Kalman filter functions

Andreas C Charitos[andch552]

10/16/2019

Contents

Kalman filter with astsa package

```
# --------------------------------------------------------------------------
library(astsa)
## script given for the lab ----------------------------------------------------
# generate data
set.seed(12345); num=50
w=rnorm(num + 1, 0, 1); v=rnorm( num, 0, 1)
mu=cumsum(w) # state : mu [ 0 ] , mu [ 1 ] ,... , mu [ 5 0 ]
y = mu[-1] + v # obs : y [ 1 ] ,... , y [ 5 0 ]
# filter and smooth ( Ksmooth 0 does both )
ks=Ksmooth0(num, y, A = 1, mu0 = 0, Sigma0 = 1, Phi = 1, cQ =1, cR = 1)
moving_avg_smooth=filter(y,rep(0.2, 5),sides = 1)
# start figure
par(mfrow = c( 2, 2)); Time = 1:num
plot(Time, mu[-1], main = 'Predict', ylim = c(-5, 10))
lines(Time ,y , col = 'green' )
lines(ks$xp)
lines(ks$xp + 2 * sqrt (ks$Pp), lty = 2, col = 4)
lines(ks$xp - 2 * sqrt (ks$Pp), lty = 2, col = 4)
plot(Time, mu[-1], main = 'Filter', ylim = c(-5, 10))
lines(Time ,y , col = 'green')
lines(ks$xf)
lines(ks$xf + 2 * sqrt (ks$Pf) ,lty = 2, col = 4)
lines(ks $ xf - 2 * sqrt ( ks $ Pf ) , lty = 2 , col = 4 )
plot(Time,moving_avg_smooth,main="Moving Average Smoother",col="black")
lines(Time,moving_avg_smooth,col="green")
plot(Time, mu[-1] , main = 'Smooth', ylim = c (-5 ,10))
lines(Time, y, col = 'green')
lines(ks$xs)
lines(ks$xs + 2 * sqrt(ks$Ps), lty = 2, col = 4)
lines(ks$xs - 2 * sqrt(ks$Ps), lty = 2, col = 4)
mu[1]; ks$x0n; sqrt(ks$P0n) # initial value info
```

Kalman filter implementation

```r
# Kalmar Filter -----------------------------------------------------------

kalman_func<-function(A,B,C,Q,R,m0,P0,xt,ut){
  n=length(xt)
  mt=double(n+1) ; mt[1]=m0
  Pt=double(n+1) ; Pt[01]=P0
  #R=chol(R) ; C=chol(C)

  for(t in 1:length(xt)) {
    # odxervation update step
    Kt=Pt[(t)]*t(C)*(1/(C*Pt[(t)]*t(C)+R))
    # print(Kt)
    mt[t]=mt[(t)]+Kt*(xt[t]-C*mt[(t)])
    #
    Pt[t]=Pt[(t)]-Kt*C*Pt[(t)]
    # prediction step
    mt[(t+1)]=A*mt[t]
    Pt[(t+1)]=A*Pt[t]*t(A)+Q
  }
  return(list(predicttions=mt[-(n+1)],Pt=Pt[-(n+1)]))
}

k=kalman_func( A=1,B=1, C=1, Q=0.1,R=10, m0=1,P0=1,y )

# plot(y,type="l")
# lines(k$predicttions)

plot(Time, mu[-1], main = 'Predict', ylim = c(-5, 10))
lines(Time ,y , col = 'green' )
lines(k$predicttions,col="red")
```

Kalman filter vectorized

```r
# Vectorized  ------------------------------------------------------------

# generate data
set.seed(12345); num=50
w=rnorm(num + 1, 0, 1); v=rnorm( num, 0, 1)
mu=cumsum(w) # state : mu [ 0 ] , mu [ 1 ] ,... , mu [ 5 0 ]
y = mu[-1] + v # obs : y [ 1 ] ,... , y [ 5 0 ]
# filter and smooth ( Ksmooth 0 does both )

kalman_func_vect<-function(A,C,Q,R,m0,P0,xt,verbose){
  #Q=chol(Q,tol=-1) ; R=chol(R,tol = -1) # cholesky decomposition of the covariance matrices
  d1=dim(P0)[1] ; d2=dim(P0)[2] ; Tt=length(xt)
  d3=length(m0)
  mt_t=matrix(NA,nrow=Tt+1,ncol=d3)
  mt_t[1,]=m0
  Pt_t=array(NA,dim=c(d1,d2,Tt+1)) #; print(dim(Pt_t))
  Pt_t[,,1]=P0
  for(t in 1:(Tt)){

    if(verbose==1){
      cat("---iteration: ",t,"\n")
    }
    # odxervation update step
    Kt=Pt_t[,,(t)]%*%t(C)%*%solve(C%*%Pt_t[,,(t)]%*%t(C)+R)
    #
    mt_t[t,]=mt_t[(t),]+Kt%*%(xt[(t)]-C%*%mt_t[(t),])
    #
    Pt_t[,,t]=(diag(d1)-Kt%*%C)%*%Pt_t[,,(t)]
    # prediction step
    mt_t[(t+1),]=A%*%mt_t[t,]
    #
    Pt_t[,,(t+1)]=A%*%Pt_t[,,t]%*%t(A)+Q
    #
  }
  return(list(mt=mt_t,Pt=Pt_t))
}

E=diag(2)*1 ; E

k=kalman_func_vect( A=E, C=E,
                    Q=E*1,R=E*1, m0=c(1,3),P0=E,y,verbose=0)
Time = 1:50
plot(Time, mu[-1], main = 'Predict', ylim = c(-5, 10))
ks=Ksmooth0(num, y, A = 1, mu0 = 0, Sigma0 = 1, Phi = 1, cQ = 1, cR = 1)
lines(Time ,y , col = 'green' )
lines(k$mt[,1],col="red")
U=k$mt[-51,2]+2*sqrt(abs(k$Pt[1,1,50]))
L=k$mt[-51,2]-2*sqrt(abs(k$Pt[1,1,50]))
lines(U,col="blue",lty=2)
lines(L,col="blue",lty=2)
#lines(ks$xf + 2 * sqrt (ks$Pf) ,lty = 2, col = 4) # the same bands with the Ksmooth
```

```
#lines(ks $ xf - 2 * sqrt ( ks$Pf ) , lty = 2 , col = 4 ) # same bands with the Ksmooth
legend("bottomright",legend = c("states","obs","kalman preds","bands"),
       col=c("black","green","red","blue"),pch=c("o",NA,NA,NA),
       lty=c(NA,1,1,2))
```