library(astsa)

dna = bnrf1ebv # the BNRF1 gene of EBV

head(dna)

gcTF = (dna==2)|(dna==3)

gc = gcTF\*1

head(gc)

length(gc)

mean(gc)

# Apply a moving average filter (window size = 100)

window\_size = 75

gc\_smooth = filter(gc, c(1/(2\*window\_size), rep(1/window\_size, window\_size), 1/(2\*window\_size)), sides=2)

# Plot the GC-content variation along the gene

plot(gc\_smooth, type='l', col='blue', lwd=2, xlab="Position",

ylab="GC", main="GC Variation")

# Define transition matrix based on expected segment lengths

P = matrix(c(0.995, 0.005, # Typical -> GC-rich

0.01, 0.99), # GC-rich -> Typical

ncol=2, byrow=TRUE)

# Define emission probabilities (Bernoulli model)

hmmFilter = function(P, l)

function(f, y) {

fNew = (f %\*% P) \* l(y)

fNew / sum(fNew)

}

advance = hmmFilter(

P,

function(y) c(dbinom(y, 1, 0.6), dbinom(y, 1, 0.7)) # Bernoulli emissions

)

# Forward filtering

fpList = Reduce(advance, gc, c(0.5, 0.5), acc=TRUE)

fpMat = sapply(fpList, cbind)

fp2Ts = ts(fpMat[2, -1], start=start(gc), freq=frequency(gc))

# Plot GC-content with filtered probabilities

plot(gc\_smooth, type='l', col='blue', main="GC-Content")

lines(fp2Ts, col='red', lwd=2)

abline(h=mean(gc), col="black", lty=2)

hmmSmoother = function(P)

function(fp, sp) {

fp \* ((sp / (fp %\*% P)) %\*% t(P))

}

backStep = hmmSmoother(P)

spList = Reduce(backStep, fpList, right=TRUE, acc=TRUE)

spMat = sapply(spList, cbind)

sp2Ts = ts(spMat[2, -1], start=start(gc), freq=frequency(gc))

# Plot GC-content with smoothed probabilities

plot(gc\_smooth, type='l', col='blue', main="GC-Content with HMM Smoothed Probability")

lines(sp2Ts, col='red', lwd=2)

abline(h=mean(gc), col="black", lty=2)

tsplot(cardox, col=4, lwd=1.5,

main="Monthly carbon dioxide levels")

library(dlm)

buildMod <- function(lwv) {

# Observation variance

V <- exp(lwv[1])

# System variances

W\_level <- exp(lwv[2])

W\_slope <- exp(lwv[3])

W\_seasonal <- exp(lwv[4])

# Locally linear trend components

trend <- dlmModPoly(order = 2, dV = V, dW = c(W\_level, W\_slope))

# Monthly seasonal component (12 months)

seasonal <- dlmModSeas(frequency = 12, dV = 0, dW = c(W\_seasonal, rep(0, 10)))

# Combine the components

mod <- trend + seasonal

return(mod)

}

# Initial parameter guesses (log-transformed variances)

init\_params <- c(log(1), log(1), log(1), log(1)) # Corresponding to V, W\_level, W\_slope, W\_seasonal

# Optimize the parameters

opt <- dlmMLE(cardox, parm = init\_params, build = buildMod)

opt

optimized\_variances <- exp(opt$par)

optimized\_variances

# 2.884121e-02 3.891608e-02 5.705924e-06 1.206646e-03

final\_mod <- buildMod(opt$par)

fit <- dlmFilter(cardox, final\_mod)

fore <- dlmForecast(fit, nAhead = 60) # Forecast 60 months

# Forecasted values

pred <- ts(c(tail(cardox, 1), fore$f),

start = end(cardox), frequency = frequency(cardox))

# Upper and lower bounds (2 standard deviations)

upper <- ts(c(tail(cardox, 1), fore$f + 2 \* sqrt(unlist(fore$Q))),

start = end(cardox), frequency = frequency(cardox))

lower <- ts(c(tail(cardox, 1), fore$f - 2 \* sqrt(unlist(fore$Q))),

start = end(cardox), frequency = frequency(cardox))

all <- ts(c(cardox, upper[-1]), start = start(cardox),

frequency = frequency(cardox))

tsplot(all, ylab = "CO2 Levels",

main = "Forecasts with 2SD Intervals")

lines(cardox, col = 4, lwd = 1.5) # Original data

lines(pred, col = 2, lwd = 2) # Forecasted values

lines(upper, col = 2) # Upper bound

lines(lower, col = 2) # Lower bound