Supplementary material to A principal component based strategy for regionalising precipitation intensity-duration-frequency (IDF) statistics

Kajsa Parding

March 22, 2022

Introduction

This is an R-markdown document describing the analysis presented in the paper "A principal component based strategy for regionalising precipitation intensity-duration-frequency (IDF) statistics" by Parding et al (2022). The document is arranged as follows: First, libraries are imported, functions are defined, and paths are set. Then data (return values and daily observations of temperature and precipitation) are imported, inspected and prepared for further analysis. Next, the return values are subjected to Principal Component Analysis (PCA) and the principal components (PCs) are investigated. Finally, statistical relationships are established beetween the spatial patterns associated with the leading PCs of the return values and climatological and geographical information for the same stations, using Bayesian linear modeling (the BAS package). The IDF curves generated by the statistical models are compared to the original IDF curves in various ways to evaluate the skill of the statistical models.

Import libraries and define functions

```
library(BAS)
library(cowplot)
library(ggplot2)
library(gnumeric)
if(!require(esd)) devtools::install_github("metno/esd")
library(esd)
if(!require(wesanderson)) {
  devtools::install_github("karthik/wesanderson")
}
library(wesanderson)
```

```
## Set a local path where R-scripts and data are stored
path.local <- '/path/to/project'
```

```
pca2matrix <- function(pca, ip=1:10) {
  dU <- dim(pca$u)
  dV <- dim(pca$v)
  U <- pca$u[,ip]
  W <- pca$d[ip]
  V <- pca$v[,ip]
  dim(U) <- c(dU[1], length(ip))
  dim(V) <- c(dV[1], length(ip))
  if(length(ip)==1) {
    diag.W <- pca.Z$d[1]
  } else {
    diag.W <- diag(pca.Z$d[ip])
  }
  zz <- U %*% diag.W %*% t(V)
  dim(zz) <- dU
  return(zz)
}
```

Import data

IDF data

IDF data were provided by Julia Lutz from the Norwegian Meteorological Institute. Here, the IDF data are imported, rearranged and saved in an Rd-file for easy access next time the data are used.

```
# IDF curves from Julia Lutz at MET Norway
file.idf <- file.path(path.local, 'idf.Rda')
force <- FALSE
if(file.exists(file.idf) & !force) {
  load(file.idf)
} else {
  ## Load file with IDF data
  load(file.path(path.local, "ivf_fixed_all_stations.Rda"))
  ## Reorganise the IDF data into a matrix
  nmin <- 10
  rp <- unique(out_bay$retperiod[!is.na(out_bay$retperiod)])
  dr <- as.numeric(as.character(
      unique(out_bay$duration[!is.na(out_bay$duration)])))
  st <- as.numeric(
    as.character(unique(out_bay$stnr[!is.na(out_bay$stnr)])))
  Z <- rep(NA, length(rp)*length(dr)*length(st))
  dim(Z) <- c(length(rp), length(dr), length(st))
  dimnames(Z) <- list(rp, dr, st)
  Z.q25 <- Z.q975 <- Z
  for(k in seq_along(rp)) {
    for(j in seq_along(dr)) {
      for(i in seq_along(st)) {
        l <- !is.na(out_bay$stnr) & out_bay$retperiod==rp[k] &
          as.numeric(as.character(out_bay$dur))==dr[j] &
          as.numeric(as.character(out_bay$stnr))==st[i]
        if(any(l)) {
          Z[k,j,i] <- out_bay$retlev[l]
          Z.q25[k,j,i] <- out_bay$retlev_2.5[l]
          Z.q975[k,j,i] <- out_bay$retlev_97.5[l]
```

```
        }
      }
    }
  }
  ## Number of seasons that go into the IDF calculations
  N <- read.csv(file.path(path.local, "station_years_new.csv"))
  N <- array(as.numeric(unlist(
      apply(N, 2, function(x) {
        strsplit(as.character(x), split="\t")}) )),
      dim=c(2, nrow(N)))
  nz <- N[2,sapply(st, function(s) which(s==N[1,]))]
  ## Exclude IDF data based on 10 seasons or less
  Z <- Z[,,nz>nmin]
  Z.q25 <- Z.q25[,,nz>nmin]
  Z.q975 <- Z.q975[,,nz>nmin]
  nz <- nz[nz>nmin]
  ## Add some meta data
  attr(Z,"n") <- nz
  attr(Z,"q2.5") <- Z.q25
  attr(Z,"q97.5") <- Z.q975
  attr(Z, "station_id") <- dimnames(Z)[[3]]
  meta <- metno.frost.meta.minute()
  attr(Z,"location") <- sapply(stid(Z),
    function(s) meta$location[meta$station_id==s][[1]])
  attr(Z,"altitude") <- sapply(stid(Z),
    function(s) meta$altitude[meta$station_id==s][[1]])
  attr(Z,"longitude") <- sapply(stid(Z),
    function(s) meta$longitude[meta$station_id==s][[1]])
  attr(Z,"latitude") <- sapply(stid(Z),
    function(s) meta$latitude[meta$station_id==s][[1]])
  attr(Z,"d.ocean") <- distance2ocean(lon(Z), lat(Z))$distance
  ## Save reorganized IDF data
  save(file=file.idf, Z, Z.q25, Z.q975)
}
```

Daily precipitation and temperature data

Daily precipitation and temperature statistics are fetched from the Norwegian Meteorological Institute for the locations where there are IDF curves, using the 'station' function from the R-package 'esd'. When data are not available from the exact same place, they are fetched from the closest station. Data are saved in an Rd-file for quicker access next time the data are used.

```
file.day <- file.path(path.local, "metno.data.daily.v2.rda")
if(file.exists(file.day)) {
  load(file.day)
} else {
  nmin <- 10; fmin <- 0.5
  it <- c(1970,2020)
  for(p in c("precip","t2m")) {
    X <- NULL; attr.x <- NULL
    for (i in seq(1,length(stid(Z)))) {
      print(paste0(i,"/",length(stid(Z))))
      x.i <- station(src="METNOD", stid=stid(Z)[i], param=p,
                     it=it, nmin=nmin, verbose=FALSE)
      if(!is.null(x.i)) if(nv(x.i)<(nmin*365*fmin)) x.i <- NULL
      lon.i <- lon(Z)[i]
      lat.i <- lat(Z)[i]
      while(is.null(x.i) & diff(range(lon.i))<=2) {
        x.i <- station(src="METNOD", param=p, nmin=nmin,
                       lon=lon.i, lat=lat.i, it=it,
                       verbose=FALSE)
        lon.i <- lon.i + c(-0.2, 0.2)
        lat.i <- lat.i + c(-0.2, 0.2)
        if(!is.null(x.i)) {
          if(is.null(dim(x.i))) {
            if(nv(x.i)<(nmin*365*fmin)) x.i <- NULL
          } else {
            x.i <- subset(x.i, is=apply(x.i,2,nv)>=(nmin-1)*365)
          }
        }
      }
      if(!is.null(dim(x.i))) {
```

```
        d.i <- distAB(mean(lon.i), mean(lat.i), lon(x.i), lat(x.i))
        x.i <- subset(x.i, is=which.min(d.i)[1])
      }
      if(is.null(X)) {
        X <- x.i
      } else {
        X <- merge(X, x.i)
      }
      for(a in c("longitude","latitude","altitude",
                 "station_id","location")) {
        eval(parse(text=paste0("attr.x$",a," <- c(attr.x$",a,
                               ", attr(x.i,'",a,"')[[1]])")))
      }
      if(i>1) {
        if(length(attr.x$longitude)!=ncol(X)) {
          browser()
        }
      }
    }
    X <- as.station(X, param=attr(x.i, "variable"),
                    unit=attr(x.i, "unit"),
                    lon=attr.x$longitude, lat=attr.x$latitude,
                    stid=attr.x$station_id, loc=attr.x$location)
    eval(parse(text=paste0("X.",p," <- X")))
  }
  save(file=file.day, X.precip, X.t2m)
}
```

Examine and prepare the data

Distance between stations

The distance between the stations of the IDF data and the daily precipitation and temperature data is calculated.

```
d.pr <- sapply(seq(length(lon(Z))), function(i) {
  if(lon(Z)[i]!=lon(X.precip)[i] | lat(Z)[i]!=lat(X.precip)[i]) {
    distAB(lon(Z)[i], lat(Z)[i], lon(X.precip)[i], lat(X.precip)[i])
  } else {
    0
  } } )

d.t2m <- sapply(seq(length(lon(Z))), function(i) {
  if(lon(Z)[i]!=lon(X.t2m)[i] | lat(Z)[i]!=lat(X.t2m)[i]) {
    distAB(lon(Z)[i], lat(Z)[i], lon(X.t2m)[i], lat(X.t2m)[i])
  } else {
    0
  } } )

attr(X.precip, "distance") <- d.pr
attr(X.t2m, "distance") <- d.t2m

print(paste("Range of distances from IDF to precip stations:",
            paste(round(range(d.pr)*1E-3,1),collapse="-"),"km"))
print(paste("Number of stations with both IDF and daily precip:",
            sum(d.pr==0)))
print(paste("Mean distance when distance > 0:",
            paste(round(mean(d.pr[d.pr>0])*1E-3,1),collapse="-"),
            "km"))
print(paste("Range of distances from IDF and t2m stations:",
            paste(round(range(d.t2m)*1E-3,1),collapse="-"),
            "km"))
print(paste("Number of stations with both IDF and daily t2m:",
            sum(d.t2m==0)))
print(paste("Mean distance when > 0:",
            paste(round(mean(d.t2m[d.t2m>0])*1E-3,1),collapse="-"),"km"))
```

```
## [1] "Range of distances from IDF to precip stations: 0-0 km"
## [1] "Number of stations with both IDF and daily precip: 74"
## [1] "Mean distance when distance > 0: NaN km"
## [1] "Range of distances from IDF and t2m stations: 0-25 km"
```

```
## [1] "Number of stations with both IDF and daily t2m: 17"
```

```
## [1] "Mean distance when > 0: 6.3 km"
```

Inspect and visualize the IDF data

Plot settings

```
# Define some stuff that are used in many of the plots
rv <-  as.numeric(rownames(Z))
dur <- as.numeric(gsub("[^0-9]","",colnames(Z)))/60
cols <- c("black","coral2","cornflowerblue","darkorchid3","chartreuse4")
col.m <- colscal(n=12, pal="cat")
col.rv <- rev(wes_palette("Zissou1", n=length(rv),
                          type="continuous"))
col.st <- wes_palette("Rushmore1", n=dim(Z)[3],
                      type="continuous")
ylim.idf <- c(1E-6, 250)
cex <- list(main=1.2, lab=1.1, axis=1, mtext=1, text=1,
            legend=1, sub=1, names=1)
## Return periods for visualisation (20 and 200 year)
kvec <- c(4,8)
## Regions for visualisation
is.regions <- list(is.1=list(lon=NULL, lat=c(68, 80)),
                   is.2=list(lon=NULL, lat=c(62, 68)),
                   is.3=list(lon=c(-20,9), lat=c(45, 62)),
                   is.4=list(lon=c(9,40), lat=c(45, 62)))
```

IDF curves for all stations

The 20 and 200 year return values are plotted for all stations, with colors representing different locations. The 200 year return values are then plotted for four geographical domains (north of 68^{o}N, 62-68^{o}N, 0-10^{o}E/50-62^{o}N, 10-15^{o}E/50-62^{o}N) to make it easier to see the regional characteristics.

```r
# Plots of the 10 and 100 year return values
par(mar=c(3.75,3.75,3,0.5), mgp=c(2.5,0.75,0),
    cex.main=cex$main, cex.lab=cex$lab, cex.axis=cex$axis)
m <- 1
for(k in kvec) {
  par(new=switch(m, FALSE, TRUE),
      fig=switch(m, c(0,0.55,0.5,1), c(0,0.55,0,0.5)))
  plot(range(dur), ylim.idf, type="n", log="x",
       main=paste(rownames(Z)[k],' year return values'),
       xlab=switch(m, "Duration (hours)", ""),
       ylab="Precipitation intensity (mm)")
  grid()
  mtext(side=3, adj=-0.05, line=2, cex=cex$mtext,
        text=paste0("(",letters[m],")"))
  for (i in 1:dim(Z)[3]) {
    points(dur, Z[k,,i], col=col.st[i], cex=0.5)
    lines(dur, Z[k,,i], col=col.st[i])
  }
  m <- m+1
}
# Map showing locations of the IDF curves
data(Oslo)
map(Oslo, xlim=c(3,35), ylim=c(55,75), cex=0.1, col="grey",
    add=TRUE, main="", add.text=FALSE, fig=c(0.55,1,0,1),
    new=FALSE)
points(lon(Z), lat(Z), col=col.st, cex=1.7, pch=19)
mtext(side=3, adj=0, line=1, cex=cex$mtext, text="(c)")
```

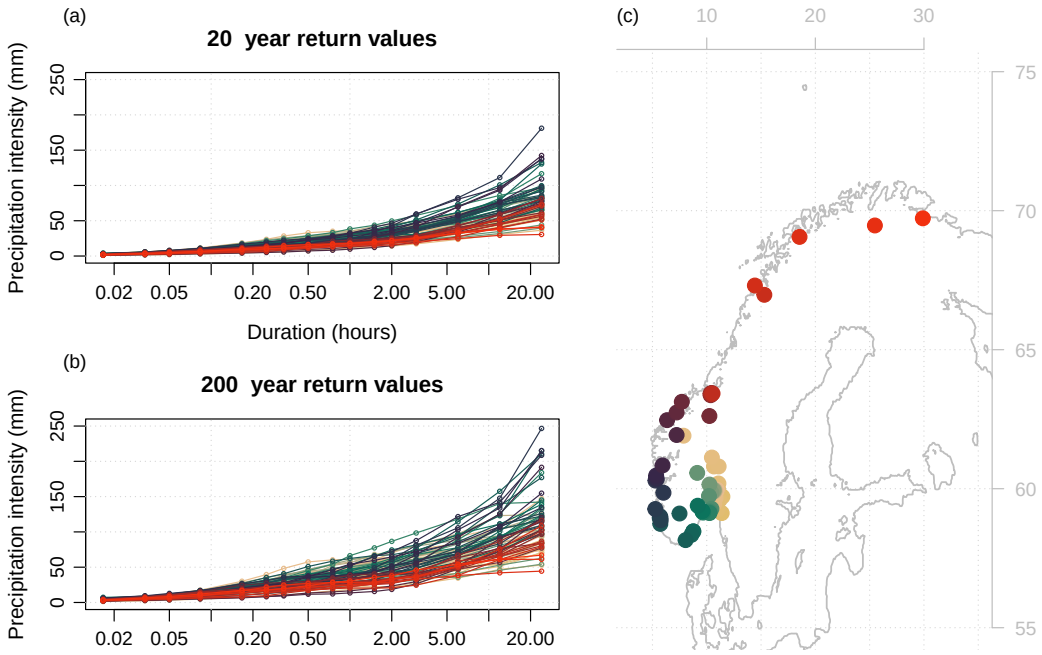

Figure S1: 20 and 200 year return values for 74 Norwegian stations. The map on the right hand side shows which locations the colors in the IDF plots represent.

```
# Look closer at the regional differences in estimated IDFs
k <- kvec[2]
par(mar=c(4,4,2,0.5), mgp=c(2.5,1,0))
for(l in seq_along(is.regions)) {
  is <- is.regions[[l]]
  if(is.null(is$lon)) is$lon <- round(range(lon(Z))+c(-1,1))
  if(is.null(is$lat)) is$lat <- range(lat(Z))+c(-1,1)
  i.region <- which(lon(Z)>is$lon[1] & lon(Z)<=is$lon[2] &
                    lat(Z)>is$lat[1] & lat(Z)<=is$lat[2])
  if(is$lon[1]>min(lon(Z)) & is$lon[2]<max(lon(Z))) {
    label.is <- paste0("longitude: ",is$lon[1],"-",is$lon[2])
  } else if(is$lon[1]>min(lon(Z))) {
    label.is <- paste("longitude >",is$lon[1])
  } else if(is$lon[2]<max(lon(Z))) {
```

```
    label.is <- paste("longitude <",is$lon[2])
  } else label.is <- c()
  if(is$lat[1]>min(lat(Z)) & is$lat[2]<max(lat(Z))) {
    label.is <- c(label.is, paste0("latitude: ",
                                   is$lat[1],"-",is$lat[2]))
  } else if(is$lat[1]>min(lat(Z))) {
    label.is <- c(label.is, paste("latitude >",is$lat[1]))
  } else if(is$lat[2]<max(lat(Z))) {
    label.is <- c(label.is, paste("latitude <",is$lat[2]))
  }
  label.is <- paste(label.is, collapse="\n")
  par(new=(l>1), fig=switch(l, c(0,0.5,0.5,1),
                           c(0.5,1,0.5,1),
                           c(0,0.5,0,0.5),
                           c(0.5,1,0,0.5)))
  plot(range(dur), ylim.idf, type="n", log="x",
     xlab=switch(round(l/2), "1"="", "2"="Duration (hours)"),
     ylab=switch(l%%2+1, "", "Precipitation intensity (mm)"))
  if(l==1) title(paste(rownames(Z)[k],'year return values'),
                 cex=0.95)
  mtext(side=3, line=1, adj=0, cex=cex$mtext,
        text=paste0("(",letters[l],")"))
  grid()
  for (i in i.region) {
    points(dur, Z[k,,i], col=col.st[i], cex=0.3)
    lines(dur, Z[k,,i], col=col.st[i])
  }
  text(0.015, ylim.idf[2]*0.75, label.is, cex=1.1, pos=4)
}
```

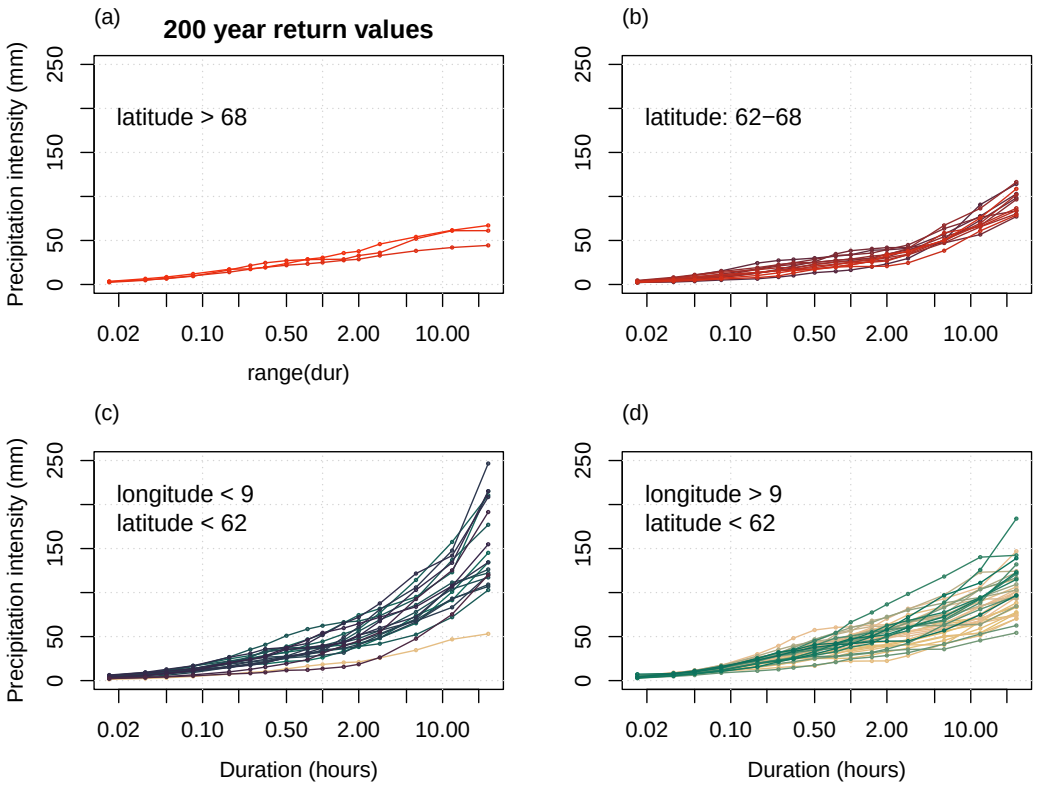

Figure S2: Estimated 200 year return values for 74 Norwegian stations. The four panels show IDF curves for different regions and the colors represent locations as depicted in Figure S1.

IDF curves for example stations

Next, we plot the IDF curves for a set of example stations for which results will be visualized. To select these, we start out with eight stations in different parts of Norway for which the IDF calculations were based on long high quality precipitation measurements (the stations were identified in another study inspecting the same IDF statistics). Out of these eight stations, we choose the ones for which there is a short distance (< 5 km) between the measurement location of the IDF data and the daily precipitation and temperature data used in this study.

```r
stid.select <- c(12290, 18701, 39150, 44190, 50480, 64300, 68230, 82310)
i.vis <- which(stid(Z) %in% stid.select)
for(i in i.vis) {
  writeLines(paste0(loc(Z)[i], ", station id: ", stid(Z)[i],
    "\nlon: ", lon(Z)[i],", lat: ", lat(Z)[i],", alt: ", alt(Z)[i],
    ", alt: ", alt(Z)[i],", distance to ocean: ",
    round(attr(Z,"d.ocean")[i]),
    "\nSeasons for IDF calculations: ",attr(Z,"n")[i]))
  print("Available precipitation data (days per month):")
  print(aggregate(X.precip, FUN="nv", by=month(X.precip))[,i])
  writeLines("\n")
}
```

```
## HAMAR II, station id: 12290
## lon: 11.095, lat: 60.8002, alt: 141, alt: 141, distance to ocean: 202
## Seasons for IDF calculations: 41
## [1] "Available precipitation data (days per month):"
##    1    2    3    4    5    6    7    8    9   10   11   12
##  360  339  372  370  991 1117 1259 1255 1209 1192  450  402
##
##
## OSLO - BLINDERN PLU, station id: 18701
## lon: 10.7202, lat: 59.9423, alt: 94, alt: 94, distance to ocean: 111
## Seasons for IDF calculations: 51
## [1] "Available precipitation data (days per month):"
##    1    2    3    4    5    6    7    8    9   10   11   12
##  345  331  555 1079 1542 1516 1564 1526 1459 1474 1058  567
##
##
## KRISTIANSAND - SØMSKLEIVA, station id: 39150
## lon: 8.052, lat: 58.1502, alt: 12, alt: 12, distance to ocean: 0
## Seasons for IDF calculations: 34
## [1] "Available precipitation data (days per month):"
##    1    2    3    4    5    6    7    8    9   10   11   12
##  450  429  533  720 1052 1129 1123 1083 1046 1017  750  726
##
##
```

```
## TIME - LYE, station id: 44190
## lon: 5.7262, lat: 58.7345, alt: 92, alt: 92, distance to ocean: 0
## Seasons for IDF calculations: 28
## [1] "Available precipitation data (days per month):"
##   1   2   3   4   5   6   7   8   9  10  11  12
## 707 686 738 665 874 792 862 872 829 861 849 757
##
##
## BERGEN - SANDSLI, station id: 50480
## lon: 5.2777, lat: 60.2913, alt: 37, alt: 37, distance to ocean: 0
## Seasons for IDF calculations: 23
## [1] "Available precipitation data (days per month):"
##   1   2   3   4   5   6   7   8   9  10  11  12
## 682 651 673 655 693 622 733 710 656 644 620 660
##
##
## KRISTIANSUND - KARIHOLA, station id: 64300
## lon: 7.7027, lat: 63.1257, alt: 5, alt: 5, distance to ocean: 30
## Seasons for IDF calculations: 35
## [1] "Available precipitation data (days per month):"
##    1    2    3    4    5    6    7    8    9   10   11   12
##  601  637  726  769  948 1015 1118 1133 1018 1114  914  785
##
##
## TRONDHEIM - RISVOLLAN, station id: 68230
## lon: 10.4228, lat: 63.3987, alt: 84, alt: 84, distance to ocean: 64
## Seasons for IDF calculations: 32
## [1] "Available precipitation data (days per month):"
##    1    2    3    4    5    6    7    8    9   10   11   12
##  865  812  878  880 1021  983 1020 1002  985 1005  897  915
##
##
## BODØ - SKIVIKA, station id: 82310
## lon: 14.4309, lat: 67.3084, alt: 5, alt: 5, distance to ocean: 0
## Seasons for IDF calculations: 18
## [1] "Available precipitation data (days per month):"
##   1   2   3   4   5   6   7   8   9  10  11  12
## 445 499 559 539 552 533 557 557 521 557 541 519
```

```
map(Oslo, cex=0, xlim=c(3,35), ylim=c(55,75), main="")
points(lon(Z)[i.vis], lat(Z)[i.vis], col="black", cex=0.5, pch=19)
text(lon(Z)[i.vis], lat(Z)[i.vis], labels=loc(Z)[i.vis], pos=4, cex=0.5)
```

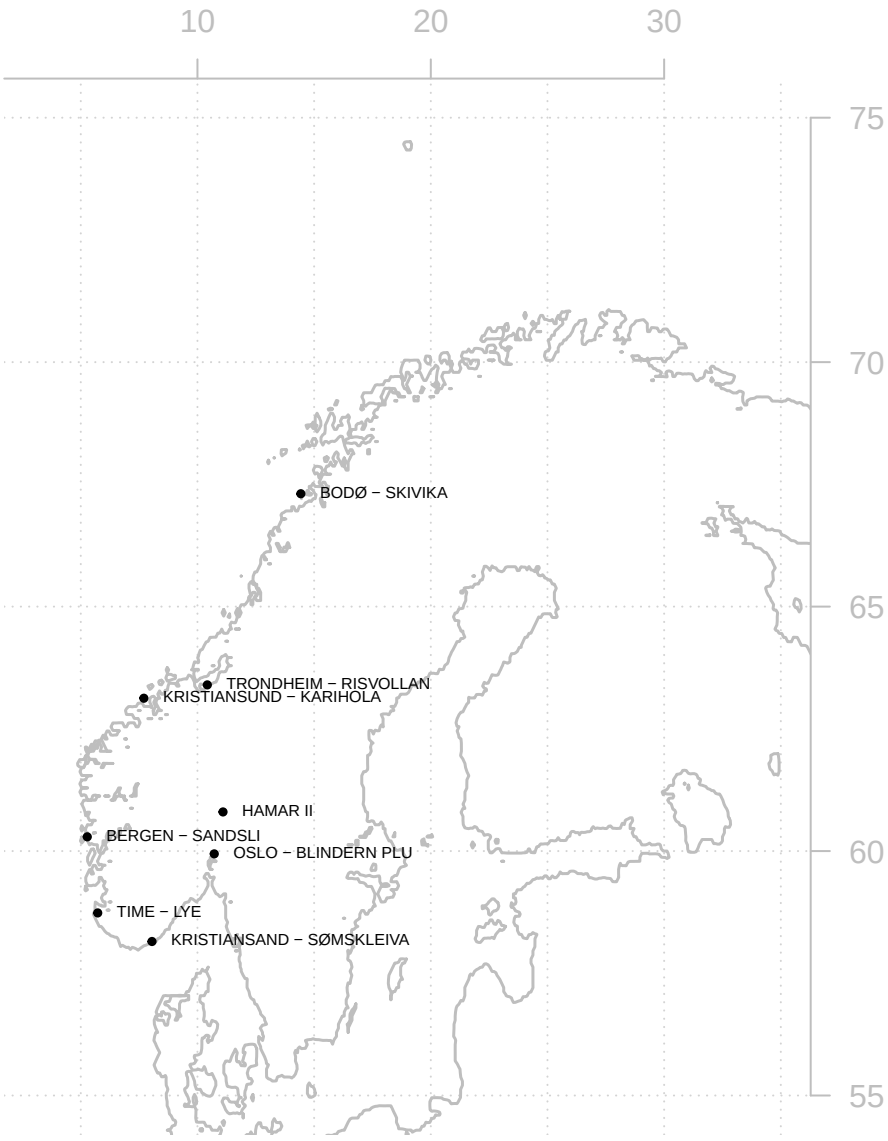

Figure S3: Map of example stations.

```
par(mfrow=c(4,2), mar=c(4,5,2.5,0.5), mgp=c(3,1,0),
    cex.main=cex$main, cex.lab=cex$lab, cex.axis=cex$axis)
for(m in seq_along(i.vis)) {
  i <- i.vis[m]
  plot(range(dur), c(1,1), type="n", log='x', new=FALSE,
       xlim=range(dur), ylim=ylim.idf,
       xlab=switch(ceiling(m/2),"","Duration (hours)"),
       ylab=switch(m %% 2,"","Precipitation intensity (mm)"),
       main=paste0(loc(Z)[i], " (", stid(Z)[i], ")"))
  grid()
  for(k in seq(1,nrow(Z))) lines(dur, Z[k,,i], col=col.rv[k],
                                 type="b", lty=1, pch=19)
  if(m==1) legend("topleft", col=col.rv, pch=19, lty=1, bty="n",
                  cex=cex$legend, ncol=2, bg="transparent",
                  legend=paste(rv,"year return values"))
  mtext(side=3, adj=0, line=1.1, text = paste0("(",letters[m],")"))
}
```

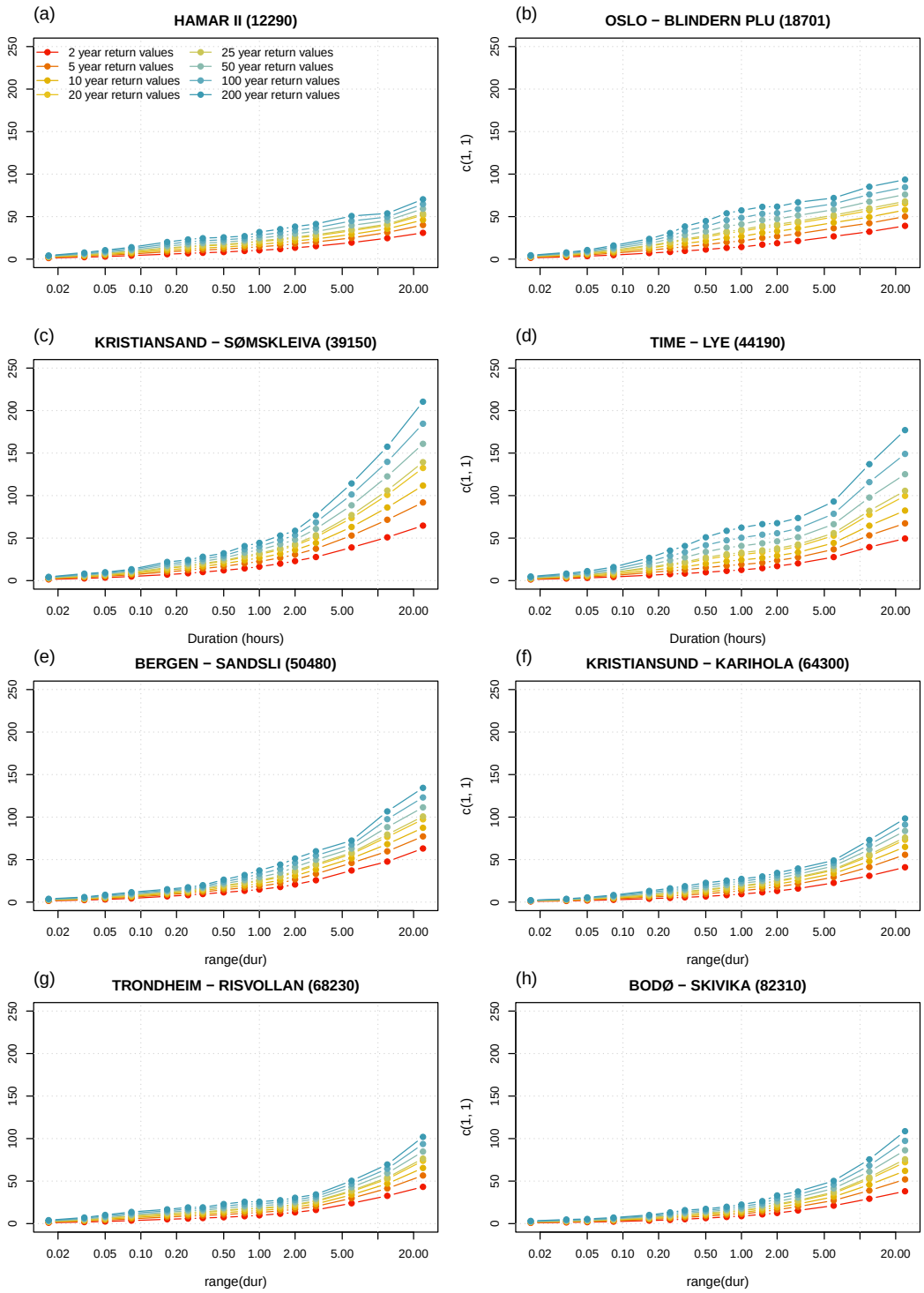

Figure S4: IDF curves for eight stations in Norway. The colors represent different return periods as described in the legend.

Inspect and visualize the daily precipitation and temperature data

Seasonal cycle

The seasonal cycle of the precipitation and temperature are calculated. In addition to the mean precipitation (pr), we examine the mean annual precipitation cycle in terms of the wet-day frequency f_w (how often it rains) and the wet-day mean precipitation μ (an indicator of the mean intensity). Here, f_w and μ are defined as the fraction of days with at least 1 mm precipitation and the mean precipitation in such days, respectively. The colors represent different stations as shown on the map in Figure S1.

Figure S4 shows that the seasonal cycle of the temperature is similar across all stations, but with lower values in the north than south and smaller seasonal variations on the westcoast. The seasonal cycles of the precipitation statistics (pr, μ and fw) are characterised by larger regional differences, both in terms of the range of values and the timing of minima and maxima. In some inland stations, no precipitation observations are available in the period December-March, likely because only rain is recorded and in the winter most if not all precipitation comes in the form of snow.

```
# Mean seasonal cycles
FW <- aggregate(X.precip,month,'wetfreq',threshold=1)
MU <- aggregate(X.precip,month,'wetmean',threshold=1)
```

```
## Warning in sqrt(coredata(n) - 1): NaNs produced
```

```
PR <- aggregate(X.precip,month,'mean',na.rm=TRUE)
```

```
## Warning in sqrt(coredata(n) - 1): NaNs produced
```

```
T2M <- aggregate(X.t2m,month,'mean')
mar <- c(3,4,3,0.5)
mgp <- c(2.5,1,0)
plot(FW, new=FALSE, map.show=FALSE, col=col.st,
     fig=c(0,0.5,0.5,1), mar=mar, mgp=mgp, cex.axis=cex$axis,
     cex.main=cex$main, cex.lab=cex$lab)
mtext(side=3, adj=-0.05, line=1, cex=cex$mtext, text="(a)")
plot(MU, new=FALSE, map.show=FALSE, col=col.st,
```

```
     fig=c(0.5,1,0.5,1), add=TRUE, mar=mar, mgp=mgp,
     cex.axis=cex$axis, cex.main=cex$main, cex.lab=cex$lab)
mtext(side=3, adj=-0.05, line=1, cex=cex$mtext, text="(b)")
plot(PR, new=FALSE, map.show=FALSE, col=col.st,
     fig=c(0,0.5,0,0.5), add=TRUE, mar=mar, mgp=mgp,
     cex.axis=cex$axis, cex.main=cex$main, cex.lab=cex$lab)
mtext(side=3, adj=-0.05, line=1, cex=cex$mtext, text="(c)")
plot(T2M, new=FALSE, map.show=FALSE, col=col.st,
     fig=c(0.5,1,0,0.5), add=TRUE, mar=mar, mgp=mgp,
     cex.axis=cex$axis, cex.main=cex$main, cex.lab=cex$lab)
mtext(side=3, adj=-0.05, line=1, cex=cex$mtext, text="(d)")
```

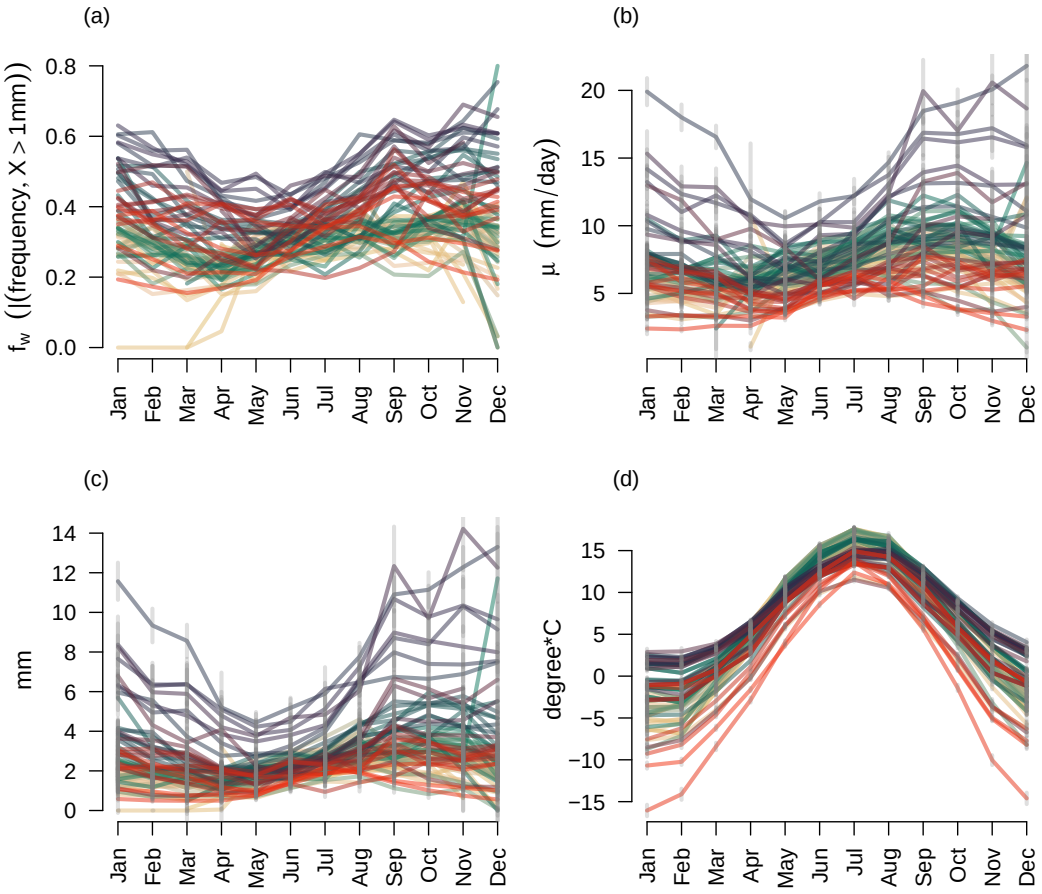

Figure S5: Mean seasonal cycle of the wet-day frequency, wet-day mean precipitation, mean precipitation, and air temperature at 2m elevation. The colors represent different locations as shown in Figure S1.

Polynomial representation of IDF curves

An attempt was made to represent the IDF statistics as polynomials. The strategy was abandoned because of poor fit and, for higher order polynomials, squiggly shapes that do not resemble IDF curves.

```
x <- seq(min(dur), max(dur), min(diff(dur))/2.)
par(bty='n', mfrow=c(3,1))
k <- 5
pvec <- seq(2,4)
for(m in seq_along(pvec)) {
  plot(range(dur), c(0,250), type='n', log="x",
       main=switch(as.numeric(m==1)+1, "",
                   paste0("Precipitation: ", rownames(Z)[[k]],
                          "-year intensity-duration-frequency")),
       xlab='Hours', ylab='mm', new=TRUE)
  text(dur[1], 250, paste0("(",letters[m],")"),
       cex=cex$mtext, adj=c(0,-1.5), xpd=NA)
  legend("topleft", pch=c(19,NA), lty=c(NA,2),
         legend=c("original return value",
                  paste0(pvec[[m]]," order polynomial fit")))
  grid()
  for (i in 1:dim(Z)[[3]]) {
    caldat <- data.frame(x=c(0,dur),y=c(0,Z[k,,i]))
    polfit <- lm(paste0("y ~ x + ",
                        paste0("I(x^",2:pvec[[m]],")",collapse=" + ")),
                 data=caldat)
    lines(x, predict(polfit, newdata=list("x"=x)),
          col=col.st[i], lty=2)
    points(dur, Z[k,,i], col=col.st[i], pch=19)
  }
}
```

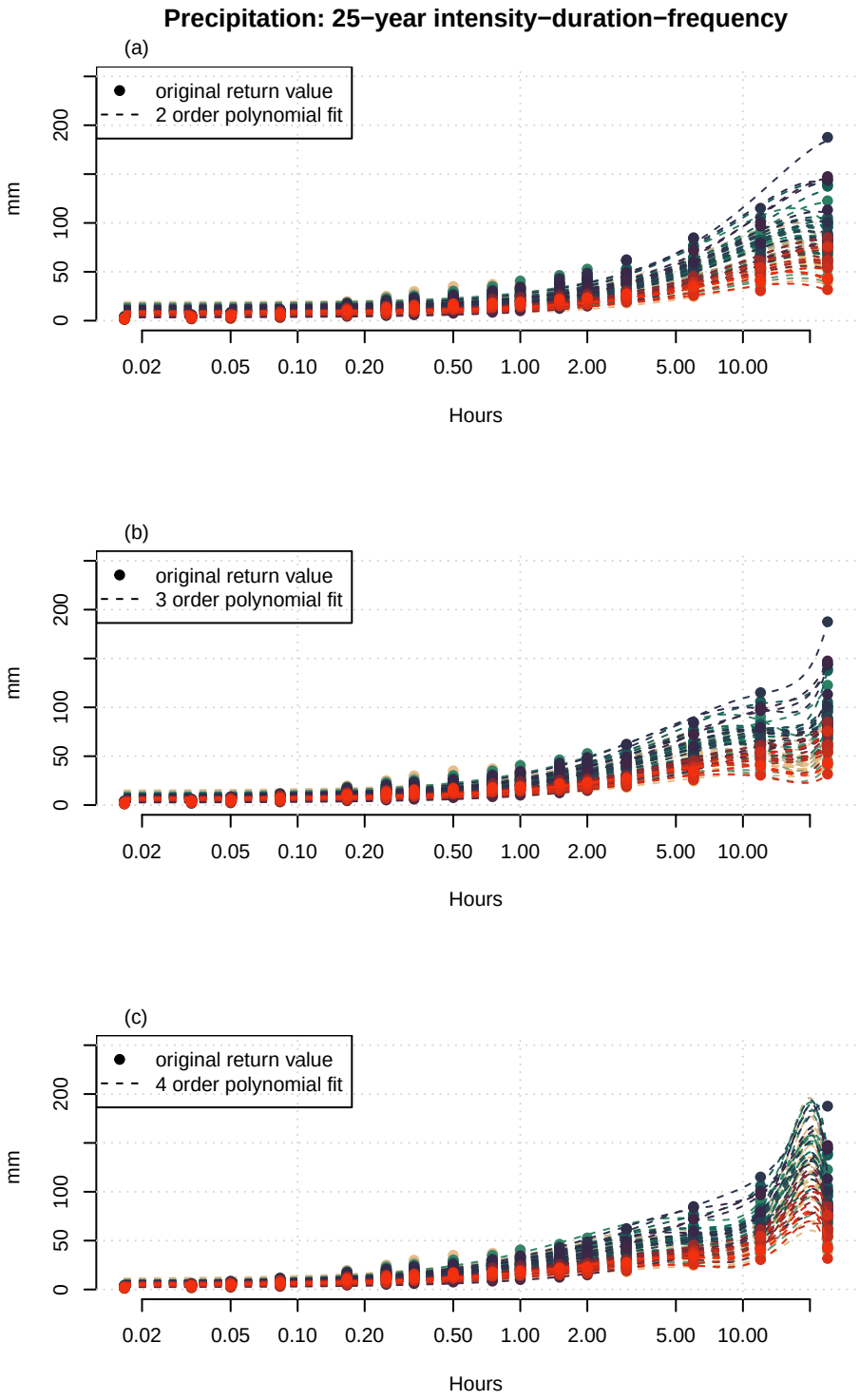

Figure S6: Comparison between the original 25 year IDF curves and a) 2nd order, b) 3rd order and c) 4th order polynomial fits to these data. The colors represend different stations as shown in the map in Figure S1.

Try with some text here.

Principal component analysis of IDF curves

Principal component analysis (PCA) is performed here by singular value decomposition (SVD). This organizes the data in modes of covariance, ranked according to the fraction of variance for which they can account. This is a way to reduce the volume size of the data and extract the embedded information. One benefit of the PCA based approach is that it can be applied to a set of return values representing different return periods and durations in one operation.

In our case, the modes of covariance have the form of spatial patterns (pca$v in the code below), associated IDF shapes (pca$u), and eigenvalues (pca$d) that determine the relative importance of the modes.

Prepare data and apply SVD

Before applying the SVD, the IDF data are rearranged into a two-dimensional matrix, where the columns represent the stations and rows represent the return values for different return periods and durations.

```
# Create 2-dimensional matrix out of the 3-dimensional Z
Z.flat <- Z
dim(Z.flat) <- c(dim(Z)[1]*dim(Z)[2], dim(Z)[3])
# Apply SVD to IDF records
pca.Z <- svd(Z.flat)
```

Examine the principal components

How many principal components to keep

The purpose of the PCA is to reduce the dimensionality of the IDF data, so that instead of analysing all return values for all stations, we can focus on only the leading m principal components (PCs). But how many PCs

should be retained in order to represent the important information of the data set? Here, we examine several different approaches to setting m, all of which include some measure of subjectivity. Based on the combined evaluation of the criteria considered below, a minimum of two leading PCs should be retained, but PCs 3-5 may also hold some information that cannot be considered only noise.

Explained variance One common rule of thumb when evaluating how many PCs to keep is the smallest number for which the cumulative explained variance (calculated from the eigenvalues) exceed some limit, e.g. 80% or 90%. The first five leading principal components of the IDF data explain 74%, 9%, 3%, 3%, and 2% of the variability of the IDF data, respectively. If we want to include the smallest number of PCs that explain at least 80% of the variance, only 2 PCs are needed, but if 90% of the variance should be retained, we need to keep 5 PCs.

```
print(paste("Explained variance:",
    paste(round(100*pca.Z$d/sum(pca.Z$d),0)[1:12], collapse=", ")))
```

```
## [1] "Explained variance: 74, 9, 3, 3, 2, 1, 1, 1, 1, 1, 0, 0"
```

```
print(paste("Cumulative:",
    paste(round(100*cumsum(pca.Z$d)/sum(pca.Z$d),0)[1:12], collapse=", ")))
```

```
## [1] "Cumulative: 74, 83, 86, 89, 91, 92, 93, 94, 95, 95, 96, 96"
```

The Scree diagram Another approach is to use a so called Scree diagram, where the explained variance or eigenvalue of each PC is plotted against its rank. The limit is set left of the 'elbow' - the point in the plot where the slope changes from steep to shallow. In our case, there is an elbow between the second and third eigenvalue in the Scree plot, which implies that 2 PCs should be retained.

```
ev <- pca.Z$d/sum(pca.Z$d)
plot(seq_along(ev)[1:20], ev[1:20], xlab="Rank of eigenvalue",
     ylab="Fractional variance", type="b")
```

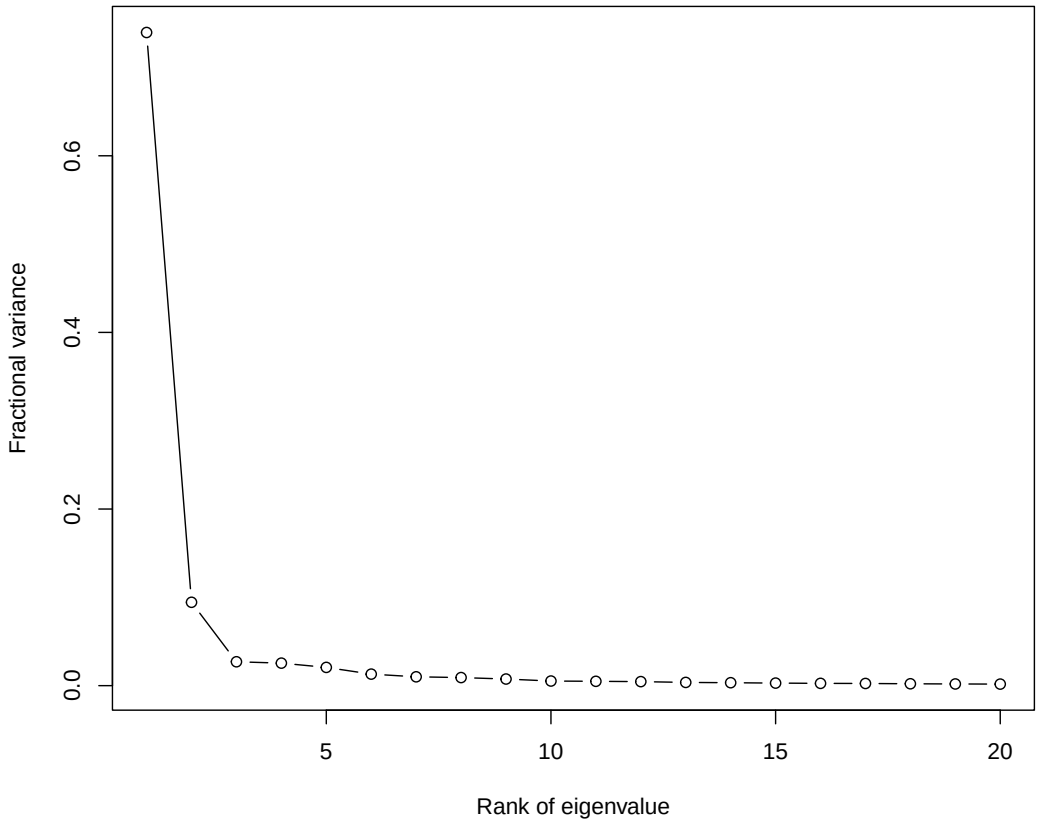

Figure S7: Scree diagram of the principal components of the IDF statistics. The figure shows the explained variance on the y-axis plotted against the rank of the eigenvalues on the x-axis.

Correlation between PCs and original variables Yet another approach is to set m to one less than the first PC for which there is no statistically significant correlation between the PC and the original variables. Based on this criterion, using a significance level of 1%, the first 4 PCs should be retained while higher order PCs can be disregarded.

Spatial patterns of the principal components

Inspect the spatial patterns associated with the two leading principal components. First visualize the spatial patterns on a map.

```
# Spatial patterns associated with the principal components
par(mar=c(0,0,0.5,2), mgp=c(0,0.75,0))
mvec <- seq(4)
for(m in mvec) {
  v.m <- pca.Z$v[,m]
  dim(v.m) <- c(1, length(v.m))
  v.m <- as.zoo(v.m, order.by=1)
  v.m <- attrcp(Z, v.m)
  class(v.m) <- c("station","zoo")
  attr(v.m,"variable") <- paste0("PC",m)
  attr(v.m,"unit") <- "unitless"
  dx <- 0.05
  breaks <- pretty(c(-1,1)*max(abs(v.m)), n=20)
  map(v.m, FUN="mean", add=m!=1, mar=c(4.5,4.5,2.5,2.5),
      fig=c(switch(m %% 2 + 1,
                   c(1/2-dx/2, 1-dx/2),
                   c(0, 1/2+dx)),
            c((2-ceiling(m/2))/2, (3-ceiling(m/2))/2)),
      main=paste0("Spatial pattern PC",m),
      xlim=c(3,35), ylim=c(55,75), cex=1.5, xlab="", ylab="",
      colbar=list(pal="t2m", show=TRUE, breaks=breaks))
  mtext(side=3, line=1, adj=0, cex=cex$mtext,
        text=paste0("(",letters[m],")"))
}
```

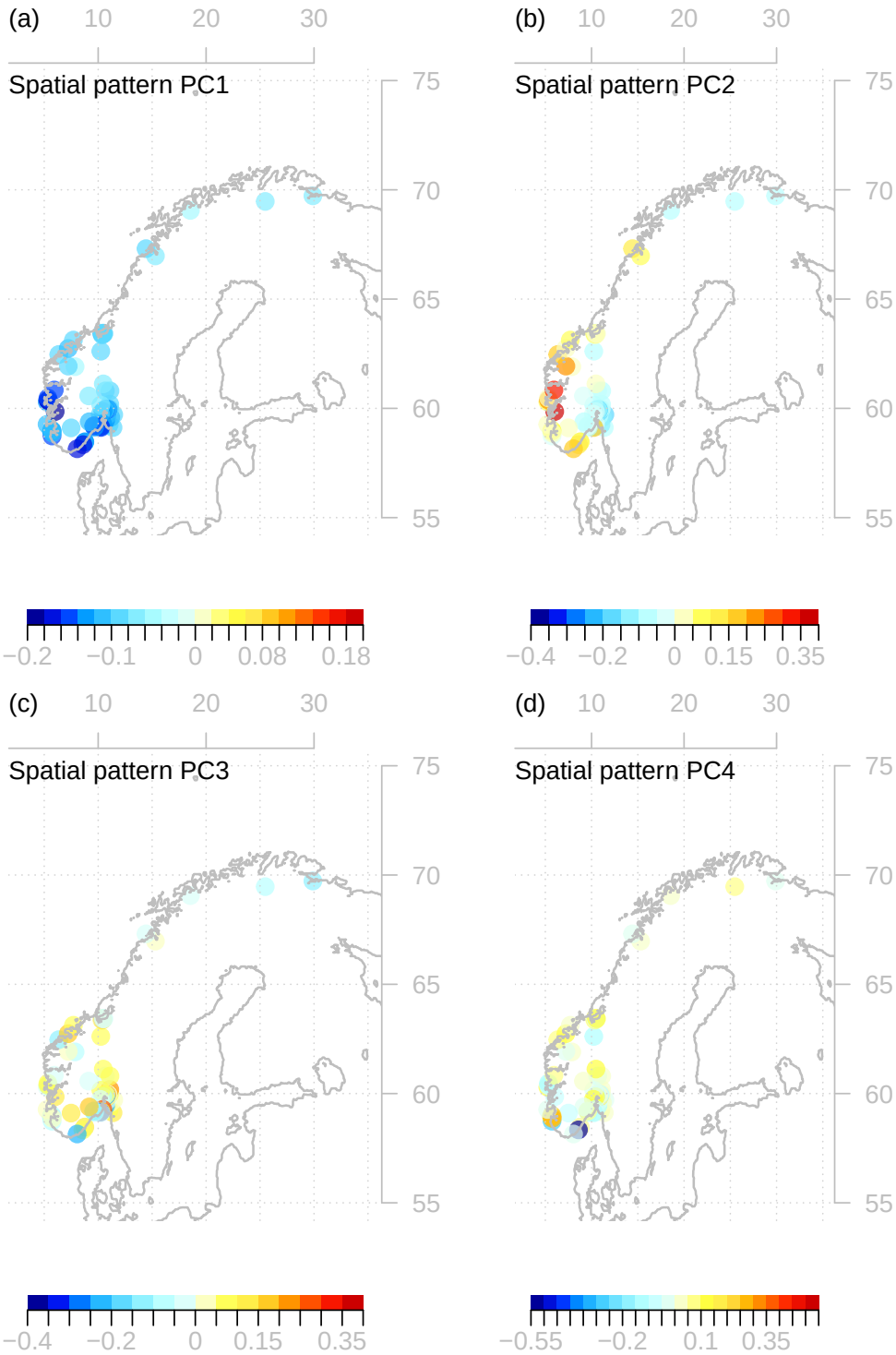

Figure S8: Spatial patterns associated with the two leading principal components of the IDF data. The spatial patterns are unitless and the different patterns do not have a common scale.

Effect of principal components on the IDF shape

The figure below demonstrates how the two leading modes of variability influence the shape of the IDF curves: the top panel shows, for the selected example stations, the original 200 year return values compared to IDF curves constructed from only the first leading principal component (PC1) and including the second (PC2), third (PC3) and fourth (PC4) leading modes. While the basic slope and level of the IDF curves are determined by PC1, PC2 alters the curvature. At most stations, PC3 and PC4 has little influence on the shape, only tweaking it slightly, but at some stations, e.g., Time - Lye (panel), the higher order modes have more influence.

```
PC1 <- pca.Z$v[,1] %*% t(pca.Z$u[,1]*pca.Z$d[1])
PC2 <- pca.Z$v[,2] %*% t(pca.Z$u[,2]*pca.Z$d[2])
PC3 <- pca.Z$v[,3] %*% t(pca.Z$u[,3]*pca.Z$d[3])
PC4 <- pca.Z$v[,4] %*% t(pca.Z$u[,4]*pca.Z$d[4])
k <- kvec[[2]]
ylim <- c(-10, max(Z[k,,i.vis], na.rm=TRUE)*1.5)
par(mfrow=c(4,2), mar=c(4,5,2,0.5), mgp=c(2.5,1,0), new=FALSE,
    cex.main=cex$main, cex.lab=cex$lab, cex.axis=cex$axis)
m <- 1
for(i in i.vis) {
  plot(dur, Z[k,,i], type='b', col="grey", lwd=2, ylim=ylim, cex=1.5,
       xlab="", ylab="Precipitation intensity (mm)",
       main=loc(Z)[i], log='x')
  mtext(side=3, line=0.8, adj=0, cex=cex$mtext,
        text=paste0("(",letters[m],")"))
  a <- PC1[i,]
  dim(a) <- dim(Z)[1:2]
  points(dur, a[k,], type='p', col=cols[2],
         cex=1.5, lwd=2, pch=21)
  a <- PC1[i,] + PC2[i,]
  dim(a) <- dim(Z)[1:2]
  points(dur, a[k,], type='p', col=cols[3],
         pch=6, cex=1.4, lwd=2)
  a <- PC1[i,] + PC2[i,] + PC3[i,]
  dim(a) <- dim(Z)[1:2]
  points(dur, a[k,], type='p', col=cols[4],
```

```
        pch=3, cex=1.3, lwd=2)
  a <- PC1[i,] + PC2[i,] + PC3[i,] + PC4[i,]
  dim(a) <- dim(Z)[1:2]
  points(dur, a[k,], type='p', col=cols[5],
        pch=8, cex=1.2, lwd=2)
  legend("topleft", lty=c(1, 0, 0, 0, 0),
        pch=c(21,21,6,3,8), lwd=c(2,2,2,2),
        legend=c(paste("original",rownames(Z)[k],
                       "year return values"),
                 "PC1", "PC1 + PC2",
                 "PC1 + PC2 + PC3",
                 "PC1 + PC2 + PC3 + PC4"),
        col=c("grey", cols[2:5]), ncol=1, cex=cex$legend,
        bty="n", bg="transparent")
  m  <- m+1
}
```

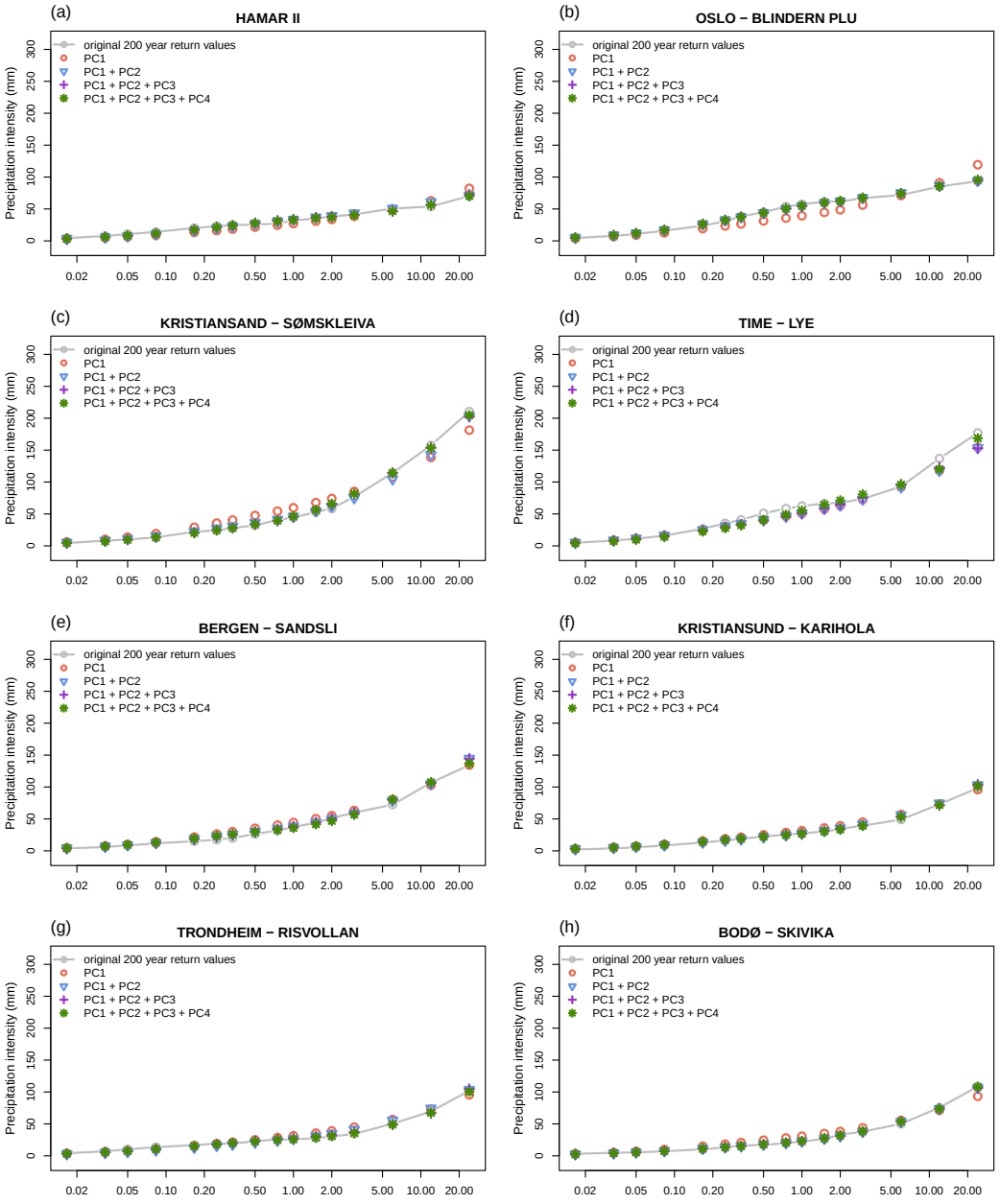

Figure S9: Examples of how the two leading principal components influence the shape of IDF curves at the stations (a) Oslo - Blindern Plu, (b) Bergen - Sandsli, (c) Trondheim - Risvollan, and (d) Bodø - Skivika. The figures show the original 200 year return values (grey) as well as return values constructed from only PC1 (orange), PC1 and PC2 (blue), PC1, PC2 and PC3 (purple), and PC1, PC2, PC3 and PC4 (green).

Statistical modeling

We want to fit statistical models relating the spatial patterns associated with the leading principal components of the IDF curves to precipitation and temperature statistics and geographical information. To do this, we will use Bayesian linear regression (the BAS library).

Preparing the predictand and predictor data

The data frame B below includes the spatial patterns associated with the two leading modes of variability of the IDF curves, here named pc1 and pc2. These will be the predictands in the model fitting. In addition, B holds a number of possible predictors: the seasonal mean values of the wet-day mean precipitation (μ), wet-day frequency (f_w) for the warm season (April - September) and the cold season (October - March), and the temperature ($T2M$) for the summer season (June - August), as well as the latitude, altitude, and distance to the ocean.

These variables describe both the summer precipitation regime, primarily regulated by convective processes, and the winter half-year precipitation regime, dominated by stratiform precipitation associated with low-pressure systems. The summer temperature is included because it is closely linked to the convective environment. The longitude is not included because it has no direct physical influence on the precipitation climate and the indirect influence should be captured by the distance to the ocean and the climatological predictor variables.

```
# Prepare data frame holding predictand and predictor data
B <- data.frame(pc1=pca.Z$v[,1], pc2=pca.Z$v[,2], pc3=pca.Z$v[,3],
       mu.cold=apply(coredata(subset(MU,it="ondjfm")),2,'mean',na.rm=TRUE),
       mu.warm=apply(coredata(subset(MU,it="amjjas")),2,'mean',na.rm=TRUE),
       fw.cold=apply(coredata(subset(FW,it="ondjfm")),2,'mean',na.rm=TRUE),
       fw.warm=apply(coredata(subset(FW,it="amjjas")),2,'mean',na.rm=TRUE),
       t2m.jja=apply(coredata(subset(T2M,it="jja")),2,'mean',na.rm=TRUE),
       lat=lat(Z), alt=alt(Z), d.ocean=attr(Z,"d.ocean"))
rownames(B) <- paste0("SN",stid(Z))
attr(B,"station_id") <- stid(Z)
```

```
attr(B,"location") <- loc(Z)
attr(B,"longitude") <- lon(Z)
attr(B,"latitude") <- lat(Z)
attr(B,"altitude") <- alt(Z)
attr(B,"npr") <- apply(coredata(X.precip),2,'nv')
attr(B,"nt2m") <- apply(coredata(X.t2m),2,'nv')
attr(B,"nv") <- attr(Z,'n')
```

Model fitting and selection

Linear models are fitted using the function bas.lm of the BAS library. Here, we use a Markov Chain Monte Carlo (MCMC) method to evaluate the probability of different models.

The results are summarized visually as:

1) An image of the model space, which shows the inclusion of the predictors in the top 20 models. Each column represents a model, with excluded parameters shown as black blocks. The x-axis shows both the model rank and the log posterior odds of the models, a measure of added value compared to a model including only the intercept.

2) A barplot displaying the posterior inclusion probability (pip) of the different predictors, which is a measure of how likely it is that the predictors are included in the true model.

3) A scatter plot comparing the original and estimated principal component values. This plot also includes summary statistics (root-mean-square error and correlation score) describing the model skill.

4) A diagnostics plots showing whether the number of MCMC iterations (N.mcmc) was enough for the posterior inclusion probability to converge. If the points fall along the 45 degree axes, the number of iterations was sufficient.

```
## Other options and settings
N.mcmc <- 1E7
modelprior <- beta.binomial(1,1)
prior <- "g-prior"
```

```r
# Function for plotting results of Bayesian linear regression
plot.mcmc <- function(fit.mcmc) {
  ## Median probability model
  fit.MPM <- predict(fit.mcmc, estimator="MPM", se.fit=TRUE)
  cex <- list(mtext=1.7, main=1.9, lab=1.9, axis=1.8, sub=1.8,
              names=1.6, text=1.6)

  ## 1) Model space
  par(mfrow=c(2,2))
  image(fit.mcmc, rotate=FALSE, new=FALSE,
        cex.lab=cex$lab, cex.axis=cex$axis, cex.sub=cex$sub)
  mtext(side=3, line=4.5, adj=0, cex=cex$mtext, text="(a)")

  ## 2) Marginal posterior inclusion probabilities (pip)
  par(mar=c(9, 8, 3, 1), mgp=c(5,1,0))
  pip <- summary(fit.mcmc)[1:fit.mcmc$n.vars,1]
  col.mcmc <- rep("grey",length(pip))
  col.mcmc[pip>0.5] <- "lightblue"
  col.mcmc[pip>0.75] <- "slateblue3"
  col.mcmc[pip>0.95] <- "navy"
  barplot(pip, width=0.2, col=col.mcmc, border=col.mcmc, las=2,
          ylab="posterior inclusion probability",
          cex.lab=cex$lab, cex.axis=cex$axis, cex.names=cex$names)
  mtext(side=3, line=1, adj=0, cex=cex$mtext, text="(b)")

  ## 3) Scatter plot
  par(mar=c(6, 6, 4, 3), mgp=c(3.5,1,0),
      cex.main=cex$main, cex.lab=cex$lab, cex.axis=cex$axis)
  ylim <- range(fit.mcmc$Y)
  ylim <- ylim + 0.2*diff(ylim)*c(-1,1)
  pci <- names(attr(fit.mcmc$terms,"dataClasses"))[[1]]
  plot(ylim, ylim, type="l", xlim=ylim, ylim=ylim,
       xlab=paste0(pci,", modeled"),
       ylab=paste0(pci,", original"))
  mtext(side=3, line=2.5, adj=0, cex=cex$mtext, text="(c)")
  f <- fit.MPM$fit
  if(!any(fit.MPM$bestmodel>0)) {
```

```
    f <- predict(bas.lm(paste0(pci," ~ 1"), data = B,
                        prior = prior, modelprior = modelprior,
                        bestmodel = c(1,1), n.models=1))$fit
  }
  points(f, fit.mcmc$Y, pch=19, col="blue")
  txt <- paste0("r = ",round(cor.test(f, fit.mcmc$Y)$estimate,2),
                " (p = ",round(cor.test(f, fit.mcmc$Y)$p.value,4),")")
  points(f, fit.mcmc$Y, pch=19, col="blue")
  text(ylim[1], ylim[2]-diff(ylim)/10, pos=4, cex=cex$text, txt)

  ## 4) Convergence plot of posterior inclusion probability
  par(mar=c(5, 5, 5, 3), mgp=c(3.5,1,0),
      cex.main=cex$main, cex.lab=cex$lab, cex.axis=cex$axis)
  diagnostics(fit.mcmc, type="pip", col = "blue", pch = 16)
  mtext(side=3, line=3.2, adj=-0.05, cex=cex$mtext, text="(d)")
}
```

```
# Plot Bayesian linear regression results for PC1
B1 <- B[,!colnames(B) %in% c("pc2","pc3")]
fit.pc1 <- bas.lm("pc1 ~ .", data=B1, modelprior=modelprior,
                  prior=prior, method="MCMC",
                  MCMC.iterations=N.mcmc)
plot.mcmc(fit.pc1)
```

```
## Warning in par(par.old): argument 1 does not name a graphical parameter
```

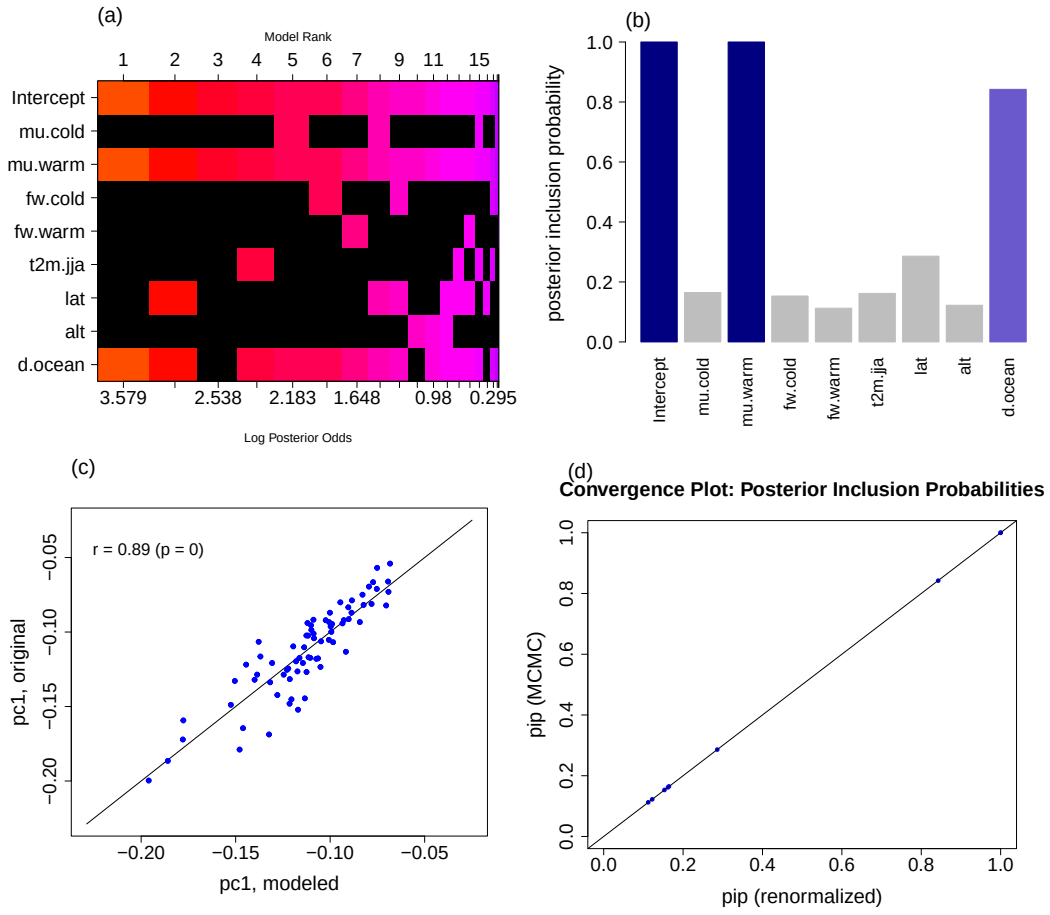

Figure S10: Summary of results of Bayesian linear regression for principal component 1 of the IDF curves. The figures present the model space (upper left), the posterior inclusion probabilities of the predictors (upper right), a scatter plot comparing original and estimated principal component values (bottom left), and a plot indicating if the number of iterations was enough for the MCMC process to converge for the posterior inclusion probabilities.

```
# Plot Bayesian linear regression results for PC2
B2 <- B[,!colnames(B) %in% c("pc1","pc3")]
fit.pc2 <- bas.lm("pc2 ~ .", data=B2, modelprior=modelprior,
                  prior=prior, method="MCMC",
                  MCMC.iterations=N.mcmc)
```

```
plot.mcmc(fit.pc2)
```

```
## Warning in par(par.old): argument 1 does not name a graphical parameter
```

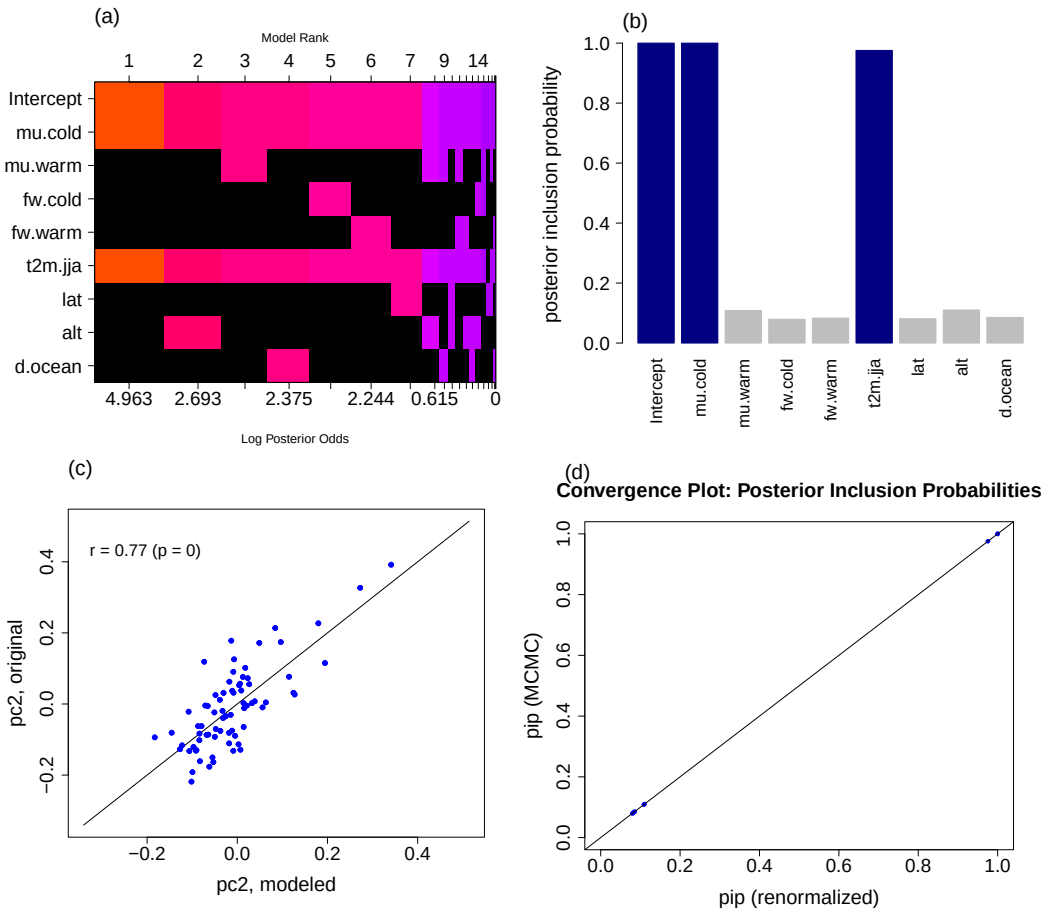

Figure S11: Summary of results of Bayesian linear regression. Like Figure S14 but for principal component 2 of the IDF curves.

```
# Plot Bayesian linear regression results for PC2
B3 <- B[,!colnames(B) %in% c("pc1","pc2")]
fit.pc3 <- bas.lm("pc3 ~ .", data=B3, modelprior=modelprior,
                  prior=prior, method="MCMC",
                  MCMC.iterations=N.mcmc)
```

```
plot.mcmc(fit.pc3)
```

```
## Warning in par(par.old): argument 1 does not name a graphical parameter
```

```
## Warning in cor(x, y): the standard deviation is zero
```

```
## Warning in cor(x, y): the standard deviation is zero
```

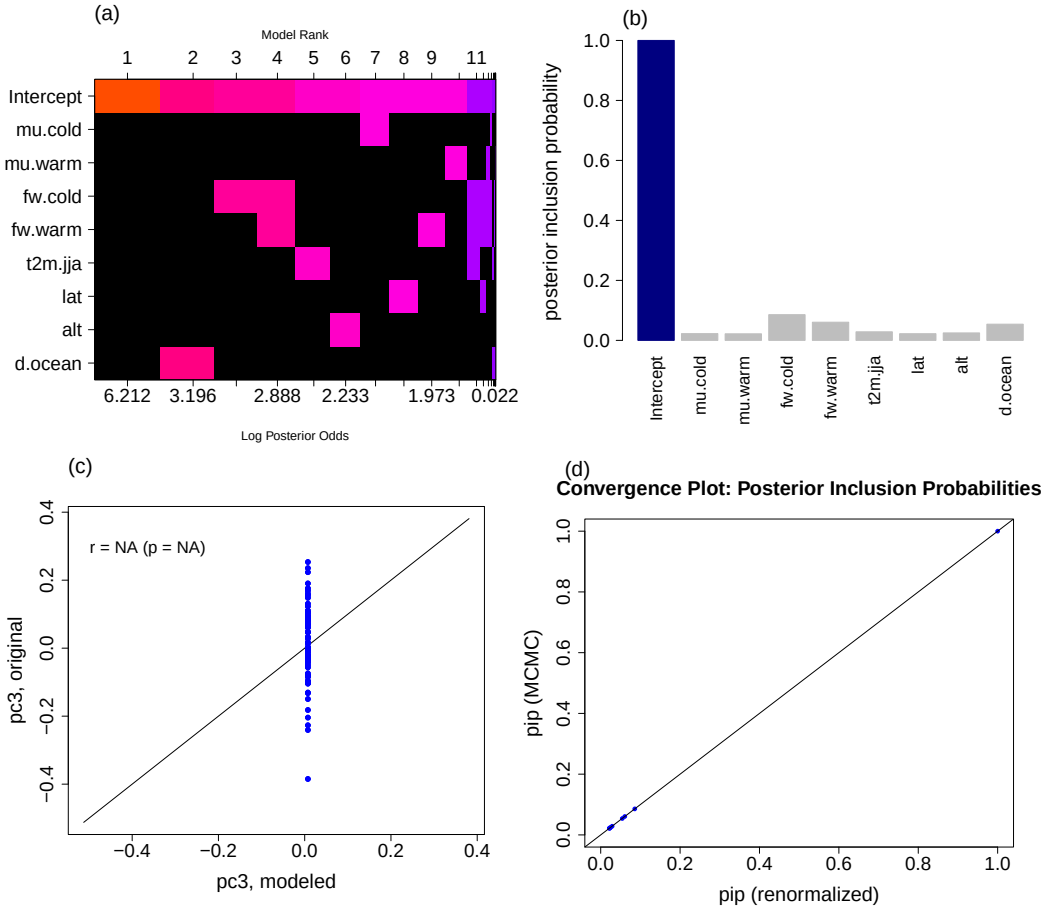

Figure S12: Summary of results of Bayesian linear regression. Like Figure S14 but for principal component 3 of the IDF curves.

```
# Re-run regression with median probability models (MPM)
force <- FALSE
file.mpm <- file.path(path.local, "IDF.MPMfit.Rd")
if(file.exists(file.mpm) & !force) {
  load(file.mpm)
} else {
  models <- list()
  fits <- list()
  ses <- list()

  mpm <- fit.pc1$namesx[predict(fit.pc1, estimator="MPM",
                                se.fit=TRUE)$bestmodel+1]
  models$pc1 <- paste0("pc1 ~ ",paste(mpm[!mpm=="Intercept"],
                                      collapse="+"))
  fits$pc1 <- bas.lm(models$pc1, data=B, modelprior=modelprior,
                     prior=prior, bestmodel=rep(1,length(mpm)),
                     n.models=1)
  ses$pc1 <- predict(fits$pc1, se.fit = TRUE)$se.pred

  mpm <- fit.pc2$namesx[predict(fit.pc2, estimator="MPM",
    se.fit=TRUE)$bestmodel+1]
  models$pc2 <- paste0("pc2 ~ ",
                       paste(mpm[!mpm=="Intercept"], collapse="+"))
  fits$pc2 <- bas.lm(models$pc2, data=B, modelprior=modelprior,
                     prior=prior, bestmodel=rep(1,length(mpm)),
                     n.models=1)
  ses$pc2 <- predict(fits$pc2, se.fit = TRUE)$se.pred

  ## Construct IDF curves from the fitted principal components
  v.fit <- cbind(predict(fits[[1]])$fit, predict(fits[[2]])$fit)
  npc <- ncol(v.fit)
  Z.fit <- t(v.fit[,1:npc] %*% diag(pca.Z$d[1:npc]) %*% t(pca.Z$u[,1:npc]))
  dim(Z.fit) <- c(dim(Z)[1:2], ncol(Z.fit))

  ## Error propagation
  se.sum <- lapply(seq(1,length(ses)), function(ip) {
    sapply(ses[[ip]], function(v) v**2*(pca.Z$d[ip]*pca.Z$u[,ip])**2)} )
```

```
  se.fit <- sqrt(se.sum[[1]] + se.sum[[2]])
  dim(se.fit) <- c(nrow(Z),ncol(Z),ncol(se.fit))

  ## Save important results
  save(file=file.mpm, Z.fit, se.fit, v.fit, fits, models)
}
```

```
# Print summary of models
pip <- round(cbind(summary(fit.pc1)[1:fit.pc1$n.vars,1],
                   summary(fit.pc2)[1:fit.pc2$n.vars,1],
                   summary(fit.pc3)[1:fit.pc3$n.vars,1]), digits=4)
colnames(pip) <- c("PC1", "PC2", "PC3")
writeLines("Marginal posterior inclusion probabilities")
print(pip)

writeLines("\n Confidence intervals of selected coefficients for PC1")
confint(coef(fits$pc1))[,1:3]
writeLines("\n Confidence intervals of selected coefficients for PC2")
confint(coef(fits$pc2))[,1:3]
```

```
## Marginal posterior inclusion probabilities
```

```
##              PC1    PC2    PC3
## Intercept 1.0000 1.0000 1.0000
## mu.cold   0.1648 1.0000 0.0225
## mu.warm   1.0000 0.1085 0.0218
## fw.cold   0.1531 0.0794 0.0857
## fw.warm   0.1124 0.0832 0.0606
## t2m.jja   0.1619 0.9756 0.0285
## lat       0.2860 0.0810 0.0222
## alt       0.1223 0.1102 0.0248
## d.ocean   0.8421 0.0854 0.0540
```

```
##
##  Confidence intervals of selected coefficients for PC1
```

```
##                     2.5%         97.5%          beta
## Intercept -1.154223e-01 -0.1089040160 -1.121631e-01
## mu.warm   -1.592513e-02 -0.0120275371 -1.397633e-02
```

```
## d.ocean    3.296741e-05  0.0001501349  9.155114e-05

##
##  Confidence intervals of selected coefficients for PC2

##                   2.5%        97.5%        beta
## Intercept -0.02850412  0.006519532 -0.01099230
## mu.cold    0.02173323  0.034200842  0.02796703
## t2m.jja   -0.03860727 -0.015039474 -0.02682337
```

Sensitivity test of predictors

Investigate the influence of the predictors on the estimated IDF curves.

```
## Function to produce plots
coefficientplots <- function(i, k=8) {
  npc <- length(fits)
  params <- unique(unlist(sapply(1:npc, function(m) {
    c(coef(fits[[m]])$namesx[coef(fits[[m]])$postmean!=0]) })))
  params <- params[params!="Intercept"]
  col <- c("black", "darkorchid3", "chocolate2")
  n.row <- 2
  n.col <- ceiling(length(params)/n.row)
  ylim <- c(0,1.5*max(Z[k,,i],na.rm=TRUE))
  par(mar=c(4,3.5,3.5,0.5), mgp=c(2.5,1,0))
  for(m in seq_along(params)) {
    p <- params[m]
    dx <- switch(gsub("[.].*","",p), "d"=200, "lat"=5, "lon"=5,
                 "alt"=500, "mu"=2, "fw"=0.2, "pr"=2, "t2m"=2)
    unit.x <- switch(gsub("[.].*","",p), "d"="km",
                     "lat"="degrees", "lon"="degrees", "alt"="m",
                     "mu"="mm", "fw"="days/month",
                     "pr"="mm", "t2m"="degC")
    X <- B
    X[[p]] <- X[[p]] + dx
    v.plus <- cbind(predict(fits[[1]], newdata=X)$fit,
                    predict(fits[[2]], newdata=X)$fit)
    Z.plus <- t(v.plus[,1:npc]%*%diag(pca.Z$d[1:npc])%*%t(pca.Z$u[,1:npc]))
```

```
    dim(Z.plus) <- c(dim(Z)[1:2], ncol(Z.plus))

    X <- B
    X[[p]] <- X[[p]] - dx
    v.minus <- cbind(predict(fits[[1]], newdata=X)$fit,
                     predict(fits[[2]], newdata=X)$fit)
    Z.minus <- t(v.minus[,1:npc]%*%diag(pca.Z$d[1:npc])%*%t(pca.Z$u[,1:npc]))
    dim(Z.minus) <- c(dim(Z)[1:2], ncol(Z.minus))

    i.row <- ceiling(m/n.col)
    i.col <- m-n.col*(i.row-1)
    par(fig=c((i.col-1)/n.col, i.col/n.col,
              (n.row-i.row)/n.row, (n.row-i.row+1)/n.row), new=m!=1)
    plot(dur, Z[k,,i], lty=1, pch=19, col="black", type="n",
         ylim=ylim, log='x',
         xlab=switch(i.row, "","Duration (hours)"),
         ylab=switch(i.col, "Precipitation intensity (mm)","",""),
         new=FALSE, cex.lab=cex$lab, cex.axis=cex$axis)
    lines(dur, Z.fit[k,,i], type="b", lty=1,
          col=col[1], pch=19, lwd=2)
    lines(dur, Z.plus[k,,i], type="l",
          col=col[2], lty=2, lwd=2)
    lines(dur, Z.minus[k,,i], type="l",
          col=col[3], lty=3, lwd=2)
    text(dur[1], max(ylim), paste0("(",letters[m],")"),
         cex=cex$mtext, adj=c(0,-1.5), xpd=NA)
    legend("topleft", col=c(col[1],col[2],col[3]),
           bty="n", bg="transparent",
           pch=c(19,46,46), lty=c(1,2,2), cex=cex$legend*0.9, ncol=1,
           legend=c(paste("Estimated return values"),
                    paste(p,"+",dx,unit.x),
                    paste(p,"-",dx,unit.x)))
  }
  mtext(paste0(rownames(Z)[k]," year return values for ",loc(Z)[i]),
        side=3, line=-1.2, outer=TRUE, cex=cex$mtext)
}
```

```
coefficientplots(i.vis[[1]])
```

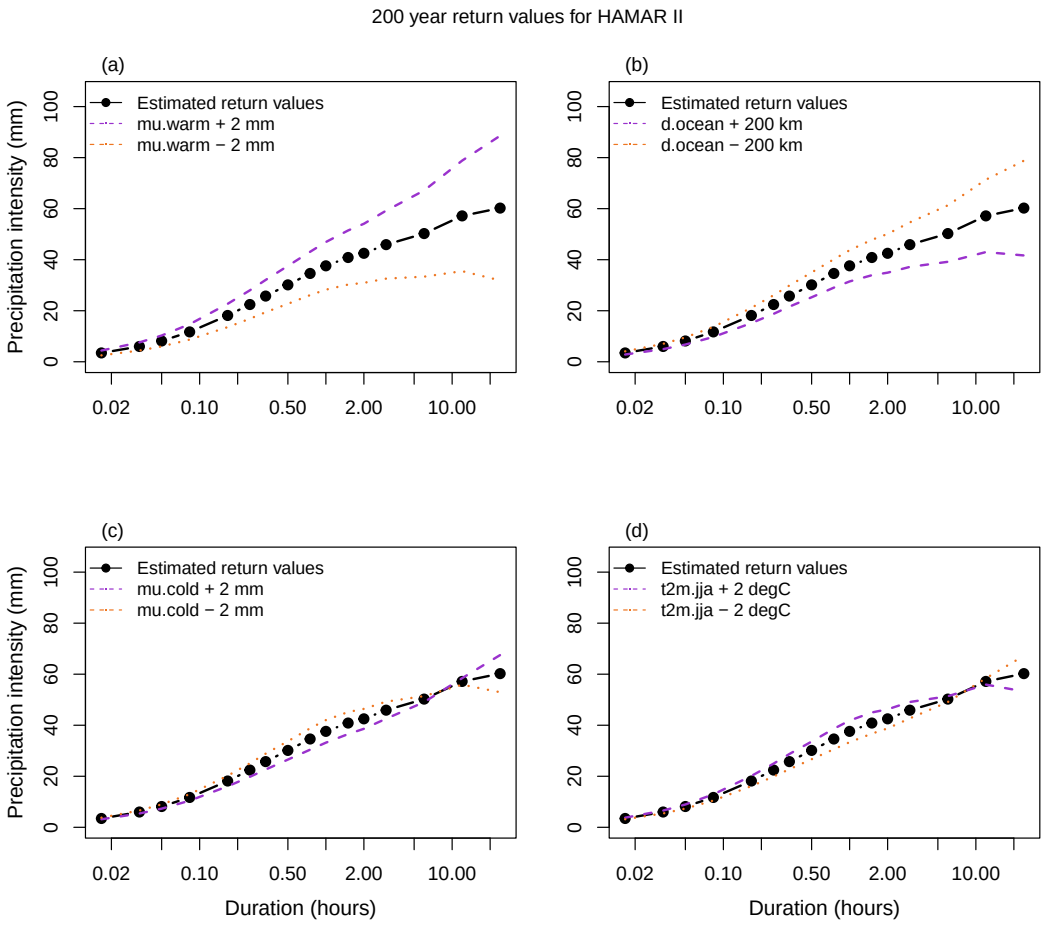

Figure S13: The plots show estimated 200 year return values for the station Hamar II and the effect of different model parameters on the estimated curves: (a) wet-day mean precipitation in the warm season, (b) distance to the ocean, (c) mean precipitation in the cold season, (d) mean temperature in the summer. The black line represents the curve estimated with paramater values from observations, and the purple and orange dashed lines are with higher and lower values of the selected parameter (details in the legends of each plot).

Skill of the statistical models

Uncertainty associated with the IDF estimates

Now we investigate the uncertainty associated with the estimated return values. Calculate and plot the confidence intervals of the estimated IDF curves based on the standard errors associated with the estimates of PC1 and PC2.

```
## Plot estimated IDF curves with confidence intervals
par(mfrow=c(4,2), mar=c(4.5, 4.5, 3, 0.5), mgp=c(3,1,0))
k <- kvec[[2]]
m <- 1
for(i in i.vis) {
  plot(dur, Z[k,,i], type="b", log='x',
       main=loc(Z)[i], col=cols[1], lty=1, pch=19, cex=1, lwd=1.5,
       ylim=ylim, cex.main=cex$main,
       cex.lab=cex$lab, cex.axis=cex$axis,
       xlab=switch(ceiling(m/6),"","Duration (hours)"),
       ylab=switch(m%%2+1, "","Precipitation intensity (mm)"))
  mtext(side=3, line=1, adj=0, cex=cex$mtext,
        text=paste0("(",letters[m],")"))
  lines(dur, Z.q25[k,,i], type="l", col=cols[1], lty=2)
  lines(dur, Z.q975[k,,i], type="l", col=cols[1], lty=2)
  lines(dur, Z.fit[k,,i], type="b", lty=1, col=cols[2],
        pch=1, cex=1, lwd=1.5)
  lines(dur, Z.fit[k,,i]-2*se.fit[k,,i], type="l", col=cols[2],
        lty=4, lwd=1.5)
  lines(dur, Z.fit[k,,i]+2*se.fit[k,,i], type="l", col=cols[2],
        lty=4, lwd=1.5)
  legend("topleft", col=c(cols[1:2],cols[1:2]),
         bty="n", bg="transparent",
         pch=c(19,1,46,46), lty=c(1,1,2,4), cex=cex$legend, ncol=1,
         legend=c(paste("Original",rownames(Z)[k],
                        "year return values"),
                  paste0("Estimated return values"),
                  paste0("95% CI original"),
                  paste0("95% CI estimated")))
```

```
  m <- m+1
}
```

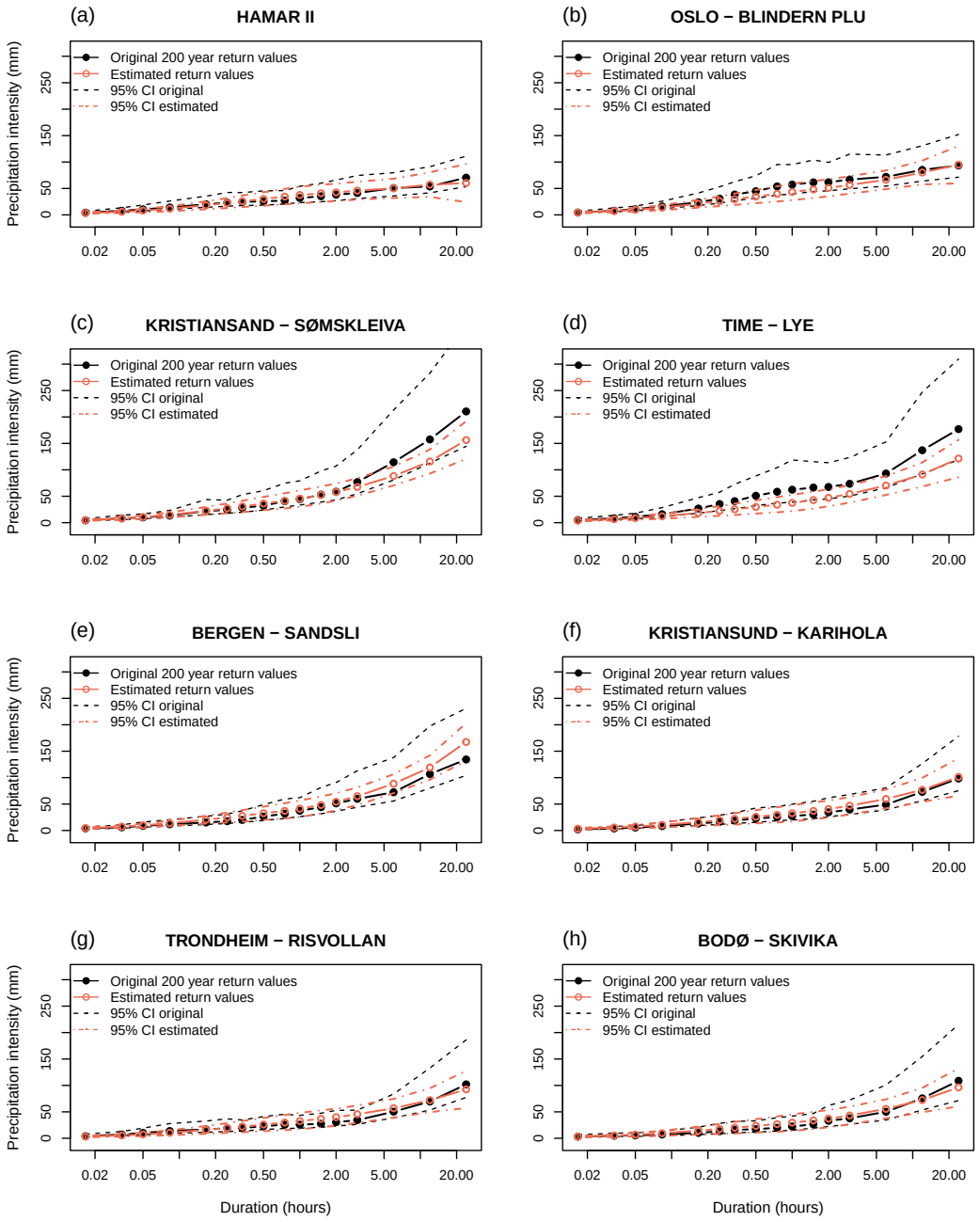

Figure S14: 50 year return values for the stations (a) Hamar II, (b) Oslo - Blindern Plu, (c) Kristiansand - Sømskleiva, (d) Time - Lye, (e) Bergen - Sandsli, (f) Kristiansund - Karihola, (e) Trondheim - Risvollan, and (g) Bodø - Skivika. The plots show the original curves (black) as well as IDF curves estimated by Bayesian linear regression (blue). Dashed lines show the confidence interval (2 standard errors) of the estimated IDF curves.

```
k <- k <- dim(Z)[[1]]
j <- dim(Z)[[2]]
col1 <- "black"; col2 <- "cornflowerblue"; col3 <- "red"
pch1 <- 19; pch2 <- 21
xticks <- seq(dim(Z)[[3]])
xticklabels <- loc(Z)
i.err <- NULL
collabels <- rep("black", length(xticks))
par(mar=c(10,5,2,2), bty="n")
plot(xticks, Z[k,j,], pch=pch1, col=col1, xaxt="n", xlab="", ylab="[mm]",
     ylim=c(10,400), main=paste0(dimnames(Z)[[1]][[k]],
                                 " year return values for ",
                                 dur[[j]], " hour duration"))
points(xticks, Z.fit[k,j,], pch=pch2, col=col2)
for(x in xticks) {
  lines(rep(x,2), Z.fit[k,j,x] + se.fit[k,j,x]*c(-2, 2),
        lty=1, pch=19, col=col2)
  if(Z[k,j,x] > Z.fit[k,j,x] + 2*se.fit[k,j,x] |
     Z[k,j,x] < Z.fit[k,j,x] - 2*se.fit[k,j,x]) {
    collabels[x] <- col3
    i.err <- c(i.err, x)
  }
}
axis(1, at=xticks, labels = FALSE)
text(xticks-0.25, -70, labels=xticklabels, srt=90, pos=1, xpd=TRUE,
     cex=0.6, col=collabels)
legend("topleft", col=c(col1, col2), pch=c(pch1, pch2),
       lty=c(NA, 1), cex=0.8,
       legend=c("Original return values",
                "Estimated return values +- 2se"))

vis.map(subset(X.precip, is=i.err), col=col3,
        add.text=FALSE, map.insert=TRUE,
        cex.axis=0.4, cex=1.5)
```

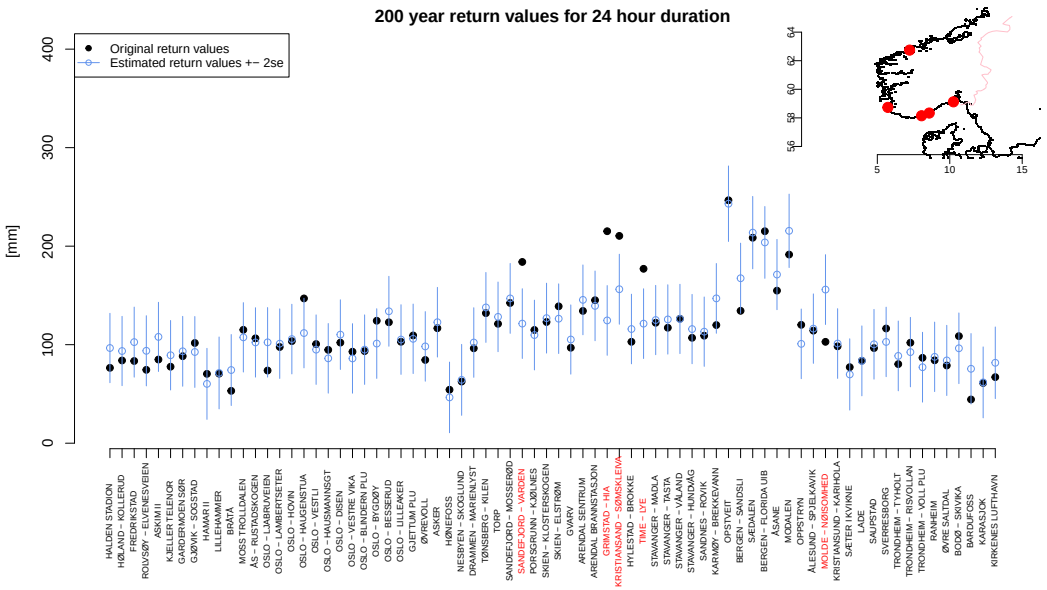

Figure S15: Return values for the return period 200 years and the duration 24 hours for 74 stations in Norway. The figure shows both the original return values (black filled circles) and estimated return values (blue empty circles) and their confidence intervals (+- 2 standard errors, shown as blue vertical lines). Stations where the original return values are outside of the confidence interval of the estimated return values are marked in red on the x-axis labels and their location are shown in the map in the top right corner.

Cross-validation of the regression models

We do a leave-one-out cross-validation to test the robustness of the modeling procedure. (The variable m changes the length of the excluded portion, e.g., m=5 gives a five fold cross-validation.)

```
m <- nrow(B)+1
exclude <- floor(seq(0,nrow(B),length.out=m))
exclude[[m]] <- nrow(B)
npc <- length(fits)
v.crossval <- matrix(NA,nrow=nrow(B),ncol=npc)
for(k in seq(1,length(exclude)-1)) {
```

```
  fit.k <- list()
  exclude.k <- seq(exclude[k]+1,exclude[k+1])
  for (i in 1:npc) {
    fit.i <- bas.lm(models[[i]], data=B[-exclude.k,],
                    prior="g-prior",
                    modelprior=beta.binomial(1,1))
    v.crossval[exclude.k,i] <-
      predict(fit.i, newdata=B[exclude.k,])$fit
  }
}

writeLines(paste(
  "Correlation between original and estimated PC",
  "\nOut-of-sample correlation for PC1:",
  round(cor.test(B[,"pc1"], v.crossval[,1])$estimate,3),
  "\nIn-sample for PC1:",
  round(cor.test(B[,"pc1"], v.fit[,1])$estimate,3),
  "\nOut-of-sample for PC2:",
  round(cor.test(B[,"pc2"], v.crossval[,2])$estimate,3),
  "\nIn-sample for PC2:",
  round(cor.test(B[,"pc2"], v.fit[,2])$estimate,3)
))
```

```
## Correlation between original and estimated PC
## Out-of-sample correlation for PC1: 0.882
## In-sample for PC1: 0.892
## Out-of-sample for PC2: 0.748
## In-sample for PC2: 0.769
```

```
Z.cv <- t(v.crossval[,1:npc]%*%diag(pca.Z$d[1:npc])%*%t(pca.Z$u[,1:npc]))
dim(Z.cv) <- c(dim(Z)[1:2], ncol(Z.cv))
Z.cv1 <- Z.cv
```

```
## Alternative cross-validation where SVD is applied without excluded data
Z.cv2 <- array(rep(NA, length(Z)), dim=dim(Z),
               dimnames=dimnames(Z))
for(k in seq(1,length(exclude)-1)) {
  Bk <- B[-exclude.k,]
```

```
  exclude.k <- seq(exclude[k]+1,exclude[k+1])
  Z.k <- Z[,,-exclude.k]
  Z.flat <- Z.k
  dim(Z.flat) <- c(dim(Z.k)[1]*dim(Z.k)[2], dim(Z.k)[3])
  pca.k <- svd(Z.flat)
  Bk$pc1 <- pca.k$v[,1]
  Bk$pc2 <- pca.k$v[,2]
  v.k <- array(rep(NA, npc*length(exclude.k)),
               dim=c(length(exclude.k), npc))
  for (i in 1:npc) {
    fit.i <- bas.lm(models[[i]], data=Bk, prior=prior,
                    modelprior=modelprior)
    v.k[,i] <- predict(fit.i, newdata=B[exclude.k,])$fit
  }
  Z.cv.k <- t(v.k[,1:npc]%*%diag(pca.k$d[1:npc])%*%t(pca.k$u[,1:npc]))
  dim(Z.cv.k) <- c(dim(Z)[1:2], length(exclude.k))
  Z.cv2[,,exclude.k] <- Z.cv.k
}
```

Construct return values from the cross-validated PCs.

```
# Calculate RMSE for cross-validation return values, Z.cv
Z.cv <- Z.cv2
stats.cv <- list()
stats.cv$rmse <- apply(Z-Z.cv, c(3,1), function(dz) {
  sum(sqrt(dz^2), na.rm=TRUE)/sum(is.finite(dz), na.rm=TRUE) })
stats.cv$rmser <- 100*stats.cv$rmse/apply(Z, c(3,1), mean,
                                          na.rm=TRUE)

# Calculate RMSE for estimated return value, Z.fit
stats.fit <- list()
stats.fit$rmse <- apply(Z-Z.fit, c(3,1),
  function(dz) sum(sqrt(dz^2), na.rm=TRUE)/sum(is.finite(dz), na.rm=TRUE))
stats.fit$rmser <- 100*stats.fit$rmse/apply(Z, c(3,1), mean, na.rm=TRUE)

# Construct IDF curves from original principal components
Z.pc <- t(pca.Z$v[,1:npc]%*%diag(pca.Z$d[1:npc])%*%t(pca.Z$u[,1:npc]))
dim(Z.pc) <- c(dim(Z)[1:2], ncol(Z.pc))
```

```
# Calculate RMSE for return values from Z.pc
stats.pc <- list()
stats.pc$rmse <- apply(Z-Z.pc, c(3,1), function(dz) {
  sum(sqrt(dz^2), na.rm=TRUE)/sum(is.finite(dz), na.rm=TRUE) })
stats.pc$rmser <- 100*stats.pc$rmse/apply(Z, c(3,1), mean,
                                          na.rm=TRUE)
```

Now visualize the results. Compare the original return values to estimated ones generated by the independent statistical models and statistical models calibrated with data from all stations. The results show that the statistical models are robust: the estimated return values tends to be almost the same independent of if the station was included in model calibration or not.

```
# Plot cross-validation IDF curves
par(mfrow=c(4,2), mar=c(4,4,2.5,0.5), mgp=c(3,1,0),
    cex.main=cex$main, cex.lab=cex$lab, cex.axis=cex$axis, cex=1)
k <- kvec[[2]]

m <- 0
for(i in i.vis) {
  m <- m+1
  plot(dur, Z[k,,i], xlim=c(0.01,25), ylim=ylim.idf, log='x',
       xlab=switch(ceiling(m/6), "", "Duration (hours)"),
       ylab=switch(m%%2+1, "", "Precipitation intensity (mm)"),
       main=loc(Z)[i], type="b", pch=21, lty=1, col=cols[1])
  grid()
  lines(dur, Z.pc[k,,i], type="b", pch=6, lty=2, col=cols[2])
  lines(dur, Z.fit[k,,i], type="b", pch=4, lty=2, col=cols[3])
  lines(dur, Z.cv[k,,i], type="b", pch=5, lty=2, col=cols[4])
  legend("topleft", lty=c(1,2,2,2), pch=c(21,6,4,5),
         col=cols, ncol=1, cex=0.9, bty="n", bg="transparent",
         legend=c(paste("original",rownames(Z)[k],
                        "year return values"),
                  paste(paste0("PC",1:npc), collapse=" + "),
                  "modeled PCs", "crossval PCs"))
  text(1.2, max(ylim.idf)*0.9, cex=cex$text*0.75,
       col=cols[2], pos=4,
```

```
      paste0("RMSE = ",round(stats.pc$rmse[i,k],1),
             " mm (",round(stats.pc$rmser[i,k]),"%)"))
  text(1.2, max(ylim.idf)*0.8, cex=cex$text*0.75,
      col=cols[3], pos=4,
      paste0("RMSE = ",round(stats.fit$rmse[i,k],1),
             " mm (",round(stats.fit$rmser[i,k]),"%)"))
  text(1.2, max(ylim.idf)*0.7, cex=cex$text*0.75,
      col=cols[4], pos=4,
      paste0("RMSE = ",round(stats.cv$rmse[i,k],1),
             " mm (",round(stats.cv$rmser[i,k]),"%)"))
  mtext(side=3, line=1, adj=0, cex=cex$mtext*1.3,
       text=paste0("(",letters[m],")"))
}
```

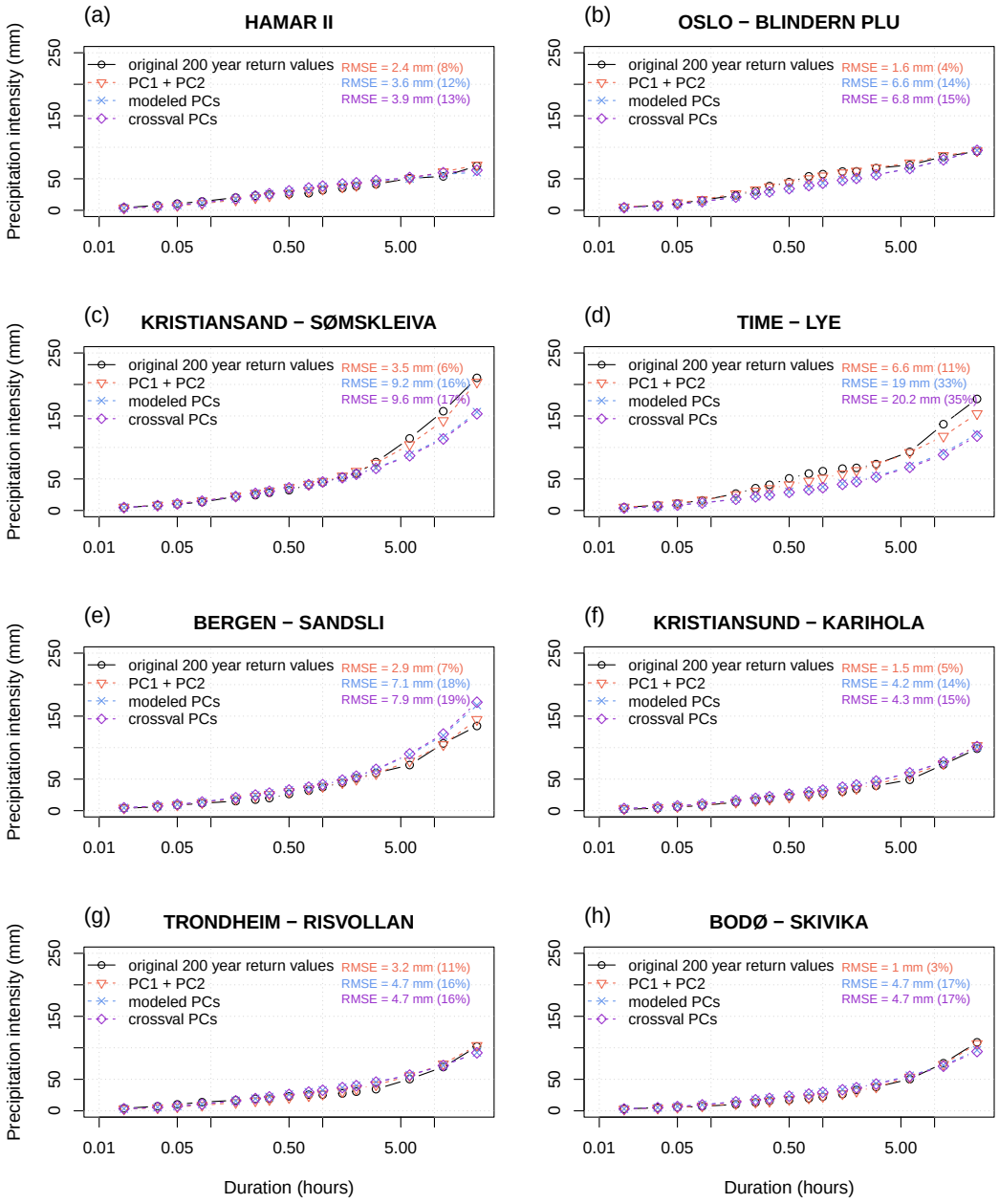

Figure S16: Original and estimated 200 year return values for (a) Hamar II, (b) Oslo - Blindern Plu, (c) Kristiansand - Sømskleiva, (d) Time - Lye, (e) Bergen - Sandsli, (f) Kristiansund - Karihola, (e) Trondheim - Risvollan, and (g) Bodø - Skivika. The plots include the original return values (black), return values reconstructed from the two leading principal components (orange), as well as return values estimated by statistical models based on all data (turquoise) and exluding the station that is being inspected (i.e., cross-validation; purple lines). The RMSE of the estimated return values based on all data (turqoise) and cross-validation (purple) are included below the figure legend.

The results of the cross-validation indicate that the statistical models are robust and the predictive skill is similar out-of-sample as in-sample.

To evaluate the performance of the statistical models, we compare the original and cross-validation return values for all return periods at the example stations.

```
k <- kvec[[2]]
par(mfrow=c(4,2), mar=c(4.6,4.6,2.6,0.5), mgp=c(3,1,0),
    cex.main=cex$main, cex.lab=cex$lab, cex.axis=cex$axis)
for(m in seq_along(i.vis)) {
  i <- i.vis[[m]]
  plot(dur, Z[1,,i], log="x", new=FALSE , type="n", ylim=ylim.idf,
       xlab=switch(m, "", "", "Duration (hours)"),
       ylab="Precipitation intensity (mm)",
       main=paste0(loc(Z)[i]," (",stid(Z)[i],")"))
  grid()
  for(k in 1:nrow(Z)) {
    lines(dur, Z[k,,i], type="b", pch=19, lty=1, col=col.rv[k])
    lines(dur, Z.cv[k,,i], type="b", pch=4, lty=2, col=col.rv[k])
  }
  mtext(side=3, adj=0, line=1, cex=cex$mtext,
        text=paste0("(",letters[m],")"))
  text(20, max(ylim.idf)*0.9, cex=cex$text, pos=2,
       labels=paste0("RMSE = ",
                     round(rmse(Z[,,i],Z.cv[,,i]),1)," mm (",
                     round(100*rmse(Z[,,i],Z.cv2[,,i])/
                             mean(Z[,,i], na.rm=TRUE)),"%)"))
  if(m==1) {
    legend("topleft", bg="transparent", lty=c(2,rep(1,length(rv))),
           pch=c(4,rep(19,length(rv))), col=c(col.rv[1], col.rv),
           ncol=1, cex=cex$legend, bty="n",
           legend=c("Estimated return values",
                    paste("Original",rv[1],"year return values"),
                    paste(rv[2:length(rv)],"year")))
  }
}
```

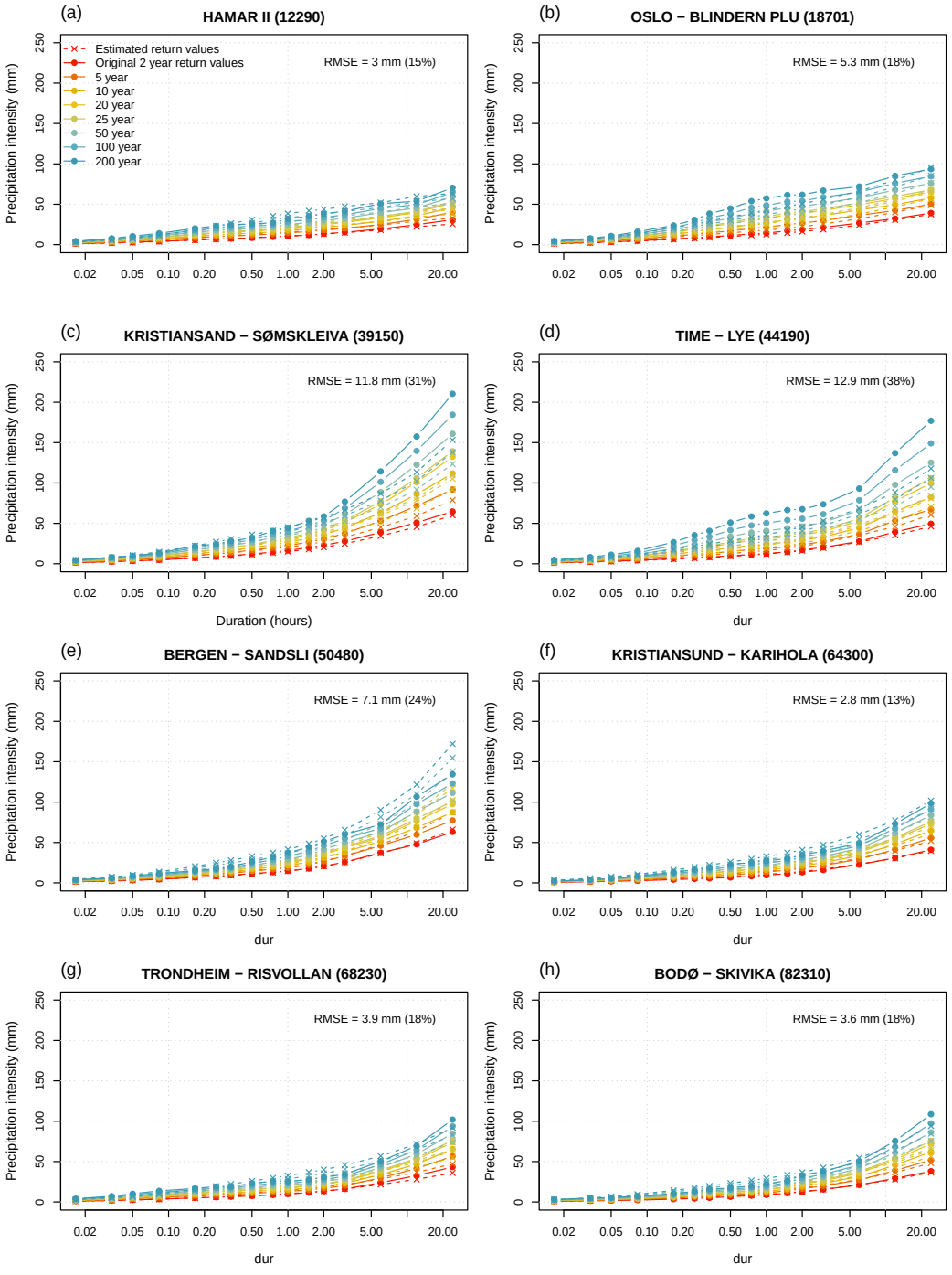

Figure S17: Original (solid lines and circles) and estimated (dashed lines and crosses) 200 year return values for the stations (a) Hamar II, (b) Oslo - Blindern Plu, (c) Kristiansand - Sømskleiva, (d) Time - Lye, (e) Bergen - Sandsli, (f) Kristiansund - Karihola, (e) Trondheim - Risvollan, and (g) Bodø - Skivika. The colors represent different return values as described in the legend.

Biases and errors of the estimated return values

Comparison with other approaches

How well does the Bayesian statistical models work and are there any biases in the estimated return values? To investigate, we compare the original return values to the cross-validation return values as well as return values obtained from the two leading original PCs, and return values estimated by a simple approximate equation defined by Benestad et al. (2021).

First, we calculate return values using the simple formula by Benestad, which is implemented in the 'esd' package as the function 'day2IDF'. The original 200 year return values at the selected example stations are then compared to return values estimated with the PCA based Bayesian inference approach, presented in this paper, and the simple formula by Benestad et al. (2021). The estimated return values are then compared to the original return values for all stations in scatterplots.

While there is some loss of information from using only the leading two PCs and discarding higher order modes, the Bayesian statistical modeling adds additional error and a bias in terms of underestimating high return values. Compared to the simple equation by Benestad, the approach with Bayesian statistical modeling of the leading PCs performs better, especially for lower return values for which the simple formula tends to overestimate the return values.

```
## Approximate IDF equation by Benestad et al. 2021
Z.b <- matrix(NA, length(Z))
dim(Z.b) <- dim(Z)
for(i in seq(ncol(X.precip))) {
  Z.i <- sapply(rv, function(x) {
    day2IDF(subset(X.precip, is=i), L=dur, tau=x) })
  Z.b[,,i] <- aperm(Z.i, c(2,1))
}
```

```
# Plot cross-validation IDF curves
pchs <- c(21,4,8,6); ltys <- c(1,2,2,2);
par(mfrow=c(4,2), mar=c(4,4,2.5,0.5), mgp=c(3,1,0),
    cex.main=cex$main, cex.lab=cex$lab, cex.axis=cex$axis, cex=1.2)
```

```
k <- kvec[[2]]
m <- 0
for(i in i.vis) {
  m <- m+1
  plot(dur, Z[k,,i], xlim=c(0.01,25), ylim=ylim.idf, log='x',
       xlab=switch(as.numeric(m>1)+1,"","Duration (hours)"),
       ylab=switch(m%%2+1, "","Precipitation intensity (mm)"),
       lwd=2, main=loc(Z)[i], type="b",
       pch=pchs[1], lty=ltys[1], col=cols[1])
  grid()
  lines(dur, Z.cv[k,,i], type="b", pch=pchs[2],
        lty=ltys[2], col=cols[2], lwd=2)
  lines(dur, Z.b[k,,i], type="b", pch=pchs[3],
        lty=ltys[3], col=cols[3], lwd=2)
  legend("topleft", lty=ltys, pch=pchs, col=cols, ncol=1, cex=0.9,
         bty="n", bg="transparent",
         legend=c(paste("original",rownames(Z)[k],"year return values"),
                  "Parding et al. 2022", "Benestad et al. 2021"))
  rmse.b <- rmse(Z[k, , i], Z.b[k, , i])
  rmser.b <- 100*rmse.b/mean(Z[k, , i], na.rm=TRUE)
  text(7E-3, max(ylim.idf)*0.55, cex=cex$text*0.8,
       col=cols[2], pos=4,
       paste0("RMSE = ",round(stats.cv$rmse[i,k],1),
              " mm (",round(stats.cv$rmser[i,k]),"%)"))
  text(7E-3, max(ylim.idf)*0.45, cex=cex$text*0.8,
       col=cols[3], pos=4,
       paste0("RMSE = ",round(rmse.b,1),
              " mm (",round(rmser.b),"%)"))
  mtext(side=3, line=1, adj=0, cex=cex$mtext*1.3,
        text=paste0("(",letters[m],")"))
}
```

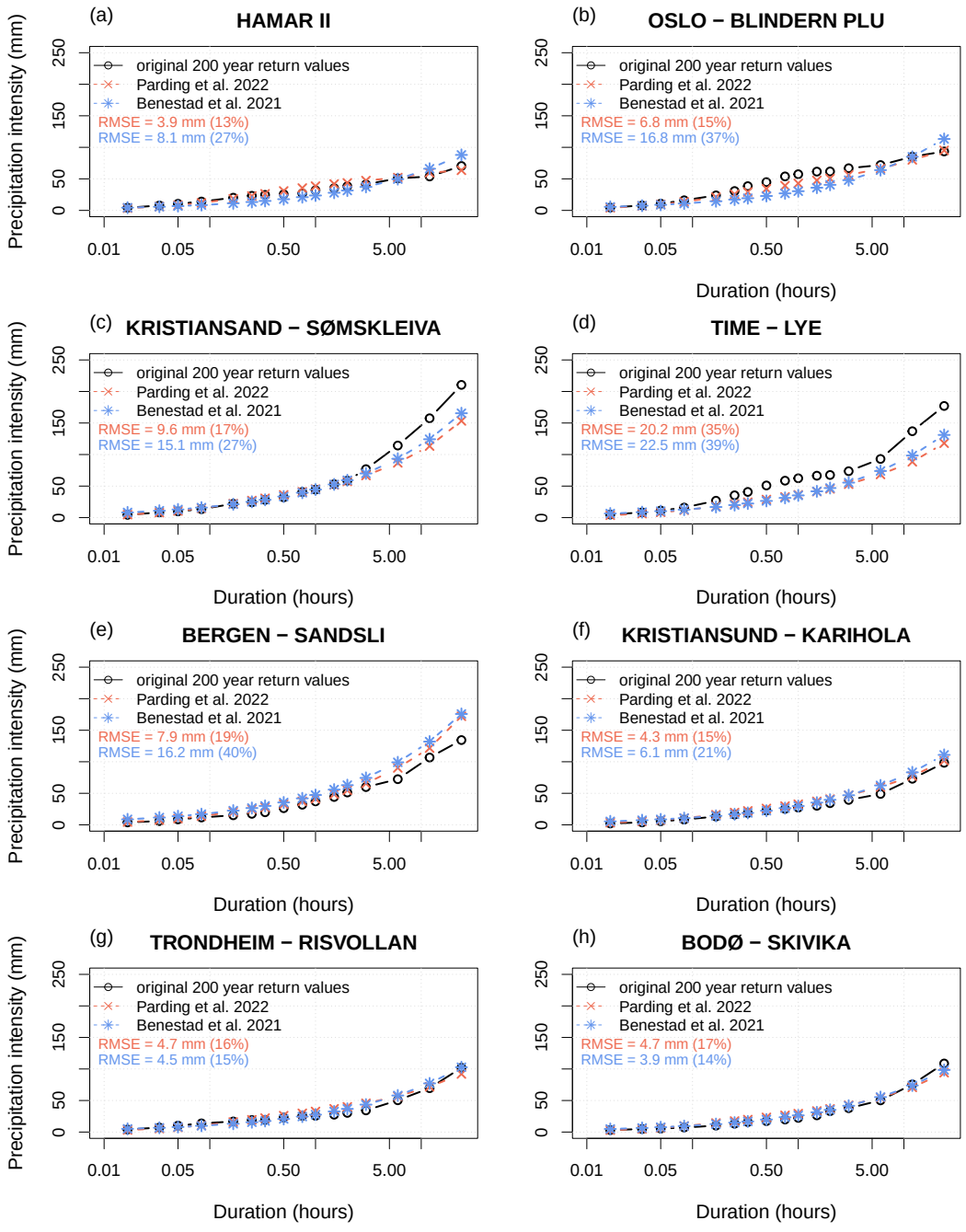

Figure S18: Original and estimated 200 year return values for (a) Hamar II, (b) Oslo - Blindern Plu, (c) Kristiansand - Sømskleiva, (d) Time - Lye, (e) Bergen - Sandsli, (f) Kristiansund - Karihola, (e) Trondheim - Risvollan, and (g) Bodø - Skivika. The plots include the original return values (black), return values estimated with the equation defined by Benestad et al. (2021) (coral) and estimated by the Bayesian statistical modeling described in this document (light blue). The RMSE of the estimated return values based on Benestad et al. (2021) (coral) and Bayesian statistical modeling (light blue) are included below the figure legend.

```
x0 <- c(-50,300); xlim=c(1,250); ylim=c(-100,100)
par(mfrow=c(3,1), mar=c(4,4.5,4,0.5), mgp=c(2.5,1,0),
    cex.lab=1.3, cex.axis=1.2, cex.main=1.2)

Z.pc <- pca2matrix(pca.Z, ip=1:2)
dim(Z.pc) <- dim(Z)
plot(x0, c(0,0), xlim=xlim, ylim=ylim,
     type="l", col="grey", lty=1, lwd=1,
     main="Reconstructed from first two PCs",
     xlab="Original return value (mm)",
     ylab="Original-estimated return value (mm)")
for(k in seq(nrow(Z),1,-1)) {
  z.k <- as.vector(Z[k,,])
  dz.k <- as.vector(Z[k,,] - Z.pc[k,,])
  points(z.k, dz.k, col=col.rv[k], pch=19, cex=0.7)
  fit.k <- lm("y ~ x", data.frame("y"=dz.k, "x"=z.k))
  lines(x0, predict(fit.k, newdata=data.frame("x"=x0)),
        col=col.rv[k], lty=2, lwd=2)
}
legend("topleft", bg="transparent", bty="n", ncol=3, cex=1.1,
       pch=c(rep(19,length(rv)),NA),
       lty=c(rep(NA, length(rv)), 2), col=c(col.rv, "black"),
       legend=c(paste(rv[1], "year return values"),
                paste(rv[2:length(rv)]," year return values"),
                "linear fit"))
mtext(side=3, line=1, adj=0, cex=cex$mtext*1.3, text="(a)")
text(0.01*diff(range(xlim)), min(ylim)+range(diff(ylim))*0.05,
     paste0("RMSE  =  ",round(RMSE(Z, Z.pc), digits=1)," mm   ( ",
            round(100*RMSE(Z, Z.pc)/mean(Z), digits=0)," % )"),
     pos=4, cex=1.1)

plot(x0, c(0,0), xlim=xlim, ylim=ylim,
     type="l", col="grey", lty=1, lwd=1,
     main="Bayesian modeling of first two PCs",
     xlab="Original return value (mm)",
     ylab="Original-estimated return value (mm)")
for(k in seq(nrow(Z),1,-1)) {
```

```
  z.k <- as.vector(Z[k,,])
  dz.k <- as.vector(Z[k,,] - Z.cv[k,,])
  points(z.k, dz.k, col=col.rv[k], pch=19, cex=0.7)
  fit.k <- lm("y ~ x", data.frame("y"=dz.k, "x"=z.k))
  lines(x0, predict(fit.k, newdata=data.frame("x"=x0)),
        col=col.rv[k], lty=2, lwd=2)
}
mtext(side=3, line=1, adj=0, cex=cex$mtext*1.3, text="(b)")
text(0.01*diff(range(xlim)), min(ylim)+range(diff(ylim))*0.05,
     paste0("RMSE  =  ",round(RMSE(Z, Z.cv), digits=1)," mm   ( ",
            round(100*RMSE(Z, Z.cv)/mean(Z), digits=0)," % )"),
     pos=4, cex=1.1)

plot(x0, c(0,0), xlim=xlim, ylim=ylim,
     type="l", col="grey", lty=1, lwd=1,
     main="Equation from Benestad et al. 2021", xlab="",
     ylab="Original-estimated return value (mm)")
for(k in seq(nrow(Z),1,-1)) {
  z.k <- as.vector(Z[k,,])
  dz.k <- as.vector(Z[k,,] - Z.b[k,,])
  points(z.k, dz.k, col=col.rv[k], pch=19, cex=0.7)
  fit.k <- lm("y ~ x", data.frame("y"=dz.k, "x"=z.k))
  lines(x0, predict(fit.k, newdata=data.frame("x"=x0)),
        col=col.rv[k], lty=2, lwd=2)
}
mtext(side=3, line=1, adj=0, cex=cex$mtext*1.3, text="(c)")
text(0.01*diff(range(xlim)), min(ylim)+range(diff(ylim))*0.05,
     paste0("RMSE  =  ",round(RMSE(Z, Z.b), digits=1)," mm   ( ",
            round(100*RMSE(Z, Z.b)/mean(Z), digits=0)," % )"),
     pos=4, cex=1.1)
```

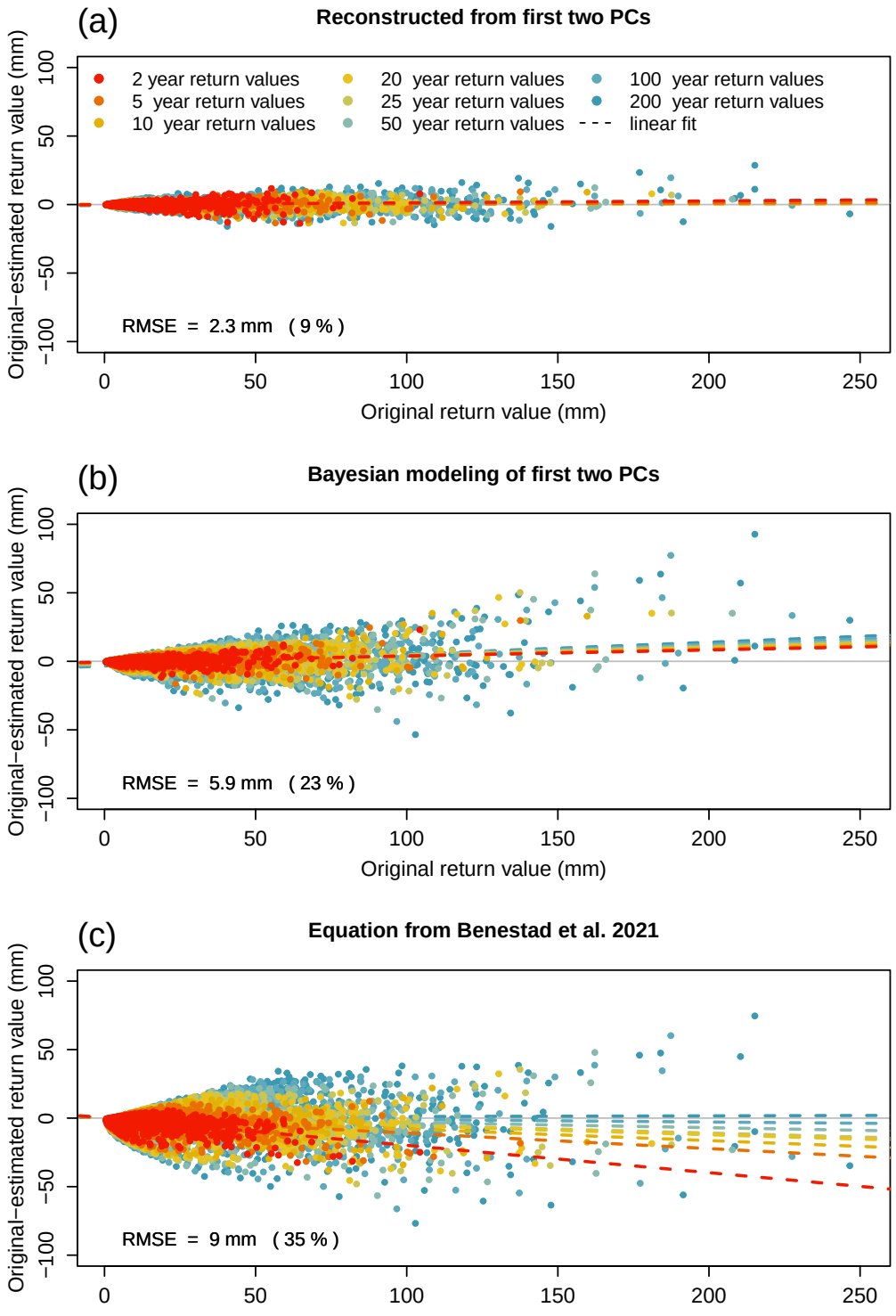

Figure S19: Original return values plotted against the error of the estimated return values (original - estimated) calculated (a) from the two leading PCs of the original return values, (b) by Bayesian statistical modeling of the two leading PCs of the return values, and (c) using the equation presented in Benestad et al. 2021. The colors represent different return periods (see legend in panel a), points show individual return values and dashed lines show linear regressions for each return period.

Spatial variations of discrepancies

Comparison between the original return values and the cross-validation estimates show that for the majority of stations, durations and return values, the RMSE is rather small (< 4mm), although there are instances of larger errors. The stations in the far north (> 65 N) tend to have a lower RMSE compared to other regions, but looking at the relative RMSE indicates that this is in large part because of low return values at these locations.

```
# Map showing the mean RMSE of the estimated return values
# Absolute RMSE
rmse.cv <- as.station(
  as.zoo(t(stats.cv$rmse), order.by=1:ncol(stats.cv$rmse)),
  lon=lon(B), lat=lat(B), param="RMSE", unit="mm")
map(rmse.cv, FUN="mean", xlim=c(3,35), ylim=c(55,75),
    fig=c(0,0.5,0,1), new=FALSE, cex=1.5,
    colbar=list(pal="rd", rev=FALSE, breaks=pretty(c(0,5),n=15)))
mtext(side=3, line=0.5, adj=0, cex=1.2, text="(a)")
# Relative RMSE
rmser.cv <- as.station(
  as.zoo(t(stats.cv$rmser), order.by=1:ncol(stats.cv$rmser)),
  lon=lon(B), lat=lat(B), param="relative RMSE", unit="%")
map(rmser.cv, FUN="mean", xlim=c(3,35), ylim=c(55,75),
    fig=c(0.5,1,0,1), new=FALSE, add=TRUE, cex=1.5,
    colbar=list(pal="rd", rev=FALSE, breaks=pretty(c(0,15),n=15)))
mtext(side=3, line=0.5, adj=0, cex=1.2, text="(b)")
```

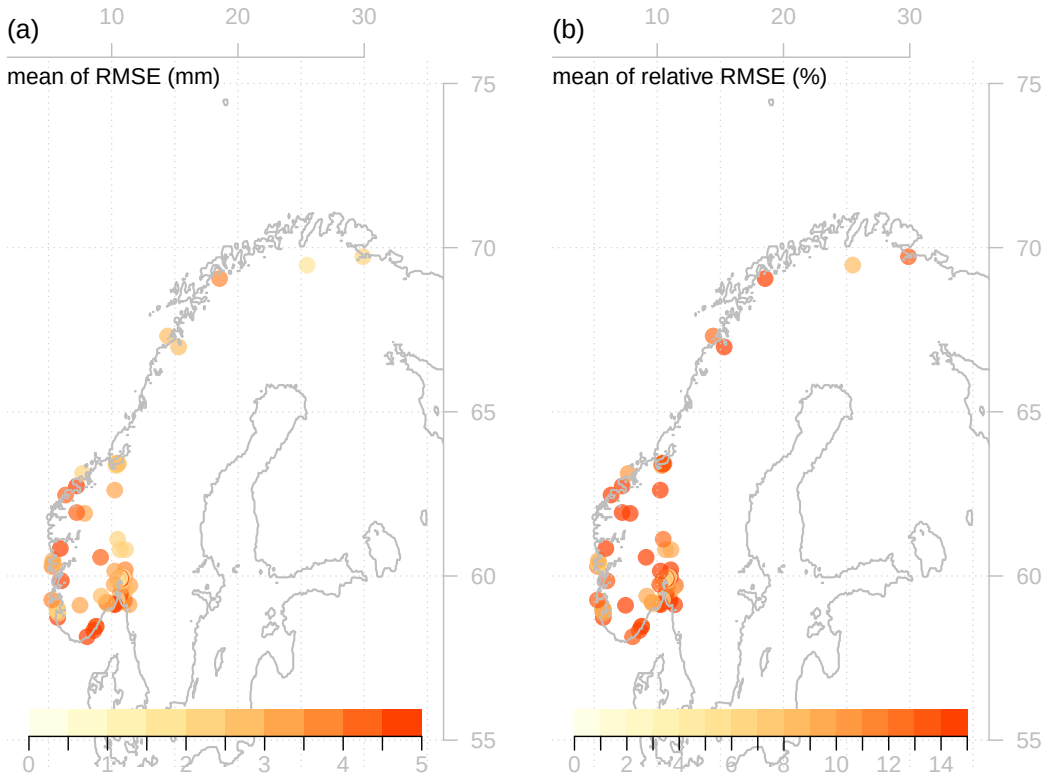

Figure S20: Map showing the mean root-mean-square error (absolute RMSE to the left and relative to the right) of the estimated return values at different sites.

Applying the statistical models to new data

The statistical models can now be applied to new data to obtain IDF estimates for stations where sub-daily precipitation data are not available.

Download predictor data

Here, we download daily temperature and precipitation data for all stations in Norway that have with at least 15 years of available data in the last 50

years.

```
# Download new data
file.new <- file.path(path.local, "metno.data.daily.all.rda")
if(file.exists(file.new)) {
  load(file.new)
} else {
  it <- c(1970,2020)
  nmin <- 10; fmin <- 0.5
  st <- select.station(src="METNOD", param="precip", nmin=nmin, it=it)
  st <- select.station(src="METNOD", param="t2m", nmin=nmin,
                       it=it, stid=st$station_id)
  pr.new <- station(src="METNOD", param="precip", stid=st$station_id,
                    it=it, verbose=FALSE)
  pr.new <- subset(pr.new, is=apply(pr.new, 2, nv)>nmin*365*fmin)
  t2m.new <- station(src="METNOD", param="t2m", stid=stid(pr.new),
                     it=it, verbose=FALSE)
  t2m.new <- subset(t2m.new, is=apply(t2m.new, 2, nv)>nmin*365*fmin)
  t2m.new <- subset(t2m.new, is=stid(t2m.new) %in% stid(pr.new))
  pr.new <- subset(pr.new, is=stid(pr.new) %in% stid(t2m.new))
  reorder <- function(X) {
    X2 <- X
    i.stid <- order(as.numeric(stid(X)))
    coredata(X2) <- coredata(X)[,i.stid]
    for(nm in names(attributes(X))) {
      if(length(attr(X, nm))==ncol(X)) {
        attr(X2, nm) <- attr(X, nm)[i.stid]
      }
    }
    return(X2)
  }
  pr.new <- reorder(pr.new)
  t2m.new <- reorder(t2m.new)
  attr(pr.new,"d.ocean") <-
    distance2ocean(lon(pr.new), lat(pr.new))$distance
  attr(t2m.new,"d.ocean") <-
    distance2ocean(lon(t2m.new), lat(t2m.new))$distance
  save(file=file.new, pr.new, t2m.new)
```

```
}
```

Calculate seasonal cycles

```
# Calculate and plot the mean seasonal cycles of the new data
PR.new <- aggregate(pr.new, month, 'mean')
MU.new <- aggregate(pr.new, month, 'wetmean', threshold=1)
FW.new <- aggregate(pr.new, month, 'wetfreq', threshold=1)
T2M.new <- aggregate(t2m.new, month, 'mean')
```

Apply statistical models to new data

```
# Prepare a data frame with new predictor data
B.new <- data.frame(
  pr.warm=apply(coredata(subset(PR.new,it="amjjas")),2,'mean',na.rm=TRUE),
  pr.cold=apply(coredata(subset(PR.new,it="ondjfm")),2,'mean',na.rm=TRUE),
  mu.warm=apply(coredata(subset(MU.new,it="amjjas")),2,'mean',na.rm=TRUE),
  mu.cold=apply(coredata(subset(MU.new,it="ondjfm")),2,'mean',na.rm=TRUE),
  fw.warm=apply(coredata(subset(FW.new,it="amjjas")),2,'mean'),na.rm=TRUE,
  fw.cold=apply(coredata(subset(FW.new,it="ondjfm")),2,'mean',na.rm=TRUE),
  t2m.jja=apply(coredata(subset(T2M.new,it="jja")),2,'mean',na.rm=TRUE),
  d.ocean=attr(pr.new, "d.ocean"),
  lat=attr(pr.new, "latitude"))
rownames(B.new) <- paste0("SN",stid(pr.new))
attr(B.new,"station_id") <- stid(pr.new)
attr(B.new,"location") <- loc(pr.new)
attr(B.new,"longitude") <- lon(pr.new)
attr(B.new,"latitude") <- lat(pr.new)
attr(B.new,"altitude") <- alt(pr.new)
attr(B.new,"npr") <- apply(coredata(pr.new),2,'nv')
attr(B.new,"nt2m") <- apply(coredata(t2m.new),2,'nv')

# Apply the statistical models to the new predictor data
npc <- length(fits)
```

```
v.new <- cbind(sapply(1:npc, function(i) {
  predict(fits[[i]], newdata=B.new)$fit }))
Z.new <- t(v.new[,1:npc]%*%diag(pca.Z$d[1:npc])%*%t(pca.Z$u[,1:npc]))
dim(Z.new) <- c(dim(Z)[1:2], ncol(Z.new))
Z.new <- attrcp(B.new, Z.new)
```

Estimated IDF curves

```
# Plot the new estimated 10 and 100 year return values
col <- wes_palette("Rushmore1", n=dim(Z.new)[3],
                   type="continuous")
par(mar=c(3,3,3,0.5), mgp=c(2,1,0), cex.main=cex$main)
for(k in kvec) {
  par(new = k!=kvec[[1]],
      fig=list(c(0,0.55,0.5,1), c(0,0.55,0,0.5))[[which(k==kvec)]])
  plot(range(dur), ylim.idf, type="n", log="x",
       main=paste(rownames(Z)[k],' year return values',sep=""),
       xlab=c("Duration (hours)", "")[k==kvec],
       ylab="Precipitation intensity (mm)")
  mtext(side=3, line=1, adj=0, cex=cex$mtext,
        text=paste0("(",letters[which(k==kvec)],")"))
  grid()
  for (i in 1:dim(Z.new)[3]) {
    points(dur, Z.new[k,,i], col=col[i], cex=0.3)
    lines(dur, Z.new[k,,i], col=col[i])
  }
}
# Map showing locations of the IDF curves
map(Oslo, xlim=c(-10,35), ylim=c(55,80),
    cex=0.1, col="grey", add=TRUE,
    main="", add.text=FALSE, new=FALSE, fig=c(0.55,1,0,1))
points(lon(Z.new), lat(Z.new), col=col, cex=1.5, pch=19)
mtext(side=3, line=1, adj=0, cex=cex$mtext, text="(c)")
```

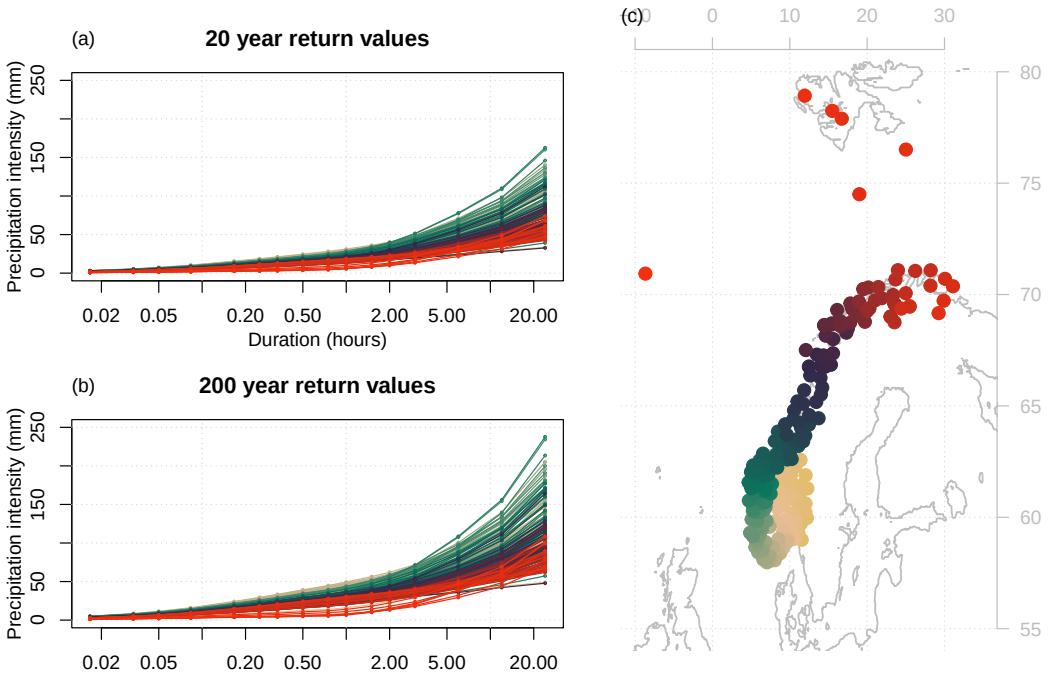

Figure S21: Estimated 10 and 200 year return values for 240 Norwegian stations, calculated using statistical models. The map on the right hand side shows which locations the colors in the IDF plots represent.

```
# Look closer at the regional differences in estimated IDFs
col.new <- wes_palette("Rushmore1", n=ncol(pr.new),
                       type="continuous")
k <- kvec[[2]]
par(mar=c(4,4,2,0.5), mgp=c(2.5,1,0))
for(l in seq_along(is.regions)) {
  is <- is.regions[[l]]
  if(is.null(is$lon)) is$lon <- round(range(lon(Z.new))+c(-1,1))
  if(is.null(is$lat)) is$lat <- range(lat(Z.new))+c(-1,1)
  i.region <- which(lon(Z.new)>is$lon[1] & lon(Z.new)<=is$lon[2] &
                    lat(Z.new)>is$lat[1] & lat(Z.new)<=is$lat[2])
  if(is$lon[1]>min(lon(Z.new)) & is$lon[2]<max(lon(Z.new))) {
    label.is <- paste0("longitude: ",is$lon[1],"-",is$lon[2])
  } else if(is$lon[1]>min(lon(Z.new))) {
```

```
    label.is <- paste("longitude >",is$lon[1])
  } else if(is$lon[2]<max(lon(Z.new))) {
    label.is <- paste("longitude <",is$lon[2])
  } else label.is <- c()
  if(is$lat[1]>min(lat(Z.new)) & is$lat[2]<max(lat(Z.new))) {
    label.is <- c(label.is,
                  paste0("latitude: ",is$lat[1],"-",is$lat[2]))
  } else if(is$lat[1]>min(lat(Z.new))) {
    label.is <- c(label.is, paste("latitude >",is$lat[1]))
  } else if(is$lat[2]<max(lat(Z.new))) {
    label.is <- c(label.is, paste("latitude <",is$lat[2]))
  }
  label.is <- paste(label.is, collapse="\n")
  par(new=(l>1), fig=switch(l,
                            "1"=c(0,0.5,0.5,1), "2"=c(0.5,1,0.5,1),
                            "3"=c(0,0.5,0,0.5), "4"=c(0.5,1,0,0.5)))
  plot(range(dur), ylim.idf, type="n", log="x",
     xlab=switch(round(l/2), "1"="", "2"="Duration (hours)"),
     ylab=switch(l %% 2 + 1, "", "Precipitation intensity (mm)"))
  if(l==1) title(paste(rownames(Z)[k],' year return values',sep=""),
                 cex=0.95)
  mtext(side=3, line=1, adj=0, cex=cex$mtext,
        text=paste0("(",letters[l],")"))
  grid()
  for (i in i.region) {
    points(dur, Z.new[k,,i], col=col.new[i], cex=0.3)
    lines(dur, Z.new[k,,i], col=col.new[i])
  }
  text(0.015, ylim.idf[2]*0.75, label.is, cex=1.1, pos=4)
}
```

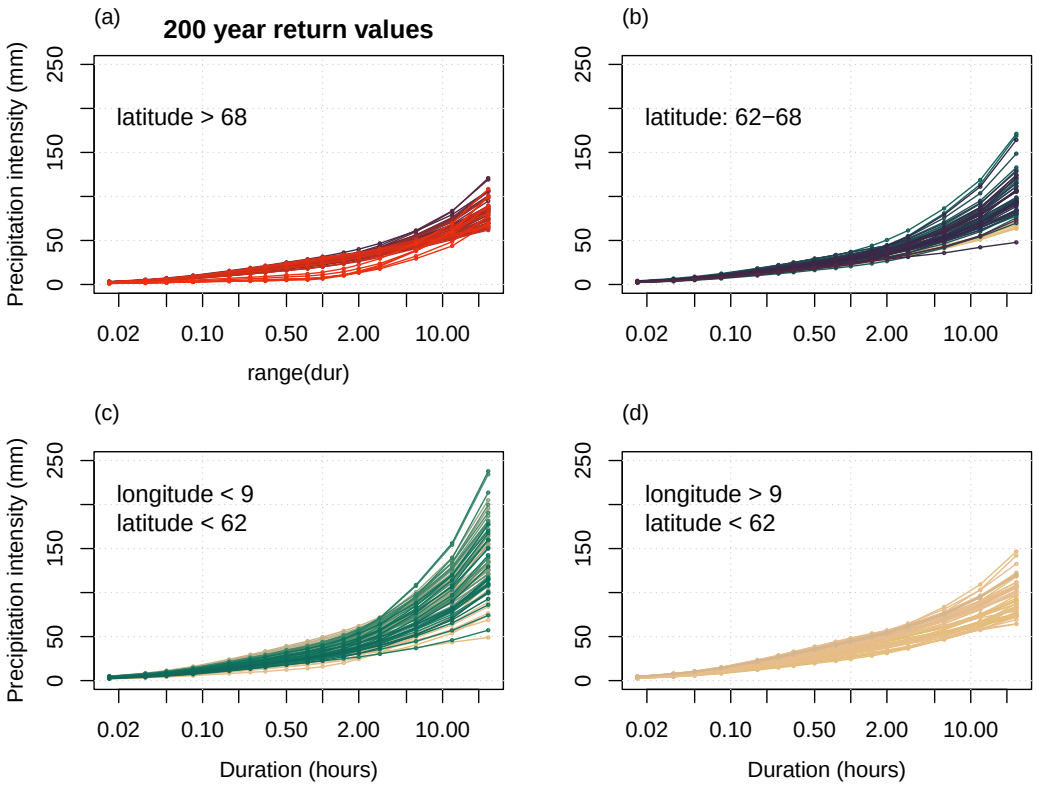

Figure S22: Estimated 200 year return values for 240 Norwegian stations, calculated using statistical models. The panels include IDF curves for different regions and the colors represent locations as depicted in Figure S28.

Comparing the original return values (Figure S1 and S2) to the estimated values (Figures S29 and S30), the estimated IDF curves are smoother than the original curves. There are obvious regional differences in the shapes of the IDF curves: Towards the west coast, the IDF curves reach the highest values and the curvature tends to be strong (i.e., a large difference between the intensity of short and long durations) compared to stations inland and up north. At northern locations, the IDF curves are lower with a moderate slope. These regional differences are more emphasized in the estimated return values. Most notably, in the south-east region (south of 62^oN, east of 9^oE), the range of estimated return values is considerably more narrow compared to the original IDF data, even though the estimated values represent more

stations covering a larger area.