meeting report 6.10

linyiguo

2019.6.10

Well, I did not send Aaron meeting report this time, cause I did not read much things recently, all content we talked was the things I have done before. And the meeting result compared with the former was better. I guess Aaron and I are getting used with each other? LOL.  
I hope to plot something to clearfy the point I want to express, but unfortunatly I can’t. I will add some photos in this folder.

# I

Note the curve of *trend + season* in **report 6.6** is pretty close to our data, which is totally opposite to the situation from StatCan… although I am not sure how they get the plot, at least I believe what I got is reasonable.  
Here is the thing, is , and differences between our data and trend+season actually reflect the deviation of the noise . By controling the deviation of the noise, we can control the smoothness of the trend+season curve(since t.s is fixed).

——————————————–**UPDATE 6.12**——————————————–  
昨天试着理清为什么Aaron说可以change into . But I still do not understand the theory behind it clearly… I give up temporarily :) Let’s see whether this could work in our s.s model before(*report 6.6*) ## change sd in expression of w from 1 to 5

library(seasonal)  
library(forecast)

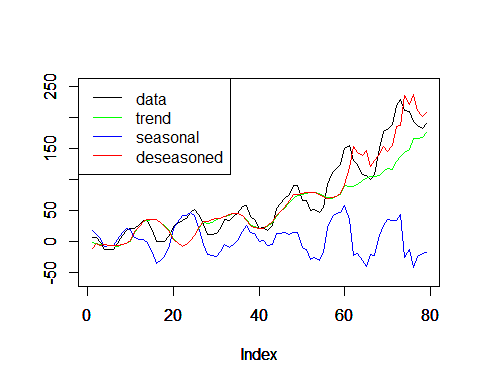
## Registered S3 methods overwritten by 'ggplot2':  
## method from   
## [.quosures rlang  
## c.quosures rlang  
## print.quosures rlang

## Registered S3 method overwritten by 'xts':  
## method from  
## as.zoo.xts zoo

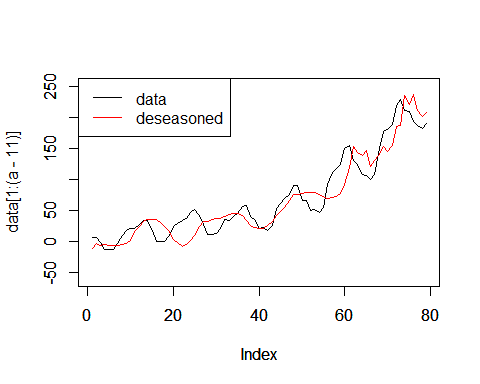
## Registered S3 method overwritten by 'quantmod':  
## method from  
## as.zoo.data.frame zoo

## Registered S3 methods overwritten by 'forecast':  
## method from   
## fitted.fracdiff fracdiff  
## residuals.fracdiff fracdiff

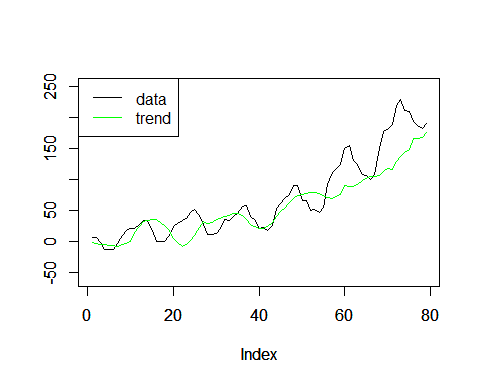
set.seed(1)  
  
# generate data  
model <- Arima(ts(rnorm(24000),freq=12), order=c(0,1,1), seasonal=c(0,1,1),fixed=c(theta=0.5, Theta=0.5))  
data <- simulate(model,nsim=240)  
  
# define m\_x11  
m\_x11 <- seas(data, x11 = "", regression.aictest = NULL)  
  
# Initialization  
S <- matrix(0,1000,251) # Cause Seasonal component's state space model, we have additional 11 zero-values.  
Tr <- matrix(0,1000,251)  
Tr[,11] <- data[1]  
S[,1:11] <- rep(as.numeric(data-final(m\_x11))[1:11],1000)  
component <- c()  
a = 90 # a-11 is the length of our t.s.  
  
for (i in 12:a) {  
   
 # update particles  
 Tr[,i] <- Tr[,i-1] + rnorm(1000,sd=5)  
 for (j in 1:11) S[,i] <- S[,i]-S[,i-j]  
 S[,i] <- S[,i] + rnorm(1000,sd=5)  
   
 # update weights  
 w <- dnorm(data[i-11]-Tr[,i]-S[,i],sd=5)  
 w <- w/sum(w)  
   
 # evaluate state value  
 t <- sum(w \* Tr[,i])  
 s <- sum(w \* S[,i])  
   
 # add to our component path  
 component <- rbind(component, c(t,s))  
   
 # resample  
 Tr[,i] <- sample(Tr[,i], size =1000, replace = TRUE, prob = w)  
 S[,i] <- sample(S[,i], size = 1000, replace = TRUE, prob = w)  
}  
  
# plot four curves together  
plot(data[1:(a-11)],type = "l",ylim = c(-60,250),ylab='')  
par(new=TRUE)  
plot(component[,1],type="l",col="green",ylim = c(-60,250),ylab='')  
par(new=TRUE)  
plot(component[,2],type="l",col="blue",ylim = c(-60,250),ylab='')  
par(new=TRUE)  
plot(data[1:(a-11)]-component[,2],type="l",col="red",ylim=c(-60,250),ylab='')  
legend("topleft",c("data","trend","seasonal","deseasoned"),col=c("black","green","blue","red"),lty=c(1,1,1,1))



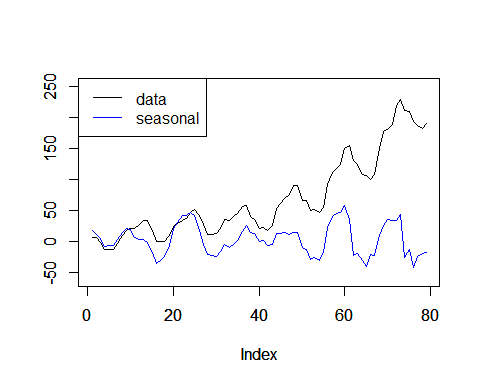
# data vs deseasoned  
plot(data[1:(a-11)],type = "l",ylim = c(-60,250))  
par(new=TRUE)  
plot(data[1:(a-11)]-component[,2],type="l",col="red",ylim=c(-60,250),ylab='')  
legend("topleft", c("data","deseasoned"),col=c("black","red"),lty=c(1,1))



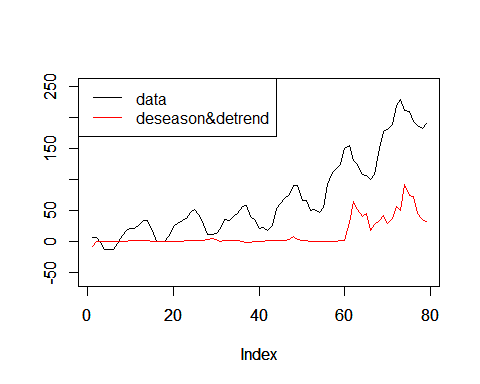
# data vs trend  
plot(data[1:(a-11)],type = "l",ylim = c(-60,250),ylab='')  
par(new=TRUE)  
plot(component[,1],type="l",col="green",ylim = c(-60,250),ylab='')  
legend("topleft",c("data","trend"),col=c("black","green"),lty=c(1,1))



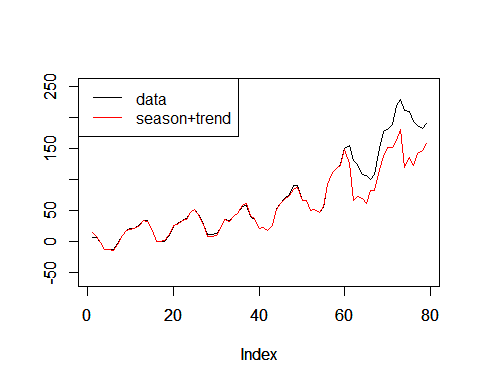
# data vs season  
plot(data[1:(a-11)],type = "l",ylim = c(-60,250),ylab='')  
par(new=TRUE)  
plot(component[,2],type="l",col="blue",ylim = c(-60,250),ylab='')  
legend("topleft",c("data","seasonal"),col=c("black","blue"),lty=c(1,1))



# data vs deseason&detrend  
plot(data[1:(a-11)],type = "l",ylim = c(-60,250),ylab='')  
par(new=TRUE)  
plot(data[1:(a-11)]-component[,1]-component[,2], type="l",col="red",ylim = c(-60,250),ylab='')  
legend("topleft",c("data","deseason&detrend"),col=c("black","red"),lty=c(1,1))

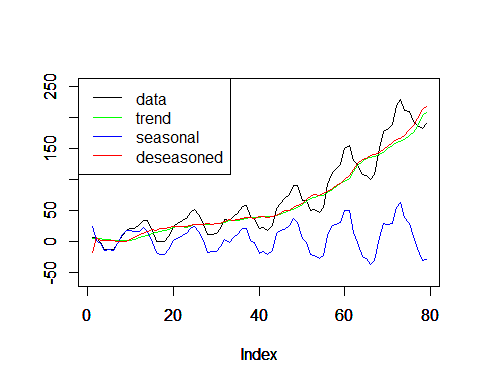


# data vs seanson+trend  
plot(data[1:(a-11)],type = "l",ylim = c(-60,250),ylab='')  
par(new=TRUE)  
plot(component[,1]+component[,2], type="l",col="red",ylim = c(-60,250),ylab='')  
legend("topleft",c("data","season+trend"),col=c("black","red"),lty=c(1,1))

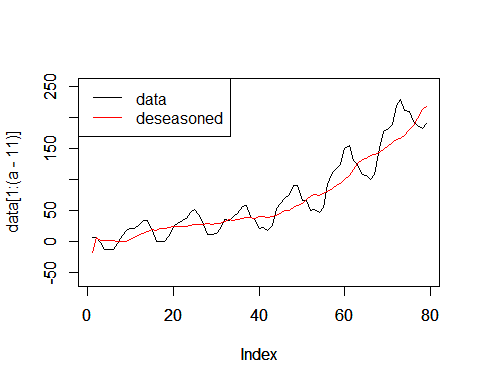


———————————————**UPDATE 6.16**———————————————  
I am busy with my resume and the self-recommendation letter recently… I need to meet aaron tmr, but for now I don’t have any things to show him, which is very very awkward… ## sd=20 in weights expression

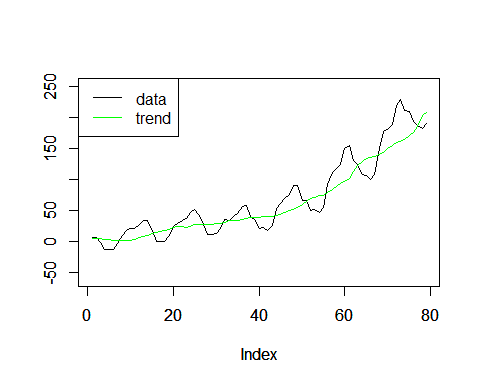
# Initialization  
S <- matrix(0,1000,251) # Cause Seasonal component's state space model, we have additional 11 zero-values.  
Tr <- matrix(0,1000,251)  
Tr[,11] <- data[1]  
S[,1:11] <- rep(as.numeric(data-final(m\_x11))[1:11],1000)  
component <- c()  
a = 90 # a-11 is the length of our t.s.  
  
for (i in 12:a) {  
   
 # update particles  
 Tr[,i] <- Tr[,i-1] + rnorm(1000,sd=5)  
 for (j in 1:11) S[,i] <- S[,i]-S[,i-j]  
 S[,i] <- S[,i] + rnorm(1000,sd=5)  
   
 # update weights  
 w <- dnorm(data[i-11]-Tr[,i]-S[,i],sd=20)  
 w <- w/sum(w)  
   
 # evaluate state value  
 t <- sum(w \* Tr[,i])  
 s <- sum(w \* S[,i])  
   
 # add to our component path  
 component <- rbind(component, c(t,s))  
   
 # resample  
 Tr[,i] <- sample(Tr[,i], size =1000, replace = TRUE, prob = w)  
 S[,i] <- sample(S[,i], size = 1000, replace = TRUE, prob = w)  
}  
  
# plot four curves together  
plot(data[1:(a-11)],type = "l",ylim = c(-60,250),ylab='')  
par(new=TRUE)  
plot(component[,1],type="l",col="green",ylim = c(-60,250),ylab='')  
par(new=TRUE)  
plot(component[,2],type="l",col="blue",ylim = c(-60,250),ylab='')  
par(new=TRUE)  
plot(data[1:(a-11)]-component[,2],type="l",col="red",ylim=c(-60,250),ylab='')  
legend("topleft",c("data","trend","seasonal","deseasoned"),col=c("black","green","blue","red"),lty=c(1,1,1,1))



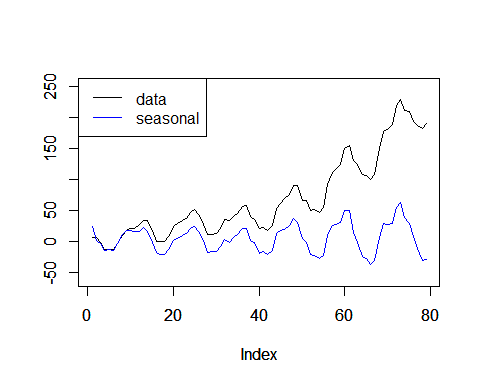
# data vs deseasoned  
plot(data[1:(a-11)],type = "l",ylim = c(-60,250))  
par(new=TRUE)  
plot(data[1:(a-11)]-component[,2],type="l",col="red",ylim=c(-60,250),ylab='')  
legend("topleft", c("data","deseasoned"),col=c("black","red"),lty=c(1,1))



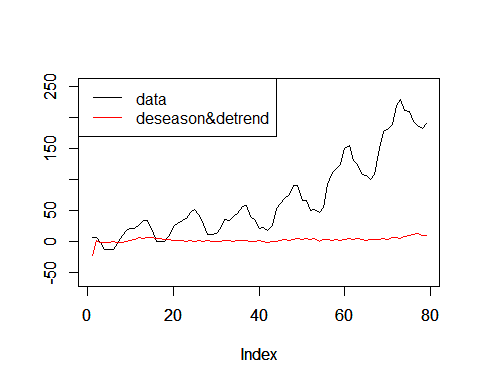
# data vs trend  
plot(data[1:(a-11)],type = "l",ylim = c(-60,250),ylab='')  
par(new=TRUE)  
plot(component[,1],type="l",col="green",ylim = c(-60,250),ylab='')  
legend("topleft",c("data","trend"),col=c("black","green"),lty=c(1,1))



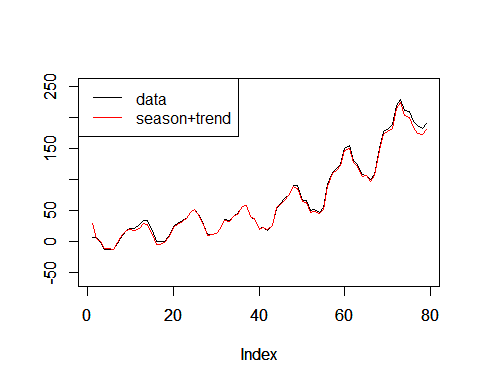
# data vs season  
plot(data[1:(a-11)],type = "l",ylim = c(-60,250),ylab='')  
par(new=TRUE)  
plot(component[,2],type="l",col="blue",ylim = c(-60,250),ylab='')  
legend("topleft",c("data","seasonal"),col=c("black","blue"),lty=c(1,1))



# data vs deseason&detrend  
plot(data[1:(a-11)],type = "l",ylim = c(-60,250),ylab='')  
par(new=TRUE)  
plot(data[1:(a-11)]-component[,1]-component[,2], type="l",col="red",ylim = c(-60,250),ylab='')  
legend("topleft",c("data","deseason&detrend"),col=c("black","red"),lty=c(1,1))



# data vs seanson+trend  
plot(data[1:(a-11)],type = "l",ylim = c(-60,250),ylab='')  
par(new=TRUE)  
plot(component[,1]+component[,2], type="l",col="red",ylim = c(-60,250),ylab='')  
legend("topleft",c("data","season+trend"),col=c("black","red"),lty=c(1,1))



Well, after enlarging the deviation of noise, the curves of trend and deseasonal become much more smooth, which should be something like that from StatCan.

I think I should understand a little bit about why aaron told me to add a demoninator in weights expression. Like we said before, we can control the smoothness of our curves by controling the deviation of our noise. Let’s say we want

which means

or

and

that is

And in R we can express in two ways:  
- dnorm((Y\_t-T\_t-S\_t)\5)  
- dnrom(Y\_t-T\_t-S\_t, sd=5)  
We choose the second one.