Benestad et al - Upper-limit estimation of future precipitation return-values

title: "Analysis and calculations in manuscript" author: "Rasmus Benestad" date: "April 4, 2016" output: pdf_document —

R set-up

First pre-amble that checks whether the esd-package is installed and installs it if needed. If it is not installed, install it from GitHub using devtools. Also install devtools if needed. This is only done once.

```
rm(list=ls())
xlim <- c(0,35); ylim <- c(45,70)
readecad <- FALSE
figshare=TRUE
nmin=50

## Check if you need to get the esd-package:
install.esd <- ("esd" %in% rownames(installed.packages()) == FALSE)

if (install.esd) {
  print('Need to install the esd package')
  ## Need online access.
  ## Use the devtools-package for simple facilitation of installing.
  if ("devtools" %in% rownames(installed.packages()) == FALSE)
    install.packages('devtools')
  library(devtools)
  ## Install esd directly from github
  install_github('metno/esd')
  print('The latest version of esd has been installed')
}

## Start the esd-library:
library(esd)
## Information about the system and session
Sys.info()
```

```
##                                                sysname
##                                                "Linux"
##                                                release
##                                    "3.8.0-26-generic"
##                                                version
## "#38~precise2-Ubuntu SMP Thu Jun 20 18:29:36 UTC 2013"
##                                               nodename
##                                               "pc4409"
##                                                machine
##                                               "x86_64"
##                                                  login
##                                              "unknown"
##                                                   user
##                                              "rasmusb"
##                                         effective_user
##                                              "rasmusb"
```

```
sessionInfo()
```

```
## R version 3.1.3 (2015-03-09)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu precise (12.04.5 LTS)
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=nb_NO.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=nb_NO.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] esd_1.1         ncdf_1.8.6      zoo_1.7-12      ncdf4_1.15
## [5] rmarkdown_0.9.5
##
## loaded via a namespace (and not attached):
##  [1] digest_0.6.9    evaluate_0.8.3  formatR_1.3     grid_3.1.3
##  [5] htmltools_0.3   knitr_1.12.3    lattice_0.20-33 magrittr_1.5
##  [9] stringi_1.0-1   stringr_1.0.0   tools_3.1.3
```

Below are definition of a number of functions used in the data processing and analysis. The first are simple functions used in ploting.

Simple functions which estimate various statistics

Functions to extract the numbers for mean,min,max,wettest month,driest month, estimate percentiles assuming an exponential distribution, skill-scores, and regression coefficients. These are used to make the R-code simpler, shorter, and improve clarity.

```
## Return statistics for mean,min,max,wettest month,driest month
muclim <- function(x) {
  y <- coredata(x)
  iX <- mean((1:12)[is.element(y,max(y))])
  iN <- mean((1:12)[is.element(y,min(y))])
  stats <- c(mean(y,na.rm=TRUE),min(y,na.rm=TRUE),max(y,na.rm=TRUE),iX,iN)
  names(stats) <- c('mean','min','max','wettest month','driest month')
  stats
}
```

```
## Produce a set of percentiles and their counterparts for the exponential distribution
qqexp <- function(x,x0=1) {
  x[x < x0] <- NA
  if (sum(is.finite(x))>0) {
    mu <- mean(x,na.rm=TRUE)
    qx <- quantile(x,probs=seq(0,1,length=101),na.rm=TRUE)
    qy <- -log(1-seq(0,1,length=101))*mu
  } else {
    qx <- rep(NA,101); qy <- rep(NA,101)
  }
  return(cbind(qx,qy))
}
```

```
## Skill associated with predicting the wet-day mean mu
muskill <- function(x) {
  ## Estimate the skill of the calibration:
  r2 <- round(summary(lm(y ~ x,data=x))$r.squared,3)
  ## Negative slopes are not credible:
  if (summary(lm(y ~ x,data=x))$coefficients[2] < 0) r2 <- 0
  r2
}
```

```
## A function for extracting the regression coefficients and their
## error terms.
beta <- function(x,verbose=FALSE) {
  wc.model <- lm(y ~ x, data=x)
  if (verbose) print(summary(wc.model))
  beta <- summary(wc.model)$coefficients[c(2,4)]
  return(beta)
}
```

Data input and procesing

Functions for reading the data and processing it so that it can readily be handelled in the analysis.

Read the data from the CMIP5 GCMs

```
readGCMs <- function(path='CMIP5.monthly/rcp45/',pattern='tas',
                     lon=c(-100,30),lat=c(0,40)) {
    ncfiles <- list.files(path=path,pattern=pattern,full.names=TRUE)
    n <- length(ncfiles)
    print(paste(n,'netCDF files'))
    X <- matrix(rep(NA,n*201),n,201)
    for (i in 1:n) {      print(paste(i,ncfiles[i]))
      ## Spatial average:
      gcm <- annual(spatial.avg.field(C.C.eq(retrieve(ncfiles[i],
                                                    lon=lon,lat=lat))))
      i1 <- is.element(1900:2100,year(gcm))
      i2 <- is.element(year(gcm),1900:2100)
      X[i,i1] <- coredata(gcm)[i2]
    }
    ## Extract the 5 & 95 percentile and the ensemble mean:
    print('Extract the 5 & 95 percentile and the ensemble mean')
    x <- apply(X,2,function(x) c(quantile(x,probs=c(0.05,0.95),na.rm=TRUE),mean(x,na.rm=TRUE)))
    print(dim(x))
    names(x) <- c('q05','q95','mean')
    x <- zoo(t(x),order.by=1900:2100)
    w2000 <- X[,is.element(1900:2100,2000)]
    w2050 <- X[,is.element(1900:2100,2020)]
    w2100 <- X[,is.element(1900:2100,2100)]
    plot(x,plot.type='single')
```

```r
    attr(x,'path') <- path
    attr(x,'2000') <- w2000
    attr(x,'2050') <- w2050
    attr(x,'2100') <- w2100
    attr(x,'N') <- n
    attr(x,'region') <- paste(lon,lat,collapse=' ')
    return(x)
}
```

Model calibration and regression analysis

Main function that is used for calibrating the statistical model based on the mean seasonal cycle.

```r
## Calibrate a model for the wet-day mean mu using temperature as input
mucal <- function(x,pre=NULL,lon=c(-100,30),lat=c(0,40),
                  plot=FALSE,verbose=FALSE) {

## If no pre, use the crude NCEP-reanalysis provided in esd
  if (is.null(pre)) {
    if (verbose) print('default predictor')
      t2m <- t2m.NCEP(lon=lon,lat=lat)
      pre <- spatial.avg.field(C.C.eq(t2m))
      if (plot) plot(EOF(t2m))
  } else
      if (is.character(pre))
          pre <- spatial.avg.field(C.C.eq(retrieve(ncfile=pre,lon=lon,lat=lat))) else
  if (inherits(pre,'field')) {
      if (is.T(pre)) pre <- spatial.avg.field(C.C.eq(pre)) else
      pre <- spatial.avg.field(pre)
  } else if (inherits(pre,'station')) pre <- pre
  z <- aggregate(pre,by=month,FUN='mean')

  cal <- data.frame(y=coredata(x),x=coredata(z))
  attr(cal,'standard.error') <- attr(x,'standard.error')
  stats <- cor.test(cal$y,cal$x)
  wc.model <- lm(y ~ x, data=cal)
  if (plot) {
      dev.new()
      par(bty='n',cex.sub=0.7,col.sub='grey40')
      ylim <- range(cal$y,na.rm=TRUE); xlim=range(cal$x,na.rm=TRUE)
      dy <- diff(ylim)/25
      plot(cal$x,cal$y,pch=19,cex=1.5,col='grey',
           ylab=expression(paste(mu,' (mm/day)')),
         xlab=expression(paste(e[s],' (Pa)')),
         ylim=ylim,xlim=xlim,
         main='"Worst-case" fit based on seasonal variations',
         sub=paste(loc(x),' (',round(lon(x),2),'E/',round(lat(x),2),'N; ',
             alt(x),'m.a.s.l.)',sep=''))
      segments(x0=cal$x,y0=cal$y,x1=cal$x,y1=cal$y+2*attr(x,'standard.error'),
               col='grey')
      segments(x0=cal$x,y0=cal$y,x1=cal$x,y1=cal$y-2*attr(x,'standard.error'),
               col='grey')
      segments(x0=cal$x,y0=cal$y,x1=cal$x+2*attr(z,'standard.error'),y1=cal$y,
               col='grey')
      segments(x0=cal$x,y0=cal$y,x1=cal$x-2*attr(z,'standard.error'),y1=cal$y,
               col='grey')
      points(cal$x,cal$y,pch=19,cex=1.5,col='grey')
      grid()
      abline(wc.model)
      text(xlim[1],ylim[2],paste('Correlation=',round(stats$estimate,2),
                                 '(','p-value=',
                                 100*round(stats$p.value,4),'%)'),
           pos=4,cex=0.7,col='grey')
      text(xlim[1],ylim[2]-dy,paste('Regression: y=',
                                    round(wc.model$coeff[1],4), '+',
                                    round(wc.model$coeff[2],4), 'x (R2=',
                                    round(summary(wc.model)$r.squared,2),')'),
           pos=4,cex=0.7,col='grey')
      par(new=TRUE,fig=c(0.5,0.97,0.1,0.5),yaxt='n',xaxt='n',xpd=TRUE,
          cex.axis=0.7,col.axis='grey')
      plot((cal$x - mean(cal$x))/sd(cal$x),type='l',lwd=2,
           ylab='',xlab='',col=rgb(0.6,0.3,0))
      lines((cal$y - mean(cal$y))/sd(cal$y),type='l',lwd=2,col=rgb(0,0.3,0.6))
      par(xaxt = "s")
      axis(1,at=1:12,labels=month.abb,col='grey')
  }
  invisible(cal)
}
```

Functions for projection and prediction for the future/past:

```r
## Projection based on the calibration with the annual cycle:
muproject <- function(x,gcm,verbose=FALSE,prct=TRUE) {
  #print('projection')
  wcmodel <- lm(y ~ x, data=x)
  if (verbose) {
    print(summary(wcmodel))
    print(dim(gcm))
  }
  pq05 <- data.frame(x=coredata(gcm[,1]))
  pq95 <- data.frame(x=coredata(gcm[,2]))
  pmea <- data.frame(x=coredata(gcm[,3]))
  y <- cbind(predict(wcmodel,newdata=pq05),
             predict(wcmodel,newdata=pq95),
             predict(wcmodel,newdata=pmea))
  if (verbose) print(dim(y))
  if (prct) {
    ii <- is.element(year(gcm),2000:2010)
    bline <- mean(y[ii,3])
    y <- 100*y/bline
  }
  y <- zoo(y,order.by=index(gcm))
  names(y) <- names(gcm)
  return(y)
}
```

```r
## Predict values of the wet-day mean mu taking a given predictor
mupredict <- function(x,pre,verbose=FALSE,prct=TRUE) {
  if (verbose) print('projection')
  wcmodel <- lm(y ~ x, data=x)
  if (verbose) {
    print(summary(wcmodel))
  }
  eval <- data.frame(x=coredata(pre))
  y <- predict(wcmodel,newdata=eval)
  if (prct) {
    ii <- is.element(year(pre),1961:1990)
    bline <- mean(y[ii])
    if (verbose) print(length(bline))
    y <- 100*y/bline
  }
  y <- zoo(y,order.by=index(pre))
  names(y) <- 'mu'
  return(y)
}
```

Generation of graphics/maps

The following functions are used for the presentation of the results

```r
## Generate a map for the PC weights for the different modes of the
## wet-day mean mu annual cycle.
mupcamap <- function(mu,pca,ipca,xlim,ylim,r2) {
  r2 <- as.numeric(r2)
  colpc <- rev(colscal("t2m",n=100))
  cz <- round(100*abs(pca$v[,ipca])/quantile(abs(pca$v[,ipca]),0.95))
  cz[cz > 100] <- 100; cz[cz < 1] <- 1
  col <- colpc[cz]
  pch <- rep(19,length(r2))
  mo <- c(r2) > 0.4
  hi <- c(r2) > 0.6
  pch[!hi] <- 1
  pch[!mo] <- 4
  cex <- 1.5*c(r2) + 0.2

  map(mu, xlim=xlim,ylim=ylim,bg='grey80',new=FALSE)
  points(lon(mu),lat(mu),pch=pch,col=col,cex=cex)
  par(xpd=TRUE)
  text(10,73,paste('Annual cycle in PC',ipca,
      'with variance of',round(100*pca$d[ipca]^2/sum(pca$d^2)),'%'),pos=4)

  colbar(pretty(pca$v[,ipca],n=100),colpc,fig = c(0.05, 0.1, 0.05, 0.2))

  par(xaxt = "n", yaxt = "s", fig = c(0.05,0.25,0.80,0.95),
      mar = c(0, 1, 0, 0), new = TRUE, las = 1, cex.axis = 0.5,bty='n')
  plot(pca$u[,ipca],type='l',lwd=3,col="red")
  par(xaxt = "s")
  axis(1,at=1:12,labels=month.abb)
}
```

```
## Plot shaded regions
shade <- function(x,col=rgb(0.5,0.5,0.5,0.3),border=NULL) {
  t <- index(x)
  if (is.null(border)) border <- col
  y <- coredata(x)
  polygon(c(t,rev(t)),c(y[,1],rev(y[,2])),col=col,border=border)
  lines(t,y[,3],lwd=5,col=col)
}
```

```
## Estimate correlation between different sets of wet-day mean mu
## The function is designed to be used in apply for best efficiency
cormu <- function(x) {
  n <- length(x); nh <- n/2
  x1 <- x[1:nh]; x2 <- x[(nh+1):n]
  ok <- is.finite(x1) & is.finite(x2)
  return(cor(x1[ok],x2[ok]))
}
```

The definition of functions is followed by code that carry out the analysis based on these. First get the data needed - if they are not stored locally, download from Figshare where they are stored.:

```
##-------------------------------------------------------------------------------------------

## Need to obtain some of the data files - fetch from Figshare:

if (!file.exists("mu.worstcasemu.rda") & figshare) {
  download.file("http://files.figshare.com/2193033/mu.worstcasemu.rda",destfile="mu.worstcasemu.rda")
}

if (!file.exists("pre.worstcasemu.rda") & figshare) {
  download.file("http://files.figshare.com/2193038/pre.worstcasemu.rda",destfile="pre.worstcasemu.rda")
}

if (!file.exists("cmip5.rda") & figshare) {
  download.file("http://files.figshare.com/2193041/cmip5.rda",destfile="cmip5.rda")
}
```

The data can also be refreshed or updated with ECA&D data:

```
## Preparations that only needs to be done once.
if (readecad) {
## This section generates a processed data file from scratch using ECA&D data:
  pr <- station(src=c('metnod','ecad'),param='precip',nmin=nmin,it=c(1961,2010),lon=xlim,lat=ylim)
  pr <- subset(pr,it=c(1961,2014))
  nt <- apply(pr,2,FUN='nv')
  pr <- subset(pr,is=(nt >=19000))
  cpr <- coredata(pr)
  cpr[cpr > 250] <- NA
  coredata(pr) <- cpr
  save(file='pr.worstcasemu.rda',pr)
  file.remove('mu.worstcasemu.rda')
}
```

```
if (!file.exists('mu.worstcasemu.rda')) {
## pr.worstcasemu.rda is a huge file with daily precipdata based on ECA&D - generated above
  load('pr.worstcasemu.rda')

  ## Time series of the annual wet-day freq & mean - for evaluation
  ## Randomly sub-sample due to excessive volume:
  FW <- annual(pr,FUN='wetfreq',nmin=350)
  MU <- annual(pr,FUN='wetmean',nmin=350)
  fw <- aggregate(pr,month,FUN='wetfreq')
  mu <- aggregate(pr,month,FUN='wetmean')

  ## Strip away stations with a lot of missing data
  n <- apply(pr,2,FUN='nv')
  y1 <- subset(subset(pr,is=(n==max(n))),is=1)
  save(file='mu.worstcasemu.rda',mu,MU,fw,FW,y1,n)
} else load('mu.worstcasemu.rda')
```

The convention here is that the wet-day mean precipitation (precipitation intensity) is represented by symbol μ and the variable name 'mu' in the R-scripts. The wet-day frequency f_w is referret to as 'fw'. Lower and upper case refer to the mean seasonal cycle and annually aggregated values respectively. There are some stations with many missing values and some with short series. Also exclude data records with unrealistic long-term trends (due to dubious data or short series).

```
## Keep stations with no missing data and stations
## without suspect outlier trends
nok <- n
ok <- apply(mu,2,function(x) sum(!is.finite(x))==0) &
      apply(fw,2,function(x) sum(!is.finite(x))==0) &
      abs(apply(MU,2,FUN='trend.coef')) <= 1 &
      abs(apply(FW,2,FUN='trend.coef')) <= 0.02
mu <- subset(mu,is=ok) # REB 2015-03-01
MU <- subset(MU,is=ok) # REB 2015-03-01
fw <- subset(fw,is=ok)
FW <- subset(FW,is=ok)

## Remove stations with little data
nval <- apply(MU,2,nv)
mu <- subset(mu,is=(nval > 50))
MU <- subset(MU,is=(nval > 50))
fw <- subset(fw,is=(nval > 50))
FW <- subset(FW,is=(nval > 50))
nval <- apply(FW,2,nv)
mu <- subset(mu,is=(nval > 50))
MU <- subset(MU,is=(nval > 50))
fw <- subset(fw,is=(nval > 50))
FW <- subset(FW,is=(nval > 50))
```

Get and process the predictor data. Then the predictand data is processed: estimate annual mean aggregates and the mean seasonal cycle. Remove stations with large gaps of missing values.

```
print("predictor")
```

```
## [1] "predictor"
```

```
if (!file.exists('pre.worstcasemu.rda')) {
  t2m <- retrieve('air.mon.mean.nc',lon=c(-100,30),lat=c(0,40))
  pre <- spatial.avg.field(C.C.eq(t2m))
  attr(pre,'region') <- '100W,30E/0N,40N'
  save(file='pre.worstcasemu.rda',pre)
} else load('pre.worstcasemu.rda')
```

Now the data is ready for the analysis. Calibrate the regression models and extract the regression coeffieients. Use ‘apply’ to speed up the process for multiple stations.

```
## Extract the monthly aggregates for all stations
print('apply mucal')
```

```
## [1] "apply mucal"
```

```
V <- apply(mu,2,FUN='mucal',pre=pre)
Beta <- lapply(V,FUN='beta')

## Collect the R-squared statistics from lm(y ~ x) for each site
print('muskill')
```

```
## [1] "muskill"
```

```
r2 <- lapply(V,muskill)
```

Plot figure 1

Figure 1 illustrates how the mean seasonal cycle in μ and the area mean predictor compare and what the regression results for one example station.

```
## Extract the mean, min, max, wettest month, and driest months in terms of mu
## for all stations:
print('muclim')
```

```
## [1] "muclim"
```

```
X <- apply(mu,2,FUN='muclim')
print(table(X[4,]))
```

```
##
##   1   6   7   8   9  10  11  12
##   4  19 469 218 132 129  22  36
```

```
wmns <- as.numeric(rownames(table(X[4,])))
nc <- max(wmns) - min(wmns) + 1
cols <- colscal(n=nc)
col1 <- cols[c(X[4,])]
cex <- 1.5*n/max(n)

## The relationship between mu and e_s for one station
## to show the calibration procedure
is <- (1:length(n))[X[4,]==8][1]
print(paste('plot mucal - is=',is))
```

```
## [1] "plot mucal - is= 3"
```

```
mucal(subset(mu,is=is),pre=pre,verbose=TRUE,plot=TRUE)
figlab('Figure 1')
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter
```

```
## Warning in text.default(xpos, ypos, x, type = 2, cex = 1.2, pos = 4, col =
## "grey30"): graphical parameter "type" is obsolete
```

Figure 2a+b

Figure 2a+b show the principal component analysis (PCA) applied to the mean seasonal cycle for the different locations to show how it varies geographically.

```
## Remove locations with missing values for PCA
print('Mu - matrix for PCA')
```

```
## [1] "Mu - matrix for PCA"
```

```
Mu <- as.matrix(coredata(mu))

## Anomalies wrt the mean value at each location.
Mu <- apply(Mu,2,function(x) (x - mean(x)))
pca <- svd(Mu)

## Plot maps with PCs:
print('Maps with PCs')
```

```
## [1] "Maps with PCs"
```

```
mupcamap(mu,pca,1,xlim,ylim,r2)
figlab('Figure 2a')
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter
```

```
## Warning in text.default(xpos, ypos, x, type = 2, cex = 1.2, pos = 4, col =
## "grey30"): graphical parameter "type" is obsolete
```

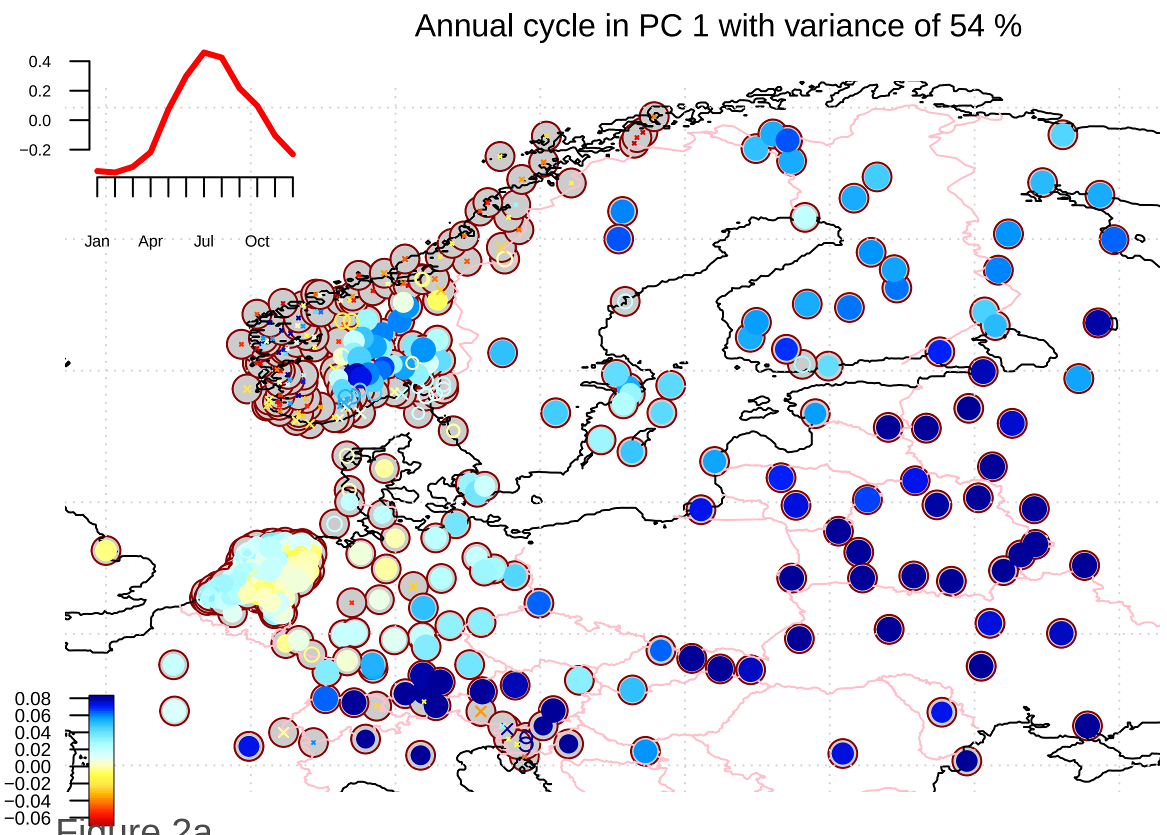
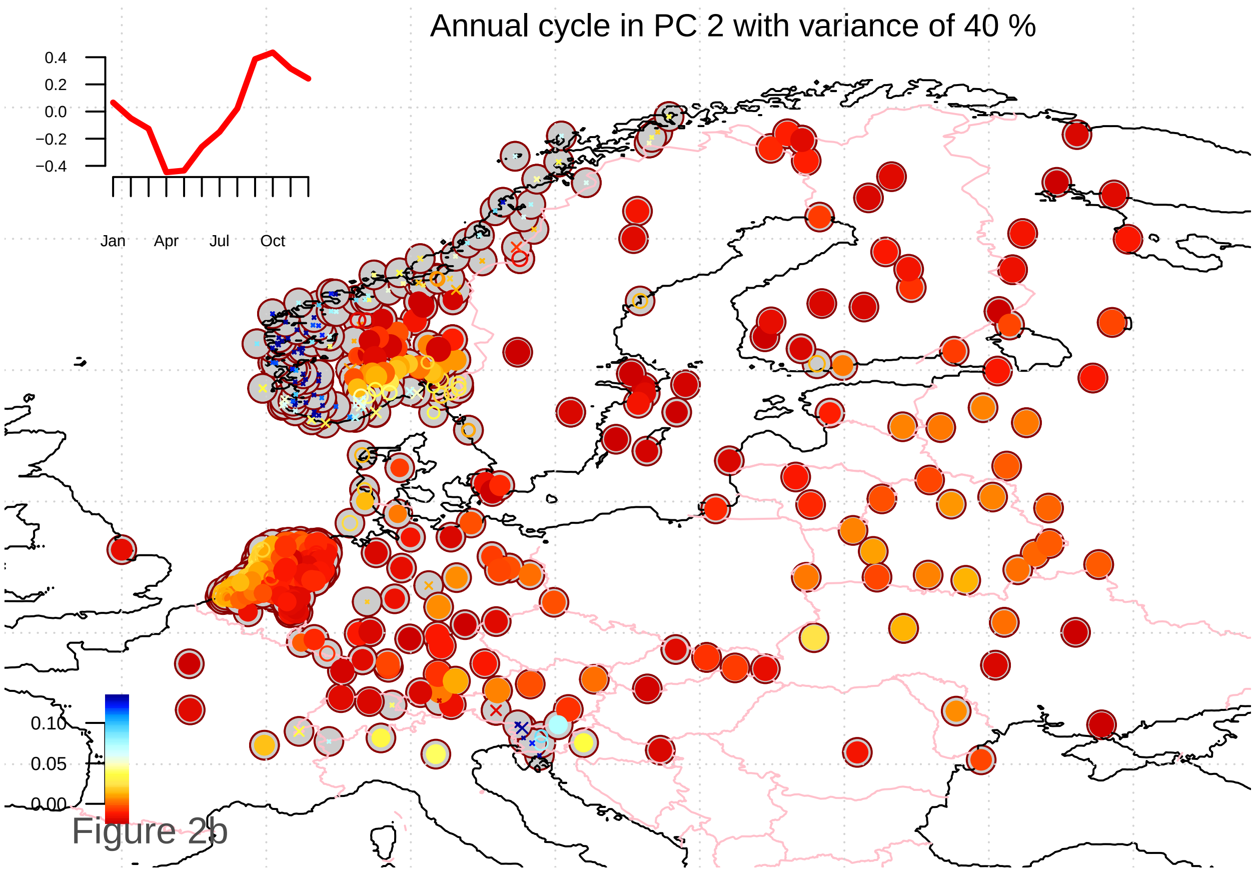


Figure 2a

```r
mupcamap(mu,pca,2,xlim,ylim,r2)
figlab('Figure 2b')
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter

## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : graphical parameter "type" is obsolete
```



Mean seasonal cycle

Examine how the components of the mean seasonal cycle correlates with the skill-score of the empirical models.

```
print('Variance accounted for by the modes:')
```

```
## [1] "Variance accounted for by the modes:"
```

```
print(round(100*pca$d**2/sum(pca$d**2),1))
```

```
##  [1] 53.5 40.2  2.6  0.9  0.7  0.5  0.5  0.4  0.3  0.2  0.2  0.0
```

```
## The sign of PCs is arbitrary...
print(cor.test(pca$v[,1],as.numeric(r2)))
```

```
##
##  Pearson's product-moment correlation
##
## data:  pca$v[, 1] and as.numeric(r2)
## t = 46.8353, df = 1027, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.8047705 0.8438465
## sample estimates:
```

```
##       cor
## 0.8252939
```

```
print(cor.test(pca$v[,2],as.numeric(r2)))
```

```
##
##  Pearson's product-moment correlation
##
## data:  pca$v[, 2] and as.numeric(r2)
## t = -49.4364, df = 1027, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.8563167 -0.8200568
## sample estimates:
##        cor
## -0.8391165
```

```
print(cor.test(pca$v[,3],as.numeric(r2)))
```

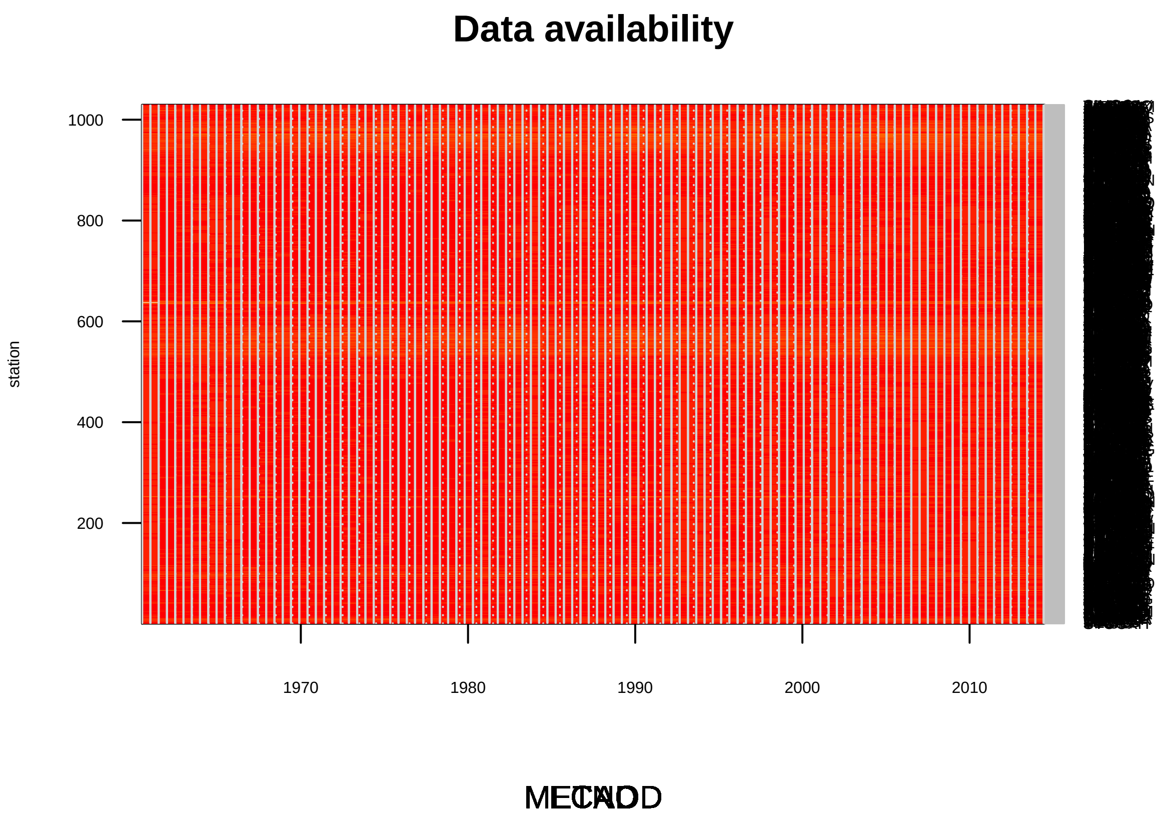
```
##
##  Pearson's product-moment correlation
##
## data:  pca$v[, 3] and as.numeric(r2)
## t = 0.0616, df = 1027, p-value = 0.9509
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.05919811  0.06302724
## sample estimates:
##         cor
## 0.001921741
```

Trend analysis

Estimate trend statistics and extract only the results for locations with a good match. The models calibrated here only seem to be valid for regions dominated by convective precipitation and are poor over regions where orographically forced precipitation dominate.

```
## Estimate trend statistics for both fw and mu:
## Use subset - in the PCA there is something strange in the 1990s.
fw.trend <- 100*apply(subset(FW,it=c(1950,1990)),2,'trend.coef')/
                      apply(FW,2,'mean',na.rm=TRUE)
mu.trend <- 100*apply(subset(MU,it=c(1950,1990)),2,'trend.coef')/
                      apply(MU,2,'mean',na.rm=TRUE)

diagnose(MU)
```



```
## Select only the sites which have an R2 greater than 0.6:
is <- (1:length(r2))[as.numeric(r2)> 0.6]
mux <- subset(mu,is=is)
MUx <- subset(MU,is=is)
FWx <- subset(MU,is=is)
Xx <- X[,is]
Vx <- V[is]
```

Aggregated results from the GCMs

Need to collect aggregated results based on the ensembles of GCM projections.

```
## Read the GCM ensembles andestimate the 5 and 95 percentiles as well as
## the ensemble mean.
## Get the annual mean temperature from GCM ensembles:
if (!file.exists('cmip5.rda')) {
  print('get CMIP5 RCP4.5')
  rcp4.5 <- readGCMs(path='CMIP5.monthly/rcp45/',pattern='tas_Amon_ens_rcp')
  print('get CMIP5 RCP8.5')
  rcp8.5 <- readGCMs(path='CMIP5.monthly/rcp85/',pattern='tas_Amon_ens_rcp')
  print('get CMIP5 RCP2.6')
  rcp2.6 <- readGCMs(path='CMIP5.monthly/rcp26/',pattern='tas_Amon_ens_rcp')
  save(file='cmip5.rda',rcp4.5,rcp8.5,rcp2.6)
} else load('cmip5.rda')

## derive time series for each location:
Z.rcp4.5 <- lapply(Vx,'muproject',rcp4.5)
Z.rcp8.5 <- lapply(Vx,'muproject',rcp8.5)
```

```
Z.rcp2.6 <- lapply(Vx,'muproject',rcp2.6)
t <- index(rcp4.5)
```

Extract the statistics of the different emission scenarios and time slices

```
## Estimates for mu in 2010:
mu2010.rcp4.5 <- lapply(Z.rcp4.5,'window',start=2010,end=2010)
mu2010.rcp8.5 <- lapply(Z.rcp8.5,'window',start=2010,end=2010)
mu2010.rcp2.6 <- lapply(Z.rcp2.6,'window',start=2010,end=2010)
x2010 <- as.numeric(lapply(mu2010.rcp4.5,function(x) x[[3]]))
x2010u <- as.numeric(lapply(mu2010.rcp4.5,function(x) x[[2]]))
y2010 <- as.numeric(lapply(mu2010.rcp8.5,function(x) x[[3]]))
y2010u <- as.numeric(lapply(mu2010.rcp8.5,function(x) x[[2]]))
z2010 <- as.numeric(lapply(mu2010.rcp2.6,function(x) x[[3]]))
z2010u <- as.numeric(lapply(mu2010.rcp2.6,function(x) x[[2]]))

## Repeat for 2100:
mu2100.rcp4.5 <- lapply(Z.rcp4.5,'window',start=2100,end=2100)
mu2100.rcp8.5 <- lapply(Z.rcp8.5,'window',start=2100,end=2100)
mu2100.rcp2.6 <- lapply(Z.rcp2.6,'window',start=2100,end=2100)
x2100 <- as.numeric(lapply(mu2100.rcp4.5,function(x) x[[3]]))
x2100u <- as.numeric(lapply(mu2100.rcp4.5,function(x) x[[2]]))
y2100 <- as.numeric(lapply(mu2100.rcp8.5,function(x) x[[3]]))
y2100u <- as.numeric(lapply(mu2100.rcp8.5,function(x) x[[2]]))
z2100 <- as.numeric(lapply(mu2100.rcp2.6,function(x) x[[3]]))
z2100u <- as.numeric(lapply(mu2100.rcp2.6,function(x) x[[2]]))

## Data frams with changes in percentages:
mu.2100 <- data.frame(mean.RCP4.5=x2100 - x2010,q95.RCP4.5=x2100u - x2010u,
                      mean.RCP2.6=z2100 - z2010,q95.RCP4.5=z2100u - z2010u,
                      mean.RCP8.5=y2100 - y2010,q95.RCP8.5=y2100u - y2010u)
```

Plot figure 3

```
## Fig 3
## Plot a map of projected values:
## Map showing RCP4.5 ensemble mean and the upper 95% change in the
## outer part of the symbol. Also an insert with box-plot diagram
## showing the other RCPs.

cols <- colscal(n=100,col='precip')
cx2100 <- round(x2100 - x2010)
cx2100u <- round(y2100u - y2010u)
cx2100[cx2100 < 1] <- 1; cx2100[cx2100 > 100] <- 100
cx2100u[cx2100u < 1] <- 1; cx2100u[cx2100u > 100] <- 100
colx <- cols[cx2100]
coly <- cols[cx2100u]
Cex <- 1.25

print("Plot a map of projected values:")
```

```
## [1] "Plot a map of projected values:"
```

```r
map(mux, xlim=xlim,ylim=ylim,cex=Cex,bg='grey70',gridlines=FALSE,
    colbar=list(col=cols,n=12,type="p",h=0.6,v=1))
points(lon(mux),lat(mux),pch=19,col=colx,cex=Cex)
points(lon(mux),lat(mux),pch=21,col=coly,cex=Cex,lwd=2)
par(xpd=TRUE)
text(20,73,'Wet-day mean: 2100')
legend(20,32,c(expression(bar(x)),expression(q[95])),
       pch=c(21,19),bty='n',col='grey',text.col='grey',horiz=TRUE)

colbar(pretty(c(x2100u- x2010u, x2100- x2010),n=15),cols,
       fig = c(0.05, 0.1, 0.05, 0.2))

par(new=TRUE,fig = c(0.05, 0.4, 0.75, 0.975),
    cex.axis=0.75,mar=c(1,1,0.1,0.1),xaxt='n')
boxplot(mu.2100,col=c(rep(rgb(0.5,0.5,0.5,0.3),2),
                      rep(rgb(0.5,1,0.5,0.3),2),
                      rep(rgb(1,0.5,0.5,0.3),2)))
par(xaxt='s')
axis(1,c(1,3,5),labels=c('RCP4.5','RCP2.6','RCP8.5'))
grid()

figlab('Figure 3')
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter
```

```
## Warning in text.default(xpos, ypos, x, type = 2, cex = 1.2, pos = 4, col =
## "grey30"): graphical parameter "type" is obsolete
```

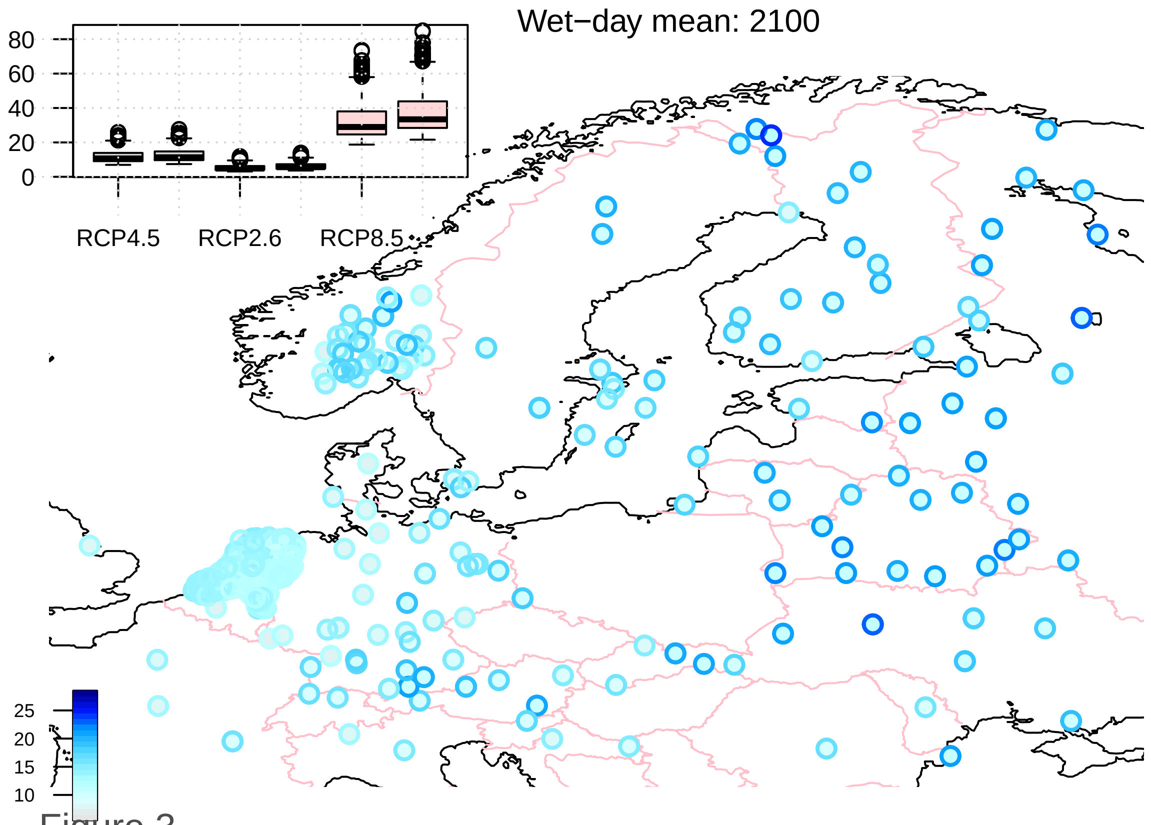


Figure 3

Supporting material/analysis (SM)

Extract the essential information

```
## Evaluation:
## Use the calibration strategy to predict the annual mu based on the
## predictor (t2m -> e_s)
print('evaluation: correlation')
```

```
## [1] "evaluation: correlation"
```

```
mu.eval <- lapply(Vx,'mupredict',annual(pre),prct=FALSE)
m <- length(mu.eval); n <- length(mu.eval[[1]])
MUz <- matrix(unlist(mu.eval),n,m)
MUz <- zoo(MUz,order.by=index(mu.eval[[1]]))
MUz <- subset(MUz,it=MUx)
MUx <- subset(MUx,it=MUz)
FWx <- subset(FWx,it=MUz)
r.eval <- apply(rbind(coredata(MUz),coredata(MUx)),2,'cormu')
print(summary(r.eval))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -0.2891  0.1177  0.2163  0.2020  0.3021  0.5728
```

** ———— **Supporting Material** ——————————**

The supporting material includes a number of diagnostics and plots that chencks a number of assumptions made in this study.

figure SM2

Presents the statistics of R^2 from the model calibration based on the different locations.

```
par(xaxt='n',yaxt='n',bty='n')
plot(c(0,1),c(0,1),type='n',xlab='',ylab='')
text(0.5,0.5,'Supporting figures',cex=2,font=2)
```

Supporting figures

```
## Plot the statistics of R2:
hist(100*as.numeric(r2),breaks=seq(0,100,by=5),lwd=2,col=rgb(0,0.3,0.5),
     xlab=expression(paste(R^2,' (%)')),freq=TRUE,
     main="Summary of regression scores")
grid()
figlab('Figure SM2')
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter
```

```
## Warning in text.default(xpos, ypos, x, type = 2, cex = 1.2, pos = 4, col =
## "grey30"): graphical parameter "type" is obsolete
```

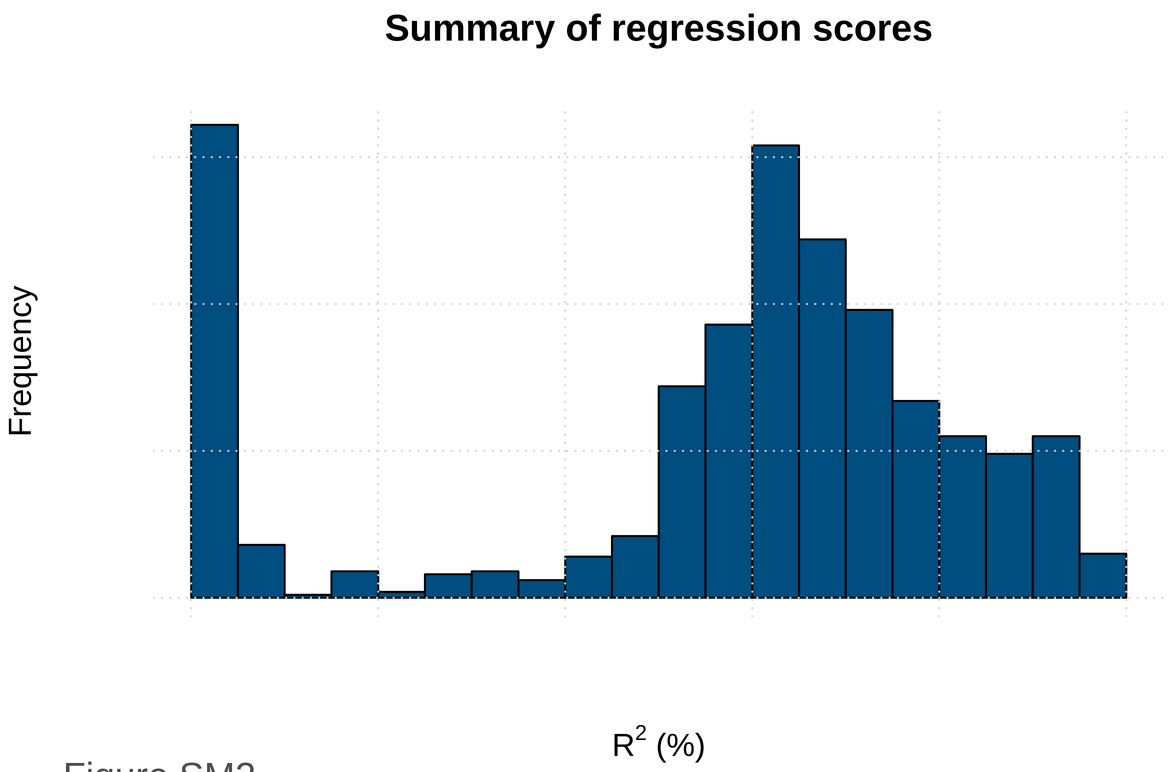


Figure SM2

##**Figure SM12**

A figure showing one example of projections for one location, showing all three emission scenarios.

```
## Example of estimates for 2050:
print('Example plot - evolution')
```

```
## [1] "Example plot - evolution"
```

```
par(bty='n')
i <- 1
N <- length(Z.rcp4.5)
plot(Z.rcp4.5[[i]],plot.type='single',lty=c(2,2,1),lwd=c(1,1,2),
     ylab='%',ylim=c(80,150),
     main=paste('Wet-day mean at',loc(subset(mux,is=i))))
shade(Z.rcp4.5[[i]],col=rgb(0.5,0.5,0.5,0.3))
shade(Z.rcp8.5[[i]],col=rgb(1,0.5,0.5,0.3))
shade(Z.rcp2.6[[i]],col=rgb(0.5,1,0.5,0.3))
grid()
figlab('Figure SM12')
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter
```

```
## Warning in text.default(xpos, ypos, x, type = 2, cex = 1.2, pos = 4, col =
## "grey30"): graphical parameter "type" is obsolete
```

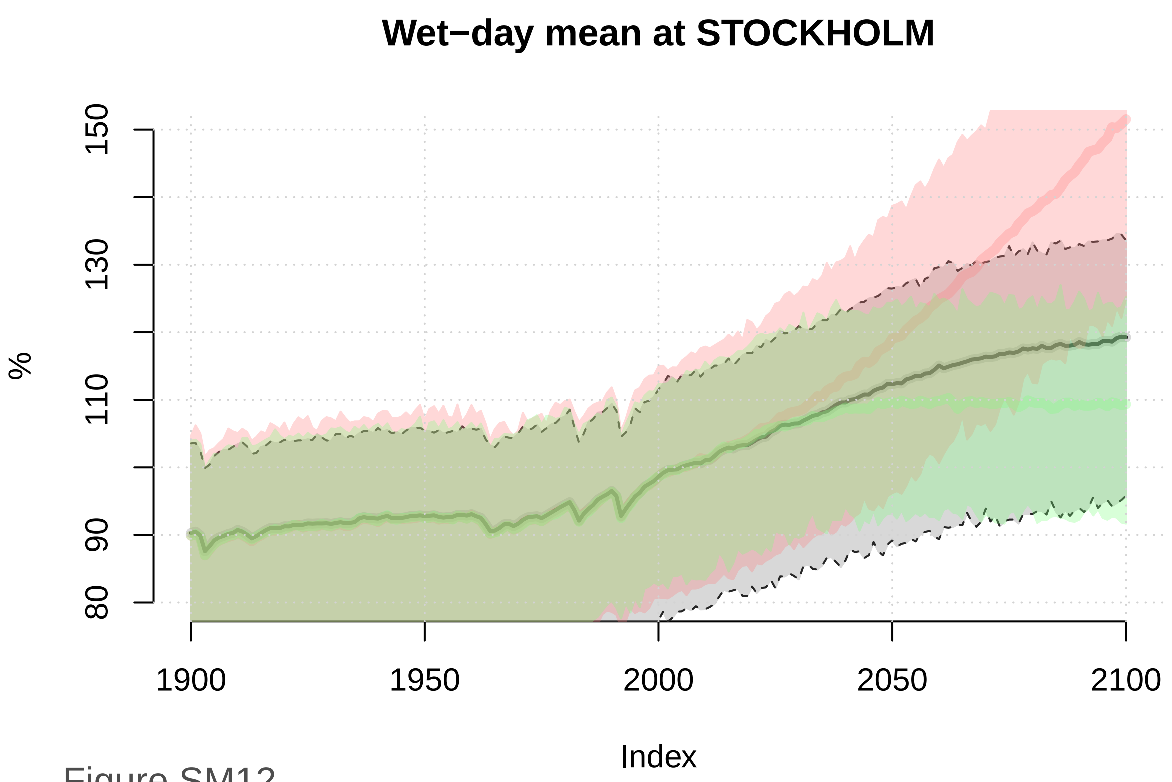


Figure SM12

Further processing for Fig SM3: extract information about trends in μ.

```
## Trend in projected wet-day mean
trendbeta <- unlist(lapply(Z.rcp4.5,function(x) trend.coef(x[,3])))
## Strange results:
print((1:N)[trendbeta < 0])
```

```
## integer(0)
```

```
## test: Does the model predict observed trends?
print('evaluation: trends')
```

```
## [1] "evaluation: trends"
```

```
## Make sure to compare series with data for same times
mask <- !is.finite(coredata(MUx))
class(MUz) <- class(MUx)
muz <- coredata(MUz)
muz[mask] <- NA; dim(muz) <- dim(MUz)
coredata(MUz) <- muz

## Only look at stations with more than 50 years with data
ok <- (apply(coredata(MUx),2,nv) > 50)
MUz <- subset(MUz,is=ok)
MUx <- subset(MUx,is=ok)
FWx <- subset(FWx,is=ok)
```

```
trend.mux <- apply(MUx,2,'trend.coef')
trend.pre <- apply(MUz,2,'trend.coef')
trenderr.mux <- apply(MUx,2,'trend.err')
trenderr.pre <- apply(MUz,2,'trend.err')

## Need to get a picture whether the predictions gives a plausible
## upper limit.
trend.sense <- data.frame(x=c(-trend.mux,trend.mux),
                          y=c(-trend.pre,trend.pre))

xylim <- max(abs(c(trend.mux,trend.pre)))*c(-1,1)
```

Figure SM3

Historical trends in the wet-day mean preciptation

```
par(bty='n',col.sub='grey',cex.sub=0.8)
plot(trend.mux,trend.pre,pch=19,col=rgb(0.6,0.2,0,0.3),cex=1.5,
     main=expression(paste('Trends in ',mu,': observed and predicted upper limit')),
     xlab='Observed trend (mm/decade)',ylab='predicted trend (mm/decade)',
     xlim=xylim,ylim=xylim,
     sub=paste('Mean correlation for local year-to-year variations over t=[',
       start(MUx),',',end(MUx),
       '] is ',round(mean(r.eval),2),' (',round(quantile(r.eval,0.05),2),', ',
       round(quantile(r.eval,0.95),2),
       ')',sep='') )
grid()

polygon(c(xylim[1],xylim[2],xylim[1],xylim[1]),c(xylim[1],xylim[2],xylim[2],xylim[1]),
        col=rgb(0.2,0.6,1,0.1),border=NA)
polygon(c(xylim[1],xylim[2],xylim[2],xylim[2]),c(xylim[1],xylim[2],xylim[2],xylim[1]),
        col=rgb(1,0.2,0.2,0.1),border=NA)
points(trend.mux,trend.pre,pch=1,col=rgb(0,0,0,0.1),cex=1.5)

## Plot error bars
apply(rbind(trend.mux,trend.pre,trenderr.mux,trenderr.pre),2,
      FUN=function(x) {lines(x[1]+c(-2,2)*x[3],x[2]+c(0,0),col=rgb(0.6,0.2,0,0.1))
                       lines(x[1]+c(0,0),x[2]+c(-2,2)*x[4],col=rgb(0.6,0.2,0,0.1))
                       lines(x[1]+c(-1,1)*0.01,x[2]+c(2,2)*x[4],col=rgb(0.6,0.2,0,0.05))
                       lines(x[1]+c(-1,1)*0.01,x[2]+c(-2,-2)*x[4],col=rgb(0.6,0.2,0,0.05))
                       lines(x[1]+c(-2,-2)*x[3],x[2]+c(-1,1)*0.01,col=rgb(0.6,0.2,0,0.05))
                       lines(x[1]+c(2,2)*x[3],x[2]+c(-1,1)*0.01,col=rgb(0.6,0.2,0,0.05))
                     })
```

```
## NULL
```

```
figlab('Figure SM3')
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter
## Warning in text.default(xpos, ypos, x, type = 2, cex = 1.2, pos = 4, col =
## "grey30"): graphical parameter "type" is obsolete
```

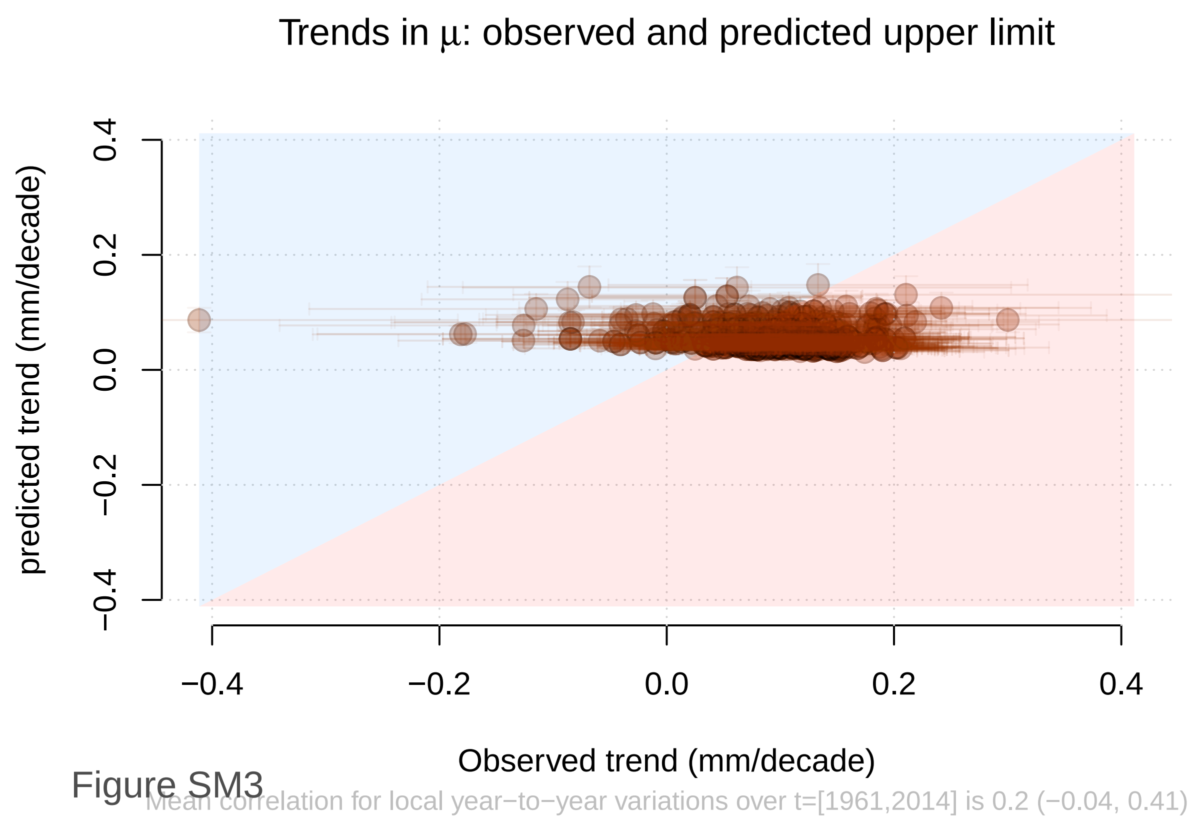


Figure SM4

```
## Map showing trends in mu
map(MU,FUN='trend',colbar=list(breaks=seq(-1,1,length=21),rev=TRUE),cex=cex)
```

```
## Warning in if (cex == 0) cex <- 1.25 * nok/max(nok, na.rm = TRUE): the
## condition has length > 1 and only the first element will be used
```

```
## Warning in if (cex < 0) cex <- abs(cex) * nok/max(nok, na.rm = TRUE): the
## condition has length > 1 and only the first element will be used
```

```
figlab('Figure SM4',ypos=0.999)
```

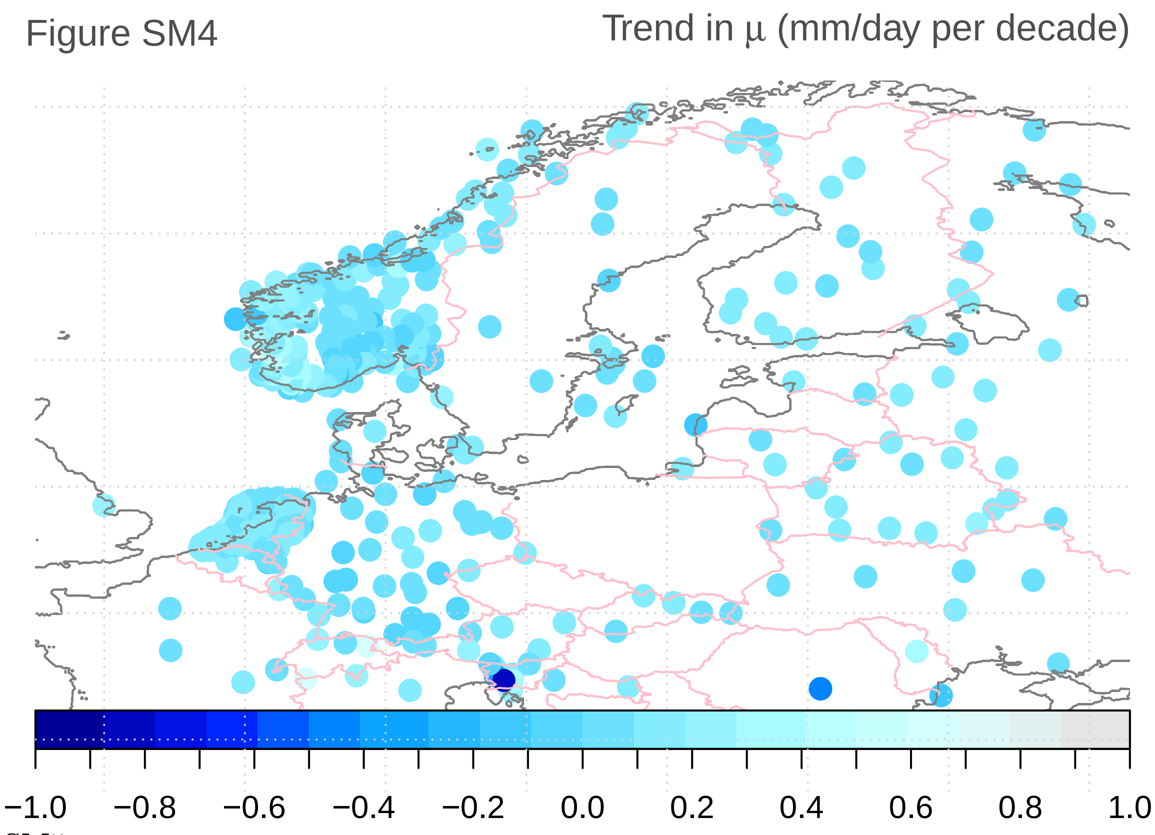
```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter
```

```
## Warning in text.default(xpos, ypos, x, type = 2, cex = 1.2, pos = 4, col =
## "grey30"): graphical parameter "type" is obsolete
```

```
figlab(expression(paste('Trend in ',mu,' (mm/day per decade)')),xpos=0.5,ypos=0.999)
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter

## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : graphical parameter "type" is obsolete
```



##Figure SM5

```
## 99-percentile wet-day mean and typical wet-day frequency
## The Tellus paper on specification approximately annual maximum.
## Prob(X>x) for annual maximum for 24hr precip
## Pr(X > x) = 1/365.25 = fw*(1-p): p = 1 - 1/(365.25*fw)
par(bty='n')
W <- apply(fw,2,FUN='muclim')
hist(100*(1-1/(W[1,]*365.25)),col='steelblue',lwd=2,breaks=seq(90,100,by=0.1),
     main='Wet-day percentile for annual maximum 24-precipitation',
     xlab='p (%)')
grid()
figlab('Figure SM5')
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter

## Warning in text.default(xpos, ypos, x, type = 2, cex = 1.2, pos = 4, col =
## "grey30"): graphical parameter "type" is obsolete
```

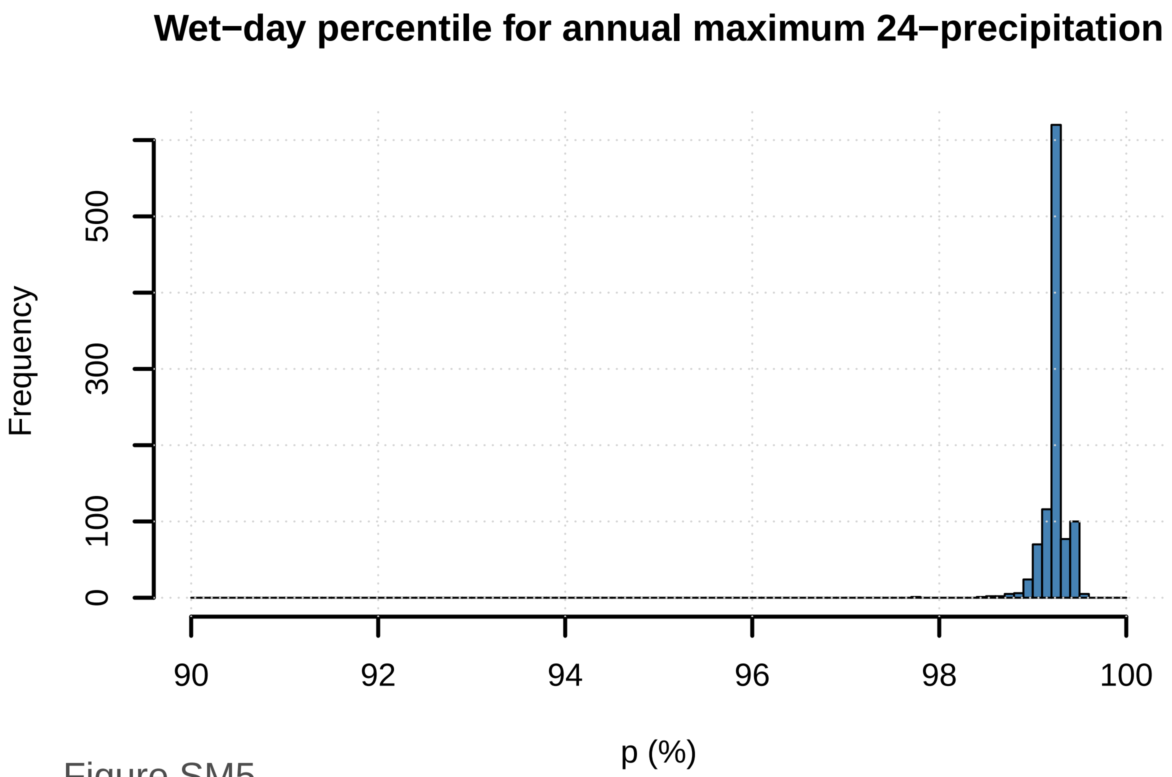


Figure SM5

Figure SM6

```
## Statistics of trend in wet-day frequency
print('Wet-day frequency statistics')
```

```
## [1] "Wet-day frequency statistics"
```

```
hist(fw.trend,breaks=seq(-50,50,by=1),col='grey',lwd=2,
     main='Trend in wet-day frequency',
     xlab=expression(paste(f[w],' (%/decade)')))
grid()
figlab('Figure SM6')
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter
```

```
## Warning in text.default(xpos, ypos, x, type = 2, cex = 1.2, pos = 4, col =
## "grey30"): graphical parameter "type" is obsolete
```

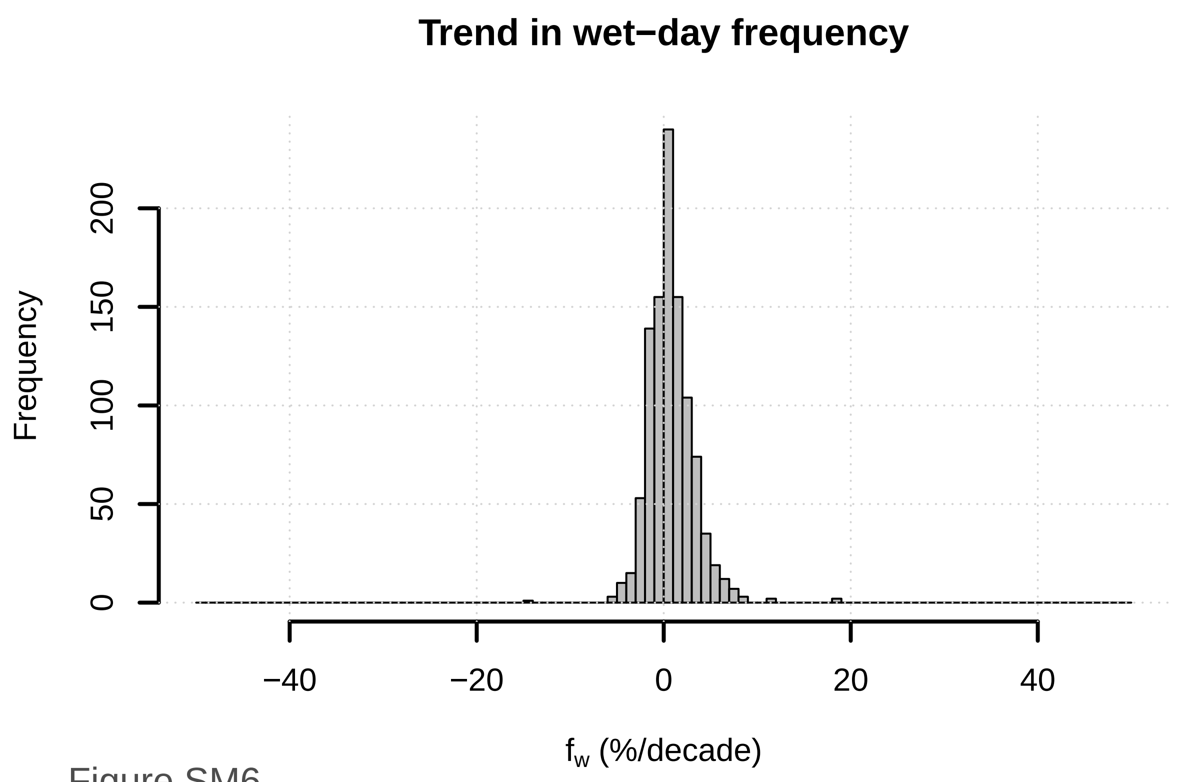


Figure SM6

##**Figure SM7**

```
## Map showing trends in fw
#dev.new()
map(FW,FUN='trend',colbar=list(breaks=seq(-0.05,0.05,length=21),rev=TRUE),cex=cex)
```

```
## Warning in if (cex == 0) cex <- 1.25 * nok/max(nok, na.rm = TRUE): the
## condition has length > 1 and only the first element will be used

## Warning in if (cex < 0) cex <- abs(cex) * nok/max(nok, na.rm = TRUE): the
## condition has length > 1 and only the first element will be used
```

```
figlab('Figure SM7',ypos=0.999)
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter

## Warning in text.default(xpos, ypos, x, type = 2, cex = 1.2, pos = 4, col =
## "grey30"): graphical parameter "type" is obsolete
```

```
figlab(expression(paste('Trend in ',f[w],' (fraction per decade)')),xpos=0.5,ypos=0.999)
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter

## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : graphical parameter "type" is obsolete
```

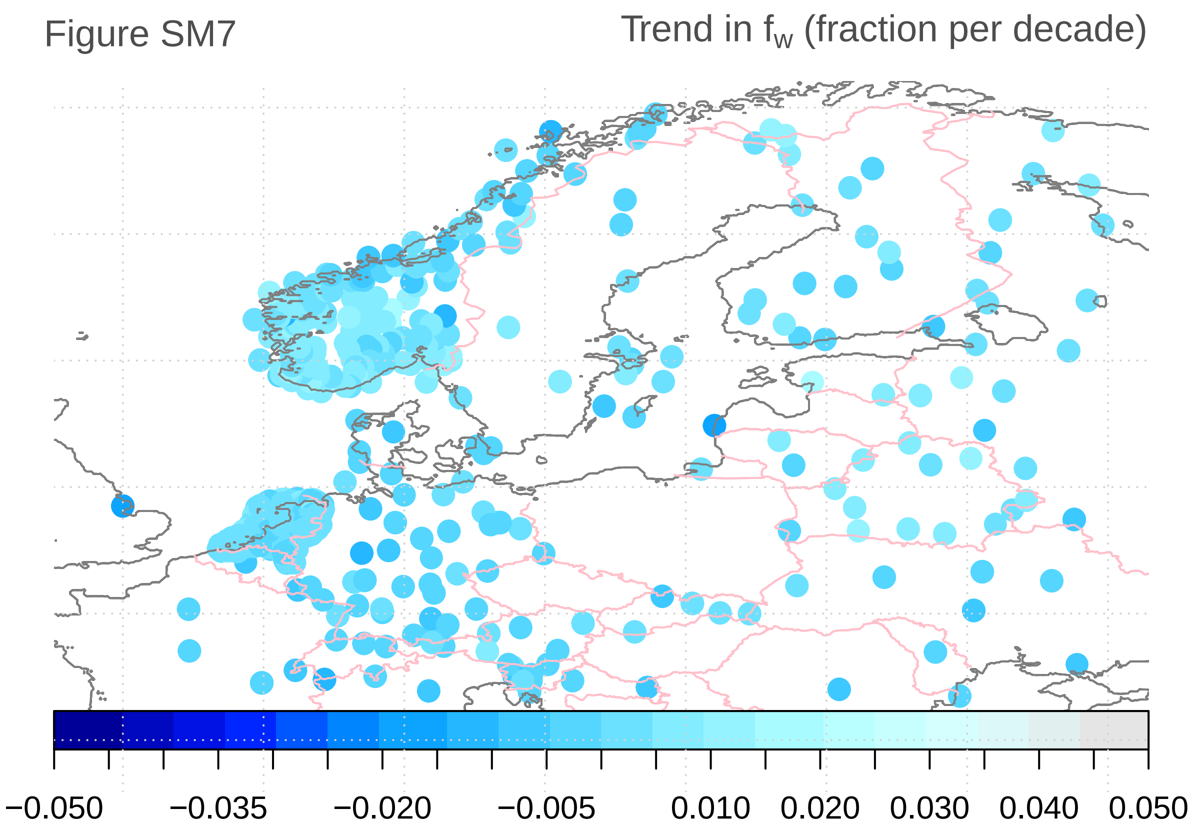


Figure SM1

```
## Figure SM1.
## test: See if the quantiles are consistent when the mean mu varies.
qtest <- aggregate(y1,year,FUN='qqexp')
qx <- c(coredata(qtest[,1:101]))
qy <- c(coredata(qtest[,102:202]))

par(bty='n')
plot(qx,qy,xlim=c(0,40),ylim=c(0,40),
     pch=19,col=rgb(0.2,0.2,0.7,0.3),cex=cex,
     main='Test: exponential distribution & changing mean',
     xlab=expression(q[p]),ylab=expression(-log(1-p)*mu))
lines(c(0,40),c(0,40),col='grey')
figlab('Figure SM1')
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter
```

```
## Warning in text.default(xpos, ypos, x, type = 2, cex = 1.2, pos = 4, col =
## "grey30"): graphical parameter "type" is obsolete
```

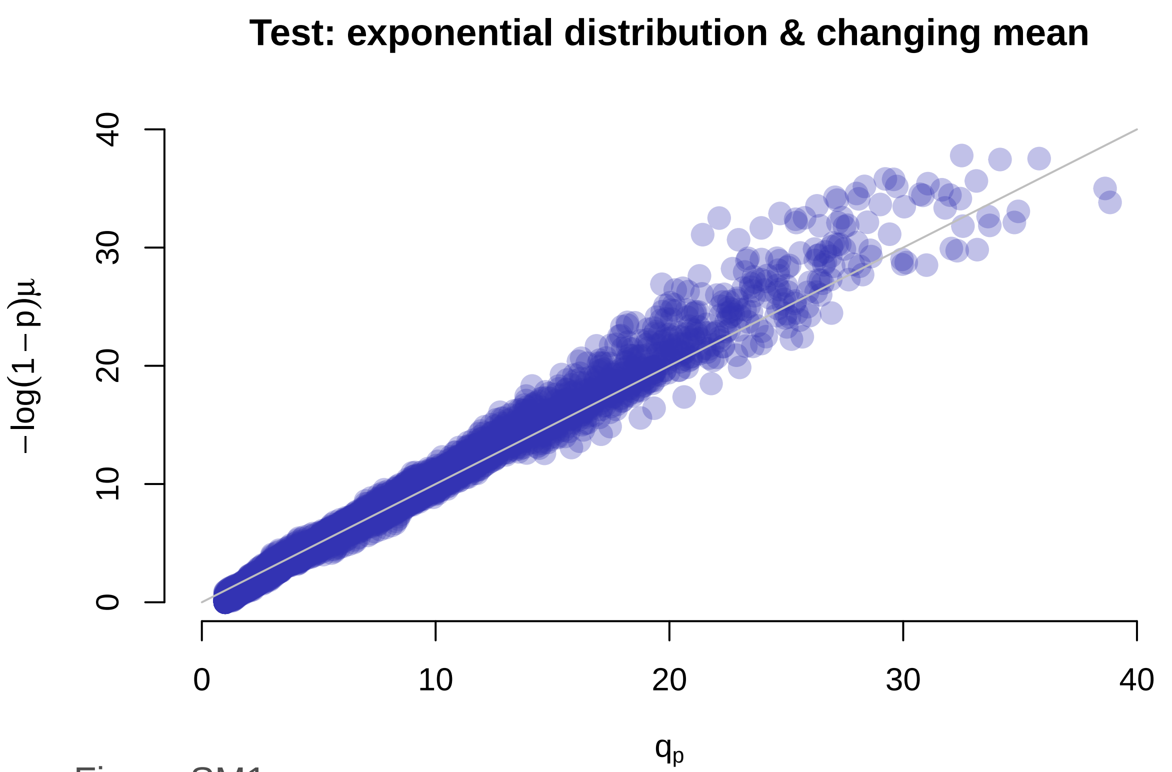


Figure SM1

Figure SM14: compare the mean seasonal variations in the different precipitation statistics

```
pr.mean <- aggregate(y1,by=month,FUN='mean')
pr.mu <- aggregate(y1,by=month,FUN='wetmean')
pr.fw <- aggregate(y1,by=month,FUN='wetfreq')
y1.l <- spell(y1,threshold=1)
pr.wet <- aggregate(subset(y1.l,is=1),by=month,FUN='mean')
pr.dry <- aggregate(subset(y1.l,is=2),by=month,FUN='mean')

par(bty='n',xaxt='n')
plot(merge(pr.mean,pr.mu,10*pr.fw,pr.wet,pr.dry),plot.type='single',
     col=c('steelblue','darkblue','grey','darkgreen','red'),
     lwd=c(3,3,1,1,1),ylab="",xlab="Calendar month",main=loc(y1))
grid()
par(yaxt='s',xaxt='s')
axis(1,at=1:12,labels=month.abb,cex.lab=0.7, col='grey')
axis(4,at=10*pretty(pr.fw),pretty(pr.fw),col='grey')

legend(1,8.5,c(expression(bar(x)),expression(mu),expression(f[w]),
               expression(bar(n[c*w*d])),expression(bar(n[c*d*d]))),bty='n',
       col=c('steelblue','darkblue','grey','darkgreen','red'),lwd=c(3,3,1,1,1))
figlab('Figure SM9')
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter
## Warning in text.default(xpos, ypos, x, type = 2, cex = 1.2, pos = 4, col =
## "grey30"): graphical parameter "type" is obsolete
```

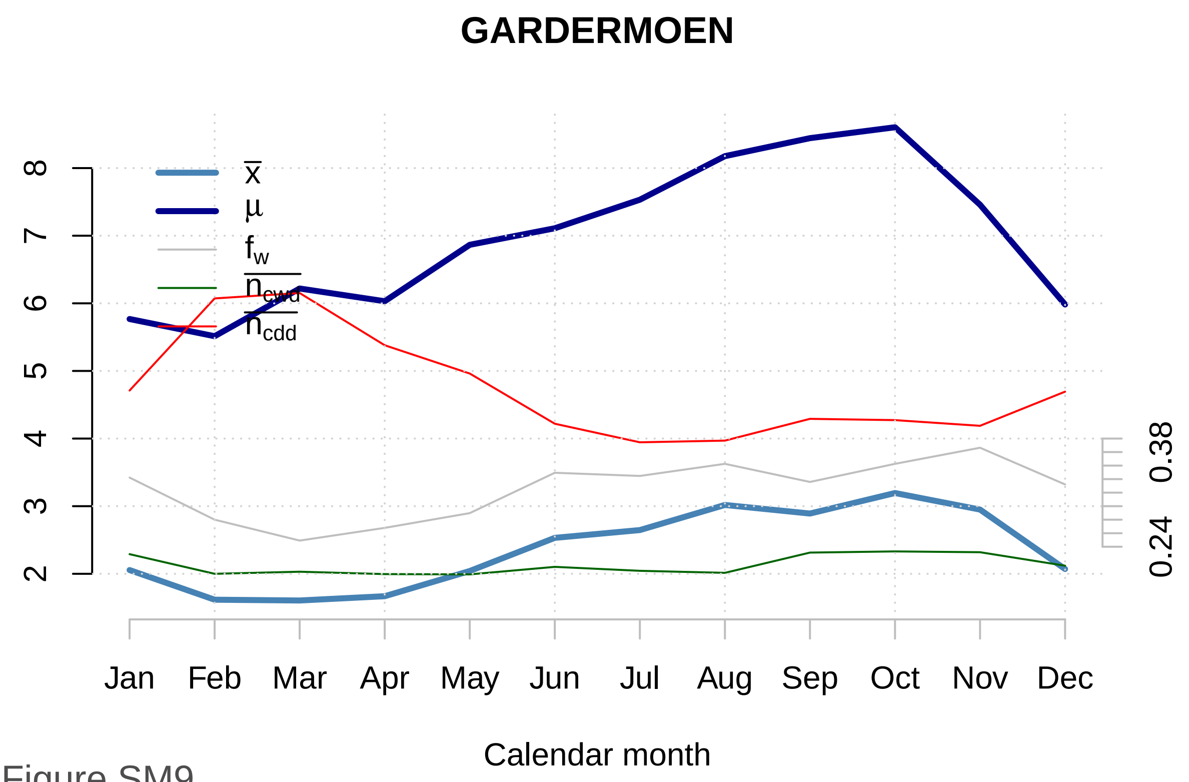


Figure SM9

Figure SM10: Map of the future return values

```
print('Map of return values for 2100 with RCP4.5')
```

```
## [1] "Map of return values for 2100 with RCP4.5"
```

```
rv <- MUx
## x2100u gives the percentage of mu,
coredata(rv) <- t((x2100u/100)*apply(coredata(MUx),2,'mean')*log(365.25*t(coredata(FWx))))
cexr2 <- 1.5*c(as.numeric(r2)) + 0.2
map(rv,FUN='mean',cex=cexr2,colbar=list(breaks=seq(30,100,by=1)))
```

```
## Warning in if (cex == 0) cex <- 1.25 * nok/max(nok, na.rm = TRUE): the
## condition has length > 1 and only the first element will be used
```

```
## Warning in if (cex < 0) cex <- abs(cex) * nok/max(nok, na.rm = TRUE): the
## condition has length > 1 and only the first element will be used
```

```
figlab('Figure SM10',ypos=0.999)
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter
## Warning in text.default(xpos, ypos, x, type = 2, cex = 1.2, pos = 4, col =
## "grey30"): graphical parameter "type" is obsolete
```

```
figlab('Return values for 2100 assuming RCP4.5',xpos=0.3,ypos=0.999)
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter

## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : graphical parameter "type" is obsolete
```

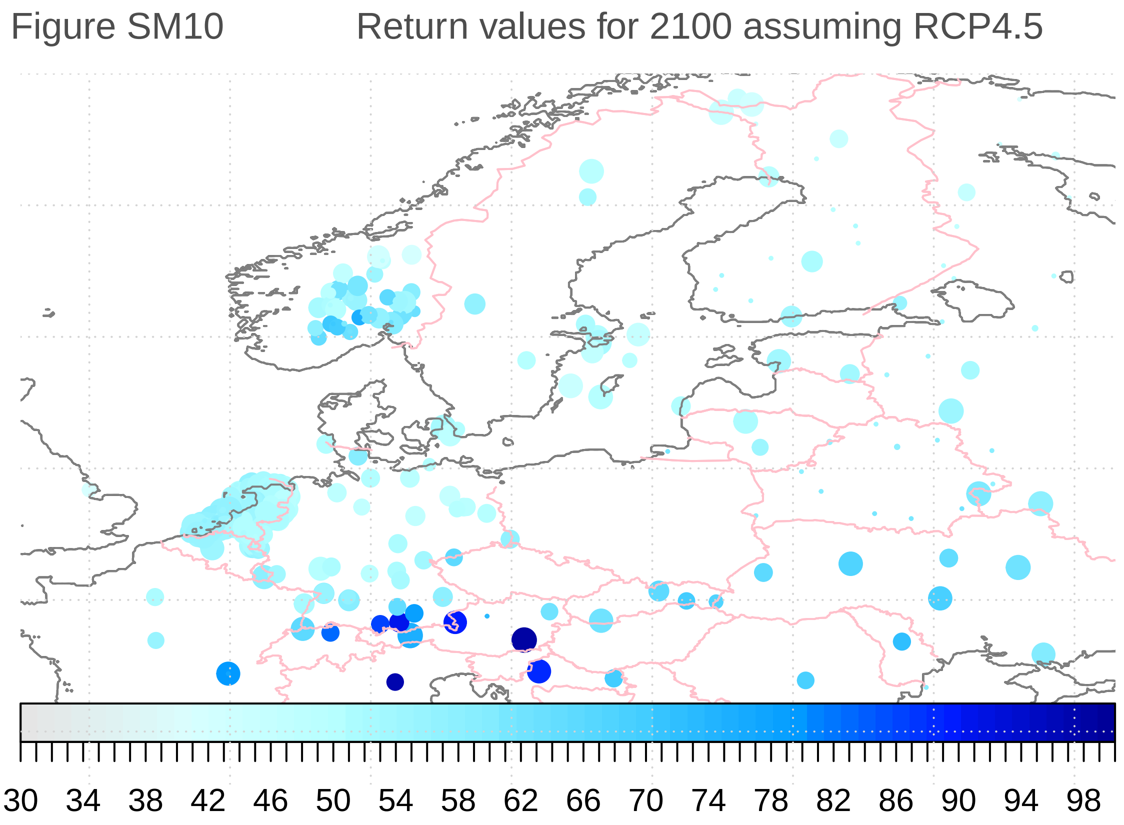


Figure SM9

```
## Print the numbers:
print('--- Changes in 20% returnvalue in terms of % from 2010:')
```

```
## [1] "--- Changes in 20% returnvalue in terms of % from 2010:"
```

```
print(lapply(mu.2100,summary))
```

```
## $mean.RCP4.5
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   6.958   9.111  10.660  12.170  13.890  26.040
##
```

```
## $q95.RCP4.5
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   7.398   9.688  11.340  12.940  14.770  27.690
##
## $mean.RCP2.6
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.111   4.085   4.790   5.487   6.264  11.930
##
## $q95.RCP4.5.1
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.639   4.778   5.603   6.417   7.327  13.950
##
## $mean.RCP8.5
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   18.71   24.64   28.96   33.31   38.04   73.70
##
## $q95.RCP8.5
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   21.59   28.43   33.41   38.43   43.88   85.02
```

```
print(paste('Summary for',length(mu.2100[[1]]),'locations'))
```

```
## [1] "Summary for 615 locations"
```

```
## Compare the regression coefficients derived from individual
## seasonal cycle with that derived from mean climatology.

data(mu.eq.f.tx)

col <- rgb(0.1,0.1,0.7,0.25)
mutx <- summary(mu.eq.f.tx)$coefficients[c(2,4)]
b1 <- as.numeric(lapply(Beta,function(x) x[1]))
e1 <- as.numeric(lapply(Beta,function(x) x[2]))

par(bty='n')
plot(b1,pch=19,col=col,cex=cexr2,
     main=expression(paste('Scaling coefficient for ',mu,' and ',e[s])),
     xlab='',ylab=expression(beta))
grid()
for (i in 1:length(b1)) {
  lines(rep(i,2),b1[i]+e1[i]*c(-2,2),col=col)
  lines(i+c(-0.45,0.45),b1[i]+e1[i]*c(-2,-2),col=col)
  lines(i+c(-0.45,0.45),b1[i]+e1[i]*c(2,2),col=col)
}
polygon(c(1,rep(length(b1),2),rep(1,2)),
        mutx[1]+mutx[2]*c(-2,-2,2,2,-2),
        border=rgb(0.5,0.5,0.5,0.4),col=rgb(0.5,0.5,0.5,0.3))
lines(c(1,length(b1)),rep(mutx[1],2),lwd=3,col=rgb(0.5,0.5,0.5,0.3))
figlab('Figure SM9')
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter
## Warning in text.default(xpos, ypos, x, type = 2, cex = 1.2, pos = 4, col =
## "grey30"): graphical parameter "type" is obsolete
```

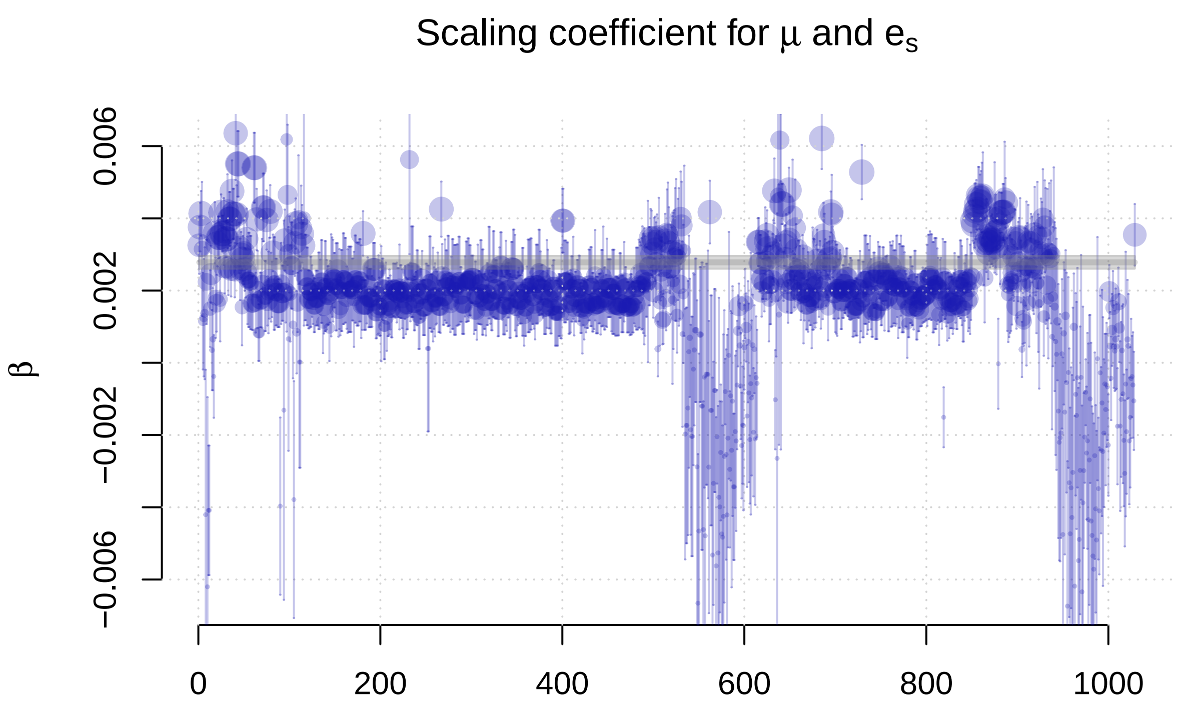


Figure SM9

Figure SM13

```
## Show the predictor area:
X <- retrieve('air.mon.mean.nc',lon=c(-100,-30),lat=c(0,40))
```

```
## [1] "Warning : Calendar attribute has not been found in the meta data and will be set automatically.
```

```
map(X,projection='sphere',colbar=list(breaks=seq(8,28,by=0.5)))
```

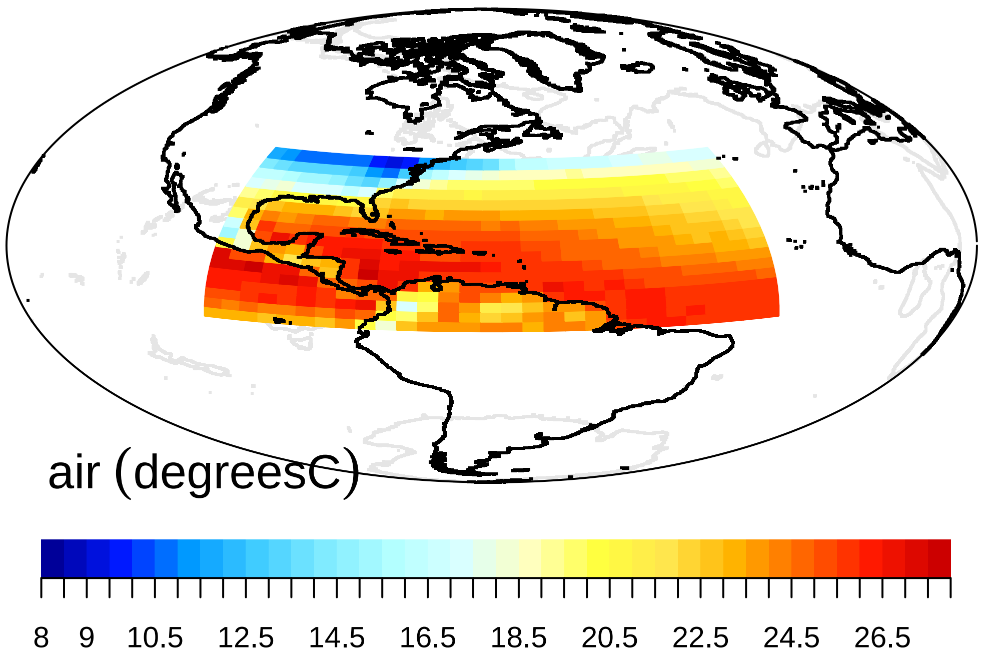
```
## [1] "Clip the value range to extremes of colour scale"
## [1] "0 set to highest colour and 0 to lowest"
```

```
figlab('Figure SM13',xpos=0.8,ypos=0.999)
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter
```

```
## Warning in text.default(xpos, ypos, x, type = 2, cex = 1.2, pos = 4, col =
## "grey30"): graphical parameter "type" is obsolete
```

Figure SM13



Analysis from other continents and with local temperature

Additional analysis testing the connection between local temperature and wet-day mean precipitation

```
## Regression analysis between the wet-day mean mu and the mean temperature.

corhalf <- function(x) {
  n <- length(x)
  x1 <- x[1:(n/2)]; x2 <- x[(n/2+1):n]
  ok <- is.finite(x1) & is.finite(x2)
  r <- cor(x1[ok],x2[ok])
  return(r)
}
```

```
## CLARIS
load('claris.Tx.rda')
load('claris.Pr.rda')

Tx1 <- Tx
Pr1 <- Pr

## COST-VALUE
load('VALUE_ECA_86_v2/stationsVALUE-exp1a.rda')
Tx2 <- Tx
Pr2 <- Pr
```

```r
## Read North american data:
if (!file.exists('mut2m.GDCN.rda')) {
  source('readGDCN.R')

  gdcn <- list.files('/disk1/GDCN-data_disk2',pattern='dly', full.names = TRUE)
  finfo <- file.info(gdcn)
  fok <- (finfo$size > 200000)
  gdcn <- gdcn[fok]
  n <- length(gdcn)

  plot(c(-180,180),c(-90,90),type='n',xlab='',ylab='')
  data(geoborders)
  lines(geoborders)

  for (i in 1:n) {
    pr <- readGDCN(gdcn[i])
    tx <- readGDCN(gdcn[i],param="tmax")
    if ( (nv(pr) >  20000) & (nv(tx) >  20000)) {
      pr <- subset(pr,it=c(1945,2015))
      tx <- subset(tx,it=c(1945,2015))

      if (i==1) {
        mu <- annual(pr,FUN='wetmean')
        fw <- annual(pr,FUN='wetfreq')
        t2m <- annual(tx,FUN='mean',na.rm=TRUE)
      } else {
        mu <- combine(mu,annual(pr,FUN='wetmean'))
        fw <- combine(fw,annual(pr,FUN='wetfreq'))
        t2m <- combine(t2m,annual(tx,FUN='mean',na.rm=TRUE))
      }
      print(paste(i,' (',n,'): ',loc(pr),', ',cntr(pr),' #validdata=',nv(pr),
                  'Tx: ',round(mean(tx,na.rm=TRUE),2),
                  round(min(tx,na.rm=TRUE),2),
                  round(max(tx,na.rm=TRUE),2),lat(tx)))
      points(lon(pr),lat(pr),pch=19,col='darkgreen')
    }
  }
  save(file='mut2m.GDCN.rda',mu,fw,t2m)
} else load('mut2m.GDCN.rda')
```

```r
## Aggregate annual statistics based on the combined data sources:
MU <- combine(mu,annual(Pr1,FUN='wetmean'),annual(Pr2,FUN='wetmean'))
FW <- combine(fw,annual(Pr1,FUN='wetfreq'),annual(Pr2,FUN='wetfreq'))
T2M <- combine(t2m,annual(Tx1,FUN='mean'),annual(Tx2,FUN='mean'))
nval <- apply(coredata(MU),2,'nv')
attr(T2M,'variable') <- 't2m'
es <- C.C.eq(T2M)

#map(MU,FUN='mean',cex=0)
#map(MU,FUN='trend',cex=0,colbar=list(rev=TRUE,breaks=seq(-1,1,length=21)))
#map(FW,FUN='trend',cex=0,colbar=list(rev=TRUE,breaks=seq(-0.01,0.01,length=21)))
calmu <- data.frame(x=as.numeric(apply(coredata(es),2,'mean',na.rm=TRUE)),
                    y=as.numeric(apply(coredata(MU),2,'mean',na.rm=TRUE)),
                    fw=as.numeric(apply(coredata(FW),2,'mean',na.rm=TRUE)),
                    z=alt(MU),lat=lat(MU),lon=lon(MU),nval=nval)
premu <- calmu; premu$x[0] <- 0
model.mutx <- lm(y ~ x, weights=fw,data=calmu)
print(summary(model.mutx))
```

```
##
## Call:
## lm(formula = y ~ x, data = calmu, weights = fw)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -4.3565 -1.0354 -0.0751  0.7310  6.0004
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 4.038e+00  2.094e-01   19.28   <2e-16 ***
## x           2.783e-03  8.788e-05   31.67   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.22 on 1418 degrees of freedom
## Multiple R-squared:  0.4142, Adjusted R-squared:  0.4138
## F-statistic:  1003 on 1 and 1418 DF,  p-value: < 2.2e-16
```

```
data(geoborders)
Fw <- apply(coredata(fw),2,'sum',na.rm=TRUE)
```

** Figure SM8**

Assess the connection between temperature and the wet-day mean precipitation.

```
par(bty='n')
col <- rgb((1+sin(pi*calmu$lat/180))/2,
           cos(pi*calmu$lon/180)^2,
           1-(1+sin(pi*calmu$lat/180))/2,0.15)
cex <- 1.5*Fw/max(Fw,na.rm=TRUE)
plot(calmu$x,calmu$y,pch=19,col=col,cex=cex,
     main='Wet-day mean precipitation temperature dependency',
     ylab=expression(mu*phantom(0)*(mm/day)),
     xlab=expression(e[s]*phantom(0)**(Pa)))
points(calmu$x,calmu$y,pch=21,col=rgb(0.5,0.5,0.5,0.2),cex=cex)
lines(calmu$x,predict(model.mutx),col=rgb(0.4,0.4,0.4,0.1),lwd=2)

par(new=TRUE,fig=c(0.15,0.45,0.7,0.9),mar=rep(0,4),xaxt="n",yaxt="n")
plot(calmu$lon,calmu$lat,pch=19,col=col,cex=0.3*cex)
lines(geoborders,col='grey')
points(calmu$lon,calmu$lat,pch=19,col=col,cex=0.3*cex)
figlab('Figure SM8')
```

```
## Warning in par(new = TRUE, pdx = NA, fig = c(0, 1, 0, 1), xaxt = "n", yaxt
## = "n", : "pdx" is not a graphical parameter
## Warning in text.default(xpos, ypos, x, type = 2, cex = 1.2, pos = 4, col =
## "grey30"): graphical parameter "type" is obsolete
```

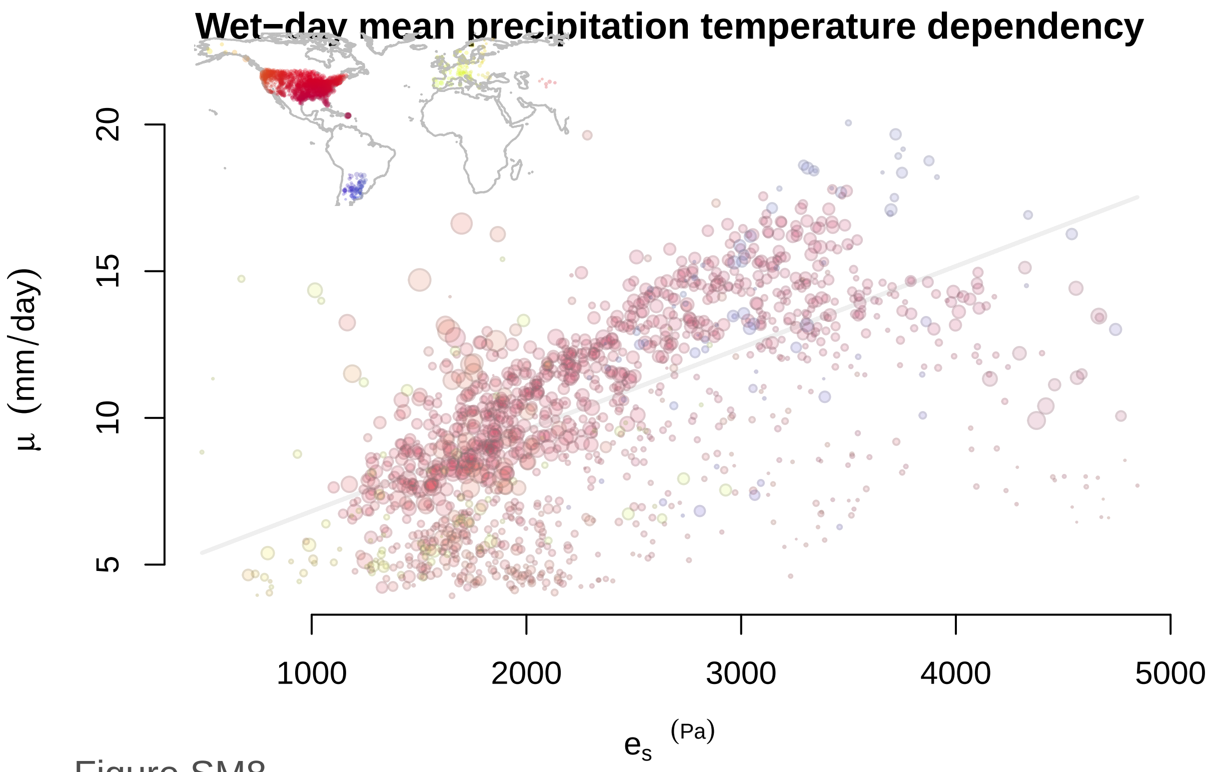
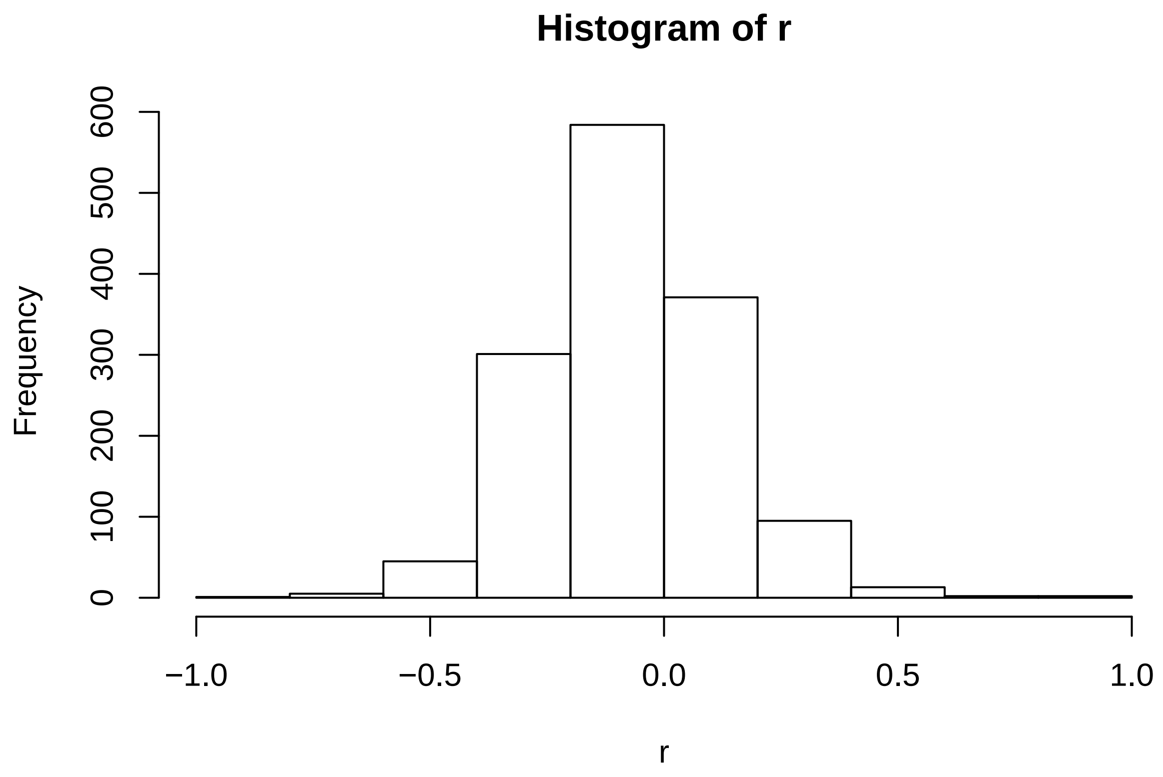


Figure SM8

```
mu.eq.f.tx <- model.mutx
attr(mu.eq.f.tx,'input') <- 'saturation water pressure e_s (Pa)'
attr(mu.eq.f.tx,'predictand') <- 'C.C.eq(tmax)'
attr(mu.eq.f.tx,'output') <- 'wet-day mean precipitation (mm/day)'
attr(mu.eq.f.tx,'calibration') <- 'mean climatology'
attr(mu.eq.f.tx,'source script') <- 'mut2m.R'
attr(mu.eq.f.tx,'timestamp') <- date()
attr(mu.eq.f.tx,'calibration_data') <- calmu
save(file='esd/data/mu.eq.f.tx.rda',mu.eq.f.tx)
```

```
## Estimate the year-by-year correlation in es and mu

X <- matchdate(MU,es)
Y <- matchdate(es,MU)
w <- apply(coredata(fw),2,'mean')
ok <- w > 0.25
Z <- rbind(coredata(X),coredata(Y))
r <- apply(Z,2,corhalf)
hist(r)
```


Histogram of r
Frequency
0
100
200
300
400
500
600
−1.0
−0.5
0.0
0.5
1.0
r