Documentation on R System for Climate Data Analysis

African Maths Initiative

BY

Frederic Ntiremganya

October 07, 2015

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES iii

ACKNOWLEDGMENT iv

INTRODUCTION 5

SIAC PRACTICALS 6

IMPORTING THE DATA FROM CSV 7

DEMONSTRATION FOR THE USERS 8

Start of the Rains 8

End of the rains 11

Seasonal Summary Rain 13

Extremes Events 16

Date in the format day and month 18

Adding a column on the data set like year, month or day 20

Water balance 21

Display the Spell Length Table 23

Display Day of the Year 24

Display daily data 27

Compare graphs 29

Cumulative and exceedance graph 32

Box-plot with jitter 36

Plot yearly summaries 38

LIST OF TABLES AND FIGURES

Table 1 Table indicating the functions by each topic 7

Table 2 List all the arguments with their defaults for start of the rain method 11

Table 3 List all the arguments with their defaults for end of the rains method 13

Table 4 List all the arguments with their defaults for seasonal summary rain method 15

Table 5 List of all the arguments with their defaults for extreme events method 18

Table 6 List of all the arguments with their defaults for day_month method 19

Table 7 List of all the arguments with their defaults for display water balance method 22

Table 8 List of all the arguments with their defaults for display spell length method 24

Table 9 List of all the arguments with their defaults for display day of the year method 25

Table 10 List of all the arguments with their defaults for display daily data method 28

Table 11 List of all the arguments with their defaults for plot yearly comparison method 30

Table 12 List of all the arguments with their defaults for cumulative and exceedance graph method 34

Table 13 List of all the arguments with their defaults for boxplot with data point method 38

Table 14 List of all the arguments with their defaults for plot yearly summaries method 39

Figure 1 Comparison of two definitions of start of the rains in graph 31

Figure 2 Comparison of two variables using vertical lines 32

Figure 3 Cumulative graphs 35

Figure 4 Exceedance graph 36

Figure 5 Boxplot with data points 38

Figure 6 Number of raindays per year 40

ACKNOWLEDGMENT

This is documentation for the work done by African Maths Initiative (AMI) for the UK met contract. We would like to acknowledge UK meteorological Office (UK met) for their support and development team for climate system in R for their effort used in this project.

INTRODUCTION

Climate data analysis is easily carried out using Instat. Instat is a simple statistic package with an additional menu for climate data analysis. Instat was used for the last run of e-SIAC to calculate events that are of interest to farmers such as *start of the rains* and *end of the rains* including other summaries such as *length of rainy season* and *rainfall totals*.

A team working for African Maths Inititiative (AMI), based in Maseno, Western Kenya has carried out the same analyses done in Instat using R, an open source programming language and software environment for statistical computing and graphics. Functions have been written in R that can provide the same output as those used in Instat for the e-SIAC course.

Initially, these functions were written as standalone functions. However, after the development by SSC, University of Reading, in collaboration with AMI, on a climate system in R, it was decided that a better way to deliver these functions was by incorporating them into this climate system. Standalone functions require data to be input by the user in a very specific format. In contrast, the climate system is capable of dealing with data in many formats. It also allows analysis to be done on several data frames, with a single command. For these reasons, it was decided that these functions would be delivered as part of the climate system.

The key to this new system is a climate object, which consists of a set of climate data objects. A climate data object is a "rectangle", plus meta-data, that takes time-series data in a form that is supplied by the data management system, e.g. CLIMSOFT or CLIDATA. The climate object can include data from alternative sources, e.g. sea-surface temperatures and recognizes common climatic elements, so that analyses can be performed that are appropriate for that element. This part of the system can be extended when needed.

The climate object also includes methods (functions) which produce the products that are needed. Some functions may manipulate the data, providing results that are again stored within the climate object for further analysis. Other functions produce results for users.

Hence, a climate object consists of the actual data, with some meta-data, plus a series of "methods". These "methods" are the functions that we have been developing. The user now

needs to know how to specify the analysis they need, not how the data need to be supplied for the analysis to work.

This document aims to explain the climate methods that have been developed by AMI for the climate system and to demonstrate how the users can use the climate system in R to carry out their climate data analyses. Here we demonstrate the methods which produce the products depending on the interest of the user.

Below are the methods that will be described and demonstrated in this document:

- Add_start_rain
- Add_end_rain
- Add_year_month_day_col
- Display_water_balance
- Display_spell_length
- Display_doy
- Display_daily
- Day_month
- Plot_yearly_comparison
- Cumulative_exceedance_graph
- Box_jitter
- plot_yearly_summary

SIAC PRACTICALS

The climate object methods are used in the first 7 topics where Instat is used in practicals assignments. The first role was going through each assignment and identifies the tasks where Instat was used then produce method which does the same output.

The following table indicates the functions by each topic in eSIAC which are used.

Topics	Function(s)
Topic 1	• Day_month • Seasonal_summary • Add_year_month_day_col

Topic 2	• Add_start_rain • Display_daily • Spell_length • Display_spell_length • Plot_yearly_comparison
Topic 3	• Add_start_rain • Add_end_rain • Display_water_balance • Display_daily
Topic 5	• Box_jitter • Plot_yearly_comparison
Topic 7	• Cumulative_exceedance_graphs • Seasonal_summary • Extremes_events • Plot_yearly_comparison

Table 1 Table indicating the functions by each topic

IMPORTING THE DATA FROM CSV

The first step of the data analysis is to import the data into R. We used different datasets from Statistics in Applied Climatology (SIAC) course to create a single object (siac_obj) in order to

demonstrate the process. The format of these data sets was csv and we read the datasets into three different data frames as follows:

SAMSMALL <- read.csv("C:/Users/Documents/AMI/SIAC/SIAC_docs/R_e_SIAC_docs/Data_sets/SAMSMALL3.csv")

samaru56 <- read.csv("C:/Users/Documents/AMI/SIAC/SIAC_docs/R_e_SIAC_docs/Data_sets/samaru56_2.csv")

samrain <- read.csv("C:/Users/Documents/AMI/SIAC/SIAC_docs/R_e_SIAC_docs/Data_sets/SAMRAIN.csv")

We then created a single climate object of all data frame together called *siac_obj* as follows:

siac_obj <- climate$new(data_tables = list(samaru = samaru56, samsmall = SAMSMALL, samrain = samrain, kibos = kibos), date_formats = list("%m/%d/%Y","%m/%d/%Y", " ", "%m/%d/%Y")),data_time_periods = list("daily", "daily", "yearly", "daily"))

As the above example shows, a climate object can contain multiple data frames, each for a single station. This allows us to carry out analysis on multiple data frames with a single command, as shown in the examples below.

DEMONSTRATION FOR THE USERS

We now proceed to analyse the data using the new methods developed.

Start of the Rains

The first question when in dry season is when the rains will come. In particular, farmers are concerned about the *start of the rains* which is an important *event of interest* for them.

Consider the following four definitions for the start of the rains as given in the online course Statistics in Applied Climatology (e-SIAC):

- The first day from ***April 01*** that gets more than ***20 mm*** on a singles day, or totalled over ***2*** consecutive days
- As definition (i), with the additional condition that there is no ***10 day*** (or longer) dry spell in the next ***30 days***.
- As (i) but with ***May 01*** as the earliest possible day.
- As (ii) but with ***May 01*** as the earliest possible day.

The items in bold can be changed, depending on the site or crop etc.

With these definitions in mind, we created a start of the rains method that is capable of calculating the start of the rains using any of the definitions.

The command:

siac_obj$add_start_rain(col_name = "Strt1")

will calculate the start of the rains with default values, which corresponds to definition (i).

This method moves from daily data to a yearly summary. The calculations done by *add_start_rain()*are stored in data frames containing yearly summaries for each of the data sets in *siac_obj*. This is done automatically by the method. If no yearly summary data frames have been produced up to this point, they will automatically be created and then the start of the rains column will be appended to each of the yearly summary data frames.

There is no visible output from this method but we can view the yearly summary data frame as shown below, for one of the original data sets, to see if the start of the rains column has been appended:

View(siac_obj$used_data_objects[[“samaru yearly”]]$data)

Here we display the 6 rows of the summary by using head command:

```
view(head(siac_obj$used_data_objects[["samaru yearly"]]$data))
```

	Year	Date	Total Rain	Number of Rainy Days	Mean Rain per Rainy Day	Strt1
1	1928	1928-01-01	1262.34	85	14.77953	115
2	1929	1929-01-01	1284.18	94	13.62915	126
3	1930	1930-01-01	1044.73	84	12.36774	108
4	1931	1931-01-01	1197.82	76	15.69724	118
5	1932	1932-01-01	1198.16	76	15.72513	115
6	1933	1933-01-01	1311.61	92	14.20707	156

The above output is the 6 first lines of the yearly summary data frame for the samaru climate object in the analysis.

To calculate using definition (ii) we specify that the dry spell condition is also required:

siac_obj$add_start_rain(data_list = list("samaru"), dry_spell_condition=TRUE, col_name = "Strt2")

Again, there is no visible output from this method but if we view one of the yearly summary data frame (i.e samaru yearly summary) we see that a second start of the rain column has been appended:

	Year	Date	Total Rain	Average rain per rain day	Number of Rain Days	Strt1	Strt2
1	1928	1928-01-01	1262.34	14.77953	85	115	115
2	1929	1929-01-01	1284.18	13.62915	94	126	126
3	1930	1930-01-01	1044.73	12.36774	84	108	168
4	1931	1931-01-01	1197.82	15.69724	76	118	118
5	1932	1932-01-01	1198.16	15.72513	76	115	115
6	1933	1933-01-01	1311.61	14.20707	92	156	156

For the third definition (iii) we specify the *earliest_day* argument to be 122 and drop the dry spell condition:

siac_obj$add_start_rain(data_list = list("samaru"),col_name = "Strt3", earliest_day = 112)

For the fourth definition (iv), we add the dry spell condition to definition (iii):

siac_obj$add_start_rain(data_list = list("samaru"),col_name = "Strt4", earliest_day = 112, dry_spell_condition = TRUE)

Other arguments may be specified in the method for added flexibility. Below we list all the arguments, with their defaults. As with all methods, if the argument is not specified, the method will run with the default values.

Start of the rains		
Argument	**Default**	**Description**
data_list	list()	A list containing stations for analysis, the years or periods to be analysed and the required variables from the data. If blank, the

		system will choose all data appropriate for the analysis.
earliest_day	122	The earliest possible day for the start of the rains.
dry_days	10	The length of the dry spell required to check the existence of within dry_length days.
dry_length	30	The number of days within which to check the existence of a dry spell not exceeding dry_days days.
total_days	2	The number of consecutive days to sum the corresponding rain observation to get the event.
Threshold	0.85	The value a day must be no less than to be considered as rainy.
dry_spell_condition	FALSE	Whether the dry spell condition is required to calculate the start of the rain.
rain_total	20	The amount of rain required over total_days days to determine the start of the rains.
col_name	"Start of the rains"	The column name to use for start of the rains.
Replace	FALSE	Whether the column should be replaced if there is already a column in the data with the value of col_name.

Table 2 List all the arguments with their defaults for start of the rain method

End of the rains

The end of the rains is among the most needed event by the farmers for their seasonal activities. Many definitions have been used to determine the end of the rain. We used the definition based on a simple water balance equation.

The definition in words is given as follows:

Water balance today = Water balance yesterday + Rainfall – Evaporation.

To calculate the end of the rains, we used the following definition:

"The first occasion after the end of rain date on which a simple water balance dropped to zero."

The following command calculates the end of rain:

siac_obj$add_end_rain()

An alternative calculation could be done by specifying arguments:

siac_obj$add_end_rain(earliest_day = 200, evaporation = 6.5, col_name = "End)")

Other arguments may be specified in this method for added flexibility. Below we list all the arguments, with their defaults. As with all methods, if the argument is not specified, the method will run with the default values.

End of the rains		
Argument	**Default**	**Description**
data_list	list()	A list containing stations for analysis, the years or periods to be analyzed and the required variables from the data. If blank, the system will choose all data appropriate for the analysis.
earliest_day	228	The earliest possible day for the end of the rains.
water_balance_col_name	"Water Balance"	The column name to use for the water balance column if it needs to be created.
col_name	"End of the rains"	The column name to use for the end of the rains.
capacity_max	100	The maximum water balance.

evaporation	5	Evaporation per day
Replace	FALSE	Whether the column should be replaced if there is already a column in the data with the value of col_name.

Table 3 List all the arguments with their defaults for end of the rains method

As with the start of the rains, we produce yearly data from daily data, so this should also be recorded in the yearly summary data frame. Just as with the start of the rains, an end of the rains column is appended to the yearly summary data frames, and if they do not exist already, they are created and stored in the used data objects list of the climate object.

Use the following command to view the yearly data summary:

View(siac_obj$used_data_objects[["samaru yearly"]]$data)

The first six rows are shown below:

	Year	Date	Total Rain	Average rain per rain day	Number of Rain Days	End of the rains
1	1928	1928-01-01	1262.34	14.77953	85	300
2	1929	1929-01-01	1284.18	13.62915	94	301
3	1930	1930-01-01	1044.73	12.36774	84	288
4	1931	1931-01-01	1197.82	15.69724	76	288
5	1932	1932-01-01	1198.16	15.72513	76	295
6	1933	1933-01-01	1311.61	14.20707	92	294

Seasonal Summary Rain

Seasonal_summary.rain method adds column(s) of seasonal summaries for rain (e.g rain totals, number of rain days, and longest dry spell) given climate object. Below we list all the arguments, with their defaults and descriptions. As with all methods, if the argument is not specified, the method will run with the default values.

Seasonal_summary.rain		
Argument	**Default**	**Description**

data_list	list()	A list containing stations for analysis, the years or periods to be analysed and the required variables from the data. If blank, the system will choose all data appropriate for the analysis.
variable_to_summarize	rain_label	A character of the label of the variable to be used for creating the summaries.
month_start	1	A vector of months (in character or integer format) indicating when the calculation of seasonal summary should be started. If not specified, the first month of the season will be used.
dry_length	TRUE	The number of days within which to check the existence of a dry spell not exceeding dry_days days.
number_month	3	An integer indicating the number of months to be considered in the calculation of seasonal summary. The default is 3.
Threshold	0.85	The value a day must be no less than to be considered as rainy.
summaries	list(sum_label, count_over_threshold_label, mean_over_threshold_label)	A list of summary function to be used on variable_to_summarize e.g Sum, Mean, max, median, mean_over_threshold
month_col_names	“ “	A vector of characters giving the name of each time period that will be generated from month_start and number_month. If not specified, a default format of "xxx-yyy" will be used were "xxx" is the three letter abbreviation of the first month and "yyy" is the three letter abbreviation of the last month.
summary_col_names	c("Total Rain", "Number of rainy days", "Mean rain per rainy day")	A vector of character giving a name o display for each summary of summaries. This will be appended to month_col_names to create the variable names for

		the summary columns. If not specified, the summary function names will be used.
na.rm	FALSE	A logical that will be passed to the summary functions indicating whether NA values should be ignored in the data for calculations.
longest_dry_spell	TRUE	A logical indicating whether a column of longest dry spell should be appended to the yearly summary object, if the interest variable is rain.
replace	FALSE	A logical indicating whether a column in the summary data frame should be replaced if a column with that name already exists.

Table 4 List all the arguments with their defaults for seasonal summary rain method

As with the end of the rains, we produce yearly data from daily data, so this should also be recorded in the yearly summary data frame. Just as with the end of the rains, the summary column of the created summary is appended to the yearly summary data frames, and if they do not exist already, they are created and stored in the used data objects list of the climate object.

We create the summary with the following command:

```
siac_obj$seasonal_summary.rain(summary_col_names = c("Tot_rain"),summaries =
list(sum_label) )
```

Use the following command to view the yearly data summary:

View(siac_obj$used_data_objects[[“moorings yearly”]]$data)

The first six rows are shown below:

Year	Date	Sum Rain	mean_over_threshold Rain	count_over_threshold Rain	Jan-Mar Tot_rain	Jan-Mar Longest dry spell
1921	1/1/1921	NA	NA	NA	NA	NA
1922	1/1/1922	NA	NA	NA	NA	NA
1923	1/1/1923	884.9	11.16538	78	666.2	3
1924	1/1/1924	662.4	11.55789	57	234.7	19
1925	1/1/1925	863.3	9.18172	93	715.7	6
1926	1/1/1926	1058.8	13.1075	80	753.2	7

Assume that the period we are interested on is between day 124 and 215 in moorings data starting from first July as the 1st day of the year or season (three months, starting in November).

The below command gives the results

siac_obj$seasonal_summary.rain(month_start = "Nov", number_month = 3, longest_dry_spell = FALSE,summaries = list(sum_label), summary_col_names = c("Tot_rain"))

The summarized data can be viewed by:

View(head(siac_obj$used_data_objects[["moorings yearly"]]$data))

	Year	Date	Sum Rain	mean_over_threshold Rain	count_over_threshold Rain	Nov-NA Tot_rain
1	1921	1921-01-01	NA	NA	NA	NA
2	1922	1922-01-01	NA	NA	NA	NA
3	1923	1923-01-01	884.9	11.16538	78	445.1
4	1924	1924-01-01	662.4	11.55789	57	455.8
5	1925	1925-01-01	863.3	9.18172	93	341.4
6	1926	1926-01-01	1058.8	13.10750	80	544.0

Extremes Events

The *extremes_event* adds a column of extreme events averaged over a number of days and the day when the event occurred (e.g maximum, minimum and extreme event day) given climate object.

It requires the following arguments with their defaults and description:

Extreme Events		
Argument	**Default**	**Description**
data_list	list()	A list containing stations for analysis, the years or periods to be analyzed and the required variables from the data. If blank, the system will choose all data appropriate for the

		analysis.
year	all years in the data.	A vector of numeric years from which to get the extreme.
required_var	rain_label	The variable name from which to get the extreme events.
na.rm	TRUE	A logical indicating whether missing values should be removed.
Threshold	0.85	The least amount of rainfall for which a day is considered rainy.
max_min	TRUE	A logical scalar. Should the maximum or the minimum be computed?
extreme	Max	Compute max or min
sum_day	1	Numeric argument. number of days to sum or average over.
val_threshold	FALSE	A logical a scalar. Should the Values over threshold be computed?
threshold_value	0	Numeric argument. A fixed limit over which Values over threshold are computed.
start_day	1	A numeric argument. It shows when in the season the computation started.
end_day	366	A numeric argument. It shows when in the season the computation ended.
values_between	FALSE	A logical scalar. Should the computation be done with values between a range?
lower_lim, upper_lim		Numeric values. A range of values over which the computation occurs.
col_name, col_name2	c("Yearly Maximum","extreme event day")	Character arguments. col_name for extreme events and col_name2 for the day the event occurred.

Replace	FALSE	Logical indicating whether the column should be replaced if there is already a column in the data with the value of col_name.

Table 5 List of all the arguments with their defaults for extreme events method

If no yearly summary data frames have been produced up to this point, they will automatically be created and then the start of the rains column will be appended to each of the yearly summary data frames. Here there is no visible output on the console but if we view the yearly summary data for one of the dataset in the analysis, we see that the extreme column has been appended:

The following command calculates the end of rain:

siac_obj$extreme_events(col_name = "extreme")

Use the following command to view the yearly data summary:

View(head(siac_obj$used_data_objects[[1]]$data))

The first six rows are shown below:

	Year	Date	sum Rain	mean_over_threshold Rain	count_over_threshold Rain	extreme
1	1928	1928-01-01	1262.34	14.77953	85	77.72
2	1929	1929-01-01	1284.18	13.62915	94	54.10
3	1930	1930-01-01	1044.73	12.36774	84	92.96
4	1931	1931-01-01	1197.82	15.69724	76	64.26
5	1932	1932-01-01	1198.16	15.72513	76	64.52
6	1933	1933-01-01	1311.61	14.20707	92	95.25

Date in the format day and month

The *day_month* method adds the date column on the actual data in the format day + month, given the date. i.e It changes the format of date so that the date appear in the format day + month (i.e. "17 Apr" rather than "108") given the date column. It takes a specific date of the year and creates a column of the date in the format day-month.

It requires the following arguments:

Date in Format Day and Month		
Argument	**Default**	**Description**
data_list	list()	A list containing stations for analysis, the years or periods to be analyzed and the required variables from the data. If blank, the system will choose all data appropriate for the analysis.
Time_period	daily_label	Data time period.
Col_name	"Day_Month"	The name of the new column created.
Month_format	"%m"	The format in which the months are stored.
Required_format	"%d-%b"	The required format of the day + month.

Table 6 List of all the arguments with their defaults for day_month method

This method allows us to add a new date column in the format day plus month. The computation here is not visible but we view the climate data frame. The new column of required date format is added to the original data set.

siac_obj$day_month(data_list = list("samsmall"))

View climate data and we will see a data frame containing a new added column in the format day and month.

```
View(siac_obj$climate_data_objects[["samsmall"]]$data)
```

The above command gives the following results: the provided output is the head of data frame for samsmall (the climate object in the analysis).

```
head(siac_obj$climate_data_objects[["samsmall"]]$data)
```

```
  Year Month Day Rain       Date Day_Month
1 1930     1   1    0 1930-01-01    01-Jan
2 1930     1   2    0 1930-01-02    02-Jan
3 1930     1   3    0 1930-01-03    03-Jan
4 1930     1   4    0 1930-01-04    04-Jan
5 1930     1   5    0 1930-01-05    05-Jan
6 1930     1   6    0 1930-01-06    06-Jan
```

Adding a column on the data set like year, month or day

During data analysis the user might need to add a column in R. For instance, he/she can add year column by using **add_year_month_day_cols** method.

Add Year, Month and Day columns		
Argument	**Default**	**Description**
data_list	list()	A list containing stations for analysis, the years or periods to be analyzed and the required variables from the data. If blank, the system will choose all data appropriate for the analysis.
update	FALSE	A logical determining whether existing columns are replaced.
YearLabel	"Year"	Variable label for Year column
MonthLabel	"Month"	Variable label for Month column
DayLabel	"Day"	Variable label for Day column

The following command will add year, month and day column to the data if not present:

siac_obj$add_year_month_day_cols()

We can view the actual data frame as shown below, for one of the original data sets, to see if the year, month and day columns have been appended:

View(head(siac_obj$climate_data_objects[["samaru"]]$data))

	Year	DOY	Rain	Date	Month	Day
1	1928	1	0	1928-01-01	1	1
2	1928	2	0	1928-01-02	1	2
3	1928	3	0	1928-01-03	1	3

	Year	DOY	Rain	Date	Month	Day
4	1928	4	0	1928-01-04	1	4
5	1928	5	0	1928-01-05	1	5
6	1928	6	0	1928-01-06	1	6

Water balance

The previous method relies on the water balance values in the daily data. If the daily data does not contain a water balance column, it will call the climate data method *add_water_balance_col*, for each data frame to calculate and add the water balance to the daily data.

The water balance tables can also be viewed with the following command:

siac_obj$display_water_balance()

Alternatively:

water_balance_tables<- siac_obj$display_water_balance()

stored the tables in lists in the variable water_balance_tables.

The table below shows the arguments for the method with their defaults and description.

Water balance		
Argument	**Default**	**Description**
data_list	list()	A list containing stations for analysis, the years or periods to be analyzed and the required variables from the data. If blank, the system will choose all data appropriate for the analysis.
print_tables	TRUE	Whether the tables should be printed, or just stored in a variable.
col_name	"Water	The column name to use for the water balance column, if it

	Balance"	needs to be created.
capacity_max	100	The maximum water balance to use, if the water balance column does not already exist.
evaporation	5	Evaporation per day to use, if the water balance column does not already exist.
decimal_places	0	The number of decimal places to use in the water balance tables
months_list	month.abb	The list of characters to use for the months in the water balance tables.
day_display	"Day"	The character to use for the day column in the water balance tables.

Table 7 List of all the arguments with their defaults for display water balance method

The resulting water balance table of a single year for one station is as follows:

```
$`1982`
   Day Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
1    1   0   0   0   0   2  12  12  29  77  38   0   0
2    2   0   0   0   0   0   7   7  24  77  33   0   0
3    3   0   0   0   0   0   2   2  19  72  28   0   0
4    4   0   0   0   0   0   0   0  14  67  23   0   0
5    5   0   0   0   0   1   1   0  14  91  18   0   0
6    6   0   0   0   0   0   0  20  18  87  13   0   0
7    7   0   0   0   0   0   0  15  13  87   8   0   0
8    8   0   0   0   0   0   0  12   9  82   3   0   0
9    9   0   0   0   0   0   0  28  23  97  38   0   0
10  10   0   0   0   0   6   0  23  18  95  33   0   0
11  11   0   0   0   0   0  11  37  16  99  28   0   0
12  12   0   0   0   0   0   6  32  11  94  23   0   0
13  13   0   0   0   0   0   1  27  11  97  18   0   0
14  14   0   0   0   0   0   0  67  13  93  13   0   0
15  15   0   0   0   0   0   0  62   8  88   8   0   0
16  16   0   0   0   8   0  23  70   5  83   3   0   0
17  17   0   0   0   3   0  18  65  28  78   0   0   0
18  18   0   0   0   1   0  13  60  23  73   0   0   0
19  19   0   0   0   0   0   8  55  18  68   0   0   0
20  20   0   0   0   0  12  28  50  13  78   0   0   0
21  21   0   0   0   0   7  24  45  16  73   0   0   0
22  22   0   0   0   0  13  18  40  63  77   0   0   0
23  23   0   0   0   0   8  14  35  65  72   0   0   0
24  24   0   0   0   0   3  18  30  67  67   0   0   0
25  25   0   0   0   0   0  17  25  65  62   0   0   0
26  26   0   0   0   1   0  12  57  63  57   0   0   0
```

27	27	0	0	0	4	0	7	52	58	52	0	0	0
28	28	0	0	0	0	0	25	47	58	47	0	0	0
29	29	0	NA	0	12	0	20	42	79	42	0	0	0
30	30	0	NA	0	7	0	15	39	74	37	0	0	0
31	31	0	NA	0	NA	17	NA	34	69	NA	0	NA	0

Display the Spell Length Table

The obvious definition for a dry day is any day with zero rainfall. We first define a dry day and choose to define it as a day with rain less than 0.85 mm to avoid any rounding problem which might occur in recording of small amount of rainfall.

The *display_spell_length* returns spell length tables. It relies on the spell length values in the daily data. If the daily data does not contain a spell length column, it will call the climate data method *add_spell_length_col*, for each data frame to calculate and add the spell length to the daily data.

The spell length tables can also be viewed with the following command:

siac_obj$ display_spell_length ()

It requires the following arguments with their defaults and description:

Display Spell Length Table		
Argument	**Default**	**Description**
data_list	list()	A list containing stations for analysis, the years or periods to be analyzed and the required variables from the data. If blank, the system will choose all data appropriate for the analysis.
col_name.	"spell length"	The name of the spell length column.
months_list	month.abb	The names of the months.
day_display	"Day"	The name of the first column in the table.

na.rm	TRUE	A logical indicating whether missing values should be removed.
Threshold	0.85	The least amount of rainfall for which a day is considered rainy.

Table 8 List of all the arguments with their defaults for display spell length method

The table below is a resulting spell length table of a single year for one of the station in the analysis.

```
$`1983`
       Day Jan Feb Mar Apr May Jun Jul Aug Sep Oct       Nov  Dec
1        1  84 115 143 174 204   1   2  --  --   6        37   67
2        2  85 116 144 175 205  --  --  --   1   7        38   68
3        3  86 117 145 176 206  --   m  --   2   8        39   69
4        4  87 118 146 177 207   1  --  --  --   9        40   70
5        5  88 119 147 178 208   2  --   1  --  10        41   71
6        6  89 120 148 179 209   3   1   2  --  11        42   72
7        7  90 121 149 180 210   4  --  --   1  12        43   73
8        8  91 122 150 181 211   5   1   1  --  13        44   74
9        9  92 123 151 182   m   6   2   2   1  14        45   75
10      10  93 124 152 183   m   m  --  --   2  15        46   76
11      11  94 125 153 184  --  --   1   1  --  16        47   77
12      12  95 126 154 185  --   1   2   2   1  17        48   78
13      13  96 127 155 186   1   2   3   3  --  18        49   79
14      14  97 128 156 187  --   3   4   4   1  19        50   80
15      15  98 129 157 188   1  --   5   5  --  20        51   81
16      16  99 130 158 189   2   1  --   6  --  21        52   82
17      17 100 131 159 190   3   2   1   7   1  22        53   83
18      18 101 132 160 191   4   m   2  --  --  23        54   84
19      19 102 133 161 192   m   m   3  --  --  24        55   85
20      20 103 134 162 193  --   m  --  --   1  25        56   86
21      21 104 135 163 194   1   m   1   1   2  26        57   87
22      22 105 136 164 195   2   m  --   2   3  27        58   88
23      23 106 137 165 196  --   m   1  --   m  28        59   89
24      24 107 138 166 197  --  --   2   1   m  29        60   90
25      25 108 139 167 198   1  --   3  --  --  30        61   91
26      26 109 140 168 199  --   1   m  --   1  31        62   92
27      27 110 141 169 200   1   2   m   1   2  32        63   93
28      28 111 142 170 201  --  --  --  --   3  33        64   94
29      29 112     171 202   1  --  --   1   4  34        65   95
30      30 113     172 203   2   1   1   2   5  35        66   96
31      31 114     173      --      2   3      36                 97
32 Maximum                                                (Overall: 211)
33         114 142 173 203 211   6   5   7   5  36        66   97
34 Missing                                                (Overall:  15)
35           0   0   0   0   3   7   3   0   2   0         0    0
```

Display Day of the Year

The *display_doy* is a climate method which displays the column storing the day of the year values in a tabular format. The day of the year column is organized in a table with 12 columns

each of the three letter abbreviations for the English names of the month and 31 rows indicating the days of the month. [Rewrite the second sentence]

The display_doy has the following arguments with their defaults:

Display Day of the Year		
Argument	**Default**	**Description**
data_list	list()	A list containing stations for analysis, the years or periods to be analyzed and the required variables from the data. If blank, the system will choose all data appropriate for the analysis.
row.names	FALSE	A logical, if *false*, the row names attributed to the data frame are removed.
months_list	month.abb	The names of the months.
day_display	"Day"	The name of the first column in the table.
File	"DOY.doc"	The path of the output file.
Width	8.5	The width of the output page.
Font_size	6	Default font size for the document in the points.
Na.string	" "	Character string of missing value to be displayed in the table.
Save_table	FALSE	A logical, if *true*, the output file is saved.

Table 9 List of all the arguments with their defaults for display day of the year method

The day of the year table can be viewed if necessary on the R console with the following command:

siac_obj$display_doy(data_list = list("samaru56"))

```
 Day Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
   1   1  32  61  92 122 153 183 214 245 275 306 336
   2   2  33  62  93 123 154 184 215 246 276 307 337
   3   3  34  63  94 124 155 185 216 247 277 308 338
   4   4  35  64  95 125 156 186 217 248 278 309 339
   5   5  36  65  96 126 157 187 218 249 279 310 340
   6   6  37  66  97 127 158 188 219 250 280 311 341
   7   7  38  67  98 128 159 189 220 251 281 312 342
```

```
   8    8  39  68  99 129 160 190 221 252 282 313 343
   9    9  40  69 100 130 161 191 222 253 283 314 344
  10   10  41  70 101 131 162 192 223 254 284 315 345
  11   11  42  71 102 132 163 193 224 255 285 316 346
  12   12  43  72 103 133 164 194 225 256 286 317 347
  13   13  44  73 104 134 165 195 226 257 287 318 348
  14   14  45  74 105 135 166 196 227 258 288 319 349
  15   15  46  75 106 136 167 197 228 259 289 320 350
  16   16  47  76 107 137 168 198 229 260 290 321 351
  17   17  48  77 108 138 169 199 230 261 291 322 352
  18   18  49  78 109 139 170 200 231 262 292 323 353
  19   19  50  79 110 140 171 201 232 263 293 324 354
  20   20  51  80 111 141 172 202 233 264 294 325 355
  21   21  52  81 112 142 173 203 234 265 295 326 356
  22   22  53  82 113 143 174 204 235 266 296 327 357
  23   23  54  83 114 144 175 205 236 267 297 328 358
  24   24  55  84 115 145 176 206 237 268 298 329 359
  25   25  56  85 116 146 177 207 238 269 299 330 360
  26   26  57  86 117 147 178 208 239 270 300 331 361
  27   27  58  87 118 148 179 209 240 271 301 332 362
  28   28  59  88 119 149 180 210 241 272 302 333 363
  29   29  60  89 120 150 181 211 242 273 303 334 364
  30   30  NA  90 121 151 182 212 243 274 304 335 365
  31   31  NA  91  NA 152  NA 213 244  NA 305  NA 366
```

To save day of the year table, we must specify in the arguments that the save table is also required. The DOY table will be saved in your current working directory.

siac_obj$display_doy(data_list = list("samaru56"), save_table = TRUE)

DOY table

Day	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
1	1	32	61	92	122	153	183	214	245	275	306	336
2	2	33	62	93	123	154	184	215	246	276	307	337
3	3	34	63	94	124	155	185	216	247	277	308	338
4	4	35	64	95	125	156	186	217	248	278	309	339
5	5	36	65	96	126	157	187	218	249	279	310	340
6	6	37	66	97	127	158	188	219	250	280	311	341
7	7	38	67	98	128	159	189	220	251	281	312	342
8	8	39	68	99	129	160	190	221	252	282	313	343
9	9	40	69	100	130	161	191	222	253	283	314	344
10	10	41	70	101	131	162	192	223	254	284	315	345
11	11	42	71	102	132	163	193	224	255	285	316	346
12	12	43	72	103	133	164	194	225	256	286	317	347
13	13	44	73	104	134	165	195	226	257	287	318	348
14	14	45	74	105	135	166	196	227	258	288	319	349
15	15	46	75	106	136	167	197	228	259	289	320	350
16	16	47	76	107	137	168	198	229	260	290	321	351
17	17	48	77	108	138	169	199	230	261	291	322	352
18	18	49	78	109	139	170	200	231	262	292	323	353
19	19	50	79	110	140	171	201	232	263	293	324	354
20	20	51	80	111	141	172	202	233	264	294	325	355
21	21	52	81	112	142	173	203	234	265	295	326	356

Day	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
22	22	53	82	113	143	174	204	235	266	296	327	357
23	23	54	83	114	144	175	205	236	267	297	328	358
24	24	55	84	115	145	176	206	237	268	298	329	359
25	25	56	85	116	146	177	207	238	269	299	330	360
26	26	57	86	117	147	178	208	239	270	300	331	361
27	27	58	87	118	148	179	209	240	271	301	332	362
28	28	59	88	119	149	180	210	241	272	302	333	363
29	29	60	89	120	150	181	211	242	273	303	334	364
30	30		90	121	151	182	212	243	274	304	335	365
31	31		91		152		213	244		305		366

Display daily data

The raw climatic daily rainfall data values can be examined in a table. This is done in display daily data method in a tabular format.

The display_daily method has the following arguments with their defaults:

Display daily data		
Argument	**Default**	**Description**
data_list	list()	A list containing stations for analysis, the years or periods to be analyzed and the required variables from the data. If blank, the system will choose all data appropriate for the analysis.
row.names	FALSE	A logical, if *false*, the row names attributed to the data frame are removed.
months_list	month.abb	The names of the months.
day_display	"Day"	The name of the first column in the table.
Print_tables	FALSE	A logical, if *true*, the method print the table in the console

Variable	rain_label	The variable to be displayed.
Threshold	0.85	The least amount of rainfall for which a day is considered rainy.
Na.rm	TRUE	A logical indicating whether missing values should be removed.

Table 10 List of all the arguments with their defaults for display daily data method

The resulting daily data table of a single year for one station is as follows:

siac_obj$display_daily()

Siac_obj$display_daily(summary_names = c("Tot", "Max", "Val>0.85"),decimal_places = 0)

```
$samaru56$`1983`
   Day Jan  Feb Mar  Apr  May  Jun  Jul Aug  Sep Oct  Nov Dec
1    1   0    0   0    0    0    0    0   2    2   0    0   0
2    2   0    0   0    0    0   10   11  35    0   0    0   0
3    3   0    0   0    0    0    4 <NA>  14    0   0    0   0
4    4   0    0   0    0    0    0   10   1    1   0    0   0
5    5   0    0   0    0    0    0    7   1   21   0    0   0
6    6   0    0   0    0    0    0    0   0   11   0    0   0
7    7   0    0   0    0    0    0    1  27    0   0    0   0
8    8   0    0   0    0    0    0    0   0   16   0    0   0
9    9   0    0   0    0 <NA>    0    0   0    0   0    0   0
10  10   0    0   0    0    0 <NA>   18  18    0   0    0   0
11  11   0    0   0    0    4    7    1   0    3   0    0   0
12  12   0    0   0    0   16    0    0   0    0   0    0   0
13  13   0    0   0    0    0    0    0   0    7   0    0   0
14  14   0    0   0    0   17    0    0   0    0   0    0   0
15  15   0    0   0    0    0    7    0   0   12   0    0   0
16  16   0    0   0    0    0    0    6   0   10   0    0   0
17  17   0    0   0    0    0    0    0   0    0   0    0   0
18  18   0    0   0    0    0 <NA>    0  48    5   0    0   0
19  19   0    0   0    0 <NA>    0    0  21    2   0    0   0
20  20   0    0   0    0   17    0    1  15    0   0    0   0
21  21   0    0   0    0    0 <NA>    0   0    0   0    0   0
22  22   0    0   0    0    0    0   18   0    0   0    0   0
23  23   0    0   0    0    2    0    0   6 <NA>   0    0   0
24  24   0    0   0    0    3   24    0   0 <NA>   0    0   0
25  25   0    0   0    0    0    1    0  26    4   0    0   0
26  26   0    0   0    0    1    0 <NA>  25    0   0    0   0
27  27   0    0   0    0    0    0    0   0    0   0    0   0
28  28   0    0   0    0    9   17   27  19    0   0    0   0
29  29   0 <NA>   0    0    0    4    6   0    0   0    0   0
30  30   0 <NA>   0    0    0    0    0   0    0   0    0   0
31  31   0 <NA>   0 <NA>    4 <NA>    0   0 <NA>   0 <NA>   0
32 Tot   0   NA   0   NA   NA   NA   NA 258   NA   0   NA   0
```

```
33       Max   0   NA   0   NA   NA   NA   NA  48   NA   0   NA   0
34 Val>0.85   0   NA   0   NA   NA   NA   NA  14   NA   0   NA   0
```

Compare graphs

The *plot_yearly_comparison* plots a graph of one or more variables on the same plot. Plotting two or more graphs in the same plot gives an idea of comparing the variables when the variables of interest are evaluated at the same x-coordinates. This can be very useful when comparing different variables. For instance, start of the rain for various definitions.

The *plot_yearly_comparison* has the following arguments with their defaults:

Plot Yearly Comparison		
Argument	**Default**	**Description**
data_list	list()	A list containing stations for analysis, the years or periods to be analyzed and the required variables from the data. If blank, the system will choose all data appropriate for the analysis.
Variables		This contains the list of variable to be plotted.
Col	c("blue", "red", "green")	The color of the points or lines appearing in the legend.
Lty	c(1,2,3)	The line types for lines appearing in the legend.
Type	c("h", "h", "h")	Character indicating the type of plotting.

Ylabel	"Observations"	The label on y axis.
Xlabel	"Year"	The label on x axis.
Main	"Vertical Lines"	Character string or expression giving a title of the plot.
Time_period	yearly_label	Data time period.
Legend.location	rep(list("topright"), length(variables))	The location of the legend on the graph.
Legend	rep(list(variables), length(variables))	Character to appear in the legend.
Legend_text_width	strwidth("0.001")	The width of the character of the legend.
na.rm	TRUE	A logical indicating whether missing values should be removed.

Table 11 List of all the arguments with their defaults for plot yearly comparison method

Here we use samrain data to demonstrate the example in siac object. The line graph is produced with just a single command by specifying the type argument as line type, ylabel, legend and legend's location.

siac_obj$plot_yearly_comparison(data_list = list("samrain"),variables = c("Strt1", "Strt2"), type = c("l", "l"),ylabel = "Start of the rain", legend=rep(list(variables), length(variables)),legend.location = rep(list("topleft")), main = "Start of the rain")

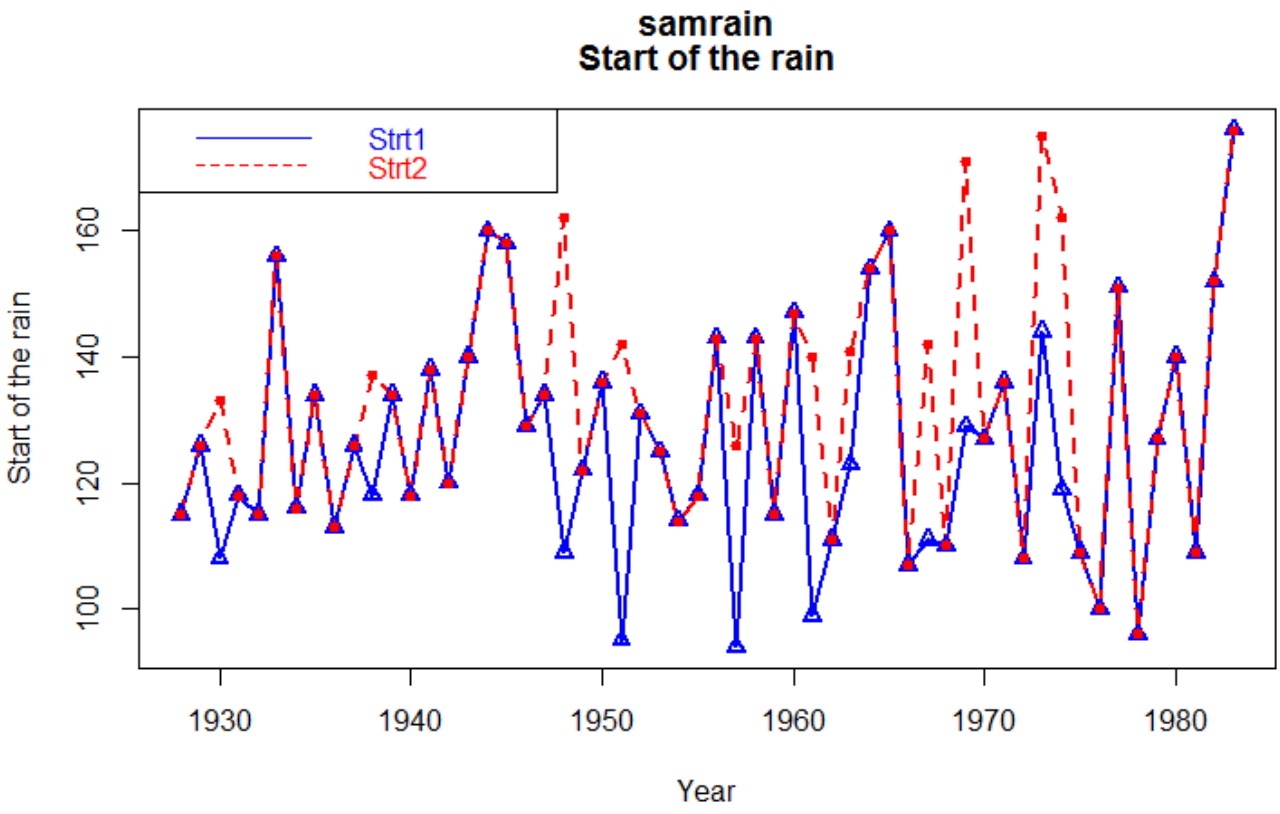

Figure 1 Comparison of two definitions of start of the rains in graph

When changing the line type, we can get vertical lines (example: **type = c("h", "h"))** as follows:

siac_obj$plot_yearly_comparison(data_list = list("samrain"),variables = c("Strt1", "Strt2"), type = c("h", "h"),ylabel = "Start of the rain", legend=rep(list(variables), length(variables)),legend.location = rep(list("topleft")), main = "Start of the rain")

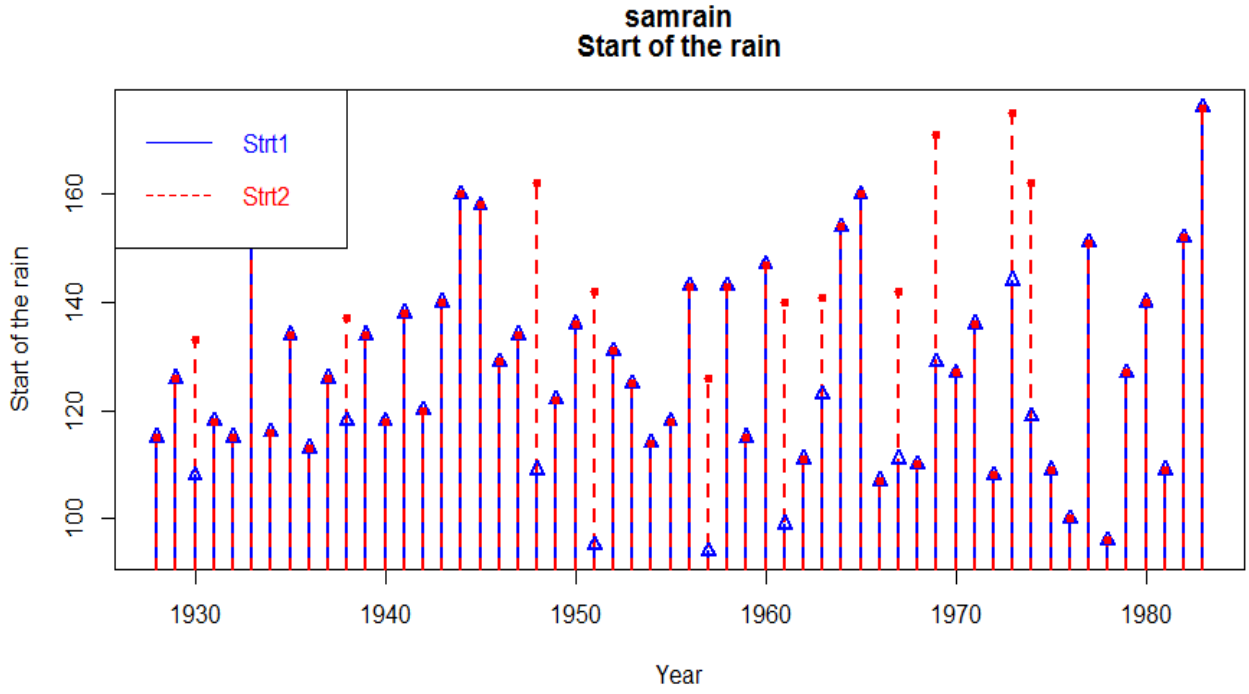

Figure 2 Comparison of two variables using vertical lines

Cumulative and exceedance graph

The *cumulative_exceedance_graphs* produces the cumulative or exceedance graphs given a variable of interest. It has the following arguments:

Cumulative and Exceedance Graphs		
Argument	**Default**	**Description**
data_list	list()	A list containing stations for analysis, the years or periods to be analysed and the required variables from the data. If blank, the system will choose all data appropriate for the analysis.
interest_var	"Total Rain"	The variable of interest to be plotted.
cumulative_graph	TRUE	A logical indicating whether cumulative graph should be plotted. Cumulative graph is produced when it is TRUE.

		Otherwise an exceedance graph is produced.
legend_bty	“n”	The type of box to be drawn around the legend.
Percent	TRUE	A logical indicating whether the calculated values should be in percent or not.
Color	rainbow(12)	The legend color.
Main	” ”	Character string or expression giving a title of the plot.
Xlabel	“ ”	The X axis label.
Ylabel	" "	The Y axis label.
line_type	"o"	The type of plot to be drawn.
legend_position	“center”	The position of the legend.
legend_label	c()	The label of the legend. This is needed when interest_var>1
station_name	TRUE	A logical indicating whether the name of the station should be included in the title of the plot.
Plot_window	TRUE	A logical indicating whether the plots per station should be on the same plot window.
Gpar	par(mfrow=c(2,4))	Used to arrange figures in n rows and m columns when plot_window is TRUE.
grid_sq	TRUE	A logical indicating whether a grid should be included in the plot.
grid_sq	TRUE	A logical indicating whether an nx by ny rectangular grid should be added to an existing plot.

nx,ny	nx=ny=5	Dimension of the rectangular grid.
Lwd	2	The width of the grid lines
box.lty	par("lty")	The grid line type.

Table 12 List of all the arguments with their defaults for cumulative and exceedance graph method

We can use samrain data to demonstrate how to produce cumulative distribution and exceedance graphs as follows:

Here we use samrain data which is yearly summaries for start of the rain.

Plotting the cumulative graph

The cumulative graph is therefore given by:

siac_obj$cumulative_exceedance_graphs (data_list = list(“samrain”), interest_var = list("Strt1","Strt2"),cumulative_graph =TRUE, main="Start of the rain for Samaru data 1928 - 1983", xlabel="Length in days", ylabel="Percent of days", station_name=FALSE,color=c("blue", "red"), legend_label=c("Strt1","Strt2"), legend_position="topleft")

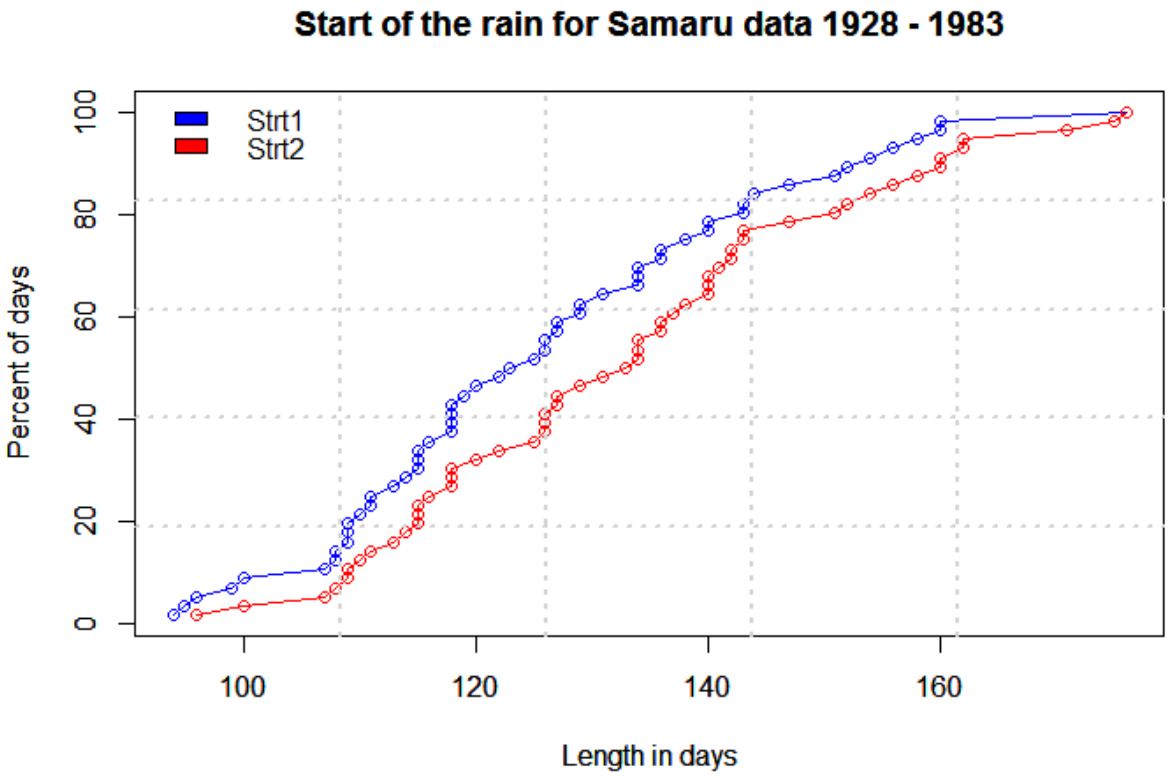

Figure 3 Cumulative graphs

To produce the exceedance graph the cumulative_graphs argument is set to be FALSE. Here is an example:

Siac_obj $cumulative_exceedance_graphs (data_list = list("samrain"), interest_var =list("Strt1","Strt2"),cumulative_graph =FALSE,main="Start of the rain for Samaru data 1928 - 1983", xlabel="Length in days",ylabel="Percent of days", station_name=FALSE,color=c("blue", "red"), legend_label=c("Strt1","Strt2"), legend_position="topright")

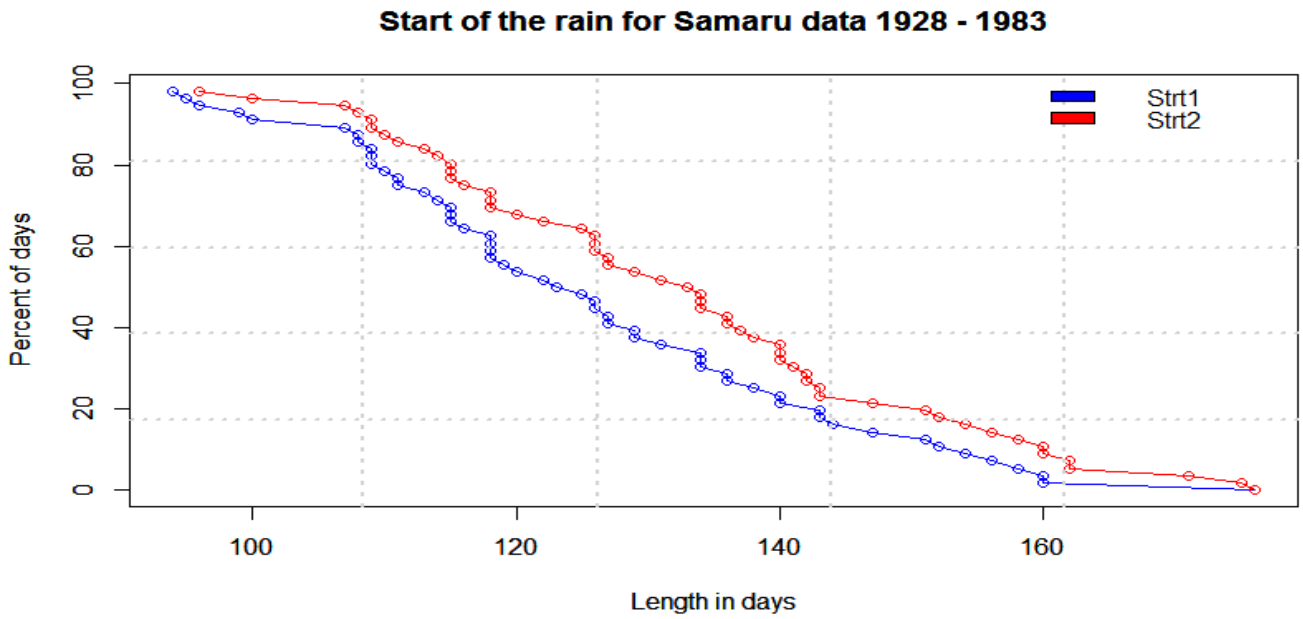

Figure 4 Exceedance graph

Box-plot with jitter

The *box_jitter* produces a boxplot with actual data points to the box given a grouped data of variables. To avoid overlap of the data points with the same value, we "jitter" the data points.

We list all the arguments, with their defaults below. If the argument is not specified, the method will run with the function default.

Boxplot with data points		
Argument	**Default**	**Description**
data_list	list()	A list containing stations for analysis, the years or periods to be analyzed and the required variables from the data. If blank, the system will choose all data appropriate for the analysis.
Var		This contains the variables list of interest in the

		plot.
Names	c()	Character string for the variables plotted.
Method	"jitter"	The method to be used to separate coincident points.
Jitter	0.1	This gives the mount of jittering applied.
Time_period	yearly_label	Data time period.
Horizontal	FALSE	A logical, If *true*, the boxplot is plotted horizontally.
Plot_jitter	FALSE	A logical, If *true*, the data points are added to the boxplot.
Ylab	"Day number for planting"	y axis label.
add	TRUE	A logical, if *true*, add the chart to the current
Colpoints	"red"	The color of the data points.
Na.rm	TRUE	A logical indicating whether missing values should be removed.
Connect.median	FALSE	A logical, if *true*, the medians of the box are connected.
Plot.sd	FALSE	A logical, if *true*, the line indicating the standard deviation is added.
Lty	1	The line type.
Main	c(data_name,"Start of the rain")	Character string or expression giving a title of the plot.
Varwidth	FALSE	A logical, if TRUE, the boxes are drawn with widths proportional to the square-roots of the number of observations in the groups.
Outline	TRUE	A logical, if *false*, the outliers are not drawn.
Show.names	TRUE	A logical, if *true*, adds group labels which will be

		printed under each boxplot.

Table 13 List of all the arguments with their defaults for boxplot with data point method

The following is the example of how **box_jitter** method is used:

siac_obj$box_jitter(data_list = list("samrain"), var = c("Strt1","Strt2","Strt3","Strt4"), names = c("Strt1","Strt2","Strt3","Strt4"))

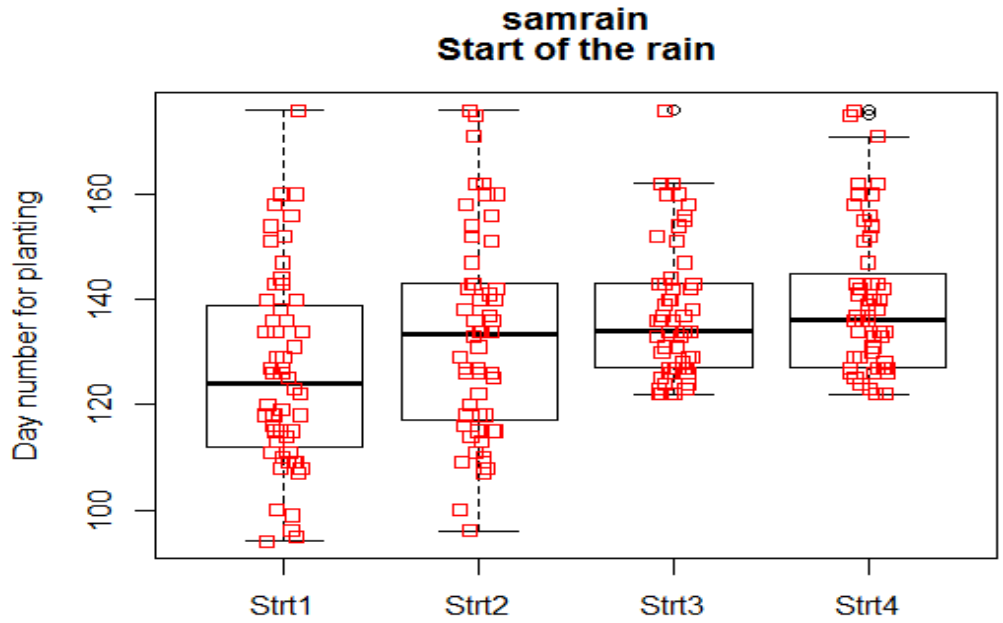

Figure 5 Boxplot with data points

Plot yearly summaries

The *plot_yearly_summary* produces the annual summaries plot given a variable of interest. It requires the following arguments:

Yearly Summaries Plot		
Argument	**Default**	**Description**

data_list	list()	A list containing stations for analysis, the years or periods to be analyzed and the required variables from the data. If blank, the system will choose all data appropriate for the analysis.
col1	"blue"	The color for the graph
Xlab, ylab	"Year", …	Title for the x axis and y axis respectively.
na.rm	TRUE	A logical indicating whether missing values should be removed.
Pch	20	A vector of plotting characters
Ylim	0	The lowest limit for y axis
Type	"b"	The type of plot to be drawn
Lty	2	The line type
col2	“red”	The color for regression line
lwd, lwd2	2, 1.5	The line width
interest_var	"Total Rain"	This indicates the variable to be plotted
var_label	rain_label	The variable to be summarized.
ygrid	0	Where the rectangular grid starts
graph_parameter	par(mfrow=c(2,2))	Sets graphical parameter.
plot_window	FALSE	A logical, if true set up the coordinate system for a graphics window.
main_title	“Plot - Summary per Year"	Character string or expression giving a title of the plot.
Grid	FALSE	A logical, if true adds an nx by ny rectangular grid to an existing plot.

Table 14 List of all the arguments with their defaults for plot yearly summaries method

The plot yearly summaries method can be used as follows:

We first read and create climate object on the data set:

We then run the following command given the variable of interest to get the plot.

siac_obj$plot_yearly_summary (interest_var = "rain number_of 1", graph_parameter = par(mfrow=c(2,2)), main_title="Number of rain days")

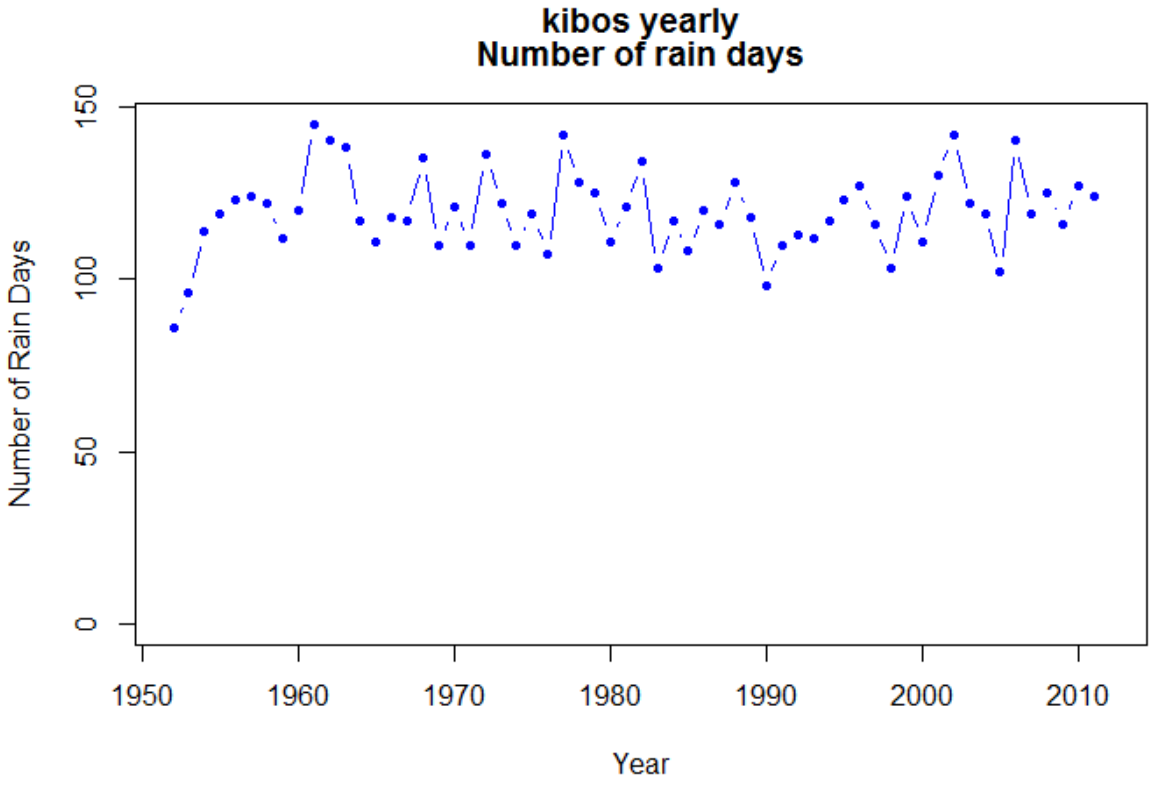

Figure 6 Number of raindays per year