2015 IEEE Symposium Series on Computational Intelligence

Predicting Rainfall in the Context of Rainfall Derivatives Using Genetic Programming

Sam Cramer*, Michael Kampouridis*, Alex A. Freitas* and Antonis Alexandridis†
*School of Computing
University of Kent
†School of Mathematics, Statistics & Actuarial Science
University of Kent

***Abstract*—Rainfall is one of the most challenging variables to predict, as it exhibits very unique characteristics that do not exist in other time series data. Moreover, rainfall is a major component and is essential for applications that surround water resource planning. In particular, this paper is interested in the prediction of rainfall for rainfall derivatives. Currently in the rainfall derivatives literature, the process of predicting rainfall is dominated by statistical models, namely using a Markov-chain extended with rainfall prediction (MCRP). In this paper we outline a new methodology to be carried out by predicting rainfall with Genetic Programming (GP). This is the first time in the literature that GP is used within the context of rainfall derivatives. We have created a new tailored GP to this problem domain and we compare the performance of the GP and MCRP on 21 different data sets of cities across Europe and report the results. The goal is to see whether GP can outperform MCRP, which acts as a benchmark. Results indicate that in general GP significantly outperforms MCRP, which is the dominant approach in the literature.**

I. Introduction

Predicting rainfall is a major component and is essential for applications that surround water resource planning and management. Over the years numerous attempts have been made at capturing rainfall. One area where it is vital to predict the rainfall amount accurately is within rainfall derivatives. Rainfall derivatives fall under the umbrella concept of weather derivatives, which are similar to regular derivatives defined as contracts between two or more parties, whose value is dependent upon the underlying asset. In the case of weather derivatives, the underlying asset is a weather type, such as temperature or rainfall. The main difference between normal derivatives and weather derivatives is that weather is not tradeable. Hence, typical methods that exist in the literature for other derivatives are not suitable for weather derivatives.

In this problem domain the underlying asset is the accumulated rainfall over a given period, which is why it is crucial to predict rainfall as accurately as possible to reduce potential mispricing. Contracts based on the rainfall index are decisive for farmers and other users whose income is directly or indirectly affected by the rain. A lack or too much rainfall is capable of destroying a farmer's crops and hence their income. Thus, rainfall derivatives are a method for reducing the risk posed by adverse or uncertain weather circumstances. Moreover, they are a better alternative than insurance, because it can be hard to prove that the rainfall has had an impact unless it is destructive, such as severe floods or drought. Similar contracts exist for other weather variables, such as temperature and wind.

Within the literature rainfall derivatives is split into two main parts. Firstly, predicting the level of rainfall over a specified time and secondly, pricing the derivatives based on different contract periods/length. The latter has its own unique problem, as rainfall derivatives constitutes an incomplete market[1]. This means the standard option pricing models such as the Black-Scholes model are incapable of pricing rainfall derivatives, because of the violation of the assumptions of the model; namely no arbitrage pricing. Thus, a new pricing framework needs to be established. This paper focuses on the first aspect of predicting the level of rainfall. Note that it is essential to have a model that can accurately predict the level of rainfall, before pricing derivatives, because the contracts are priced on the predicted accumulated rainfall over a period of time.

In order to predict the level of rainfall for rainfall derivatives, the statistical approach of Markov-chain extended with rainfall prediction (MCRP) [1] is used. Other methods do exist, but this approach in particular is the most commonly used, and will thus be acting as a benchmark for our proposed methodology. The use of these models allows for the simulation of rainfall on a daily time scale, thus giving more flexibility in the problem domain. The reason why we are interested in daily amounts, rather than monthly or annual amount models is because the models are a lot more flexible to changes. Moreover, one is able to capture trends and more information from studying daily values. Thus, increasing the accuracy of pricing, which is crucial because contracts are priced ahead of time—sometimes this can be up to a year ahead. It is outside the scope of this paper to cover rainfall derivatives in detail. However, the path chosen reflects the literature surrounding this application such as [2], [3] and [4].

The amount of literature surrounding rainfall derivatives is quite light, due to rainfall derivatives being quite a new concept and rainfall being very difficult to accurately measure. As already mentioned, the use of MCRP is the most prevalent

[1] In incomplete markets, the derivative can not be replicated via cash and the underlying asset; this is because you can not store, hold or trade weather variables.

978-1-4799-7560-0/15 $31.00 © 2015 IEEE
DOI 10.1109/SSCI.2015.108

approach, due to its simplicity. The general approach of MCRP is often referred to as a ‘chain-dependent process’ [5], which splits the model into capturing first the occurrence pattern, and then the rainfall intensities. The occurrence pattern is produced by calculating the probability of what the outcome of today will be given what happened in the previous day(s). The process of deciding upon what state to be in is performed by a Markov-chain, where state 0 is a dry day and state 1 is a wet day. On the other hand, the intensities are produced by generating random numbers from a distribution that fits the daily data. This step is only calculated if we are in state 1, i.e. a wet day. Typically in the literature, the Gamma and Mixed-Exponential distributions provide the best fit for rain data and are most commonly used [1]. We refer the reader to [1] for a complete description of the MCRP approach.

However, even though the MCRP approach is quite popular, it faces several drawbacks. First of all, the model is very simplistic and is heavily reliant on past information being reflective of the future. Additionally, the predicted amount is essentially the average level of rainfall observed across the study period and does not take into account annual deviations in weather patterns. Furthermore, the model for each city needs to be specifically tuned as each exhibits different statistical properties, i.e. a new model for each city. Lastly, MCRP produces weak predictive models, as its only focus is on fitting the historical data. This last point is very important, as one should not only be interested in deriving models that describe past data effectively, as it currently happens; instead, we should also be focusing on producing effective predictive models, which can offer us insights on future weather trends.

Due to the disadvantages highlighted above, we divert away from the use of statistical approaches and in this paper we propose using a machine learning technique called Genetic Programming (GP). Rainfall prediction has not been covered in great detail within the machine learning literature and the applications are mainly focussed on the short term predictions i.e. up to a few hours [6]. Little literature exists for the daily predictions, e.g. [7] used a feed-forward back-propagation neural network for rainfall prediction in Sri Lanka, which was inspired by the chain-dependent approach from statistics. To the best of our knowledge, the only work that exists for daily predictions using Genetic Programming is [8]. However, the GP performed poorly by itself, although when assisted by wavelets the predictive accuracy did improve. However, there has been no previous work in using GP in the context of rainfall weather derivatives.

The goal of this paper is thus to explore whether GP is able to outperform the usual approach adopted within the rainfall derivative literature, namely MCRP. GP is chosen for this paper over other machine learning techniques, because it has the benefit of producing white box (interpretable, as opposed to black box) models, which allows us to probe the models produced. Moreover, we can capture nonlinear patterns in data without any assumptions regarding the data. This should allow us to produce a model that can reflect the ever changing process of rainfall. As a result, we could capture yearly deviations that the current MCRP is unable to replicate. Additionally, we are able to produce a more general model, which can be applied to a range of cities/climates, without having to build a new model each time.

Hence, the main contribution of this paper is that we propose a new GP for the problem of rainfall prediction, and compare its predictive performance against the performance of the current state-of-the-art MCRP approach. This will be the first step towards pricing rainfall derivatives using GP.

The remainder of this paper is organised as follows. Section II will cover the setup of the data including the data sets that will be used. Section III will outline the fitness criteria used for GP and MCRP. Section IV then reviews the MCRP approach. Section V describes in detail our proposed GP for rainfall prediction, Section VI will then discuss the experimental setup, and Section VII will discuss the results from GP and MCRP. Finally, Section VIII will conclude findings and suggest future research.

II. Data setup

There are two elements to the setup of the data, first is the number of cities we will test our experiments on, including the length of each training set. Second, is how the data will be treated and the number of attributes that will be passed to the algorithms.

A. Choice of data

The daily rainfall data used is summarised in Table I, which includes a total of 21 cities from around Europe. The cities were chosen based on two aspects, firstly, the availability of data, hence minimising the potential for missing values. The data corresponding to the European cities were provided by the National Centers for Environmental Information[2] (NCEI). Secondly, the climate of each city. In order to get an approach that can be generalised, different climates are present across the selection of cities, ranging from very wet climates to very dry climates. This is an important factor as the climate has an impact upon an algorithm’s performance, in the literature individual models are built for each city.

TABLE I
The list of all cities whose daily rainfall amounts will be used for experiments.

Cities to use for daily rainfall
Amsterdam (Netherlands), Arkona (Germany), Basel (Switzerland), Bourges (France), Bremen (Germany), Caceres (Spain), Castricum (Netherlands), De Kooy (Netherlands), Delft (Netherlands) , Gorlitz (Germany), Ljubljana (Slovenia), Luxembourg (Luxembourg), Marseille (France), Oberstdorf (Germany), Paris (France), Perpignan (France), Potsdam (Germany), Regensburg (Germany), Santiago (Portugal), Strijen (Netherlands).

[2]http://www.ncdc.noaa.gov/

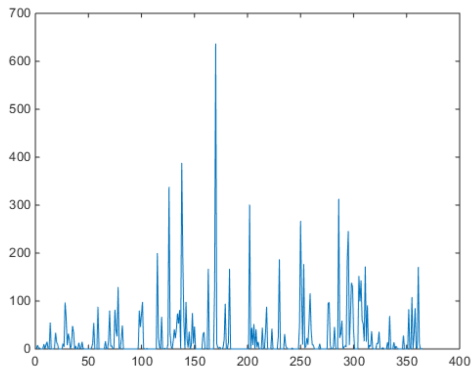

Fig. 1. The daily level of rainfall in tenths of mm of Luxembourg over the period from 01/01/2013 till 31/12/2013.

The length of data was chosen to be 10 years of daily rainfall for training and 1 year of daily rainfall for testing. We leave it as a future investigation whether different training lengths can impact the results. The length of training data is an important aspect, given climatic shifts can occur across long periods of time. Therefore, by using 10 years allows us to have sufficient observations to build a model on, without having to worry about climatic shifts within the period. Additionally, this will capture the periodic shifts in rainfall that occur each year, not associated with climatic shifts. As rainfall derivative contracts are written several months ahead of time and could span several months at a time, a testing period of 1 year is an appropriate length. Additionally, forecasting one year ahead really tests the robustness and suitability of the algorithm.

B. Treatment of data

The way the data is treated is an additional factor, as it is uncommon that giving raw data values to an algorithm will return anything of use. Therefore, the data should be transformed to better suit our problem domain. The end goal of this work is to price rainfall derivative contracts based on the accumulated amount of rainfall, over the specified contract length. For example, a contract for the month of January would require the summation of daily rainfall over 31 days. An important aspect, which should be taken into account is that contracts must be in the future, usually up to a year ahead of time and the contract period can be of any length. The most common period lengths being monthly or seasonally, but there is nothing stopping having a contract of 37 days or 164 days being specified. In addition, there is an even greater necessity for transforming the data, given the unique aspect of rainfall. Daily rainfall is one of the most volatile and hardest data sets to predict, which includes (depending upon climate) long or frequent periods of wet and/or dry spells. Findings from [8] suggest that using daily values for GP is unsuitable given the relative poor performance of their GP. Figure 1 shows the annual rainfall for Luxembourg and just how volatile and unpredictable the rainfall process is over a year.

Therefore, we propose using a sliding window approach, which will transform the data to something more manageable and better suited for the problem domain. In this work, our sliding window is defined by the length of a contract, i.e., the accumulated daily rainfall amount over the contract length. For example, pricing contracts for the month of January would require accumulating the daily rainfall amounts over 31 days. This allows us to model a contract over any given contract length, which is a crucial advantage of our methodology. Once the contract length has been specified, then the cumulative amounts will be produced using the daily rainfall amounts, as shown in Table 2. Figure 2 shows the benefit of applying a sliding window approach to the data. The output appears a lot less random, which was the motivation behind applying the sliding window, i.e., to help smooth out the data. Additionally the day-by-day volatility appears to have decreased and a pattern in rainfall is more easily noticeable. This approach is very flexible to the problem of predicting rainfall.

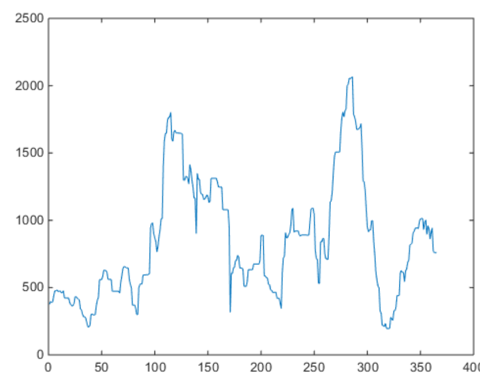

Fig. 2. The daily level of rainfall in tenths of mm of Luxembourg using the sliding window approach over the period from 01/01/2013 till 31/12/2013.

TABLE II
AN EXAMPLE OF HOW THE SLIDING WINDOW IS PERFORMED, ACROSS THREE DIFFERENT START DATES.

Start Date	Daily data points to accumulate
1st January	1st January - 31st January
2nd January	2nd January - 1st February
3rd January	3rd January - 2nd February

C. Data variables

In order to predict the accumulated amount produced by the sliding window period, data from a previous period are required in order to predict what the accumulated rainfall will be in the current sliding window. For example, if looking to predict the sliding window period January 1st 2015 - January 31st 2015 (contract length of 31 days), then only the data from December 31st 2014 and prior is available. Therefore, we define a set variables that use the sliding window approach to help predict the next sliding window. The first set is the sliding window approach but in reverse, t periods ago (where the length of a period t in this example is 31 days). For example, t–1 on January 1st 2015 would be the accumulated rainfall amount from December 1st 2014 - December 31st 2014 (the last known sliding window), t–2 would be October 31st 2014 - November 30th 2014 (the next known sliding window with no overlap of t–1) and so on. This process is done for a specified number of t's.

The second set is what was the sliding window value for a given day y years ago. For example, the value for y–1 on January 1st 2015 would be the accumulated rainfall amount from January 1st 2014 - January 31st 2014, y–2 would be January 1st 2013 - January 31st 2013 and so on. This process is done for a specified number of y's.

To sum up what we have discussed in this section, the data sets that we will use consist of 21 different European cities, from different climate types. In addition, we will use a sliding window approach to summarise the data, instead of daily predictions. Lastly, the attributes we will be using for predicting the rainfall amounts are the previous contract length periods t, e.g., t-1 period ago, t-2 periods ago, and so on, as well as the previous years y, e.g. y-1 years go, y-2 years ago, and so on.

III. Fitness (evaluation) function

For each of the following algorithms covered in Section IV and Section V the fitness used for evaluation will be the root mean squared error, given by:

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

where N is the length of the data 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).

IV. Markov-chains extended with rainfall prediction

Similar to the literature, we implement MCRP, which will act as a benchmark for our GP. MCRP's configurations are summarised in Table III. Here we opt for looking at different orders of Markov-chains, to see whether using information from the previous one or two days helps capturing the behaviour of rainfall. We will use the most commonly used distribution for rainfall modelling: gamma distribution.

Three different approaches of smoothing out the transitional probabilities and distribution parameters are used. The first is by the use of a fourier series, which is used to smooth out the daily volatility. More information regarding the implementation can be found in [9]. Therefore, having 365 different transitional probabilities and distribution parameters (one for each day of the year). The second is by the average transitional probabilities and distribution parameters across a month, thus having 12 different transitional probabilities and distribution parameters (same daily value for each month of a year). Both of these approaches are well established in the literature, whereas the third approach utilises the sliding window approach. Based on the contract length, a moving average will be calculated on the transitional probabilities and distribution parameters for each day. Due to calculating the accumulated amount for the next contract length period, the transitional probabilities and distribution parameters mimic this in the same way as described in Section II-C.

We test each configuration value of Table III (a total of 6 different combinations) on each city using one year of testing (01/Jan/2013 - 31/Dec/2013). Similar to the literature [1], we use the previous 50 years of data to tune the parameters of each city and run the process for 10,000 iterations.

TABLE III
The different configurations of the MCRP approach. For experimentation every possible combination will be tested.

MCRP configuration	Configuration values
Accumulation Method	Daily, Monthly, Contractly
Order of Markov chain	1, 2
Distribution for rainfall amount	Gamma

V. The proposed Genetic Programming method

Here we outline a tailored GP for the problem of rainfall prediction. For this paper, we opt for an extension over the original Koza type of GP [10], and use a Strongly-typed GP (STGP) [11], because we can include different types to avoid illegal trees being generated. Several modifications have been made to the STGP, which will be covered briefly here. There are three types of elements to the terminal set. The first set of elements in the terminal set includes all the variables available within the data. The variables are defined by the original y's and t's calculated from the original data. The second element is an ephemeral random constant (ERC), which will pick a uniformly distributed random number. We allow our ERC to choose a random number between the limits of -500 to 500. We want to generate a larger spread, due to predicting accumulated rainfall over a contract length, rather than daily amounts. Additionally, we allow for flexibility in our ERC and include a separate range for positive and negative numbers. Therefore, allowing a way to reduce the search space for choosing meaningful random numbers. The ERC requires four parameters to control the range of random numbers. Two parameters to control the positive range and two to control the negative range. Each different range requires a parameter for its upper bound and a parameter for its lower bound.

The third element is a set of constants from -4 to 4, at 0.25 intervals, which will take a separate type from the terminals already discussed. These are constants that are specific to the power function. Due to using a 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. We opt for choosing from within this range, to avoid excessively large numbers being created, whilst maintaining a reasonable amount of options for our GP to choose from during initialisation and evolution.

TABLE IV
GP FUNCTION AND TERMINAL SETS.

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

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, because the data includes zeroes and negative numbers. If the input is zero or negative then SQRT and LOG will return zero. If the second argument passed to DIV is zero (denominator), then zero is also returned. Protecting these values will stop NaN's (not a number) and Inf's (infinity) from being generated. The final function that has been modified is POW. It has been forced such that the second argument will be a constant within a specified range as mentioned within the previous discussion regarding the terminals. This will stop very large values from being generated, avoiding Inf's. Additionally, we allow for fractional powers, which means there is the potential for rooting negative values and producing NaN. One final check is whether the first argument (number to be raised by a power) is negative, if so then the second argument must be a whole number, which will be rounded to the nearest number if fractional. These adjustments will avoid illegal trees being generated.

Finally, another adjustment made involves dealing with negative number outputs. For this problem domain the values have to be greater than or equal to zero, it is impossible to have negative rainfall amounts. Therefore, we include a wrapper around each individual (candidate solution) to change the prediction to zero if the prediction was less than zero. The final adjustment made was to ensure a good balance between variables and random numbers in an individual. Therefore, when initialising the population using the ramped-half-and-half, we make sure that the first child is either a function or a variable, whereas the second child can either be a variable, an ERC or another function. This will avoid trees being dominated by random numbers.

All functions and terminals presented in this section are summarised in Table IV.

VI. EXPERIMENTAL SETUP

A. Parameter tuning - GP

iRace is a tool that is used to optimise parameters of most algorithms [12]. It is an iterative process and will sample many different parameter configurations and evaluate them across multiple problem instances to find an optimal configuration for the instances given to the algorithm. The advantage of using such a tool is that no prior knowledge is required and even for experienced users of a certain algorithm, iRace will consider 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 worked well on a previous problem may not necessarily work for a different problem. Across each iteration, iRace will resample configurations that performed well. Therefore, allowing iRace to search the space of the problem, and focus on promising areas.

Each city's complete data set will be split into 9 different smaller subsets consisting of 10 years of rainfall data with a preserved temporal order, and a 5 year overlap between datasets[3]. To increase generalisation and reduce issues of overfitting, we will let iRace optimise the parameters for GP using 11 out of 21 data sets. The data sets chosen are Amsterdam, Arkona, Basel, Bourges, De Kooy, Ljubljana, Luxembourg, Marseille, Potsdam, Regensburg and Santiago, which were presented earlier in Table I. To keep the process as fair as possible, we arbitrarily chose each city in regard to the data's climate. Therefore, the different climates used for optimising the GP parameters are similar to those in the remaining data sets.

To further reduce issues of overfitting when using iRace, we will split each city's training data set into a build and validate set as shown in Figure 3. The build set will consist of the first 9 years of rainfall data and the validate set will consist of the final year of rainfall data. The validation set length was chosen, such that, it is consistent with the testing set length. In total we had 99 training sets to be used by iRace, where each city had 9 different data folds.

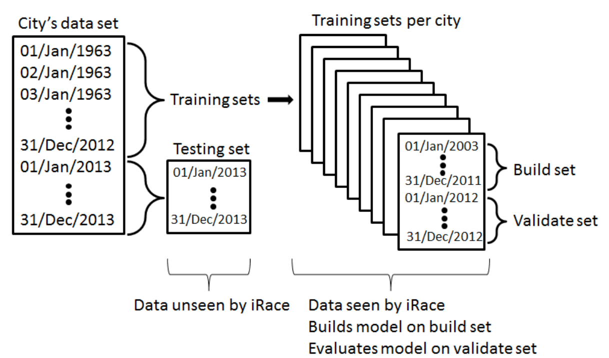

Fig. 3. The setup of each city's data set and how iRace interacts with the training set.

The results from iRace returned 5 top configurations. To decide which configuration was the best overall, we used the mean rank, which calculates on average how each approach

[3]This means that one dataset would be for the 10 year period of 2003-2012, another one for 1998-2007, another one for 1993-2002, and so on. Using such an overlap allows for the generation of a higher number of datasets.

ranked across all 11 cities. The best ranking configuration, which we will be using as part of our GP experiments, is presented in Table V.

TABLE V
THE BEST CONFIGURATION OF GP FROM OPTIMISING THE PARAMETERS USING IRACE.

GP Parameter	Run 5
Max depth of tree	8
Population size	1400
Crossover probability	0.76
Mutation probability	0.69
Primitive probability	0.55
Terminal/Node bias	0.2
Elitism percentage	0.03
Number of gens	30
ERC negative low	-495.36
ERC negative high	-102.56
ERC positive low	100.77
ERC positive high	438.58

B. 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 is 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), 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 gamma distribution parameters) we will use maximum-likelihood estimation (MLE), which is a standard technique used within rainfall prediction in estimating parameters for statistical models [9]. Therefore, we keep our benchmark consistent with the literature.

In addition, to decide which was the best MCRP configuration among the ones presented in Table III, we again used the mean rank across all cities. Results showed that the best performing approach was to use an order 2 Markov chain, with daily data, which would then be fitted into a gamma distribution. Thus, this will be the approach we will be using with MCRP.

C. Experimental methodology

Once the we have completed the choice of the best configuration for GP and MCRP, we are then ready to move on to the experimental comparison of the two algorithms. Both algorithms are tested on all 21 datasets. GP will use the full and most recent training set (01/Jan/2003 - 31/Dec/2012), before testing on the unseen test set (01/Jan/2013 - 31/Dec/2013). The same test set is used for MCRP. As GP is a stochastic algorithm, we run the configuration for 50 times on each city. For MCRP, as noted earlier in Section IV, we run our MCRP 10,000 times on each city.

VII. RESULTS

The performance of MCRP and GP is presented in Table VI based on the average RMSE performance from the testing set for each city. The table has been split between those data sets seen by iRace (top) and unseen (bottom), arranged by alphabetical order. We have chosen to do this, as the best configurations were chosen based on the validation set of the 11 cities shown in the top half. Therefore, we would expect GP to perform better on those cities, because GP's optimal parameters were selected (using iRace) based on those data sets. Whereas, the bottom 10 cities have not influenced the parameters for the best configuration of GP and help show the ability to generalise using GP. Thus, allowing us to use our best configuration on future data sets that exhibit a similar climate.

TABLE VI
THE AVERAGE RMSE PERFORMANCE IN TENTHS OF MM AND THE ABSOLUTE DIFFERENCE FOR THE BEST MCRP METHOD AND GP CONFIGURATION ACROSS EACH CITY.

Data	MCRP	GP	Absolute difference
Amsterdam	475.72	**432.14**	43.58
Arkona	283.89	**221.38**	62.51
Bilbao	980.78	**826.98**	153.80
Bourges	400.97	**341.35**	59.62
Dekooy	365.29	**348.08**	17.21
Luxembourg	424.31	**410.74**	13.57
Ljubljana	706.42	**669.97**	36.45
Marseille	890.82	**264.08**	626.74
Potsdam	263.30	**221.69**	41.61
Regensburg	**378.40**	400.38	21.98
Santiago	**1387.84**	1428.11	40.27
Basel	281.20	**269.93**	11.27
Bremen	**278.26**	279.15	0.89
Caceres	717.11	**531.65**	185.46
Castricum	539.83	**489.17**	50.66
Delft	569.33	**471.50**	97.83
Gorlitz	331.18	**263.56**	67.62
Oberstdorf	678.43	**651.18**	27.25
Paris	257.14	**247.61**	9.53
Perpignan	955.02	**516.50**	438.52
Strijen	371.01	**353.01**	18

As we can observe, across the 21 different cities, GP outperforms MCRP 18 times. There are only 3 occasions where MCRP outperforms GP. This is a remarkable result, which demonstrates the superiority of our GP against the MCRP approach, which as we have already explained is currently considered the state-of-the-art in the domain of rainfall prediction for weather derivatives. It is also worth noting that the GP has in several cases introduced a substantial reduction in the RMSE values, e.g. in Marseille from 890.82 to 264.08, a gain of 626.74, and in Perpignan from 955.02 to 516.50, a gain of 438.52. Another interesting point is in the three cases where MCRP outperformed GP, the differences are very small; on average a gain of 21.05.

To firstly check which of our algorithms performed better in terms of wins, we will work out the mean rank based on Table VI — the lower the rank, the better the algorithm's performance. Furthermore, in order to determine whether the above results are statistically significant, we compare the two approaches by using the Wilcoxon signed-rank test [13]. The Wilcoxon signed-rank test is a nonparametric test for comparing two related samples, to test whether the mean ranks of these samples differ. Essentially, the Wilcoxon signed-rank performs a paired test by comparing the difference between the performance of both methods on each city. The null hypothesis is that there is no significant difference between the average RMSE of GP and MCRP. We apply the test at the 5% significance level.

TABLE VII
THE MEAN RANKINGS OF MCRP AND GP, AND THE WILCOXON SIGNED-RANK p-VALUE TO TEST WHETHER GP OR MCRP STATISTICALLY OUTPERFORMED THE OTHER.

Approach	Ranking
MCRP	1.86
GP	**1.14**
Wilcoxon p-value	0.0007

Table VII shows the mean rank of both MCRP and GP, a value of 1.86 and 1.14 respectively, where a lower rank indicates better performance. Therefore, across each city on average GP outperformed MCRP. Also shown, is the p-value rounded to 4 decimal points for the Wilcoxon signed-rank test. As we can observe, the Wilcoxon signed-rank statistic has a value of 0.0007, which is less than the 5% significance level and is in fact significant at the 99.9% level. Therefore, there is strong evidence to reject the null hypothesis, and conclude that our proposed GP statistically outperformed the MCRP approach.

From the above results, we can conclude that GP is a suitable method for predicting rainfall in the context of weather derivatives, by statistically outperforming the current state-of-the-art (MCRP). This is an important result, as it indicates that GP is able to outperform the most commonly used statistical methods currently used within the rainfall derivatives application domain. Moreover, having more accurate rainfall predictions could help increase the accuracy of pricing rainfall derivatives, which as we explained at the beginning of this paper, is another important problem of the field of weather derivatives. Lastly, as we are able to give more confidence surrounding the prediction of rainfall, this will help to reduce potential mispricing and attract more investors to the rainfall derivative market.

VIII. Conclusion

This paper proposes a new approach to predicting rainfall for the application of rainfall derivatives. The motivation is to provide a better methodology to overcome the weaknesses of the current approach of the Markov-chain extended with rainfall prediction (MCRP). Such an approach does not have the same predictive power as other nonparametric approaches such as Genetic Programming (GP). In fact, MCRP tends to average out the past historical data, which is unable capture patterns within the data.

Strongly-typed Genetic Programming (STGP) was our chosen methodology, due to producing white box (interpretive) models and to being a technique that can detect and learn from nonlinear data. Furthermore, STGP was chosen over the standard GP, because we can influence types to avoid illegal trees being created. In this paper we compared our STGP against MCRP, which is currently used within the literature. This was the first application of GP within the context of rainfall derivatives.

Instead of using daily data to feed into our GP, which is the data required for MCRP, we proposed using a sliding window for the problem domain. Thus, instead of accumulating rainfall after predicting it on a daily basis, we modeled directly on the contract length that we are interested in pricing. We implemented the most common MCRP methods that are used within the literature, to act as our benchmark. When comparing GP against MCRP, we found sufficient evidence to suggest that GP is capable of predicting rainfall for various different climates significantly better than the MCRP approach.

Future work will include testing other state-of-the-art regression algorithms to compare against GP, to see how effective GP really is. Furthermore, we will investigate whether the construction of new features can improve the GP performance. Lastly, since we have obtained promising rainfall prediction results, we can also move towards the pricing task of rainfall derivatives and investigate if our current results have an overall positive effect in pricing.

References

[1] D. S. Wilks, "Multisite generalization of a daily stochastic precipitation generation model," *Journal of Hydrology*, vol. 210, pp. 178–191, Sep. 1998.

[2] B. L. Cabrera, M. Odening, and M. Ritter, "Pricing Rainfall Derivatives at the CME," *SFB 649 Discussion Papers*, Jan. 2013.

[3] M. Ritter, O. Muhoff, and M. Odening, "Minimizing geographical basis risk of weather derivatives using a multi-site rainfall model," *Computational Economics*, vol. 44, no. 1, pp. 67–86, 2014.

[4] G. Leobacher and P. Ngare, "On modelling and pricing rainfall derivatives with seasonality," *Applied Mathematical Finance*, vol. 18, no. 1, pp. 71–91, 2011.

[5] R. W. Katz, "Precipitation as a chain-dependent process," *Journal of Applied Meteorology and Climatology*, vol. 16, no. 7, pp. 671–676, 1977.

[6] N. Q. Hung, M. S. Babel, S. Weesakul, and N. K. Tripathi, "An artificial neural network model for rainfall forecasting in bangkok, thailand," *Hydrology and Earth System Sciences*, vol. 13, no. 8, pp. 1413–1425, 2009.

[7] H. Weerasinghe, H. Premaratne, and D. Sonnadara, "Performance of neural networks in forecasting daily precipitation using multiple sources," *Journal of the National Science Foundation of Sri Lanka*, vol. 38, no. 3, 2010.

[8] O. Kisi and J. Shiri, "Precipitation forecasting using wavelet-genetic programming and wavelet-neuro-fuzzy conjunction models," *Water Resources Management*, vol. 25, no. 13, pp. 3135–3152, 2011.

[9] D. A. Woolhiser and G. G. S. Pegram, "Maximum Likelihood Estimation of Fourier Coefficients to Describe Seasonal Variations of Parameters in Stochastic Daily Precipitation Models." *Journal of Applied Meteorology*, vol. 18, pp. 34–42, Jan. 1979.

[10] J. Koza, *Genetic Programming: On the programming of computers by means of natural selection*. MIT Press, 1992.

[11] D. J. Montana, "Strongly typed genetic programming," *Evolutionary Computation*, vol. 3, no. 2, pp. 199–230, 1995.

[12] M. López-Ibáñez, J. Dubois-Lacoste, T. Stützle, and M. Birattari, "The Rpackageirace package, iterated race for automatic algorithm configuration," IRIDIA, Université Libre de Bruxelles, Belgium, Tech. Rep., 2011.

[13] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. pp. 80–83, 1945.