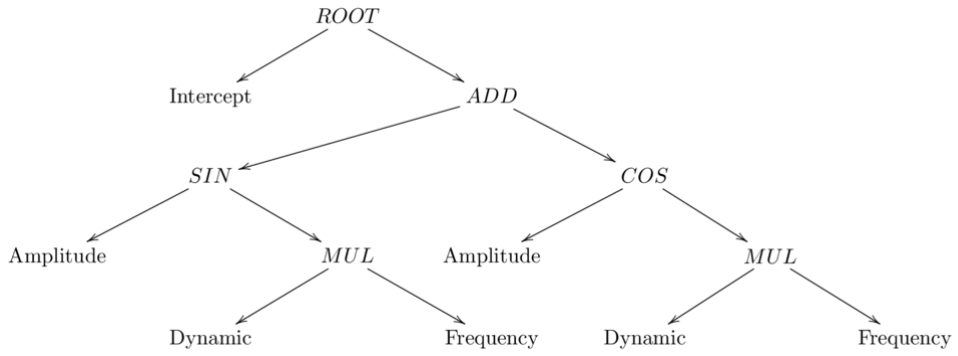


Fig. 6.11 An example tree showing the syntactic structure of a GP individual for the equation expressed as a truncated Fourier series: f(x) = $a_0 + a_1 sin(h_1 x) + b_1 cos(h_2 x)$, where h refers to $\frac{2\pi}{T}$.

as their arguments, whereas the other set can only be chosen directly within a sine or cosine environment. This allows for other functions and the terminals dynamic and frequency. Moreover, we have the same wrapper that is used in the DGP, which rounds to zero any negative value after evaluating the tree. An example tree showing the syntactic structure is given in Figure 6.11.

6.5.4 Creating a Stochastic Equation

Now that the two major GP components are introduced, namely ϕ and κ. We introduce the proposed technique for transforming the deterministic behaviour of our GP's individuals into a stochastic equation. We introduce to our general model from Equation 6.6 the use of weights. The motivation behind doing so is that certain parts of the year may be more dominated by κ or ϕ, due to the irregularities of annual rainfall. This allows the SMGP to estimate the most likely outcome at a particular point in time. We propose three variants as an extension to the previous model by using Equations 6.8, 6.9 and 6.10. Each variant specifies the weights differently on how they interact with our model.

$$y_t = \omega_t(\phi_t - \kappa_t) + \kappa_t + \varepsilon_t, \tag{6.8}$$

$$y_t = \omega_t^{\phi}\phi_t + \omega_t^{\kappa}\kappa_t + \varepsilon_t, \tag{6.9}$$

$$y_t = \omega_t(\phi_t + \kappa_t + \varepsilon_t). \tag{6.10}$$

In these equations, ω is the weight in the interval [0,1]. The motivation in three variations of ω is to promote different behaviours during the evolution when estimating the value of y_t. Under all approaches, there is a balance between ϕ and ω, which forms the basis for our stochastic process that each individual evolves. Under Equation 6.8,

there is a direct tradeoff between ϕ and ω, where we can contribute more from one or the other, or an equal weighting. Under Equation 6.9, there exists two separate weights, which allows for estimating the independent effect in their respective amounts. Finally, under Equation 6.10, the combined effect is controlled by one weight. We expect Equation 6.9 to perform the best, because it aims to capture the behaviour independently. However, this is more complicated to estimate correctly and we may find the simpler model of Equation 6.10 to be more appropriate.

Through the estimation of ω, we are looking for the optimal value of ω that minimises the RMSE of the GP. A typical technique would be using a local search technique to optimise the value of ω throughout the evolutionary process. However, by doing so does not allow us to formulate a stochastic process. Since the end result would be a constant, and a deterministic model would be achieved. To create the stochastic nature of an equation for each individual, the goal is to estimate the weights by using a probabilistic approach. This allows us to perform a random walk on our rainfall values and to estimate a density that reflects each day in our testing set. Going back to the pricing problem, by calculating the probability that a rainfall event occurs under P, we can translate this into the risk-neutral measure of Q.

Algorithm 3 shows the general algorithm, which is described within the following sections.

Algorithm 3 Overview of algorithm creating the stochastic behaviour

1: Initialise ω.
2: Set S (sample size).
3: **for** Generation $g = 1, ..., G$ **do**
4: Evaluate population.
5: Sort population on fitness.
6: $\omega* \leftarrow$estimateWeights(Predictions$_g \in S$) (Algorithm 4).
7: $\omega \leftarrow$updateWeights($\omega_g* \in S, \omega_{g-1} \in S$) (Algorithm 5).
8: **end for**
9: indi* $\leftarrow$ Best individual from training.
10: Error $\leftarrow$ predictWeights(ω, indi*) (Algorithm 6).

Sampling and Estimating the Weights

In order to estimate the value of ω to produce a stochastic equation, we specify that the weight that follows a beta distribution, given by the form:

$$f(x;\alpha,\beta) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\mathrm{B}(\alpha,\beta)}, \tag{6.11}$$

where $\alpha, \beta > 0$ are both shape functions and $B(\alpha, \beta)$ is the normalising constant. The benefit of the beta distribution is being a continuous probability distribution and is strictly bounded in the interval [0,1]. This property is suitable given we are bounding ω in the same interval, without the need to truncate other distributions within the same range. By sampling ω randomly via the beta distribution we are able to transform Equations 6.8, 6.9 and 6.10 into a stochastic process.

In order to estimate the weights for a given day, we first initialise the weights to be equal to 1, and start updating the weights after the first generation. To estimate the weights, we calculate the percentage difference for each day away from the expected value of rainfall for a set of individuals in the population. Then a beta distribution is fitted to those percentages based on the MLE of the parameters α and β. The mean of the estimated beta distribution is the weight for that day, that is calculated by $\frac{\alpha}{\alpha+\beta}$. We also keep track of whether the percentage is increasing or decreasing, where our prediction is less than or greater than the expected rainfall amount respectively for our random walk purposes. This is summarised in Algorithm 4.

For Equation 6.9, we estimate ω^{ϕ} first before estimating the effect of ω^{κ} on the modified values. For Equations 6.8 and 6.10, ω can be estimated based on the combined value of κ and ϕ.

Updating and Evaluating the Weights

As our individuals evolve, we need to modify and update the weights. Firstly, we estimate the new weights for the day, given the procedure listed above and then we decide whether we choose to accept or reject the new α and β. We do so via Monte Carlo simulation using inversion sampling to generate a uniform selection over our new distribution, by taking into account our previous and current values of α and β. We evaluate whether the new values of α and β lead to an improvement in fitness across the set of individuals, otherwise we keep the old α and β. We choose this method, because of the possible shapes that can be generated using the beta distribution, where the range can be extremely high. This affects the generalisation of weights throughout evolution. By updating the prior belief with the additional information resulting from the evolution of our DGP, the weights should converge. With respect to the three Equations 6.8, 6.9 and 6.10, all three are handled in the same manner.

As previously mentioned, we keep track of whether the weights increases or decreases the predicted rainfall value, by comparing the actual level of rainfall for that day with the amount predicted. In situations where we over predicted the rainfall, then we need the weight to reduce the predicted rainfall amount and vice versa. If we need

Algorithm 4 Estimating weights for producing stochastic equations

$S \leftarrow$ sample size for calculating weights.
Set $\omega_t^1 = 1 \forall t = 1, \ldots, T$.
Set $\omega_t^2 = 1 \forall t = 1, \ldots, T$.
for Generation $g = 1, ..., G$ **do**
    Evaluate population.
    Sort population on fitness.
    for all $i \in S$ **do**
        for all $t \in T$ **do**
            if $\text{Prediction}_i^t < \text{Actual}_t$ **then**
                $\text{increasingWeights} \leftarrow \frac{\text{Actual}_t - \text{Prediction}_i^t}{\text{Prediction}_i^t}$.
            else
                $\text{decreasingWeights} \leftarrow \frac{\text{Prediction}_i^t - \text{Actual}_t}{\text{Actual}_t}$.
            end if
        end for
    end for
    $\alpha^1, \beta^1 \leftarrow$ fitBetaDistribution(increasingWeights).
    $\alpha^2, \beta^2 \leftarrow$ fitBetaDistribution(decreasingWeights).
    for all $t \in T$ **do**
        $\omega_t^1 = \frac{\alpha_t^1}{\alpha_t^1 + \beta_t^1}$.
        $\omega_t^2 = \frac{\alpha_t^2}{(\alpha_t^2 + \beta_t^2)}$.
    end for
end for

to increase the predicted rainfall then the inverse of ω is used, as shown by Equation 6.12:

$$\omega_t = \begin{cases} \omega_t & \text{if } r_{actual} < r_{predicted} \\ \frac{1}{\omega_t} & \text{otherwise.} \end{cases} \tag{6.12}$$

By producing weights in this manner, we are able to predict the extremes in both directions. To avoid excessively large values being generated, we separate the weights according to whether they were under or over estimated. From understanding the data and previous experimentations, we expect weights for the positive shift to be no less than 0.3. Additionally, we would expect the full range from 0 to 1 being used to reduce the rainfall level. The process is summarised in Algorithm 5.

Sampling Future Weights

Up until the final generation, we are merely trying to estimate the best weights for the predictions produced. This is based on the evolutionary process of ϕ and κ, by fitting ω to learn on a daily basis how to achieve y. In order to evaluate the performance in

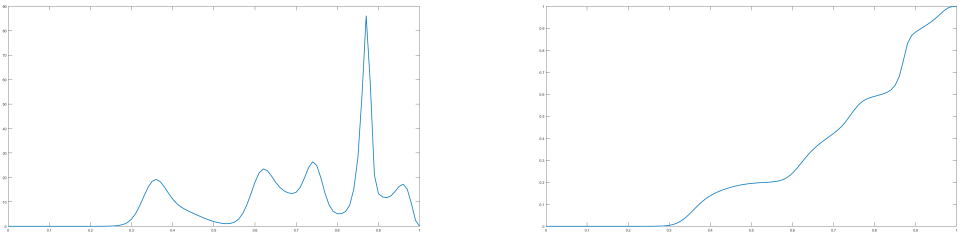
Algorithm 5 Updating weights based on new information for stochastic equations

for Generation $g = 1, ..., G$ **do**
  estimateWeights (Algorithm 4).
  for $t \in T$ **do**
    Compute new density for time t from additional information.
    Draw N samples from proposed density.
    Draw N samples from prior density.
    for $n \in N$ **do**
      if $\text{Predicted}_t^s < \text{actual}_t \ \forall s \in S$ **then**
        $\text{Predicted}_t^s \leftarrow \frac{\text{Predicted}_t^s}{\omega_t}$.
      else
        $\text{Predicted}_t^s \leftarrow \text{Predicted}_t^s * \omega_t$.
      end if
    end for
    Compute fitness.
    if newFitness $<$ oldFitness **then**
      Accept proposed density for t.
    else
      Reject and use prior density for t.
    end if
  end for
end for

the testing set and to have a stochastic process for pricing, we propose a Markovian approach to sample the weights. By creating a chain, we can produce a random walk with the final result after simulations being a density for each day.

Firstly, the weights calculated for each day are combined into a daily basis. For each day, we sum the respective PDFs to generate a mixture of beta distributions. This gives an indication of the expected weights for a particular period of time. Although we do not expect the same pattern to always occur, often it is the case that the possibility is witnessed in the past. We perform this for both sets of increasing and decreasing weights. In order to sample a random value from these new mixture of distributions created, we perform inverse transform sampling. This allows the random sampling of weights directly from the cumulative distribution function (CDF) of our new distribution. Figure 6.12 shows the PDFs and the CDF resulting from the summation of beta distributions for a given day.

Our Markov chain determines two aspects. Firstly, whether we sample from the increasing or decreasing weights. Secondly, samples from a particular area of that mixture. In order to determine the states, we calculate the transitional probabilities of moving from increasing to decreasing (denoted as $P(d|i)$), decreasing to increasing (denoted as $P(i|d)$), or stay within the same state. Ultimately, this is a two state Markov



(a) PDF (b) CDF

Fig. 6.12 The PDF and CDF of weights for Jan-01-2011.

chain similar to MCRP. However, since we usually have longer periods where we stay in the same state, we also consider the length of the under or over prediction. From previous experimentations, generally speaking the GP spends a sufficient period of time either under or over predicting. In a minority of cases, there is a frequent switching behaviour. Therefore, we also incorporate a long-run effect that decays geometrically based on the transitional probabilities of switching from either state. This is given by:

$$P(X = x) = p(1-p)^{x-1}, \tag{6.13}$$

where x is the day in the current run and p represents the probability of being in either state $P(d|i)$ or $P(i|d)$. Therefore, we are more likely to have longer runs sampling from the increasing weights or the decreasing weights. Once it is decided on a state, the probability of choosing which part of the mixture to sample from is calculated. The parts of the mixtures are directly linked to the partitions provided by the decomposition part of the DGP. Therefore, the probability is calculated conditioning on the previous day's state, namely high, medium or low. The rationale is link ω to how our decomposition perceives the range of values we expect. The motivation is that in the low state and over predicting, we expect a lower weight than normal to decrease our rainfall amount. For example, going from 200 down to 50 requires a weight much lower compared to 350 down to 200.

After calculating the relevant probabilities, we are able to create a stochastic equation for rainfall prediction by using the beta distribution. From the distribution of possible pathways, the density can be derived.

The computational steps for predicting the weights can be summarised as follows:

- Sum probabilistic densities of ω $\forall d \in D$.
- Calculate transitional probabilities.
- Calculate the renewal process.

- Extract densities for $\omega_h, \omega_m, \omega_l$.
- Calculate probabilities for $\omega_h, \omega_m, \omega_l$.
- Run Markov chain (Algorithm 6).
- Calculate median result.

Algorithm 6 Markov chain for predicting the weights

for Iteration $i = 1, \ldots, N$ **do**
  for $t \in T$ **do**
    Sample state.
    Sample weight given current state and decomposition level.
    if State = increase **then**
      $\text{predictions}_t^i \leftarrow \frac{\text{Predicted}_t^s}{\omega_t}$.
    else
      $\text{predictions}_t^i \leftarrow \text{Predicted}_t^s * \omega_t$.
    end if
  end for
end for
Error $\leftarrow$ Calculate median of predictions.

Extending Beta for the Market Price of Risk

We outline a possible extension for the calculation of our beta's considering the dynamics of the MPR. Due to the unavailability of daily trading data we cannot estimate the MPR (θ) sufficiently and we are unable to test the modification. However, we propose that when the weights are calculated, they are based on the risk-neutral adjusted beta distributions. In other words, following the extension we should be able to predict and to estimate risk-neutral densities of rainfall through the MPR parameter θ.

The extension is left as future work, but it is based on exponential tilting (equivalent of Esscher transform) of the beta distribution. Let X be a random variable with moment generating function (m.g.f.) $M(\theta) = E[\exp(\theta X)] < \infty$ and has the probability distribution given by:

$$P_\theta(X \in dx) = \frac{E[\exp(\theta X) I\{X \in dx\}]}{M(\theta)}, \tag{6.14}$$

is the exponential tilted distribution, with P_θ having a density:

$$f_\theta(x) = \frac{\exp(\theta x) f(x)}{M_X(\theta)}. \tag{6.15}$$

As the beta distribution does not belong to the natural exponential family, we cannot apply exponential tilting in a similar way to other distributions. Additionally, given that both parameters are unknown, we cannot fix θ on the shape parameters α or β. Hence, there are two forms for exponential tilting. Note that importance sampling (an MCMC algorithm), can estimate the value of θ in Equation 6.15, as suggested in Dagpunar (2007). As noted earlier, we leave this approach for future work, until we have the full trading data for rainfall derivatives.

6.6 Experimental Set-Ups

For all three methodologies, we briefly give an overview of the experimental procedure. For this chapter, we are interested in pricing rainfall derivatives. In addition, since we are proposing a new algorithm it is necessary to evaluate the rainfall predictive performance against the data sets used throughout. Assessing the predictive performance is key, because minimising predictive error leads to a decrease in mispricing (Alexandridis and Zapranis, 2013). We have two new techniques to be considered: the DGP-P and the SMGP. The former considers contract-specific equations and the latter, a stochastic equation generator for the problem of rainfall. We consider the predictive performance of these algorithms against the respective benchmark of the DGP.

We have four set-ups of the DGP-P, which are described earlier related to differing lengths of training within a year. Before pricing, we must choose the most appropriate one based on the 8 contract periods, but on the validation set within 2010. Also, we have three set-ups for the SMGP, on how the weights are tied into our model. We compare the approaches against each other on the validation set Jan-01-2010 until Dec-31-2010. The best method is then chosen in both approaches and is compared against the DGP over the testing period of Jan-01-2011 until Dec-31-2011.

When applying a new algorithm, we run iRace to determine the optimal configuration using the same search space of parameters and procedure outlined in previous chapters. For the first two methodologies, we keep the same configuration of the DGP used in Chapter 5. Table 6.2 shows the optimal configuration for the SMGP.

After presenting the predictive performance on those 42 cities, we then show the prices for cities within the USA. We expect the methodology with the lowest RMSE on the testing data to price the closest to the prices given by the CME. One important aspect to note is only Detroit, Jacksonville and New York are used for pricing, due to limited availability of accurate pricing information. Currently, the complete pricing data is unavailable and only a few prices for futures contracts exist. The futures data used is from Carmona and Diko (2005). They are not the result of actual trading

Table 6.2 The optimal configuration of the SMGP found by iRace.

GP Parameters	SMGP
Max depth of tree	11
Population size	1200
Crossover	83%
Mutation	36%
Primitive	23%
Terminal/Node bias	52%
Elitism	4%
Number of generations	35

and are the initial contract values predicted at the end of 2010. Along with our three versions of GP, we also include MCRP and Burn Analysis (BA) for a comparison of pricing. We use BA as it is the most frequently used benchmark in financial applications. It calculate prices under P based on the cost and payout of the same contract in the previous year. It computes the expected outcome over the accumulation period $I(\tau_1, \tau_2)$ with an additional risk premium that may occur. Therefore, $Q = P$ and the MPR is zero. The BA cannot price contracts on a daily basis, but acts as a reasonable benchmark.

We train the DGP from Jan-01-2001 till Dec-01-2010 before testing on the unseen test set of Jan-01-2011 until Dec-31-2011. Recall that in Section 6.3.1, we discuss the effect of sampling different numbers of individuals for the DGP before applying MCMC. Based on previous experiments, we discover that out of 100 runs, with the best 10 predictions per run saved, gives us the best performance to estimate a PDF. We report the average predictive performance over the 100 runs for each city for both the DGP and the DGP after partitioning the data (DGP-P). We run MCRP 10,000 times using the gamma and the mixed-exponential distribution to estimate the level of daily rainfall.

For the SMGP, we create 2 versions. For the first, we assume the weights to be constant ($\omega = 1$) and is named SMGPnw (equivalent of Equation 6.6). For the second, it is the best SMGP equation based on the validation set (Section 6.6.2). Both are trained on the same training set as the DGP and tested on the same unseen testing set, taking the average RMSE over 50 runs. We compare the predictive performance among: SMGP, SMGPnw, DGP and MCRP over the year long predictive accuracy of rainfall, before pricing.

Another set of experiments are for the comparison of the rainfall predictive performance for the best training and testing set-up of the DGP-P, based on results for the validation set (Section 6.6.1). We compare the average predictive accuracy of rainfall

over 100 runs after saving the best 10 predictions per run for the DGP-P. It is compared among: DGP, SMGP, SMGPnw and MCRP on the same testing intervals, using the predictions from the year long experiments.

Before we show the final results, we present the results of choosing the best approach for the DGP-P and the SMGP along with a short analysis of the results from the validation period. We summarise this information in Table 6.6 after the tuning results.

6.6.1 Choosing the Optimal Training Period

Tables 6.3 and 6.4 show the average normalised RMSE for each data set over the contract periods on the validation set, along with the minimum and maximum error. By normalising across multiple different data sets, we can validate the performance of the best method. The results are quite mixed. We decide the best based on the mean rank of each contract for all cities, with a total of 336 data sets. The mean ranks show that three contract lengths (slightly) performed the best on average with a mean rank of 2.13. This is followed by: four contract lengths, two contract lengths and one contract length, with respective mean ranks of 2.14, 2.67 and 3.05. One interesting observation is the DGP-P is not capable of capturing the extreme values very well. Also, we observe periods of much larger RMSE than others, in its normalised form. It is interesting to note that whether this behaviour is consistent within the testing set. However, we proceed with contract length of three for our testing set based on the tuning results.

6.6.2 Choosing the Optimal Weight Equation

Table 6.5 shows that in the USA, Equation 6.10 appears to perform the best with a mean rank of 1.36. In Europe, Equation 6.8 appears to perform the best with a mean rank of 1.80. From performing the mean rank across both tables, we observe that Equation 6.10 provides the lowest mean rank of 1.62. One key advantage over the previous DGP is that the computational cost of this technique is heavily reduced. Since we no longer need to run GP 100 times as required, nor perform MCMC simulations for each day. From the validation set, we proceed with Equation 6.10 to be used on our testing set (unseen during training) to compare against all other approaches. Table 6.6 shows the acronyms used for all methods for experimentation and the set-up of the SMGP and the DGP-P following the tuning process with a brief comment regarding the approach.

Table 6.3 The normalised RMSE across different training and validation lengths for the USA, including the mean and range for each set-up of DGP-P, as outlined in Section 6.4 on the validation set Jan-01-2010 - Dec-31-2010. Values in bold represent the best in average result.

Cities	One Contract Length		Two Contract Length		Three Contract Length		Four Contract Length	
	Average	Range	Average	Range	Average	Range	Average	Range
Atlanta	0.31	0.14-0.41	0.25	0.16-0.33	0.22	0.15-0.42	**0.17**	0.10-0.28
Boston	0.35	0.11-0.74	0.26	0.12-0.69	0.22	0.11-0.70	**0.14**	0.09-0.19
Cape Hatteras	0.27	0.20-0.39	0.28	0.15-0.40	**0.26**	0.13-0.40	0.27	0.14-0.39
Cheyenne	0.37	0.23-0.87	0.31	0.14-0.76	0.31	0.09-0.80	**0.27**	0.07-0.80
Chicago	0.43	0.21-0.71	0.33	0.11-0.76	0.27	0.15-0.42	**0.24**	0.09-0.46
Cleveland	0.21	0.13-0.35	0.19	0.10-0.30	0.17	0.09-0.25	**0.16**	0.09-0.24
Dallas	0.36	0.13-0.79	0.29	0.13-0.72	**0.21**	0.12-0.34	0.29	0.13-0.83
Des Moines	0.36	0.18-0.71	0.31	0.18-0.43	0.26	0.15-0.39	**0.25**	0.14-0.41
Detroit	0.31	0.14-0.43	0.28	0.11-0.38	**0.22**	0.10-0.36	0.23	0.14-0.32
Indianapolis	0.29	0.14-0.53	0.29	0.13-0.44	**0.22**	0.11-0.43	0.24	0.12-0.39
Jacksonville	0.23	0.15-0.31	0.17	0.08-0.28	0.21	0.11-0.43	**0.16**	0.10-0.25
Kansas	0.28	0.21-0.42	0.21	0.15-0.28	0.21	0.17-0.29	**0.20**	0.15-0.24
Las Vegas	0.23	0.05-0.38	0.17	0.01-0.37	**0.14**	0.00-0.32	0.18	0.02-0.41
Los Angeles	0.18	0.01-0.45	0.15	0.00-0.48	**0.09**	0.02-0.23	0.21	0.00-0.92
Louisville	0.32	0.21-0.49	0.30	0.17-0.59	**0.27**	0.14-0.44	0.29	0.14-0.52
Nashville	0.49	0.30-0.99	0.36	0.18-0.93	0.34	0.16-0.98	**0.31**	0.13-0.99
New York	0.38	0.14-0.95	0.22	0.14-0.35	0.21	0.13-0.40	**0.17**	0.12-0.23
Phoenix	0.39	0.11-0.70	0.23	0.09-0.44	0.17	0.04-0.44	**0.16**	0.05-0.40
Portland	0.47	0.17-1.12	0.35	0.12-0.92	0.29	0.09-0.66	**0.29**	0.10-0.66
Raleigh	0.29	0.13-0.47	0.21	0.13-0.33	**0.19**	0.10-0.28	0.20	0.12-0.26
St Louis	0.22	0.13-0.39	0.16	0.10-0.21	**0.15**	0.08-0.23	0.17	0.10-0.22
Tampa	0.29	0.15-0.42	0.23	0.12-0.33	0.21	0.11-0.33	**0.17**	0.07-0.26
Mean rank	3.86		2.81		1.66		**1.61**	

Table 6.4 The normalised RMSE across different training and validation lengths for Europe, including the mean and range for each set-up of DGP-P, as outlined in Section 6.4 on the validation set Jan-01-2010 - Dec-31-2010. Values in bold represent the best in average result.

Cities	One Contract Length		Two Contract Length		Three Contract Length		Four Contract Length	
	Average	Range	Average	Range	Average	Range	Average	Range
Amsterdam	0.34	0.11-0.65	0.23	0.12-0.59	0.23	0.12-0.61	**0.21**	0.11-0.62
Arkona	0.38	0.16-0.78	0.31	0.15-0.54	**0.31**	0.11-0.55	0.31	0.14-0.58
Basel	0.27	0.15-0.36	0.22	0.14-0.33	**0.18**	0.14-0.22	0.22	0.14-0.30
Bilbao	0.39	0.15-0.83	0.30	0.13-0.61	**0.26**	0.12-0.55	0.31	0.15-0.61
Bourges	0.19	0.13-0.26	0.15	0.11-0.22	**0.14**	0.11-0.22	0.16	0.09-0.32
Caceres	0.36	0.17-0.67	**0.23**	0.12-0.65	0.39	0.12-1.35	0.33	0.11-1.22
Delft	0.22	0.16-0.30	0.25	0.14-0.57	0.21	0.11-0.38	**0.16**	0.08-0.35
Gorlitz	0.41	0.20-0.64	0.35	0.21-0.76	**0.33**	0.18-0.69	0.33	0.16-0.70
Hamburg	0.31	0.16-0.71	0.21	0.11-0.26	0.17	0.09-0.22	**0.16**	0.08-0.23
Ljubljana	0.40	0.17-0.74	0.30	0.17-0.66	**0.27**	0.15-0.63	0.29	0.16-0.60
Luxembourg	0.26	0.15-0.43	0.23	0.11-0.40	**0.23**	0.13-0.39	0.23	0.13-0.44
Marseille	0.35	0.17-0.46	0.27	0.18-0.43	0.23	0.16-0.30	**0.22**	0.15-0.29
Oberstdorf	0.27	0.17-0.47	0.20	0.11-0.35	**0.19**	0.10-0.36	0.19	0.10-0.38
Paris	0.21	0.15-0.36	0.16	0.13-0.19	**0.14**	0.10-0.22	0.15	0.11-0.28
Perpignan	0.32	0.08-0.89	0.25	0.08-0.82	0.25	0.03-0.86	**0.24**	0.04-0.82
Potsdam	0.32	0.20-0.51	0.26	0.17-0.42	0.23	0.14-0.47	**0.22**	0.13-0.35
Regensburg	0.37	0.22-0.57	0.29	0.16-0.47	**0.28**	0.13-0.52	0.30	0.13-0.50
Santiago	0.85	0.14-1.81	0.26	0.16-0.37	0.23	0.08-0.35	**0.20**	0.08-0.31
Strijen	0.19	0.11-0.27	0.19	0.09-0.35	0.16	0.07-0.26	**0.15**	0.07-0.29
Texel	0.27	0.16-0.60	0.27	0.16-0.56	0.30	0.12-0.72	**0.22**	0.09-0.64
Mean rank	3.80		2.63		1.90		**1.68**	

Table 6.5 The mean RMSE for the USA (left) and Europe (right) from applying weights in three variants outlined earlier in Section 6.5.4. Values in bold represent the best value on the validation set Jan-01-2010 - Dec-31-2010.

City	Equation 6.8	Equation 6.9	Equation 6.10	City	Equation 6.8	Equation 6.9	Equation 6.10
Atlanta	524.99	543.30	**507.90**	Amsterdam	**471.31**	539.53	509.57
Boston	617.27	589.22	**545.45**	Arkona	411.68	422.23	**378.85**
Cape Hatteras	347.28	**292.03**	324.87	Basel	**374.38**	486.14	396.96
Cheyenne	**856.53**	902.22	875.38	Bilbao	302.21	319.00	**283.11**
Chicago	**276.02**	372.21	289.82	Bourges	**302.13**	310.40	329.77
Cleveland	681.11	**535.16**	618.03	Caceres	305.13	**277.02**	295.97
Dallas	780.30	658.92	**618.00**	Delft	442.21	442.21	**367.57**
Des Moines	481.20	462.93	**393.98**	Gorlitz	491.60	421.37	**408.73**
Detroit	487.96	**398.70**	418.34	Hamburg	**317.15**	376.91	334.39
Indianapolis	415.27	452.07	**369.54**	Ljubljana	482.41	**475.21**	557.44
Jacksonville	**406.43**	533.82	498.39	Luxembourg	**390.88**	406.95	411.02
Kansas	497.25	475.15	**432.49**	Marseille	**452.05**	523.43	461.57
Las Vegas	137.76	129.36	**128.96**	Oberstdorf	**564.45**	599.18	695.40
Los Angeles	332.60	399.12	**316.64**	Paris	**228.57**	283.16	272.21
Louisville	779.53	863.80	**743.29**	Perpignan	935.13	**714.48**	851.07
Nashville	**377.21**	461.66	394.55	Potsdam	423.73	414.10	**369.08**
New York	1043.1	1008.33	**846.07**	Regensburg	318.95	318.95	**304.99**
Phoenix	115.17	129.56	**111.94**	Santiago	589.27	566.60	**538.50**
Portland	512.95	435.42	**419.79**	Strijen	**577.63**	585.14	621.29
Raleigh	452.30	446.51	**409.16**	Texel	**347.95**	451.82	359.90
St Louis	**457.83**	568.82	543.01				
Tampa	660.68	669.98	**580.65**				
Mean rank	2.27	2.36	**1.36**	Mean rank	**1.80**	2.30	1.90

Table 6.6 The acronyms of all approaches with their brief description.

List of approaches	
SMGP	Stochastic Model Genetic Programming in the form $y_t = \omega_t(\phi_t + \kappa_t + \varepsilon_t)$.
SMGPnw	Same as SMGP but with weights fixed at 1 with no sampling.
DGP-P	Decomposition GP built for contract-specific equations using three contract lengths.
DGP	Decomposition GP as given in previous chapter.
MCRP	Markov chain extended with rainfall prediction used as our benchmark.

6.7 Results

Here we outline the results from the inclusion of: DGP, DGP-P, SMGP, SMGPnw, MCRP and BA. We do not include BA in the error tables for rainfall prediction, but it is included later for pricing, as it is not a predictive technique.

6.7.1 Predictive Error

Within the rainfall prediction comparison, we consider two different analysis. The first analysis is the year long predictive accuracy of each approach, excluding DGP-P, as it consider fragments of time and not the whole year. The second analysis is the predictive performance of all algorithms compared against the DGP-P segments by interpolating the predictions from: DGP, SMGP, SMGPnw and MCRP.

Yearly Approaches

We present the findings for all algorithms in Tables 6.7 and 6.8. Those starred are data sets used for rainfall derivatives. Those with a double stars indicate the availability of future prices.

One of the clear observations from Tables 6.7 and 6.8, is the consistency of the SMGP (mean rank 1.02), which has the lowest RMSE (shown in bold) for almost every city. This clearly indicates that the use of weights has a positive effect on our model. Looking at the SMGPnw without the weights (or assuming weights are 1), the improvements to the DGP are relatively small. We find the DGP (mean rank 2.83) and the SMGPnw (mean rank 2.36) often swap victories. This is an interesting result, which shows the effect of fitting seasonality, if there is no consistently reoccurring

Table 6.7 The RMSE results for the USA from SMGP, SMGPnw and DGP for the rainfall prediction over the testing period Jan-01-2011 till Dec-31-2011. Values in bold represent the best value, those cities ending in * are cities that have traded rainfall derivatives and ** are cities we have the pricing data for.

City	SMGP	SMGPnw	DGP	MCRP
Atlanta	**421.00**	499.94	526.25	667.21
Boston	**516.52**	663.19	637.68	669.11
Cape Hatteras	**309.81**	357.75	368.82	425.51
Cheyenne	**718.70**	895.92	984.52	1222.20
Chicago*	**230.01**	317.14	348.51	433.53
Cleveland	**560.77**	750.22	757.80	987.07
Dallas*	**542.82**	768.99	753.91	1070.50
Des Moines*	**346.57**	514.60	525.10	750.93
Detroit**	**396.72**	557.55	536.11	725.53
Indianapolis	**353.63**	482.65	477.87	577.34
Jacksonville**	**416.72**	490.56	527.49	764.02
Kansas*	**365.13**	456.41	480.43	599.99
Las Vegas	121.60	150.40	160.00	**106.29**
Los Angeles*	**265.32**	390.41	379.04	307.89
Louisville	**722.34**	1003.25	1101.13	1283.01
Nashville	**360.51**	459.29	493.86	816.29
New York**	**805.78**	1125.89	1103.81	1456.02
Phoenix	**104.11**	125.96	128.53	151.91
Portland*	**370.77**	499.74	537.35	642.39
Raleigh*	**390.20**	487.75	541.94	709.07
St Louis	**470.71**	594.58	619.36	823.79
Tampa	**525.95**	744.42	809.16	983.76

pattern. One of the issues we observe is that the seasonal pattern does not reflect what happens within the testing set, which has a negative impact on our model. We notice that this effect can be seen within the training data, where SMGPnw is less likely to generalise and is more prone to overfitting. MCRP was the worst performer on the testing set, in a few occasions it was outperforming the DGP and/or the SMGPnw. But it did outperform all other methods in only one instance.

We use the Friedman hypothesis test to determine any statistically significance results at the 5% significance level. Table 6.9 shows the Friedman statistic of $1.6782\text{x}10^{-21}$, which is less than the 5% significance level and shows that one or more algorithms statistically outperformed another. Therefore, we apply the Holm post-hoc test by using the SMGP (the best method) as the control method. The results are displayed within Table 6.9. We observe that the SMGP statistically outperformed all other algorithms at the 5% significance level.

Table 6.8 The RMSE results for Europe from SMGP, SMGPnw and DGP for the rainfall prediction over the testing period Jan-01-2011 till Dec-31-2011.

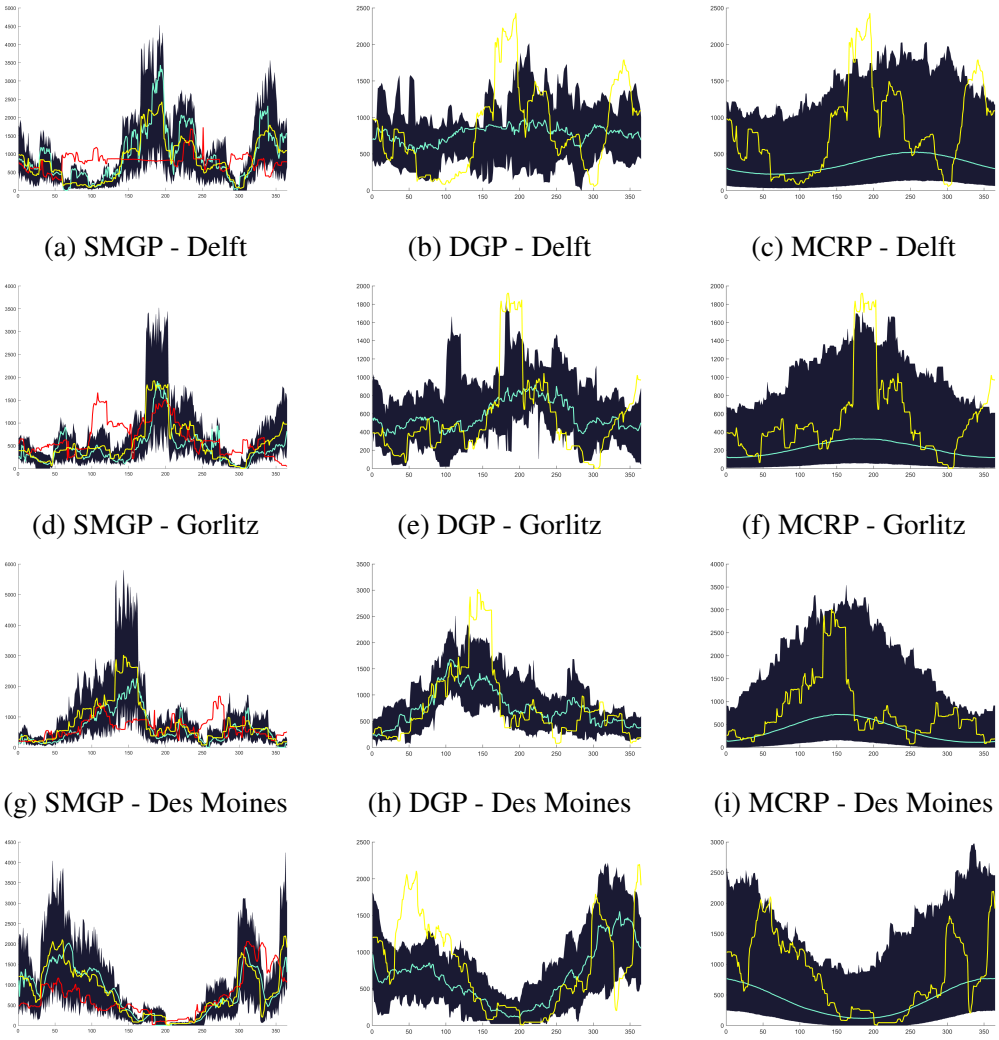
City	SMGP	SMGPnw	DGP	MCRP
Amsterdam	**428.93**	496.12	516.79	865.60
Arkona	**333.82**	428.55	451.11	632.09
Basel	**386.45**	501.33	522.22	638.98
Bilbao	**266.08**	344.78	374.76	439.66
Bourges	**327.16**	366.57	394.16	735.25
Caceres	**267.66**	324.00	352.18	424.56
Delft	**361.21**	524.01	508.75	587.84
Gorlitz	**376.23**	532.99	522.54	796.65
Hamburg	**325.22**	407.61	433.63	535.34
Ljubljana	**503.37**	579.84	637.18	639.48
Luxembourg	**380.32**	490.42	500.43	549.21
Marseille	**413.78**	506.88	517.23	559.13
Oberstdorf	**613.45**	717.01	796.68	1020.22
Paris	**256.68**	337.91	324.91	358.60
Perpignan	**780.80**	906.12	963.95	1117.57
Potsdam	**311.07**	421.86	426.12	522.21
Regensburg	**271.83**	380.56	362.44	483.31
Santiago	**499.03**	713.89	693.09	624.36
Strijen	**564.35**	633.18	688.23	564.51
Texel	**323.36**	445.84	489.94	676.33

Table 6.9 The Friedman test statistic and Holm post-hoc test with the best performing algorithm as the control method (SMGP) based on the RMSE of rainfall prediction. Values in bold represent a significant result at the 5% level.

Friedman test statistic		**1.6782x10^{-21}**	
Approach	Ranks	p-value	Holm score
SMGP	1.02	-	-
SMGPnw	**2.36**	**1.796x10^{-6}**	**0.050**
DGP	**2.83**	**1.760x10^{-10}**	**0.025**
MCRP	**3.79**	**1.084x10^{-22}**	**0.017**

On the focus of the SMGP, the best improvements can be seen in Tampa with a decrease in RMSE by 35% over the DGP, but overall we have an average decrease of 26%. This is a substantial decrease in predictive error and is the highest decrease out of all algorithms tested so far on the transformed data. One of the key aspects of

the SMGP is modifying the predicted value taking into account the irregular pattern observed in rainfall.



(a) SMGP - Delft (b) DGP - Delft (c) MCRP - Delft

(d) SMGP - Gorlitz (e) DGP - Gorlitz (f) MCRP - Gorlitz

(g) SMGP - Des Moines (h) DGP - Des Moines (i) MCRP - Des Moines

(j) SMGP - Portland (k) DGP - Portland (l) MCRP - Portland

Fig. 6.13 The 95% credible interval (shaded range) of rainfall for the year Jan-01-2011 till Dec-31-2011 for SMGP, DGP and MCRP to be used to estimate pricing over all pathways. The median observation is shown in light blue, the SMGPnw output is shown in red, and the actual rainfall in yellow.

We show in Figure 6.13, four cities and the effect that the stochastic equation has. The left column shows the 95% credible interval (shaded range), the median observation (light blue) and the SMGPnw (red), with the actual rainfall (yellow) for the SMGP. The central column shows the results for the DGP after MCMC and the right column shows the results for MCRP. The four cities used in Figure 6.13 present results broadly similar (from a qualitative perspective) to the results in the other cities. Hence, we focus on these four cities to simplify the discussion.

For the SMGP, the first observation is that all points are covered within the credible intervals. This indicates our stochastic equation can adequately predict rainfall pathways. The second observation, is the fluctuations around the median values take into account parts of the year where we observe very diverse and inconsistent rainfall periods. The third observation is the DGP predictions prior to the modification of the weights is reasonably flat, where the weights creating a more dynamic effect. Therefore, the use of weights indicates that the GP is capable of producing rainfall equations that represent a similar behaviour to rainfall. One remarkable aspect is that during the most volatile periods, our stochastic equation is capable of mimicking well.

The central column of Figure 6.13 shows the extrapolation of predictions from the DGP using MCMC to estimate a density for each day. It is possible to visualise where the improvements are realised within the SMGP. The construction of the 95% credible interval shows that the peaks and troughs of our time series are not represented adequately. Additionally, none of the four data sets show that the DGP is able to cover the minimum and maximum of the rainfall amount. This is a concern for our model when we consider pricing, because the posterior median probability is not contained within the interval, which results in the probability of pricing a derivative to be zero. Thus, it reflects in poor pricing and it causes a loss of confidence in our model.

The right column of Figure 6.13 shows the credible interval and median predictions for MCRP. The intervals of MCRP is almost capable of predicting all of the minimum and the maximums of rainfall. However, the wide variation of predictions possible for each day causes concerns and shows that predictively MCRP is very weak. It produces a substantial number of pathways not representative of the rainfall process.

We can take away the large benefit from estimating the irregularities in seasonal effect and randomly sampling according to an underlying Markovian process. The key benefit of the method is computationally less expensive and is effective. It requires fewer generations and runs to estimate a density (reflected by the GP parameters), as well as, no estimation required through MCMC. This translates to efficiency gains between 2-3 times compared to the DGP. Moreover, the predictive error is consistently reduced on the testing set over all other approaches by around 26%.

Contract-Specific Approaches

We consider the predictive performance of contract-specific results of the DGP-P against: DGP, SMGP, SMGPnw and MCRP. Since we are testing more than one period, we take the average performance over all contract periods and report the minimum RMSE and maximum RMSE. We use the predictions generated by other approaches and calculate the errors for the same period for the 8 equations produced by DGP-P.

Table 6.10 The mean RMSE results for the USA (left) and Europe (right) by partitioning the data into separate contracts based on two contract lengths prior to contract and the proceeding contract length. Only DGP-P has been specifically training and tested for each contract, whereas all other methods use the same predictions used to calculate Table 6.7. Values in bold represent the best result.

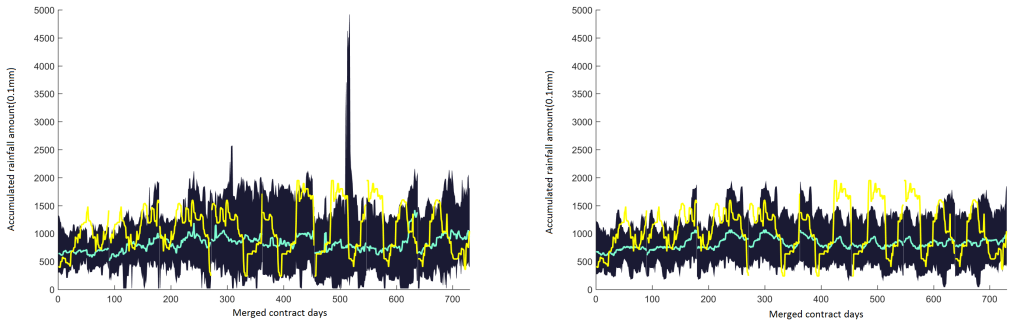
City	SMGP	SMGPnw	DGP-P	DGP	MCRP	City	SMGP	SMGPnw	DGP-P	DGP	MCRP
Atlanta	**407.95**	630.62	540.73	557.46	769.69	Amsterdam	**392.28**	551.36	532.24	516.74	1071.31
Boston	**513.21**	660.8	680.63	691.76	782.86	Arkona	**348.84**	423.75	398.33	504.66	627.79
Cape Hatteras	325.3	439.65	**304.69**	424.14	495.97	Basel	**336.02**	437.66	402.65	542.85	728.44
Cheyenne	**766.06**	932.68	825.36	1181.37	1503.31	Bilbao	**215.50**	400.46	356.27	363.54	464.28
Chicago	**220.33**	319.55	316.35	383.92	436	Bourges	**288.90**	387.01	394.75	436.63	753.63
Cleveland	**600.36**	910.54	733.16	725.9	1074.92	Caceres	**210.69**	406.13	356.99	368.03	430.19
Dallas	**645.77**	836.51	765.63	773.36	1024.25	Delft	**327.91**	599.73	473.69	478.48	533.29
Des Moines	**351.84**	684.93	625.18	651.23	620.64	Gorlitz	**434.32**	587.14	567.45	561.84	855.72
Detroit	**321.34**	764.96	487.77	497.72	617.43	Hamburg	**338.92**	395.46	391.51	436.84	586.25
Indianapolis	**300.14**	513.88	457.45	466.78	639.26	Ljubljana	**480.62**	580.83	575.02	608.83	738.86
Jacksonville	**501.81**	570.96	536.71	682.49	814.37	Luxembourg	**407.55**	574.31	455.79	495.43	524.36
Kansas	**459.68**	482.56	495.39	480.96	736.31	Marseille	**398.72**	501.81	444.87	473.27	587.31
Las Vegas	**92.06**	128.5	123.36	161.28	108.77	Oberstdorf	**604.28**	798.57	734.68	923.51	956.25
Los Angeles	**208.97**	388.17	277.51	301.64	331.91	Paris	**228.45**	344.67	326.42	333.08	459.55
Louisville	**754.45**	1261.54	903.38	950.93	1442.55	Perpignan	**680.23**	949.7	952.75	934.07	1171.88
Nashville	**301.42**	359.44	355.85	553.67	679.64	Potsdam	**361.14**	406.25	406.25	435.67	665.97
New York	**917.38**	1121.56	1000.85	1064.74	1720.87	Regensburg	**303.95**	370.91	382.04	383.21	471.23
Phoenix	**105.25**	131.5	110.96	110.96	133.85	Santiago	**526.83**	813.83	575.31	612.03	695.04
Portland	**429.44**	439.2	443.59	517.49	723.07	Strijen	**527.84**	748.51	741.03	787.95	566.99
Raleigh	**399.90**	479.12	459.95	580.2	682.13	Texel	**332.40**	461.58	438.5	470.49	610.05
St Louis	**486.71**	782.47	798.12	842.17	723.99						
Tampa	**545.88**	797.27	749.44	873.89	1068.86						
Mean rank	**1.05**	3.59	2.36	3.50	4.50	Mean rank	**1.00**	3.525	2.38	3.40	4.70

Table 6.11 The range of RMSE results for the USA by partitioning the data into separate contracts based on two contract lengths prior to contract and the proceeding contract length. Only DGP-P has been specifically training and tested for each contract, whereas all other methods use the same predictions used to calculate Table 6.7.

City	SMGP	SMGPnw	DGP-P	DGP	MCRP
Atlanta	265.23-534.67	494.94-694.92	389.33-713.77	452.58-673.60	573.80-800.65
Boston	284.09-666.31	464.23-941.73	385.01-863.14	459.13-765.22	655.73-849.77
Cape Hatteras	201.38-418.24	189.61-482.96	216.33-423.52	191.79-505.28	417.00-638.27
Cheyenne	495.90-1006.18	456.92-1227.41	573.03-1270.43	945.14-1348.79	1173.31-1833.30
Chicago	149.51-324.31	304.45-421.80	240.43-430.24	188.20-522.77	221.10-550.58
Cleveland	420.58-768.25	630.18-1087.82	579.19-938.44	462.26-947.25	631.72-1322.67
Dallas	510.25-743.66	553.67-1153.49	589.53-1010.63	678.52-1002.70	792.17-1434.47
Des Moines	291.12-457.47	514.60-730.73	500.14-862.75	504.10-724.64	420.52-1006.25
Detroit	230.1-484.00	557.55-808.45	390.22-595.08	375.28-755.92	377.28-964.95
Indianapolis	183.89-434.96	429.56-699.84	343.09-548.94	420.53-640.35	421.46-819.82
Jacksonville	337.54-591.74	358.11-652.44	424.00-697.72	480.02-727.94	473.69-1123.11
Kansas	328.62-514.83	429.03-565.95	371.54-624.19	321.89-701.43	527.99-887.99
Las Vegas	24.85-152.00	94.75-194.02	96.22-156.67	86.40-201.60	91.41-132.86
Los Angeles	156.54-318.38	261.57-538.77	199.81-346.89	197.10-458.64	295.57-381.78
Louisville	563.43-960.71	1003.25-1494.84	713.67-1219.57	632.05-1504.88	859.62-1693.57
Nashville	285.25-457.85	234.24-555.74	270.44-491.07	449.41-646.96	440.80-1036.69
New York	620.45-1047.51	1035.82-1542.47	760.65-1361.16	971.35-1368.72	1121.14-2125.79
Phoenix	82.25-136.38	120.92-181.38	78.78-148.69	86.12-163.23	92.67-208.12
Portland	218.75-541.32	274.86-669.65	350.43-536.74	279.42-725.42	359.74-931.47
Raleigh	195.10-581.40	429.22-590.18	363.36-570.34	395.62-753.30	439.62-872.16
St Louis	353.03-682.53	541.07-856.20	622.53-1013.61	588.39-929.04	535.46-1021.50
Tampa	410.24-715.29	699.75-1071.96	547.09-996.75	655.42-1165.19	767.33-1249.38

Table 6.12 The range of RMSE results for Europe by partitioning the data into separate contracts based on two contract lengths prior to contract and the proceeding contract length. Only DGP-P has been specifically training and tested for each contract, whereas all other methods use the same predictions used to calculate Table 6.8.

City	SMGP	SMGPnw	DGP-P	DGP	MCRP
Amsterdam	248.78-613.37	496.12-689.61	402.23-633.99	361.75-661.49	753.07-1177.22
Arkona	267.06-467.35	239.99-642.83	211.54-630.14	275.18-618.02	423.50-783.79
Basel	235.73-479.20	270.72-701.86	318.09-547.60	475.22-731.11	594.25-862.62
Bilbao	154.33-329.94	293.06-499.93	285.01-463.14	333.54-502.18	395.69-659.49
Bourges	248.64-415.49	311.58-439.88	280.27-548.70	299.56-532.12	507.32-999.94
Caceres	133.83-329.22	307.80-437.40	285.59-471.22	271.18-464.88	275.96-543.44
Delft	195.05-541.82	356.33-786.02	331.59-620.54	279.81-727.51	393.85-717.16
Gorlitz	252.07-530.48	351.77-735.53	448.29-686.62	485.96-684.53	740.88-1099.38
Hamburg	315.46-468.32	313.86-493.21	289.72-528.54	333.90-615.75	476.45-770.89
Ljubljana	448.00-644.31	365.30-730.60	425.51-701.52	324.96-834.71	524.37-869.69
Luxembourg	346.09-521.04	308.96-725.82	323.61-606.20	360.31-630.54	428.38-675.53
Marseille	320.37-504.81	471.40-643.74	355.90-613.92	263.79-682.74	447.30-682.14
Oberstdorf	318.99-901.77	552.10-1061.17	565.71-918.36	462.07-1187.05	765.17-1336.49
Paris	138.61-374.75	243.30-446.04	254.61-414.55	220.94-438.63	322.74-512.80
Perpignan	452.86-1093.12	897.06-1123.59	676.45-1209.99	751.88-1214.58	614.66-1575.77
Potsdam	230.19-444.83	333.27-569.51	325.00-560.63	349.42-558.22	480.43-741.54
Regensburg	236.49-331.63	209.31-525.17	286.53-458.45	282.70-507.42	241.66-700.80
Santiago	424.18-608.82	699.61-928.06	437.24-747.90	415.85-991.12	605.63-905.32
Strijen	411.98-711.08	531.87-816.80	526.13-911.47	447.35-1025.46	350.00-699.99
Texel	310.43-436.54	245.21-659.84	337.64-548.12	347.86-622.22	432.85-953.63



(a) DGP-P (b) DGP

Fig. 6.14 Effect of partitions on the prediction process for eight contracts for Detroit.

The results from Tables 6.10, 6.11 and 6.12, show a good representation of the errors obtained for each data set. We can observe that the DGP-P does reduce the predictive error in comparison to the DGP by achieving a mean rank of 2.38 compared to 3.44. However, the SMGP still comprehensively outperforms all other approaches with its mean rank of 1.02. Taking the fairest comparison of the DGP against the DGP-P, it is clear contract-specific equations work well in producing better rainfall predictions and we generally see average decreases of 8% in error.

One of the reasons for performing this comparison is to verify whether more accurate densities can be generated by using a non-stochastic method. By using the DGP as a reference point, we observe the effect in Figure 6.14. The light blue represents the median prediction, the yellow represents the actual rainfall value and the dark blue represents the 95% credible interval. On the left is the effect of all eight contracts for Detroit with the DGP-P and the right shows the predictions of the DGP split into its respective periods. Both methods' rainfall densities are estimated by MCMC for its respective mass around the mean. We observe, the rainfall amounts do appear to be more representative of the actual amounts when we consider the spread of the credible interval. We also observe, more dynamic equations are generated, but we still see the tendency for equations' predictions to be flat, especially when we consider the effect shown in Figure 6.13. In general, across all data sets, there is a tradeoff between training lengths and the generalisable nature of the DGP. We believe that a 2-3 month period for predicting one contract carries sufficient information to learn a pattern, without being affected by the irregularities of longer run predictions.

The issue with the DGP, is the lack of consideration for extreme values. However, densities are more reflective of possible rainfall pathways, as shown by more points covered by the credible interval.

Table 6.13 The coverage of all algorithms over the testing set.

Algorithm	Median	Range
DGP	71%	48% - 100%
DGP-P	76%	63% - 100%
SMGP	100%	100% - 100%
MCRP	100%	83% - 100%

Table 6.13, shows the coverage for all algorithms across all data sets. Recall that the coverage is defined by the percentage between the range of each algorithm's predictions and the range of rainfall in the data set, given by:

$$Coverage = \frac{r_{max} - r_{min}}{\hat{r}_{max} - \hat{r}_{min}}. \tag{6.16}$$

One key aspect for representative pricing is covering all potential rainfall pathways, regardlessly of a wrong prediction. After all, we are interested in the concentration of values and in the calculation of the probability of a rainfall event occurring. If the possibility is never predicted, then the probability of an event occurring is wrongly assumed to be zero. Again, this has a knock on effect to pricing, especially when it approaches the contract window. One aspect in Table 6.13, is the SMGP consistently covers 100% of points on average. As we observe in Figure 6.13, the SMGP covers in a more realistic manner compared to MCRP. Moreover, producing contract-specific equations assists the coverage of the DGP predicted on an annual basis.

In conclusion, the idea of contract-specific equations adds value to the quality of predictions generated by the DGP, both in terms of predictive accuracy and underlying nature of the data. However, it still cannot predict the extremes in the data set. The clear winner in both scenarios is the SMGP, which carries two other benefits other than the higher predictive accuracy. Firstly, the computational overhead can be significantly reduced. Secondly, it can account for extremes for rainfall. Based on the results, we expect the SMGP to price better than all other methods.

6.7.2 Pricing Performance

In the aspect of pricing performance, we fit each density (P) with the NIG distribution by using the expectation-maximisation algorithm to estimate the four parameters. The risk-neutral density follows a Lévy process, so that we are able to shift the distribution (Q) according to the MPR (θ) through the Esscher transform: NIG$(\alpha,\beta,\gamma,\delta)$ = NIG$(\alpha,\beta+\theta,\gamma,\delta)$. Once it is performed, the expected level of rainfall of the new distribution becomes our risk-neutral prices.

Table 6.14 Contract prices at the CME and proposed prices by all algorithms along with the actual rainfall for that contract period. Values in bold represent the best in absolute difference to the CME prices and underlined values represent the best in absolute difference to the actual rainfall value.

City	Contracts	SMGP	SMGPnw	DGP-P	DGP	MCRP	BA	CME	Actual rainfall
Detroit	March	**4.42**	3.65	3.52	3.59	3.75	4.26	4.20	4.46
	April	5.12	3.73	4.22	3.68	3.96	**4.3**	4.40	5.61
	May	4.12	**3.64**	2.69	3.71	4.54	3.76	3.20	5.44
	June	4.52	4.25	5.14	4.37	**4.69**	4.42	5.00	0.94
	July	6.23	3.14	5.21	2.87	**4.85**	4.93	4.50	7.67
	August	**4.31**	4.44	4.50	4.46	4.53	4.52	4.30	2.16
	September	5.35	4.79	**4.61**	4.95	3.99	3.88	4.20	6.20
	October	3.16	5.42	**4.05**	6.06	3.98	3.90	4.60	2.44
Jacksonville	March	**2.32**	2.94	2.24	2.90	2.57	2.38	2.30	1.05
	April	**2.62**	2.94	2.93	2.99	3.05	2.88	2.70	2.55
	May	2.57	4.21	**4.18**	**4.18**	3.52	3.4	4.10	2.05
	June	4.51	3.79	3.68	3.23	**3.46**	3.55	3.50	5.91
	July	6.42	2.8	**3.58**	3.32	3.69	3.24	3.60	7.33
	August	5.64	2.71	**3.05**	3.06	3.60	3.31	3.00	5.26
	September	5.23	3.51	2.86	3.21	**3.09**	3.33	3.00	6.59
	October	4.25	3.42	3.71	3.44	2.91	**2.53**	2.40	3.92
New York	March	5.23	5.12	3.47	5.32	3.99	**3.87**	3.70	6.49
	April	3.78	2.56	**2.32**	2.55	2.87	2.84	2.40	5.50
	May	3.25	2.69	**2.70**	2.67	3.18	2.5	2.80	4.82
	June	4.51	6.6	**7.94**	6.39	5.72	6.16	7.50	3.55
	July	4.59	6.32	**7.13**	6.91	7.57	6.32	7.00	3.13
	August	**7.41**	7.91	5.23	4.10	8.29	6.44	7.00	18.95
	September	8.23	6.24	6.05	4.72	6.90	**8.02**	8.10	8.67
	October	4.29	4.11	**2.86**	2.92	4.19	4.03	2.60	7.20

We present the findings for all algorithms in Table 6.14, along with the inclusion of BA. As previously mentioned, due to the unavailability of data, we only have the initial prices for Detroit, Jacksonville and New York for all eight months that contracts are traded. The prices are the initial prices at the CME and are not those from actual trading.

Table 6.14 is unable to show which is the best technique, when we consider the number of victories of each approach. We observe that the pricing predictions are fairly similar for all approaches. In order to examine the results, we use the Friedman test based on the absolute difference from each contract. We are looking to test whether or not there is a significant difference between the algorithms and the prices at the 95% confidence level. Since the Friedman test detects a statistically significant difference ($p = 0.0022$) at the 5% significance level, we use the Holm post-hoc test to compare the control (best) algorithm against each of the others. Table 6.15, shows the Friedman test statistic and results from the Holm post-hoc test, including the mean ranks.

Table 6.15 Friedman test statistic and Holm post-hoc test with the best performing algorithm as the control method, for both pricing accuracy (top section, with DGP-P as the control method) and accuracy for the actual contract rainfall amount (bottom with SMGP as the control method), both based on the absolute deviance from the CME prices and actual rainfall respectively.

Friedman test statistic		**0.0022**	
Approach	Ranks	p-value	Holm score
DGP-P	2.42	-	-
BA	2.85	0.418	0.050
MCRP	3.42	0.064	0.025
SMGPnw	3.69	0.019	0.017
DGP	**3.90**	**0.006**	**0.013**
SMGP	**4.73**	**$1.852\text{x}10^{-5}$**	**0.010**
Friedman test statistic		**$1.1256\text{x}10^{-7}$**	
SMGP	1.38	-	-
BA	**3.48**	**$9.773\text{x}10^{-5}$**	**0.050**
MCRP	**3.54**	**$6.024\text{x}10^{-5}$**	**0.025**
SMGPnw	**3.90**	**$3.046\text{x}10^{-6}$**	**0.017**
DGP-P	**4.31**	**$5.352\text{x}10^{-8}$**	**0.013**
DGP	**4.40**	**$2.225\text{x}10^{-8}$**	**0.010**

Table 6.15 backs up the first impressions from Table 6.14. It shows the hypothesis test was inconclusive, with an exception that the SMGP is statistically outperformed by DGP-P, with DGP-P priced closer to the CME than all other approaches. This result

is a surprise given that the predictive accuracy of the SMGP is much higher than all other approaches. However, the DGP-P performs exceptionally well at pricing close to the prices at the CME. In the next section, we present an alterative analysis on the predictive results for pricing and the comparison of the actual rainfall level for each algorithm. The prices presented are the initial values, they represent the most probable outcome of rainfall adjusted for risk. From the results in Table 6.14, our SMGP is proposing prices that sometimes do not agree with the CME. It would be interesting to see whether our proposed algorithm can predict the rainfall amount for each contract more accurately.

6.7.3 Comparing Actual Rainfall Amounts for Each Contract

Within rainfall derivatives, the price of a contract reflects to the accumulated level of rainfall over a period of time. Therefore, the most accurate forecast of rainfall results in the most accurate price. The exchanges, such as the CME, are responsible for pricing contracts based on all available information to make arbitrage free prices. The contracts are often written a long period in advance to give the market an indicator for the perceived value of the asset and to attract investors to the market. The ideal scenario is to predict the asset's value close to the realised value of the asset upon maturity (end of contract). However, it is not possible to guarantee that an exchange foresees exact future prices. Therefore, a proposed contract price can be some distance in value away from the actual value. This occurs more often than not.

According to theory, the proposed (initial) futures price *must* converge to the actual value of the underlying asset (Hull, 2006) [4]. However, *only* in the grains market this does not occur (Guo and Leung, 2017) due to transportation costs. This behaviour of convergence is required to eliminate opportunities of arbitrage, on the availability of more information. On a daily basis, new information regarding rainfall and other variables becomes available. The models can then be retrained to further improve predictive accuracy, as it approaches the accumulation window for the price at maturity. The closer a derivatives contract is to maturity, the more accurate the price is to the actual value (convergence). There exists less uncertainty, when there are fewer days to estimate.

The data we have is the proposed or initial contract prices for eight contracts across three cities. The prices are estimated in 2010 for all contracts trading in 2011. This means that contract prices are estimated up to eleven months in advance without any update. As we investigate in this thesis, the problem landscape of rainfall is exceptionally hard and is highly irregular with little seasonality. The observed contract

[4] It may differ very slightly due to trading costs

prices in Table 6.14, indicate that the CME and the proposed algorithms for pricing are influenced around the historical mean. Moreover, the results from BA show that the previous year has a large influence over prices. This indicates that the prices are not necessarily reflective of future events, given the analysis shows a weak annual seasonality. Therefore, there is no guarantee that the price quoted by the CME is accurate. We may observe the rainfall value to be far away from the actual accumulated rainfall level of the contract. Based on derivatives theory (Hull, 2006), we expect [5] the prices to converge towards the actual value of rainfall recorded for a day. However, if the initial price is a long way from the true price, then there exists a situation of mispricing for a long period of time. It may be just prior to the contract start date that the price reflects a reasonable estimate with the availability of weather forecasts to assist. This indicates that the method used is not appropriate and leads to issues of investors being deterred (Cao and Wei, 2004). Investors are then misinformed for long periods of time before sharp volatile price movements occur closer to the maturity of the contract.

Therefore, if we compare the predicted rainfall amounts against the actual rainfall amounts for each contract, we can observe how far away the proposed prices are. This determines whether the initial prices quoted by the CME are reflective of future events. Since we know that prices should eventually converge, it is important to generate values closer to the true value ahead of time to avoid volatile price changes. We identify this as a crucial step to attract investors, by providing more representative prices, to gain more confidence in the market. We anticipate the SMGP predicts the rainfall amount closer to the actual value. Thus, we aim to capture the underlying variable as accurately as possible, which build up the confidence and transform the market, similar to temperature derivatives.

Table 6.14 shows the actual rainfall value for the contracts we have prices for. In the majority of cases, the SMGP is closer to the real value of rainfall, with all other approaches performing equally. Looking at Table 6.15, the SMGP statistically outperforms all algorithms in this respect. This is a significant result as pricing is concerned, since our model can more accurately predict the rainfall more than 3 months ahead. This can have positive effects, as the market for rainfall derivatives is hindered by the difficulty of rainfall prediction. We observe from Table 6.14, the sheer difference between some contracts prices and the respective rainfall amounts for that contract period. This is the reason why so much uncertainty exists within these markets, as the underlying variable is so difficult to predict and hence price accurately.

[5] We cannot validate the claim of convergence in practice within weather derivatives, as no research has published the closing prices.

When the market data exists, we witness the convergence of the futures price to the actual value of the underlying asset with consideration for any uncertainty over time (Hull, 2006). We would expect the same convergence to happen in this scenario as well. In other words, we expect the difference between the price of the contract and the actual level of rainfall to be much closer when the contract approaches the contract window. With the expectation of the behaviour of a derivative value in the market, we have shown that the SMGP is a very strong tool by predicting closer to the actual accumulated rainfall value compared to all other methods, conditioning on a long period of time. This relates to improved and more confident levels of pricing, as inaccurate initial prices and large fluctuations of prices over time causes uncertainty and deters investors.

The results in this chapter, show that more accurate pricing is possible based on improvements in modelling the underlying variable. We witness this in two forms. Firstly, the DGP-P is able to price closer to the expected prices by the CME for contracts in 2011. Secondly, we are able to derive more accurate initial prices without taking into account the MPR (assuming MPR = 0). Upon the availability of all the data for pricing, we can closer examine the changes in daily prices and the convergence nearer to the true value. We expect the DGP-P to perform better based on information closer to the contract window, but overall we would expect our SMGP to be the top performer for pricing.

To summarise our findings, we find our proposed methods further reduce the rainfall predictive error. Moreover, the SMGP leads to a consistent decrease in RMSE against DGP. It achieved on average 26% reduction in error across all data sets, compared to the base algorithm of DGP. We observe the DGP-P is unable to consistently outperform its predecessor, but improves in predictive accuracy by approximately 8% on average. For pricing accuracy, all methods are similar in terms of performance, but the DGP-P has a lower mean rank in comparison to other methods. Finally, we observe in the level of rainfall prediction for each contract period, the SMGP comprehensively outperforms all other techniques. The results are very significant for the field, which increases the confidence and accuracy of pricing for rainfall derivatives.

6.8 Conclusion

Within this chapter, we recap the pricing requirements by using the Esscher transform and established the need to produce a density of rainfall predictions to calculate future prices. The Esscher transform allows for our theoretical prices to be shifted according

to the market price of risk, which can be calculated by calibrating the model to the Chicago Mercantile Exchange (CME) prices.

We then introduced three methodologies for creating a series of probabilistic densities by using the GP to calculate the probability of rainfall on a given day. Our first two methodologies were changes in the experimental procedure. The first methodology was based on Chapter 5's proposal of decomposition GP (DGP). The second methodology was the proposal of contract-specific equations (DGP-P). For both methods, we proposed a sampling approach from the final GP generation over a series of runs and we estimated the population density by using Monte Carlo Markov chain (MCMC) algorithm (Gibbs Sampler). It allows the fitting of the Normal Inverse Gaussian (NIG) distribution. Our third proposed methodology was a novel algorithm built on the basis of DGP, where we transformed the idea of deterministic models to stochastic models, named Stochastic Model GP (SMGP). Hence, it estimates a daily density directly through GP. To achieve the stochastic nature, we formulated a general model with the addition of weights that followed a beta distribution, which is randomly sampled over time.

Various sampling procedures were suggested before applying MCMC. We found that out of 100 runs by taking the best 10 individuals from the final population, led to a reasonable balance between computational overhead and quality of predictions. For DGP-P, we found that predicting on 3 month windows for each contract produced the most competitive results for our contract-specific equations. Less data resulted in poorer fitting equations, and more data caused problems of flatter predictions. Finally for the SMGP, we found that a single weight affecting the combination of the autoregressive and seasonal components showed the best performance, compared to a tradeoff approach and two weights affecting each component independently.

The rainfall prediction results showed that the SMGP was the most suitable algorithm, which outperformed the DGP and DGP-P on all data sets. It achieved the lowest predictive error and is favourable for rainfall derivatives, based on the correlation between predictive error and the pricing accuracy (Alexandridis and Zapranis, 2013; Jewson et al., 2010). Whilst we observed evidence that this statement is true, we were unable to fully test the hypothesis, because of the unavailability of daily prices. However, we noticed that the SMGP predicted the actual rainfall for each contract more accurately than all other algorithms. The results achieved contributes significantly both in literature and in practice for rainfall derivatives. Therefore, by further improving techniques for rainfall prediction, more accurate pricing can be followed through.

Chapter 7

Conclusion

7.1 Summary of Contributions

Throughout this thesis we have been addressing the problem surrounding rainfall prediction for the pricing of rainfall derivatives. Rainfall derivatives are a form of financial instrument protecting investors against the adverse risks of rainfall. The main motivation behind our research in the area of rainfall prediction and pricing rainfall derivatives is the lack of attention and research that is encountered in both. This could be due to the rapid and frequent changes in weather, in particular rainfall. Hence, the uncertainties and the unknowns make predictions more difficult for future rainfall events as well as the rainfall amount at a specific site. We set out eight original contributions within this thesis, namely:

- The superior predictive performance of machine learning methods over the currently used methodologies in rainfall derivatives. (Chapter 4)
- A new GP tailored for the problem of rainfall derivatives. (Chapter 4)
- A data transformation technique to address the issues of the underlying data. (Chapter 4)
- A new GP method called Decomposition GP (DGP) using a GA to create subproblems for the problem of rainfall prediction. (Chapter 5)
- Three new algorithms for the problem of pricing rainfall derivatives. The first two use Monte Carlo Markov chain to assist DGP to produce a probabilistic output, with one focussing on contract-specific equations and the other focussing on a single equation for all contracts. The third is a new GP method for producing stochastic equations. (Chapter 6)

- Provided more accurate rainfall futures prices than listed by the CME using SMGP. (Chapter 6)
- A thorough comparison of all proposed GPs' predictive errors against six other machine learning methods on daily and transformed data. (Chapters 4-6)
- A thorough analysis through alternative model performance measures of all proposed GP algorithms and the other machine learning benchmarks, based on climatic indicators and the coverage of all algorithms. (Chapters 4-6)

Our first major contribution in the area of Genetic Programming (GP) is found in Chapter 4, where we proposed a GP framework for the problem of rainfall prediction. This chapter also outlines the first application of GP to the problem of rainfall derivatives. The contribution provides a novel way of dealing with the problem of rainfall prediction through a transformation of the data. The daily rainfall values are transformed and summated over a sliding window. The new process not only smooths out the rainfall values, but also lessens the discontinuity that exists within the rainfall time series. We then evaluate the predictive performance of our proposed GP against MCRP and other well known machine learning methods across data sets from Europe and the USA. The predictive performance was measured by the Root Mean Squared Error (RMSE) commonly used in time series problems. Our findings show that, without using the proposed data transformation, GP and other machine learning methods are not able to fit the daily data correctly. In addition, MCRP outperforms other methods in this particular case. We also notice that GP ranks third out of seven algorithms in terms of RMSE and is not statistically outperformed by MCRP.

After the use of the proposed data transformation, we find that the machine learning methods of Radial Basis Function (RBF), Support Vector Regression (SVR) and GP (in this order) outperform MCRP, with GP performing statistically the same as RBF based on the RMSE. One issue to highlight is that the machine learning methods, including GP, provided equations that led to predictions fluctuating around the mean level of rainfall. The consequence of this behaviour gives rise to the inability of the final model to capture the dry and wet areas sufficiently well. More specifically, the final model is able to capture approximately 50% of rainfall values on average across all data sets.

To enhance the effectiveness of the GP, our next contribution was a novel extension to our GP, detailed in Chapter 5. The novelty was achieved by decomposing the problem of rainfall into a series of partitions, with an individual equation focussing on each of the three types of rainfall levels: wet, dry and normal rainfall levels. In order to determine which equation is used on a given day, we propose a Genetic Algorithm

(GA) to make the selection according to a set of evolved rules. Each rule determines, based on present rainfall information, which equation will be used for prediction (low, medium or high rainfall). We discover that this new GP, called Decomposition GP (DGP) increases the predictive accuracy, by ranking first out of seven algorithms and statistically outperforming all other algorithms, except for RBF, SVR and the original GP. DGP did lead to consistent decreases in RMSE of 8% and outperformed the original GP in 33 out of 42 cities. Additionally, DGP is able to increase the spread of the final models' rainfall predictions by predicting 73% of total rainfall values across all cities. Furthermore, we examined in detail the effect of using other classification algorithms instead of the GA in order to choose the equation to be evaluated. Our results show that the GA ranked third, producing similar results to RBF (top rank).

As our next contributions, we propose methods for determining futures contracts prices based on the predicted rainfall amount using the arbitrage free approach. However, to achieve this we needed to transform the deterministic equations evolved by DGP to a stochastic model. Our initial contribution in Chapter 6 consists of two methods for generating a density of rainfall amounts for each day, which are based on the extrapolation of rainfall amounts assisted by Monte Carlo Markov chain. We considered two different set-ups. The first one is having an equation predict rainfall amounts for a whole year, and the second one is having a separate equation for each contract. Results for rainfall prediction indicate that contract specific equations provide a lower RMSE when compared against DGP, but the computational cost is increased by producing eight models (one for each contract) instead of one.

The next contribution in Chapter 6 is a novel GP capable of producing a stochastic equation through Monte Carlo simulation, named Stochastic Model GP (SMGP). The novelty was based on a two component GP model, consisting of an autoregressive and a variable seasonal part, where both parts are controlled by weights. The weights are proportional to the probability of a certain rainfall event occurring, based on a combination of both parts. The weights were chosen via the beta distribution, thus transforming our deterministic equation to a stochastic model via the random sampling of weights. The results indicate that the obtained SMGP provides superior predictive accuracy for rainfall than DGP, and SMGP outperforms DGP in all but one city (41 times). The average decrease in RMSE of SMGP across all cities was 26% and it was capable of predicting every rainfall amount.

Our final two contributions also consisted of pricing the rainfall derivatives using the new GP methods presented in Chapter 6, based on the pricing accuracy. The results show that the first two methods, based on extrapolating the rainfall predictions, are able to provide pricing results closer to that of the CME prices than MCRP and Burn Analysis. However, in consideration of the uncertainties in predicting future events

or future prices, we find that our proposed SMGP predicts rainfall amounts much closer to the true values of rainfall amounts. Therefore, the more accurate prediction in rainfall amounts directly reflects the accuracy in predicting a derivative price. In other words, the more accurate one can predict the rainfall amount, the more precise one can price the rainfall derivatives in the future.

7.2 Future Research Directions

Based on the work presented within this thesis, the novelties can be expanded in the following areas. The first key research area is to improve the algorithm of DGP and to further explore the area of decomposing problems. The improvement can be focused on the decision criteria through a more sophisticated GA algorithm. The work that has been done so far assumes the rule lists of DGP to be static, which does not fully capture the irregularity of the data. However, by changing to a dynamic setting and considering additional weather parameters would better explain the real-time updating of information from the data. This will help to improve the guidance of DGP to more promising areas, which could lead to an improvement in predictive accuracy of rainfall. The research on DGP does not just end with improving the classification side, but extends further to integrating the GA-part alongside the GP-part of DGP. Therefore, we can create a symbiotic relationship through a co-evolutionary process where both components can exchange performance information to learn and improve from each other, rather than just using itself to evolve. Furthermore, the framework of DGP remains untested on other problem domains. This is a potential area for a more general framework of DGP to be derived, which can be formulated to deal with a range of different time series other than rainfall. This leads into the exploration of decomposition methods and whether a more robust method other than simply partitioning the data would be favourable.

The second key research area is on deriving and proving a stochastic equation to explain rainfall. This is a key step for contributions within the machine learning field, financial field and meteorological field by proving and having theoretical backing for the proposed stochastic differential equation. Performing this step would provide the literature with a general framework for predicting rainfall, where other researchers can incorporate the stochastic equation to be solved by a range of algorithms. This fundamental step opens up three main pathways. The first is on the extensions to the model, by the consideration of additional parameters explaining the rainfall process other than rainfall amounts, which has general multidisciplinary applications. The second is the specific application to finance, where the study of the market price of

risk can be examined through its inclusion within the model as a parameter. Within finance, this is not just limited to rainfall derivatives, but could be applicable to all types of derivatives. Finally, machine learning methodologies can be developed to solve the stochastic differential equation.

The third key research area is through the study of the pricing dynamics within rainfall derivatives. The availability of pricing data opens up numerous research directions within the field. The main direction is to understand how do prices change and update over time, and how can this information be incorporated to assist the prediction of the futures prices. This would allow the validation of proposed approaches in the literature, including our new GP methods. Moreover, the most important parameter within all incomplete markets, the market price of risk, can be examined in detail.

The final area of research is building on SMGP through the adoption of Bayesian inference techniques. The novel use of the weights (probability of rainfall event occurring) can be further improved through a more formal definition of the weights, allowing the use of filtration techniques and a stochastic process to decide how to better estimate and predict the weights. This can be extended beyond the weights, by transforming the underlying DGP into a probabilistic decomposition approach. Similarly, this can be extended to the seasonality component, for a dynamically changing seasonal pattern, given the nonrecurring seasonal pattern that exists in rainfall. This would allow the stochastic nature of SMGP to be exploited throughout the model, rather than being applied on top of the model.

References

Agapitos, A., O'Neill, M., and Brabazon, A. (2012). Evolving seasonal forecasting models with genetic programming in the context of pricing weather-derivatives. In *Applications of Evolutionary Computation*, volume 7248 of *Lecture Notes in Computer Science*, pages 135–144. Springer Berlin Heidelberg.

Aguilar-Rivera, R., Valenzuela-Rendón, M., and Rodríguez-Ortiz, J. (2015). Genetic algorithms and darwinian approaches in financial applications: A survey. *Expert Systems with Applications*, 42(21):7684 – 7697.

Alexandridis, A. and Kampouridis, M. (2013). Temperature forecasting in the concept of weather derivatives: A comparison between wavelet networks and genetic programming. In Iliadis, L., Papadopoulos, H., and Jayne, C., editors, *EANN*, volume 383 of *CCIS*, pages 12–21. Springer Berlin Heidelberg.

Alexandridis, A. and Zapranis, A. (2013). *Weather Derivatives: Modeling and Pricing Weather-Related Risk*. New York, NY: Springer New York.

Alexandridis, A. K., Kampouridis, M., and Cramer, S. (2017). A comparison of wavelet networks and genetic programming in the context of temperature derivatives. *International Journal of Forecasting*, 33(1):21 – 47.

Allianz (2013). *The Weather Business - How companies can protect against increasing weather volatility*.

Arganis, M., Val, R., Prats, J., Rodriguez, K., Dominguez, R., and Dolz, J. (2009). Genetic programming and standardization in water temperature modelling. *Advances in Civil Engineering*.

Arshad, J., Zameer, A., and Khan, A. (2014). Wind power prediction using genetic programming based ensemble of artificial neural networks GPEANN. In *Proceedings of the 2014 12th International Conference on Frontiers of Information Technology*, FIT '14, pages 257–262, Washington, DC, USA. IEEE Computer Society.

Azamathulla, H. M., Ghani, A. A., Leow, C. S., Chang, C. K., and Zakaria, N. A. (2011). Gene-expression programming for the development of a stage-discharge curve of the pahang river. *Water Resources Management*, 25(11):2901–2916.

Babovic, V. and Keijzer, M. (2000). Genetic programming as a model induction engine. *Journal of Hydroinformatics*, 1(1):35–60.

Babovic, V. and Keijzer, M. (2002). Rainfall runoff modelling based on genetic programming. *Nordic Hydrology*, 33(5):331–346.

Banzhaf, W. (1993). Genetic programming for pedestrians. In Forrest, S., editor, *Proceedings of the 5th International Conference on Genetic Algorithms, ICGA-93*, page 628, University of Illinois at Urbana-Champaign. Morgan Kaufmann.

Barndorff-Nielsen, O. E. (1997). Normal inverse gaussian distributions and stochastic volatility modelling. *Scandinavian Journal of Statistics*, 24(1):1–13.

Beadle, L. and Johnson, C. (2009). Semantic analysis of program initialisation in genetic programming. *Genetic Programming and Evolvable Machines*, 10(3):307–337.

Benth, F. E. and Benth, J. Š. (2012). *Modelling and pricing derivatives on precipitation*, chapter 8, pages 179–195. World Scientific.

Bühlmann, H., Delbaen, F., Embrechts, P., and Shiryaev, A. N. (1998). On Esscher transforms in discrete finance models. *ASTIN Bulletin: The Journal of the International Actuarial Association*, 28(02):171–186.

Borrelli, A., Falco, I. D., Cioppa, A. D., Nicodemi, M., and Trautteur, G. (2006). Performance of genetic programming to extract the trend in noisy data series. *Physica A: Statistical Mechanics and its Applications*, 370(1):104 – 108.

Brabazon, A., O'Neill, M., and McGarraghy, S. (2015). *Natural Computing Algorithms*. Springer Publishing Company, Incorporated, 1st edition.

Broomhead, D. S. and Lowe, D. (1988). Multivariable Functional Interpolation and Adaptive Networks. *Complex Systems 2*, pages 321–355.

Buishand, T. A. (1977). *Stochastic modelling of daily rainfall sequences*. PhD thesis, Wageningen University.

Cabrera, B. L., Odening, M., and Ritter, M. (2013). Pricing rainfall futures at the CME. *Journal of Banking & Finance*, 37(11):4286 – 4298.

Cao, M., Li, A., and Wei, J. Z. (2004). Precipitation modeling and contract valuation. *The Journal of Alternative Investments*, 7(2):93–99.

Cao, M. and Wei, J. (2004). Weather derivatives valuation and market price of weather risk. *Journal of Futures Markets*, 24(11):1065–1089.

Carmona, R. (2009). *Indifference Pricing: Theory and Applications*. Princeton University Press.

Carmona, R. and Diko, P. (2005). Pricing precipitation based derivatives. *International Journal of Theoretical and Applied Finance*, 08(07):959–988.

Castelli, M., Trujillo, L., and Vanneschi, L. (2015). Energy consumption forecasting using semantic-based genetic programming with local search optimizer. *Intell. Neuroscience*, 2015:57:57–57:57.

Castle, T. and Johnson, C. (2012). Evolving high-level imperative program trees with strongly formed genetic programming. In Moraglio, A., Silva, S., Krawiec, K., Machado, P., and Cotta, C., editors, *Genetic Programming*, volume 7244 of *Lecture Notes in Computer Science*, pages 1–12. Springer Berlin Heidelberg.

Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27.

Chin, E. H. (1977). Modeling Daily Precipitation Occurrence Process With Markov Chain. *Water Resources Research*, 13:949–956.

Cox, D. R. and Isham, V. (1988). A Simple Spatial-Temporal Model of Rainfall. *Royal Society of London Proceedings Series A*, 415:317–328.

Cox, D. R. and Isham, V. (1994). Stochastic models of precipitation. In Barnett, V. and Turkman, K., editors, *Statistics for the Environment*, volume 2, pages 3–19. Wiley, Chichester, UK.

Cramer, N. L. (1985). A representation for the adaptive generation of simple sequential programs. In *Proceedings of the 1st International Conference on Genetic Algorithms*, pages 183–187, Hillsdale, NJ, USA. L. Erlbaum Associates Inc.

Şen, Z. and Öztopal, A. (2001). Genetic algorithms for the classification and prediction of precipitation occurrence. *Hydrological Sciences Journal*, 46(2):255–267.

Dabhi, V. K. and Chaudhary, S. (2016). Financial time series modeling and prediction using postfix-gp. *Computational Economics*, 47(2):219–253.

Dagpunar, J. S. (2007). *Variance Reduction*, pages 79–105. John Wiley & Sons, Ltd.

Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30.

Donne, S., Nicolau, M., Bean, C., and O'Neill, M. (2014). Wave height quantification using land based seismic data with grammatical evolution. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 2909–2916.

Esche, F. and Schweizer, M. (2005). Minimal entropy preserves the Lévy property: how and why. *Stochastic Processes and their Applications*, 115(2):299 – 327.

Esscher, F. (1932). On the probability function in the collective theory of risk. *Scandinavian Actuarial Journal*, 1932(3):175–195.

Falco, I., Cioppa, A., and Tarantino, E. (2005). A genetic programming system for time series prediction and its application to El Niño forecast. In Hoffmann, F., Köppen, M., Klawonn, F., and Roy, R., editors, *Soft Computing: Methodologies and Applications*, volume 32 of *Advances in Soft Computing*, pages 151–162. Springer Berlin Heidelberg.

Ferreira, C. (2001). Gene expression programming: a new adaptive algorithm for solving problems. *Complex Systems*, 13(2):87–129.

Gabriel, K. R. and Neumann, J. (1962). A Markov chain model for daily rainfall occurrence at Tel Aviv. *Quarterly Journal of the Royal Meteorological Society*, 88(375):90–95.

Garg, A., Sriram, S., and Tai, K. (2013). Empirical analysis of model selection criteria for genetic programming in modeling of time series system. In *Computational Intelligence for Financial Engineering Economics (CIFEr), 2013 IEEE Conference on*, pages 90–94.

Gerber, H. and Shiu, E. (1995). Option pricing by Esscher transforms. *Insurance Mathematics and Economics*, 16(3):287.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition.

Guo, K. and Leung, T. (2017). Understanding the non-convergence of agricultural futures via stochastic storage costs and timing options. *Journal of Commodity Markets*, pages –.

Haan, C. T., Allen, D. M., and Street, J. O. (1976). A Markov Chain Model of Daily Rainfall. *Water Resources Research*, 12:443–449.

Haidar, A. and Verma, B. (2016). A genetic algorithm based feature selection approach for rainfall forecasting in sugarcane areas. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–8.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I. H. (2009). The WEKA data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18.

Hall, T., Brooks, H. E., and Doswell, C. A. (1999). Precipitation Forecasting Using a Neural Network. *Weather and Forecasting*, 14(3):338–345.

Haynes, T. D., Schoenefeld, D. A., and Wainwright, R. L. (1996). Type inheritance in strongly typed genetic programming. In *Advances in Genetic Programming 2, chapter 18*, pages 359–376. MIT Press.

Holmes, G., Hall, M., and Frank, E. (1999). Generating rule sets from model trees. In *Proceedings of the 12th Australian Joint Conference on Artificial Intelligence: Advanced Topics in Artificial Intelligence*, pages 1–12. Springer-Verlag.

Hsu, C.-M., Fu, Y.-C., Liu, Y.-C., and Peng, C.-Y. (2015). Forecasting the prices of TAIEX options by using genetic programming and support vector regression. In Ao, S. I., Castillo, O., Douglas, C., Feng, D. D., and Lee, J.-A., editors, *Proceedings of the International MultiConference of Engineers and Computer Scientists, IMECS 2015*, volume 1, pages 57–62, Hong Kong. International Association of Engineers.

Hull, J. (2006). *Options, futures, and other derivatives*. Pearson Prentice Hall, Upper Saddle River, NJ [u.a.], 6. ed., pearson internat. ed edition.

Hung, N. Q., Babel, M. S., Weesakul, S., and Tripathi, N. K. (2009a). An artificial neural network model for rainfall forecasting in Bangkok, Thailand. *Hydrology and Earth System Sciences*, 13(8):1413–1425.

Hung, N. Q., Babel, M. S., Weesakul, S., and Tripathi, N. K. (2009b). An artificial neural network model for rainfall forecasting in Bangkok, Thailand. *Hydrology and Earth System Sciences*, 13(8):1413–1425.

Jenson, B. and Nielsen, J. (1996). Pricing by 'no arbitrage'. In Cox, D., Hinkley, D., and Barndorff-Nielsen, O., editors, *Time Series Models: In econometrics, finance and other fields*, Chapman & Hall/CRC. Taylor & Francis.

Jewson, S., Ziehmann, C., and Brix, A. (2010). *Weather Derivative Valuation*. Cambridge University Press.

Katz, R. W. (1977). Precipitation as a chain-dependent process. *Journal of Applied Meteorology and Climatology*, 16(7):671–676.

Khu, S. T., Liong, S.-Y., Babovic, V., Madsen, H., and Muttil, N. (2001). Genetic programming and its application in real-time runoff forecasting1. *JAWRA Journal of the American Water Resources Association*, 37(2):439–451.

Kisi, O. and Cimen, M. (2012). Precipitation forecasting by using wavelet-support vector machine conjunction model. *Engineering Applications of Artificial Intelligence*, 25(4):783 – 792. Special Section: Dependable System Modelling and Analysis.

Kisi, O. and Shiri, J. (2011). Precipitation forecasting using wavelet-genetic programming and wavelet-neuro-fuzzy conjunction models. *Water Resources Management*, 25(13):3135–3152.

Kitchen, J. and Monaco, R. (2003). Real-Time Forecasting in Practice: The U.S. Treasury Staff's Real-Time GDP Forecast System. MPRA Paper 21068, University Library of Munich, Germany.

Konak, A., Coit, D. W., and Smith, A. E. (2006). Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering & System Safety*, 91(9):992 – 1007. Special Issue - Genetic Algorithms and Reliability Special Issue - Genetic Algorithms and Reliability.

Koza, J. (1992). *Genetic Programming: On the programming of computers by means of natural selection*. MIT Press.

Krawiec, K. and Pawlak, T. (2013). Approximating geometric crossover by semantic backpropagation. In et al., C. B., editor, *GECCO '13: Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference*, pages 941–948, Amsterdam, The Netherlands. ACM.

Kremer, E. (1982). A characterization of the Esscher-transformation. *ASTIN Bulletin: The Journal of the International Actuarial Association*, 13(01):57–59.

Langdon, W. B. (1995). Evolving data structures using genetic programming. In Eshelman, L., editor, *Genetic Algorithms: Proceedings of the Sixth International Conference (ICGA95)*, pages 295–302, Pittsburgh, PA, USA. Morgan Kaufmann.

Le Cam, L. (1961). A Stochastic Description of Precipitation. In Neyman, J., editor, *Fourth Berkeley Symposium on Mathematical Statistics and Probability*, pages 165–186.

Leobacher, G. and Ngare, P. (2011). On modelling and pricing rainfall derivatives with seasonality. *Applied Mathematical Finance*, 18(1):71–91.

Lin, G.-F., Chen, G.-R., Wu, M.-C., and Chou, Y.-C. (2009). Effective forecasting of hourly typhoon rainfall using support vector machines. *Water Resources Research*, 45(8):n/a–n/a. W08440.

López-Ibáñez, M., Dubois-Lacoste, J., Stützle, T., and Birattari, M. (2011). The Rpackageirace package, iterated race for automatic algorithm configuration. Technical report, IRIDIA, Université Libre de Bruxelles, Belgium.

Luke, S. and Spector, L. (1998). A revised comparison of crossover and mutation in genetic programming. In *Genetic Programming 1997: Proceedings of the Second Annual Conference*, pages 240–248. Morgan Kaufmann.

Manahov, V., Hudson, R., and Gebka, B. (2014). Does high frequency trading affect technical analysis and market efficiency? and if so, how? *Journal of International Financial Markets, Institutions and Money*, 28:131–157.

Manahov, V., Hudson, R., and Hoque, H. (2015). Return predictability and the 'wisdom of crowds': Genetic Programming trading algorithms, the Marginal Trader Hypothesis and the Hayek Hypothesis. *Journal of International Financial Markets, Institutions and Money*, 37(C):85–98.

Mckay, R. I., Hoai, N. X., Whigham, P. A., Shan, Y., and O'Neill, M. (2010). Grammar-based genetic programming: A survey. *Genetic Programming and Evolvable Machines*, 11(3-4):365–396.

Miller, J. F. (1999). An empirical study of the efficiency of learning boolean functions using a cartesian genetic programming approach. In Banzhaf, W., Daida, J., Eiben, A. E., Garzon, M. H., Honavar, V., Jakiela, M., and Smith, R. E., editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, volume 2, pages 1135–1142, Orlando, Florida, USA. Morgan Kaufmann.

Miller, J. F. and Thomson, P. (2000). Cartesian genetic programming. In Poli, R., Banzhaf, W., Langdon, W. B., Miller, J. F., Nordin, P., and Fogarty, T. C., editors, *Genetic Programming, Proceedings of EuroGP'2000*, volume 1802 of *LNCS*, pages 121–132, Edinburgh. Springer-Verlag.

Mislan, Haviluddin, Hardwinarto, S., Sumaryono, and Aipassa, M. (2015). Rainfall monthly prediction based on artificial neural network: A case study in tenggarong station, east Kalimantan - Indonesia. *Procedia Computer Science*, 59:142 – 151. International Conference on Computer Science and Computational Intelligence (ICCSCI 2015).

Montana, D. J. (1995). Strongly typed genetic programming. *Evol. Comput.*, 3(2):199–230.

Moraglio, A., Krawiec, K., and Johnson, C. (2012). Geometric semantic genetic programming. In Coello, C., Cutello, V., Deb, K., Forrest, S., Nicosia, G., and Pavone, M., editors, *Parallel Problem Solving from Nature - PPSN XII*, volume 7491 of *Lecture Notes in Computer Science*, pages 21–31. Springer Berlin Heidelberg.

Moustris, K., Larissi, I., Nastos, P., and Paliatsos, A. (2011). Precipitation forecast using artificial neural networks in specific regions of Greece. *Water Resources Management*, 25(8):1979–1993.

Nasseri, M., Moeini, A., and Tabesh, M. (2011). Forecasting monthly urban water demand using extended Kalman filter and genetic programming. *Expert Systems with Applications*, 38(6):7387–7395.

Nhita, F. and Adiwijaya (2013). A rainfall forecasting using fuzzy system based on genetic algorithm. In *2013 International Conference of Information and Communication Technology (ICoICT)*, pages 111–115.

Noven, R. C., Veraart, A. E. D., and Gandy, A. (2015). A lévy-driven rainfall model with applications to futures pricing. *Advances in Statistical Analysis*, 99(4):403–432.

Odening, M., Musshoff, O., and Xu, W. (2007). Analysis of rainfall derivatives using daily precipitation models: opportunities and pitfalls. *Agricultural Finance Review*, 67(1):135–156.

O'Neill, M. and Ryan, C. (2003). *Grammatical Evolution: Evolutionary Automatic Programming in an Arbitrary Language*. Kluwer Academic Publishers, Norwell, MA, USA.

Pawlak, T. P. (2016). *Geometric Semantic Genetic Programming Is Overkill*, pages 246–260. Springer International Publishing, Cham.

Plummer, M. (2003). JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. In *Proceedings of the 3rd International Workshop on Distributed Statistical Computing*.

Poli, R., Langdon, W. B., and McPhee, N. F. (2008). *A field guide to genetic programming*. Published via `http://lulu.com` and freely available at `http://www.gp-field-guide.org.uk`.

Prigent, J.-L. (1999). Incomplete markets: convergence of options values under the minimal martingale measure. *Adv. in Appl. Probab.*, 31(4):1058–1077.

Quinlan, J. R. (1992). Learning with continuous classes. In *In Proceedings of the Australian Joint Conference on Artificial Intelligence*, pages 343–348. World Scientific.

Ramesh, K., Anitha, R., and Ramalakshmi, P. (2015). Prediction of lead seven day minimum and maximum surface air temperature using neural network and genetic programming. *Sains Malaysiana*, 44(10):1389–1396.

Richardson, C. W. (1981). Stochastic simulation of daily precipitation, temperature, and solar radiation. *Water Resources Research*, 17(1):182–190.

Ritter, M., Mußhoff, O., and Odening, M. (2014). Minimizing geographical basis risk of weather derivatives using a multi-site rainfall model. *Computational Economics*, 44(1):67–86.

Rodriguez-Iturbe, I., Cox, D. R., and Isham, V. (1987). Some Models for Rainfall Based on Stochastic Point Processes. *Royal Society of London Proceedings Series A*, 410:269–288.

Rodriguez-Iturbe, I., Cox, D. R., and Isham, V. (1987). Some models for rainfall based on stochastic point processes. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 410(1839):269–288.

Rodriguez-Iturbe, I., Cox, D. R., and Isham, V. (1988). A Point Process Model for Rainfall: Further Developments. *Royal Society of London Proceedings Series A*, 417:283–298.

Rodriguez-Iturbe, I., Gupta, V. K., and Waymire, E. (1984). Scale Considerations in the Modeling of Temporal Rainfall. *Water Resources Research*, 20:1611–1619.

Rogers, A. and Prügel-Bennett, A. (1999). Modelling the dynamics of a steady state genetic algorithm. In Banzhaf, W. and Reeves, C., editors, *Foundations of Genetic Algorithms 5*, pages 57–68. Springer. Address: San Francisco.

Roldán, J. and Woolhiser, D. A. (1982). Stochastic daily precipitation models: 1. a comparison of occurrence processes. *Water Resources Research*, 18(5):1451–1459.

Ryan, C. (1994). Pygmies and civil servants. In Kinnear, Jr., K. E., editor, *Advances in Genetic Programming*, chapter 11, pages 243–263. MIT Press.

Ryan, C., Collins, J., and Neill, M. (1998). Grammatical evolution: Evolving programs for an arbitrary language. In Banzhaf, W., Poli, R., Schoenauer, M., and Fogarty, T., editors, *Genetic Programming*, volume 1391 of *Lecture Notes in Computer Science*, pages 83–96. Springer Berlin Heidelberg.

Sanso, B. and Guenni, L. (1999). A stochastic model for tropical rainfall at a single location. *Journal of Hydrology*, 214:64–73.

Schoof, J. and Pryor, S. (2008). On the proper order of Markov chain model for daily precipitation occurrence in the contiguous united states. *J. Appl. Meteor. Climatol.*, 47:2477–2486.

Seo, K., Hyeon, B., Hyun, S., and Lee, Y. (2013). Genetic programming-based model output statistics for short-range temperature prediction. In et al., A. I. E.-A., editor, *Applications of Evolutionary Computing, EvoApplications 2013: EvoCOMNET, EvoCOMPLEX, EvoENERGY, EvoFIN, EvoGAMES, EvoIASP, EvoINDUSTRY, EvoNUM, EvoPAR, EvoRISK, EvoROBOT, EvoSTOC*, volume 7835 of *LNCS*, pages 122–131, Vienna. Springer Verlag.

Shukla, A., Pandey, H. M., and Mehrotra, D. (2015). Comparative review of selection techniques in genetic algorithm. In *2015 International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE)*, pages 515–519.

Srikanth, R., George, R., Warsi, N., Prabhu, D., Petry, F., and Buckles, B. (1995). A variable-length genetic algorithm for clustering and classification. *Pattern Recognition Letters*, 16(8):789 – 800.

Stern, R. D. (1980). Analysis of daily rainfall at Samaru, Nigeria, using a simple two-part model. *Archives for Meteorology Geophysics and Bioclimatology Series B Theoretical and Applied Climatology*, 28:123–135.

Stern, R. D. and Coe, R. (1984). A model fitting analysis of daily rainfall data. *Journal of the Royal Statistical Society. Series A (General)*, 147(1):pp. 1–34.

Stowasser, M. (2011). Modelling rain risk: a multi-order Markov chain model approach. *The Journal of Risk Finance*, 13(1):45–60.

Suhaila, J., Ching-Yee, K., Fadhilah, Y., and Hui-Mean, F. (2011). Introducing the Mixed Distribution in Fitting Rainfall Data. *Open Journal of Modern Hydrology*, 1(2):11–22.

Taewichit, C., Soni, P., Salokhe, V. M., and Jayasuriya, H. P. W. (2013). Optimal stochastic multi-states first-order Markov chain parameters for synthesizing daily rainfall data using multi-objective differential evolution in Thailand. *Meteorological Applications*, 20(1):20–31.

Teller, A. and Veloso, M. (1995). Program evolution for data mining. *The International Journal of Expert Systems*, 8(3):216–236.

Todorovic, P. and Woolhiser, D. A. (1975). A Stochastic Model of n-Day Precipitation. *Journal of Applied Meteorology*, 14:17–24.

Tripathi, S., Srinivas, V., and Nanjundiah, R. S. (2006). Downscaling of precipitation for climate change scenarios: A support vector machine approach. *Journal of Hydrology*, 330(3–4):621 – 640.

Vamplew, P., Yearwood, J., Dazeley, R., and Berry, A. (2008). On the limitations of scalarisation for multi-objective reinforcement learning of pareto fronts. In Wobcke, W. and Zhang, M., editors, *AI 2008: Advances in Artificial Intelligence*, volume 5360 of *Lecture Notes in Computer Science*, pages 372–378. Springer Berlin Heidelberg.

Vanneschi, L., Castelli, M., and Silva, S. (2014a). A survey of semantic methods in genetic programming. *Genetic Programming and Evolvable Machines*, 15(2):195–214.

Vanneschi, L., Silva, S., Castelli, M., and Manzoni, L. (2014b). Geometric semantic genetic programming for real life applications. In Riolo, R., Moore, J. H., and Kotanchek, M., editors, *Genetic Programming Theory and Practice XI*, Genetic and Evolutionary Computation, pages 191–209. Springer New York.

Vapnik, V., Golowich, S. E., and Smola, A. (1996). Support vector method for function approximation, regression estimation, and signal processing. *Advances in Neural Information Processing Systems*, 9:281–287.

Vazquez, K. (2001). Genetic programming in time series modelling: an application to meteorological data. In *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, volume 1, pages 261–266 vol. 1.

Verhoest, N. E. C., Vandenberghe, S., Cabus, P., Onof, C., Meca-Figueras, T., and Jameleddine, S. (2010). Are stochastic point rainfall models able to preserve extreme flood statistics? *Hydrological Processes*, 24(23):3439–3445.

Wagner, N. and Michalewicz, Z. (2001). Genetic programming with efficient population control for financial time series prediction. In Goodman, E. D., editor, *2001 Genetic and Evolutionary Computation Conference Late Breaking Papers*, pages 458–462, San Francisco, California, USA.

Wagner, N., Michalewicz, Z., Khouja, M., and McGregor, R. R. (2007). Time series forecasting for dynamic environments: The dyfor genetic program model. *IEEE Transactions on Evolutionary Computation*, 11(4):433–452.

Wan, H., Zhang, X., and Barrow, E. M. (2005). Stochastic modelling of daily precipitation for canada. *Atmosphere-Ocean*, 43(1):23–32.

Waymire, E. and Gupta, V. K. (1981a). The mathematical structure of rainfall representations: 1. A review of the stochastic rainfall models. *Water Resources Research*, 17:1261–1272.

Waymire, E. and Gupta, V. K. (1981b). The mathematical structure of rainfall representations: 2. A review of the theory of point processes. *Water Resources Research*, 17:1273–1285.

Waymire, E. and Gupta, V. K. (1981c). The mathematical structure of rainfall representations: 3. Some applications of the point process theory to rainfall processes. *Water Resources Research*, 17:1287–1294.

Waymire, E., Gupta, V. K., and Rodriguez-Iturbe, I. (1984). A Spectral Theory of Rainfall Intensity at the Meso-β Scale. *Water Resources Research*, 20:1453–1465.

Weerasinghe, H., Premaratne, H., and Sonnadara, D. (2010). Performance of neural networks in forecasting daily precipitation using multiple sources. *Journal of the National Science Foundation of Sri Lanka*, 38(3).

Whigham, P. and Crapper, P. (2001). Modelling rainfall-runoff using genetic programming. *Mathematical and Computer Modelling*, 33(6–7):707–721.

Whigham, P. A. (1995). A schema theorem for context-free grammars. In *In 1995 IEEE Conference on Evolutionary Computation*, pages 178–181. IEEE Press.

Whigham, P. A. (1996). Search bias, language bias and genetic programming. In *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation*, GECCO '96, pages 230–237, Cambridge, MA, USA. MIT Press.

Wilks, D. (1999). Interannual variability and extreme-value characteristics of several stochastic daily precipitation models. *Agricultural and Forest Meteorology*, 93(3):153 – 169.

Wilks, D. S. (1998). Multisite generalization of a daily stochastic precipitation generation model. *Journal of Hydrology*, 210:178–191.

Wilks, D. S. and Wilby, R. L. (1999). The weather generation game: a review of stochastic weather models. *Progress in Physical Geography*, 23(3):329–357.

Woodward, J. R. (2003). GA or GP? that is not the question. In *Evolutionary Computation, 2003. CEC '03. The 2003 Congress on*, volume 2, pages 1056–1063 Vol.2.

Woolhiser, D. A. and Pegram, G. G. S. (1979). Maximum Likelihood Estimation of Fourier Coefficients to Describe Seasonal Variations of Parameters in Stochastic Daily Precipitation Models. *Journal of Applied Meteorology*, 18:34–42.

Wu, J., Liu, M., and Jin, L. (2010). A hybrid support vector regression approach for rainfall forecasting using particle swarm optimization and projection pursuit technology. *International Journal of Computational Intelligence and Applications*, 09(02):87–104.

Yin, Z., Brabazon, A., O'Sullivan, C., and O'Neill, M. (2015). Realised volatility forecasting: A genetic programming approach. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 3305–3311.

Yu, T. (2001). Polymorphism and genetic programming. In Miller, J., Tomassini, M., Lanzi, P., Ryan, C., Tettamanzi, A., and Langdon, W., editors, *Genetic Programming*, volume 2038 of *Lecture Notes in Computer Science*, pages 218–233. Springer Berlin Heidelberg.

Yuval and Hsieh, W. W. (2003). An Adaptive Nonlinear MOS Scheme for Precipitation Forecasts Using Neural Networks. *Weather and Forecasting*, 18(2):303–310.

Zameer, A., Arshad, J., Khan, A., and Raja, M. A. Z. (2017). Intelligent and robust prediction of short term wind power using genetic programming based ensemble of neural networks. *Energy Conversion and Management*, 134:361 – 372.

Zerenner, T., Venema, V., Friederichs, P., and Simmer, C. (2016). Downscaling near-surface atmospheric fields with multi-objective genetic programming. *Environmental Modelling & Software*, 84:85 – 98.

Zhou, A., Qu, B.-Y., Li, H., Zhao, S.-Z., Suganthan, P. N., and Zhang, Q. (2011). Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation*, 1(1):32 – 49.

Zumbach, G., Pictet, O., and Masutti, O. (2002). Genetic programming with syntactic restrictions applied to financial volatility forecasting. In Kontoghiorghes, E., Rustem, B., and Siokos, S., editors, *Computational Methods in Decision-Making, Economics and Finance*, volume 74 of *Applied Optimization*, pages 557–581. Springer US.