The multivariate distribution in hmmTMB

Théo Michelot

2024-04-17

This vignette describes the use of the multivariate normal distribution as an observation distribution for a hidden Markov model (HMM) in the R package hmmTMB. We illustrate it using an application to stock prices, and compare it to the use of several univariate normal distributions.

It is common to say that an HMM is "multivariate" when it has more than one observation variable, even when those variables are not modelled with a multivariate distribution. This is because the dependence of the observation distributions on the state process can create correlation between observed variables even when they are assumed to be conditionally independent (given the state). Here, we focus specifically on the case where dependence between variables *within each state* of the HMM should be modelled explicitly.

1 Model description

We consider the observation vector $\boldsymbol{Z}_t = (Z_{t1}, Z_{t2}, \dots, Z_{td})$ at time $t = 1, 2, \dots$. Further, we assume that $\boldsymbol{Z}_t$ follows a multivariate normal distribution with parameters dependent on the current value of the hidden state S_t, i.e.,

$$\boldsymbol{Z}_t \mid S_t = k \sim MVN(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

where, in state k, the observation distribution is parameterised by the mean vector

$$\boldsymbol{\mu}_k = (\mu_{1k}, \mu_{2k}, \dots, \mu_{dk})$$

and the covariance matrix

$$\boldsymbol{\Sigma}_k = \begin{pmatrix} \sigma_{1k}^2 & \sigma_{1k}\sigma_{2k}\rho_{12k} & \sigma_{1k}\sigma_{3k}\rho_{13k} & \cdots & \sigma_{1k}\sigma_{dk}\rho_{1dk} \\ \sigma_{2k}\sigma_{1k}\rho_{21k} & \sigma_{2k}^2 & \sigma_{2k}\sigma_{3k}\rho_{23k} & \cdots & \sigma_{2k}\sigma_{dk}\rho_{2dk} \\ \sigma_{3k}\sigma_{1k}\rho_{31k} & \sigma_{3k}\sigma_{2k}\rho_{32k} & \sigma_{3k}^2 & \cdots & \sigma_{3k}\sigma_{dk}\rho_{3dk} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sigma_{dk}\sigma_{1k}\rho_{d1k} & \sigma_{dk}\sigma_{2k}\rho_{d2k} & \sigma_{dk}\sigma_{3k}\rho_{d3k} & \cdots & \sigma_{dk}^2 \end{pmatrix}$$

Here, $\sigma_{ik} > 0$ is the standard deviation of the i-th variable in state k, and $\rho_{ijk} = \rho_{jik} \in [0, 1]$ is the correlation coefficient between the i-th and j-th variables in state k.

In each state, there are therefore d mean parameters, d standard deviation parameters, and $d(d-1)/2$ correlation parameters to estimate for this observation model. (For example, there are 5 parameters when $d = 2$, and 9 parameters when $d = 3$.) In hmmTMB, all those parameters can be modelled as functions of covariates, although care may be needed to avoid numerical and computational problems in large models.

2 Example data

We consider a bivariate data set of daily log-returns for the stock prices of the Coca-Cola Company and PepsiCo Inc, between 2000 and 2024. We use the package quantmod to download the data from Yahoo (Ryan and Ulrich (2024)), and we derive the log-returns R_t from the time series (Z_t) (where Z_t is the stock price for Coca-Cola or PepsiCo) as $R_t = 100 \times (\log(Z_{t+1}) - \log(Z_t))$.

```
library(ggplot2)
theme_set(theme_bw())
library(quantmod)

start <- as.Date('2000-01-01')
end <- as.Date('2024-04-01')

# Get stock prices from Yahoo
names <- c("KO", "PEP")
raw <- lapply(names, function(name) {
    dat <- getSymbols(name, src = 'yahoo',
                      auto.assign = FALSE,
                      from = start,
                      to = end)
    return(as.data.frame(dat))
})

# Transform to log-returns
n <- nrow(raw[[1]])
logret <- 100*sapply(raw, function(x) {
    log(x[-1,4]) - log(x[-n,4])
```

```r
})
data <- as.data.frame(logret)
colnames(data) <- names
# Add time column
data$time <-  lubridate::ymd(rownames(raw[[1]]))[-n]

head(data)
```

```
         KO        PEP       time
1  0.1108033 -2.5752496 2000-01-03
2  0.8820344 -2.4649135 2000-01-04
3  0.1097093  4.3598884 2000-01-05
4  6.3715814  2.6937656 2000-01-06
5 -3.2412665 -2.0134908 2000-01-07
6  3.3440943 -0.3395589 2000-01-10
```

The data frame has 6097 rows, and two columns: one for Coca-Cola (`KO`) and one for PepsiCo (`PEP`). In hmmTMB, the multivariate distribution requires that a single column of the input data frame include all variables. That column should be a list where each entry is a vector of observations at that time point (one observation for each variable). This can be done with the function `asplit()`, which we use to combine `KO` and `PEP` into a single column `z`.

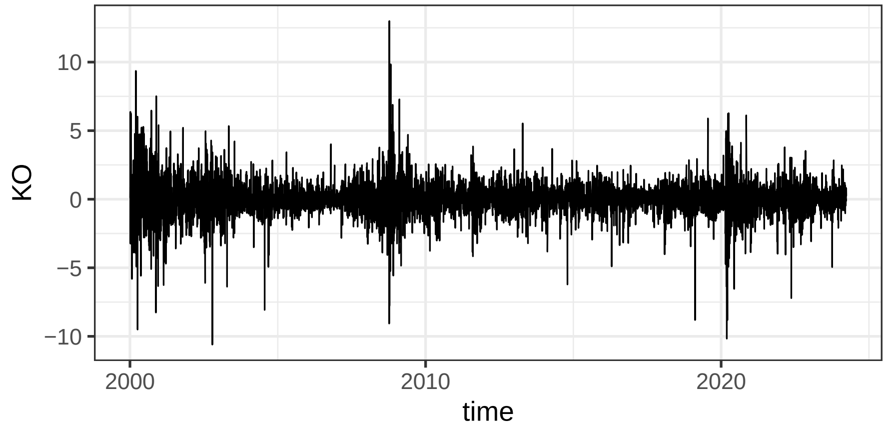
```r
# Create multivariate column for HMM analysis
data$z <- asplit(data[,names], MARGIN = 1)

head(data)
```

```
         KO        PEP       time                     z
1  0.1108033 -2.5752496 2000-01-03 0.1108033, -2.5752496
2  0.8820344 -2.4649135 2000-01-04 0.8820344, -2.4649135
3  0.1097093  4.3598884 2000-01-05  0.1097093, 4.3598884
4  6.3715814  2.6937656 2000-01-06    6.371581, 2.693766
5 -3.2412665 -2.0134908 2000-01-07  -3.241266, -2.013491
6  3.3440943 -0.3395589 2000-01-10 3.3440943, -0.3395589
```

A time series plot of the data suggests that the stocks alternate between periods of high variance and periods of low variance, which can be modelled with a hidden Markov model.

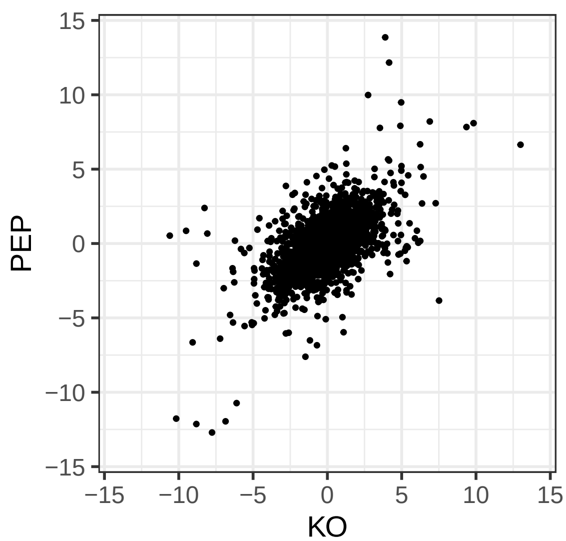
```r
# Time series of KO log-returns
ggplot(data, aes(time, KO)) +
```

```
    geom_line(linewidth = 0.3)
```



We can also look at a scatterplot of the two observation variables, to see that they are correlated (correlation ≈ 0.6). This correlation could not be captured by univariate observation distributions, so we decide to use a multivariate normal model.

```
# Scatterplot of KO vs PEP log-returns
ggplot(data, aes(KO, PEP)) +
    geom_point(size = 0.5) +
    coord_equal(xlim = c(-14, 14), ylim = c(-14, 14))
```



3 Multivariate normal HMM

Model definition is similar to other models, and more details can be found in the general vignette "*Analysing time series data with hidden Markov models in hmmTMB*". We first create an object of class `MarkovChain` for the hidden state process. We will use a 2-state

HMM in the hope that the two states will capture "low variance" and "high variance" periods, so we specify `n_states = 2`.

```
library(hmmTMB)

# Create state model
hid <- MarkovChain$new(data = data,
                       n_states = 2)
```

To create the `Observation` model object, we need to specify an observation distribution. We set this to `mvnorm` to model the multivariate variable `z` with a multivariate normal distribution. We also need to enter initial parameter values for the model fitting. As described above, this model has five parameters in each state k:

- μ_{1k} is the mean log-return for `KO`, and μ_{2k} is the mean log-return for `PEP`;
- σ_{1k} is the standard deviation for `KO`, and σ_{2k} is the standard deviation for `PEP`;
- ρ_{12k} is the correlation coefficient between the log-returns of `KO` and `PEP`.

Log-returns are typically centred around zero, and we don't expect the means to depend on the state much, so we set the starting values for μ_{1k} (`mu1`) and μ_{2k} (`mu2`) to 0. We expect the two states to capture low-variance and high-variance periods, respectively, so we use a larger standard deviation in the second state than for the first state. Here, we use the same initial values for the standard deviations for the two variables (`sd1` for `KO` and `sd2` for `PEP`), because they seem to have the same order of magnitude of variance in the data, but this need not be the case. Finally, we initialise the correlation coefficient `corr12` to 0.6 in both states, as this is the overall correlation between the two variables. If we expected the two states to display different correlations, we could use two different values here.

```
# Create observation model
dists <- list(z = "mvnorm")
par0 <- list(z = list(mu1 = c(0, 0),
                      mu2 = c(0, 0),
                      sd1 = c(0.5, 2),
                      sd2 = c(0.5, 2),
                      corr12 = c(0.6, 0.6)))
obs1 <- Observation$new(data = data,
                        dists = dists,
                        n_states = 2,
                        par = par0)
```

We can now combine the two model components into one `HMM` object, and fit it to estimate all model parameters. Fitting this model to around 6000 observations takes 3 seconds on a laptop. We use the method `$par()` to output the estimated parameters for the observation model (`$obspar`) and for the state process (`$tpm`).

```r
# Create HMM
hmm1 <- HMM$new(obs = obs1, hid = hid)
hmm1$fit(silent = TRUE)

# Estimated parameters
lapply(hmm1$par(), round, 3)
```

```
$obspar
, , 1

         state 1 state 2
z.mu1      0.044  -0.112
z.mu2      0.047  -0.059
z.sd1      0.819   2.400
z.sd2      0.804   2.305
z.corr12   0.654   0.567



$tpm
, , 1

        state 1 state 2
state 1   0.969   0.031
state 2   0.124   0.876
```

We find

$$
\begin{aligned}
&\boldsymbol{\mu}_1 = (0.044, 0.047)\\
&\boldsymbol{\mu}_2 = (-0.112, -0.059)\\
&\sigma_{11} = 0.819, \quad \sigma_{12} = 2.4\\
&\sigma_{21} = 0.804, \quad \sigma_{22} = 2.305\\
&\rho_{121} = 0.654, \quad \rho_{122} = 0.567
\end{aligned}
$$

The two states have mean log-returns close to 1 (although log-returns are on average slightly positive in state 1 and slightly negative in state 2). As expected, state 1 has lower variance than state 2 for both variables, perhaps corresponding to periods when the market was more stable. There is strong positive correlation between the two variables in both states (slightly stronger in state 1).

Alternatively, we can use the function `obs1$par_alt()` to output the covariance matrix in each state (rather than the standard deviations and correlations).

```
rapply(object = obs1$par_alt(var = "z"), f = round, how = "list", digits = 3)
```

```
$S1
$S1$mu
  mu1   mu2
0.044 0.047

$S1$Sigma
      1     2
1 0.671 0.430
2 0.430 0.646


$S2
$S2$mu
   mu1    mu2
-0.112 -0.059

$S2$Sigma
      1     2
1 5.760 3.137
2 3.137 5.313
```

4 Comparison to univariate normal distributions

An alternative approach for this data set would be to treat the log-returns for Coca-Cola and PepsiCo as conditionally independent given the state. That is, we could model each variable with a univariate normal distribution in each state. This is equivalent to the multivariate model presented above, but with the correlation parameter ρ_{12k} fixed to zero. We use the observation distribution `norm`, and choose starting parameter values as we did before. Fitting this model is around 6 times quicker than the multivariate formulation; the difference is inconsequential here, but could become a limitation of multivariate models for large data sets or more complex model formulations.

```r
# Observation model for two univariate normal dists
par0 <- list(KO = list(mean = c(0, 0), sd = c(0.5, 2)),
             PEP = list(mean = c(0, 0), sd = c(0.5, 2)))
dists <- list(KO = "norm", PEP = "norm")
obs2 <- Observation$new(data = data,  n_states = 2,
                        dists = dists, par = par0)

# Fit model
hmm2 <- HMM$new(obs = obs2, hid = hid)
hmm2$fit(silent = TRUE)

# Show estimated parameters
lapply(hmm2$par(), round, 3)
```

```
$obspar
, , 1

         state 1 state 2
KO.mean    0.050  -0.120
KO.sd      0.759   2.380
PEP.mean   0.050  -0.061
PEP.sd     0.747   2.287



$tpm
, , 1

        state 1 state 2
state 1   0.947   0.053
state 2   0.190   0.810
```

The estimated parameters are very similar in both states, with the exception that no correlation parameters were estimated in this case. We can also check that the two models return fairly similar state classifications using the `$viterbi()` method. Around 93.7% of observations were classified identically by the two models.

```r
states_hmm1 <- hmm1$viterbi()
states_hmm2 <- hmm2$viterbi()
length(which(states_hmm1 == states_hmm2))/nrow(data)
```

```
[1] 0.9368542
```

The two models seem to capture similar patterns in the data, i.e., periods of low and high variability. However, the AIC indicates that the multivariate normal model (which accounts for correlation between the variables) is a much better fit, even accounting for the additional complexity.

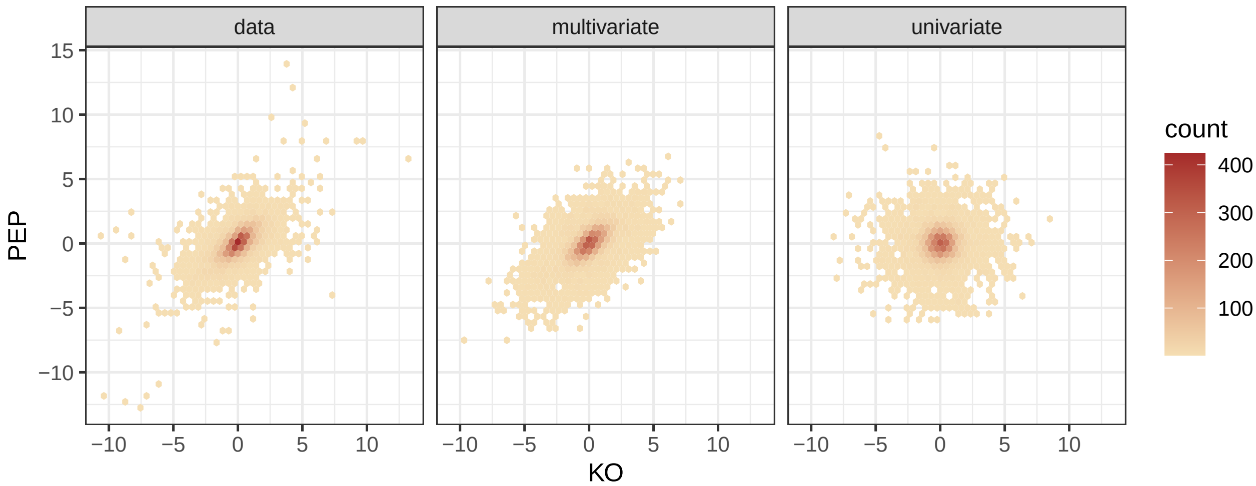
```r
AIC(hmm1, hmm2)
```

```
     df      AIC
hmm1 13 32933.96
hmm2 11 35653.07
```

We can also simulate from the two fitted models to check which features of the data are not captured appropriately. We simulate a time series of the same length as the data set from each model, and plot a heatmap of the simulated points. Contrasting this with the observed points, it is clear that the multivariate model captures the correlation well (whereas the univariate model does not, as expected).

```r
# Simulate from multivariate normal dist
sim1 <- hmm1$simulate(n = nrow(data), silent = TRUE)
sim1 <- cbind(sim1, do.call(rbind, sim1$z))
colnames(sim1)[3:4] <- names

# Simulate from two univariate normal dists
sim2 <- hmm2$simulate(n = nrow(data), silent = TRUE)

# Data frame for plots
df <- data.frame(KO = c(data$KO, sim1$KO, sim2$KO),
                 PEP = c(data$PEP, sim1$PEP, sim2$PEP),
                 group = rep(c("data", "multivariate", "univariate"),
                             each = nrow(data)))

ggplot(df, aes(KO, PEP)) +
    geom_hex(bins = 50) +
    facet_wrap("group") +
```

```r
  coord_equal() +
  scale_fill_gradient(low = "wheat", high = "brown")
```



References

Ryan, Jeffrey A., and Joshua M. Ulrich. 2024. *Quantmod: Quantitative Financial Modelling Framework*. https://CRAN.R-project.org/package=quantmod.