

Supplementary Materials

2024-09-02

This document contains supplementary materials to the research article “Climate Change Drives Hydrological Decoupling in Deciduous Forests” by Simon Drollinger, Michael Dietze, Dominik Seidel, Daniel Schwindt, Jago Birk and Daniela Sauer.

The markdown file documents the full analysis chain subjected to the data set. It requires the R package ebergoetzen (<https://github.com/coffeemuggler/ebergoetzen>). The analysis is based on a parameter list object that contains all the relevant control parameters. The Supplementary Materials include all event specific data sets.

Load required packages

```
library(ebergoetzen)
library(fields)

## Loading required package: spam

## Spam version 2.10-0 (2023-10-23) is loaded.
## Type 'help( Spam)' or 'demo( spam)' for a short introduction
## and overview of this package.
## Help for individual functions is also obtained by adding the
## suffix '.spam' to the function name, e.g. 'help( chol.spam)'.

##
## Attaching package: 'spam'

## The following objects are masked from 'package:base':
##
##      backsolve, forwardsolve

## Loading required package: viridisLite

##
## Try help(fields) to get started.
```

Helper function to group data into clusters

For convenience and minimal code line usage, a helper function is defined, to classify continuous event data into groups of data. The breaks that define the groups must be given beyond the actual data range and must include both the lower and the upper class breaks. Hence, for a data set 1:10, break definitions must be for example `c(0, 5, 11)`. Alternatively, also quantiles can be provided as class limit definitions, but those also must be exclusive, e.g. `c(0, 0.5, 1)`.

```
## define helper function
classify <- function(x, y, quantiles, breaks, list = FALSE) {
```

```

## define breaks
if(missing(breaks) == TRUE) {
  brks <- quantile(x = x, probs = quantiles)
} else {
  brks <- breaks
}

## classify data by breaks
i_grp <- cut(x = x,
             breaks = brks,
             labels = as.character(2:length(brks)),
             include.lowest = TRUE,
             right = FALSE)

## classify y into list
y_list <- lapply(X = levels(i_grp), FUN = function(grp, i_grp, y) {

  z <- y[grp == i_grp]

  if(length(z) == 0) {

    z <- NA
  }

  return(z)
}, i_grp, y)
names(y_list) <- brks[-1]

## optionally convert to data frame, padded with NAs
if(list == FALSE) {

  ## get number of samples per class
  n_class <- sapply(X = y_list, FUN = length)

  ## get maximum number of samples per class
  n_max <- max(n_class)

  ## create data frame
  y_df <- matrix(data = rep(NA, length(y_list) * n_max), nrow = n_max)

  ## fill data frame
  for(i in 1:ncol(y_df)) {

    y_df[1:n_class[i], i] <- y_list[[i]]
  }

  ## provide col names
  colnames(y_df) <- names(y_list)

  ## rename output

```

```

    y_list <- y_df
  }

  ## return output
  return(y_list)
}

```

Import of required data sets

The rda file contains five R objects; the full rain event information (1) `event_info`, a list with information on start, duration, rain sum, maximum rain intensity, average rain intensity, the throughfall sums from all valid samplers, the corresponding throughfall ratios, and the throughfall values of all samplers including the non-valid ones, (2) `events`, a data frame with start, stop, duration, rain sum, maximum rain intensity and mean rain intensity for all events, (3) `t1s`, a data frame with laser scan derived information of canopy openness above each throughfall sampler once for summer and once for winter, (4) `p_mnth`, a numeric vector of monthly rain sums from the open sky meteo station, and (5) `t_mnth` a numeric vector of the monthly year fractions corresponding to the monthly rain sums.

```
load(file = "base_data_ebergoetzen.rda")
```

Definition of analysis parameters

The following list object contains the parameters used in the study

```

par <- list(

  info = list(

    time_full = c("2017-09-01", "2024-09-07"), # analysis interval
    tf_sampler_ok = c(2, 12), # valid ordered range of throughfall samplers
    event_duration_max = 7 * 24 * 3600, # maximum rain event duration
    tfr_dur_max = 5, # maximum allowed throughfall ratio
    summer = c(5, 9), # summer months
    winter = c(12, 3) # winter months
  ),

  event = list( # definition of rain events

    intensity_min = 0.2,
    duration_min = 0.25 * 3600,
    amount_min = 1,
    pause_min = 4 * 3600,
    lagtime = 1 * 3600
  ),

  breaks = list( # definition of clusters for boxplots (Fig. 4)

    duration = c(0.50, 1, 2, 4, 8, 32) * 3600,
    intensity = c(0, 0.1, 0.15, 0.2, 0.3, 0.4, 2),

```

```

    amount = c(1.0, 1.5, 2.0, 5.0, 10, 55)
  )
)

```

Operation status of samplers and removal of inappropriate data

extract throughfall ratios for each event

```

tfr_na <- do.call(rbind, lapply(X = event_info, FUN = function(x) {
  x$tfr
})))

```

extract event timing

```

t_event <- do.call(c, lapply(X = event_info, FUN = function(x) {x$start}))

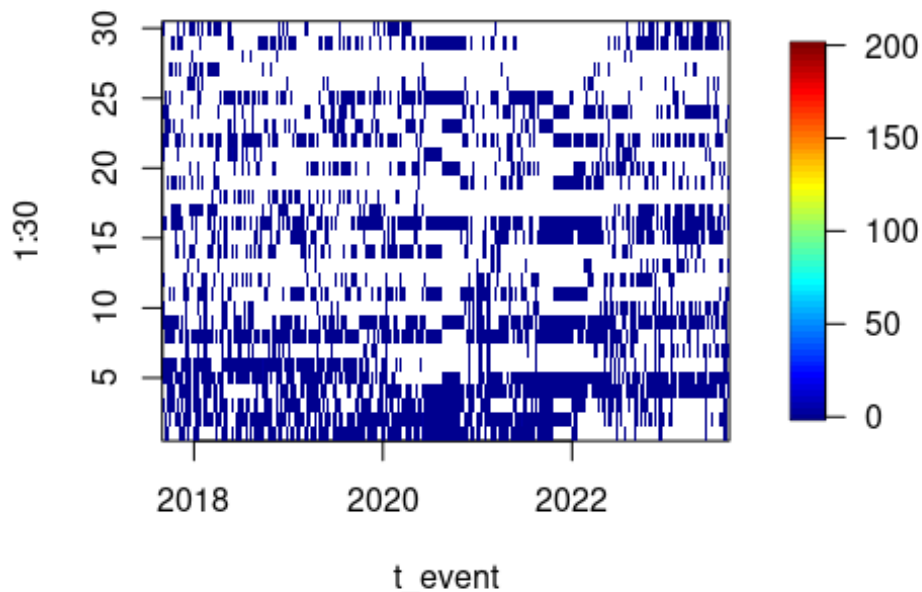
```

plot throughfall sampler operation status

```

image.plot(x = t_event, y = 1:30, z = tfr_na, zlim = c(0, 200))

```



Definition of seasons

define seasons

```

month <- as.numeric(format(events$start, "%m"))
season <- rep(NA, nrow(tfr_na))
season[month >= par$info$summer[1] & month <= par$info$summer[2]] <- "summer"
season[month >= par$info$winter[1] | month <= par$info$winter[2]] <- "winter"

col_season <- rep("grey", length(season))
col_season[season == "summer"] <- "orange"
col_season[season == "winter"] <- "lightblue"

```

Relationship of throughfall ratios with rain duration

get median TFR and duration of each event

```
tfr_dur <- lapply(X = event_info, FUN = function(x) {  
  
  return(data.frame(tfr = median(x$tfr, na.rm = TRUE),  
                      dur = x$duration))  
})  
tfr_dur <- do.call(rbind, tfr_dur)
```

get median TFR and rain sum of each event

```
tfr_sum <- lapply(X = event_info, FUN = function(x) {  
  
  return(data.frame(tfr = median(x$tfr, na.rm = TRUE),  
                    sum = x$sum))  
})  
tfr_sum <- do.call(rbind, tfr_sum)
```

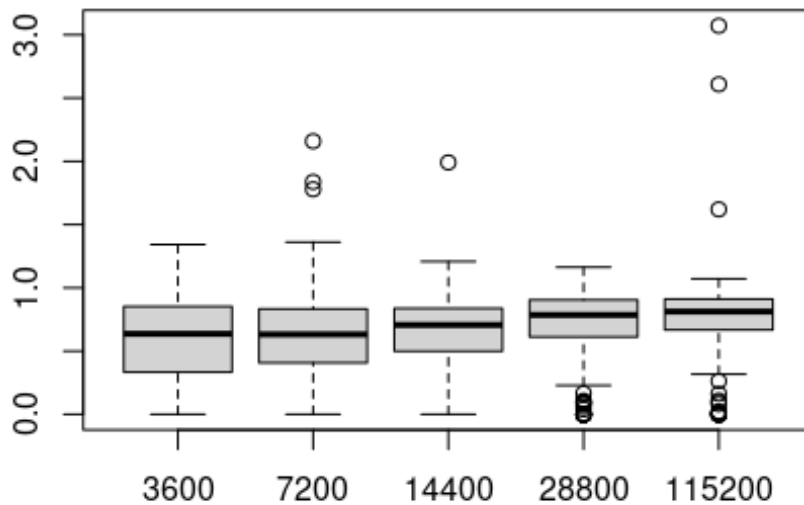
classify data by duration

```
tfr_classified <- classify(x = tfr_dur$dur,  
                          y = tfr_dur$tfr,  
                          breaks = par$breaks$duration,  
                          list = FALSE)
```

get number of events per class

```
tfr_classified_n <- apply(tfr_classified, MARGIN = 2, function(x) {  
  sum(!is.na(x))  
})
```

```
boxplot(tfr_classified)
```



Relationship of throughfall ratios with rain intensity

get median TFR and mean intensity of each event

```
tfr_int <- do.call(rbind, lapply(X = event_info, FUN = function(x, par) {

  return(data.frame(tfr = median(x$tfr, na.rm = TRUE),
                    int = x$avg,
                    win = as.numeric(format(x$start, "%m")) >=
par$info$winter[1] |
                    as.numeric(format(x$start, "%m")) <=
par$info$winter[2]))
}, par))
```

classify data by intensity

```
tfr_classified <- classify(x = tfr_int$int,
                         y = tfr_int$tfr,
                         breaks = par$breaks$intensity,
                         list = FALSE)
```

get number of events per class

```
tfr_classified_n <- apply(tfr_classified, MARGIN = 2, function(x) {
  sum(!is.na(x))
})
```

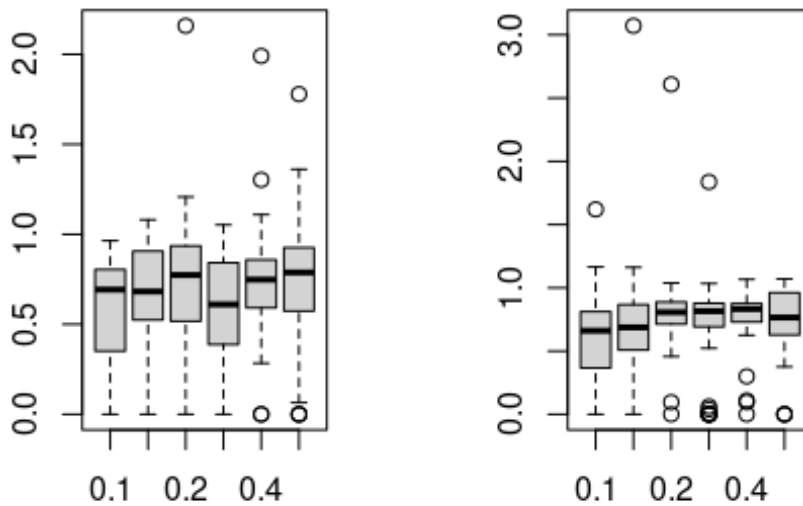
separate by season

```
tfr_int_sum <- tfr_int[tfr_int$win == FALSE,]
tfr_int_win <- tfr_int[tfr_int$win == TRUE,]
```

```
## classify data by duration
```

```
tfr_int_sum_cls <- classify(x = tfr_int_sum$int,
                           y = tfr_int_sum$tfr,
                           breaks = par$breaks$intensity,
                           list = FALSE)
tfr_int_win_cls <- classify(x = tfr_int_win$int,
                           y = tfr_int_win$tfr,
                           breaks = par$breaks$intensity,
                           list = FALSE)
```

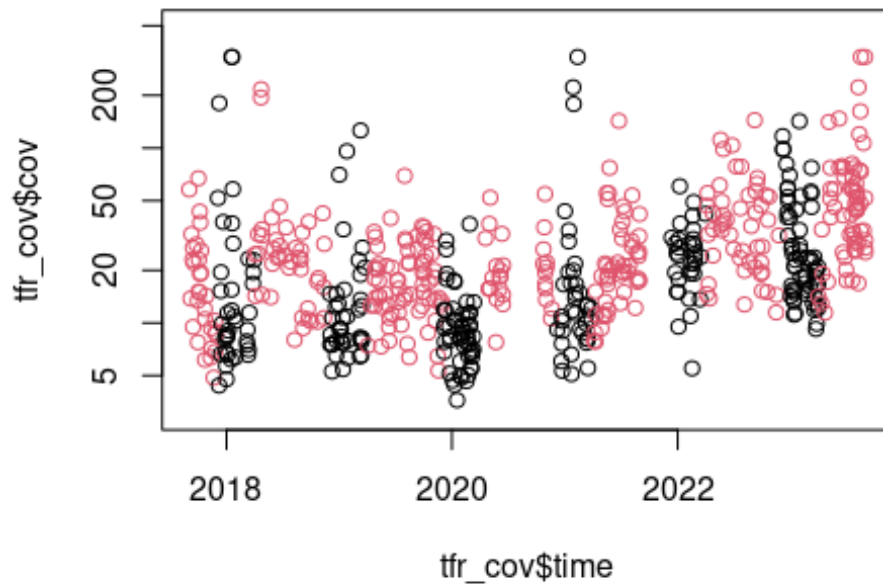
```
par(mfcol = c(1, 2))
boxplot(tfr_int_sum_cls)
boxplot(tfr_int_win_cls)
```



Throughfall coefficient of variation

```
tfr_cov <- do.call(rbind, lapply(X = event_info, FUN = function(x, par) {
  data.frame(time = x$start,
             cov = sd(x$tfr, na.rm = TRUE) / mean(x$tfr, na.rm = TRUE) * 100,
             mon = as.numeric(format(x$start, "%m")),
             win = as.numeric(format(x$start, "%m")) >= par$info$winter[1] |
                   as.numeric(format(x$start, "%m")) <=
par$info$winter[2])
}, par))

plot(tfr_cov$time, tfr_cov$cov, log = "y", ylim = c(3, 500), col = 2 -
tfr_cov$win)
```



Effect of vegetation density (canopy openness)

extract TF ratios of each sampler for each event

```
tfr_co <- do.call(rbind, lapply(X = event_info, FUN = function(x, par) {

  tf <- x$tfr

  data.frame(time = x$start,
             t(tf),
             mon = as.numeric(format(x$start, "%m")),
             win = as.numeric(format(x$start, "%m")) >= par$info$winter[1] |
                   as.numeric(format(x$start, "%m")) <=
par$info$winter[2])
}, par))
```

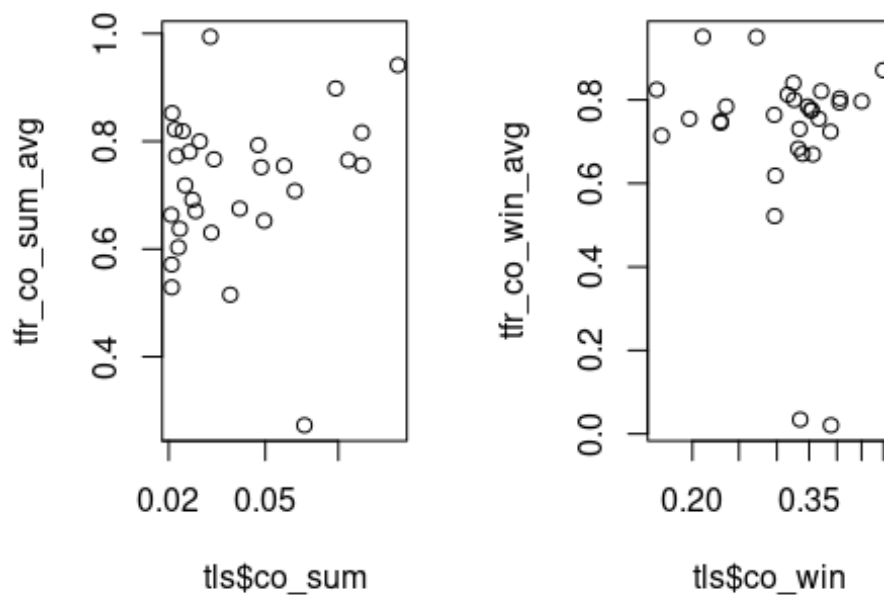
average TFRs by season

```
tfr_co_sum <- tfr_co[tfr_co$win == FALSE,]
tfr_co_win <- tfr_co[tfr_co$win == TRUE,]

tfr_co_sum_avg <- apply(X = tfr_co_sum[,2:31], MARGIN = 2, FUN = median,
na.rm = TRUE)
tfr_co_win_avg <- apply(X = tfr_co_win[,2:31], MARGIN = 2, FUN = median,
na.rm = TRUE)
```

plot CO-TFR data

```
par(mfcol = c(1, 2))
plot(x = tls$co_sum, y = tfr_co_sum_avg, log = "x")
plot(x = tls$co_win, y = tfr_co_win_avg, log = "x")
```

Figure

classify data by duration

```
tfr_dur_cls <- classify(x = tfr_dur$dur,
                      y = tfr_dur$tfr,
                      breaks = par$breaks$duration,
                      list = FALSE)
colnames(tfr_dur_cls) = as.numeric(colnames(tfr_dur_cls)) / 3600
```

```
n_tfr_dur <- nrow(tfr_dur_cls) -
  apply(X = tfr_dur_cls,
        MARGIN = 2,
        FUN = function(x) {
          sum(is.na(x))
        })
```

classify data by intensity

```
tfr_int_cls <- classify(x = tfr_int$int,
                      y = tfr_int$tfr,
                      breaks = par$breaks$intensity,
                      list = FALSE)
colnames(tfr_int_cls) = as.numeric(colnames(tfr_int_cls))
```

```
n_tfr_int <- nrow(tfr_int_cls) -
  apply(X = tfr_int_cls,
        MARGIN = 2,
        FUN = function(x) {
```

```

    sum(is.na(x))
})

## classify data by rain sum
tfr_sum_cls <- classify(x = tfr_sum$sum,
                      y = tfr_sum$tfr,
                      breaks = par$breaks$amount,
                      list = FALSE)
colnames(tfr_sum_cls) = as.numeric(colnames(tfr_sum_cls))

n_tfr_sum <- nrow(tfr_sum_cls) -
  apply(X = tfr_sum_cls,
        MARGIN = 2,
        FUN = function(x) {

    sum(is.na(x))
  })

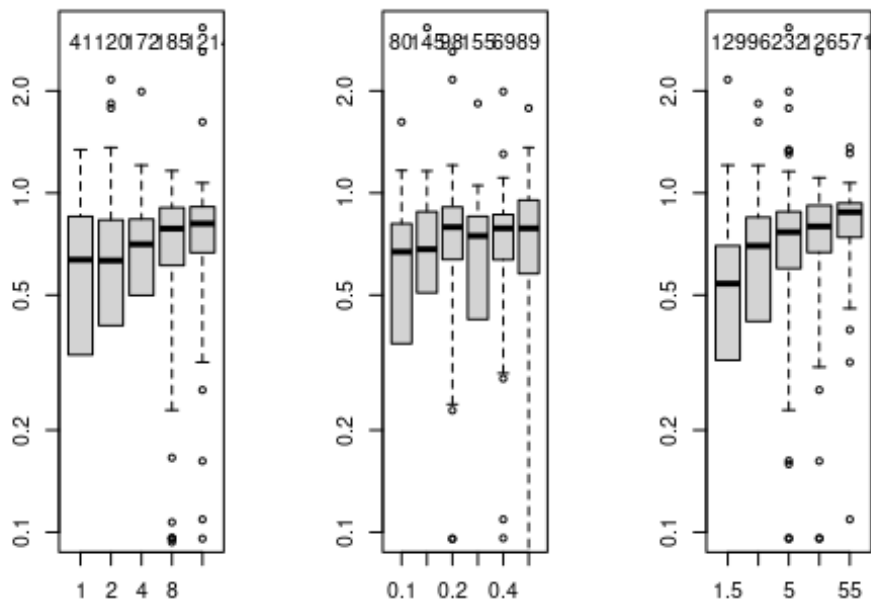
par(mfcol = c(1, 3))

boxplot(tfr_dur_cls, log = "y", ylim = c(0.1, 3))
text(x = 1:6, y = rep(2.8, 5), labels = n_tfr_dur,
     xlab = "Rain event duration (hours)")

boxplot(tfr_int_cls, log = "y", ylim = c(0.1, 3))
text(x = 1:6, y = rep(2.8, 6), labels = n_tfr_int,
     xlab = "Rain event intensity (mm/h)")

boxplot(tfr_sum_cls, log = "y", ylim = c(0.1, 3))
text(x = 1:6, y = rep(2.8, 6), labels = n_tfr_sum,
     xlab = "Rain event sum (mm)")

```



```
apply(X = tfr_dur_cls, MARGIN = 2, FUN = median, na.rm = TRUE)
```

```
##           1           2           4           8           32
## 0.6358202 0.6320027 0.7061724 0.7853403 0.8124373
```

```
apply(X = tfr_dur_cls, MARGIN = 2, FUN = function(x) {
  quantile(x = x, probs = 0.75, na.rm = TRUE) -
  quantile(x = x, probs = 0.25, na.rm = TRUE)
})
```

```
##           1           2           4           8           32
## 0.5193798 0.4249590 0.3380685 0.2932918 0.2441331
```

```
apply(X = tfr_int_cls, MARGIN = 2, FUN = median, na.rm = TRUE)
```

```
##           0.1           0.15           0.2           0.3           0.4           2
## 0.6705902 0.6824588 0.7936132 0.7475083 0.7865169 0.7870370
```

```
apply(X = tfr_dur_cls, MARGIN = 2, FUN = function(x) {
  quantile(x = x, probs = 0.75, na.rm = TRUE) -
  quantile(x = x, probs = 0.25, na.rm = TRUE)
})
```

```
##           1           2           4           8           32
## 0.5193798 0.4249590 0.3380685 0.2932918 0.2441331
```

Figure 2 of the main publication

```
## convert event start time stamps to running month number
mnth <- as.numeric(events$start) / 3600 / 24 / 30.5
```

```

mnth <- mnth - min(mnth)

## define point size for plots based on rain intensity
xy <- data.frame(x = mnth, y = tfr_int$tfr)
m <- lm(formula = y ~ x, data = xy)
size <- log(tfr_sum$sum) / max(log(tfr_sum$sum)) * 2 + 0.5

## calculate monthly rain sums
sum_mnth <- data.frame(time = t_mnth, sum = p_mnth)

## calculate rain events per month
nbr_mnth <- format(events$start, "%Y-%m-01 00:00:00")
nbr_mnth <- table(nbr_mnth)
nbr_mnth <- data.frame(time = as.POSIXct(names(nbr_mnth)),
                        n = as.numeric(nbr_mnth))

## calculate monthly inter-rain-durations
t_dry <- c(0, diff(as.numeric(events$start)) / 3600)
dry_mnth <- aggregate(x = t_dry,
                      by = list(format(events$start, "%Y-%m-01 00:00:00")),
                      FUN = median, na.rm = TRUE)
dry_mnth$Group.1 <- as.POSIXct(x = dry_mnth$Group.1, tz = "UTC")
names(dry_mnth) = c("time", "dry")

## calculate monthly averaged TFR dynamics (sum, intensity, duration)
sum_mnth <- aggregate(x = tfr_sum$sum,
                      by = list(format(events$start, "%Y-%m-01 00:00:00")),
                      FUN = median, na.rm = TRUE)
sum_mnth$Group.1 <- as.POSIXct(x = sum_mnth$Group.1, tz = "UTC")
names(sum_mnth) = c("time", "sum")

dur_mnth <- aggregate(x = tfr_dur$dur,
                      by = list(format(events$start, "%Y-%m-01 00:00:00")),
                      FUN = median, na.rm = TRUE)
dur_mnth$Group.1 <- as.POSIXct(x = dur_mnth$Group.1, tz = "UTC")
names(dur_mnth) = c("time", "dur")

int_mnth <- aggregate(x = tfr_int$int,
                      by = list(format(events$start, "%Y-%m-01 00:00:00")),
                      FUN = median, na.rm = TRUE)
int_mnth$Group.1 <- as.POSIXct(x = int_mnth$Group.1, tz = "UTC")
names(int_mnth) = c("time", "int")

## calculate monthly average rain dynamic CoV
cov_dur_mnth <- aggregate(x = tfr_dur$dur,
                          by = list(format(events$start, "%Y-%m-01
00:00:00")),
                          FUN = function(x) {

sd(x, na.rm = TRUE) / mean(x, na.rm = TRUE) * 100

```

[illegible]

[illegible]

[illegible]

```

        na.action = "na.omit")

m_int_mnth_0.05 <- quantreg::rq(y ~ x, tau = 0.05,
                               data = data.frame(x = mnth,
                                                  y = tfr_int$int),
                               na.action = "na.omit")
m_int_mnth_0.50 <- quantreg::rq(y ~ x, tau = 0.5,
                               data = data.frame(x = mnth,
                                                  y = tfr_int$int),
                               na.action = "na.omit")
m_int_mnth_0.95 <- quantreg::rq(y ~ x, tau = 0.95,
                               data = data.frame(x = mnth,
                                                  y = tfr_int$int),
                               na.action = "na.omit")

m_nbr_mnth_0.05 <- quantreg::rq(y ~ x, tau = 0.05,
                               data = data.frame(x = nbr_mnth$time,
                                                  y = nbr_mnth$n),
                               na.action = "na.omit")
m_nbr_mnth_0.50 <- quantreg::rq(y ~ x, tau = 0.50,
                               data = data.frame(x = nbr_mnth$time,
                                                  y = nbr_mnth$n),
                               na.action = "na.omit")
m_nbr_mnth_0.95 <- quantreg::rq(y ~ x, tau = 0.95,
                               data = data.frame(x = nbr_mnth$time,
                                                  y = nbr_mnth$n),
                               na.action = "na.omit")

## print slopes at annual basis
print(m_sum_mnth_0.05$coefficients[2] * 12)

##           x
## -0.006157405

print(m_sum_mnth_0.50$coefficients[2] * 12)

##           x
## -0.1032158

print(m_sum_mnth_0.95$coefficients[2] * 12)

##           x
## -0.4511293

print(m_dur_mnth_0.05$coefficients[2] * 12)

##           x
## -1.417943e-17

print(m_dur_mnth_0.50$coefficients[2] * 12)

```



```

##          x
## -0.2978071

print(m_dur_mnth_0.95$coefficients[2] * 12)

##          x
## -0.5001851

print(m_int_mnth_0.05$coefficients[2] * 12)

##          x
## 0.00190395

print(m_int_mnth_0.50$coefficients[2] * 12)

##          x
## 0.007287301

print(m_int_mnth_0.95$coefficients[2] * 12)

##          x
## 0.04856314

print(m_nbr_mnth_0.05$coefficients[2] * 12)

##          x
## 8.44288e-08

print(m_nbr_mnth_0.50$coefficients[2] * 12)

##          x
## 2.126771e-22

print(m_nbr_mnth_0.95$coefficients[2] * 12)

##          x
## 5.799119e-07

## print percentage of change
print(m_sum_mnth_0.05$coefficients[2] * 12 * 100 /
      m_sum_mnth_0.05$coefficients[1])

##          x
## -0.5696815

print(m_sum_mnth_0.50$coefficients[2] * 12 * 100 /
      m_sum_mnth_0.50$coefficients[1])

##          x
## -3.288187

print(m_sum_mnth_0.95$coefficients[2] * 12 * 100 /
      m_sum_mnth_0.95$coefficients[1])

##          x
## -3.27605

```

```

print(m_dur_mnth_0.05$coefficients[2] * 12 * 100 /
      m_dur_mnth_0.05$coefficients[1])

##          x
## -1.890591e-15

print(m_dur_mnth_0.50$coefficients[2] * 12 * 100 /
      m_dur_mnth_0.50$coefficients[1])

##          x
## -6.579016

print(m_dur_mnth_0.95$coefficients[2] * 12 * 100 /
      m_dur_mnth_0.95$coefficients[1])

##          x
## -3.226848

print(m_int_mnth_0.05$coefficients[2] * 12 * 100 /
      m_int_mnth_0.05$coefficients[1])

##          x
## 2.705069

print(m_int_mnth_0.50$coefficients[2] * 12 * 100 /
      m_int_mnth_0.50$coefficients[1])

##          x
## 4.123317

print(m_int_mnth_0.95$coefficients[2] * 12 * 100 /
      m_int_mnth_0.95$coefficients[1])

##          x
## 9.855037

print(m_nbr_mnth_0.05$coefficients[2] * 12 * 100 /
      m_nbr_mnth_0.05$coefficients[1])

##          x
## -1.092169e-06

print(m_nbr_mnth_0.50$coefficients[2] * 12 * 100 /
      m_nbr_mnth_0.50$coefficients[1])

##          x
## 2.363079e-21

print(m_nbr_mnth_0.95$coefficients[2] * 12 * 100 /
      m_nbr_mnth_0.95$coefficients[1])

##          x
## -9.98593e-07

```

```

## estimate total percent decrease of average rain duration
p_1 <- m_dur_mnth_0.50$coefficients[1]
p_2 <- m_dur_mnth_0.50$coefficients[1] +
  m_dur_mnth_0.50$coefficients[2] * max(mnth)
print(p_2 * 100 / p_1)

## (Intercept)
##      60.65004

## plot

par(mfcol = c(3, 2))
plot(sum_mnth$time, caTools::runmean(sum_mnth$sum, 6), type = "l")
plot(nbr_mnth$time, caTools::runmean(nbr_mnth$n, 6), type = "l")
plot(dry_mnth$time, caTools::runmean(dry_mnth$dry, 6), type = "l")

plot(events$start, tfr_sum$sum, col = adjustcolor(col = col_season, 0.5),
      log = "y", cex = size, pch = 20)
lines(x = events$start, y = predict(object = m_sum_mnth_0.05), col = "grey")
lines(x = events$start, y = predict(object = m_sum_mnth_0.50), col = "grey")
lines(x = events$start, y = predict(object = m_sum_mnth_0.95), col = "grey")

plot(events$start, tfr_dur$dur / 3600, col = adjustcolor(col = col_season,
0.5),
      log = "y", cex = size, pch = 20)
lines(x = events$start, y = predict(object = m_dur_mnth_0.05), col = "grey")
lines(x = events$start, y = predict(object = m_dur_mnth_0.50), col = "grey")
lines(x = events$start, y = predict(object = m_dur_mnth_0.95), col = "grey")

plot(events$start, tfr_int$int, col = adjustcolor(col = col_season, 0.5),
      log = "y", cex = size, pch = 20)
lines(x = events$start, y = predict(object = m_int_mnth_0.05), col = "grey")
lines(x = events$start, y = predict(object = m_int_mnth_0.50), col = "grey")
lines(x = events$start, y = predict(object = m_int_mnth_0.95), col = "grey")

```

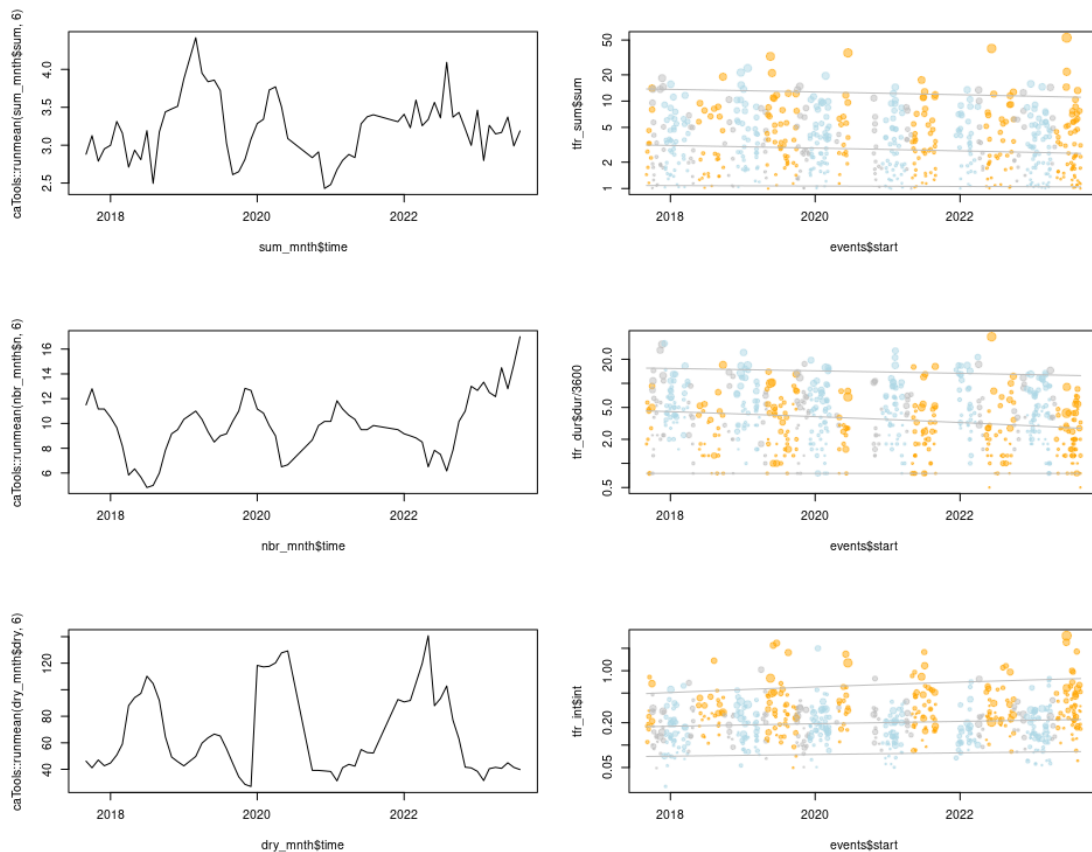


Figure 3 of the main text

extract TF ratios of each sampler for each event

```
tfr_co <- do.call(rbind, lapply(X = event_info, FUN = function(x, par) {
```

```
  x$tfr
```

```
}, par))
```

```
tfr_co_sum <- tfr_co[season == "summer",]
```

```
tfr_co_win <- tfr_co[season == "winter",]
```

```
tfr_co_sum_avg <- apply(tfr_co_sum, 2, median, na.rm = TRUE)
```

```
tfr_co_win_avg <- apply(tfr_co_win, 2, median, na.rm = TRUE)
```

```
summary(tls$co_sum)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.02050 0.02238 0.02982 0.04865 0.05723 0.17583
```

```
summary(tls$co_win)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.1686  0.2780  0.3333  0.3199  0.3641  0.4997
```

```

summary(tfr_co_sum_avg)

##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.2727  0.6063  0.7270  0.7015  0.7800  1.0776

summary(tfr_co_win_avg)

##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.02062 0.71668 0.76940 0.71793 0.80268 0.95109

m_sum <- quantreg::rq(y ~ x, tau = 0.5,
                     data = data.frame(x = tls$co_sum, y = tfr_co_sum_avg),
                     na.action = "na.omit")
m_win <- quantreg::rq(y ~ x, tau = 0.5,
                     data = data.frame(x = tls$co_win, y = tfr_co_win_avg),
                     na.action = "na.omit")

## fit quantile models to tfr data
df_tfr <- data.frame(x = mnth,
                     y = tfr_int$tfr)

df_tfr_sum <- df_tfr[season == "summer",]
df_tfr_win <- df_tfr[season == "winter",]

m_tfr_0.05 <- quantreg::rq(y ~ x, tau = 0.05,
                          data = df_tfr,
                          na.action = "na.omit")
m_tfr_0.50 <- quantreg::rq(y ~ x, tau = 0.50,
                          data = df_tfr,
                          na.action = "na.omit")
m_tfr_0.95 <- quantreg::rq(y ~ x, tau = 0.95,
                          data = df_tfr,
                          na.action = "na.omit")
print(m_tfr_0.05$coefficients[2] * 12)

## x
## 0

print(m_tfr_0.05$coefficients[2] * 12 * 100 /
      m_tfr_0.05$coefficients[1])

## x
## NaN

print(m_tfr_0.50$coefficients[2] * 12)

## x
## -0.0514586

print(m_tfr_0.50$coefficients[2] * 12 * 100 /
      m_tfr_0.50$coefficients[1])

```

```

##          x
## -5.748802

print(m_tfr_0.95$coefficients[2] * 12)

##          x
## -0.04070164

print(m_tfr_0.95$coefficients[2] * 12 * 100 /
      m_tfr_0.95$coefficients[1])

##          x
## -3.543136

m_tfr_sum_0.05 <- quantreg::rq(y ~ x, tau = 0.05,
                              data = df_tfr_sum,
                              na.action = "na.omit")
m_tfr_sum_0.50 <- quantreg::rq(y ~ x, tau = 0.50,
                              data = df_tfr_sum,
                              na.action = "na.omit")
m_tfr_sum_0.95 <- quantreg::rq(y ~ x, tau = 0.95,
                              data = df_tfr_sum,
                              na.action = "na.omit")

m_tfr_win_0.05 <- quantreg::rq(y ~ x, tau = 0.05,
                              data = df_tfr_win,
                              na.action = "na.omit")
m_tfr_win_0.50 <- quantreg::rq(y ~ x, tau = 0.50,
                              data = df_tfr_win,
                              na.action = "na.omit")
m_tfr_win_0.95 <- quantreg::rq(y ~ x, tau = 0.95,
                              data = df_tfr_win,
                              na.action = "na.omit")

## get median throughfall ratio at start and end of instrumented period
tfr_2017 <- tfr_int$tfr[events$start < as.POSIXct("2017-10-01")]
tfr_2023 <- tfr_int$tfr[events$start > as.POSIXct("2023-06-01") &
                      events$start < as.POSIXct("2023-10-01")]

summary(tfr_2017)

##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.3472  0.7790  0.8099  0.8657  0.9448  1.3415

summary(tfr_2023)

##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.2484  0.4800  0.4472  0.6483  0.8995

mnth_break <- 40

df_cov <- data.frame(x = mnth, y = tfr_cov$cov)
df_cov_a <- df_cov[mnth < mnth_break,]
df_cov_b <- df_cov[mnth >= mnth_break,]

```

```

df_cov_sum <- df_cov[season == "summer",]
df_cov_sum_a <- df_cov_sum[df_cov_sum$x < mnth_break,]
df_cov_sum_b <- df_cov_sum[df_cov_sum$x >= mnth_break,]

df_cov_win <- df_cov[season == "winter",]
df_cov_win_a <- df_cov_win[df_cov_win$x < mnth_break,]
df_cov_win_b <- df_cov_win[df_cov_win$x >= mnth_break,]

m_cov_a_0.05 <- quantreg::rq(y ~ x, tau = 0.05,
                             data = df_cov_a,
                             na.action = "na.omit")
m_cov_a_0.50 <- quantreg::rq(y ~ x, tau = 0.50,
                             data = df_cov_a,
                             na.action = "na.omit")
m_cov_a_0.95 <- quantreg::rq(y ~ x, tau = 0.95,
                             data = df_cov_a,
                             na.action = "na.omit")

m_cov_a_sum_0.05 <- quantreg::rq(y ~ x, tau = 0.05,
                                 data = df_cov_sum_a,
                                 na.action = "na.omit")
m_cov_a_sum_0.50 <- quantreg::rq(y ~ x, tau = 0.50,
                                 data = df_cov_sum_a,
                                 na.action = "na.omit")
m_cov_a_sum_0.95 <- quantreg::rq(y ~ x, tau = 0.95,
                                 data = df_cov_sum_a,
                                 na.action = "na.omit")

m_cov_a_win_0.05 <- quantreg::rq(y ~ x, tau = 0.05,
                                 data = df_cov_win_a,
                                 na.action = "na.omit")
m_cov_a_win_0.50 <- quantreg::rq(y ~ x, tau = 0.50,
                                 data = df_cov_win_a,
                                 na.action = "na.omit")
m_cov_a_win_0.95 <- quantreg::rq(y ~ x, tau = 0.95,
                                 data = df_cov_win_a,
                                 na.action = "na.omit")

m_cov_b_0.05 <- quantreg::rq(y ~ x, tau = 0.05,
                             data = df_cov_b,
                             na.action = "na.omit")
m_cov_b_0.50 <- quantreg::rq(y ~ x, tau = 0.50,
                             data = df_cov_b,
                             na.action = "na.omit")
m_cov_b_0.95 <- quantreg::rq(y ~ x, tau = 0.95,
                             data = df_cov_b,
                             na.action = "na.omit")

```

```

m_cov_b_sum_0.05 <- quantreg::rq(y ~ x, tau = 0.05,
                                data = df_cov_sum_b,
                                na.action = "na.omit")
m_cov_b_sum_0.50 <- quantreg::rq(y ~ x, tau = 0.50,
                                data = df_cov_sum_b,
                                na.action = "na.omit")
m_cov_b_sum_0.95 <- quantreg::rq(y ~ x, tau = 0.95,
                                data = df_cov_sum_b,
                                na.action = "na.omit")

m_cov_b_win_0.05 <- quantreg::rq(y ~ x, tau = 0.05,
                                data = df_cov_win_b,
                                na.action = "na.omit")
m_cov_b_win_0.50 <- quantreg::rq(y ~ x, tau = 0.50,
                                data = df_cov_win_b,
                                na.action = "na.omit")
m_cov_b_win_0.95 <- quantreg::rq(y ~ x, tau = 0.95,
                                data = df_cov_win_b,
                                na.action = "na.omit")

print(m_cov_a_0.05$coefficients[2] * 12)

##           x
## -0.4613486

print(m_cov_a_0.05$coefficients[2] * 12 * 100 /
      m_cov_a_0.05$coefficients[1])

##           x
## -7.262684

print(m_cov_b_0.05$coefficients[2] * 12)

##           x
##  2.714392

print(m_cov_b_0.05$coefficients[2] * 12 * 100 /
      m_cov_b_0.05$coefficients[1])

##           x
## -120.9326

cov_detrend_a <- df_cov_a$y - (df_cov_a$y * m_cov_a_0.50$coefficients[2] +
m_cov_a_0.50$coefficients[1])
cov_detrend_b <- df_cov_b$y - (df_cov_b$y * m_cov_b_0.50$coefficients[2] +
m_cov_b_0.50$coefficients[1])

cov_detrend_a_sum <- cov_detrend_a[season == "summer"]
cov_detrend_a_win <- cov_detrend_a[season == "winter"]

cov_detrend_b_sum <- cov_detrend_b[season == "summer"]

```



```

cov_detrend_b_win <- cov_detrend_b[season == "winter"]

## plots
par(mfcol = c(2, 2))
boxplot(data.frame(s = c(tfr_co_sum_avg), w = tfr_co_win_avg))

plot(mnth, tfr_int$tfr, col = adjustcolor(col = col_season, 0.5),
     ylim = c(0, 2), cex = size, pch = 20)

lines(x = m_tfr_0.05$x[,2], y = predict(object = m_tfr_0.05), col = "grey")
lines(x = m_tfr_0.50$x[,2], y = predict(object = m_tfr_0.50), col = "grey")
lines(x = m_tfr_0.95$x[,2], y = predict(object = m_tfr_0.95), col = "grey")

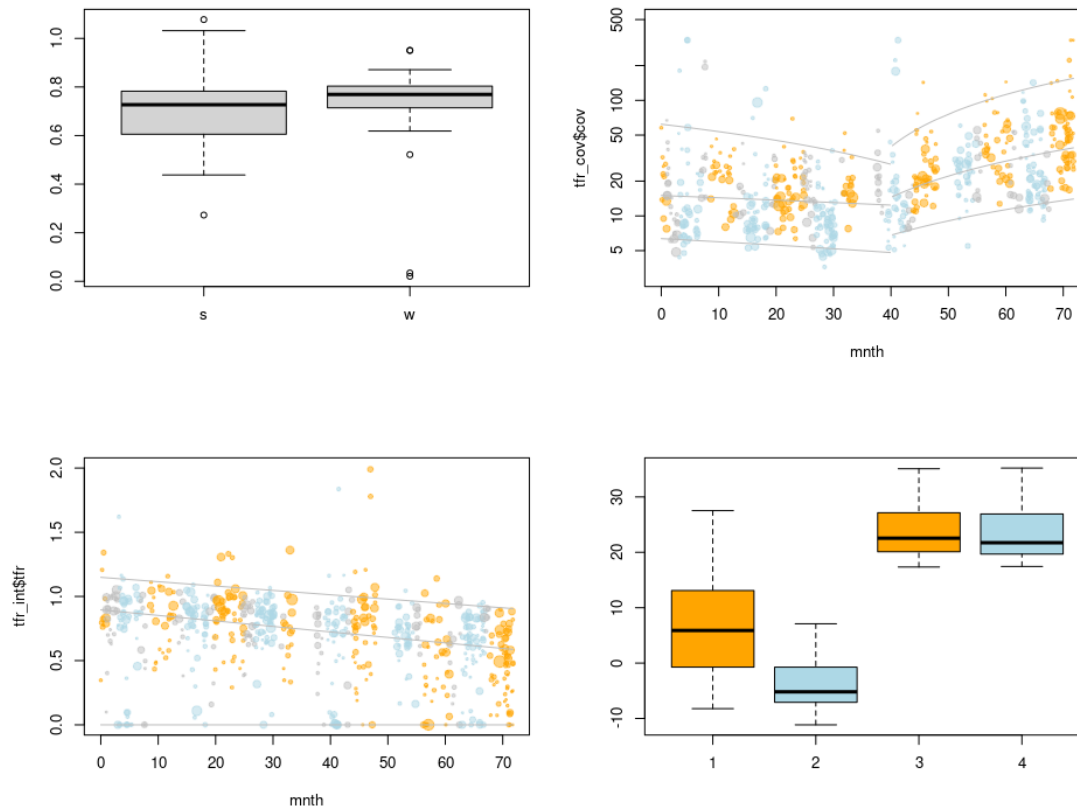
plot(mnth, tfr_cov$cov, log = "y", ylim = c(3, 500),
     col = adjustcolor(col = col_season, 0.5), cex = size, pch = 20)

lines(x = m_cov_a_0.05$x[,2], y = predict(object = m_cov_a_0.05), col =
"grey")
lines(x = m_cov_a_0.50$x[,2], y = predict(object = m_cov_a_0.50), col =
"grey")
lines(x = m_cov_a_0.95$x[,2], y = predict(object = m_cov_a_0.95), col =
"grey")

lines(x = m_cov_b_0.05$x[,2], y = predict(object = m_cov_b_0.05), col =
"grey")
lines(x = m_cov_b_0.50$x[,2], y = predict(object = m_cov_b_0.50), col =
"grey")
lines(x = m_cov_b_0.95$x[,2], y = predict(object = m_cov_b_0.95), col =
"grey")

boxplot(list(cov_detrend_a_sum, cov_detrend_a_win,
             cov_detrend_b_sum, cov_detrend_b_win),
        outline = FALSE,
        col = rep(c("orange", "lightblue"), 2))

```



```
print(range(tls$co_sum))
## [1] 0.02049682 0.17583459

print(range(tls$co_win))
## [1] 0.1686255 0.4996576

summary(tls$co_win - tls$co_sum)

##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.1480  0.2524  0.2938  0.2713  0.3072  0.3256

## plot CO-TFR data
par(mfcol = c(1, 2))

plot(1:nrow(tls), tls$co_sum, col = "red", ylim = c(0, 0.6))
points(1:nrow(tls), tls$co_win, col = "blue")

plot(NA, xlim = c(0, 0.5), ylim = c(0.3, 1.1))
points(x = tls$co_sum, y = tfr_co_sum_avg, col = "orange", pch = 20, cex =
1.5)
points(x = tls$co_win, y = tfr_co_win_avg, col = "lightblue", pch = 20, cex =
1.5)
```

```
segments(x0 = tls$co_sum, y0 = tfr_co_sum_avg, x1 = tls$co_win, y1 =
tfr_co_win_avg, col = "grey")
```

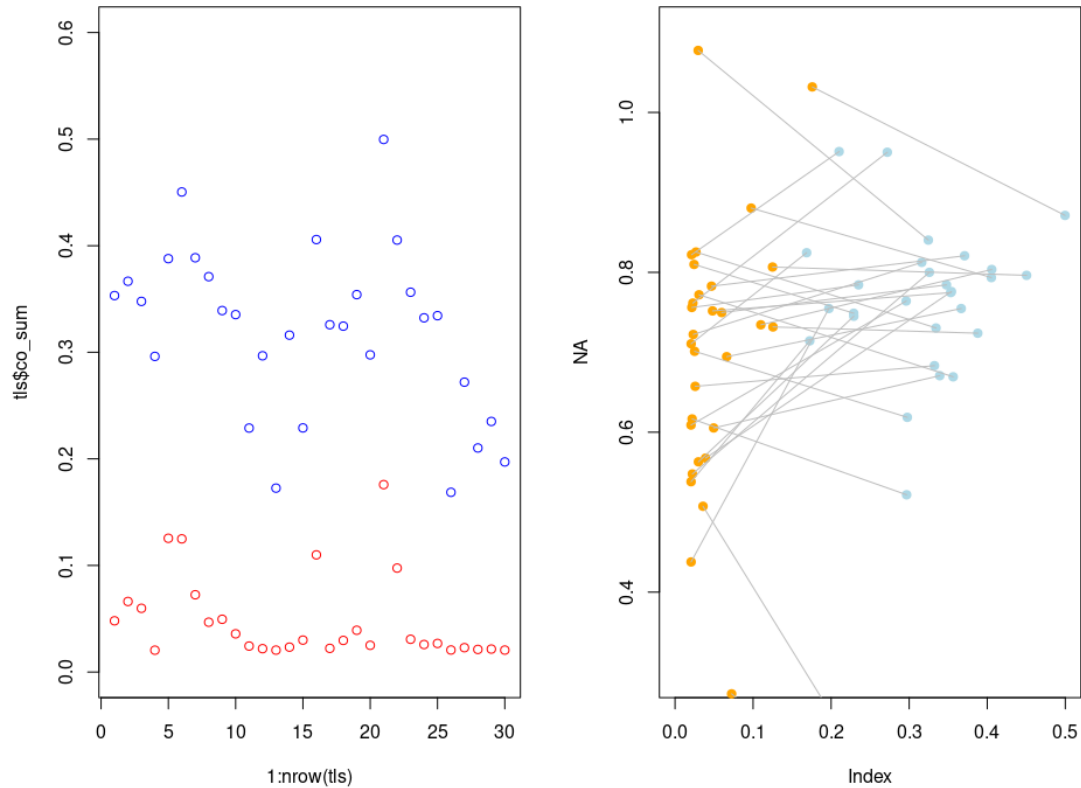


Figure 4 of the main publication

```
## isolate summer throughfall by duration
```

```
tfr_dur_sum <- tfr_dur[season == "summer",]
```

```
## classify data by duration
```

```
tfr_dur_cls_sum <- classify(x = tfr_dur_sum$dur,
                           y = tfr_dur_sum$tfr,
                           breaks = par$breaks$duration,
                           list = FALSE)
```

```
colnames(tfr_dur_cls_sum) = as.numeric(colnames(tfr_dur_cls_sum)) / 3600
```

```
n_tfr_dur_sum <- nrow(tfr_dur_cls_sum) -
```

```
  apply(X = tfr_dur_cls_sum,
        MARGIN = 2,
        FUN = function(x) {
```

```
    sum(is.na(x))
  })
```

```
## isolate winter throughfall by duration
```

```
tfr_dur_win <- tfr_dur[season == "winter",]
```

```

## classify data by duration
tfr_dur_cls_win <- classify(x = tfr_dur_win$dur,
                           y = tfr_dur_win$tfr,
                           breaks = par$breaks$duration,
                           list = FALSE)
colnames(tfr_dur_cls_win) = as.numeric(colnames(tfr_dur_cls_win)) / 3600

n_tfr_dur_win <- nrow(tfr_dur_cls_win) -
  apply(X = tfr_dur_cls_win,
        MARGIN = 2,
        FUN = function(x) {

          sum(is.na(x))
        })

## isolate summer throughfall by intensity
tfr_int_sum <- tfr_int[season == "summer",]

## classify summer throughfall by intensity
tfr_int_cls_sum <- classify(x = tfr_int_sum$int,
                           y = tfr_int_sum$tfr,
                           breaks = par$breaks$intensity,
                           list = FALSE)
colnames(tfr_int_cls_sum) = as.numeric(colnames(tfr_int_cls_sum))

n_tfr_int_sum <- nrow(tfr_int_cls_sum) -
  apply(X = tfr_int_cls_sum,
        MARGIN = 2,
        FUN = function(x) {

          sum(is.na(x))
        })

## isolate winter throughfall by intensity
tfr_int_win <- tfr_int[season == "winter",]

## classify summer throughfall by intensity
tfr_int_cls_win <- classify(x = tfr_int_win$int,
                           y = tfr_int_win$tfr,
                           breaks = par$breaks$intensity,
                           list = FALSE)
colnames(tfr_int_cls_win) = as.numeric(colnames(tfr_int_cls_win))

n_tfr_int_win <- nrow(tfr_int_cls_win) -
  apply(X = tfr_int_cls_win,
        MARGIN = 2,
        FUN = function(x) {

          sum(is.na(x))
        })

```

```

})

## isolate summer throughfall by amount
tfr_sum_sum <- tfr_sum[season == "summer",]

## classify summer throughfall by amount
tfr_sum_cls_sum <- classify(x = tfr_sum_sum$sum,
                           y = tfr_int_sum$tfr,
                           breaks = par$breaks$amount,
                           list = FALSE)
colnames(tfr_sum_cls_sum) = as.numeric(colnames(tfr_sum_cls_sum))

n_tfr_sum_sum <- nrow(tfr_sum_cls_sum) -
  apply(X = tfr_sum_cls_sum,
        MARGIN = 2,
        FUN = function(x) {

          sum(is.na(x))
        })

## isolate winter throughfall by amount
tfr_sum_win <- tfr_sum[season == "winter",]

## classify summer throughfall by amount
tfr_sum_cls_win <- classify(x = tfr_sum_win$sum,
                           y = tfr_sum_win$tfr,
                           breaks = par$breaks$amount,
                           list = FALSE)
colnames(tfr_sum_cls_win) = as.numeric(colnames(tfr_sum_cls_win))

n_tfr_sum_win <- nrow(tfr_sum_cls_win) -
  apply(X = tfr_sum_cls_win,
        MARGIN = 2,
        FUN = function(x) {

          sum(is.na(x))
        })

par(mfcol = c(2, 3))

## summer TFR by duration
boxplot(tfr_dur_cls_sum, log = "", ylim = c(0, 2), col = "orange")
text(x = 1:6, y = rep(1.8, 5), labels = n_tfr_dur_sum,
      xlab = "Rain event duration (hours)")

## winter TFR by duration
boxplot(tfr_dur_cls_win, log = "", ylim = c(0, 2), col = "lightblue")
text(x = 1:6, y = rep(1.8, 5), labels = n_tfr_dur_win,
      xlab = "Rain event duration (hours)")

```

```
## summer TFR by intensity
```

```
boxplot(tfr_int_cls_sum, log = "", ylim = c(0, 2), col = "orange")
text(x = 1:6, y = rep(1.8, 5), labels = n_tfr_int_sum,
     xlab = "Rain event duration (hours)")
```

```
## winter TFR by intensity
```

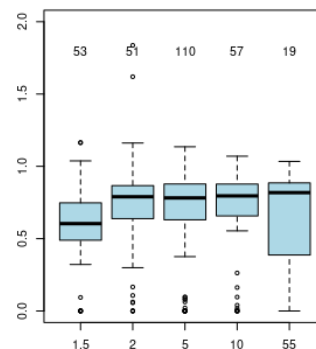
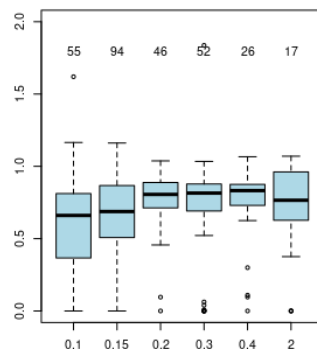
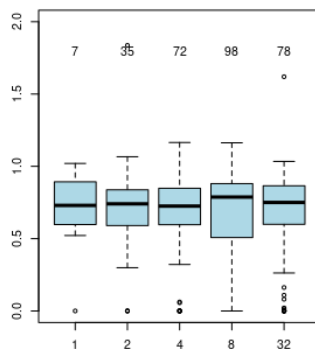
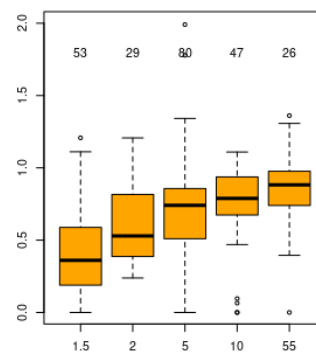
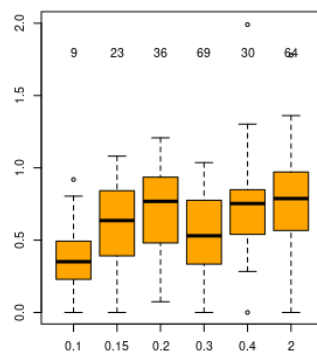
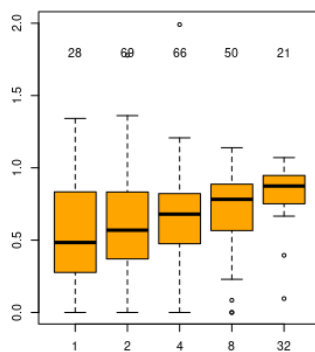
```
boxplot(tfr_int_cls_win, log = "", ylim = c(0, 2), col = "lightblue")
text(x = 1:6, y = rep(1.8, 5), labels = n_tfr_int_win,
     xlab = "Rain event duration (hours)")
```

```
## summer TFR by amount
```

```
boxplot(tfr_sum_cls_sum, log = "", ylim = c(0, 2), col = "orange")
text(x = 1:6, y = rep(1.8, 5), labels = n_tfr_sum_sum,
     xlab = "Rain event duration (hours)")
```

```
## winter TFR by amount
```

```
boxplot(tfr_sum_cls_win, log = "", ylim = c(0, 2), col = "lightblue")
text(x = 1:6, y = rep(1.8, 5), labels = n_tfr_sum_win,
     xlab = "Rain event total amount (mm)")
```



Instrumentation of the study area

Throughfall volumes were measured using 30 tipping bucket rain gauges with integrated data logger and local radio interface (precipitation skip scale enviLog Mini, ecoTech Umwelt-Messsysteme GmbH, Bonn, Germany). Gross precipitation was measured at an adjacent meadow with a resolution of 0.001 mm and an accuracy of ± 0.1 mm using the amount RT-NRT output of the precipitation gauge Pluvio² S (Ott Hydromet GmbH, Kempen, Germany) as a combination of real-time and non-real-time output. The collecting area covers 200 cm² for both, gross and net precipitation buckets. Canopy structure metrics were derived using the 3D-terrestrial laser scanner Faro M70 (Faro Technologies Inc., Lake Mary, USA) above the respective throughfall rain gauges during growing and dormant season.



Fig. S1: Instrumented study area of the *Ebergötzen research and teaching project*



Fig. S2: Throughfall sampler from the front and from above.