Data preparation

Willem Vervoort, Michaela Dolk & Floris van Ogtrop

2017-02-07

```
# root dir
knitr::opts_knit$set(root.dir = "C:/Users/rver4657/ownCloud/Virtual Experiments/VirtExp")
```

This rmarkdown document and the resulting pdf are stored on github. All directories (apart from the root working directory) refer to the directories in this repository

Introduction

This document is related to the manuscript "Disentangling climate change trends in Australian streamflow" (vervoort et al.), submitted to Journal of Hydrology. This document outlines the preparation of the original data into the dataframes that have been analysed in the project. The decision making on how stations were identified, is outlined in the methods of the submitted manuscript. This document is aimed at documenting the code of the analysis.

sources of data

As outlined in the manuscript, the original data were sourced from the following locations: Streamflow is from the Bureau of Meteorology (BOM) hydrological reference stations http://www.bom.gov.au/water/hrs/ Rainfall and temperature were obtained from the BOM gridded data and the BOM station data http://www.bom.gov.au/climate/data-services/

Reading in the data

The data consists of comma delimited (csv) files, as downloaded from the websites. All data cover the period 1970 - 2010. The following flow stations were used:

Table 1: Stations used in this project (continued below)

Name.used.in.this.project	Number	Region	Catchment.area.smaller.250km2.
COTT	410730	ACT	130
RUTH	219001	NSW	14
CORA	215004	NSW	166
ELIZ	G8150018	NT	96
COCH	113004A	QLD	95
COEN	922101B	QLD	170
SCOT	A5030502	SA	29
HELL	312061	TAS	101
NIVE	304497	TAS	174
MURR	405205	VIC	106
SOUT	225020A	VIC	10
YARR	614044	WA	80
DOMB	607155	WA	116

Latitude	Longitude	Rain.St	HQTmax.st
-35.59	148.8	70316	70351
-36.59	149.4	69003	70351
-35.15	150	69049	68072
-12.61	131.1	14149	14015
-17.74	145.6	31083	34084
-13.96	143.2	27005	27045
-35.1	138.7	23734	23373
-41.42	145.7	96023	96003
-42.03	146.4	91065	96003
-37.41	145.6	88028	85072
-37.83	146.4	85238	85072
-32.81	116.2	9538	9021
-34.58	116	9590	9518

Define decades to analyze

```
study_period_decades <- c("70_80", "80_90", "90_00", "00_10")
decade_start <- c(as.Date("1/1/1970", format="%d/%m/%Y"),
                  as.Date("1/1/1980", format="%d/%m/%Y"),
                  as.Date("1/1/1990", format="%d/%m/%Y"),
                  as.Date("1/1/2000", format="%d/%m/%Y"))
decade_end <- c(as.Date("31/12/1979", format="%d/%m/%Y"),
                as.Date("31/12/1989", format="%d/%m/%Y"),
                as.Date("31/12/1999", format="%d/%m/%Y"),
                as.Date("31/12/2010", format="%d/%m/%Y"))

# define the overall period
start_date <- as.Date("1970-01-01")
end_date <- as.Date("2010-12-31")
```

Read in the daily stream flow data

This includes conversion from ML/day (as indicated on the source website) to mm to match the rainfall data and to use in models. This means that the data needs to be scaled to the catchment size:

- convert ML/day to mm
- 1 ML = 10^6 L = 10^6 dm^3 is 10^9 cm^3 is 10^12 mm^3
- 1 km2 = 10^6 m^2 = 10^12 cm^2 = 10^14 mm^2
- ML/day to mm –> flow/area(km2)/100 = mm

```
# read in the flow data and convert to zoo
for (i in seq_along(Stations[,1])) {
  temp <- read.csv(paste("data/Original streamflow data/", Stations[i,2],
                         "_daily_ts2.csv", sep=""))
  year <- substr(as.character(temp$Date), nchar(as.character(temp$Date))-1,
                 nchar(as.character(temp$Date)))
  Dates <- as.Date(paste(substr(as.character(temp$Date), 1,
                                nchar(as.character(temp$Date))-2),
                         ifelse(as.numeric(year)>=50,paste("19",year,sep=""),
  assign(paste(Stations[i,1], "_daily_flow", sep=""),
         zoo(temp$Q/(Stations[i,4]),order.by=Dates))
```

```
}
  #####

# use zoo to merge all catchments to use same time interval
flow_zoo<-merge(COTT_daily_flow, RUTH_daily_flow, CORA_daily_flow,
                ELIZ_daily_flow, COCH_daily_flow, COEN_daily_flow,
                SCOT_daily_flow, HELL_daily_flow, NIVE_daily_flow,
                MURR_daily_flow, SOUT_daily_flow, YARR_daily_flow,
                DOMB_daily_flow)
# limit to 1970 - 2010
flow_zoo <- window(flow_zoo, start=start_date, end=end_date)

# Also create a dataframe for flow
flow_data_70_10<-data.frame(Date=time(flow_zoo), coredata(flow_zoo))
colnames(flow_data_70_10)[2:14] <- Stations[,1]
#####
```

Read in the Rainfall stations

This section reads in the data related to the closest possible rainfall stations.

```
closerainfall_stns <- Stations[,7]

# read in the data and subset to the required period
for (i in seq_along(closerainfall_stns)) {
  temp<-read.csv(paste("data/Original Rainfall data/", "IDCJAC0009_",
                       ifelse(nchar(closerainfall_stns[i])<5,
                        "00",
                        ifelse(nchar(closerainfall_stns[i])<6,"0","")),
                       closerainfall_stns[i], "_1800_Data.csv", sep=""))
  temp$Date<-ISOdate(year=temp$Year, month=temp$Month, day=temp$Day)
  temp$Date<-as.Date(temp$Date)
  temp<-subset(temp, Date>=start_date & Date<=end_date,
               select=c(9, 6))
  colnames(temp) <- c("Date", "Rainfall")
  assign(paste(Stations[i,1], "Rain", sep=""),
         zoo(temp$Rainfall, order.by=temp$Date))
}
# merge
rain_zoo<-merge(COTTRain, RUTHRain, CORARain, ELIZRain, COCHRain,
                COENRain, SCOTRain, HELLRain, NIVERain, MURRRain,
                SOUTRain, YARRRain, DOMBRain)
rainfall_data_70_10<-data.frame(Date=time(rain_zoo), coredata(rain_zoo))
colnames(rainfall_data_70_10)[2:14] <- Stations[,1]
```

Read in the Temperature stations

```
HQmaxT_stns <- Stations[,8]


for (i in seq_along(HQmaxT_stns)) {
  temp <- read.csv(paste("data/Original Temperature data/",
```

```
                        ifelse(nchar(HQmaxT_stns[i])<5,
                         "00",
                         ifelse(nchar(HQmaxT_stns[i])<6,"0",
                          HQmaxT_stns[i], ".csv", sep=""))
  temp$Date <- as.Date(temp$Date, format="%d/%m/%Y")
  temp$maxT[temp$maxT==99999.9] <- NA
  temp<-subset(temp, Date>=start_date & Date<=end_date)
  assign(paste(Stations[i,1], "temp.maxT", sep=""),
         zoo(temp$maxT, order.by=temp$Date))
}
# merge and create data.frame
maxT_zoo<-merge(COTTtemp.maxT,RUTHtemp.maxT,CORAtemp.maxT,
                ELIZtemp.maxT,COCHtemp.maxT,COENtemp.maxT,
                SCOTtemp.maxT,HELLtemp.maxT,NIVEtemp.maxT,
                MURRtemp.maxT,SOUTtemp.maxT,YARRtemp.maxT,
                DOMBtemp.maxT)

maxT_data_70_10<-data.frame(Date=time(maxT_zoo), coredata(maxT_zoo))
```

Summarising to weekly data

Because all the statistical analyses were run on weekly data, the summaries were all created the same way.

```
for (i in 1:length(Stations[,1])) {
  flow_t <- apply.weekly(flow_zoo[,i], sum)
  if (i ==1) flow_weekly <- flow_t else flow_weekly <- merge(flow_weekly,flow_t,all=T)
}
colnames(flow_weekly) <- colnames(flow_zoo)
for (i in 1:length(Stations[,1])) {
  rain_t <- apply.weekly(rain_zoo[,i], sum)
  if (i ==1) rain_weekly <- rain_t else rain_weekly <- merge(rain_weekly,rain_t,all=T)
}
colnames(rain_weekly) <- colnames(rain_zoo)
for (i in 1:length(Stations[,1])) {
  maxT_t <- apply.weekly(maxT_zoo[,i], mean)
  if (i ==1) maxT_weekly <- maxT_t else maxT_weekly <- merge(maxT_weekly,maxT_t,all=T)
}
colnames(maxT_weekly) <- colnames(maxT_zoo)
```

Stacking and merging weekly data into one dataset

Now stack all the data together to create one transportable data set

```
# flow
flow_weekly_stack <- data.frame(Date=time(flow_weekly),
                                coredata(flow_weekly))
colnames(flow_weekly_stack) <- c("Date",
                                 paste("flow",Stations[,1],sep="."))

# add a column for the decade
for (j in 1:length(decade_start)) {
  flow_weekly_stack$decade[as.Date(flow_weekly_stack$Date) >=
```

```
                              as.Date(decade_start[j]) &
                              as.Date(flow_weekly_stack$Date) <=
                              as.Date(decade_end[j])] <-
    study_period_decades[j]
}
# stack
flow_weekly_stack <- reshape(flow_weekly_stack, direction="long",
                              varying=2:14, sep=".")

# Now do the same for rainfall
rain_weekly_stack <- data.frame(Date=time(rain_weekly),
                                coredata(rain_weekly))
colnames(rain_weekly_stack) <- c("Date",
                                 paste("rain",Stations[,1],sep="."))

# add a column for the decade
for (j in 1:length(decade_start)) {
  rain_weekly_stack$decade[as.Date(rain_weekly_stack$Date) >=
                              as.Date(decade_start[j]) &
                              as.Date(rain_weekly_stack$Date) <=
                              as.Date(decade_end[j])] <-
    study_period_decades[j]
}
# stack
rain_weekly_stack <- reshape(rain_weekly_stack, direction="long",
                              varying=2:14, sep=".")

# and for temperature
maxT_weekly_stack <- data.frame(Date=time(maxT_weekly),
                                coredata(maxT_weekly))
colnames(maxT_weekly_stack) <- c("Date",
                                 paste("maxT",Stations[,1],sep="."))

# add a column for the decade
for (j in 1:length(decade_start)) {
  maxT_weekly_stack$decade[as.Date(maxT_weekly_stack$Date) >=
                              as.Date(decade_start[j]) &
                              as.Date(maxT_weekly_stack$Date) <=
                             as.Date(decade_end[j])] <-
    study_period_decades[j]
}
# stack
maxT_weekly_stack <- reshape(maxT_weekly_stack, direction="long",
                              varying=2:14, sep=".")


# Now merge all together into one dataset
flow_rain_maxT_weekly <- cbind(flow_weekly_stack[,1:4],
                                rain_weekly_stack[,4],
                                maxT_weekly_stack[,4])
colnames(flow_rain_maxT_weekly) <- c("Date","Decade", "Station", "Flow",
                                     "Rain", "MaxT")
```

Gridded rainfall comparison

Because of the size of the overall gridded rainfall data set, only the script that describes the extraction is given in the Rcode folder: **ExtractRasterPointGriddedRainfall.R**. However the data will be compared against the station rainfall data.

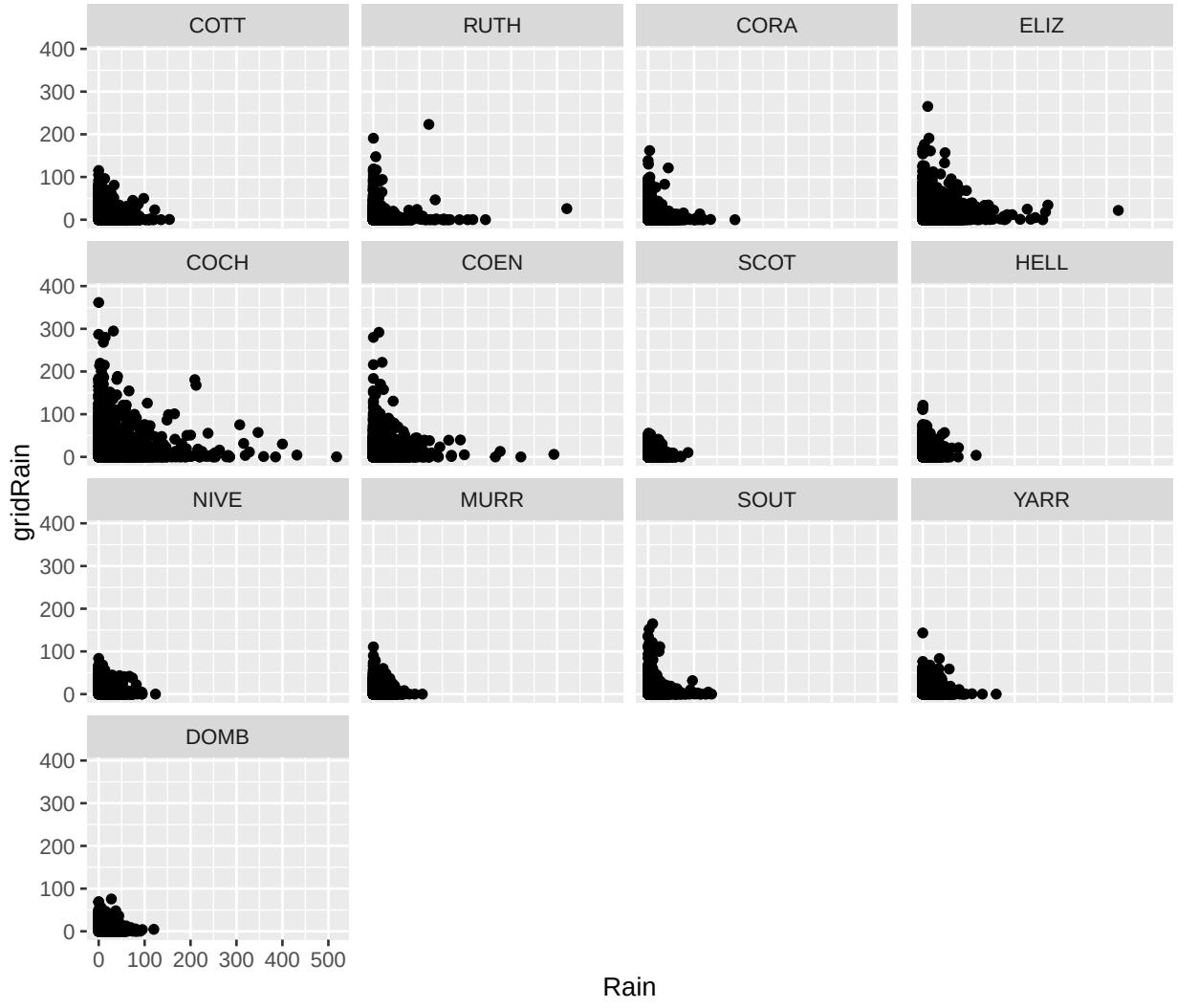
```
load("Data/GriddedRainfallData.Rdata")
# this has a data.frame called output.z, which is the gridded data
colnames(output.z) <- Stations[,1]
GridRainAllDataout <- melt(as.data.frame(output.z))
```

```
## No id variables; using all as measure variables
```

```
colnames(GridRainAllDataout) <- c("Station","gridRain")
GridRainAllDataout$Rain <- melt(as.data.frame(rain_zoo))[,2]
```

```
## No id variables; using all as measure variables
```

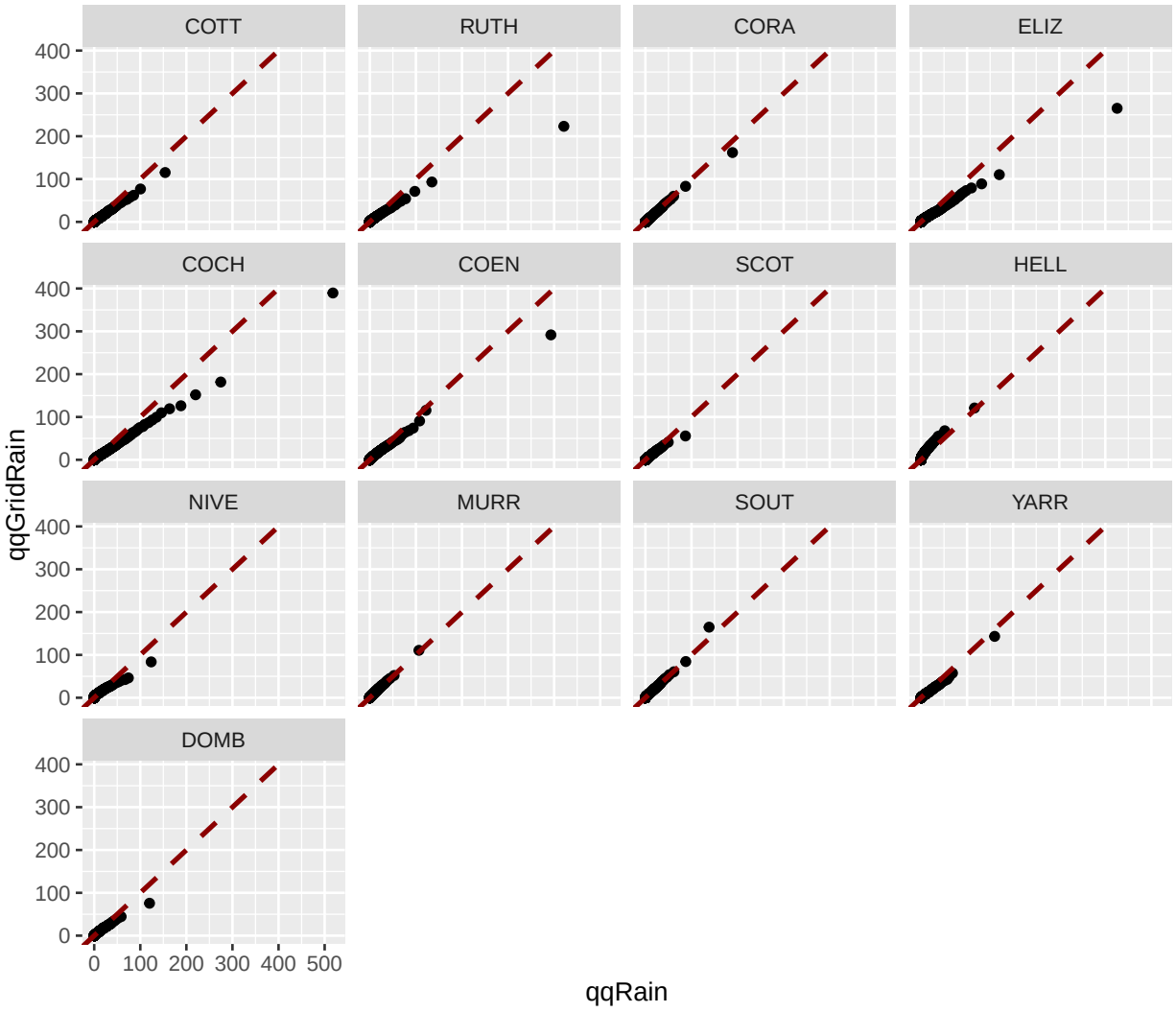
```
# First make a simple xyplot by catchment
xyp_rain <- ggplot(data=GridRainAllDataout,aes(x=Rain,  y=gridRain))
xyp_rain <- xyp_rain + geom_point() + facet_wrap(~Station)
print(xyp_rain)
```



This shows that in general there is very little 1:1 agreement between the gridded and the station rainfall, but that is not strang. The gridded rainfall is an interpolated surface based on several stations. A better way to compare the data is probably using a qqplot

```
# make a qqplot
qqGridRain <- do.call(cbind,tapply(GridRainAllDataout$gridRain,
                                   GridRainAllDataout$Station,
                                   FUN=quantile,seq(0,1,length=1000)))
qqRain <- do.call(cbind,tapply(GridRainAllDataout$Rain,
                                   GridRainAllDataout$Station,
                                   FUN=quantile,seq(0,1,length=1000),na.rm=T))
qqGridRain_s <- melt(qqGridRain,measure.vars=1:13)
qqRain_s <- melt(qqRain,measure.vars=1:13)
Rainqq <- data.frame(Station=qqRain_s$Var2,qqRain=qqRain_s$value,
                     qqGridRain=qqGridRain_s$value)
xyp_rainqq <- ggplot(data=Rainqq,aes(x=qqRain,y=qqGridRain))
xyp_rainqq <- xyp_rainqq + geom_point() + facet_wrap(~Station)
xyp_rainqq <- xyp_rainqq + geom_abline(intercept = 0, slope = 1,
                             col="darkred",linetype="dashed",size=1)

print(xyp_rainqq)
```



This shows fairly good agreement across the distributions with only the high rainfall quantiles showing distinct deviations. This is logical as the gridded rainfall is a smoothed replica of the station data due to the interpolation. Another important characteristic of rainfall could be the memory,so it might be worth comparing autocorrelation graphs for the different series.

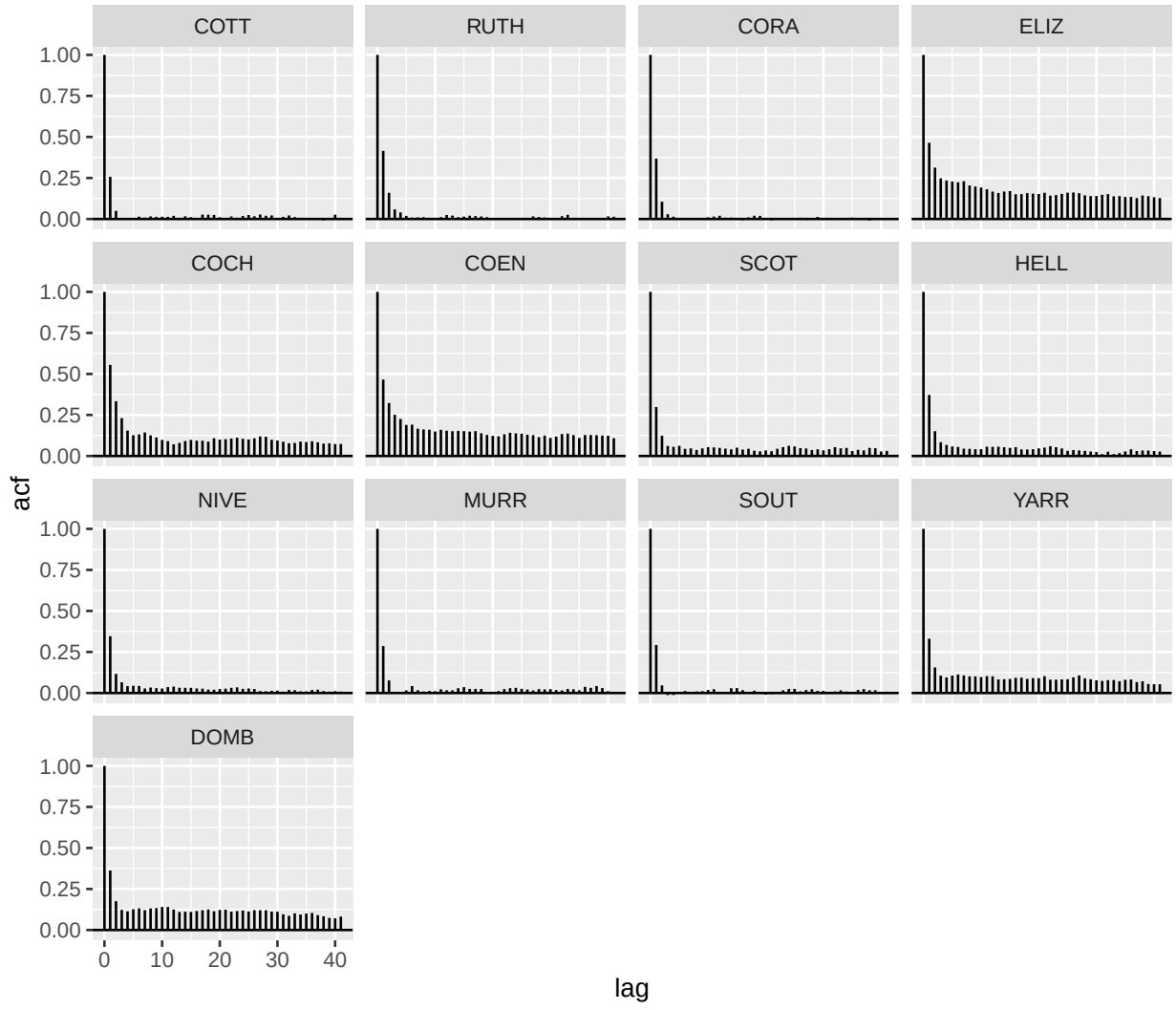
```
# Now show acfs
# Gridded rainfall
gridacf <- tapply(GridRainAllDataout$gridRain,
                            GridRainAllDataout$Station,
                            acf, plot = FALSE)
gridacfdf <- do.call(rbind,lapply(gridacf,function(x)
  with(x,data.frame(lag, acf))))
gridacfdf$Station <- rep(unique(GridRainAllDataout$Station),each=42)


acfGrid <- ggplot(data = gridacfdf, mapping = aes(x = lag, y = acf)) +
  geom_hline(aes(yintercept = 0)) +
  geom_segment(mapping = aes(xend = lag, yend = 0)) +
  facet_wrap(~Station)
```

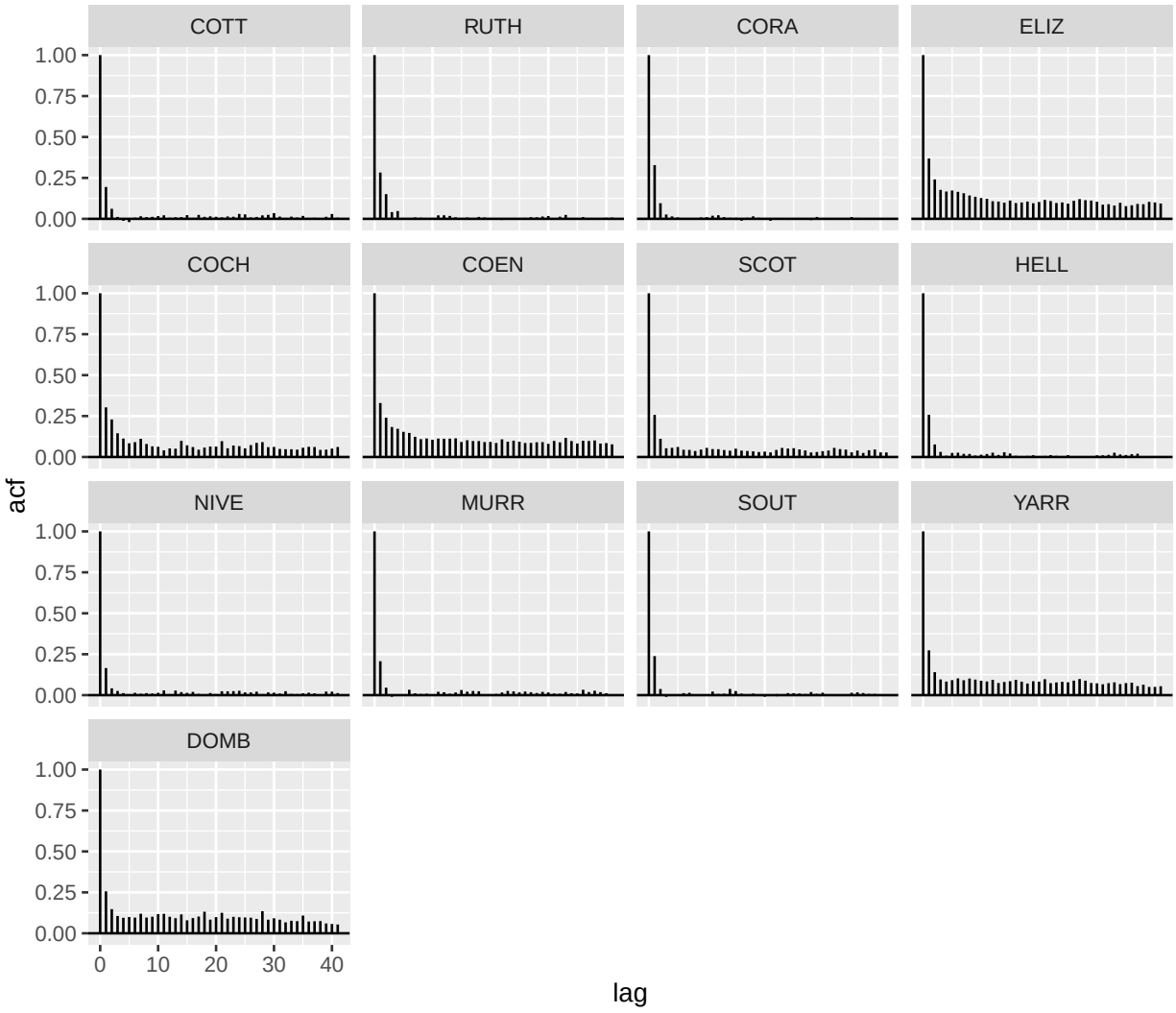
```
# Now normal rainfall
rainacf <- tapply(GridRainAllDataout$Rain,
                  GridRainAllDataout$Station,
                  acf, plot = FALSE,na.action=na.pass)
rainacfdf <- do.call(rbind,lapply(rainacf,function(x)
  with(x,data.frame(lag, acf))))
rainacfdf$Station <- rep(unique(GridRainAllDataout$Station),each=42)


acfRain <- ggplot(data = rainacfdf, mapping = aes(x = lag, y = acf)) +
  geom_hline(aes(yintercept = 0)) +
  geom_segment(mapping = aes(xend = lag, yend = 0)) +
  facet_wrap(~Station)

# Now print both
print(acfGrid)
```




```
print(acfRain)
```



This again shows little difference between the autocorrelation functions, with possibly the gridded rainfall (first graph) being slightly more auto-correlated than the station rainfall data (Second graph).

Storage, combining and publishing data

The last thing to do is to combine the data into a storage data object to be read in during all the analysis. Two different objects need to be developed. One is the daily data, combining the gridded rainfall data and the station rainfall data. The second object combines all the weekly data, and this means the gridded data needs to be combined to weekly.

```
for (i in 1:length(Stations[,1])) {
  gridrain_t <- apply.weekly(output.z[,i], sum)
  if (i ==1) {
      gridRainWeekly <- gridrain_t
    } else {
      gridRainWeekly <- merge(gridRainWeekly,gridrain_t,all=T)
    }
}
colnames(gridRainWeekly) <- colnames(output.z)
```

Stacking and merging into one dataset

Now stack all the data together into one dataframe

```
gridRain_weekly_stack <- data.frame(Date=time(gridRainWeekly),
                                    coredata(gridRainWeekly))
colnames(gridRain_weekly_stack) <- c("Date", paste("gridRain",
                                          Stations[,1],sep="."))

# add a column for the decade
for (j in 1:length(decade_start)) {
  gridRain_weekly_stack$decade[as.Date(gridRain_weekly_stack$Date) >=
                                 as.Date(decade_start[j]) &
                               as.Date(gridRain_weekly_stack$Date) <=
                                 as.Date(decade_end[j])] <-
    study_period_decades[j]
}
# stack
gridRain_weekly_stack <- reshape(gridRain_weekly_stack, direction="long",
                                 varying=2:14, sep=".")

# add to flow_rain_maxT_weekly
weekGridRainAllDataout <- cbind(flow_rain_maxT_weekly,gridRain_weekly_stack[,4])
colnames(weekGridRainAllDataout)[7] <- "gridRain"
```

Write data frames out as Rdata and csv

This section creates the on disk csv files and Rdata objects to be used in the rest of the research.

```
save(weekGridRainAllDataout,
     file="data/WeeklyDataIncludingGridded.Rdata")
write.csv(weekGridRainAllDataout,
          file="data/weeklyDataIncludingGridded.csv",
          row.names=F)

save(GridRainAllDataout,
     file="data/DailyDataIncludingGridded.Rdata")
write.csv(GridRainAllDataout,
          file="data/DailyDataIncludingGridded.csv",
          row.names=F)
```