Financial Risk Forecasting
Seminar Seminar 7

Jon Danielsson
London School of Economics

Version 4.0 August 2024

7 Multivariate volatility

In this seminar, we will use the `Returns.RData` file that we created in last session. We focus on coding the multivariate GARCH with the `rmgarch` package. You need to install both GARCH packages before proceeding. Please review the theory of GARCH that we discussed in the lecture.

7.1 The plan for this week:

1. Introduce multivariate volatility models
2. Build a bivariate EWMA model
3. Run DCC models with different specifications

4. Compare models

7.2 Loading data and libraries

Load the necessary libraries.

```
library(rmgarch)
```

```
load('Returns.RData')
```

7.3 Multivariate EWMA Volatility

We need to use `as.matrix` because `Returns` is a `data frame` and we cannot do matrix algebra with data frames, so have to convert it.

```
y=as.matrix(Returns[,c("JPM","MCD")])
```

We then do the setup. Note the results go into a three column matrix.

```
dim(y)
EWMA = matrix(nrow=dim(y)[1],ncol=3)
lambda = 0.94
```

```
S = cov(y)
S
```

Note how we then can take the two diagonal elements of a matrix, and one of the off diagonal elements and put them into a row. What are these three numbers a measure of?

```
EWMA[1,] = c(S)[c(1,4,2)]
EWMA[1:2,]
```

7.3.1 Matrix multiplication

We have to use matrix multiplication to get the outer product of each day's returns. See the relevant section in the notebook For the details, below you see an overview.

We get a two by two matrix by.

```
i=2
y[i-1,] %*% t(y[i-1,])
```

If instead, we do

```
i=2
y[i-1,] %*% y[i-1,]
```

or

```
i=2
t(y[i-1,]) %*% y[i-1,]
```

We get the inner product, that is, a scaler, which is not what we want.

Then it is straightforward to run the EWMA loop.

```
for (i in 2:dim(y)[1]){
  S = lambda*S+(1-lambda)*
      y[i-1,] %*% t(y[i-1,])
  S
  EWMA[i,] = c(S)[c(1,4,2)]
}
```

We can use the EWMA matrix of variances and covariances to get the conditional correlations.

```
rhoEWMA = EWMA[,3]/sqrt(EWMA[,1]*EWMA[,2])
```

7.4 DCC

The CCC and DCC models give you an alternative.

First set up the specification of the univariate models. In this case we make each the same, but in general, it is possible to make them different.

```
# Create the univariate specification
uni_spec = ugarchspec(
```

```
  variance.model = list(
      garchOrder = c(1,1)),
  mean.model = list(
      armaOrder = c(0,0),
      include.mean = FALSE)
)
mspec = multispec(replicate(2, uni_spec))
mspec
```

Now we proceed to create the specification for the DCC model.

```
spec = dccspec(
  uspec = mspec,
  dccOrder = c(1,1),
  distribution = "mvnorm"
)
spec
```

Now we can proceed to fit the specification to the data:

```
res = dccfit(spec, data = y)
res
names(res@model)
res@mfit$matcoef
res@mfit$llh
```

We can plot the result

```
plot(res)
```

The conditional variance matrix is a **three dimensional** matrix called `H`.

```
H = res@mfit$H
dim(H)
H[,,1]
```

We can extract the conditional correlation in two ways. One is computing it from H:

```
rhoDCC = H[1,2,] / sqrt(H[1,1,]*H[2,2,])
```

7.5 Compare EWMA and DCC

Since we extracted the correlation vector for each method, we can plot them and compare the result.

```
par(mar=c(2,4,2,0))
matplot(cbind(rhoEWMA,rhoDCC),
  type='l',
  bty='l',
  lty=1,
  col=c("green","blue"),
  main="EWMA and DCC correlations for JPM and C",
```

```
  ylab="Correlations",
  las=1
)
legend("bottomright",
  legend=c("EWMA","DCC"),
  lty=1,
  col=c("green","blue"),
  bty='n'
)
```

7.6 Recap

In this seminar we have covered:

- Making miltivariate volatility models

Some new functions used:

- `multispec()`
- `replicate()`
- `dccspec()`
- `dccfit()`

7.7 Optional exercises

1. Run a DCC model for all stocks in the sample and compare to EWMA for the same.
2. Write a function to implement EWMA, where on eof the arguments in the stocks you want to use and another λ.