Risk & Asset Allocation
Homework for Week 5

John Dodson

October 1, 2014

A solution to these problems is due at the beginning of the next session, which is 5:30 PM on Wednesday, October 8, our last session this semester.

The GJR-GARCH(1,1) model for log-returns x_t can be written as

$$\begin{aligned}
\text{“residuals”} \quad & \epsilon_t \triangleq x_t - \mathrm{E}\left[X_t | \mathcal{F}_{t-1}\right] = x_t - \mu + \tfrac{1}{2} h_t \\
\text{“conditional variances”} \quad & h_t \triangleq \operatorname{var}\left[X_t | \mathcal{F}_{t-1}\right] = \omega + \beta h_{t-1} + \tfrac{1}{2}\left(\alpha^+ + \alpha^- + \left(\alpha^+ - \alpha^-\right) \operatorname{sgn} \epsilon_{t-1}\right) \epsilon_{t-1}^2 \\
\text{“devolatized residuals”} \quad & z_t \triangleq \frac{\epsilon_t}{\sqrt{h_t}}
\end{aligned}$$

for parameters μ, $\omega > 0$, $\alpha^+ > 0$, $\alpha^- > 0$, and $\beta > 0$ satisfying the constraint $\alpha^+ + \alpha^- < 2 - 2\beta$.

Problems

Let us again work with FB (Facebook) and GOOGL (Google) common equity daily closing price time series for two years through September 24.

1. Evaluate the maximum likelihood estimates of the model parameters μ, ω, β, α^+, α^- for each stock. (**6 points)**
2. Forecast the conditional variance of each stock for September 25. (**2 points)**
3. Evaluate the Kendall's rank-order correlation estimate of the devolatized residuals. (**2 points)**

Solution

In MATLAB, it is convenient to bundle the various outputs of the GARCH model into an M-file function. The main output should be the log-likelihood function, since this will be the objective of the optimization for the MLE fit. To avoid also passing back the conditional variances (h_t), it is useful if this function also calculates the forecasts and the de-volatilized residuals. This may seem wasteful—because for the vast majority of calls to this function these outputs will be discarded—but it adds only a tiny amount of extra processing. In any case, the entire script described in this solution[1] runs in 0.2 seconds on my old MacBook Pro.

[1] except for downloading the data

```
function [logLike foreVar foreMean stdResid]=GJRGARCH(x,theta,logLike)
% evaluate the GJR-GARCH timeseries model
% 'theta=[mu omega beta alphaUp alphaDown]' are model parameters
% input arg 'logLike' is returned in output if parameters are out-of-bounds
% assumes that 'x(1)' is the oldest log-return
if nargin==2
    logLike=NaN;
end
foreVar=NaN;
foreMean=NaN;
stdResid=nan(size(x));
if (min(theta(2:end))>0&&theta(4)+theta(5)<2-2*theta(3))||...
        (theta(2)>0&&theta(3)==0&&theta(4)==0&&theta(5)==0) % GBM
    resi=nan(size(x));
    vari=nan(size(x));
    for i=1:length(x)+1
        if i==1
            foreVar=theta(2)/(1-theta(3)-(theta(4)+theta(5))/2);
        else
            foreVar=theta(2)+theta(3)*vari(i-1)+(theta(4)+theta(5)...
                +(theta(4)-theta(5))*sign(resi(i-1)))/2*resi(i-1)^2;
        end
        foreMean=theta(1)-foreVar/2;
        if i<=length(x)
            vari(i)=foreVar;
            resi(i)=x(i)-foreMean;
        end
    end
    logLike=-sum(resi.^2 ./vari+log(2*pi*vari))/2;
    stdResid=resi./sqrt(vari);
end
```

It might seem appealing to implement this as a recursive function to avoid the explicit for-loop, but this seems to be less efficient in MATLAB, probably because of expensive call stack management.

There are several things to note in this implementation. The case of h_1 has special handling based on the assumption that $h_0 = \mathrm{E}\,\epsilon_0^2 = \sigma^2$, the unconditional variance. Also, the parameter out-of-bounds condition is relaxed slightly to allow for the geometric Brownian motion solution ($\beta = \alpha^+ = \alpha^- = 0$) as a formal solution. Furthermore, this implementation allows for a third optional argument representing the default return value of the log-Likelihood for out-of-bounds parameters. It is useful to be able to manage this to influence how `fminsearch()` behaves near this boundary.

To solve the problem, first of course you must load the historical data. As we did last week,

```
raw=yahoo_prices({'FB','GOOGL'},'2012-09-21','2014-09-24'); % start adjusted for weekend
tsc=resample(raw,raw.Time(~isnan(raw.FB.Data))); % remove holidays
ret.FB=diff(log(tsc.FB.Data));
ret.GOOGL=diff(log(tsc.GOOGL.Data));
```

This yields two sets of 503 consecutive daily log-returns, adjusted for splits and dividends.

In fitting the model parameters, note that in using a local numerical optimizer, where you start your search may matter to the solution. I suggest starting with the GBM solution. You can copy forward your solutions from last week, or you can re-produce them here.

Here is an example with the Facebook series.

```
theta_GBM.FB=fminsearch(@(theta)-GJRGARCH(ret.FB,[theta 0 0 0]),[0 var(ret.FB)]);
[logLike_GBM.FB foreVar_GBM.FB foreMean_GBM.FB]=GJRGARCH(ret.FB,[theta_GBM.FB 0 0 0]);
```

While it does not really matter, notice that I started the search for the GBM parameters with zero drift and squared volatility equal to the sample variance.

One you have these values, you can instruct the optimizer in the next step to use this solution as a worst case for the more general optimization.

```
theta.FB=fminsearch(@(theta)-GJRGARCH(ret.FB,theta,logLike_GBM.FB),[theta_GBM.FB 0 0 0]);
[logLike.FB foreVar.FB foreMean.FB devolResid.FB]=GJRGARCH(ret.FB,theta.FB);
```

Results for both stocks are tabulated in Table 1.

	Facebook		Google	
MLE results	GBM	GJR-GARCH	GBM	GJR-GARCH
$\hat{\mu}$	0.00285	0.00285	0.00107	0.00141
$\hat{\sigma}^2$ / $\hat{\omega}$	0.000795	0.000795	0.000186	0.000145
$\hat{\beta}$		0		8.92E-08
$\hat{\alpha}^+$		0		0.206
$\hat{\alpha}^-$		0		0.474
$\mathrm{E}\left[X_{T+1}\vert\mathcal{F}_T\right]$	0.00245	0.00245	0.000973	0.00133
$\mathrm{var}\left[X_{T+1}\vert\mathcal{F}_T\right]$	0.000795	0.000795	0.000186	0.000169
$\log f_{X\vert\theta}(x)$	1080.9	1080.9	1446.0	1451.2

Table 1: Timeseries model parameters for daily log-returns from September 21, 2012 to September 24, 2014.

Note that for Facebook, the optimizer could not improve on the GBM model based on the data over this time period; but for Google GJR-GARCH does provide a somewhat better fit.

I calculated the rank-order concordance the same way as in the last assignment,

```
pairs=0;
conc=0;
for i=1:length(devolResid.FB)-1
    for j=i+1:length(devolResid.FB)
        pairs=pairs+1;
        conc=conc+sign(devolResid.FB(i)-devolResid.FB(j))...
            *sign(devolResid.GOOGL(i)-devolResid.GOOGL(j));
    end
end
tau=conc/pairs;
rho=sin(tau*pi/2);
```

and I get very similar results, $\tau \approx 0.279$ with corresponding pseudo-correlation 0.424.