732A62 Time Series Analysis

Computer Lab C

Andreas C Charitos (andch552),Ruben Muñoz (rubmu773)

2019-10-10

Contents

1 Instructions

- The lab is assumed to be done in groups.
- Create a report to the lab solutions in PDF.
- Be concise and do not include unnecessary printouts and figures produced by the software and not required in the assignments.
- **Include all your codes as an appendix into your report.**
- A typical lab report should 2-4 pages of text plus some number of figures plus appendix with codes.
- The group lab report should be submitted via LISAM before the deadline specified in LISAM.
- Use 12345 as a random seed everywhere where the result of the simulation differs with the run unless stated otherwise.

2 Implementation of Kalman filter

2.1 Assignment 1

In table 1 a script for generation of data from simulation o fthe following state space model and implementation of the Kalman filter on the data is given.

$$z_t = A_{t-1}z_{t-1} + e_t, \quad e_t \sim \mathcal{N}(0, Q_t).$$
$$x_t = C_t z_t + v_t, \quad v_t \sim \mathcal{N}(0, R_t)$$

2.1.1 a)

2.1.1.1 Instructions.

Write down the expression for the state space model that is being simulated.

2.1.1.2 Results.

According to the simulation values we have $A_t = 1, C_t = 1, R_t = 1, Q_t = 1$ so the state space model is given by the following expression:

$$z_t = z_{t-1} + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, 1)$$
$$x_t = z_t + v_t, \quad v_t \sim \mathcal{N}(0, 1)$$

2.1.2 b)

2.1.2.1 Instructions.

Run this script and compare the filtering results with a moving average smoother of order 5.

2.1.2.2 Results.

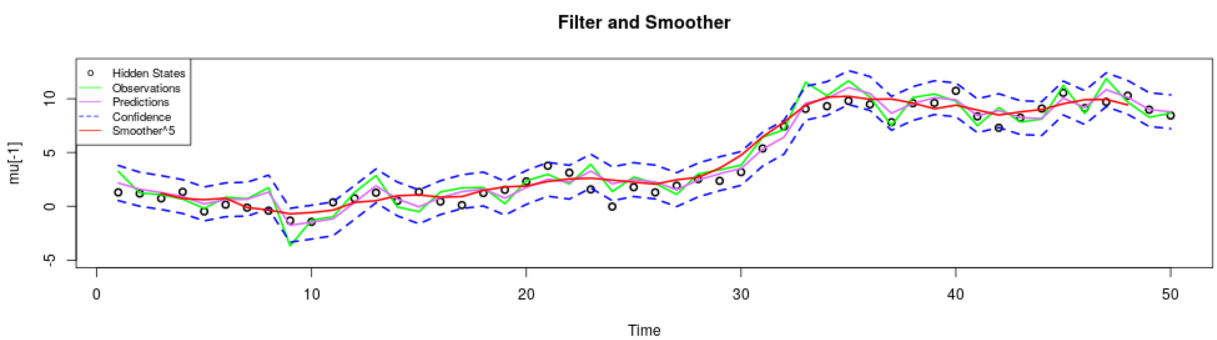

Figure 1: *Visualization of the performance of the filter and a simple moving average.*

As we can see from the plot the Kalman filter approximates the hidden states pattern quite well.The moving average amoother also manages to capture the overall trend of the signal quite weell but as it was expected is much smoother than kalman filter.

2.1.3 c)

2.1.3.1 Instructions.

Also, compare the filtering outcome when R in the filter is 10 times smaller than its actual value while Q in the filter is 10 times larger than its actual value. How does the filtering outcome varies?

2.1.3.2 Results.

The Q and R act as "knobs" for the white noise that might be a inherit part of the model generating the states z_t and x_t respectively. When we in this artificial situation increase Q, this can be translated to the following statement "the error/variance" of the true model generating the states z_t is far from perfect and thus has a greater posibility of error when trying to explain the true hiden state.

As we can see if figure 2 when Q is increased, then the previous situation is also mirrowed here, but in this case it means that the error of the observation is big. This causes in our plot for the variance in prediction lines to be very tight and thus the Karman filter model "greatly trusts" the prediction/observations, practically causing a very good filtering so as we can see in the plot the filter predictions and confidence intervals are the same as the observations.

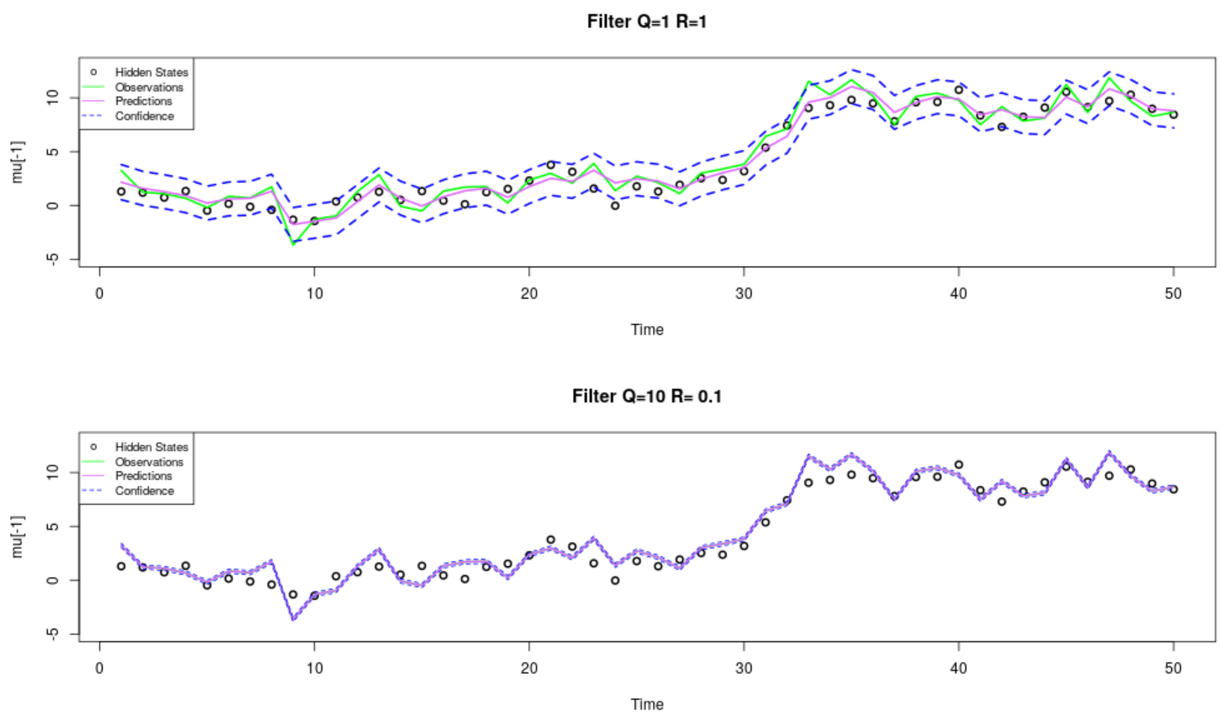

Figure 2: *Visualization of the difference in R and Q ratio in the results.*

2.1.4 d)

2.1.4.1 Instructions.

Now compare the filtering outcome when R in the filter is 10 times larger than its actual value while Q in the filter is 10 times smaller than its actual value. How does the filtering outcome varies?

2.1.4.2 Results.

As we can see in figure 3, the R is very large now and the Q very small, this is translated as, dont trust the observations because they have great error thus, thus we see that the filter is doing its job according on the parameters we have provided filtering out most of the noise that we told the observations had, but still following the general trend of the data.We can see that the filter predictions are very smooth and don't capture the states pattern.

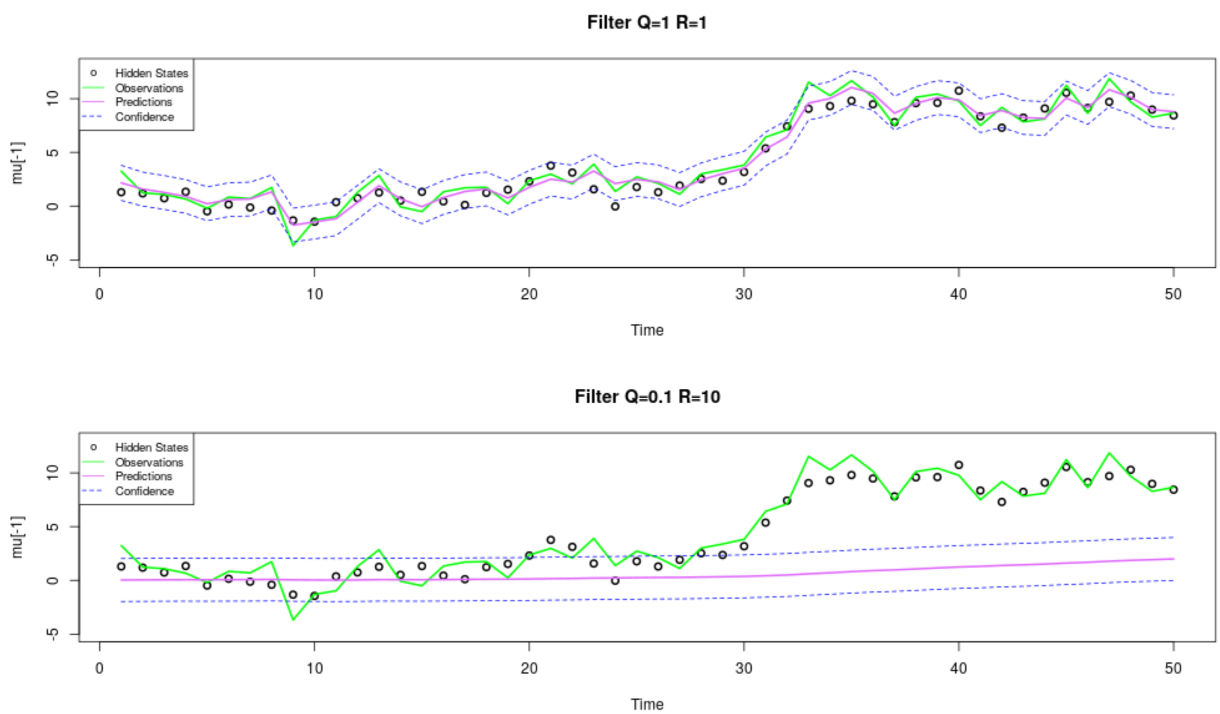

Figure 3: *Visualization of the difference in R and Q ratio in the results.*

To sum up, Q incorporates the noise between the nearby states,since it affects the change of states. Thus, all the changes among states are acceptable, so do the pattern of observations.The R incorporates the noise between state and observed point at each time.

- if Q is small, the estimated pattern of states will be change slightly.
- if Q is large, all change is acceptable, follow more the observations
- if R large, the confidence intervals of estimated states is large
- if R small, the confidence intervals of estimated staes is small

2.1.5 e)

2.1.5.1 Instructions.

Implement your own Kalman filter and replace ksmooth0 function with your script.

2.1.5.2 Results.

In the plots above we can see the results of our implementation compared with the Ksmooth function with the parameters for R,Q used in the previous questions. As we can see we where able to produce a somehow good approximation to Ksmooth function.

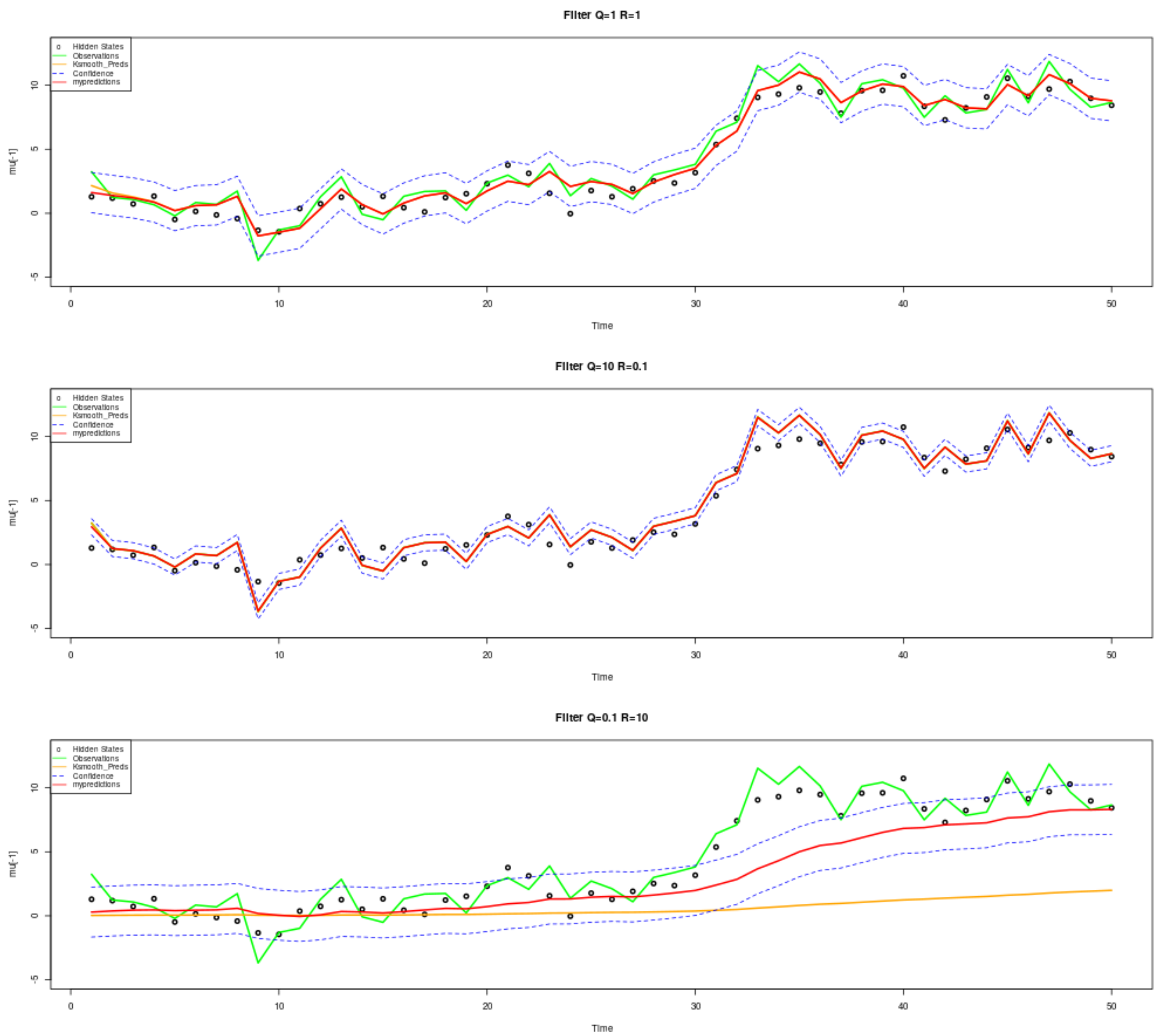

Figure 4: *Visualization of the difference in R and Q ratio in the results with mykalmanfilter().*

2.1.6 f)

2.1.6.1 Instructions.

How do you interpret the Kalman gain?

2.1.6.2 Results.

The Kalman gain is the relative weight given to the measurements and current state estimate, and can be "tuned" to achieve a particular performance. With a high gain, the filter places more weight on the most recent measurements, and thus follows them more responsively. With a low gain, the filter follows the model predictions more closely. At the extremes, a high gain close to one will result in a more jumpy estimated trajectory, while a low gain close to zero will smooth out noise but decrease the responsiveness. When performing the actual calculations for the filter (as discussed below), the state estimate and covariances are coded into matrices to handle the multiple dimensions involved in a single set of calculations. This allows for a representation of linear relationships between different state variables (such as position, velocity, and acceleration) in any of the transition models or covariances. `source Wikepedia [link]`

3 3 Code Appendix

```
library(ggplot2)
# only use when knitting tables in to png
# library(kableExtra)
color_palette <- c(rgb(38,50,72,alpha=160, max = 255),  # [1] MX robots blue
                   rgb(255,152,0,alpha=160, max = 255), # [2] MX robots orange
                   rgb(255,0,0,alpha=100, max = 255),   # [3] Red faded
                   rgb(0,0,0,alpha=160, max = 255),     # [4] black
                   rgb(10,150,10,alpha=160, max = 255), # [5] green
                   rgb(0,0,0,alpha=40, max = 255),      # [6] light Gray
                   rgb(255,0,0,alpha=255, max = 255),   # [7] Red full
                   rgb(255,152,0,alpha=200, max = 255)) # [8] MX robots orange fulish
set.seed(12345)
Title <- "732A62 Time Series Analysis"
Subtitle <- "Computer Lab C"
Author <- "Andreas C Charitos (andch552),Ruben Muñoz (rubmu773)"
Date <- Sys.Date()
Chapter01 <- "Instructions"
Chapter02 <- "Implementation of Kalman filter"
## b) ---------------------------------------------------
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 ]
Time = 1:num
# filter and smooth ( Ksmooth 0 does both )
ks=Ksmooth0(num, y, A = 1, mu0 = 0, Sigma0 = 1, Phi = 1, cQ = 1, cR = 1)
# moving average smoother for comparison
mysmoother <- function(x, n){filter(as.vector(x),rep(1 / n, n), sides = 2)} # x=vector
↪  n=order
# start figure
png(filename="images/plotA1.png", width = 1000, height = 300)
par(mfrow = c( 1, 1)); Time = 1:num
# plot(Time, mu[-1], main = 'Predict', ylim = c(-5, 10))
# lines(Time ,y , col = 'green' )
# lines(ks$xp) # one-step-ahead prediction of the state
# lines(ks$xp + 2 * sqrt (ks$Pp), lty = 2, col = 4)
# lines(ks$xp - 2 * sqrt (ks$Pp), lty = 2, col = 4)
plot(Time, mu[-1], main = 'Filter and Smoother', ylim = c(-5, 13),lwd=2)
lines(Time, y, col='green',lwd=2)
lines(ks$xf,col="mediumorchid1",lwd=2)
lines(ks$xf + 2 * sqrt (ks$Pf) ,lty = 2, col = 4,lwd=2)
lines(ks $ xf - 2 * sqrt (ks$Pf) , lty = 2 , col = 4 ,lwd=2)
lines(Time, mysmoother(y,5), col='red',lwd=2)
legend("topleft",
       legend=c("Hidden States","Observations","Predictions","Confidence","Smoother^5"),
       col=c('black','green','mediumorchid1','blue','red'), pch=c("o",NA,NA,NA,NA),
       lty=c(NA,1, 1, 2, 1), cex=0.8)
```

```r
# plot(Time, mu[-1] , main = 'Smooth', ylim = c (-5 ,10))
# lines(Time, y, col = 'green')
# lines(ks$xs) # state smoothers
# lines(ks$xs + 2 * sqrt(ks$Ps), lty = 2, col = 4)
# lines(ks$xs - 2 * sqrt(ks$Ps), lty = 2, col = 4)
# dev.off()
# mu[1]; ks$x0n; sqrt(ks$P0n) # initial value info
knitr::include_graphics(c("images/plotA1.png"))
## c) ---------------------------------------------------
# start figure
png(filename="images/plotA2.png", width = 1000, height = 600)
par(mfrow = c( 2, 1)); Time = 1:num
# plot with Q=1,R=1
ks=Ksmooth0(num, y, A = 1, mu0 = 0, Sigma0 = 1, Phi = 1, cQ = 1, cR = 1)
plot(Time, mu[-1], main = 'Filter Q=1 R=1', ylim = c(-5, 13),lwd=2)
lines(Time ,y , col = 'green',lwd=2)
lines(ks$xf,col="mediumorchid1",lwd=2)
lines(ks$xf + 2 * sqrt (ks$Pf) ,lty = 2, col = 4,lwd=2)
lines(ks $ xf - 2 * sqrt (ks$Pf) , lty = 2 , col = 4 ,lwd=2)
legend("topleft",
       legend=c("Hidden States","Observations","Predictions","Confidence"),
       col=c('black','green','mediumorchid1','blue','red'), pch=c("o",NA,NA,NA),
       lty=c(NA,1, 1, 2), cex=0.8)
# plot with R=0.1 ,R=10
ks=Ksmooth0(num, y, A = 1, mu0 = 0, Sigma0 = 1, Phi = 1, cQ = 10, cR = 0.1)
plot(Time, mu[-1], main ='Filter Q=10 R= 0.1', ylim = c(-5, 13),lwd=2)
lines(Time ,y , col = 'green',lwd=2)
lines(ks$xf,col="mediumorchid1",lwd=2)
lines(ks$xf + 2 * sqrt (ks$Pf) ,lty = 2, col = 4)
lines(ks $ xf - 2 * sqrt ( ks$Pf ) , lty = 2 , col = 4 )
legend("topleft",
       legend=c("Hidden States","Observations","Predictions","Confidence"),
       col=c('black','green','mediumorchid1','blue','red'), pch=c("o",NA,NA,NA),
       lty=c(NA,1, 1, 2), cex=0.8)
#
dev.off()
mu[1]; ks$x0n; sqrt(ks$P0n) # initial value info
knitr::include_graphics(c("images/plotA2.png"))
## d) ---------------------------------------------------
# start figure
png(filename="images/plotA3.png", width = 1000, height = 600)
par(mfrow = c( 2, 1))
# plot with Q=1, R=1
ks=Ksmooth0(num, y, A = 1, mu0 = 0, Sigma0 = 1, Phi = 1, cQ = 1, cR = 1)
plot(Time, mu[-1], main = 'Filter Q=1 R=1', ylim = c(-5, 13),lwd=2)
lines(Time ,y , col = 'green',lwd=2)
lines(ks$xf,col="mediumorchid1",lwd=2)
lines(ks$xf + 2 * sqrt (ks$Pf) ,lty = 2, col = 4)
lines(ks $ xf - 2 * sqrt (ks$Pf) , lty = 2 , col = 4 )
legend("topleft",
       legend=c("Hidden States","Observations","Predictions","Confidence"),
       col=c('black','green','mediumorchid1','blue'), pch=c("o",NA,NA,NA),
       lty=c(NA,1, 1, 2), cex=0.8)
```

```r
# plot with Q=0.1,R=10
ks=Ksmooth0(num, y, A = 1, mu0 = 0, Sigma0 = 1, Phi = 1, cQ = 0.1, cR = 10)
plot(Time, mu[-1], main = 'Filter Q=0.1 R=10', ylim = c(-5, 13),lwd=2)
lines(Time ,y , col = 'green',lwd=2)
lines(ks$xf,col="mediumorchid1",lwd=2)
lines(ks$xf + 2 * sqrt (ks$Pf) ,lty = 2, col = 4)
lines(ks $ xf - 2 * sqrt ( ks$Pf ) , lty = 2 , col = 4 )
legend("topleft",
       legend=c("Hidden States","Observations","Predictions","Confidence"),
       col=c('black','green','mediumorchid1','blue'), pch=c("o",NA,NA,NA),
       lty=c(NA,1, 1, 2), cex=0.8)
#
dev.off()
mu[1]; ks$x0n; sqrt(ks$P0n) # initial value info
knitr::include_graphics(c("images/plotA3.png"))
## e) -------------------------------------------------
kalman_filter <- function(At, Ct, Qt, Rt, m0, P0, xt) {
  # initializaction
  bigT=length(xt)
  mt=rep(0, bigT+1)
  mt[1]=m0
  Pt=rep(0, bigT+1)
  Pt[1]=P0
  for (t in 1:(bigT)) {
    Kt=Pt[t]*t(Ct)*solve(Ct*Pt[(t)]*t(Ct)+Rt)
    mt[t]=mt[(t)] + Kt*(xt[(t)] - Ct*mt[t])
    Pt[t]= (1 - Kt*Ct)*Pt[t]
    #
    mt[(t+1)]= At*mt[t]
    Pt[(t+1)]= At*Pt[t]*t(At) + Qt
  }
  return(list(mt = mt, Pt = Pt))
}
# start the plot
png(filename="images/plotA4.png", width = 1000, height = 900)
par(mfrow = c( 3, 1))
# Comparison Q and R one to one ratio
# our kalman predictions
k=kalman_filter(At = 1, Ct = 1, Qt = 1, Rt = 1, m0=0, P0=1, xt =y)
ks=Ksmooth0(num, y, A = 1, mu0 = 0, Sigma0 = 1, Phi = 1, cQ = 1, cR = 1)
plot(Time, mu[-1], main = 'Filter Q=1 R=1', ylim = c(-5, 13),lwd=2)
lines(Time ,y , col = 'green',lwd=2)
lines(ks$xf,col="orange",lwd=2)
r=length(k$Pt)-1
U=k$mt[-r-1]+2*sqrt(k$Pt[r])
L=k$mt[-r-1]-2*sqrt(k$Pt[r])
lines(U,col="blue",lty=2)
lines(L,col="blue",lty=2)
lines(k$mt[-length(k$mt)], col="red",lwd=2)
legend("topleft",
       legend=c('Hidden
        ↪  States','Observations',"Ksmooth_Preds",'Confidence','mypredictions'),
       col=c('black','green',"orange",'blue','red'), pch=c("o",NA,NA,NA,NA),
```

```
      lty=c(NA,1, 1, 2,1), cex=0.8)
# comparison greater Q ratio
k=kalman_filter(At = 1, Ct = 1, Qt = 10, Rt = 0.1, m0=0, P0=1, xt =y)
ks=Ksmooth0(num, y, A = 1, mu0 = 0, Sigma0 = 1, Phi = 1, cQ = 10, cR = 0.1)
plot(Time, mu[-1], main = 'Filter Q=10 R=0.1', ylim = c(-5, 13),lwd=2)
lines(Time ,y , col = 'green',lwd=2)
lines(ks$xf,col="orange",lwd=2)
r=length(k$Pt)-1
U=k$mt[-r-1]+2*sqrt(k$Pt[r])
L=k$mt[-r-1]-2*sqrt(k$Pt[r])
lines(U,col="blue",lty=2)
lines(L,col="blue",lty=2)
lines(k$mt[-length(k$mt)], col="red",lwd=2)
legend("topleft",
      legend=c('Hidden
       ↪  States','Observations',"Ksmooth_Preds",'Confidence','mypredictions'),
      col=c('black','green',"orange",'blue','red'),
      pch=c("o",NA,NA,NA,NA),lty=c(NA,1, 1, 2,1), cex=0.8)
# comparison with grater R ratio
k=kalman_filter(At = 1, Ct = 1, Qt = 0.1, Rt = 10, m0=0, P0=1, xt =y)
ks=Ksmooth0(num, y, A = 1, mu0 = 0, Sigma0 = 1, Phi = 1, cQ = 0.1, cR = 10)
plot(Time, mu[-1], main = 'Filter Q=0.1 R=10', ylim = c(-5, 13),lwd=2)
lines(Time ,y , col = 'green',lwd=2)
lines(ks$xf,col="orange",lwd=2)
r=length(k$Pt)-1
U=k$mt[-r-1]+2*sqrt(k$Pt[r])
L=k$mt[-r-1]-2*sqrt(k$Pt[r])
lines(U,col="blue",lty=2)
lines(L,col="blue",lty=2)
lines(k$mt[-length(k$mt)], col="red",lwd=2)
legend("topleft",
      legend=c('Hidden
       ↪  States','Observations',"Ksmooth_Preds",'Confidence','mypredictions'),
      col=c('black','green',"orange",'blue','red'),
      pch=c("o",NA,NA,NA,NA),lty=c(NA,1, 1, 2,1), cex=0.8)
dev.off()

knitr::include_graphics(c("images/plotA4.png"))
```