

# MATH 545: INTRODUCTION TO TIME SERIES ANALYSIS

## WINTER 2025

### THE KALMAN FILTER FOR ARMA MODELS

The Kalman filter can be used for inference for ARMA( $p, q$ ) models if we specify the two equations as follows: let  $m = \max\{p, q + 1\}$ , and

$$\begin{array}{ll} \text{OBSERVATION EQUATION} & Y_t = \mathbf{F}_t \mathbf{X}_t + V_t \\ \text{STATE EQUATION} & \mathbf{X}_{t+1} = \mathbf{G}_t \mathbf{X}_t + \mathbf{W}_t \end{array}$$

where we define the state  $\mathbf{X}_t = (X_{1,t-m+1}, X_{1,t-m+2}, \dots, X_{1,t})^\top$ , and  $\mathbf{W}_t = (0, 0, 0, \dots, Z_{t+1})^\top$ , so that

$$\mathbf{G}_t = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 \\ \phi_m & \phi_{m-1} & \phi_{m-2} & \cdots & \phi_2 & \phi_1 \end{bmatrix} \quad \mathbf{F}_t = [\theta_{m-1} \quad \theta_{m-2} \quad \cdots \quad \theta_1 \quad \theta_0].$$

with  $V_t \equiv 0$  and  $Z_t \sim \text{Normal}(0, \sigma_Z^2)$ .

**EXAMPLE:** Consider the ARMA(1, 1) case

$$Y_t - \phi Y_{t-1} = Z_t + \theta Z_{t-1} \quad t \in \mathbb{R}$$

so that  $m = \max\{1, 2\} = 2$ , and

$$\mathbf{G}_t = \begin{bmatrix} 0 & 1 \\ 0 & \phi \end{bmatrix} \quad \mathbf{F}_t = [\theta \quad 1].$$

as  $\phi_2 \equiv 0$ , and  $\theta_0 = 1$ . Therefore

$$\begin{array}{ll} \text{OBSERVATION EQUATION} & Y_t = [\theta \quad 1] \begin{bmatrix} X_{1,t-1} \\ X_{1,t} \end{bmatrix} + V_t \\ \text{STATE EQUATION} & \begin{bmatrix} X_{1,t} \\ X_{1,t+1} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & \phi \end{bmatrix} \begin{bmatrix} X_{1,t-1} \\ X_{1,t} \end{bmatrix} + \begin{bmatrix} 0 \\ Z_{t+1} \end{bmatrix} \end{array}$$

From the first equation, we have

$$Y_t = X_{1,t} + \theta X_{1,t-1} = (1 + \theta B)X_t$$

and from the second equation the identity  $X_{1,t} = X_{1,t}$ , and

$$X_{1,t+1} = \phi_1 X_{1,t} + Z_{t+1}$$

that is  $(1 - \phi)X_{1,t+1} = Z_{t+1}$  or equivalently  $(1 - \phi B)X_{1,t} = Z_t$ , so that

$$X_{1,t} = (1 - \phi B)^{-1} Z_t$$

Hence

$$Y_t = (1 + \theta B)X_t = (1 + \theta B)(1 - \phi B)^{-1} Z_t$$

as required. As a special case with  $\phi = 0$ , we have the MA(1) model where

$$\text{OBSERVATION EQUATION} \quad Y_t = [\theta \ 1] \begin{bmatrix} X_{1,t-1} \\ X_{1,t} \end{bmatrix} + V_t$$

$$\text{STATE EQUATION} \quad \begin{bmatrix} X_{1,t} \\ X_{1,t+1} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} X_{1,t-1} \\ X_{1,t} \end{bmatrix} + \begin{bmatrix} 0 \\ Z_{t+1} \end{bmatrix}$$

or equivalently

$$\text{OBSERVATION EQUATION} \quad Y_t = [1 \ \theta] \begin{bmatrix} X_{1,t} \\ X_{1,t-1} \end{bmatrix} + V_t$$

$$\text{STATE EQUATION} \quad \begin{bmatrix} X_{1,t+1} \\ X_{1,t} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} X_{1,t} \\ X_{1,t-1} \end{bmatrix} + \begin{bmatrix} Z_{t+1} \\ 0 \end{bmatrix}$$

**EXAMPLE:** In the following simulation  $\sigma_Z = 1$ , with  $\mathbf{X}_1 = (0, 0, 0, 0)^\top$ , for the ARMA(2, 3) model given by

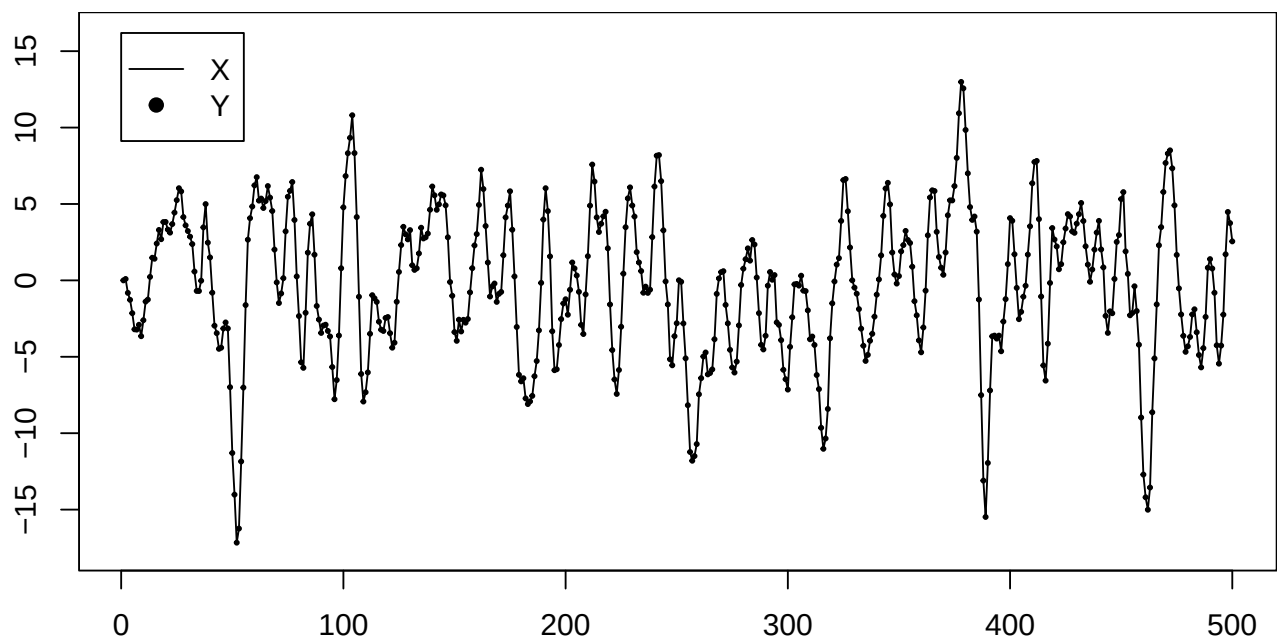
$$(1 - 0.8B + 0.15B^2)X_t = (1 + B + B^2 + B^3)Z_t$$

We have  $m = q + 1 = 4$ ,

$$\text{OBSERVATION EQUATION} \quad Y_t = [1 \ 1 \ 1 \ 1] \begin{bmatrix} X_{1,t-3} \\ X_{1,t-2} \\ X_{1,t-1} \\ X_{1,t} \end{bmatrix} + V_t$$

$$\text{STATE EQUATION} \quad \begin{bmatrix} X_{1,t-2} \\ X_{1,t-1} \\ X_{1,t} \\ X_{1,t+1} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -0.15 & 0.8 \end{bmatrix} \begin{bmatrix} X_{1,t-3} \\ X_{1,t-2} \\ X_{1,t-1} \\ X_{1,t} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ Z_{t+1} \end{bmatrix}$$

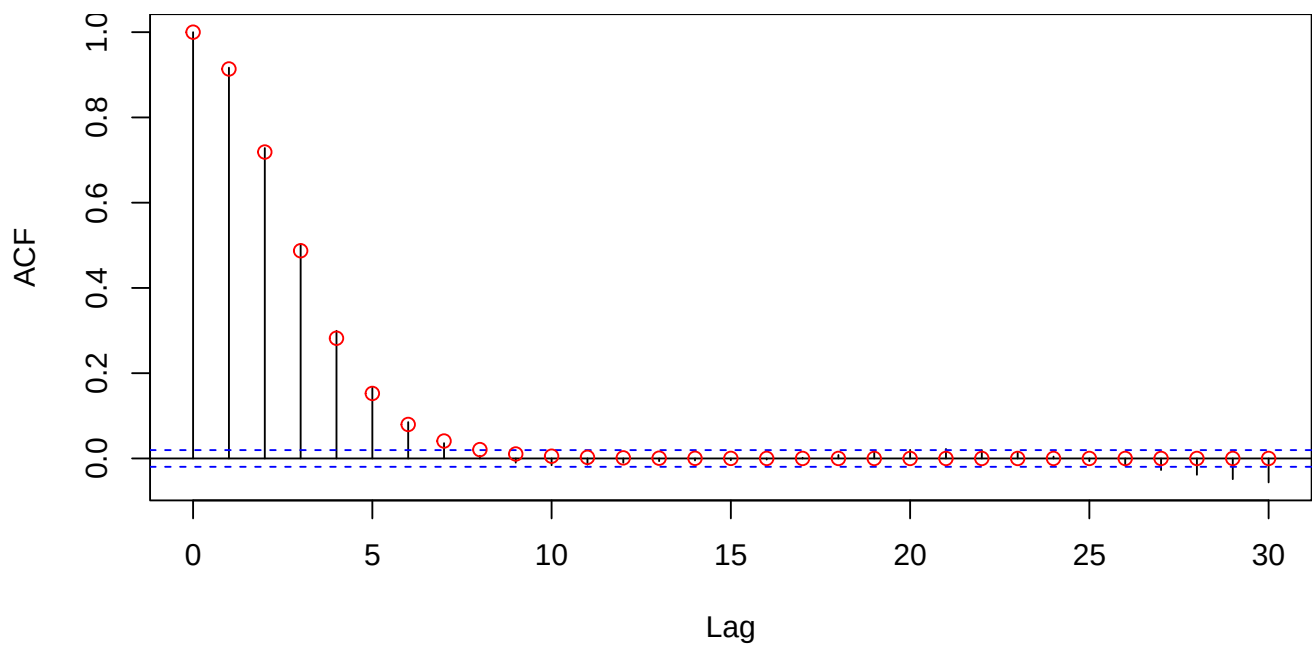
```
set.seed(3834); n<-10000; p<-2; q<-3; m<-max(p,q+1)
phi.vec<-theta.vec<-rep(0,m)
phi.vec[1:p]<-c(0.8,-0.15)
theta.vec[1:q]<-c(1,1,1)
X<-matrix(0,ncol=n,nrow=m); Y<-mt<-rep(0,n)
F<-c(rev(theta.vec[1:(m-1)]),1)
G<-matrix(0,m,m); for(j in 1:(m-1)){G[j,j+1]<-1}
G[m,]<-rev(phi.vec)
sigV<-0; sigZ<-1
V<-rnorm(n)*sigV; Z<-rnorm(n)*sigZ
X[,1]<-rep(0,m)
Y[1]<-(F%*%X[,1])[1,1]+V[1]
for(i in 2:n){
  Wi<-c(rep(0,m-1),Z[i])
  X[,i]<-G%*%X[,i-1]+Wi
  mt[i]<-(F%*%X[,i])[1,1]
  Y[i]<-mt[i]+V[i]
}
par(mar=c(2,3,0,0)); plot(mt[1:500],type='l',ylim=range(Y));
points(Y[1:500],pch=19,cex=0.3)
legend(0,max(Y),c('X','Y'),col=c('black','black'),pch=c(-1,19),lty=c(1,-1))
```



```
psi.vec<-ARMAtoMA(ar=phi.vec,ma=theta.vec,lag.max=30)
varY<-1+sum(psi.vec^2)
true.acf<-ARMAacf(ar=phi.vec,ma=theta.vec,lag.max=30)
true.acv<-true.acf*varY
varY;var(Y)

+ [1] 20.30252
+ [1] 20.81826

par(mar=c(4,4,0,0))
acf(Y,lag.max=30,main='ACF of Y');points(0:30,true.acf,col='red')
```



To implement the Kalman Filter, we specify the various components in the usual code.

```
##Compute Estimates using optim and a likelihood computing function

ml.armacq<-function(x,nv,yv,pv,qv){

  m<-max(pv,qv+1);Kval<-1;Lval<-m
  phiv<-thetav<-rep(0,m)
  phiv[1:pq]<-x[1:pq]
  thetav[1:qv]<-x[(pq+1):(pq+qv)]

  Fm<-c(rev(thetav[1:(m-1)]),1)
  Gm<-matrix(0,m,m);for(j in 1:(m-1)){Gm[j,j+1]<-1}
  Gm[m,]<-rev(phiv)
  Sigmat<-0
  Omegat<-diag(c(rep(0,m-1),exp(x[pq+qv+1])))

  att<-array(0,c(nv,Lval))
  apt<-array(0,c(nv+1,Lval))
  Ptt<-array(0,c(nv,Lval,Lval))
  Ppt<-array(0,c(nv+1,Lval,Lval))
  rt<-array(0,c(nv,Kval))
  Mt<-array(0,c(nv,Kval,Kval))

  apt[1,]<-rep(0,m)
  Ppt[1,,]<-Omegat

  #Filter Forward

  loglike<-0
  for(i in 1:nv){

    Fmat<-matrix(Fm,nrow=Kval,ncol=Lval)
    rt[i,]<-yv[i] - Fmat %*% apt[i,]
    Mt[i,,]<-Fmat %*% Ppt[i,,] %*% t(Fmat) + Sigmat

    if(Sigmat == Inf || Mt[i,,] == 0 || is.na(Mt[i,,])){return(Inf)}

    if(Mt[i,]==Inf){
      Mtinv<-matrix(0,1,1)
    }else{
      Mtinv<-solve(Mt[i,,])
    }

    att[i,]<-apt[i,] + Ppt[i,,] %*% t(Fmat)%*% Mtinv%*% rt[i,]
    Ptt[i,,]<-Ppt[i,,]-Ppt[i,,] %*% t(Fmat)%*%Mtinv%*%Fmat %*% Ppt[i,,]
    apt[i+1,]<-Gm %*% att[i,]
    Ppt[i+1,,]<-Gm %*% Ptt[i,,] %*% t(Gm) + Omegat

    loglike<-loglike-0.5*log(det(matrix(Mt[i,,],Kval,Kval)))-
      0.5*t(rt[i,]) %*% Mtinv %*% rt[i,]
  }
  return(as.numeric(loglike))
}
```

```

#Perform the optimization using optim
Y<-Y[1:1000]
xv<-c(0.1,rep(0,p-1),rep(0,q+1))
ARMA.ml<-optim(xv,ml.armacq,nv=length(Y),pv=p,qv=q,yv=Y,method='BFGS',hessian=T,
               control=list(trace=10,fnscale=-1,maxit=500,abstol=1e-6))

+ initial value 8612.265473
+ iter 10 value 708.671217

+ Warning in log(det(matrix(Mt[i, , ], Kval, Kval))): NaNs produced

+ iter 20 value 510.060811

+ Warning in log(det(matrix(Mt[i, , ], Kval, Kval))): NaNs produced

+ iter 30 value 485.923624
+ final value 484.094681
+ converged

```

After the optimization, the parameter estimates and standard errors are as follows:

```

phi.hat<-ARMA.ml$par[1:p]
theta.hat<-ARMA.ml$par[(p+1):(p+q)]
ese<-sqrt(diag(solve(-ARMA.ml$hessian)))
phi.es<-ese[1:p];theta.es<-ese[(p+1):(p+q)]
cbind(c(phi.hat,theta.hat),c(phi.es,theta.es))

+           [,1]      [,2]
+ [1,]  0.8053809 0.031380440
+ [2,] -0.1361270 0.031381222
+ [3,]  0.9998136 0.001859405
+ [4,]  0.9987398 0.002225881
+ [5,]  1.0001139 0.001978247

(sigZ.hat<-exp(ARMA.ml$par[p+q+1]))

+ [1] 0.9686916

```

These results resemble closely the results from arima:

```

arima(Y,order=c(2,0,3),method='ML',include.mean=FALSE)

+
+ Call:
+ arima(x = Y, order = c(2, 0, 3), include.mean = FALSE, method = "ML")
+
+ Coefficients:
+      ar1      ar2      ma1      ma2      ma3
+      0.8040 -0.1345  0.9999  0.9949  0.9949
+ s.e.  0.0316  0.0316  0.0064  0.0061  0.0075
+
+ sigma^2 estimated as 0.9688:  log likelihood = -1411.8,  aic = 2835.59

```