**SMF Solutions 7. 6.12.2013.**

The code we provide here is inspired from Lai-Xing, which is available here: http://www.stanford.edu/~xing/statfinbook/_BookFun/ex5.3.2_kalman_capm.txt The code presented here, is however, rewritten consistently with respect to the course material. Moreover, the scripts do not perform exactly the same tasks (which translates into different recursive equations). The Lai-Xing text deals with the estimating problem: for each time $n$, what is the best estimate $\hat{\beta}(n)$ given the information up to time $n$? What we are trying to do in this exercise is a prediction problem, which is dynamic: our task is to compute the estimate for $\beta$ at time $n+1$ given the information up to time $n$. That means that you have observations up to time $n$ and you want to forecast next month's beta. Then, when the next month comes, you have one extra piece of data, which you use to update your estimate for the next period. (i) The function **kalman.update** simply implements the recursive equations driving the Kalman filter in dimension one. It can be written as follows:

```
kalman.update<-function(y, H.n, x.n, sigma.n, Phi.n, Q, R){
K.n<-(Phi.n*sigma.n*H.n)/(H.n*sigma.n*H.n+R)
sigmainc.n<-Phi.n*sigma.n*Phi.n+Q-K.n*(H.n*sigma.n*H.n+R)*K.n
Psi.n<-Phi.n - K.n*H.n
xinc.n<-Psi.n *x.n+ K.n*y
list(x.n = xinc.n,sigma.n=sigmainc.n)
}
```

(ii) The function **kalmanf.estx** takes 5 parameters as input. Here, it is assumed that the regression to initialise the filter is made outside. Note that the parameter $H0$, which corresponds to the value of $H$ over the initialisation period, does not necessarily need to be set as an input parameter, even though it makes the code (a very little) less horrible to read.

```
 kalmanf.estx<-function(y,H,H0,fit0,Q){
#estimate vectors initialised to 0
est.x<-est.sigma<-rep(0,length(y)+1)
#in our case, Phi is one
Phi<-1
R<-sum(fit0$resid^2)/length(fit0$resid)
est.x[1]<-x.n<-as.numeric(fit0$coeff)
est.sigma[1]<-sigma.n<-sum(fit0$resid^2)/(length(fit0$resid)-1)/sum(H0^2)
for (i in 1:length(y)){
kalmanf<-kalman.update(y[i],H[i],x.n,sigma.n,Phi,Q,R)
est.x[i+1]<-x.n<-kalmanf$x.n
est.sigma[i+1]<-sigma.n<-kalmanf$sigma.n
}
```

```
list(est.x = est.x, est.sigma = est.sigma)
}
```

(iii) The final function estimating the betas is

```
ts.beta<-function(y,G,seg0,Q){
#run the regression, the ‘‘weird’’ syntax lm(y~x-1)
#means regress y agianst x with intercept set to zero
fit0<-lm(y[seg0]~G[seg0]-1)
#run the Kalman filter estimation
#G[-seg0] means take G without the first 20 values,
#which are the ones used in the initialisation
est<-kalmanf.estx(y[-seg0],G[-seg0],G[seg0],fit0,Q)
return(est)
}
```

The file's "main" looks like this

```
# learning period = 20 months
seg0 <- seq(1,20)

#red the data
data <- read.table('kalman_data.txt', header=T)
Q <- 0.01
est.Intel <- ts.beta(data$Intel[length(data$Intel):1], data$DJ, seg0, Q)
est.Citi <- ts.beta(data$JPM[length(data$JPM):1], data$DJ, seg0, Q)
est.AMX <- ts.beta(data$AMX[length(data$AMX):1], data$DJ, seg0, Q)
```

Finally, if one wants to save the output in a file named, say, *kalman_est_beta.txt*, this can be done using the following lines:

```
#write all the betas in a file
cat("%Am C Exx G In Pf\n", file="kalman_est_beta.txt")
write( t(cbind(est.Intel$est.x, est.JPM$est.x, est.AMX$est.x)),
file="kalman_est_beta.txt", ncol=3, append=TRUE)
```

One can plot the forecasts using the following commands to superimpose the graphs:

```
 ylim1<-c(-2,3)
plot(est.Intel$est.x,type="l",ylim=ylim1, col="blue",ylab="Beta")
par(new=TRUE)
plot(est.JPM$est.x,type="l",ylim=ylim1,col="red",axes=F,ann=F)
par(new=TRUE)
plot(est.AMX$est.x,type="l",ylim=ylim1,col="black",axes=F,ann=F)
```

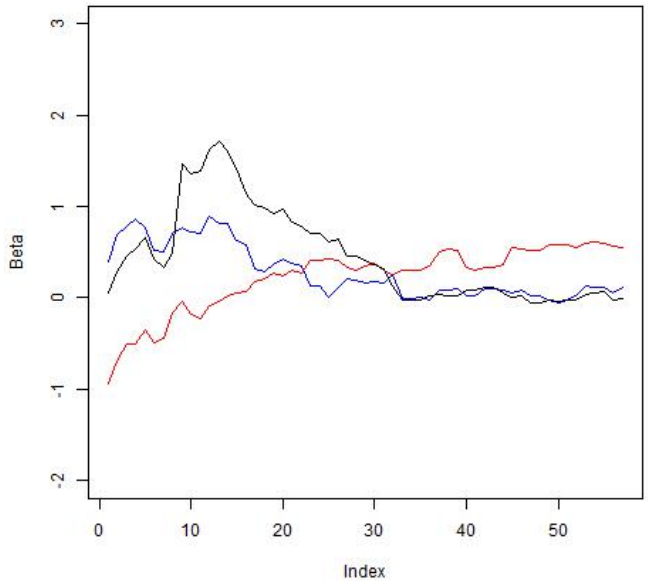


Figure 1: One-period ahead beta forecasts for Intel (blue), JP Morgan (red) and American Express (black)