London R Practice

Rob Hayward

September 22, 2013

1 Introduction

This is a file to run through the presentation that was made at the September 2013 London R by Maarten Speekenbrink.

I am not sure how to load the appropriate packages (so this can probably be deleted in the future). However, I will do that in a chunk. The information is in the title: dependent mixed models. This is to implement micture and hidden Markov models.

2 Mixed Models

From the London R presentation. In a *mixture model* each observation is assumed to be drawn from a number of distinct subpopulation ("component distributions" according to Maarten). The distribution from which the observation is drawn is not directly observable and therefore it is represented by a *latent state*.

A mixture distribution is defined as

$$p(Y_1 = y) = \sum_{i-1}^{N} p(Y_t = y|S_t = i)P(S_t = i) \tag{1}$$

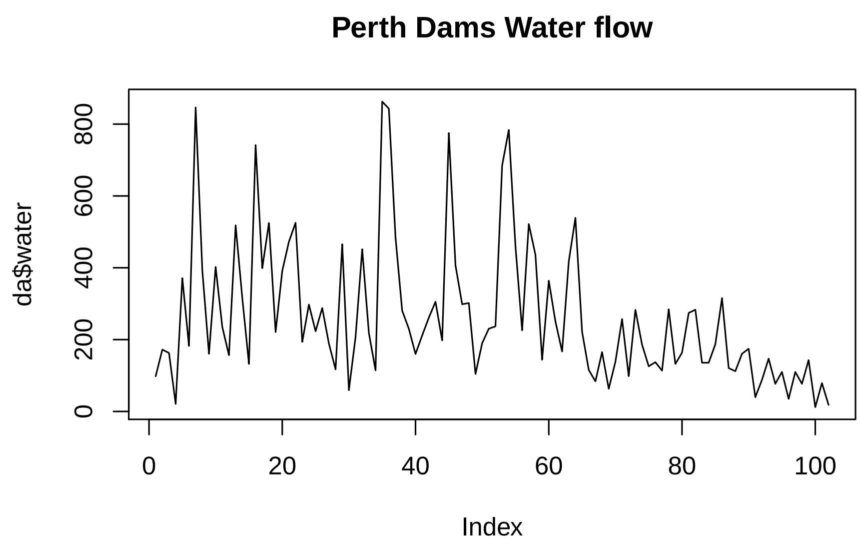
where,

- $S_t \in 1, \ldots, N$ denotes the latent state or class of observation t
- $P(S_t = i)$ denots the probability of the latent state t equals i
- $p(Y_t = y|S_t = i)$ denotes the density of observation of Y_t conditional on latent state being $S_t = i$.

3 Perth Water Example

First download the data and plot.

```r
da <- read.csv("./Perth.csv", header = TRUE)
plot(da$water, type = "l", main = "Perth Dams Water flow")
```



The aim is to model the water flow. the typical method would try to impose a level model (lm1) or a linear (lmyr) or quadratic model (lmyr2).

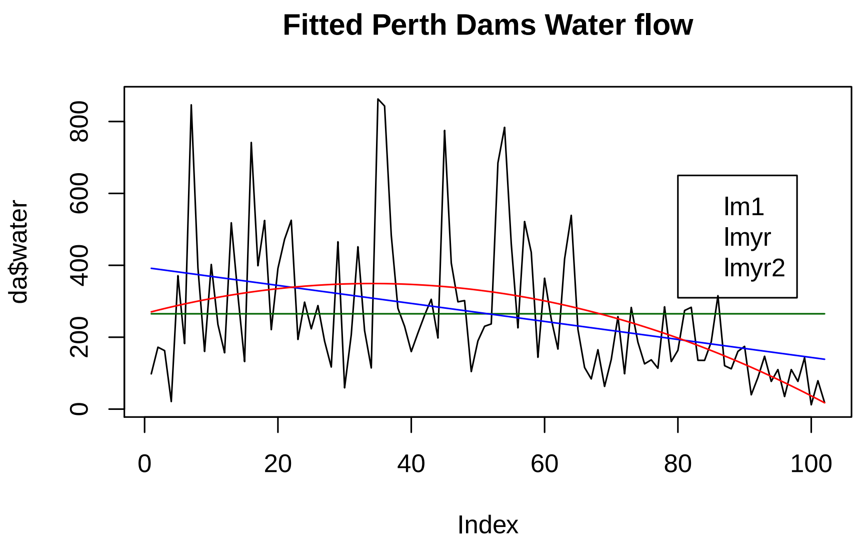
```r
lm1 <- lm(da$water ~ 1, data = da)
lmyr <- lm(da$water ~ yr, data = da)
lmyr2 <- lm(da$water ~ yr + I(yr^2), data = da)
dd <- xtable(anova(lm1, lmyr, lmyr2), digits = 2)
print(dd)
```

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	101.00	3811695.17				
2	100.00	3257024.21	1.00	554670.95	18.68	0.00
3	99.00	2939677.35	1.00	317346.87	10.69	0.00

The linear and quadratic equiations show a vaiance that is significantly lower than the model with just the fixed level.

```r
da <- read.csv("./Perth.csv", header = TRUE)
plot(da$water, type = "l", main = "Fitted Perth Dams Water flow")
lines(lm1$fitted, col = "dark green")
lines(lmyr$fitted, col = "blue")
```

```
lines(lmyr2$fitted, col = "red")
legend(x = 80, y = 650, legend = c("lm1", "lmyr", "lmyr2"))
```

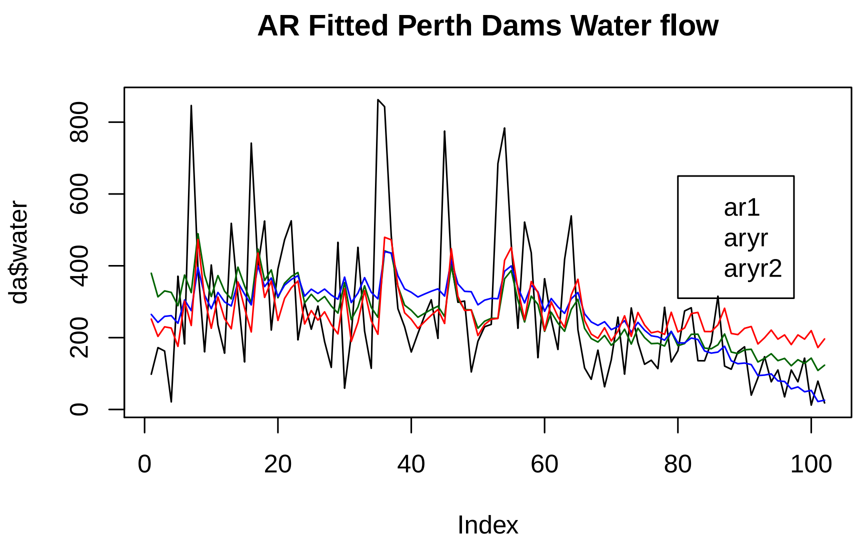


AR models can also be used and assessed using the Akaike Information Criteria or Bayesian Information Criteria.

```
arOrder <- ar(da$water)$order
ar1 <- arima(da$water, c(arOrder, 0, 0))
aryr <- arima(da$water, c(arOrder, 0, 0), xreg = da$yr)
aryr2 <- arima(da$water, c(arOrder, 0, 0), xreg = cbind(yr = scale(da$yr), yr2 = scale(da$yr
print(c(ar1 = AIC(ar1), aryr = AIC(aryr), aryr2 = AIC(aryr2)))

##   ar1  aryr aryr2
##  1355  1349  1344

print(c(lm1 = AIC(lm1), lmyr = AIC(lmyr), lmyr2 = AIC(lmyr2)))

##   lm1  lmyr lmyr2
##  1367  1353  1345
```

```
require(forecast)
da <- read.csv("./Perth.csv", header = TRUE)
plot(da$water, type = "l", main = "AR Fitted Perth Dams Water flow")
lines(fitted(aryr), col = "dark green")
lines(fitted(aryr2), col = "blue")
lines(fitted(ar1), col = "red")
```

```r
legend(x = 80, y = 650, legend = c("ar1", "aryr", "aryr2"))
```



4 Change Model

This is a model that assumes one or more discrete change points in the data. It may be the mean, trend or other parameters that may change. In this example with the S&P 500 it is the mean and the standard deviation. There is a transition matrix that defines the change points. For example, if there is one transition the matrix would be along the lines of

$$\begin{pmatrix} p_1 & 1-p_1 \\ 0 & 1 \end{pmatrix}$$

Where p_1 is the probability that the system will be in state 1. One there is a switch to state two. This matrix can be extended for more states. Need to come back and look at this if I can get the data.

```r
mod2 <- depmix(water ~ 1, data = da, ns = 2, trst = c(0.9, 0.1, 0, 1), inst = c(1,
    0))
set.seed(1)
fm2 <- fit(mod2, verbose = FALSE)

## iteration 3 logLik: -654.5

summary(fm2)
```

```
## Initial state probabilties model
## Model of type multinomial (identity), formula: ~1
## <environment: 0x0000000008b89cd0>
## Coefficients:
##      [,1] [,2]
## [1,]    1    0
##
## Transition model for state (component) 1
## Model of type multinomial (identity), formula: ~1
## <environment: 0x000000000867dd80>
## Coefficients:
## [1] 0.9845 0.0155
##
## Transition model for state (component) 2
## Model of type multinomial (identity), formula: ~1
## <environment: 0x000000000867dd80>
## Coefficients:
## [1] 0 1
##
##
## Response model(s) for state 1
##
## Response model for response 1
## Model of type gaussian (identity), formula: water ~ 1
## Coefficients:
## [1] 337
## sd  204.1
##
##
## Response model(s) for state 2
##
## Response model for response 1
## Model of type gaussian (identity), formula: water ~ 1
## Coefficients:
## [1] 141.4
## sd  76.16
```

The information here is the inital state probabilities, the transition model for state 1 and state 2; the response model for state 1 and the response model for state 2. These show a mean value of 337 and 141 respectively and a standard deviation of 204.1 and 76.16. There is a decline in the amount of water and there is a fall in the variability.

The posterior() function can be used to obtain the maximum *a posteriori* state sequence (column 1) and the posterior state probabilities (remaining columns).

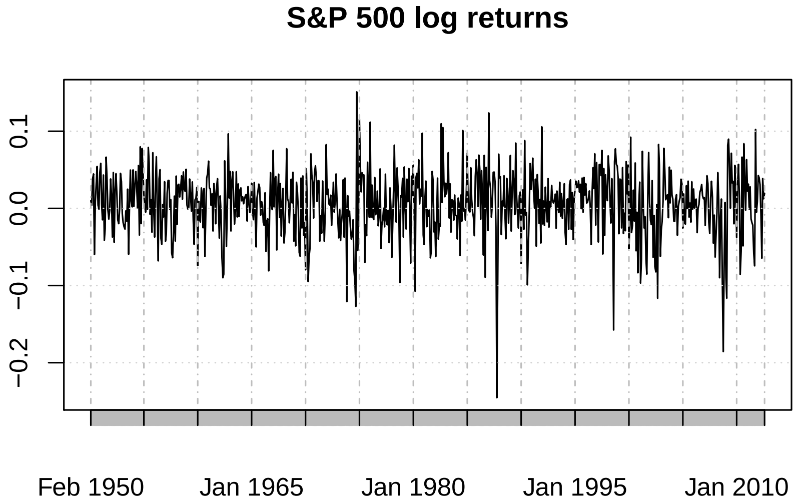
```r
pst <- posterior(fm2)
head(pst)

##   state     X1      X2
## 1     1 1.0000 0.00000
## 2     1 0.9489 0.05110
## 3     1 0.8312 0.16875
## 4     1 0.6558 0.34420
## 5     1 0.9849 0.01513
## 6     1 0.9536 0.04642

tail(pst)

##     state        X1 X2
## 97      2 3.266e-13  1
## 98      2 7.622e-14  1
## 99      2 1.783e-14  1
## 100     2 7.811e-15  1
## 101     2 1.806e-15  1
## 102     2 7.272e-16  1

pst[, 1]

##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [36] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2
##  [71] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
```

5 S&P 500 Example

```r
library(TTR)
# load SP500 returns
Sys.setenv(tz = "UTC")
sp500 <- getYahooData("^GSPC", start = 19500101, end = 20120909, freq = "daily")
ep <- endpoints(sp500, on = "months", k = 1)
sp500 <- sp500[ep[2:(length(ep) - 1)]]
sp500$logret <- log(sp500$Close) - lag(log(sp500$Close))
sp500 <- na.exclude(sp500)
```

Now plot the data to get an idea of what it looks like.

```r
plot(sp500$logret, main = "S&P 500 log returns")
```



The aim is to identify the bull and bear markets. First set up the model (mod). This is a model of the log return with two states. Then fit the model.

```
mod <- depmix(logret ~ 1, nstates = 2, data = sp500)
set.seed(1)
fm2 <- fit(mod, verbose = FALSE)

## iteration 88 logLik: 1348
```

The number of iterations and the log likelihood is printed. Now summarise the information.

```
depmixS4::summary(fm2)

## Initial state probabilties model
## Model of type multinomial (identity), formula: ~1
## <environment: 0x0000000008efd2c8>
## Coefficients:
##           [,1] [,2]
## [1,] 1.735e-47    1
##
## Transition model for state (component) 1
## Model of type multinomial (identity), formula: ~1
## <environment: 0x0000000011cdddc0>
## Coefficients:
## [1] 0.8215 0.1785
##
```

```
## Transition model for state (component) 2
## Model of type multinomial (identity), formula: ~1
## <environment: 0x0000000011cdddc0>
## Coefficients:
## [1] 0.03914 0.96086
##
##
## Response model(s) for state 1
##
## Response model for response 1
## Model of type gaussian (identity), formula: logret ~ 1
## Coefficients:
## [1] -0.01505
## sd  0.06484
##
##
## Response model(s) for state 2
##
## Response model for response 1
## Model of type gaussian (identity), formula: logret ~ 1
## Coefficients:
## [1] 0.01045
## sd  0.03378
```