Analysis-RTS

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

Objective

I perform an analysis of Mt Rainier time series along the same lines as the analysis of the accidental deaths (AD) series starting with lecture overhead III–82 (the 3rd set of R code on the course Web site has the code used to analyze the AD series).

Please feel free to alter choices that were made in the analysis of the AD series if you deem them to be inappropriate for your analysis of the climate series. I annotate with brief descriptions of the steps I took in your analysis. Finally, state briefly your conclusions about how well the simple modeling approach worked for the climate series.

```
onest <- read.csv("data/AB08-A2_hourly.csv")
onest$dt <- strptime(onest$DATE, format = "%Y-%m-%d %H:%M:%S")
```

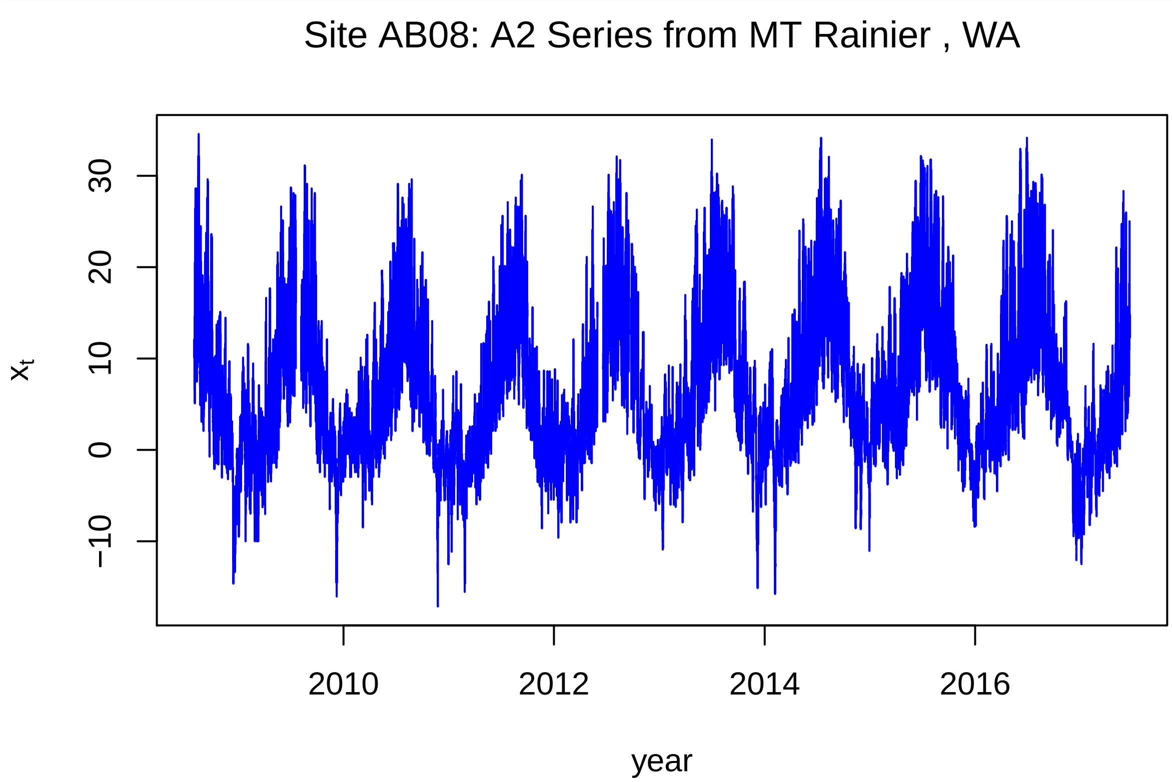
```
## Warning in strptime(onest$DATE, format = "%Y-%m-%d %H:%M:%S"): unknown
## timezone 'zone/tz/2017c.1.0/zoneinfo/America/Los_Angeles'
```

```
str(onest)
```

```
## 'data.frame':    77882 obs. of  4 variables:
##  $ DATE      : Factor w/ 77882 levels "2008-08-01 08:00:00",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ TZ        : Factor w/ 1 level "Etc/GMT-7": 1 1 1 1 1 1 1 1 1 1 ...
##  $ series_xts: num  10.1 10.7 11.1 11.4 11.6 ...
##  $ dt        : POSIXlt, format: "2008-08-01 08:00:00" "2008-08-01 09:00:00" ...
```

```
### define function to do filtering ...

filter.with.padding <- function(x,the.filter,iter=1)
{
    q <- (length(the.filter)-1)/2
    n <- length(x)
    w <- stats::filter(c(rep(x[1],q),x,rep(x[n],q)),the.filter)[(q+1):(q+n)]
    if(iter > 1) for(i in 2:iter) w <- filter(c(rep(w[1],q),w,rep(w[n],q)),the.filter)[(q+1):(q+n)]
    return(w)
}

plot.ACFest <- function(ts, main=NULL, n.lags=40)
{
    ts.acf <- acf(ts, lag.max=n.lags, plot=FALSE)
    n.ts <- length(ts)
```

```
    xs <- 1:n.lags
    ys <- ts.acf$acf[2:(n.lags+1)]
    plot(xs,ys,typ="h",xlab="h  (lag)",ylab="ACF",ylim=c(-1,1),col="blue",main=main)
    points(xs,ys,col="red",cex=0.5)
    xs <- 1:n.lags
    xs[1] <- xs[1] - 0.25
    xs[n.lags] <- xs[n.lags] + 0.25
    lines(xs,1.96*sqrt(n.ts-xs)/n.ts,col="magenta",lty="dashed")
    lines(xs,-1.96*sqrt(n.ts-xs)/n.ts,col="magenta",lty="dashed")
    abline(h=0,lty="dashed")
    CI.hw <- 1.96/sqrt(n.ts)
    lines(c(0.75,n.lags+0.25),rep(CI.hw,2),col="blue",lty="dashed")
    lines(c(0.75,n.lags+0.25),rep(-CI.hw,2),col="blue",lty="dashed")
    return(ts.acf$acf)
}
### overhead III-2

plot(onest$dt,onest$series_xts,col="blue",xlab="year",typ="l",
     ylab=expression(x[t]),main=expression(paste("Site AB08: A2",
                                                 " Series from MT Rainier , WA")))
```



III–85 - Seasonal component taken out

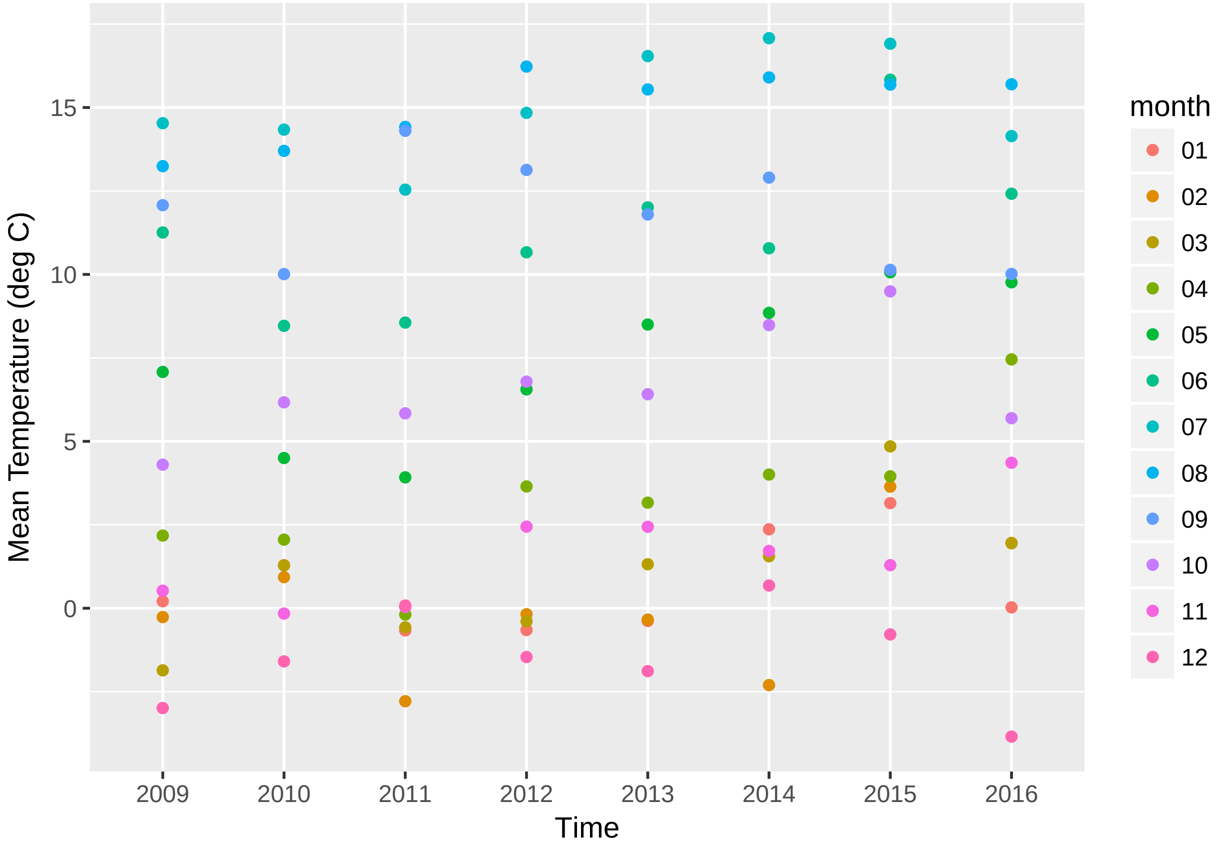
Use a smoothing filter to take the seasonal component out

Better explanation needed.

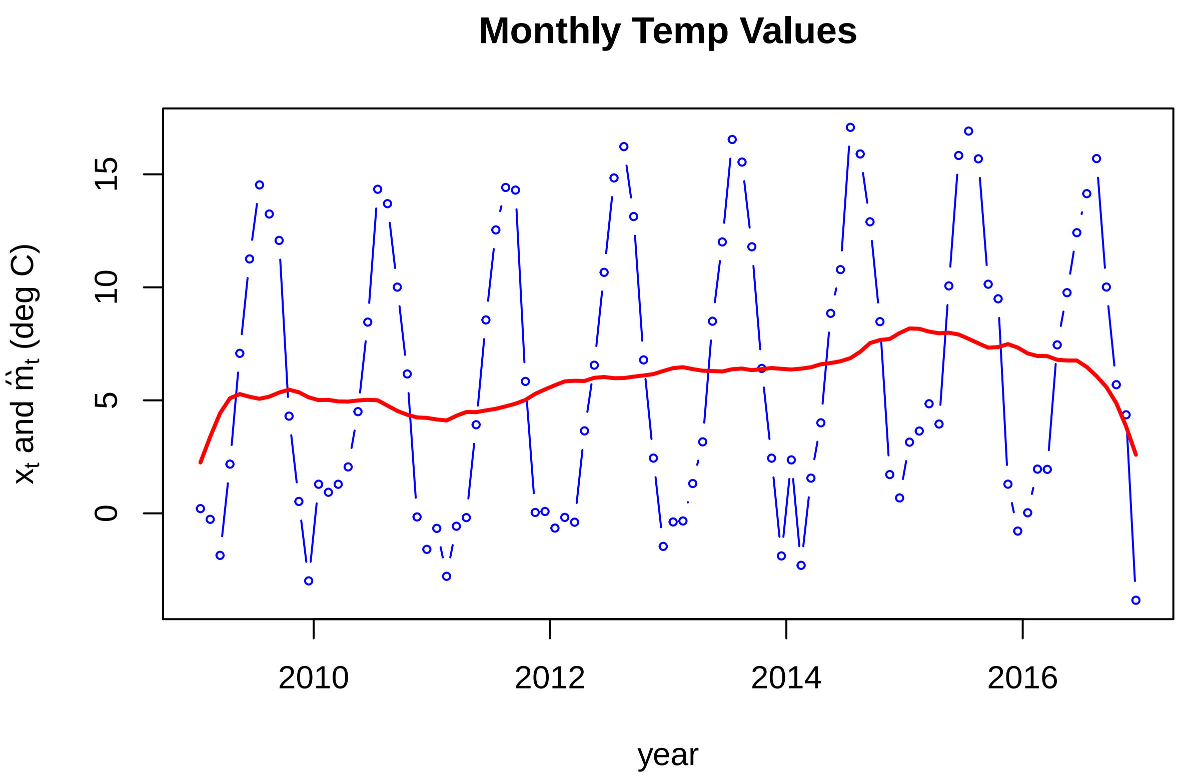
III–85

```
#ggplot needs date in POSIXct
onest$dt_txn <- as.POSIXct(onest$DATE,ormat = "%Y-%m-%d %H:%M:%S")
# Add month column
# Add day column
# Add hour column
onest$month <-format(onest$dt_txn,"%m")
onest$day <- format(onest$dt_txn,"%d")
onest$hour <- format(onest$dt_txn,"%H")


#subset it to only 9 years
# Removed 1288 rows containing missing values
oneset_9yrs <- onest %>% select(c('dt_txn','series_xts','year','month','day','hour')) %>%
  filter(!is.na(series_xts) & year > 2008 & year < 2017) %>% as.data.frame()



#Summarize by months
oneset_9yrs_by_month <- oneset_9yrs %>% group_by(year,month) %>% summarise(min_t=min(series_xts),max_t=

# Add key
oneset_9yrs_by_month$xaxis <- as.double(paste(oneset_9yrs_by_month$year,oneset_9yrs_by_month$month,sep =

#1 .042
#2 .125
#3 .208
#4 .292
#5 .375
#6 .458
#7 .542
#8 .625
#9 .708
#10 .792
#11 .875
#12 .958

#create new xaxis
oneset_9yrs_by_month$xaxisred <- "0"
#Redo x-axis
oneset_9yrs_by_month[oneset_9yrs_by_month$month=="01",]$xaxisred = "042"
oneset_9yrs_by_month[oneset_9yrs_by_month$month=="02",]$xaxisred = "125"
oneset_9yrs_by_month[oneset_9yrs_by_month$month=="03",]$xaxisred = "208"
oneset_9yrs_by_month[oneset_9yrs_by_month$month=="04",]$xaxisred = "292"
oneset_9yrs_by_month[oneset_9yrs_by_month$month=="05",]$xaxisred = "375"
oneset_9yrs_by_month[oneset_9yrs_by_month$month=="06",]$xaxisred = "458"
oneset_9yrs_by_month[oneset_9yrs_by_month$month=="07",]$xaxisred = "542"
oneset_9yrs_by_month[oneset_9yrs_by_month$month=="08",]$xaxisred = "625"
oneset_9yrs_by_month[oneset_9yrs_by_month$month=="09",]$xaxisred = "708"
oneset_9yrs_by_month[oneset_9yrs_by_month$month=="10",]$xaxisred = "792"
oneset_9yrs_by_month[oneset_9yrs_by_month$month=="11",]$xaxisred = "875"
oneset_9yrs_by_month[oneset_9yrs_by_month$month=="12",]$xaxisred = "958"
```

```
oneset_9yrs_by_month$nxaxis <- as.double(paste(oneset_9yrs_by_month$year,oneset_9yrs_by_month$xaxisred,

# For every year we should have 12 values

oneset_9yrs_by_month %>% ggplot(aes(year,mean_t,color=month)) + geom_point() +xlab("Time") + ylab("Mean
```




```
m.hat.oneset <- filter.with.padding(oneset_9yrs_by_month$mean_t,c(1/24,rep(1/12,11),1/24))

#plot(oneset_9yrs_by_month$xaxis,oneset_9yrs_by_month$mean_t,col="blue",xlab="year",typ="b",
#     ylab=expression(paste(x[t]," and ", hat(m)[t]," (deg C)")),main="Monthly Temp Values",cex=0.5)
#lines(oneset_9yrs_by_month$xaxis,m.hat.oneset,col="red",lwd=2)

plot(oneset_9yrs_by_month$nxaxis,oneset_9yrs_by_month$mean_t,col="blue",xlab="year",typ="b",
     ylab=expression(paste(x[t]," and ", hat(m)[t]," (deg C)")),main="Monthly Temp Values",cex=0.5)
lines(oneset_9yrs_by_month$nxaxis,m.hat.oneset,col="red",lwd=2)
```

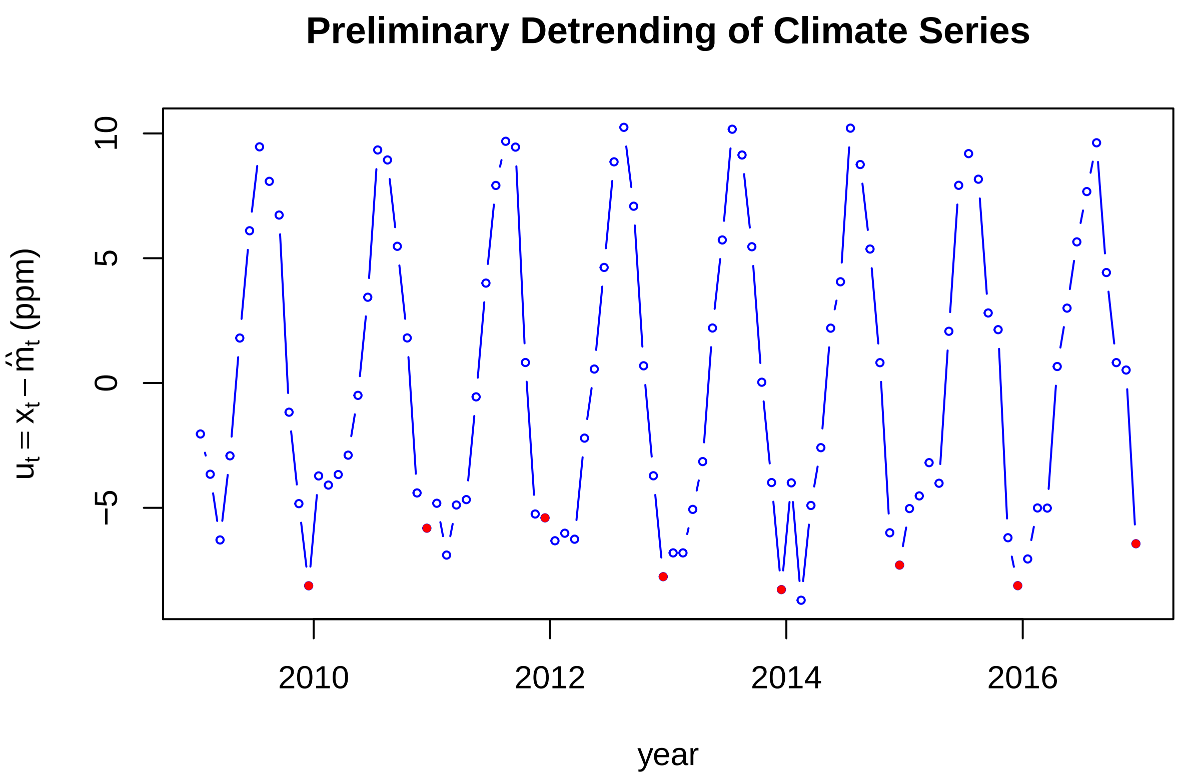


###

III–87

Removing the trend component and plotting.

```
oneset.u <- oneset_9yrs_by_month$mean_t - m.hat.oneset

plot(oneset_9yrs_by_month$nxaxis,oneset.u,col="blue",xlab="year",typ="b",ylab=expression(paste(u[t]==x[t
points(oneset_9yrs_by_month$nxaxis[seq(12,96,12)],oneset.u[seq(12,96,12)],pch=16,col="red",cex=0.6)
```

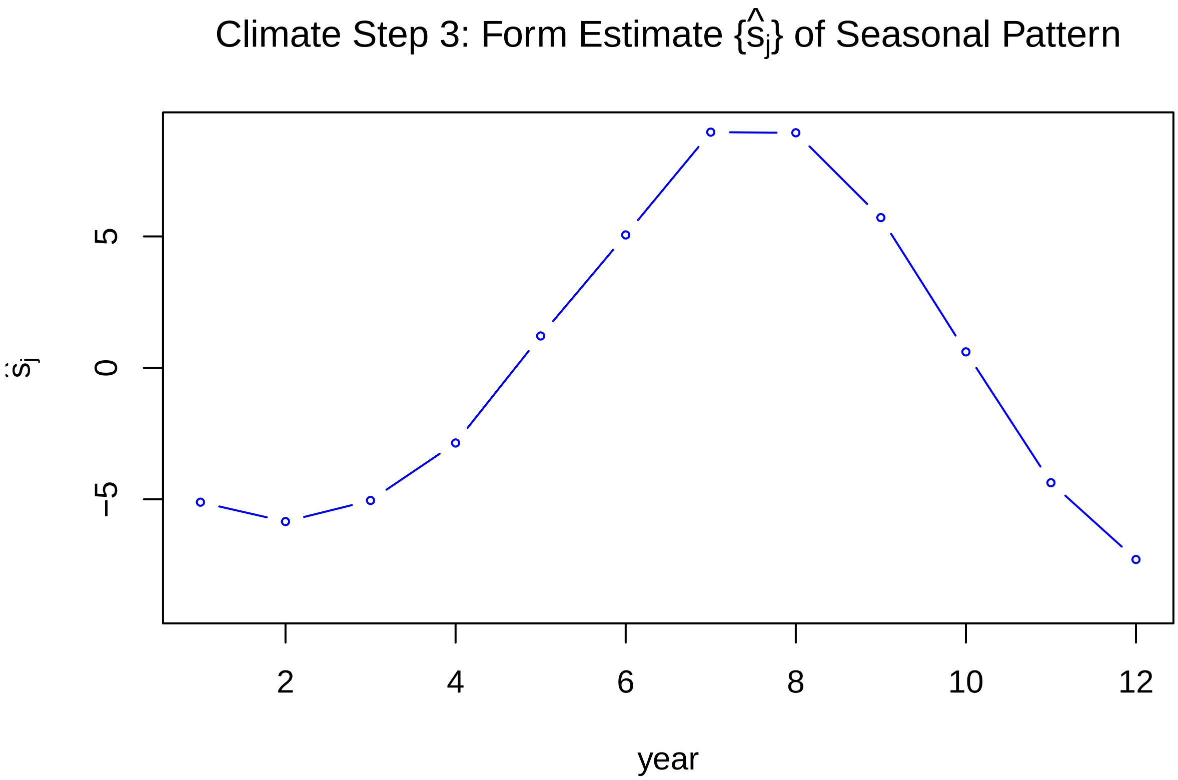


###

III–89

Extracting for one year to show the seasonal pattern.

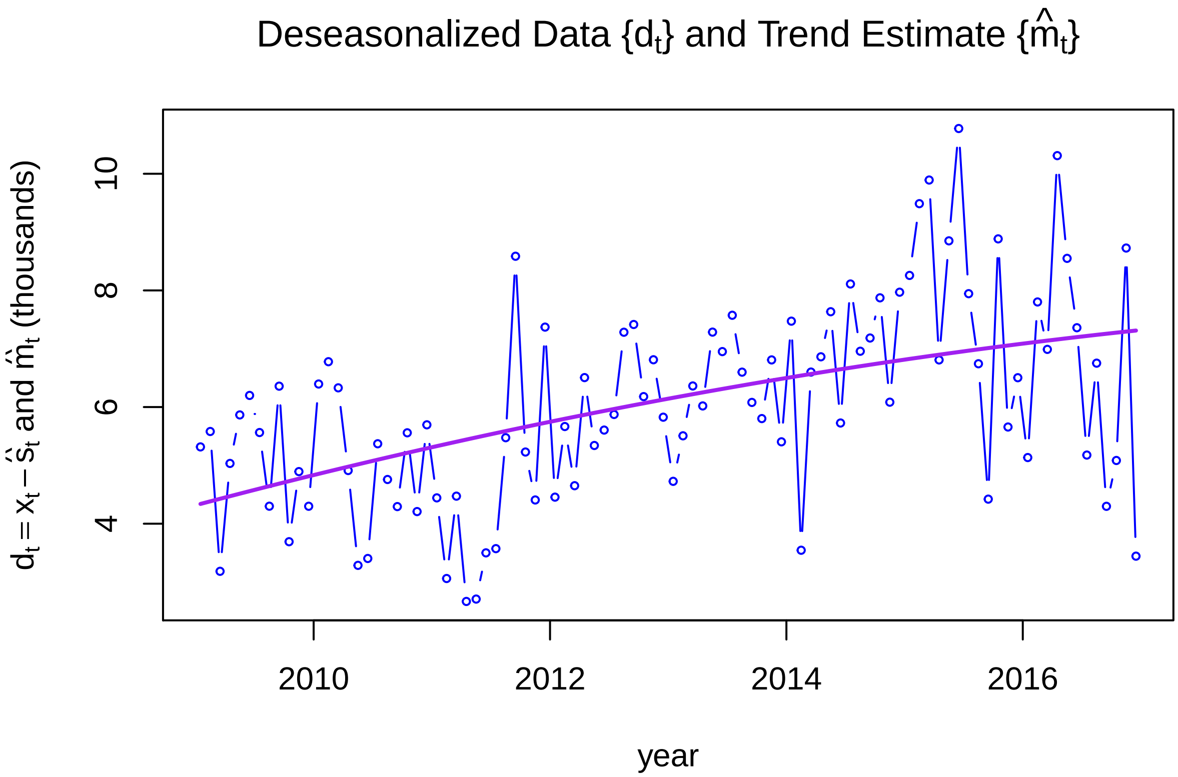
```
oneset.w.j <- rowMeans(matrix(oneset.u ,nrow=12))

plot(1:12,oneset.w.j- mean(oneset.w.j),col="blue",xlab="year",
     typ="b",ylab=expression(hat(s)[j]),
     main=expression(paste("Climate Step 3: Form Estimate {",hat(s)[j],"} of Seasonal Pattern")),
     ylim=c(-9,9),cex=0.5)
```



Verify it by the seasonal plot

III–92 (Deasonalized data with trend estimate)

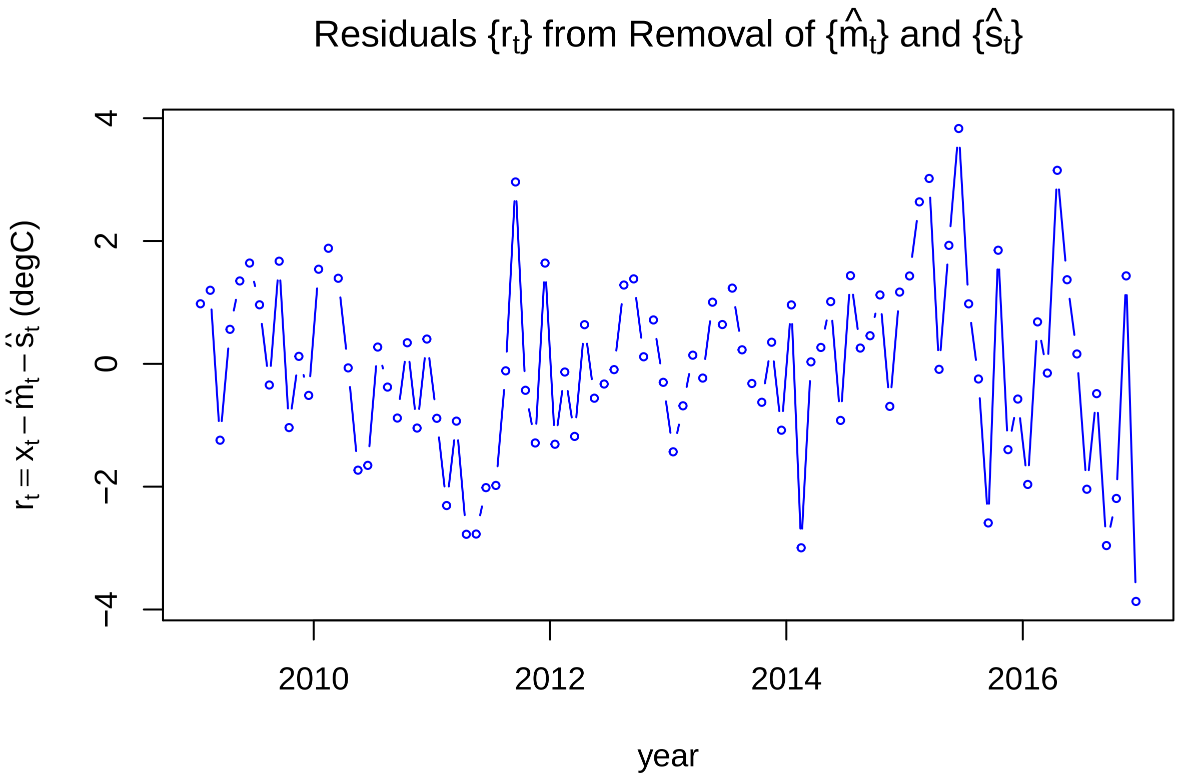
```
#get all the years(columns), rows(minths)
#average all the jan, feb... dec
oneset.w.j <- rowMeans(matrix(oneset.u ,nrow=12))
oneset.s.j.hat <- rep(oneset.w.j  - mean(oneset.w.j ),8)
oneset.d <- oneset_9yrs_by_month$mean_t - oneset.s.j.hat

oneset.d.reg <- lm(oneset.d  ~ oneset_9yrs_by_month$nxaxis + I(oneset_9yrs_by_month$nxaxis^2))

plot(oneset_9yrs_by_month$nxaxis,oneset.d,col="blue",xlab="year",typ="b",
     ylab=expression(paste(d[t] == x[t] - hat(s)[t]," and ", hat(m)[t]," (thousands)")),
     main=expression(paste("Deseasonalized Data {", d[t], "} and Trend Estimate {",hat(m)[t],"}")),
     cex=0.5)
lines(oneset_9yrs_by_month$nxaxis,fitted(oneset.d.reg ),col="purple",lwd=2)
```



###

III–94 Residuals removed

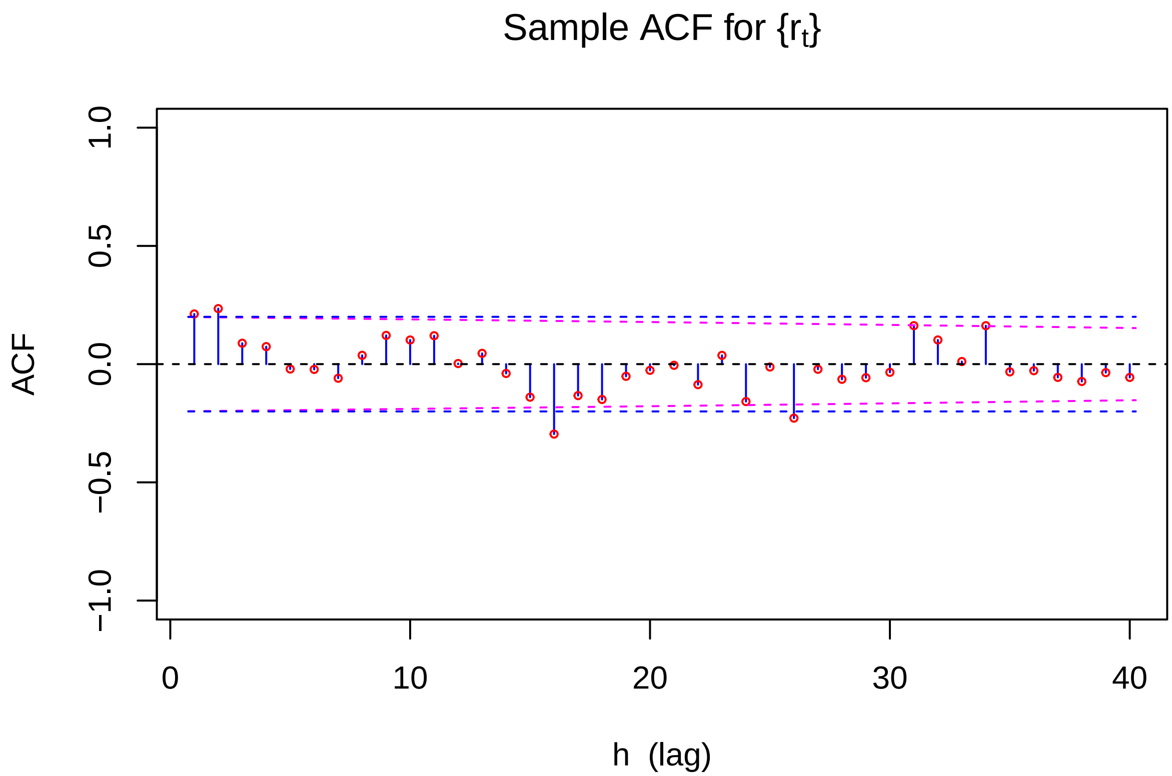
```
plot(oneset_9yrs_by_month$nxaxis,resid(oneset.d.reg),col="blue",xlab="year",typ="b",
     ylab=expression(paste(r[t] == x[t] - hat(m)[t] - hat(s)[t]," (degC)")),
     main=expression(paste("Residuals {",r[t],"} from Removal of {", hat(m)[t],"} and {",hat(s)[t],"}")
     cex=0.5)
```



III–96

** Fails null hypothesis ** Plotting the residuals and showing the 95%CIs

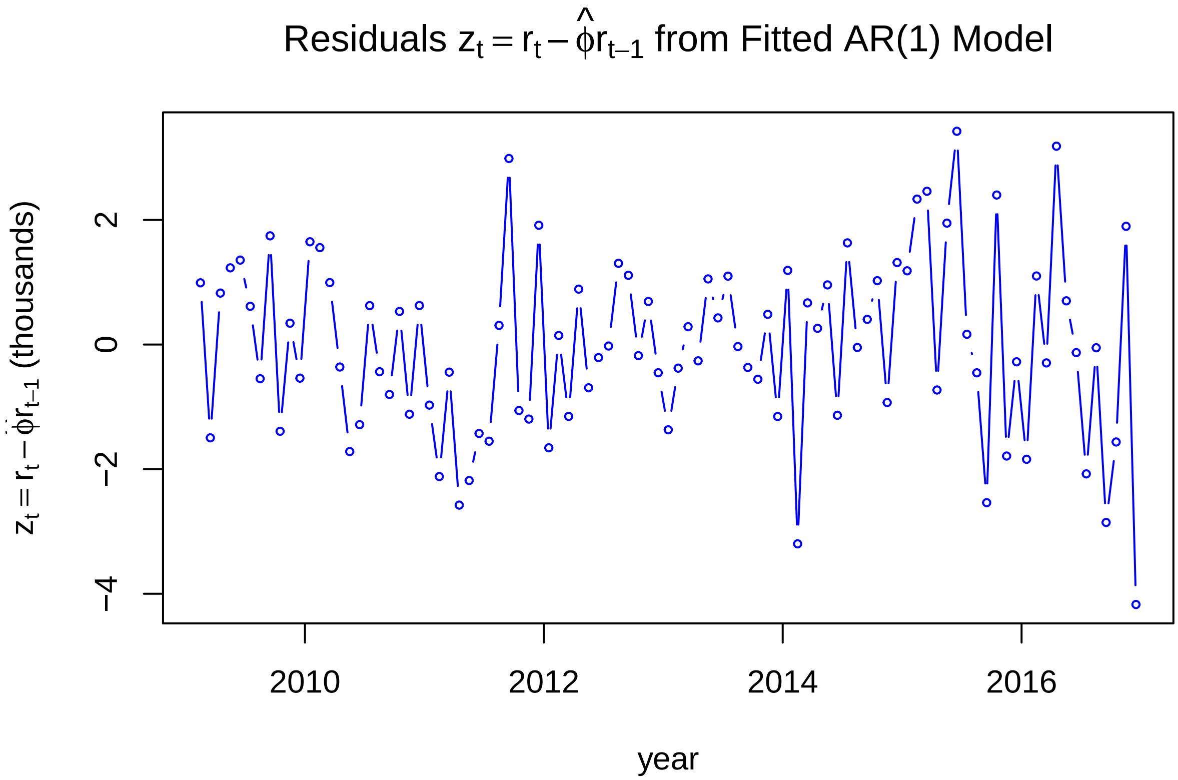
```
phi <- plot.ACFest(resid(oneset.d.reg),
         expression(paste("Sample ACF for {", r[t],"}")))[2]
```



III–97

Testing whether the resultant is a AR(1) model.

```
oneset.z <- resid(oneset.d.reg)[-1] - phi*resid(oneset.d.reg)[-length(resid(oneset.d.reg))]

plot(oneset_9yrs_by_month$nxaxis[-1],oneset.z ,col="blue",xlab="year",
     typ="b",ylab=expression(paste(z[t]==r[t]-hat(phi)*r[t-1]," (thousands)")),
     main=expression(paste("Residuals ",
       z[t]==r[t]-hat(phi)*r[t-1]," from Fitted AR(1) Model")),
     cex=0.5)
```

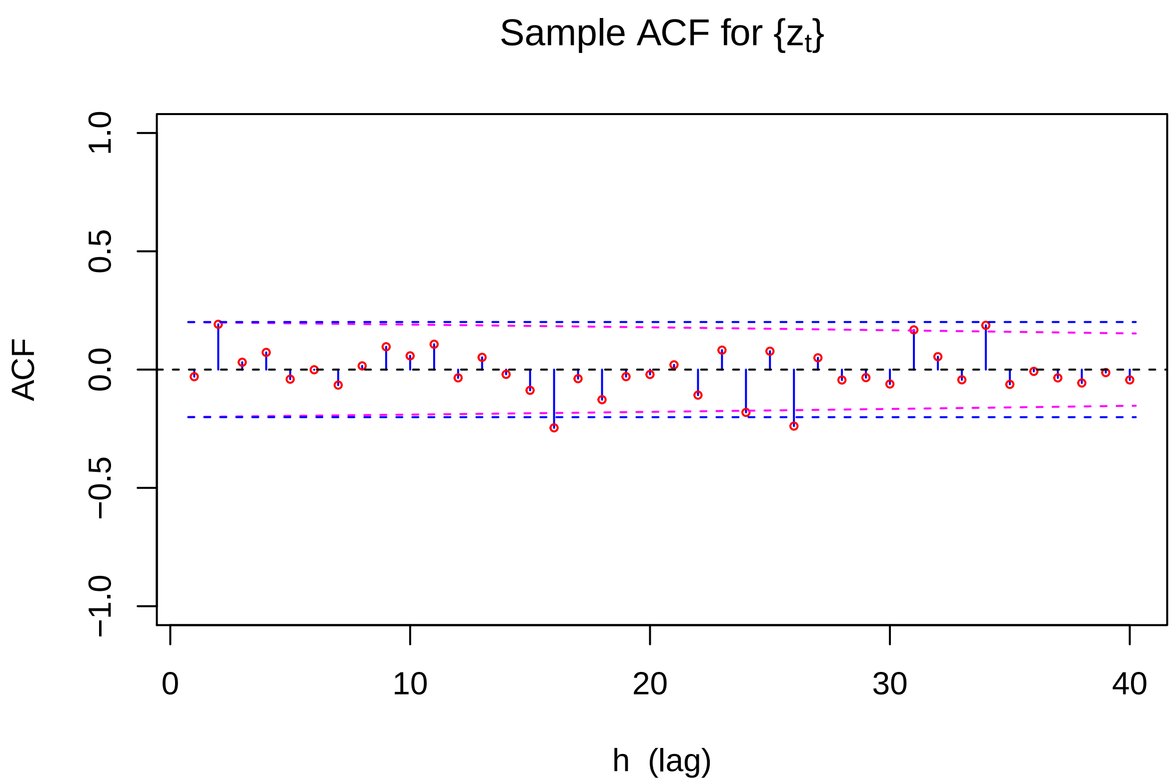


###

III–98

ACF for residuals from the fitted AR(1) model, very good, but three exceptions - not that bad

```
plot.ACFest(oneset.z, expression(paste("Sample ACF for {", z[t],"}")))[2]
```



```
## [1] -0.02988363
```