npc_AAA

Brittany

10/29/2018

load npc data

```
dnpc <- read.csv("New Policy Count.csv", sep = "\t")
dnpc <- dnpc[,1:3] #get rid of cols
names(dnpc) <- c("t", "date", "y") #name cols
dnpc <- subset(dnpc, !is.na(y)) #get rid of nans (missing data for most recent months)
dnpc$y <-  dnpc$y/1e3 #make smaller
dnpc$date <- as.Date(dnpc$date, '%m/%d/%Y')
yts <- ts(dnpc$y, start=c(2011,1), frequency=12) #turn into time series class
dygraph(yts)
```

fit AAA model w/ ets

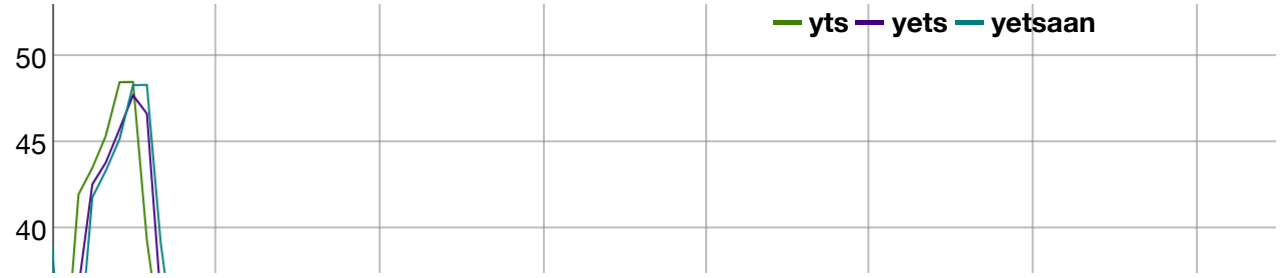
```
etsfit <- ets(yts,model='AAA')
yets <- fitted.values(etsfit)
etsfit
```

```
## ETS(A,A,A)
##
## Call:
##  ets(y = yts, model = "AAA")
##
##   Smoothing parameters:
##     alpha = 0.9739
##     beta  = 1e-04
##     gamma = 1e-04
##
##   Initial states:
##     l = 41.0336
##     b = -0.3068
##     s = -2.2885 -3.7537 -2.7609 -1.1166 2.1649 3.6611
##            4.0441 3.2941 2.6625 1.6332 -2.6673 -4.8729
##
##   sigma:  2.2148
##
##      AIC     AICc      BIC
## 571.6081 579.9917 614.2928
```

```
yetsaan <- fitted.values(ets(yts,model='AAN'))

#ets initialization
l1ets <- 41.0336
b1ets <- -0.3068
s1ets <- c(-2.2885, -3.7537, -2.7609, -1.1166, 2.1649, 3.6611, 4.0441, 3.2941, 2.6625,
1.6332, -2.6673, -4.8729)

#ets optimized params
alphaets = 0.9739
betaets  = 1e-04
gammaets = 1e-04
sigets <- 2.2148
```

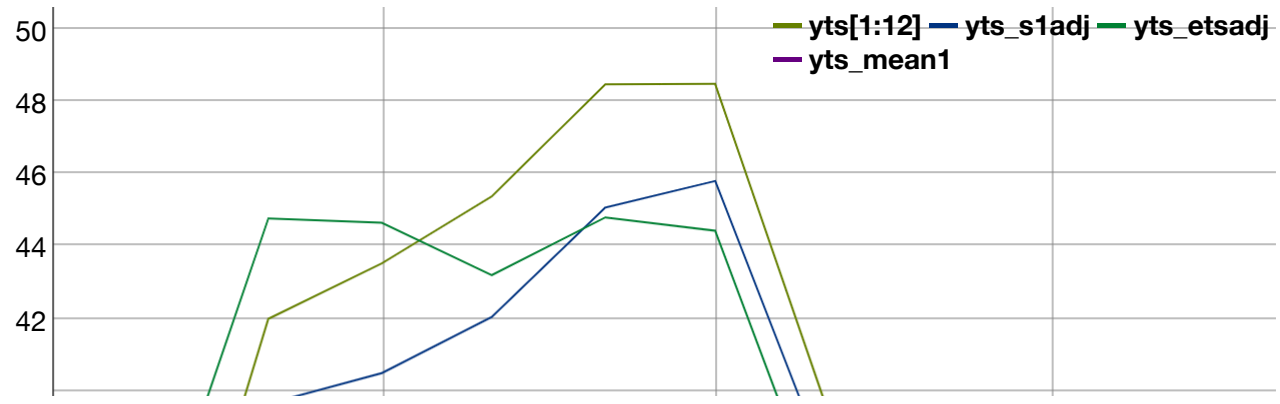
plot npc time series and AAA/AAN model

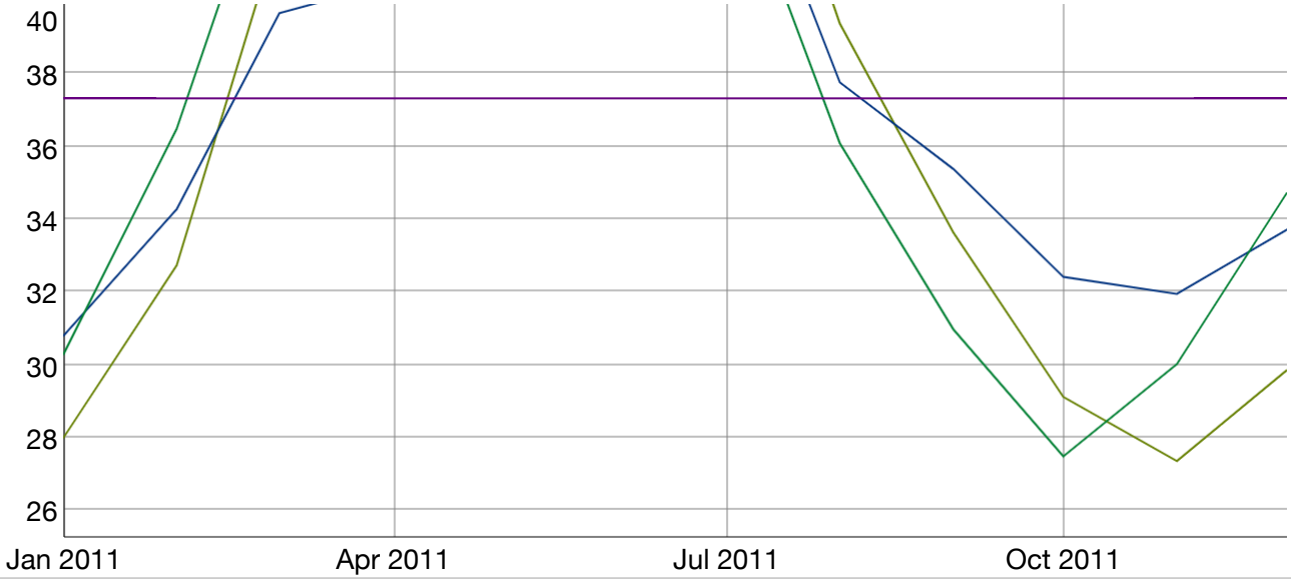
```
dygraph(cbind(yts,yets,yetsaan))
```



get initial params

```
m <- 12
N <- nrow(dnpc)
Ns <-  as.integer(N/m+1)


#get yearly avg
savg <- vector()
for (j in 1:Ns) {
  savgsum <- 0.
  scount <- 0
  for (k in (m*(j-1)+1):(m*j)) {
    #print(c(j,m,k))
    if (k<=N){
      savgsum <- savgsum + yts[k]
      scount <- scount + 1
      #print( c(j,k,savgsum,scount) )
    }
  }
  savg[j] <- savgsum/scount
}

#initial seasonality
#get avg diff from average for each period
s0 <- vector()
for (i in 1:m) {
  value <-  0
  vcount <-  0
  for (j in 1:Ns) {
    if ((m*(j-1)+i)<=N) {
      value <- value + yts[m*(j-1)+i]-savg[j]
      vcount <- vcount + 1
      #print( c(i,j,m*(j-1)+i,value,vcount) )
    }
  }
  s0[i] <-  value/vcount
}

yts_s1adj <-  ts(yts[1:12]-s0, start=c(2011,1), frequency=12)
yts_etsadj <- ts(yts[1:12]-s1ets, start=c(2011,1), frequency=12)
yts_mean1 <- ts(rep(mean(yts[1:12]),times=12), start=c(2011,1), frequency=12)

dygraph(cbind(yts[1:12],yts_s1adj,yts_etsadj,yts_mean1)) #should this be flatter?
```






```
#get l0 and b0 from linear regression of first 10 data points
x <- 1:10
initreg <- lm(yts_s1adj[x]~x)
l0 <- summary(initreg)$coefficients['(Intercept)','Estimate']
b0 <- summary(initreg)$coefficients['x','Estimate']
```

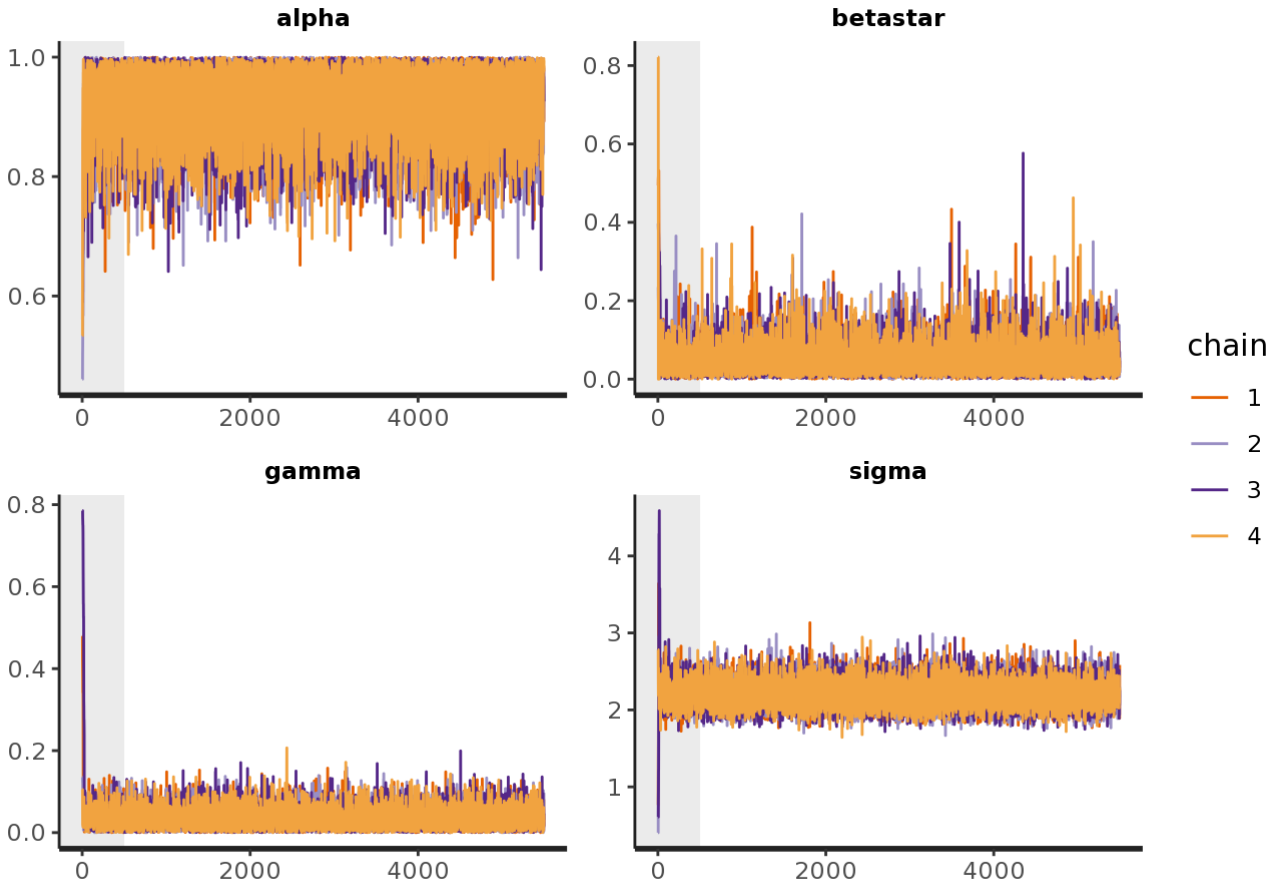
hopefully fit AAA model w/ stan

```
# fit
standata <- list(y = yts, N = N, m = m, Ns = Ns, l0 = l0, b0 = b0, s0 = s0)

rstanfit <- stan(file = "npc_AAA.stan", data = standata, warmup = 500, iter = 5500,
    cores = 2)
```

look at fit params/convergence

look at traces

```
plot(rstanfit, plotfun='trace', pars = c('alpha','betastar','gamma','sigma'), inc_warmup
=TRUE)
```

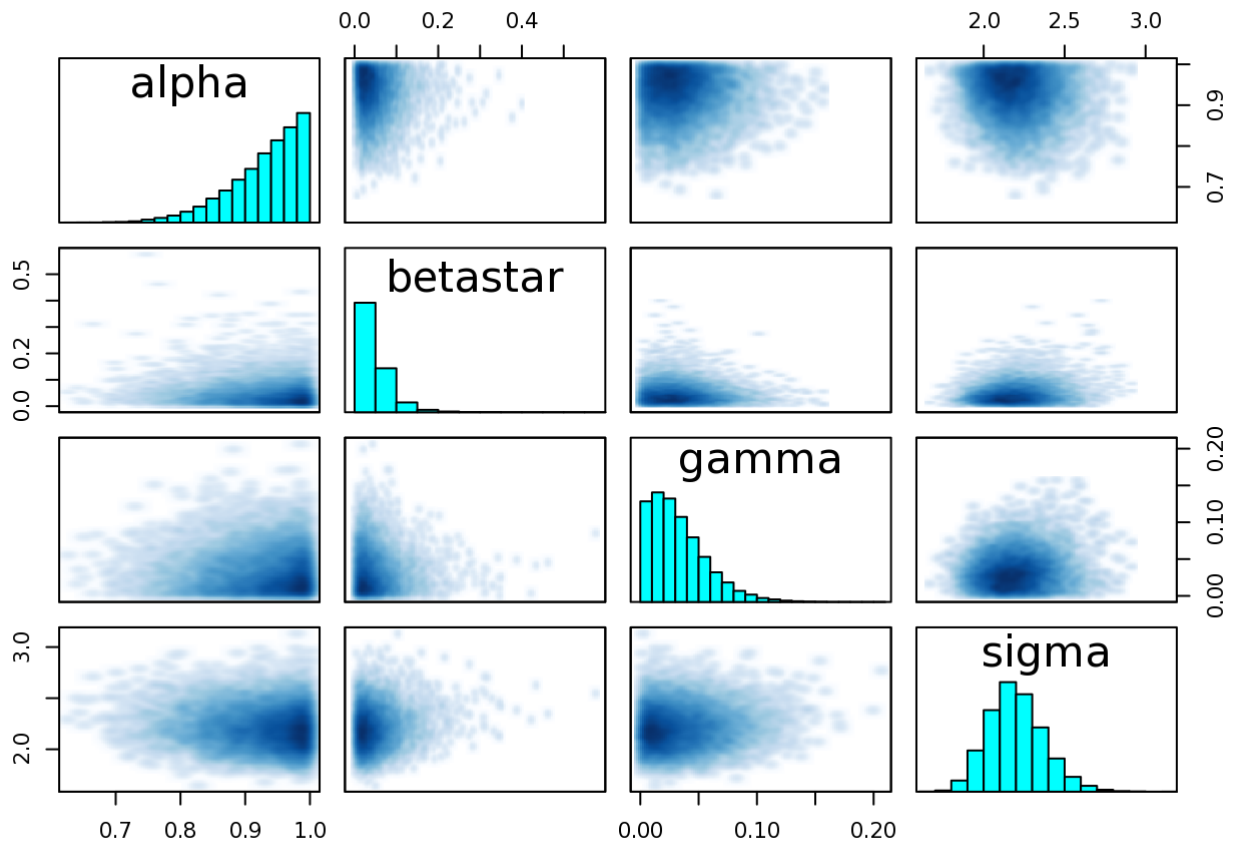



```
#plot(rstanfit3,plotfun='trace',pars=c('sigma'),inc_warmup=TRUE)
```

```
stansum <- summary(rstanfit)$summary
#stansum2 <- summary(rstanfit2)$summary
#stansum3 <- summary(rstanfit3)$summary

#stan mean params
alpha <- stansum['alpha','mean']
betastar <- stansum['betastar','mean']
gamma <- stansum['gamma','mean']
sigma <- stansum['sigma','mean']


#stansum
#sapply(get_sampler_params(stanfit,inc_warmup=0), function(x) mean(x[,'accept_stat__']))
```

look at parameter contours

```
pairs(rstanfit, pars=c('alpha','betastar','gamma','sigma'))
```




```
#pairs(rstanfit2, pars=c('alpha','betastar','gamma','sigma'))
#plot(rstanfit3,plotfun='hist',pars=c('sigma'),inc_warmup=TRUE)
```

compare parameters from stan and ets

```
print(c(alpha,alphaets))
```

```
## [1] 0.9297209 0.9739000
```

```
print(c(betastar,betaets/alphaets))
```

```
## [1] 0.0462258832 0.0001026799
```

```
print(c(gamma,gammaets))
```

```
## [1] 0.03273468 0.00010000
```

```
print(c(sigma,sigets))
```

```
## [1] 2.199513 2.214800
```

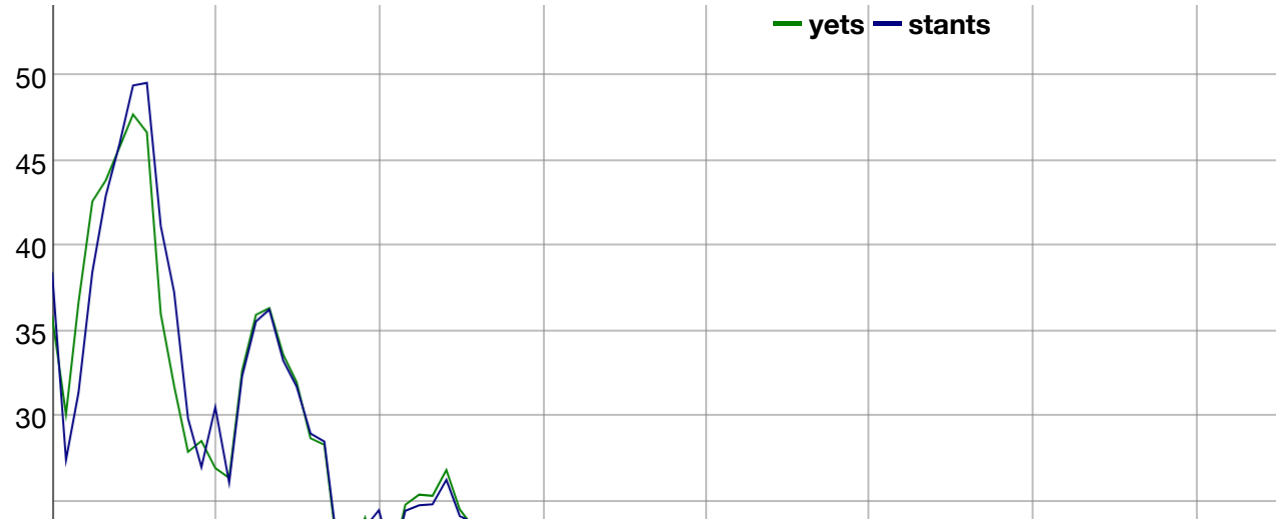
```
#alpha and sigma are similar, not betastar and gamma
```

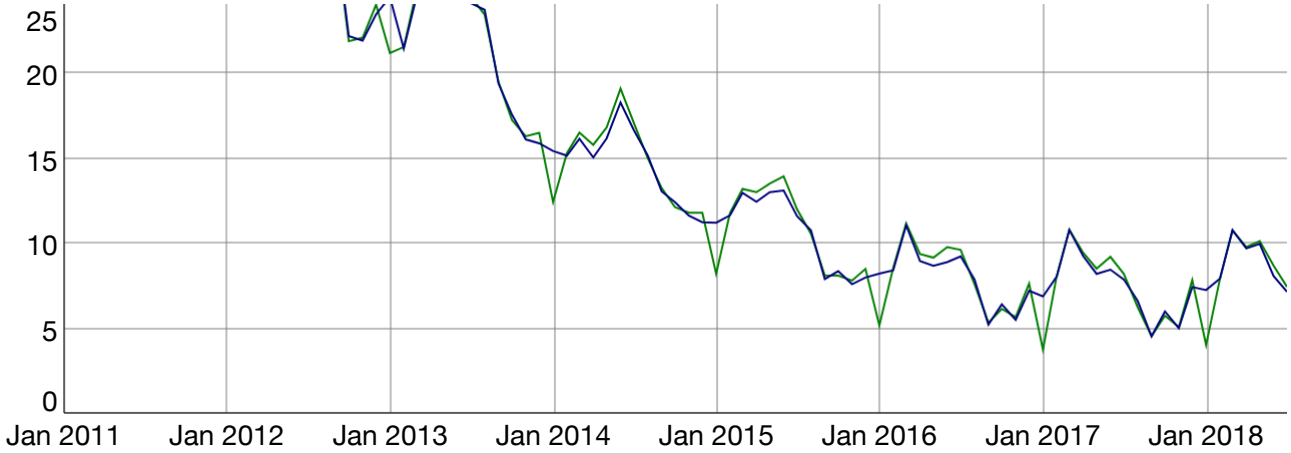
plot best fitting model

```
ystan <- vector()

ystan[1] <- l1ets+b1ets+s1ets[1]

for (i in 2:12) {
  lprev <- stansum[sprintf('l[%i]',i-1),'mean']
  bprev <- stansum[sprintf('b[%i]',i-1),'mean']
  sprev <- s1ets[i]
  ystan[i] <- lprev+bprev+sprev
}

for (i in 13:N) {
  lprev <- stansum[sprintf('l[%i]',i-1),'mean']
  bprev <- stansum[sprintf('b[%i]',i-1),'mean']
  sprev <- stansum[sprintf('s[%i]',i-m),'mean']
  ystan[i] <- lprev+bprev+sprev
}
```

plot npc time series, and ets and stan models

```
stants <- ts(ystan, start=c(2011,1), frequency=12)

#print(RMSE(yts,stants))
#print(RMSE(yts,yets))

dygraph(cbind(yets,stants))
```

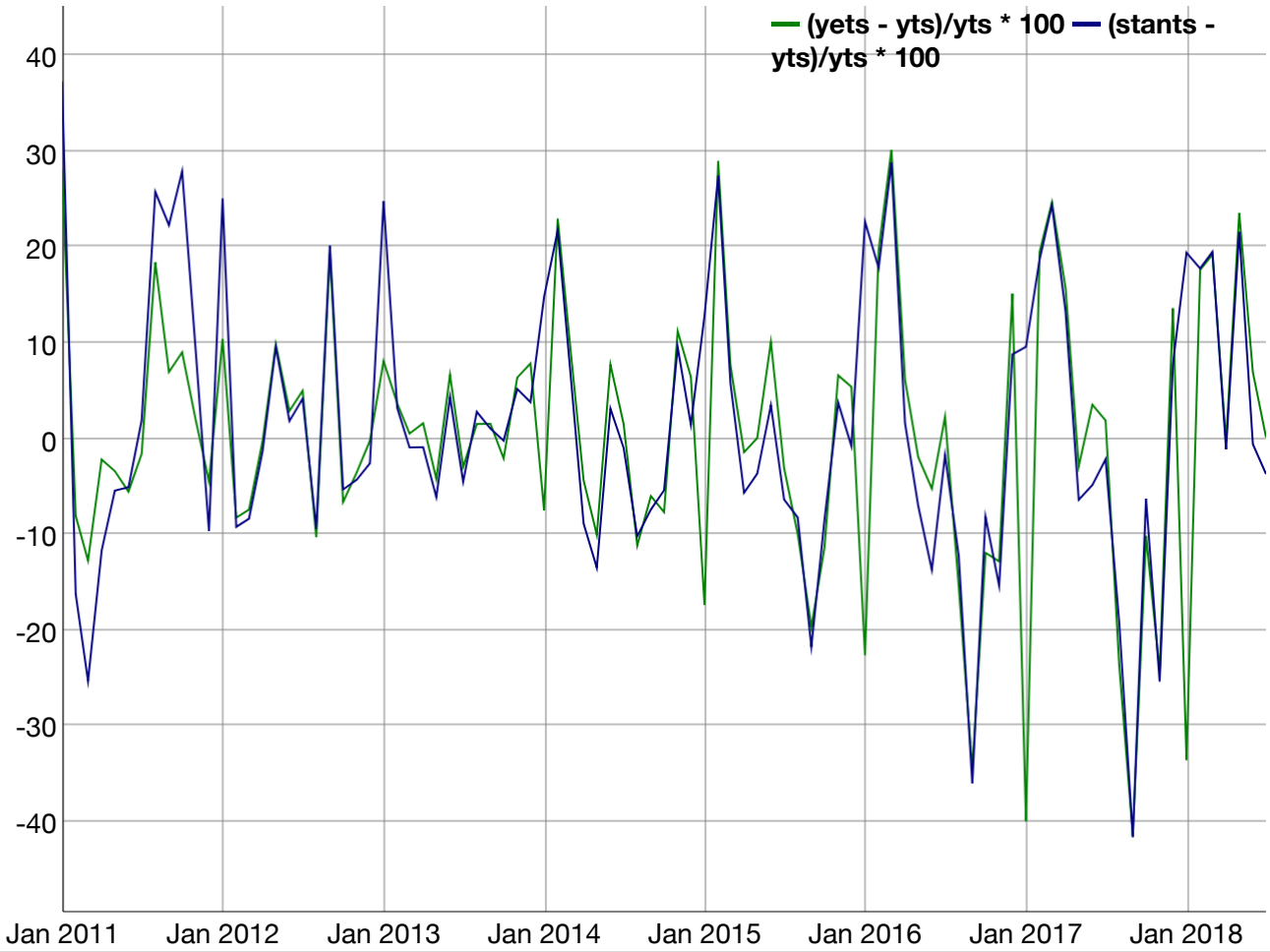





```
#looks very similar to yets
```

look at percent difference between the data and ets/my code

```
dygraph(cbind((yets-yts)/yts*100, (stants-yts)/yts*100))
```




```
#my smoothing isn't too different
```

compare my initialization and ets's

```
print(c(l0,l1ets))
```

```
## [1] 37.31478 41.03360
```

```
print(c(b0,b1ets))
```

```
## [1]  0.1924154 -0.3068000
```

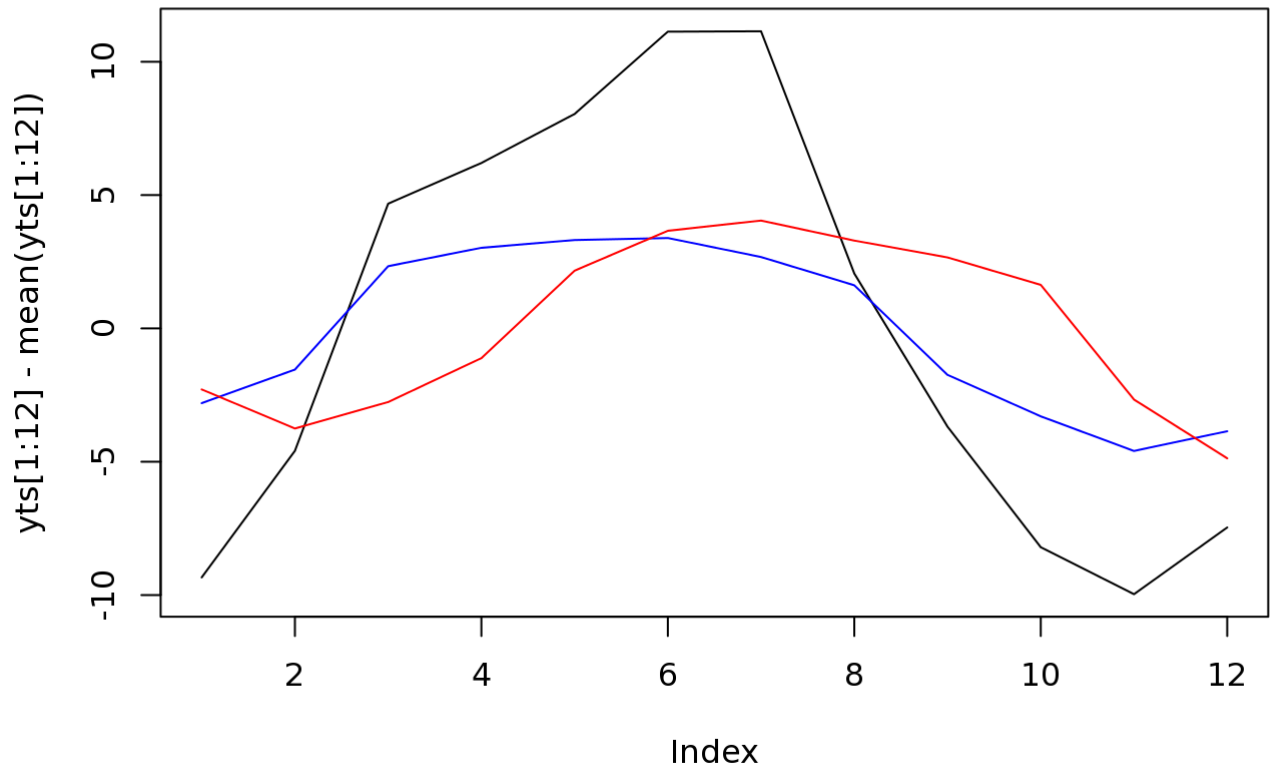
```
print(s0)
```

```
##  [1] -2.803495 -1.544102  2.332593  3.022509  3.315665  3.391085  2.677251
##  [8]  1.617307 -1.742550 -3.297898 -4.596016 -3.856850
```

```
print(s1ets)
```

```
##  [1] -2.2885 -3.7537 -2.7609 -1.1166  2.1649  3.6611  4.0441  3.2941
##  [9]  2.6625  1.6332 -2.6673 -4.8729
```

```
plot(yts[1:12]-mean(yts[1:12]),type='l')
lines(s0,col='blue')
lines(s1ets,col='red')
```



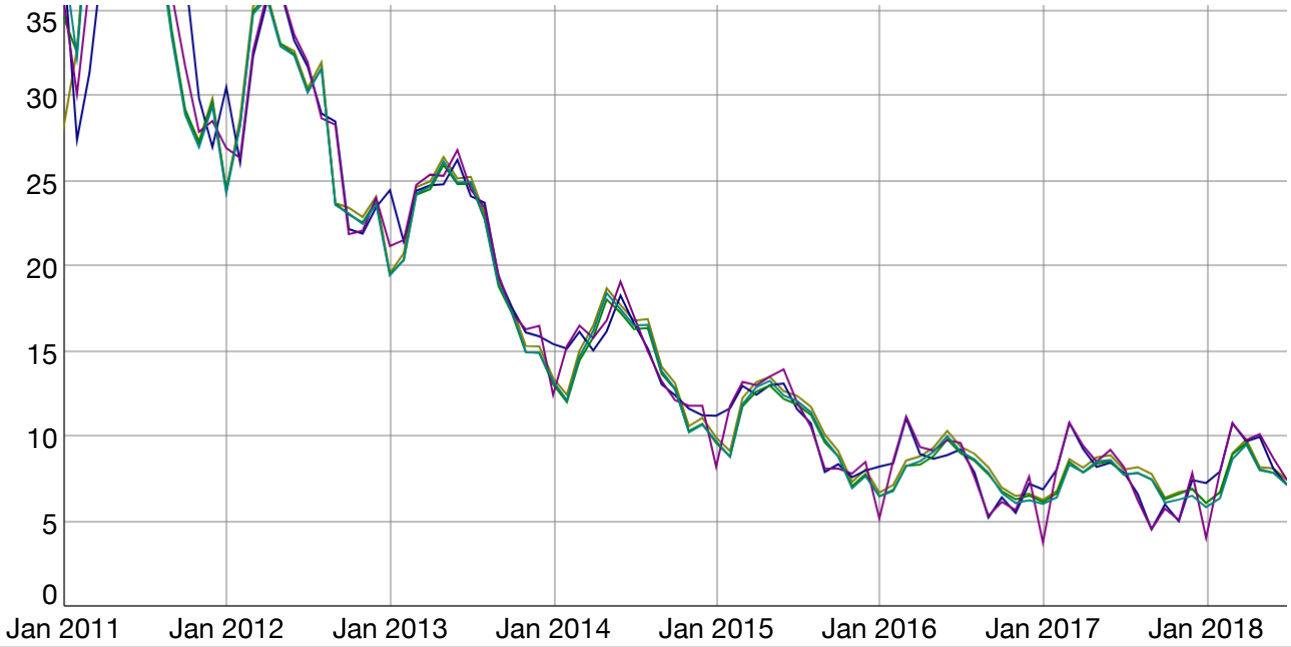

```
#our initial points are quite different, but their seasonality looks weird, right?
```

make function to smooth data, if components=TRUE, will return components instead of model

```
modelAAA <- function(y,alpha,betastar,gamma,l0,b0,s0,components=FALSE) {
  m <- 12
  N <- length(y)
  Ns <-  as.integer(N/m+1)

  smoothy <- vector()
  l <- vector()
  b <- vector()
  s <- vector()

  smoothy[1] <- l0+b0+s0[1]
  l[1] <-  alpha*(y[1]-s0[1]) + (1-alpha)*(l0+b0)
  b[1] <-  betastar*(l[1]-l0) + (1-betastar)*b0
  s[1] <-  gamma*(y[1]-l0-b0) + (1-gamma)*s0[1]

  for (t in 2:12) {
    l[t] <- alpha*(y[t]-s0[t]) + (1-alpha)*(l[t-1] + b[t-1])
    b[t] <- betastar*(l[t]-l[t-1]) + (1-betastar)*b[t-1]
    s[t] <- gamma*(y[t]-l[t-1]-b[t-1]) + (1-gamma)*s0[t]
    smoothy[t] <- l[t]+b[t]+s0[t]
  }

  for (t in 13:N) {
    l[t] <- alpha*(y[t]-s[t-m]) + (1-alpha)*(l[t-1] + b[t-1])
    b[t] <- betastar*(l[t]-l[t-1]) + (1-betastar)*b[t-1]
    s[t] <- gamma*(y[t]-l[t-1]-b[t-1]) + (1-gamma)*s[t-m]
    smoothy[t] <- l[t]+b[t]+s[t-m]
  }
  if (!components) {return(smoothy)}
  if (components) {return(list(l=l,b=b,s=s))}

}

RMSE <- function(actual, predicted){
  sqrt(mean((actual - predicted)^2))
}
```

test out function

```
stanaaa <- modelAAA(yts,alpha,betastar,gamma,l0,b0,s0)
etsaaa <- modelAAA(yts,alphaets,betaets/alphaets,gammaets,l1ets,b1ets,s1ets)
dygraph(cbind(yts,stants,stanaaa,yets,etsaaa))
```




```
rmsestan <- RMSE(yts,stanaaa)
rmseets <- RMSE(yts,etsaaa)
print(c(rmsestan,rmseets))
```

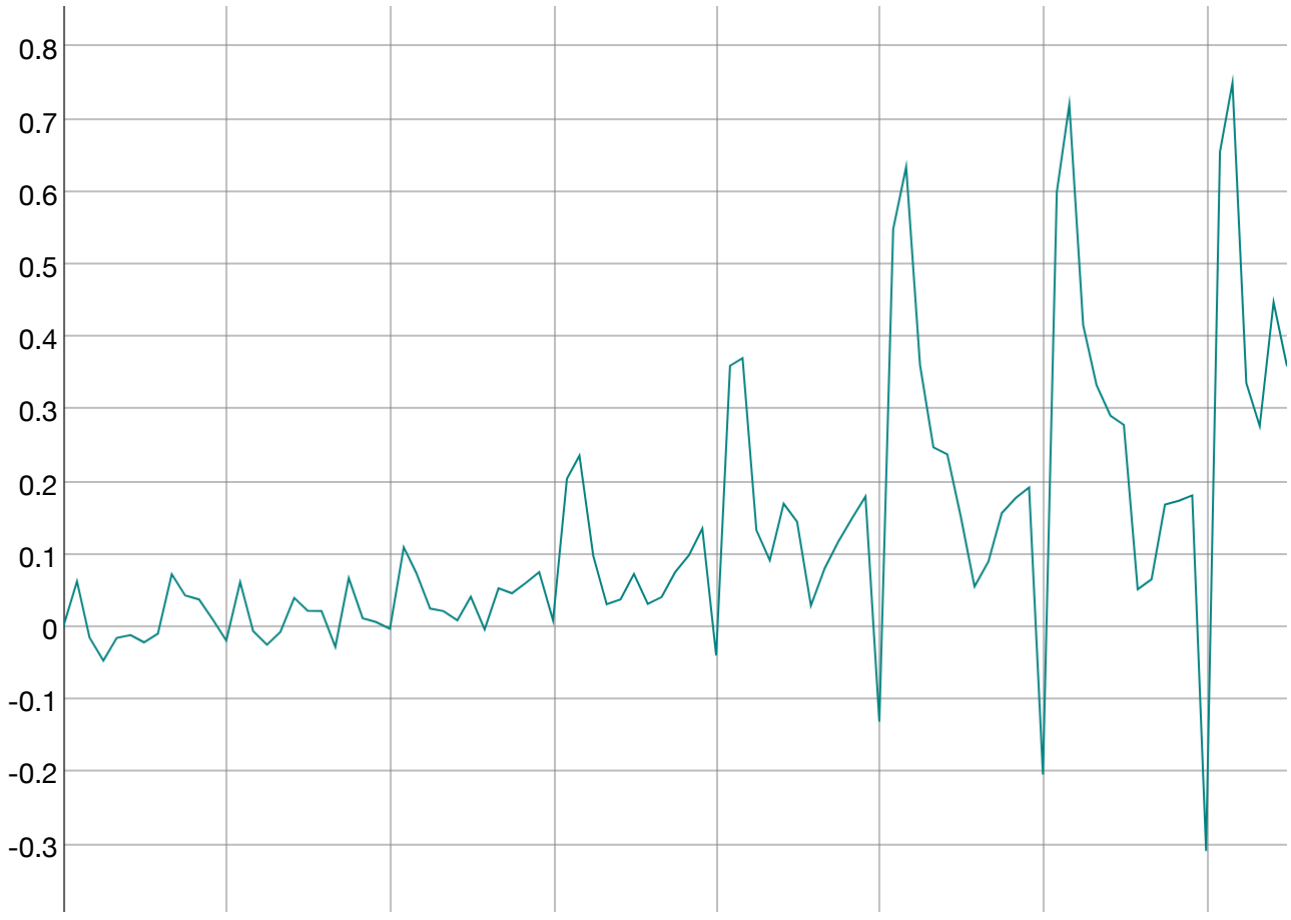
```
## [1] 0.7864064 1.1402944
```

```
#so i think difference is that what I pull from ets/stan is including best fitting erro
r, while the lines that look like yts are just the smoothed lines?
```

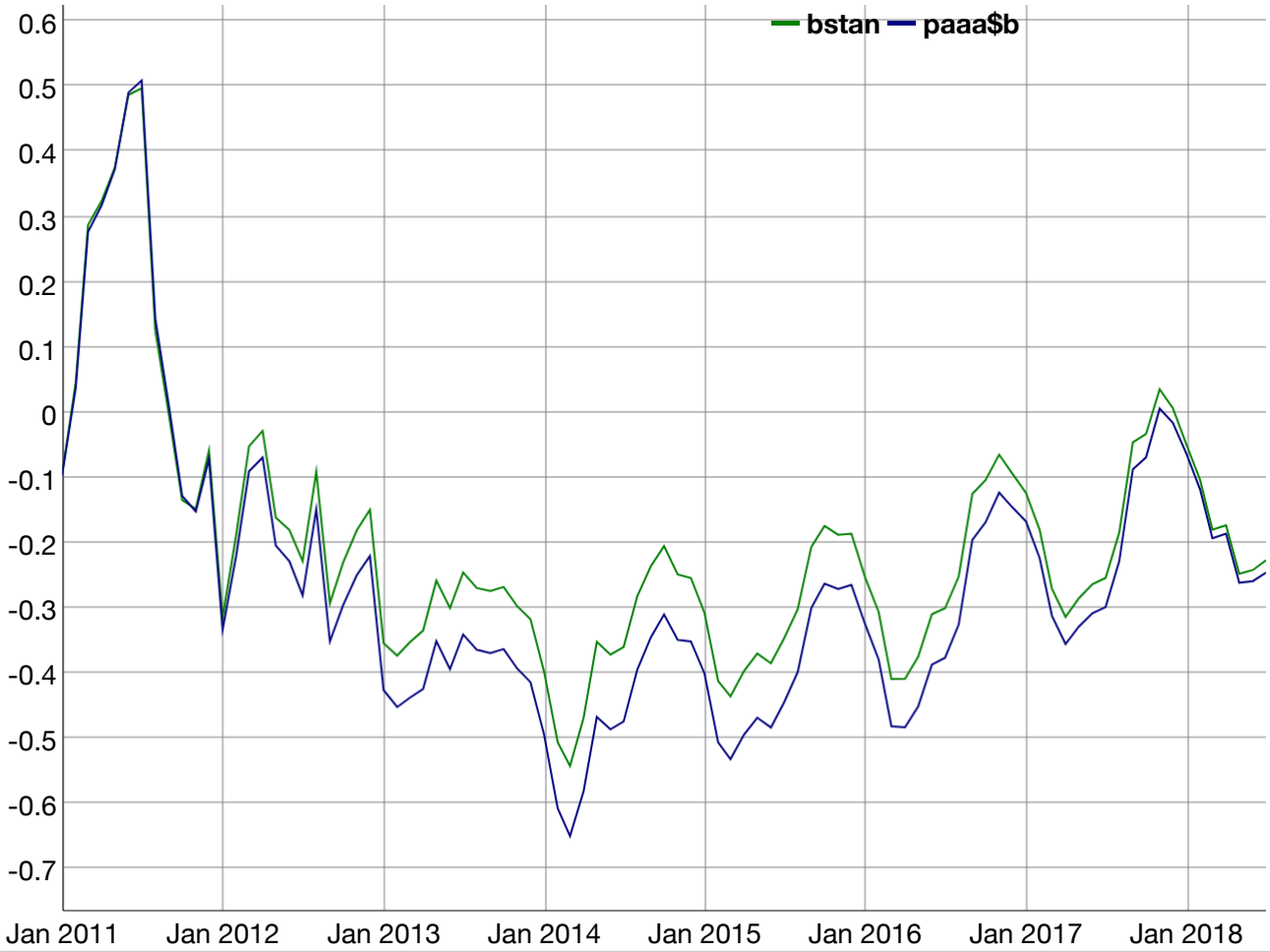
Compare the smoothed data from my function, and from stan's output

```
paaa <- modelAAA(yts,alpha,betastar,gamma,l0,b0,s0,components=TRUE)

lstan <- vector()
bstan <- vector()
sstan <- vector()

for (t in 1:N) {
  lstan[t] <- stansum[sprintf('l[%i]',t),'mean']
  bstan[t] <- stansum[sprintf('b[%i]',t),'mean']
  sstan[t] <- stansum[sprintf('s[%i]',t),'mean']
}


lstan <- ts(lstan, start=c(2011,1), frequency=12)
bstan <- ts(bstan, start=c(2011,1), frequency=12)
sstan <- ts(sstan, start=c(2011,1), frequency=12)

#they look the same
dygraph(cbind(lstan,paaa$l))
```

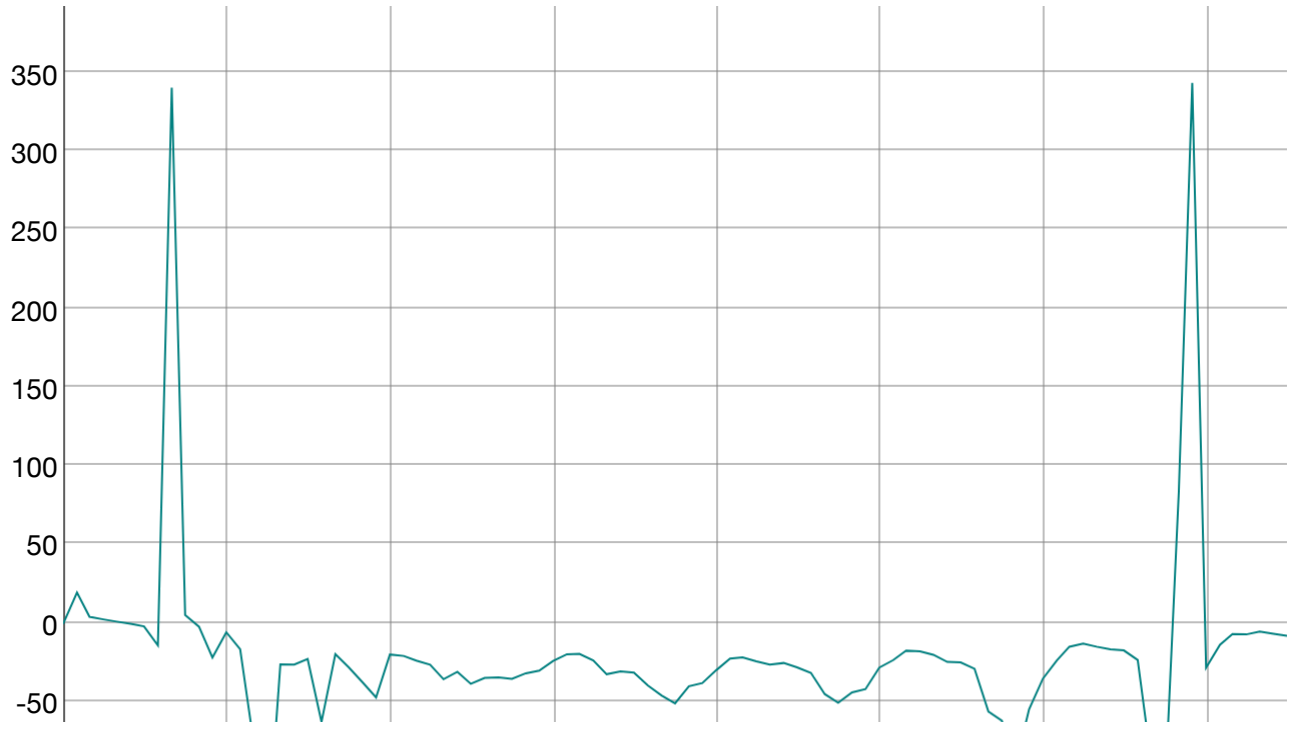

```
dygraph((lstan-paaa$l)/lstan*100)
```

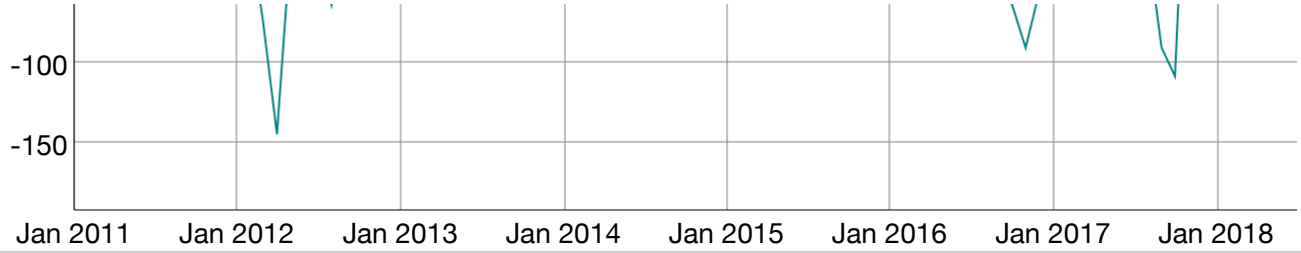



```
#pretty different... why?
dygraph(cbind(bstan,paaa$b))
```

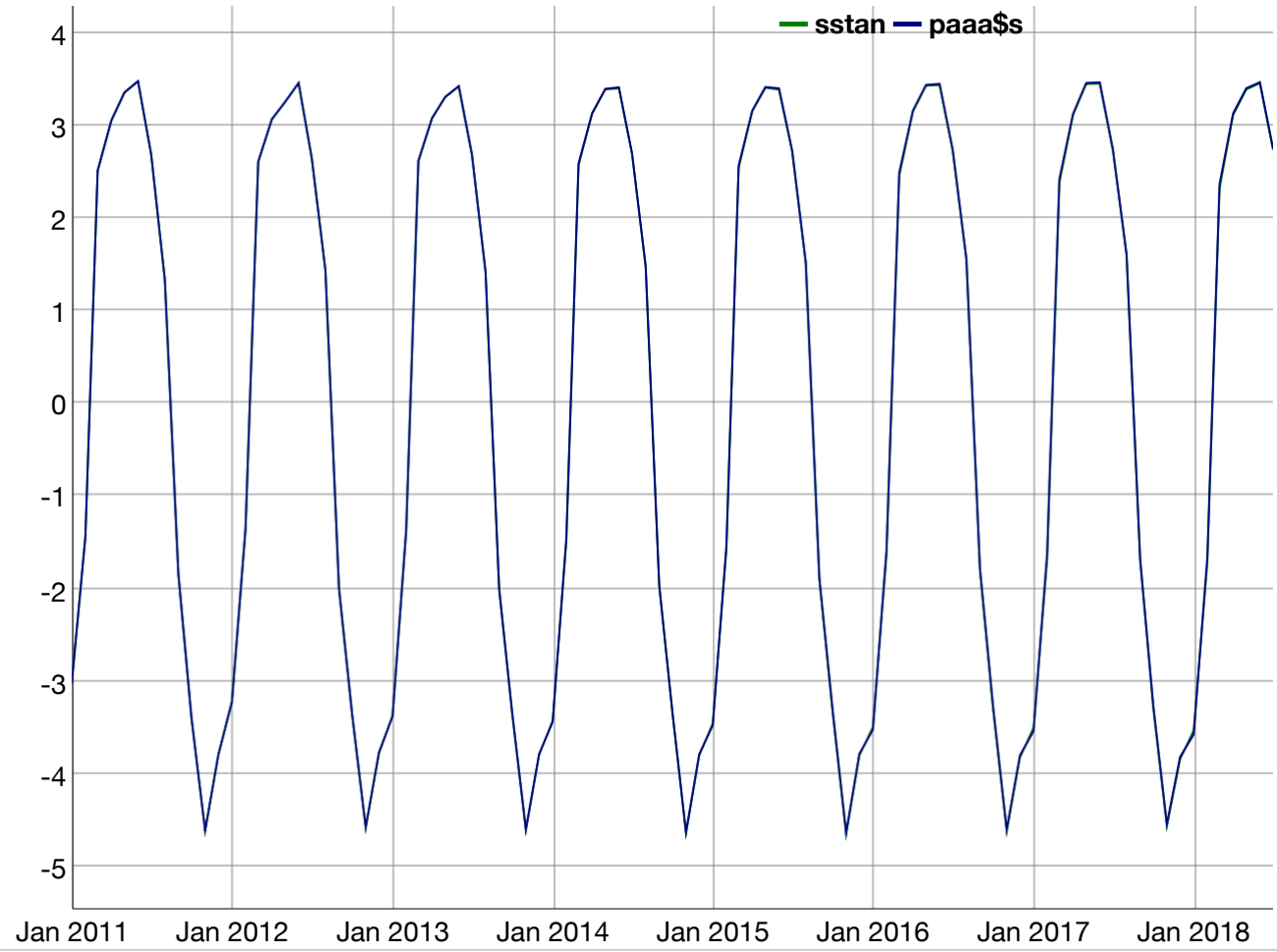



```
dygraph((bstan-paaa$b)/bstan*100)
```

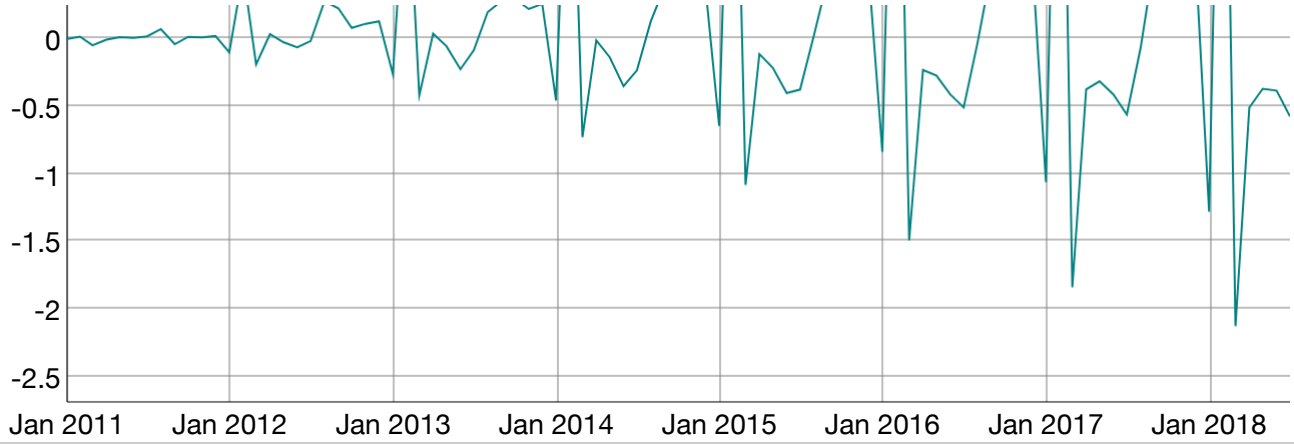





```
#pretty similar
dygraph(cbind(sstan,paaa$s))
```




```
dygraph((sstan-paaa$s)/sstan*100)
```



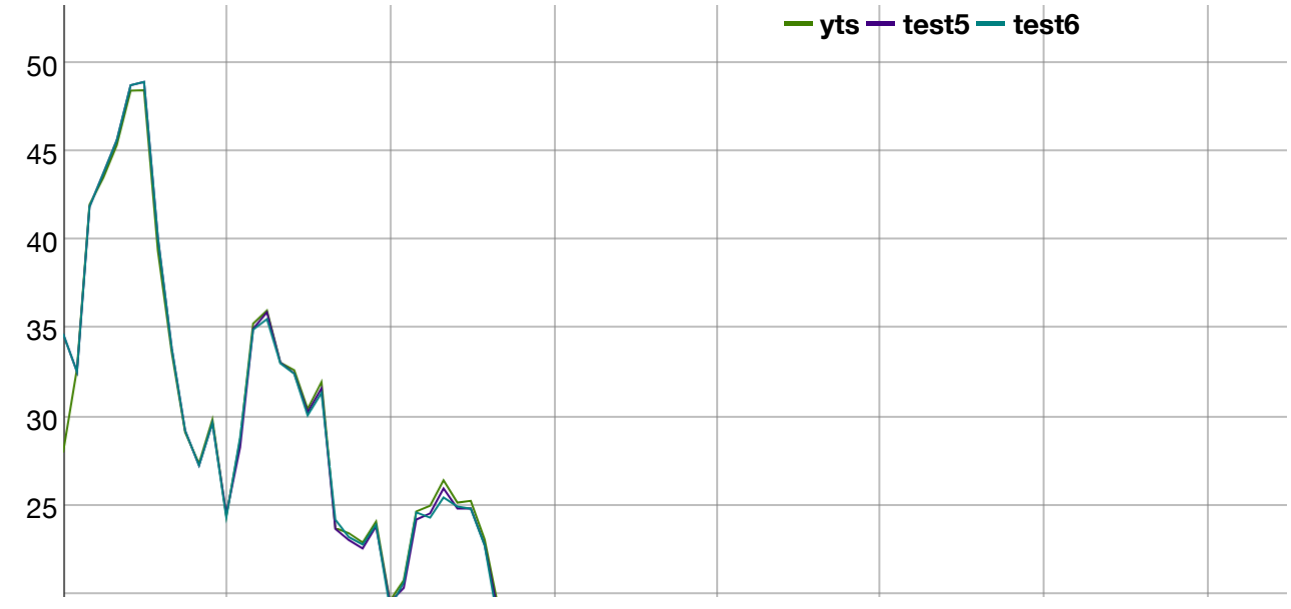

```
#....so what is the difference in the two methods?
```

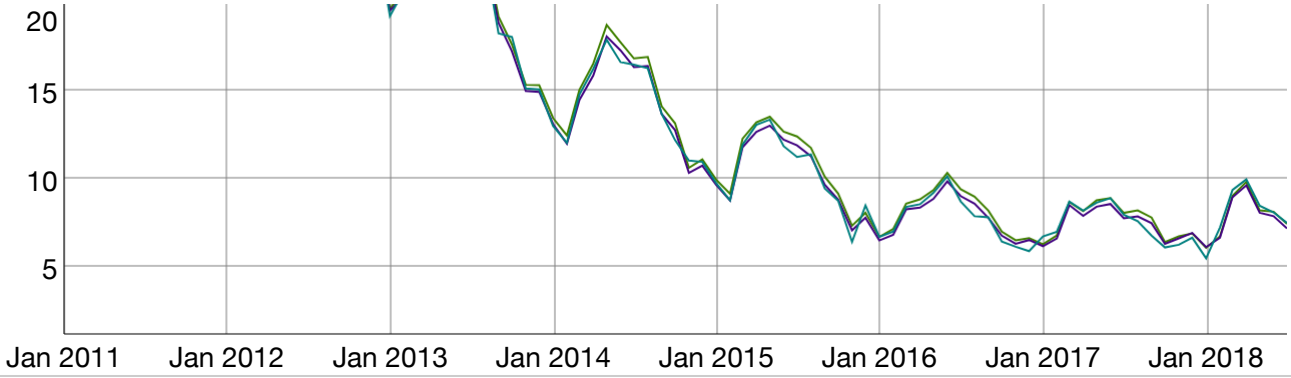
Check out how varying smoothing parameters changes my coded up model

```
#min/max of alpha
test1 <- modelAAA(yts,0.,betastar,gamma,l0,b0,s0)
test2 <- modelAAA(yts,1.,betastar,gamma,l0,b0,s0)
dygraph(cbind(yts,test1,test2))
```

```
#min/max of betastar
test3 <- modelAAA(yts,alpha,0.,gamma,l0,b0,s0)
test4 <- modelAAA(yts,alpha,1.,gamma,l0,b0,s0)
dygraph(cbind(yts,test3,test4))
```


```
#min/max of gamma
test5 <- modelAAA(yts,alpha,betastar,0.,l0,b0,s0)
test6 <- modelAAA(yts,alpha,betastar,1.,l0,b0,s0)
dygraph(cbind(yts,test5,test6))
```

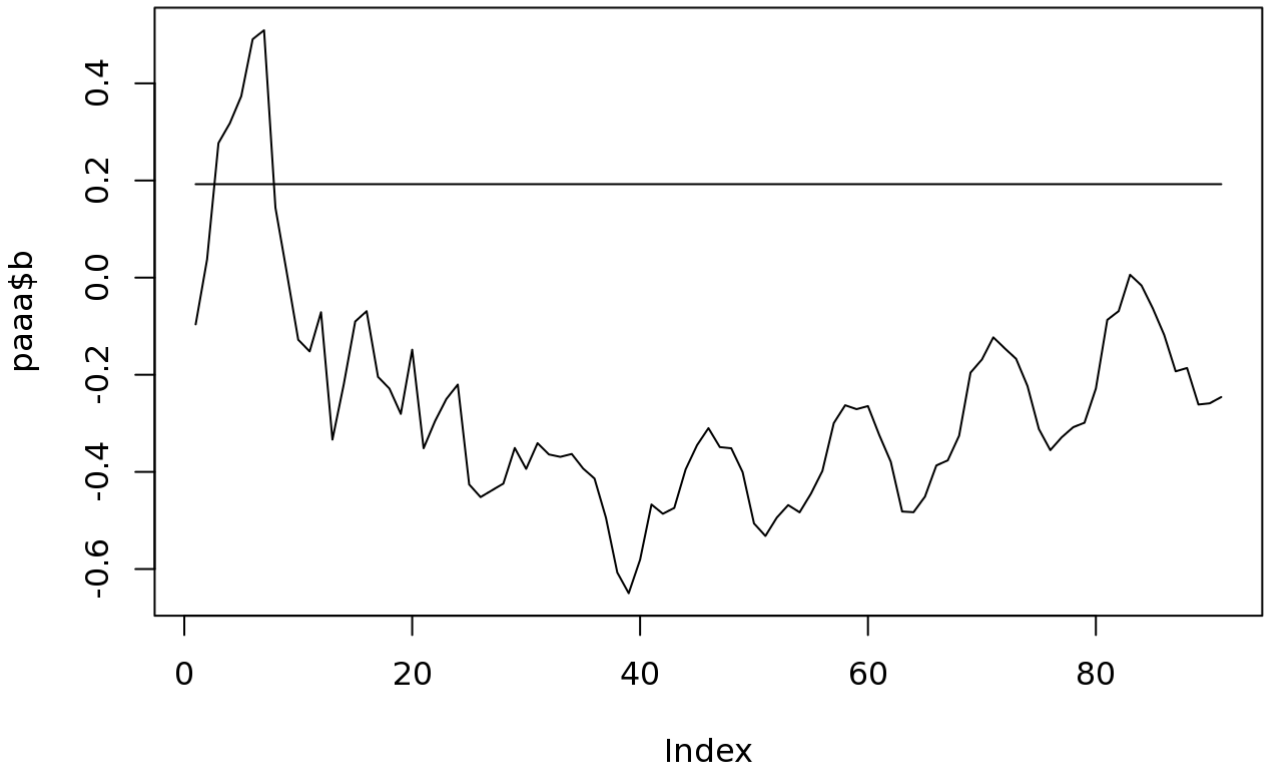




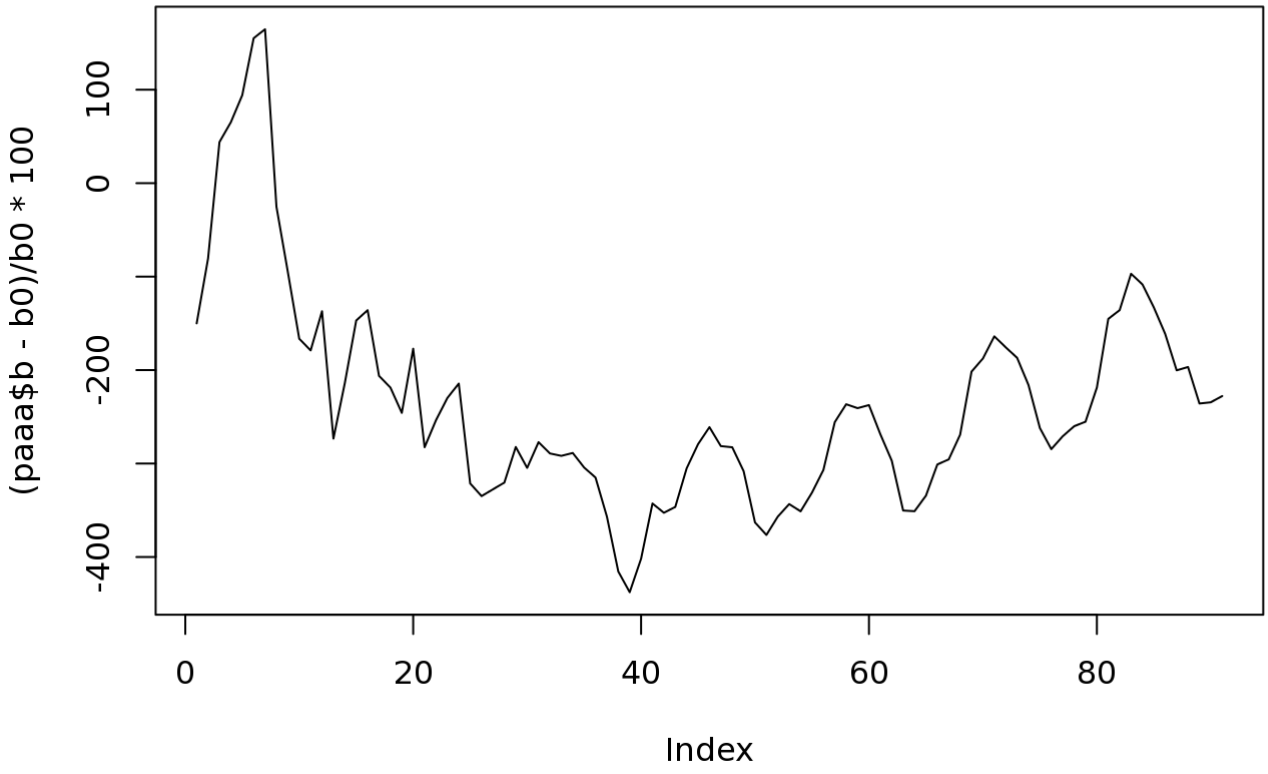

```
##looks like something with trend is broken
```

could it be that level is changing a lot, but trend and seasonality don't change from original points too much?

```
plot(paaa$b,type='l')
lines(rep(b0,N))
```




```
plot((paaa$b-b0)/b0*100,type='l')
```




```
#still some seasonality in there.... wonder if the problem is that additive seasonality
 isn't a good model
```