QUICS: Quantifying Uncertainty in Integrated Catchment Studies

Deliverable D2.3. Software tools for aggregation & propagation uncertain input variables (ESR3)

Lead Partner: LIST

Revision: 22/11/16

Report Details

Title: Software tools for aggregation & propagation uncertain input variables (ESR3)

Deliverable Number: D2.3.

Author(s): J. A. Torres-Matallana; U. Leopold; G.B.M. Heuvelink

Dissemination Level: scientific and practitioners communities. Public in general.

Document History

Version	Date	Status	Submitted by	Checked by	Comment
1.0	25/10/2016	Draft First	J.A. Torres-Matallana	Simon Tait; Will Shepherd	
1.1	22/11/2016	Final	J.A. Torres-Matallana	Lutz Breuer	

Acronyms and Abbreviations

BCOD	Chemical oxygen demand load in the overflow
BNH4	Ammonium load in the overflow
COD	Chemical oxygen demand
CCOD	Chemical oxygen demand concentration in the overflow
CODr	Chemical oxygen demand rainwater pollution
CODs	Sewage chemical oxygen demand pollution
CSO	Combined sewer overflow
CNH4	Ammonium concentration in the overflow
NH4	Ammonium
NH4s	Sewage ammonium pollution
VTank	Volume in the combined sewer overflow tank
Vov	Overflow volume

Acknowledgements

This project has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement no 607000.

Executive Summary

The partners involved in the deliverable D2.3. are the Luxembourg Institute of Science and Technology (LIST), and the Wageningen University (WU). This deliverable is a result of the research project of the fellow ESR3.

The main objectives of the deliverable are to developed uncertainty propagation tools for lumped urban drainage models, and temporal aggregation tools for precipitation and pollutants of urban drainage models.

As a first tool, the R-package EmiStatR: Estimation of Wastewater Emissions in Combined Sewer Systems, has been developed. This R-package provides a fast and parallelised calculator to estimate combined wastewater emissions. Also, it supports the planning and design of urban drainage systems, without the requirement of extensive simulation tools.

A second tool was developed, the R-package AggProp. It provides several R methods and functions for temporal aggregation of environmental variables as precipitation and pollutants of UDMs. Also, it provides methods and functions for uncertainty propagation for lumped Urban Drainage Models (UDMs) via Monte Carlo simulation.

We illustrated the class *input* and the main method *EmiStatR* of the homonymous R-package through a brief user's guide. This guide also illustrates the class setup and the methods and functions *MC.setup*, *MC.sim* and *MC.analysis* of the R-package AggProp.

We hope that these tools constitute an important advancement in the state-of-art in uncertainty quantification in urban drainage modelling. Also, we presented how feasible their implementation and use is for both the scientific and the practitioners communities.

CONTENTS

1 Introduction

1.1 Partners Involved in Deliverable

The partners involved in the deliverable are the Luxembourg Institute of Science and Technology (LIST) and the Wageningen University (WU). This deliverable is a result of the research project of the fellow ESR3.

1.2 Deliverable Objectives

1) Uncertainty propagation tools (scripts) for lumped urban drainage models, currently implemented in R, such as EmiStatR.
2) Temporal aggregation tools (scripts) for precipitation and pollutants of urban drainage models.
3) Specific analysis tool, EmiStatR, for urban drainage system simulation to evaluate water quantity and quality.

The tool EmiStatR was implemented as an R package. This R package has been used in conjunction with tools 1) and 2) above in an uncertainty propagation analysis.

2 Tool one: EmiStatR 1.2.0.

An R-package has been developed called EmiStatR: Estimation of Wastewater Emissions in Combined Sewer Systems. The EmiStatR provides a fast and parallelised calculator to estimate combined wastewater emissions. It supports the planning and design of urban drainage systems, without the requirement of extensive simulation tools. The EmiStatR package implements modular R methods. This enables to add new functionalities through the R framework. Furthermore, EmiStatR was implemented with an interactive user interface with sliders and input data exploration. In Annex A we present the users' manual of the EmiStatR package.

3 Tool two: AggProp 1.0

A second R-package has been developed. The Aggregation and uncertainty Propagation R-package AggProp, provides several R methods and functions for temporal aggregation of environmental variables as precipitation and pollutants of UDMs. Also, it provides methods and functions for uncertainty propagation for lumped Urban Drainage Models (UDMs) via Monte Carlo simulation. This package, moreover, provides specific analysis functions for urban drainage system simulation to evaluate water quantity and quality in combined sewer overflows (CSOs). In Annex B we present the users' manual of the AggProp package.

4 Users' guide

A users' guide is provided. It illustrates the class input and the main method EmiStatR of the homonymous R-package. Also, the class setup and the methods and functions MC.setup, MC.sim and MC.analysis of the R-package AggProp are illustrated. In Annex C we present the users' guide of the EmiStatR and AggProp packages.

5 Conclusions

1) We have developed and presented the R-package, EmiStatR: Estimation of Wastewater Emissions in Combined Sewer Systems. This R-package provides a fast and parallelised calculator to estimate combined wastewater emissions. Also, it supports the planning and design of urban drainage systems, without the requirement of extensive simulation tools.
2) We have developed and presented the R-package AggProp. It provides several R methods and functions for temporal aggregation of environmental variables as precipitation and pollutants of Urban Drainage Models (UDMs). Also, it provides methods and functions for uncertainty propagation for lumped UDMs via Monte Carlo simulation.
3) We illustrated the class *input* and the main method *EmiStatR* of the homonymous R-package through a users' guide. This guide also illustrates the class setup and the methods and functions *MC.setup*, *MC.sim* and *MC.analysis* of the R-package AggProp.
4) We hope that these tools constitute an important advancement in the state-of-art in uncertainty quantification in urban drainage modelling. Also, we presented how feasible their implementation and use is for both the scientific and the practitioners communities.

Annex A

Package 'EmiStatR'

July 22, 2016

Type Package

Title Estimation of Wastewater Emissions in Combined Sewer Systems

Version 1.2.0

Date 2016-07-22

Author J.A. Torres-Matallana [aut, cre]
K. Klepiszewski [aut, cre]
U. Leopold [ctb]
G.B.M. Heuvelink [ctb]

Maintainer J.A. Torres-Matallana <arturo.torres@list.lu>

Description The EmiStatR provides a fast and parallelised calculator to estimate combined wastewater emissions.
It supports the planning and design of urban drainage systems, without the requirement of extensive simulation tools. The EmiStatR package implements modular R methods. This enables to add new functionalities through the R framework. Furthermore, EmiStatR was implemented with an interactive user interface with sliders and input data exploration.

License GPL (>= 3)

Depends methods, foreach, parallel, doParallel, lattice, shiny

R topics documented:

`EmiStatR-package` *Estimation of Wastewater Emissions in Combined Sewer Systems.*

Description

The EmiStatR provides a fast and parallelised calculator to estimate combined wastewater emissions. It supports the planning and design of urban drainage systems, without the requirement of extensive simulation tools. The EmiStatR package implements modular R methods. This enables to add new functionalities through the R framework. Furthermore, EmiStatR was implemented with an interactive user interface with sliders and input data exploration.

Details

The DESCRIPTION file:

Package:	EmiStatR
Type:	Package
Version:	1.2.0
Date:	2016-07-22
License:	GPL (>= 3)
Depends:	R (>= 2.10), methods, foreach, parallel, doParallel, lattice, shiny

Author(s)

J.A. Torres-Matallana [aut, cre] K. Klepiszewski [aut, cre] U. Leopold [ctb] G.B.M. Heuvelink [ctb]

Maintainer: J.A. Torres-Matallana

See Also

See also the class the method `EmiStatR`

`EmiStatR-methods` *S4 Methods for Function* `EmiStatR`

Description

S4 methods for function `EmiStatR`. Given the inputs either from the Shiny applications "EmiStatR_input - Shiny" and "EmiStatR_inputCSO - Shiny" or user-defined, the methods invoke the main core of the tool and writes the output files in the specified folder.

Usage

```
EmiStatR(x)
```

Arguments

`x`	An object of class input

Value

Object of class `"list"`. This object contains N lists, where N is the number of structures to simulate. Each list contains a list with three elements: a data.frame named "out1", a data.frame named "out2", a vector named "lista". "out1" contains n observations of 21 variables, where n is the length of the precipitation time series. The 21 variables are the following time series:

1. id, identification number
2. Time [y-m-d h:m:s]
3. P [mm], precipitation
4. i [mm/(min)], intensity (if available)
5. V_r [m3], rain water volume
6. V_dw [m3], dry weather volume
7. cs_mr [-], combined sewage mixing ratio
8. o_tfyn [yes=1/no=0], status variable to know when the Combined Sewer Overflow Tank (CSOT)

is filling up
9. V_Tank [m3], volume of CSOT filling up
10. V_Ov [m3], overflow volume
11. B_COD_Ov [kg], Chemical Oxygen Demand (COD) overflow load
12. B_NH4_Ov [kg], ammonium (NH4) overflow load
13. C_COD_Ov [mg/l], COD overflow concentration
14. C_NH4_Ov [mg/l], NH4 overflow concentration
15. d_Ov [min], total duration of overflows
16. f_Ov [ocurrence], frequency of overflows (just an approximation)
17. V_InTank [m3], volume at entrance of the CSOT
18. B_COD_InTank [Kg], COD load at entrance of the CSOT
19. B_NH4_InTank [Kg], NH4 load at entrance of the CSOT
20. C_COD_InTank [mg/l], COD concentration at entrance of the CSOT
21. C_NH4_InTank [mg/l], NH4 concentration at entrance of the CSOT

The summary of the overflow data, "out2", contains 11 observations of 2 variables. The 11 observations are:
1. Period [day], length of time of the precipitation time series
2. Duration, d_Ov, [min], overflow duration
3. Frecuency, f_Ov, [ocurrence] (aprox.), overflow frecuency
4. Volume, V_Ov, [m3], total overflow volume
5. Flow, Q_Ov, [l/s], total overflow flow
6. COD load, B_COD_Ov, [kg], total COD load
7. Average COD concentration, C_COD_ov_av, [mg/l], in overflows
8. Maximum COD concentration, C_COD_Ov_max, [mg/l], in overflows
9. NH4 load, B_NH4_Ov, [kg], total NH4 load
10. Average NH4 concentration,C_NH4_Ov_av, [mg/l], in overflows
11. Maximum NH4 concentration, C_NH4_Ov_max, [mg/l], in overflows

"Lista" contains the identification name(s) of the N structure(s). If export is allowed then three plain text .csv files are created, one for "out1", the second for "out2", the third one a summary for all the structures based in "out2". Also, one .pdf file is printed which illustrates the precipitation and Combined Sewer Overflow (CSO) volume, COD concentration, and NH4 concentration time series. These files are exported to the directory EmiStatR_output located in the `folderOutput` path.

Methods

`signature(x = "input")` execute EmiStatR function

Examples

```
## running GUI
library("EmiStatR")

appDir <- system.file("shiny", package = "EmiStatR")
setwd(appDir)

## (uncomment for running)
# runApp("EmiStatR_input")
# runApp("EmiStatR_inputCSO")

## executing EmiStatR
input.default <- input()
sim   <- EmiStatR(input.default)
```

```
str(sim)

## a dummy example of plot
par(mfrow=c(2,2), oma = c(0,0,2,0))
plot(x=sim[[1]][[1]][[2]], y=sim[[1]][[1]][[3]], typ="l", col="blue",
     xlab = "time", ylab = colnames(sim[[1]][[1]])[3], main = "Precipitation")
plot(x=sim[[1]][[1]][[2]], y=sim[[1]][[1]][[10]], typ="l", col="blue",
     xlab = "time", ylab = colnames(sim[[1]][[1]])[10], main = "CSO, volume")
plot(x=sim[[1]][[1]][[2]], y=sim[[1]][[1]][[13]], typ="l", col="blue",
     xlab = "time", ylab = colnames(sim[[1]][[1]])[13], main = "CSO, COD concentration")
plot(x=sim[[1]][[1]][[2]], y=sim[[1]][[1]][[14]], typ="l", col="blue",
     xlab = "time", ylab = colnames(sim[[1]][[1]])[14], main = "CSO, NH4 concentration")
mtext(paste("Structure", sim[[1]][[3]][[1]]), outer=TRUE, cex = 1.5)
```

`Esch_Sure2010` *An example time series for the EmiStatR package*

Description

This dataset is a data.frame with two columns: Time [y-m-d h:m:s] and Precipitation P [mm]. The station of measuring, Esch Sure, is located close to the catchment of the combined sewer overflow tank at Goesdorf, Grand-Duchy of Luxembourg.

Usage

```
data("Esch_Sure2010")
```

Format

A data frame with 52560 observations on the following 2 variables.

`time` a POSIXct

`'P [mm]'` a numeric vector

Examples

```
data(Esch_Sure2010)

plot(Esch_Sure2010[,1], Esch_Sure2010[,2], col="blue", typ="l",
     xlab = "time", ylab = "Precipitation [mm]")
```

input-class *Class* "input"

Description

The class provides a container for inputs required to invoke EmiStatR method.

Objects from the Class

Objects can be created by calls of the form input() or new("input").

Slots

spatial: Object of class "numeric", 0 (default) for non-spatial input, 1 for spatial input (not implemented).

zero: Object of class "numeric", aproximation to zero value. Default 1E-5.

folder: Object of class "character", path of the Shiny applications. Default

```
system.file("shiny", package = "EmiStatR")
```

folderOutput: Object of class "character", path of the directory to save outputs. By default the same as folder.

cores: Object of class "numeric", number of CPU cores to be used in parallel computing. If cores = 0 no parallel computation is done. Default 1.

ww: Object of class "list". This list contains three numeric elements for the wastewater characteristics. First element qs, individual water consumption of households [l/(PE d)]. Second element CODs, sewage pollution - COD concentration [g/(PE d)]. Third element NH4s, sewage pollution - NH4 concentration [g/(PE d)].

inf: Object of class "list". This list contains three numeric elements for infiltration water characteristics. First element qf, infiltration water inflow [l/(s ha)]. Second element CODf, infiltration water pollution - COD concentration [g/(PE d)]. Third element NH4f, infiltration water pollution - NH4 concentration [g/(PE d)].

rw: Object of class "list". This list contains three elements for rainwater characteristics. First element CODr (numeric), rainwater pollution - COD concentration [mg/l]. Second element NH4r (numeric), rainwater pollution - NH4 concentration [mg/l]. Third element stat (character), name of the rain measurement station.

tf: Object of class "numeric", stormwater runoff. Flow time in the sewer system [min]. If tf is less or equal than 20 min, then af = 1, i.e. no attenuation of the rainfall.

P1: Object of class "data.frame" with columns named tt (date and time), P (rain time series), and i (intensity).

st: Object of class "list". This object contains n lists, where n is the number of structures to simulate. Every list should contain 12 elements: id, identification number [-]; ns, name of the structure [-]; nm, name of the municipality [-]; nc, name of the catchment [-]; numc, number of the catchment [-]; use, use of the soil [-]; Ages, total area [ha]; Ared, reduced area - impervious area [ha]; tfS, time flow structure [min]; pe, population equivalent [PE]; Qd, throttled outflow [l/s]; and V, volume [m3].

export: Object of class "numeric". If 1 (default) then the results are saved in folderOutput. Set to 0 for not writing in output files.

Methods

EmiStatR `signature(x = "input")`: execute EmiStatR function

Author(s)

J.A. Torres-Matallana

Examples

```
showClass("input")

## running EmiStatR with default input
library("EmiStatR")

  # creating an Input object
  input.default <- new("input")
  str(input.default)

  # running EmiStat
  sim1 <- EmiStatR(input.default)

## running EmiStatR with user defined input
  data("P1")

  # defining estructures E1 and E2
  E1 <- list(id = 1, ns = "Goesdorf", nm = "Goesdorf", nc = "Obersauer", numc = NA,
             use = "Residencial/Industrial", Ages = 16.5, Ared = 7.6, tfS = 10,
             pe = 611, Qd = 9, V = 190)

  E2 <- list(id = 2, ns = "Kaundorf", nm = "Kaundorf", nc = "Obersauer", numc = NA,
             use = "Residencial/Industrial", Ages = 22, Ared = 11, tfS = 10,
             pe = 358, Qd = 9, V = 180)

  # defining Input objet
  input.user <- input(spatial = 0, zero = 1e-5, folder = system.file("shiny", package = "EmiStatR"),
                    folderOutput = system.file("shiny", package = "EmiStatR"), cores = 1,
                      ww = list(qs = 150, CODs = 120, NH4s = 11),
                      inf = list(qf= 0.05, CODf = 0, NH4f =0),
                      rw = list(CODr = 107, NH4r = 0, stat = "Dahl"),
                      tf = 20, P1 = P1,
                      st = list(E1=E1, E2=E2), export = 1)
  str(input.user)

  # invoking EmiStatR
  sim2 <- EmiStatR(input.user)
```

P1 *An example time series for the EmiStatR package*

Description

This dataset is a list that contains a data.frame with two columns: Time [y-m-d h:m:s] and Precipitation P [mm]. The station of measuring, Dahl, is located close to the catchment of the combined sewer overflow tank at Goesdorf, Grand-Duchy of Luxembourg.

Usage

```
data("P1")
```

Format

A data frame with 4464 observations on the following 2 variables.

`time` a POSIXct

`'P [mm]'` a numeric vector

Source

http://agrimeteo.lu/

Examples

```
data("P1")

plot(P1[,1], P1[,2], col="blue", typ="l", xlab = "time", ylab = "Precipitation [mm]")
```

Index

Annex B

Package 'AggProp'

November 15, 2016

Type Package

Title Temporal Aggregation and Uncertainty Propagation

Version 1.0

Date 2016-10-22

Author J.A. Torres-Matallana [aut, cre]
U. Leopold [ctb]
G.B.M. Heuvelink [ctb]

Maintainer J.A. Torres-Matallana <arturo.torres@list.lu>

Description AggProp, provides several R functions for temporal aggregation of environmental variables used in Urban Drainage Models (UDMs), as precipitation and pollutants.
Also, it provides methods and functions for uncertainty propagation via Monte Carlo simulation.
This package, moreover, provides specific analysis functions for urban drainage system simulation to evaluate water quantity and quality in combined sewer overflows (CSOs).

License GPL (>=3)

Depends methods, mAr, data.table, lmom, xts

R topics documented:

AggProp-package *Temporal Aggregation and Uncertainty Propagation*

Description

AggProp, provides several R functions for temporal aggregation of environmental variables used in Urban Drainage Models (UDMs), as precipitation and pollutants. Also, it provides methods and functions for uncertainty propagation via Monte Carlo simulation. This package, moreover, provides specific analysis functions for urban drainage system simulation to evaluate water quantity and quality in combined sewer overflows (CSOs).

Details

The DESCRIPTION file:

Package:	AggProp
Type:	Package
Version:	1.0
Date:	2016-10-22
License:	GPL (>= 3)
Depends:	R (>= 2.10), methods, mAr, data.table, lmom, xts

Author(s)

J.A. Torres-Matallana [aut, cre]; U. Leopold [ctb]; G.B.M. Heuvelink [ctb].

Maintainer: J.A. Torres-Matallana.

Agg *Temporal aggregation of environmental variables*

Description

Function for temporal aggregation of environmental variables. `Agg` is a wrapper function of `aggregate` from `stats` package.

Usage

```
Agg(data, nameData, delta, func, namePlot)
```

Arguments

`data`	A `data.frame` that contains the time series of the environmental variable to be aggregated, e.g. precipitation. This data.frame should have at two columns: the first one, Time [y-m-d h:m:s]; the second one, a numeric value equal to

	the magnitude of the environmental variable. If the environmental variable is different than precipitation, then the column name of the values should be named as `value`.
`nameData`	A `character` string that defines the name of the environmental variable to be aggregated.
`delta`	A `numeric` value that specifies the level of aggregation required in minutes.
`func`	The name of the function of aggregation e.g. `mean`, `sum`.
`namePlot`	A `character` string that defines the title of the plot generated.

Value

A `data.frame` with two columns:

`time`	the date-time time series of the aggregated variable
`value`	time series with the magnitude of the aggregated variable.

Author(s)

J.A. Torres-Matallana

Examples

```
## temporal aggregation

library(EmiStatR)
data(P1)
head(P1)

library(STUnc)
P1.agg <- Agg(data = P1, nameData = "P1", delta = 120 , func = sum,
              namePlot = "Temporal aggregation of precipitation P1 in mm")

head(P1.agg)
tail(P1.agg)
```

`MC.analysis` *Analysis of the Monte Carlo simulation*

Description

Function for running the analysis of the Monte Carlo simulation.

Usage

```
MC.analysis(x, delta, qUpper, p1.det, sim.det, event.ini, event.end, ntick, summ.data = NULL)
```

Arguments

x	A list.
delta	A numeric value that specifies the level of aggregation required in minutes.
qUpper	A character string that defines the upper percentile to plot the confidence band of results, several options are possible "q999" the 99.9th percentile, "q995" the 99.5th percentile, "q99" the 99th percentile, "q95" the 95th percentile, "q50" the 50th percentile. The lower boundary of the confidence band (showed in gray in the output plots) is the 5th percentile in all cases.
p1.det	A data.frame that contains the time series of the main driving force of the system to be simulated deterministically, e.g. precipitation. This data.frame should have only two columns: the first one, Time [y-m-d h:m:s]; the second one, a numeric value equal to the magnitude of the environmental variable.
sim.det	A list that contains the results of the deterministic simulation, here the output of EmiStatR given p1.det. See the method EmiStatR from the homonym package for details.
event.ini	A time-date string in POSIXct format that defines the initial time for event analysis.
event.end	A time-date string in POSIXct format that defines the final time for event analysis.
ntick	A numeric value to specify the number of ticks in the x-axis for the event time-window plots.
summ	A list by default NULL. If provided, the list should contain an output of the MC.analysis function, and the analysis is done again without the calculation of some of the internal variables, therefore the analysis is faster.

Value

A list of length 2:

summ	A list that contains the summary statistics of the Monte Carlo simulation per output variable. Each output variable is summarised by calculating the mean "Mean", standard deviation "sd", variance "Variance", 5th, 25th, 50th, 75th, 95th, 99.5th, 99.9th percentiles "q05", "q25", "q50", "q75", "q95", "q995", "q999", the max "Max", the sum "Sum", time "time", and the deterministic precipitation "p1", all variables as time series.
variance	A data.frame that contains the summary statistics of the variance of the Monte Carlo simulation per output variable.

Author(s)

J.A. Torres-Matallana

See Also

See also setup, MC.setup, MC.sim.

Examples

```
## the Monte Carlo simulation
library(EmiStatR)
data(P1)

library(STUnc)

setting_EmiStatR <-  setup(id       = "MC_sim1",
                           nsim     = 15,
                           seed     = 0.7010607,
                           mcCores  = 1,
                           ts.input = P1,
                           rng      = rng <- list(
                             qs   = 150,    # [l/PE/d]
                            CODs = c(pdf = "nor", mu = 4.378, sigma = 0.751),    # [g/PE/d]
                            NH4s = c(pdf = "nor", mu = 1.473, sigma = 0.410),    # [g/PE/d]
                             qf   = 0.05,     # [l/s/ha]
                             CODf = 0,              # [g/PE/d]
                             NH4f = 0,              # [g/PE/d]
                            CODr = c(pdf = "nor", mu = 3.60, sigma = 1.45),     # [mg/l]
                             NH4r = 2,              # [mg/l]
                             tf   = 20,          # [min]
                             nameCSO = "E1",        # [-]
                             id      = 1,           # [-]
                             ns      = "FBH Goesdorf",  # [-]
                             nm      = "Goesdorf",  # [-]
                             nc      = "Obersauer", # [-]
                             numc    = 1,          # [-]
                             use     = "R/I",  # [-]
                             Ages    = 16.5,        # [ha]
                             Ared    = 7.6,  # [ha]
                             tfS     = 10, # [min]
                             pe      = 611, # [PE]
                             Qd      = 9,   # [l/s]
                             V       = 190),  # [m3]
                           ar.model  = ar.model <- list(
                             CODs    = 0.5,#0.9,
                             NH4s    = 0.5,#0.9,
                             CODr    = 0.46),
                           folderOutput = "~/R-output")

MC_setup <- MC.setup(setting_EmiStatR)

sims <- MC.sim(MC_setup, EmiStatR.cores = 0)

## Monte Carlo simulation analysis

# Deterministic simulation
# Definition of structure 1, E1:

E1 <- list(id = 1, ns = "FBH Goesdorf", nm = "Goesdorf", nc = "Obersauer", numc = 1,
           use = "R/I", Ages = 16.5, Ared = 7.6, tfS = 10, pe = 611, Qd = 9, V = 190)
```

```
# Definition of the deterministic input:
library(EmiStatR)
data(P1)

input.det <- input(spatial = 0, zero = 1e-5,
                   folder = system.file("shiny", package = "EmiStatR"),
                   folderOutput ="~/R-output", cores = 0,
                   ww = list(qs = 150, CODs = 104, NH4s = 4.7),
                   inf = list(qf= 0.05, CODf = 0, NH4f = 0),
                   rw = list(CODr = 107, NH4r = 2, stat = "Dahl"),
                   tf = 20, P1 = P1, st = list(E1=E1), export = 0)

# Invoking `EmiStatR` with the deterministic input:
sim.det   <- EmiStatR(input.det)

# further arguments
delta <- 30 # minutes
qUpper <- "q999"
event.ini <- as.POSIXct("2016-01-14")
event.end <- as.POSIXct("2016-01-16 12:00:00")

analysis <- MC.analysis(sims, delta, qUpper,  P1, sim.det, event.ini, event.end, 5)
str(analysis)
```

`MC.setup` ~~ *Methods for Function* `MC.setup` *in Package* **STUnc** ~~

Description

Given an object of class `setup`, the method can be invoked for setting-up the Monte Carlo simulation. The variables are sampled accordingly to their parameters specified in the slot `rng` of the `setup` object. If `ar.model` is defined in slot `ar.model`, then the specified variables are sampled from the pdf `nor` as an autorregresive (AR) model via the function `arima.sim` from base package `stats`. If `var.model` is defined in slot `var.model`, then the specified variables are sampled from the pdf `nor` as an vector autorregresive (VAR) model via the function `mAr.sim` from package `mAr` (see Barbosa, 2015, and Luetkepohl, 2005, for details). See `setup-class` for further details to define the AR and VAR models.

Usage

```
MC.setup(x)
```

Arguments

`x`	an object of class `setup`.

Methods

```
signature(x = "setup")
```

Author(s)

J.A Torres-Matallana

References

S. M. Barbosa, Package "mAr": Multivariate AutoRegressive analysis, 1.1-2, The Comprehensive R Archive Network, CRAN, 2015.

H. Luetkepohl, New Introduction to Multiple Time Series Analysis, Springer, 2005.

Examples

```
# loading a precipitation time series as input for the setup class

library(EmiStatR)
data(P1)

# A setup with three variables to be considered in the Monte Carlo simulation:
# var1, a constant value variable; var2, a variable sampled from a uniform (uni)
# probability distribution function (pdf) with parameters min and max;
# var3, a variable sampled from a normal (nor) pdf with parameteres mu and sigma

ini <- setup(id = "MC_sim1", nsim = 500, seed = 123, mcCores = 1, ts.input = P1,
             rng = list(var1 = 150, var2 = c(pdf = "uni", min = 50, max = 110),
                        var3 = c(pdf = "nor", mu = 90, sigma = 2.25)),
             folderOutput = "~/r-output"
)

MC_setup <- MC.setup(ini)
str(MC_setup)

## definition of AR models for variables var2 and var3 with AR coefficients 0.995 and 0.460

library(EmiStatR)
data(P1)

ini_ar <- setup(id = "MC_sim1_ar", nsim = 500, seed = 123, mcCores = 1, ts.input = P1,
                rng = list(var1 = 150, var2 = c(pdf = "nor", mu = 150, sigma = 5),
                           var3 = c(pdf = "nor", mu = 90, sigma = 2.25)),
                ar.model = ar.model <- list(var2 = 0.995, var3 = 0.460),
                folderOutput = "~/r-output"
)

MC_setup_ar <- MC.setup(ini_ar)
str(MC_setup_ar)

## definition of a bi-variate VAR model for variables var2 and var3
```

```
ini_var <- setup(id = "MC_sim1_ar", nsim = 500, seed = 123, mcCores = 1, ts.input = P1,
                 rng = rng <- list(var1 = 150,
                                   var2 = c(pdf = "nor", mu = 150, sigma = 5),
                                   var3 = c(pdf = "nor", mu = 90, sigma = 2.25)),
                 var.model = var.model <- list( inp = c("var2", "var3"),
                                                w = c(0.048, 0.021),
                                             A = matrix(c(0.992, -8.8e-05, -31e-4, 0.995),
                                                nrow=2, ncol=2),
                                             C = matrix(c(0.0091, 0.0022, 0.0022, 0.0019),
                                                nrow=2, ncol=2)),
                 folderOutput = "~/r-output"
)


MC_setup_var <- MC.setup(ini_var)
str(MC_setup_var)
```

`MC.sim` *~~ Methods for Function* `MC.sim` *~~*

Description

Method to be invoked for running the Monte Carlo simulation. The simulator used is the method `EmiStatR` from the homonym package. This method should be rewritted for working with another simulator.

Usage

```
MC.sim(x, EmiStatR.cores)
```

Arguments

`x`	an object of class `list` as is defined by method `MC.setup`.
`EmiStatR.cores`	a `numeric` value for specifying the number of cores (CPUs) to be used in the EmiStatR method. Use zero for not use parallel computation. See class `input` of package `EmiStatR` for details.

Value

A `list` of length 2:

`mc`	A `list` that contains the `MC_setup`, `timing` and `lap` objects.
`sim1`	A `list` that contains the Monte Carlo matrices of the simulator output.

Methods

```
signature(x = "list", EmiStatR.cores = "numeric")
```

Examples

```
## simulation of precipitation time series, P1, within EmiStatR

# loading the precipitation time series P1
library(EmiStatR)
data(P1)

library(STUnc)
setting_EmiStatR <-  setup(id       = "MC_sim1",
                           nsim     = 23,
                           seed     = 0.7010607,
                           mcCores  = 1,
                           ts.input = P1,
                           rng      = rng <- list(
                             qs   = 150,    # [l/PE/d]
                          CODs = c(pdf = "nor", mu = 4.378, sigma = 0.751),    # log[g/PE/d]
                          NH4s = c(pdf = "nor", mu = 1.473, sigma = 0.410),    # log[g/PE/d]
                               qf   = 0.05,           # [l/s/ha]
                               CODf = 0,              # [g/PE/d]
                               NH4f = 0,              # [g/PE/d]
                            CODr = c(pdf = "nor", mu = 3.60, sigma = 1.45),     # log[mg/l]
                               NH4r = 2,              # [mg/l]
                               tf   = 20,             # [min]
                               nameCSO = "E1",        # [-]
                               id      = 1,           # [-]
                               ns      = "FBH Goesdorf",  # [-]
                               nm      = "Goesdorf",  # [-]
                               nc      = "Obersauer", # [-]
                               numc    = 1,           # [-]
                               use     = "R/I",       # [-]
                               Ages    = 16.5,        # [ha]
                               Ared    = 7.6,         # [ha]
                               tfS     = 10,          # [min]
                               pe      = 611,         # [PE]
                               Qd      = 9,           # [l/s]
                               V       = 190),        # [m3]
                             ar.model  = ar.model <- list(
                               CODs    = 0.5,
                               NH4s    = 0.5,
                               CODr    = 0.46),
                             folderOutput = "~/r-output")

MC_setup <- MC.setup(setting_EmiStatR)

sims <- MC.sim(MC_setup, EmiStatR.cores = 0)
str(sims)
```

---

setup *Class* "setup"

Description

Class to create objects of signature setup. setup object should be passed to the method MC.setup.

Usage

```
setup(id = "MC_sim_1", nsim = 1, seed = 0.7010607, mcCores = 1, ts.input = NULL,
      rng = NULL, ar.model  = list(NULL), var.model = list(NULL),
      folderOutput = "~/R-output ")
```

Objects from the Class

Objects can be created by calls of the form `setup(...)`.

Slots

`id`: Object of class `"character"` to identify the Monte Carlo simulation.

`nsim`: Object of class `"numeric"` to specify the number of Monte Carlo runs.

`seed`: Object of class `"numeric"` to specify the seed of the random numbers generator.

`mcCores`: Object of class `"numeric"` to specify the number of cores (CPUs) to be used in the Monte Carlo simulation.

`ts.input`: Object of class `"data.frame"` that contains the time series of the main driving force of the system to be simulated, e.g. precipitation. This data.frame should have at least two columns: the first one, Time [y-m-d h:m:s]; the second one, a numeric value equal to the magnitude of the environmental variable. This data.frame can also contain more that one column to allow several time series in several columns. If the data.frame has more than two columns, then the number of columns should be at least equal to nsim. If the number of columns is greater than nsim, the columns in excess are not recycled because the simulation will last nsim iterations.

`rng`: Object of class `"list"` that contains the names and values of the variables to be used in the Monte Carlo simulation. Five modes are available: 1) constant value, i.e. this variable will have a constant value along the Monte Carlo simulation; 2) a variable sampled from a uniform (uni) probability distribution function (pdf) with parameters for the lower boundary min and upper boundary max; 3) a variable sampled from a normal (nor) pdf with parameteres mean mu and standard deviation sigma; 4) a variable sampled from an autorregresive (AR) model and normal (nor) pdf with parameteres mean mu and standard deviation sigma, the coefficients of the AR model should be defined in the slot ar.model; 5) a variable sampled from an vector autorregresive (VAR) model and normal (nor) pdf with parameteres mean mu and standard deviation sigma, this mode is enabled by defining the vector of intercept terms w, the matrix of AR coefficients A, and the noise covariance matrix C in the slot var.model. See examples for the definition of this slot.

`ar.model`: Object of class `"list"` containing the coefficients of the AR model as vectors which name is the variable to be modeled and length the order of the model as is required for function arima.sim from the base package stats. The named variables here should correspond to a pdf nor in the slot rng. See examples for the definition of this slot.

`var.model`: Object of class `"list"` containing the the vector of intercept terms w, the matrix of AR coefficients A, and the noise covariance matrix C of the VAR model which name is the variable to be modeled and length the order of the model as is required for function mAr.sim from the package mAr. The named variables in this slot should correspond to a pdf nor in the slot rng. The current implementation considers the bi-variate case. See examples for the definition of this slot. For mathemamtical details see Luetkepohl (2005).

`folderOutput`: Object of class `"character"` defining the working directory to save the outputs of the Monte Carlo simulation. This folder should already exist.

Author(s)

J.A Torres-Matallana

References

S. M. Barbosa, Package "mAr": Multivariate AutoRegressive analysis, 1.1-2, The Comprehensive R Archive Network, CRAN, 2015.

H.Luetkepohl, New Introduction to Multiple Time Series Analysis, Springer, 2005.

Examples

```
# loading a precipitation time series as input for the setup class

library(EmiStatR)
data(P1)

# A setup with three variables to be considered in the Monte Carlo simulation:
# var1, a constant value variable; var2, a variable sampled from a uniform (uni)
# probability distribution function (pdf) with parameters min and max;
# var3, a variable sampled from a normal (nor) pdf with parameteres mu and sigma

ini <- setup(id = "MC_sim1", nsim = 500, seed = 123, mcCores = 1, ts.input = P1,
             rng = list(var1 = 150, var2 = c(pdf = "uni", min = 50, max = 110),
                        var3 = c(pdf = "nor", mu = 90, sigma = 2.25)),
             folderOutput = "~/r-output"
)

str(ini)

## definition of AR models for variables var2 and var3 with AR coefficients 0.995 and 0.460

library(EmiStatR)
data(P1)

ini_ar <- setup(id = "MC_sim1_ar", nsim = 500, seed = 123, mcCores = 1, ts.input = P1,
                rng = list(var1 = 150, var2 = c(pdf = "nor", mu = 150, sigma = 5),
                           var3 = c(pdf = "nor", mu = 90, sigma = 2.25)),
                ar.model = ar.model <- list(var2 = 0.995, var3 = 0.460),
                folderOutput = "~/r-output"
)
```

```
str(ini_ar)

## definition of a bi-variate VAR model for variables var2 and var3

ini_var <- setup(id = "MC_sim1_ar", nsim = 500, seed = 123, mcCores = 1, ts.input = P1,
                 rng = rng <- list(var1 = 150,
                                   var2 = c(pdf = "nor", mu = 150, sigma = 5),
                                   var3 = c(pdf = "nor", mu = 90, sigma = 2.25)),
                 var.model = var.model <- list( inp = c("var2", "var3"),
                                                w = c(0.048, 0.021),
                                              A = matrix(c(0.992, -8.8e-05, -31e-4, 0.995),
                                                nrow=2, ncol=2),
                                              C = matrix(c(0.0091, 0.0022, 0.0022, 0.0019),
                                                nrow=2, ncol=2)),
                 folderOutput = "~/r-output"
)

str(ini_var)
```

Index

Annex C

Deliverable D2.3 - ESR3: User's guide

J.A. Torres-Matallana (ESR3)

October 24th, 2016

This document constitutes a brief user's guide of the R-packages `EmiStatR 1.2.0` and `AggProp 1.0`. `EmiStatR` provides a fast and parallelised calculator to estimate combined wastewater emissions. It supports the planning and design of urban drainage systems, without the requirement of extensive simulation tools. The `EmiStatR` implements modular R methods, this enables to add new functionalities through the R framework.

The Agregation and Disaggregation, and uncertainty Propagation R-package, `AggProp`, provides several R methods and functions for temporal aggregation of environmental variables as precipitation and pollutants of UDMs, and for temporal disaggregation of precipitation as a model input for UDMs. Also, it provides methods and functions for uncertainty propagation for lumped Urban Drainage Models (UDMs) via Monte Carlo simulation. This package, moreover, provides specific analysis functions for urban drainage system simulation to evaluate water quantity and quality in combined sewer overflows (CSOs).
This guide illustrates the class `input` and the main method `EmiStatR` of the homonimus R-package. Also, the class `setup` and the methods and functions `MC.setup`, `MC.sim` and `MC.analysis` of the R-package `AggProp` are illustrated.

The precipitation dataset (`EmiStatR`)

First at all the precipitation dataset is loaded

```
library(EmiStatR)

data("P1")
```

This dataset is a list that contains a data.frame with two columns: Time [y-m-d h:m:s] and precipitation P [mm]. The station of measuring, Dahl, is located close to the catchment of the combined sewer overflow tank at Goesdorf, Grand-Duchy of Luxembourg. The dataset is selected to contain records since 1st January 2016 until 31st December 2016. The time step is 10 minutes.

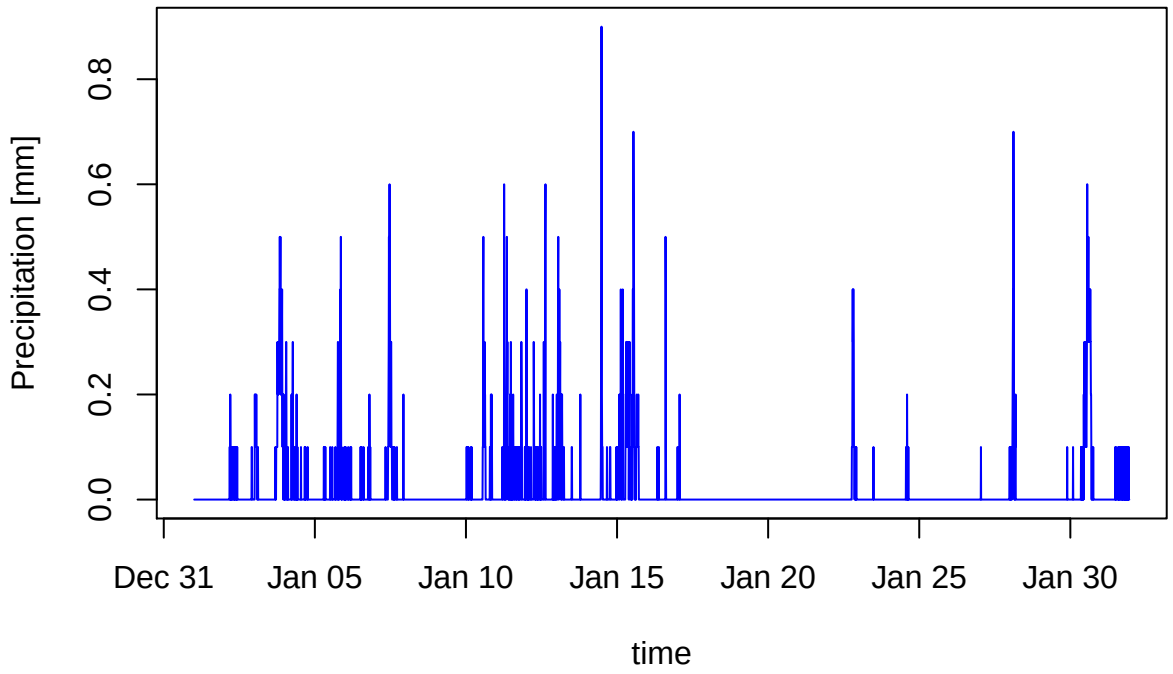
```
P1 <- P1[P1$time > "2016-01-01",]

head(P1); tail(P1)
```

```
##                   time P [mm]
## 8  2016-01-01 00:10:00      0
## 9  2016-01-01 00:20:00      0
## 10 2016-01-01 00:30:00      0
## 11 2016-01-01 00:40:00      0
## 12 2016-01-01 00:50:00      0
## 13 2016-01-01 01:00:00      0
```

```
##                     time P [mm]
## 4459 2016-01-31 22:00:00    0.1
## 4460 2016-01-31 22:10:00    0.0
## 4461 2016-01-31 22:20:00    0.1
## 4462 2016-01-31 22:30:00    0.0
```

```
## 4463 2016-01-31 22:40:00    0.0
## 4464 2016-01-31 22:50:00    0.0
```

```
plot(P1[,1], P1[,2], col="blue", typ="l", xlab = "time", ylab = "Precipitation [mm]")
```



The setup object (AggProp)

Then the object settings of class setup is defined:

```
library(AggProp)

settings <-  setup(id       = "MC_sim1",
                   nsim     = 50,
                   seed     = 0.7010607,
                   mcCores  = 1,
                   ts.input = P1,
                   rng      = rng <- list(
                     qs   = 150,             # [l/PE/d]
                     CODs = c(pdf = "nor", mu = 4.378, sigma = 0.751),    # log[g/PE/d]
                     NH4s = c(pdf = "nor", mu = 1.473, sigma = 0.410),    # log[g/PE/d]
                     qf   = 0.05,            # [l/s/ha]
                     CODf = 0,               # [g/PE/d]
                     NH4f = 0,               # [g/PE/d]
                     CODr = c(pdf = "nor", mu = 3.60, sigma = 1.45),     # log[mg/l]
                     NH4r = 2,               # [mg/l]
                     tf   = 20,              # [min]
                     nameCSO = "E1",         # [-]
                     id      = 1,            # [-]
                     ns      = "FBH Goesdorf",  # [-]
                     nm      = "Goesdorf",  # [-]
```

```
                    nc      = "Obersauer", # [-]
                    numc    = 1,           # [-]
                    use     = "R/I",       # [-]
                    Ages    = 16.5,        # [ha]
                    Ared    = 7.6,         # [ha]
                    tfS     = 10,          # [min]
                    pe      = 611,         # [PE]
                    Qd      = 9,           # [l/s]
                    V       = 190),        # [m3]
                  ar.model  = ar.model <- list(
                    CODs    = 0.5,
                    NH4s    = 0.5,
                    CODr    = 0.46),
                  folderOutput = "~/Documents/02_working/3-Production/03_II-Year/05_models/06_uncertai
```

The `MC.setup` method (`AggProp`)

Setting-up Monte Carlo simulation:

```
MC_setup <- MC.setup(settings)
```

The `MC_setup` object is a list of 10 elements:

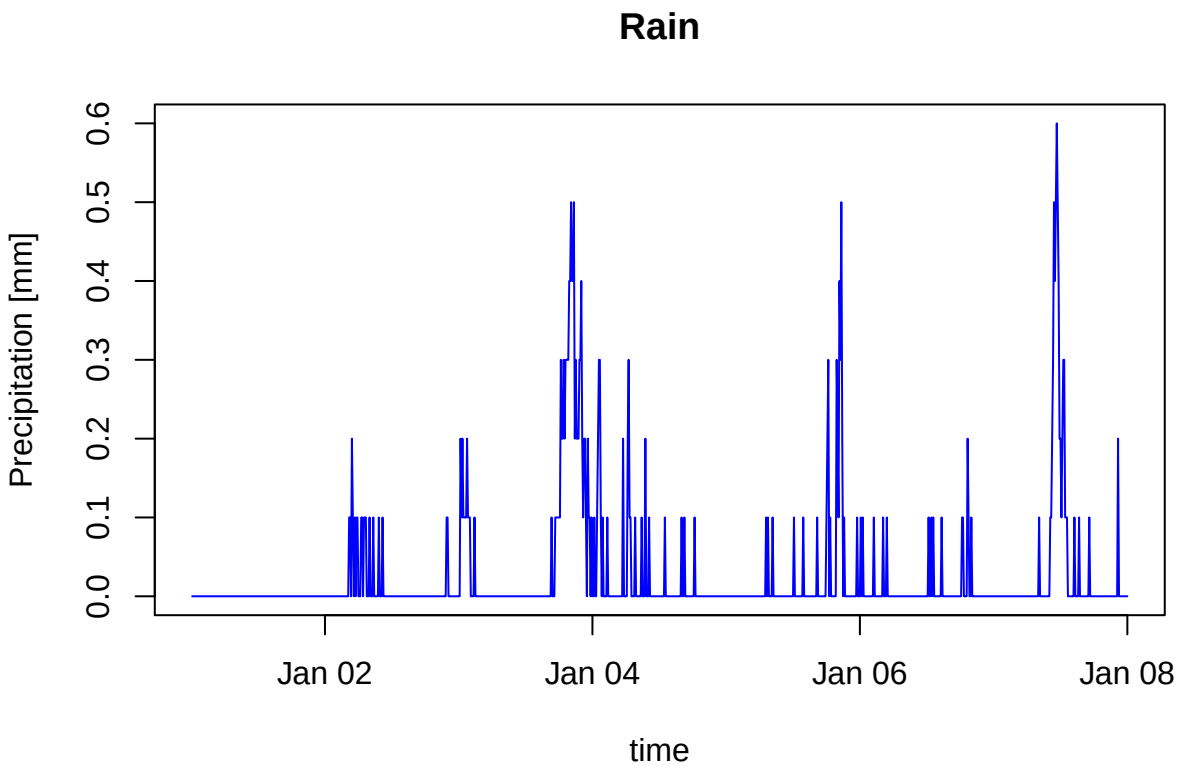
```
str(MC_setup)
```

```
## List of 10
##  $ id          : chr "MC_sim1"
##  $ nsim        : num 50
##  $ seed        : num 0.701
##  $ mcCores     : num 1
##  $ ts.input    :'data.frame':    4457 obs. of  2 variables:
##   ..$ time  : POSIXct[1:4457], format: "2016-01-01 00:10:00" ...
##   ..$ P [mm]: num [1:4457] 0 0 0 0 0 0 0 0 0 0 ...
##  $ rng         :List of 22
##   ..$ qs     : num 150
##   ..$ CODs   : Named chr [1:3] "nor" "4.378" "0.751"
##   .. ..- attr(*, "names")= chr [1:3] "pdf" "mu" "sigma"
##   ..$ NH4s   : Named chr [1:3] "nor" "1.473" "0.41"
##   .. ..- attr(*, "names")= chr [1:3] "pdf" "mu" "sigma"
##   ..$ qf     : num 0.05
##   ..$ CODf   : num 0
##   ..$ NH4f   : num 0
##   ..$ CODr   : Named chr [1:3] "nor" "3.6" "1.45"
##   .. ..- attr(*, "names")= chr [1:3] "pdf" "mu" "sigma"
##   ..$ NH4r   : num 2
##   ..$ tf     : num 20
##   ..$ nameCSO: chr "E1"
##   ..$ id     : num 1
##   ..$ ns     : chr "FBH Goesdorf"
##   ..$ nm     : chr "Goesdorf"
##   ..$ nc     : chr "Obersauer"
```

```
##   ..$ numc   : num 1
##   ..$ use    : chr "R/I"
##   ..$ Ages   : num 16.5
##   ..$ Ared   : num 7.6
##   ..$ tfS    : num 10
##   ..$ pe     : num 611
##   ..$ Qd     : num 9
##   ..$ V      : num 190
##  $ par         :List of 22
##   ..$ qs     : num 150
##   ..$ CODs   : num [1:50, 1:4457] 5.59 5.24 4.23 3.16 3.7 ...
##   ..$ NH4s   : num [1:50, 1:4457] 1.19 0.632 0.893 0.923 1.562 ...
##   ..$ qf     : num 0.05
##   ..$ CODf   : num 0
##   ..$ NH4f   : num 0
##   ..$ CODr   : num [1:50, 1:4457] 3.19 5.05 6.18 2.77 4.56 ...
##   ..$ NH4r   : num 2
##   ..$ tf     : num 20
##   ..$ nameCSO: chr "E1"
##   ..$ id     : num 1
##   ..$ ns     : chr "FBH Goesdorf"
##   ..$ nm     : chr "Goesdorf"
##   ..$ nc     : chr "Obersauer"
##   ..$ numc   : num 1
##   ..$ use    : chr "R/I"
##   ..$ Ages   : num 16.5
##   ..$ Ared   : num 7.6
##   ..$ tfS    : num 10
##   ..$ pe     : num 611
##   ..$ Qd     : num 9
##   ..$ V      : num 190
##  $ ar.model    :List of 3
##   ..$ CODs: num 0.5
##   ..$ NH4s: num 0.5
##   ..$ CODr: num 0.46
##  $ var.model   :List of 1
##   ..$ : NULL
##  $ folderOutput: chr "~/Documents/02_working/3-Production/03_II-Year/05_models/06_uncertainty/Uncert
```

The input variables chosen for the uncertainty propagation analysis and precipitation are plotted. Is worth noting that the three input variables are simulated as autorregresive models order 1, AR1. A time window of 7 days is chosen for plotting:

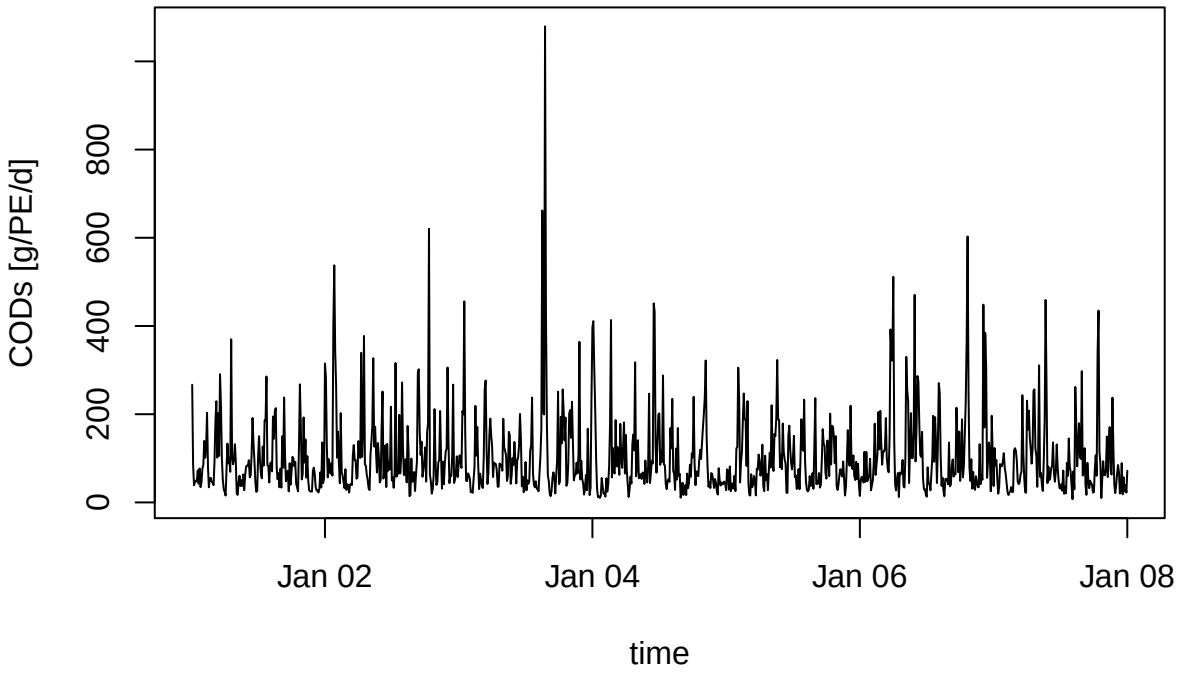
```
id.ini <- 1
id.end <- 7*24*60/10

time <- MC_setup$ts.input$time

plot(as.POSIXct(time[id.ini:id.end]),MC_setup$ts.input[id.ini:id.end,2],
     main = "Rain", type="l", col="blue", xlab="time", ylab="Precipitation [mm]")
```



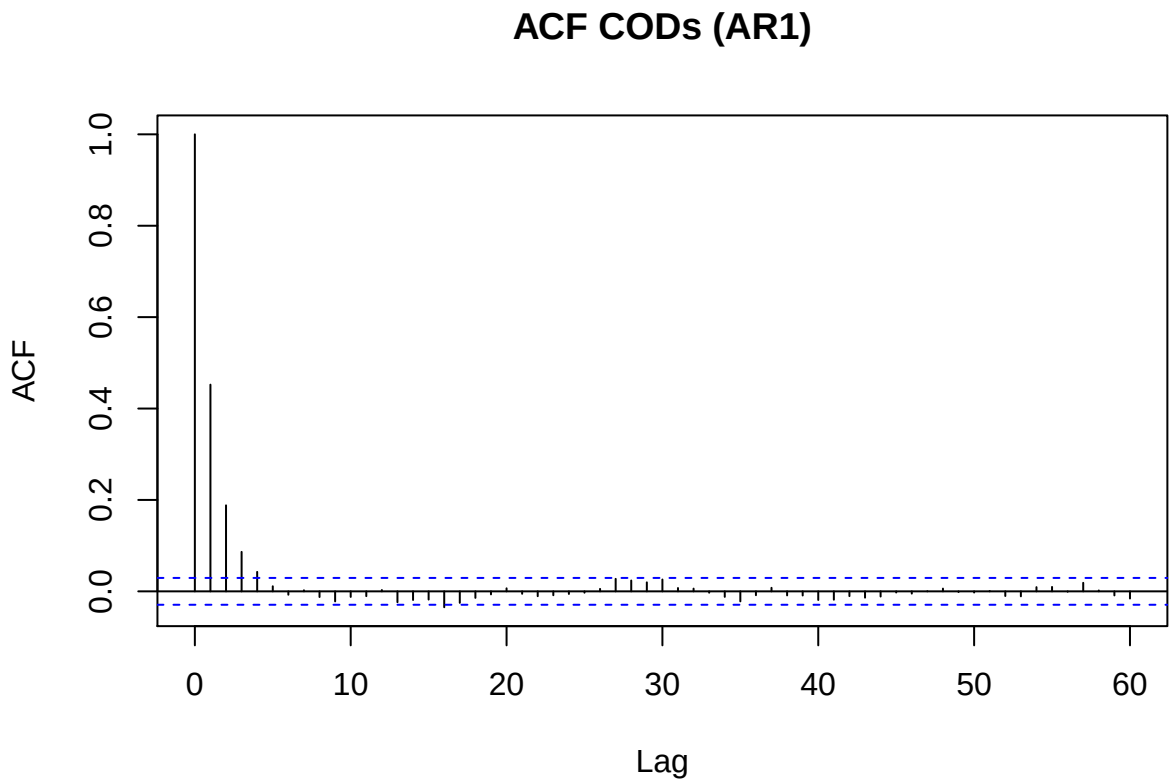
Sewage chemical oxygen demand (COD) pollution, `CODs`

The COD sewage pollution per capita [PE] load per day [g/PE/d], `CODs`, was modeled as an AR1 model with coefficiente 0.5.

```
plot(as.POSIXct(time[id.ini:id.end]),exp(MC_setup$par[["CODs"]][1,id.ini:id.end]),
     main = "", type="l", xlab="time", ylab= "CODs [g/PE/d]")
```



```
acf(exp(MC_setup$par[["CODs"]][1,]), plot=T, type="correlation", na.action=na.pass,
    main = "ACF CODs (AR1)", lag.max = 60)
```

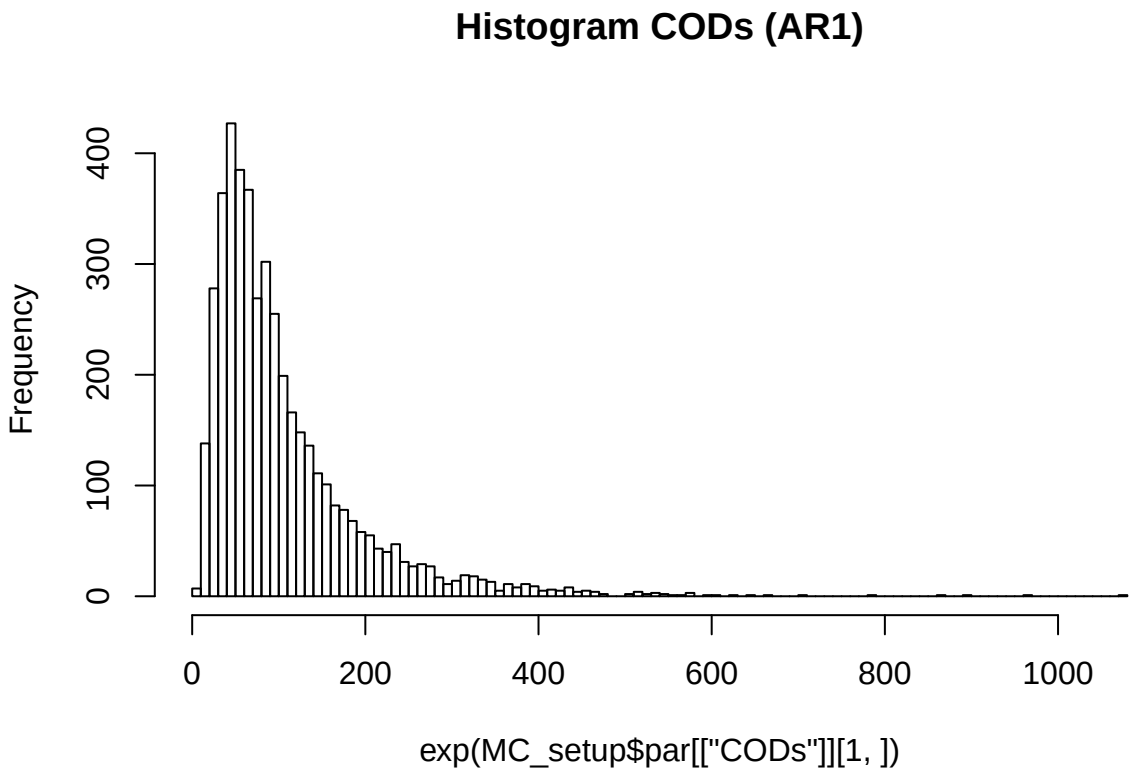



```
mean(exp(MC_setup$par[["CODs"]])); sd(exp(MC_setup$par[["CODs"]]))
```

```
## [1] 105.6079
```

```
## [1] 91.92436
```

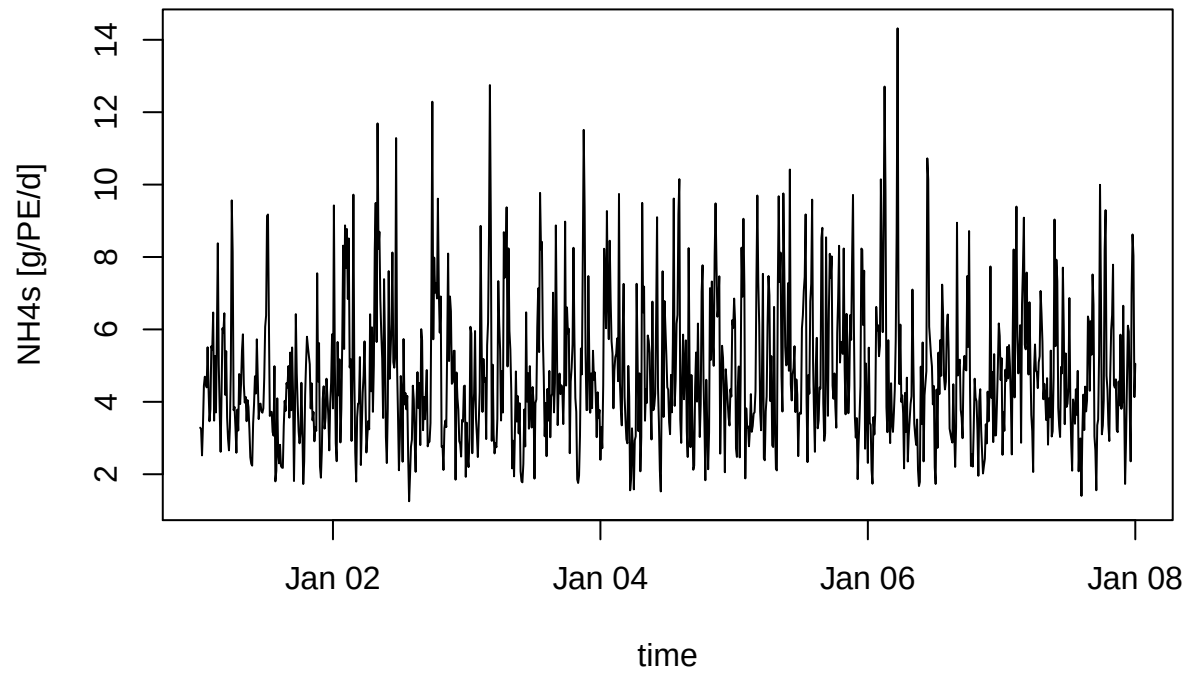
```
hist(exp(MC_setup$par[["CODs"]][1,]),  main = "Histogram CODs (AR1)", breaks=150)
```



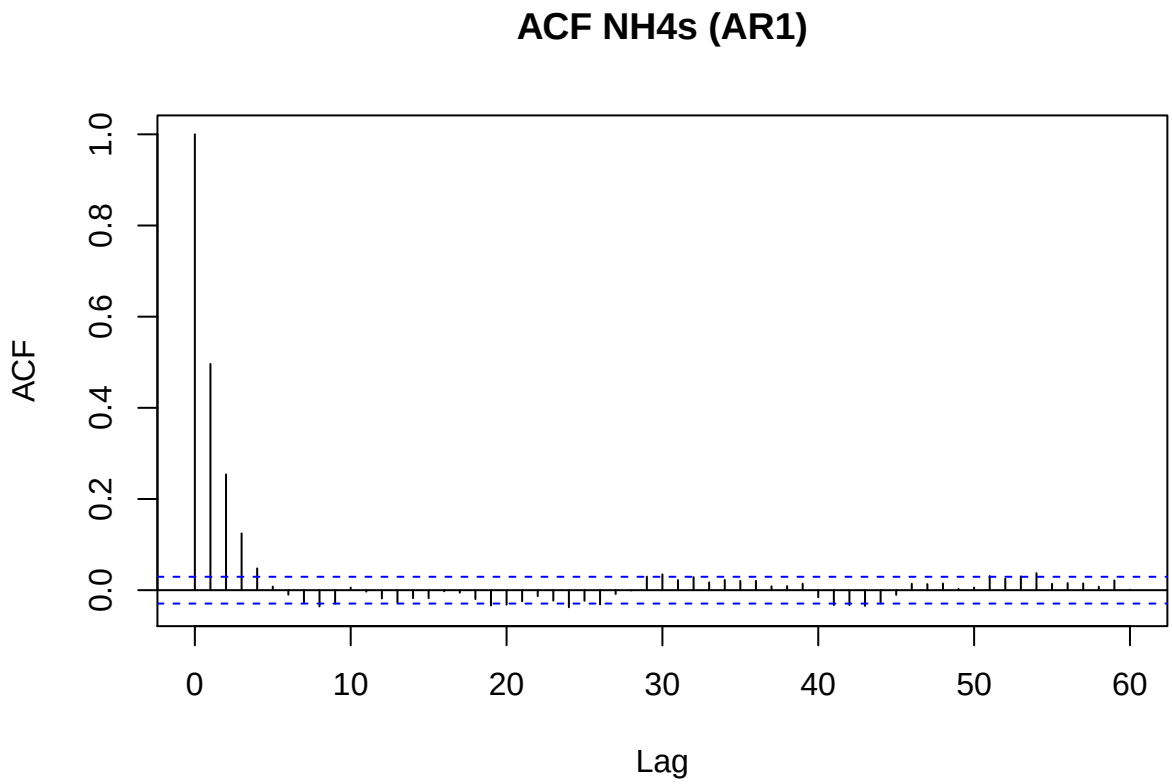
Sewage ammonium (NH4) pollution, `NH4s`

The NH4 sewage pollution per capita [PE] load per day [g/PE/d], `NH4s`, was modeled as an AR1 model with coefficiente 0.5.

```
plot(as.POSIXct(time[id.ini:id.end]),exp(MC_setup$par[["NH4s"]][1,id.ini:id.end]),
     main = "", type="l", xlab="time", ylab= "NH4s [g/PE/d]")
```



```
acf(exp(MC_setup$par[["NH4s"]][1,]), plot=T, type="correlation", na.action=na.pass,
    main = "ACF NH4s (AR1)", lag.max = 60)
```

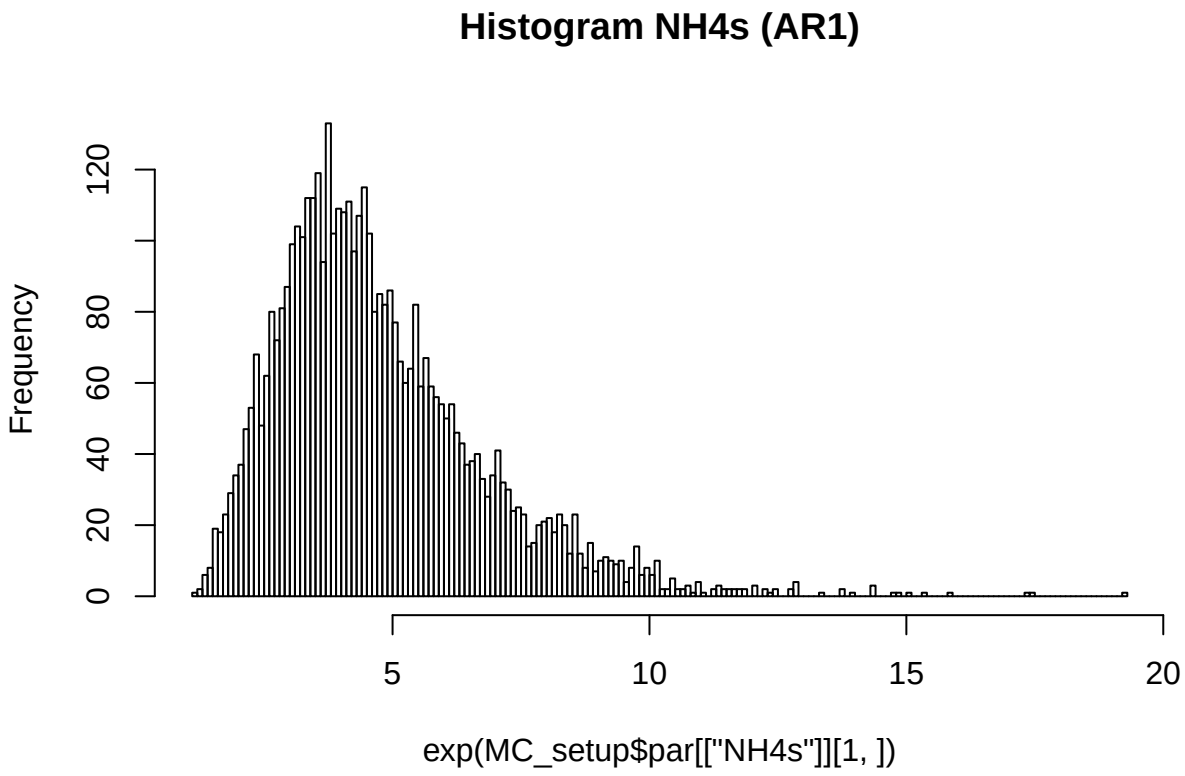



```
mean(exp(MC_setup$par[["NH4s"]])); sd(exp(MC_setup$par[["NH4s"]]))
```

```
## [1] 4.745006
```

```
## [1] 2.031505
```

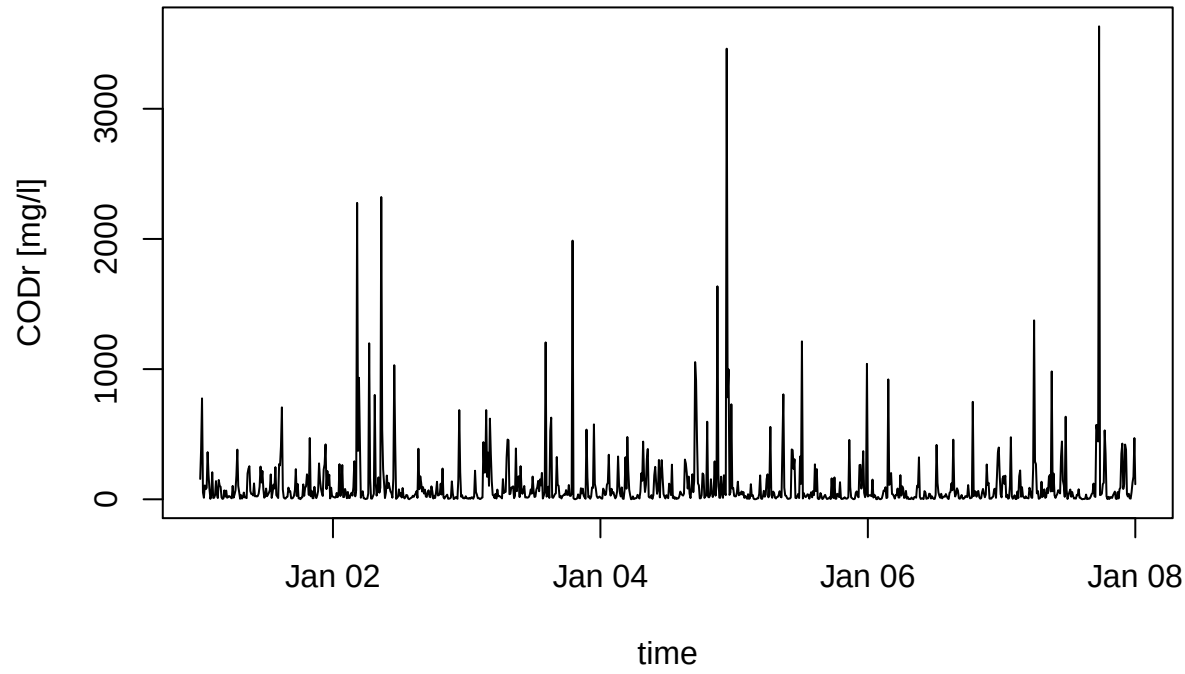
```
hist(exp(MC_setup$par[["NH4s"]][1,]),  main = "Histogram NH4s (AR1)", breaks=150)
```



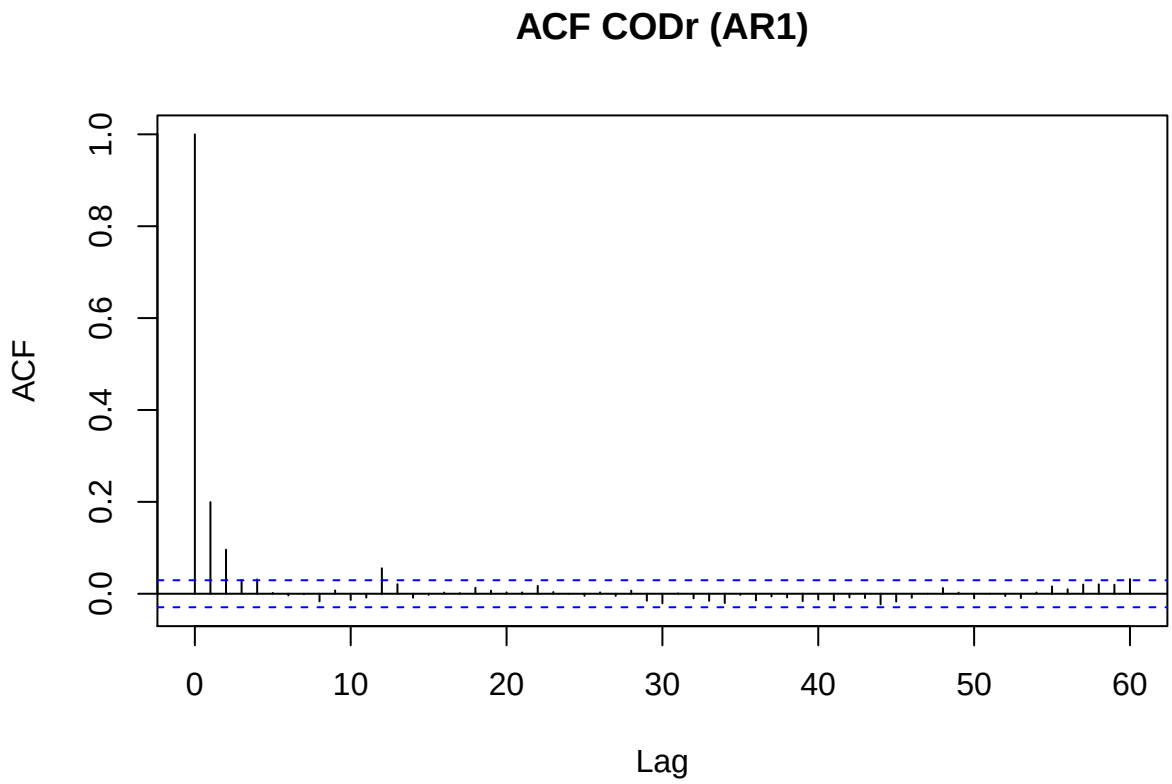
COD rainwater pollution, `CODr`

The rainwater pollution in COD concentration [mg/l], `CODr`, was modeled as an AR1 model with coefficiente 0.46.

```
plot(as.POSIXct(time[id.ini:id.end]),exp(MC_setup$par[["CODr"]][2,id.ini:id.end]),
     main = "", type="l",  xlab="time", ylab= "CODr [mg/l]")
```



```r
acf(exp(MC_setup$par[["CODr"]][1,]), plot=T, type="correlation", na.action=na.pass,
    main = "ACF CODr (AR1)", lag.max = 60)
```

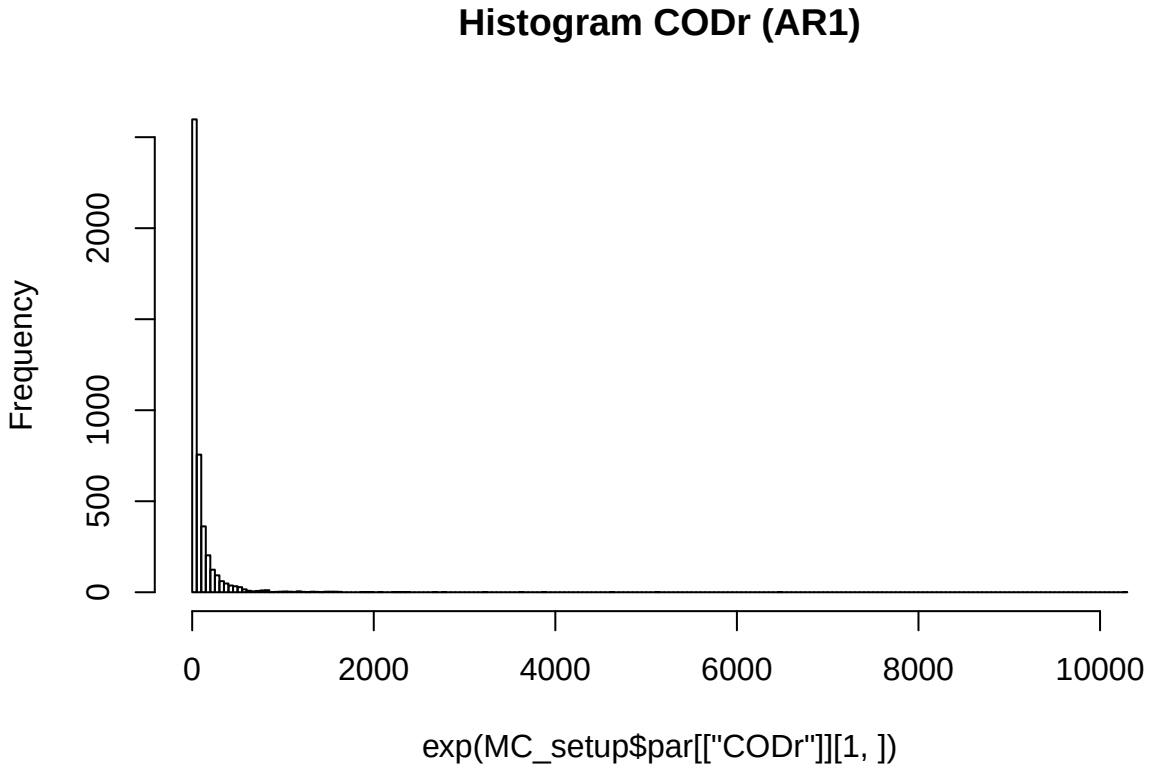



```r
mean(exp(MC_setup$par[["CODr"]])); sd(exp(MC_setup$par[["CODr"]]))
```

```
## [1] 104.3052
```

```
## [1] 266.2689
```

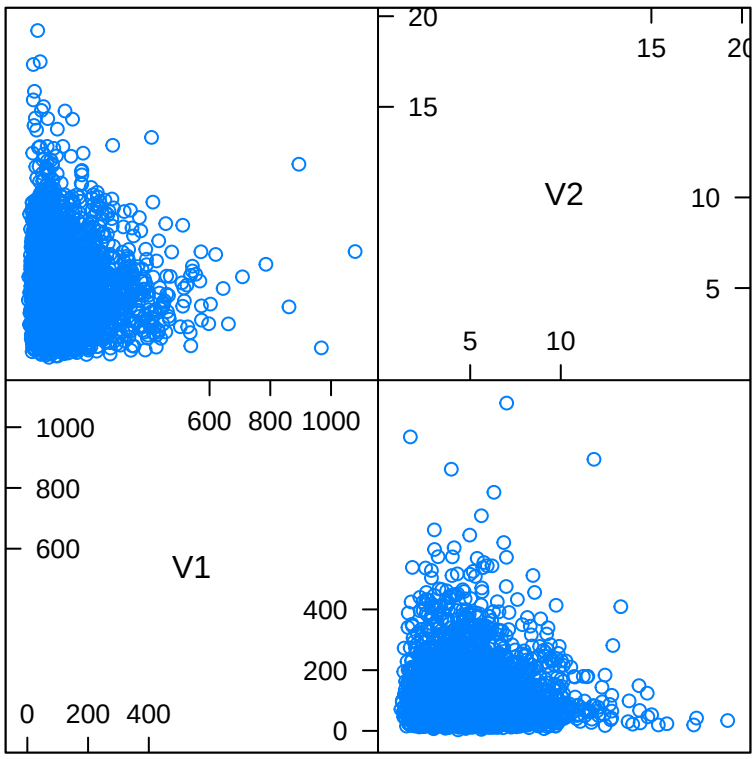
```r
hist(exp(MC_setup$par[["CODr"]][1,]), main = "Histogram CODr (AR1)", breaks=150)
```



Cross-correlation between input variables

```
y <- cbind(exp(MC_setup$par[["CODs"]][1,]), exp(MC_setup$par[["NH4s"]][1,]))
cor(y); splom(y)
```

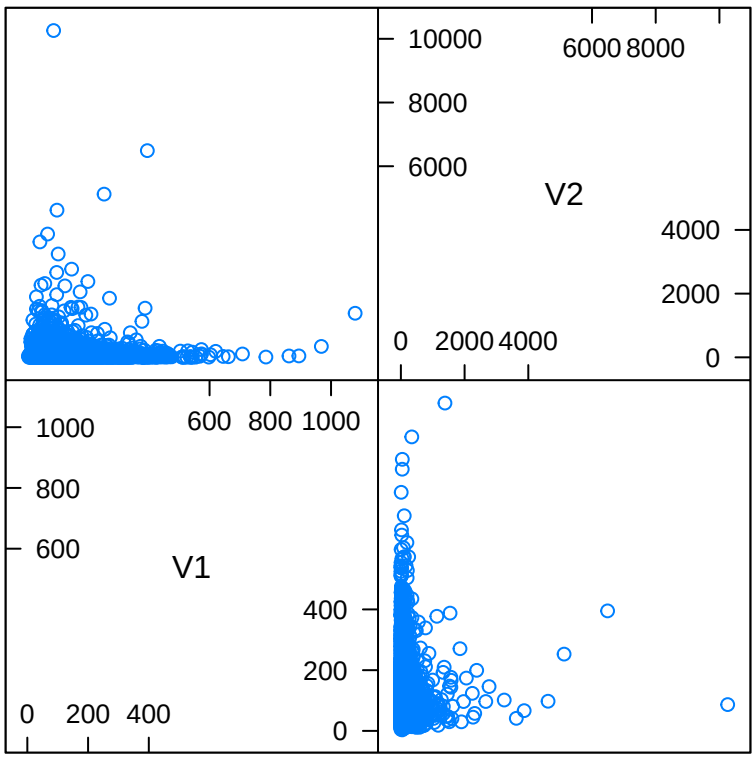
```
##              [,1]         [,2]
## [1,]  1.000000000 -0.004482916
## [2,] -0.004482916  1.000000000
```



Scatter Plot Matrix

```
y <- cbind(exp(MC_setup$par[["CODs"]][1,]), exp(MC_setup$par[["CODr"]][1,]))
cor(y); splom(y)
```

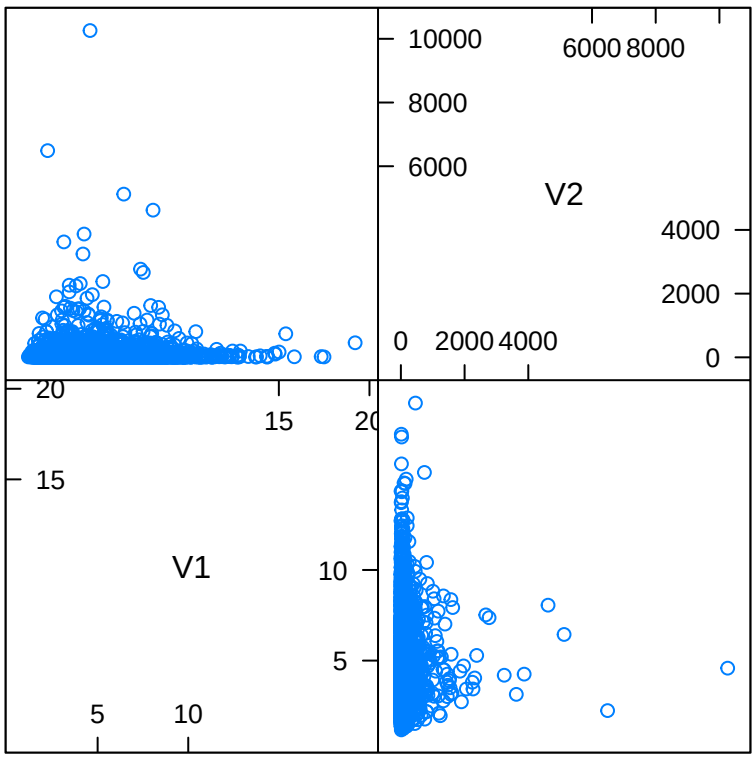
```
##             [,1]       [,2]
## [1,] 1.00000000 0.02787323
## [2,] 0.02787323 1.00000000
```



Scatter Plot Matrix

```
y <- cbind(exp(MC_setup$par[["NH4s"]][1,]), exp(MC_setup$par[["CODr"]][1,]))
cor(y); splom(y)
```

```
##              [,1]        [,2]
## [1,] 1.000000000 0.003420666
## [2,] 0.003420666 1.000000000
```



Scatter Plot Matrix

Deterministic simulation

Definition of structure 1, `E1`:

```
E1 <- list(id = 1, ns = MC_setup[["par"]]$ns, nm = MC_setup[["par"]]$nm,
           nc = MC_setup[["par"]]$nc, numc = MC_setup[["par"]]$numc,
           use = MC_setup[["par"]]$use, Ages = MC_setup[["par"]]$Ages,
           Ared = MC_setup[["par"]]$Ared, tfS = MC_setup[["par"]]$tfS,
           pe = MC_setup[["par"]]$pe, Qd = MC_setup[["par"]]$Qd,
           V = MC_setup[["par"]]$V)
```

Defining `input.user` object of class `input`:

```
input.user <- input(spatial = 0, zero = 1e-5,
                    folder = system.file("shiny", package = "EmiStatR"),
                    folderOutput = MC_setup[["folderOutput"]], cores = 0,
                    ww = list(qs = MC_setup[["par"]]$qs, CODs = 104,
                              NH4s = 4.7),
                    inf = list(qf= MC_setup[["par"]]$qf, CODf = MC_setup[["par"]]$CODf,
                               NH4f = MC_setup[["par"]]$NH4f),
                    rw = list(CODr = 107, NH4r = MC_setup[["par"]]$NH4r,
                              stat = "Esch_Sure2010"),
                    tf = MC_setup[["par"]]$tf, P1 = P1,
                    st = list(E1=E1),
                    export = 1)
```

Invoking `EmiStatR` with the deterministic input:

```
sim   <- EmiStatR(input.user)
```

```
## [1] "done, please check your output folder (used 1 of 4 cores, no parallel mode, cores=0)"
```

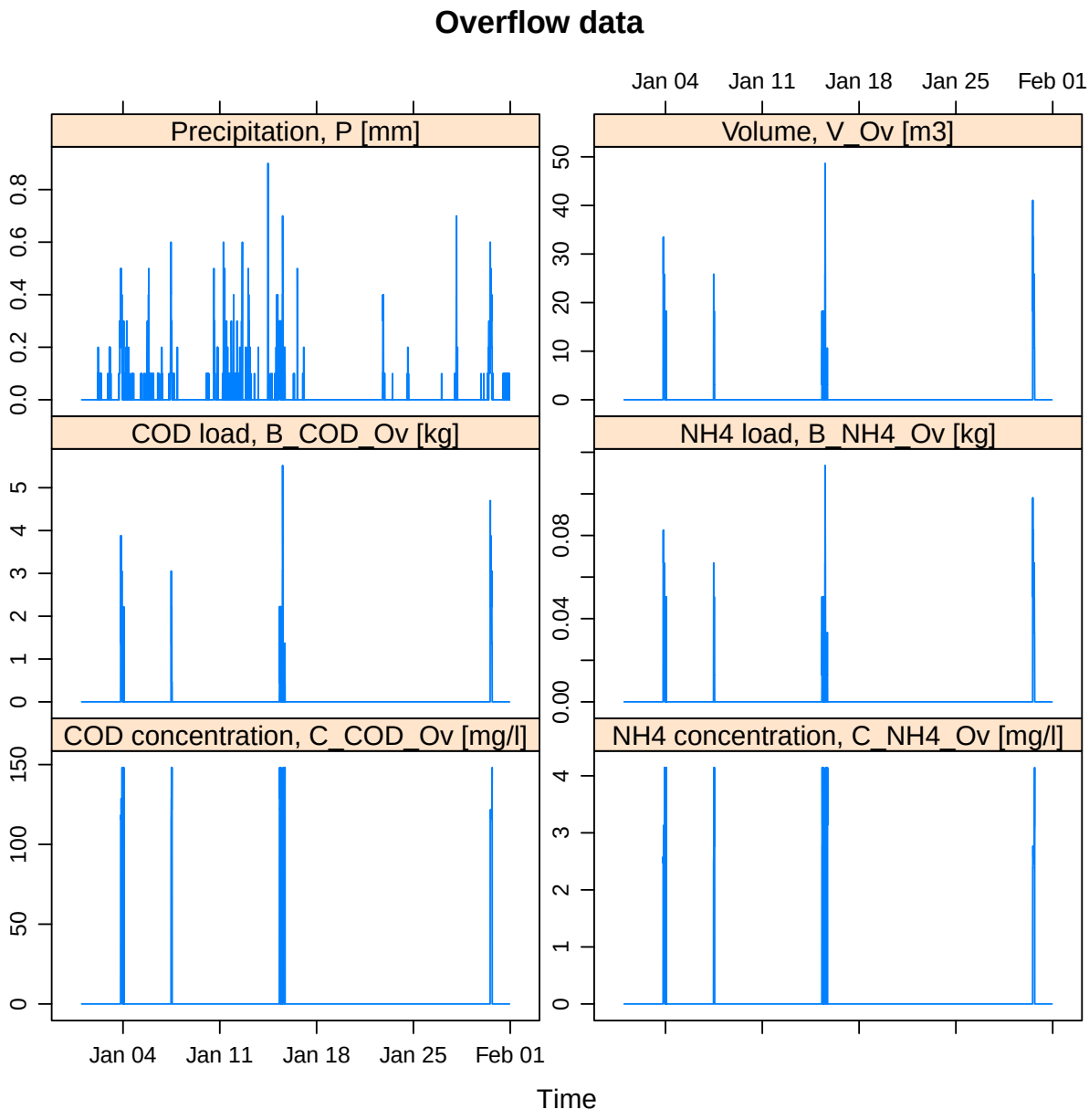


Figure 1: Deterministic simulation with `EmiStatR` given precipitation time series `P1`.

Monte Carlo simulation

```
sims  <- MC.sim(MC_setup, EmiStatR.cores = 0)
```

Monte Carlo simulation analysis

In order to run the analysis is necesary to define four additional variables. The first one is `delta` in minutes that corresponds to the level of aggregation required. Secondly, `qUpper`, the upper percentile to plot the confidence band of results, several options are possible `"q999"` the 99.9th percentile, `"q995"` the 99.5th percentile, `"q99"` the 99th percentile, `"q95"` the 95th percentile, `"q50"` the 50th percentile. The lower boundary of the confidence band is the 5th percentile. Thirdly, the initial time for event analysis `event.ini` in POSIXct format. Fourthly, the final time for event analysis `event.end` in POSIXct format.

```
delta <- 30 # minutes

qUpper <- "q999"

event.ini <- as.POSIXct("2016-01-14")
event.end <- as.POSIXct("2016-01-16 12:00:00")
```

Once the additional variables are setting-up, the function `MC.analysis` is invoked:

```
analysis <- MC.analysis(sims, delta, qUpper,  P1, sim, event.ini, event.end, ntick = 5)
```

```
## [1] "End MC analysis. Please check your output folder."
```

At the end of the Monte Carlo simulation analysis, 10 pdf files are created in the working directory, these files contain plots that summarise the results of analysis. Basically, each plot presents the mean value of the output variable as a blue line, and shows as a grey shadow the confidence band defined between the 5th percentile and the upper percentile define in the variable `qUpper`, in this case the upper percentile is 99.9.

The 10 plots are shown below.

Volume in the CSO tank, `VTank`

As is expected the volume in the combined sewer overflow (CSO) tank is represented as the mean value of the simulations. No confidence band is shown because there is not uncertain in the simulation of volume (the precipitation input is the same for all the simulations).

Overflow volume, `Vov`

Similarly to the volume in the tank, the overflow volume or CSO volume is represented as the mean value of the simulations equal to the deterministic run. Also no confidence band is shown given no uncertain in this variable.

COD load in the overflow, `BCOD`

Due to the fact that the chemical oxygen demand (COD) in the overflow was simulated as a AR1 process, BCOD has a perceptible confidence band.

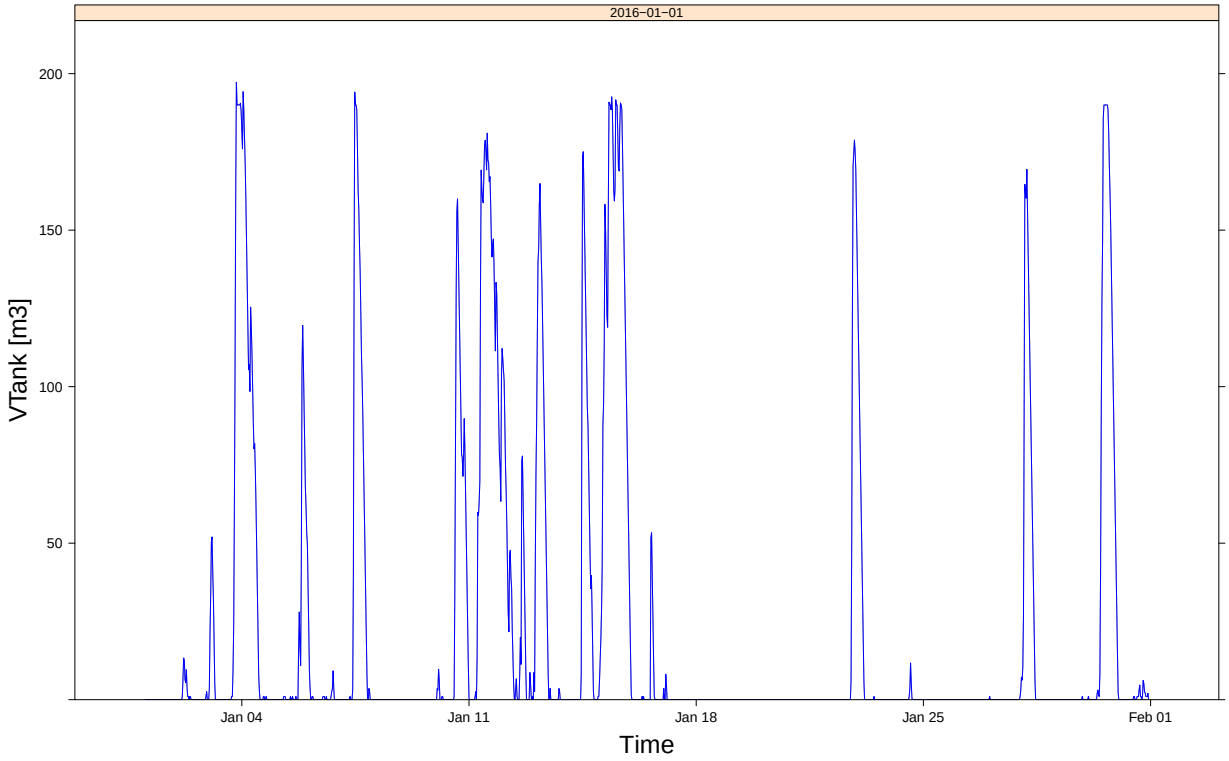


Figure 2: Volume in the CSO tank.

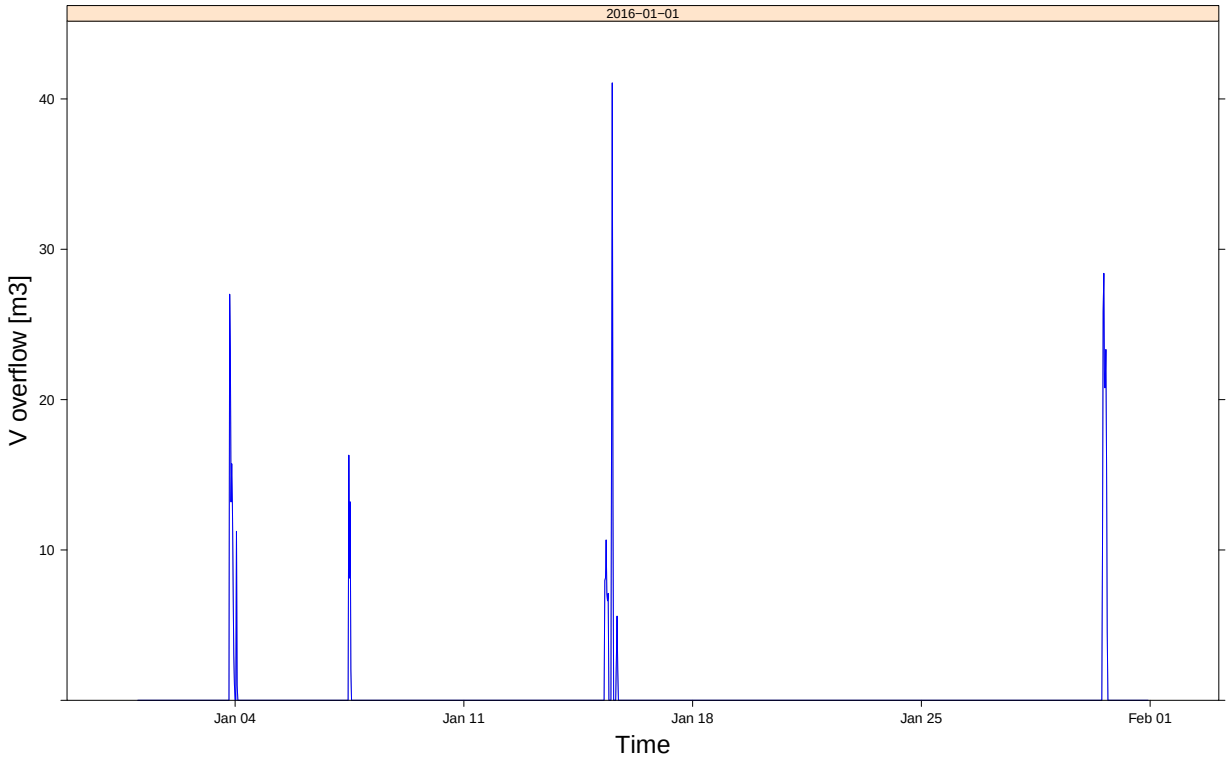


Figure 3: Overflow volume.

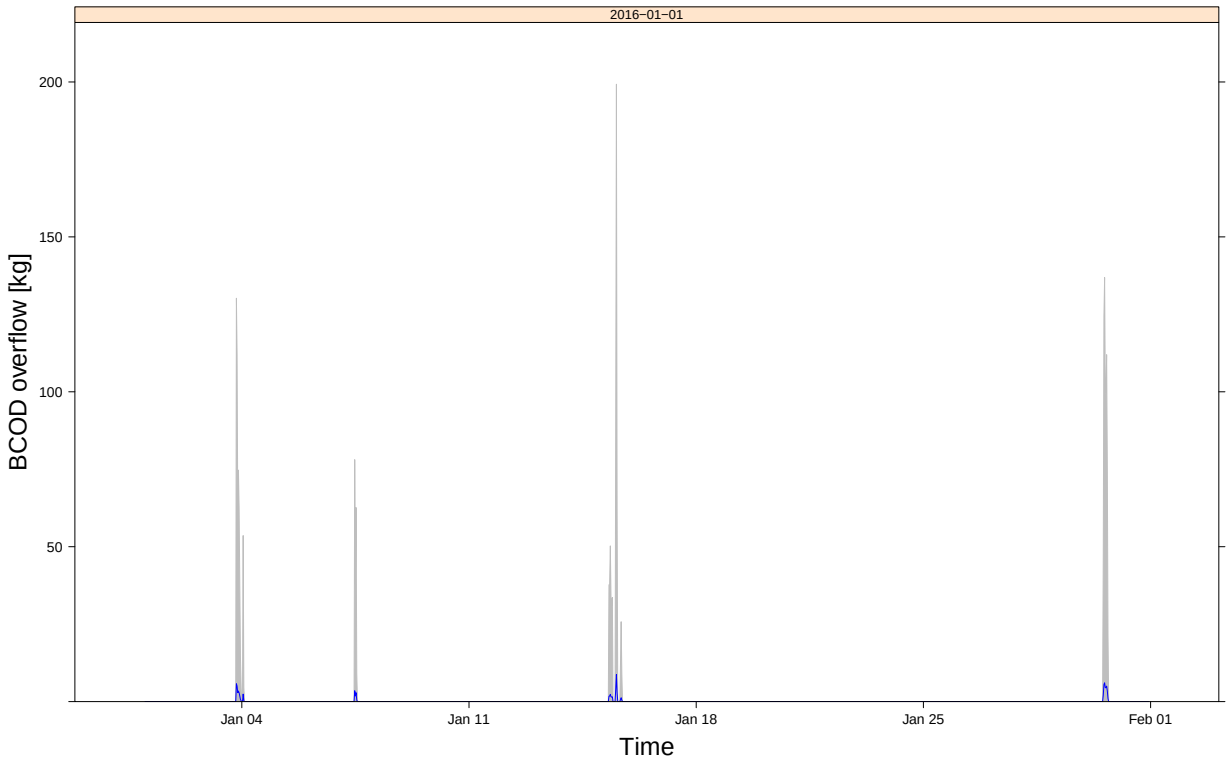


Figure 4: COD load in the overflow.

COD concentration in the overflow, `CCOD`

Given the direct relationship between `BCOD` and `CCOD`, `CCOD` has a perceptible confidence band as well.

NH4 load in the overflow, `BNH4`

Similarly to COD, ammonium (NH4) in the overflow was simulated as a AR1 process, therefore BNH4 has a perceptible confidence band.

NH4 concentration in the overflow

Given the direct relationship between `BNH4` and `CNH4`, `CNH4` has a perceptible confidence band as well.

Event based analysis

Besides the plots for the entire month of January 2016, a event based analysis is performed in a time window from `event.ini` to `event.end`. Therefore, one plot is produced for Volume in the CSO tank and overflow volume.

The same time window is applied to analyse `BCOD` and `CCOD` and one plot is produced.

Similarly, the time window is applied to plot `BNH4` and `CNH4`.

Finallly, one plot is produced to illustrate the concentrations of COD and NH4, `CCOD` and `CNH4`.

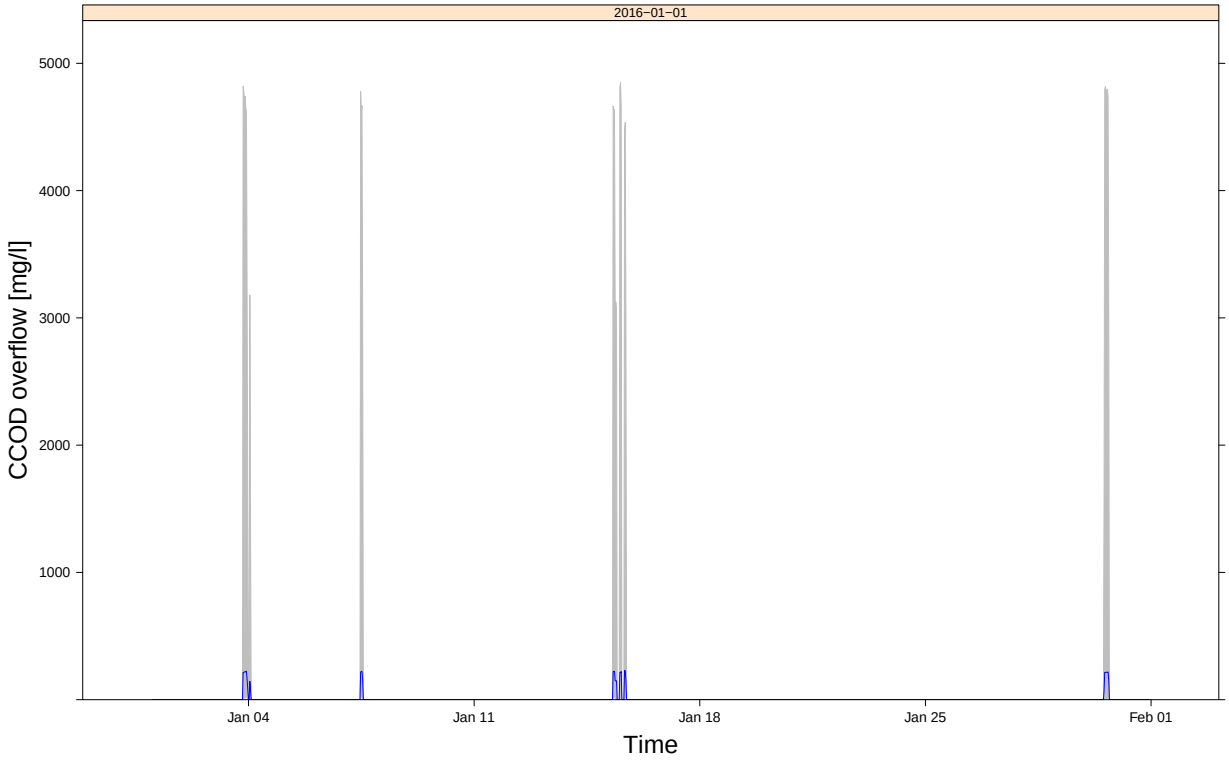


Figure 5: COD concentration in the overflow.

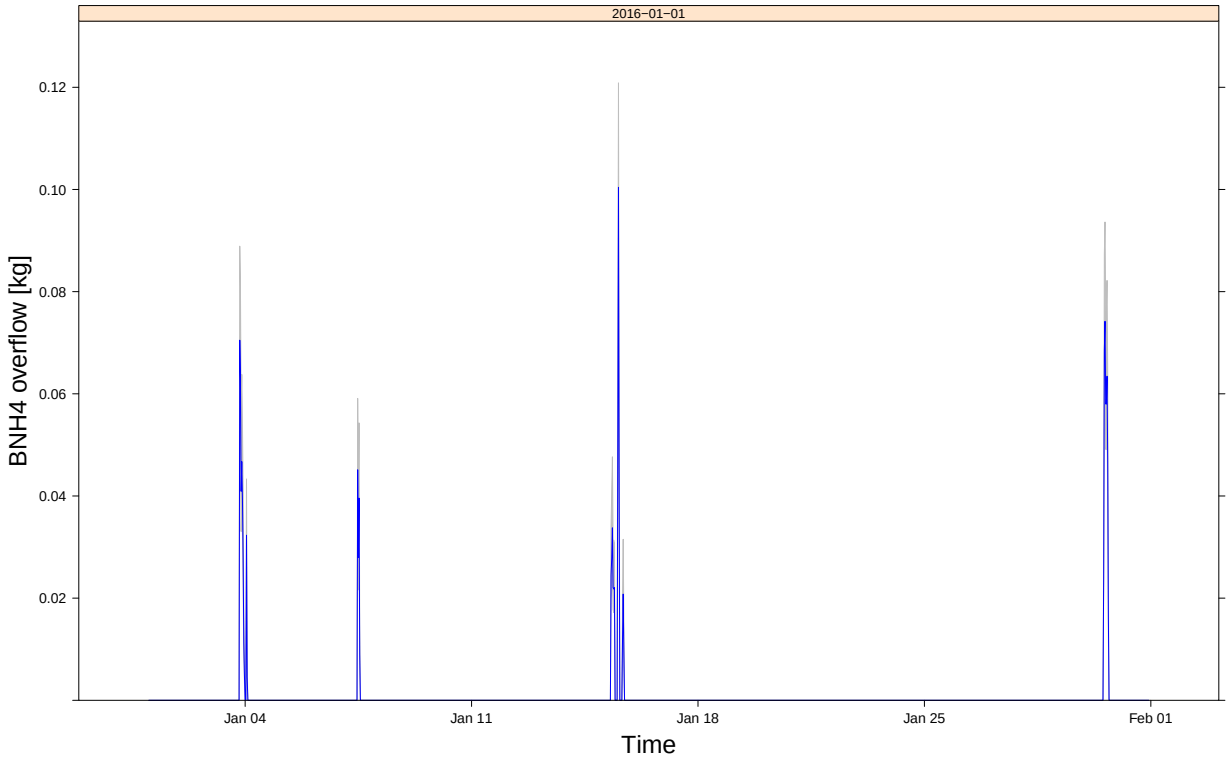


Figure 6: NH4 load in the overflow.

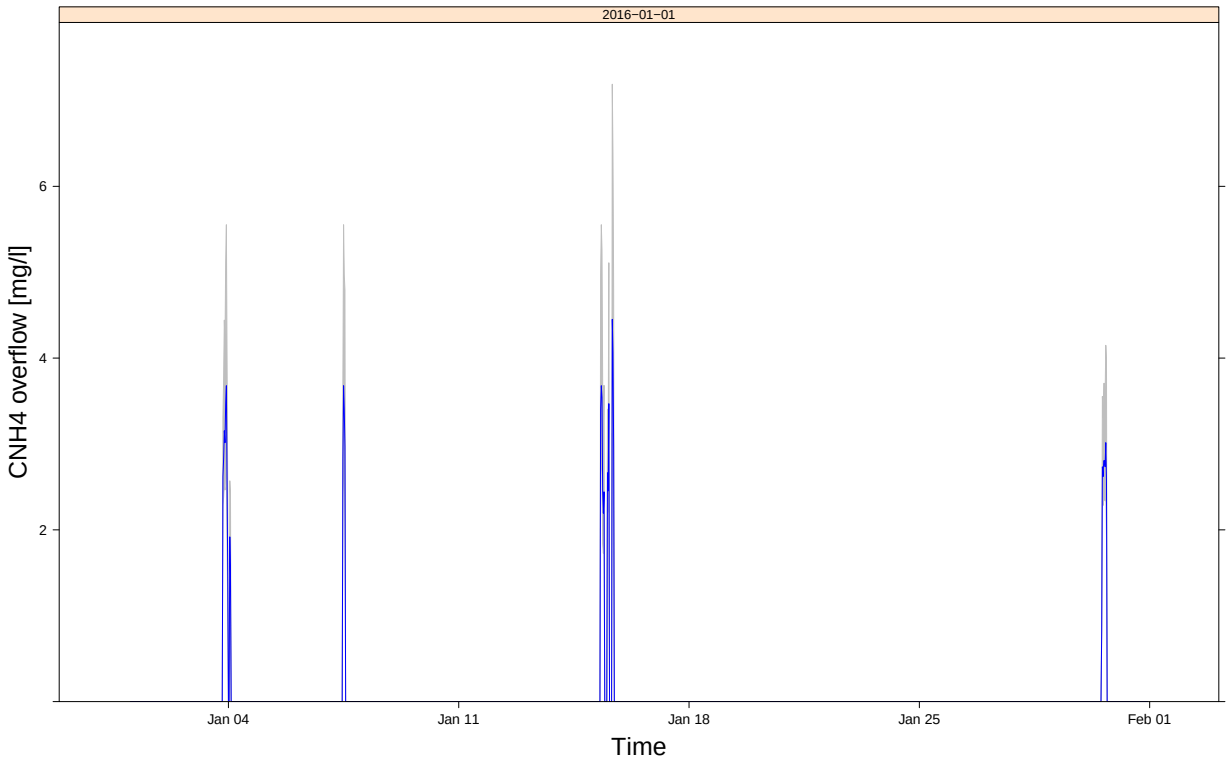


Figure 7: NH4 concentration in the overflow.

```
# library(rmarkdown)
# rmarkdown::render("UsersGuide.R", pdf_document())
```

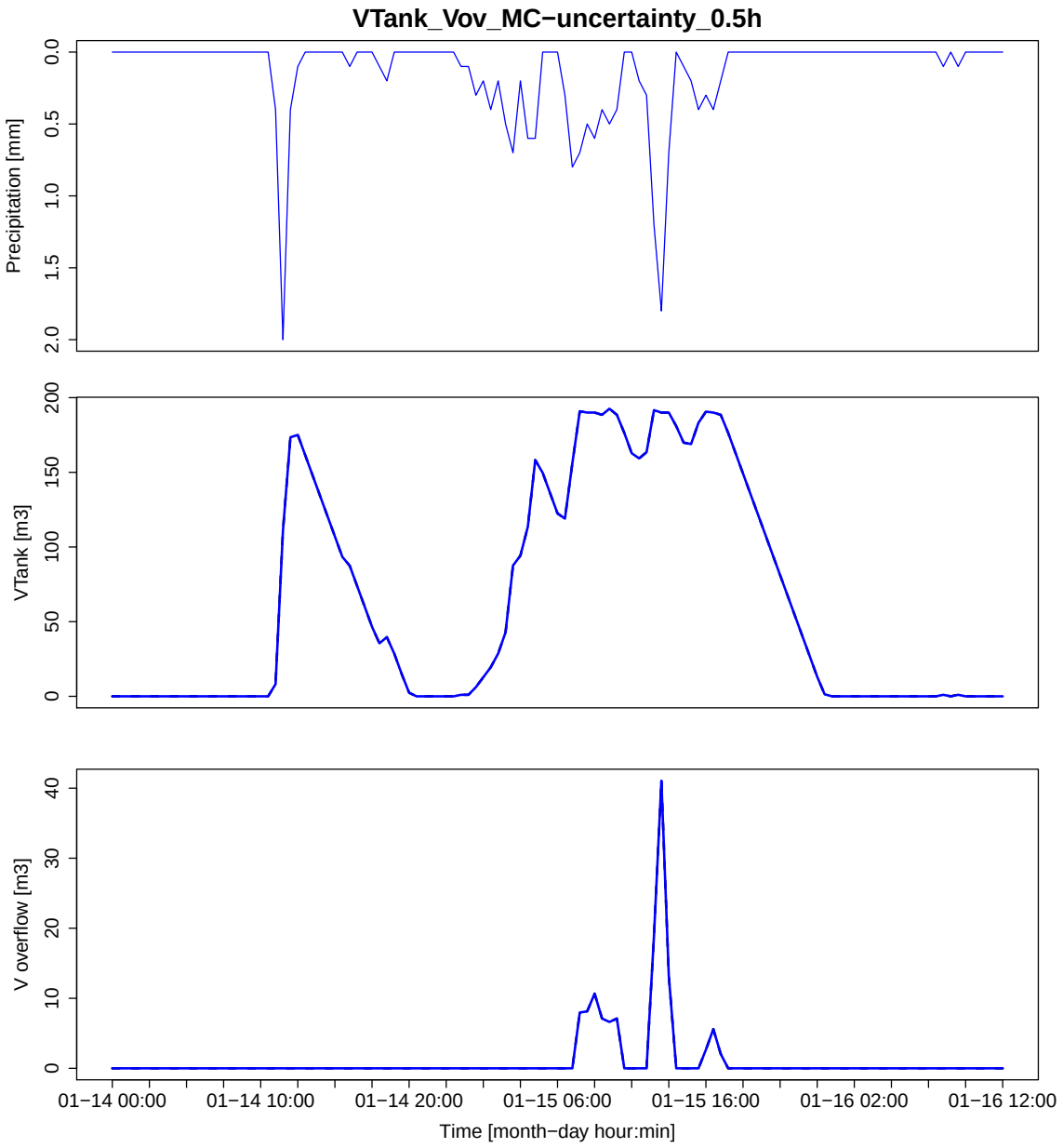


Figure 8: Precipitation, volume in the CSO tank and overflow, event.

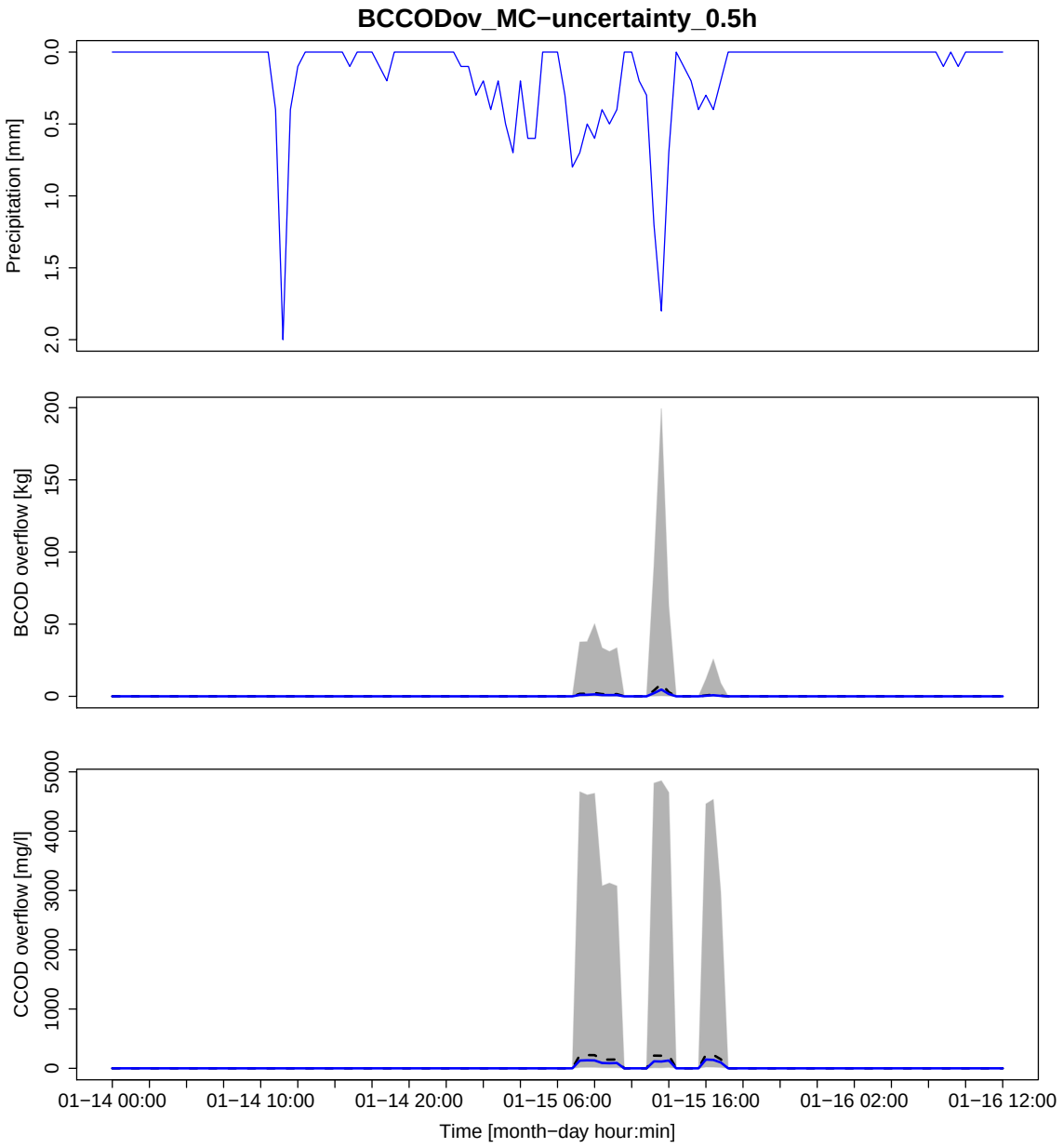


Figure 9: Precipitation, and load and concentration of COD, event.

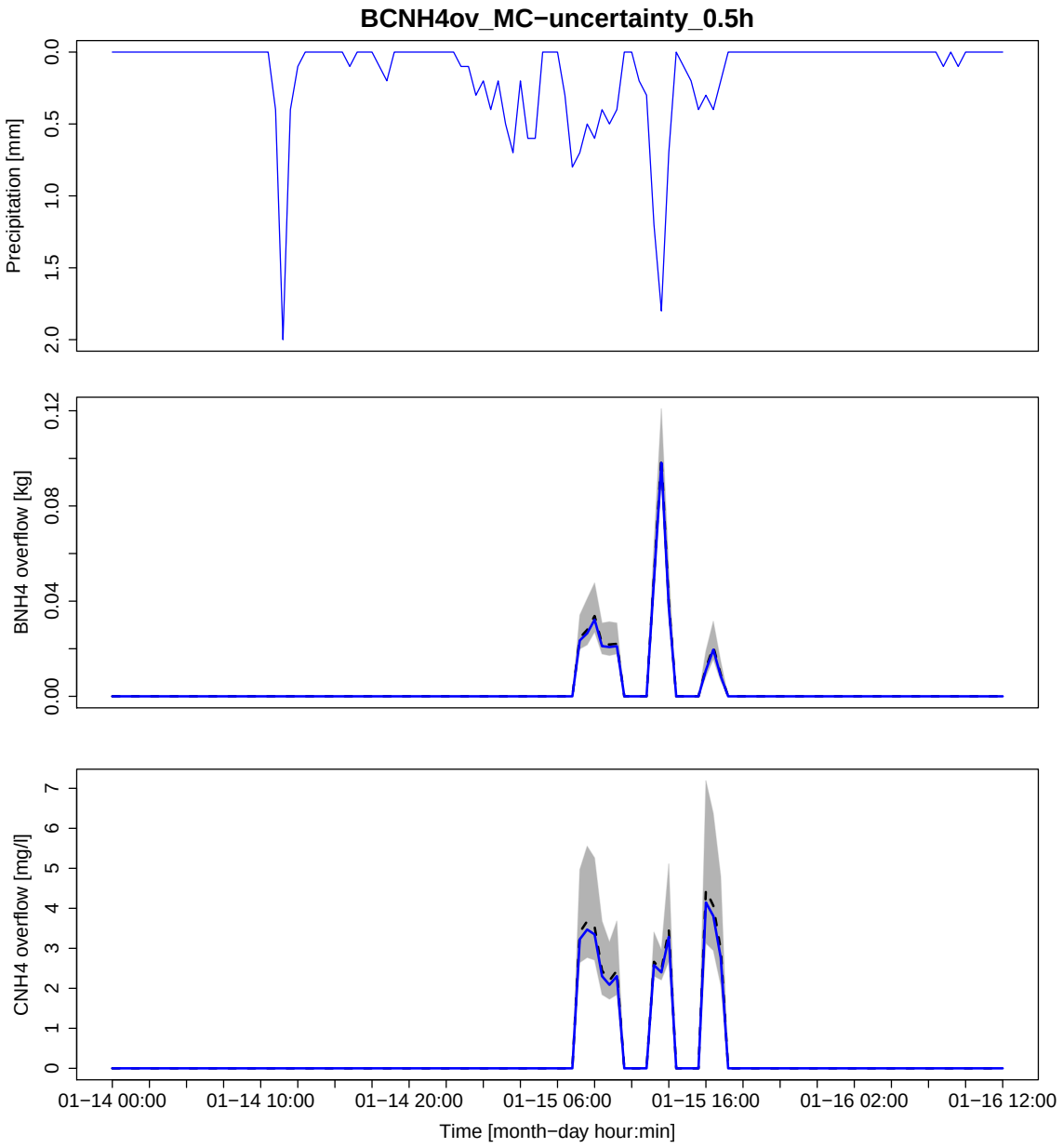


Figure 10: Precipitation, and load and concentration of NH4, event.

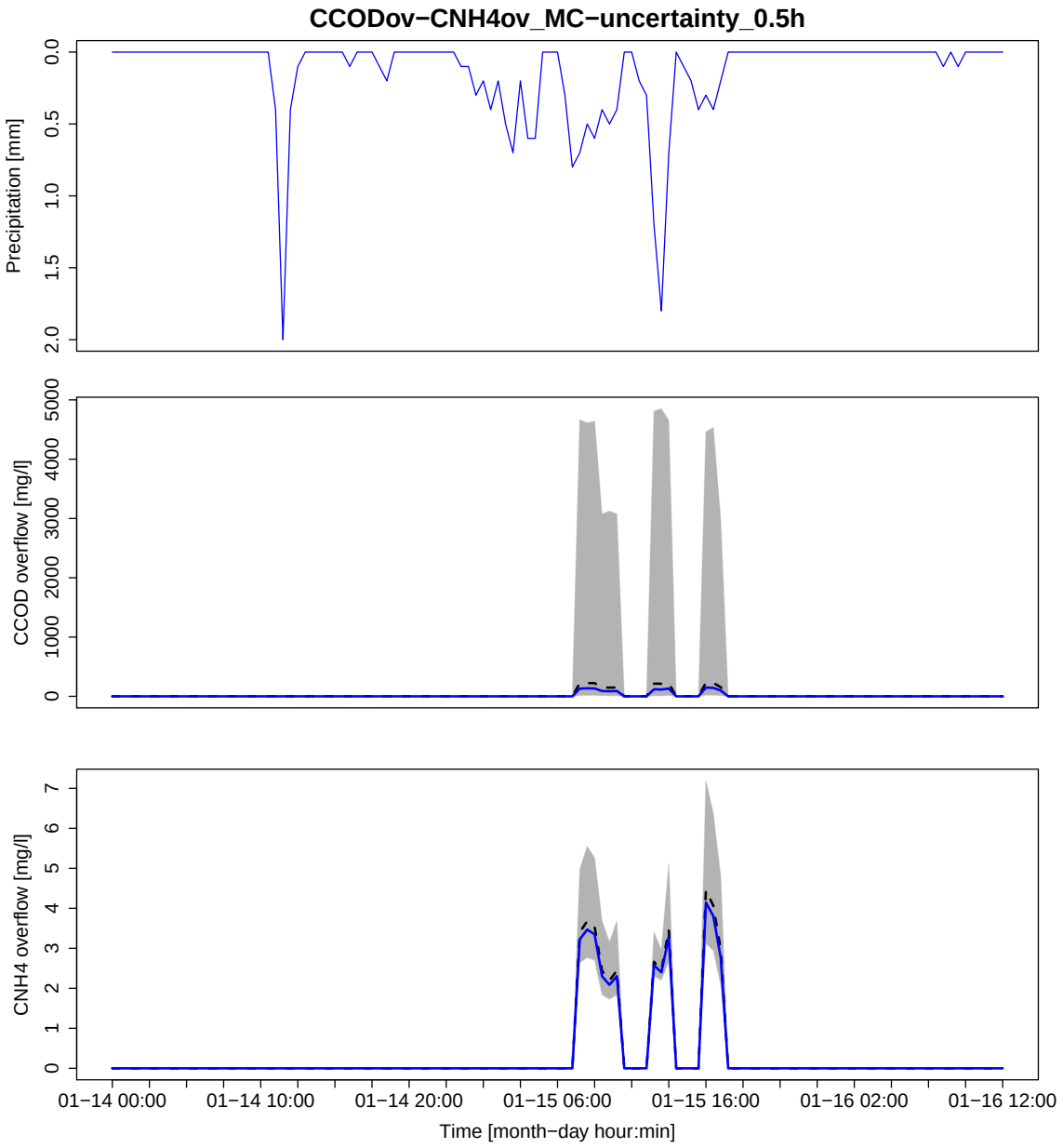


Figure 11: Precipitation, and COD and NH4 concentrations, event.

References

S. M. Barbosa. *Package "mAr": Multivariate AutoRegressive analysis*. The Comprehensive R Archive Network, CRAN, 1.1-2 edition, February 2015.

J.M. Hammersley and D.C. Handscomb. *Monte Carlo Methods*. Methuen & Co Ltd, London, 1964.

G. B. M. Heuvelink, J. D. Brown, and E. E. van Loon. A probabilistic framework for representing and simulating uncertain environmental variables. *International Journal of Geographical Information Science*, 21(5):497–513, 2007. ISSN 1365-8816. doi: 10.1080/13658810601063951.

J. R. M. Hosking. *Package 'lmom': L-moments*. The Comprehensive R Archive Network, CRAN, 1.6 edition, 2012.

J.R.M. Hosking. L-moments: Analysis and estimation of distributions using linear combinations of order statistics. *Journal of the Royal Statistical Society*, pages 105–124, 1990.

J.R.M. Hosking and J.R. Wallis. *Regional frequency analysis-an approach based on L-moments*. Cambridge University Press, 1997. URL `http://books.google.de/books/about/Regional`$_F$`requency`$_A$`nalysis.html?id=`$gurAnfB4nvUCredir_esc=y$.

Malvin H. Kalos and Paula A. Whitlock. *Monte Carlo Methods*. Wiley-Blackwell, 2 edition, 2008.

Helmut Lütkepohl. *New Introduction to Multiple Time Series Analysis*. Springer, 2005.

L. Nol, G. B M Heuvelink, a. Veldkamp, W. de Vries, and J. Kros. Uncertainty propagation analysis of an N2O emission model at the plot and landscape scale. *Geoderma*, 159(1-2):9–23, 2010. ISSN 00167061. doi: 10.1016/j.geoderma.2010.06.009. URL `http://dx.doi.org/10.1016/j.geoderma.2010.06.009`.