R による時系列分析 4[†]

1. GARCH モデルを推定する

1.1 パッケージ“rugarch”をインスツールする。

パッケージとは通常の R には含まれていない、追加的な R のコマンドの集まりのようなものである。R には追加的に 600 以上のパッケージが用意されており、それぞれ分析の目的に応じて標準の R にパッケージを追加していくことになる。

インターネットに接続してあるパソコンで R を起動させ、「パッケージ」→「パッケージのインストール...」→「Japan(Tokyo)」→「rugarch」→「OK」とクリックする。すると(いろいろとインストールの途中経過が表示されて)パッケージのインストールが自動的に終わる(ここは時間がかかる場合がある)。上記の作業は次回以降はやる必要はないが、以下の作業は R を起動するたびに毎回やる必要がある。次にインストールしたパッケージを使うためにコマンドウィンドウ (R Console) に

```
> library(rugarch)
```

と入力し(library()関数はインスツールしたパッケージを読み込むための関数) 再びコマンドウィンドウ上にいろいろと表示され(ここも全て終わるのに時間がかかる場合がある)パッケージ rugarch を使用できる様になる。

データとして topixdata.txt を使用する。topixdata.txt ファイルを適当なディレクトリに保存し、ファイル→作業ディレクトリの変更によって topixdata.txt ファイルを保存したディレクトリを指定し、OK をクリック。指定したら、実際にファイルがあるか dir()関数によって確認する。

```
> dir()
```

表示されたディレクトリ内のファイルに topixdata.txt ファイルがあるが確認できたら、次に topixdata.txt を read.table()関数で読み込む。

```
> topixdata=read.table("topixdata.txt",header=T,skip=1)
```

topixdata の最初の 5 行を見るには

```
> head(topixdata,5)
```

と入力する。結果は

[†] この資料は私のゼミおよび講義で R の使用法を説明するために作成した資料です。ホームページ上で公開しており、自由に参照して頂いて構いません。ただし、内容について、一応検証してありますが、間違いがあるかもしれません。間違いがあった場合でもそれによって生じるいかなる損害、不利益について責任は負いかねますますのでご了承下さい。

```
    date   topixrate
1 Feb-75  8.2422041
2 Mar-75  4.4650419
3 Apr-75  2.2301301
4 May-75  2.7930824
5 Jun-75 -0.5810874
```

と表示される。今、必要なのは `topixrate` のデータだけなので

```
> topixrate=topixdata$topixrate
```

として取り出しておく。`topixrate` に対して GARCH モデルを推定する。

1.2 ugarchspec および ugarchfit 関数を用いて推定する

まず `ugarchspec()` 関数でどのような GARCH モデルを推定するかを**定式化**する。例えば誤差項が GARCH(1, 1)モデルに従い、かつ、平均はただの定数、つまり

$$r_t = \mu + \sqrt{h_t}\varepsilon_t, \quad \varepsilon_t \sim i.i.d.N(0, 1)$$
$$h_t = \omega + \beta_1 h_{t-1} + \alpha_1 r_{t-1}^2$$

というモデルを推定するとする(ここで r_t が `topixrate` の t 時点のデータを表すとする)。この定式化するのに

```
> garch11=ugarchspec(variance.model=list(model="sGARCH",garchOrder=
+ c(1,1)), mean.model=list(armaOrder=c(0,0), include.mean=TRUE))
```

とする。明らかにコマンドが終わっていない所でエンターを押すと改行し“ + ” が表示される。上の場合は「`garchOrder=`」まで入力した時点でエンターを押して改行している。コマンドが長くなる場合は適当なところで改行して続きを打ち込んだ方が見やすい。

上記のコマンドにおいて、「`model="sGARCH"`」の部分は推定するボラティリティモデルを指定している。(sGARCH は standard GARCH の略)例えば EGARCH モデルや GJR モデルを推定したい場合はここを「`model="eGARCH"`」や「`model="gjrGARCH"`」とすればよい。「`garchOrder=c(1,1)`」の部分で GARCH モデルの次数を指定している。「`c(1, 1)`」は GARCH(1, 1) モデルである。たとえば GARCH(1, 2) モデルを推定したい場合は、この部分を「`garchOrder=c(1, 2)`」とすればよい。また平均の部分を ARMA モデルによってモデル化したい場合は「`mean.model=list(armaOrder=c(0,0), include.mean=TRUE)`」の部分を変更してやればよい。例えば定数項のない ARMA(2, 1) モデルによって平均をモデル化したい場合は「`mean.model=list(armaOrder=c(2,1), include.mean=FALSE)`」としてやればよい。この時、(GARCH の次数は(1,1)とすると)

$$r_t = \phi_1 r_{t-1} + \phi_2 r_{t-2} + u_t + \theta_1 u_{t-1}, \quad u_t = \sqrt{h_t}\varepsilon_t \quad , \varepsilon_t \sim i.i.d.N(0, 1)$$
$$h_t = \omega + \beta_1 h_{t-1} + \alpha_1 u_{t-1}^2$$

とうモデルを推定する事になる。「include.mean」を TRUE にするか FALSE にするかよって定数項を含む(TRUE)か含まない(FALSE)を指定する。

以上のコマンドによって garch11 にモデルの定式化が記録される(この名前は garch11 でなくてもなんでもよい)。試しに

```
> garch11
```

と入力すると、どのような GARCH モデルを定式化したかが出力される。

次に、この定式化に従う GARCH モデルを**推定する**。ugarchfit( )関数を使う。推定結果に garch11result という名前を付けるとすると(この名前もなんでもよい)

```
> garch11result=ugarchfit(spec=garch11,data=topixrate)
```

と入力し推定する。推定結果は

```
> garch11result
```

と入力すれば表示される。推定値は

```
…
Optimal Parameters
------------------------------------
       Estimate  Std. Error  t value Pr(>|t|)
mu      0.71121    0.181305   3.9228 0.000088
omega   0.62851    0.356089   1.7650 0.077559
alpha1  0.18762    0.058799   3.1909 0.001418
beta1   0.78532    0.062060  12.6542 0.000000

Robust Standard Errors:
       Estimate  Std. Error  t value Pr(>|t|)
mu      0.71121    0.220782   3.2213 0.001276
omega   0.62851    0.430509   1.4599 0.144313
alpha1  0.18762    0.065919   2.8462 0.004424
beta1   0.78532    0.068817  11.4116 0.000000

LogLikelihood : -1001.735
…
```

の青字の部分である。mu が μ の推定値、omega が ω の alpha1 が α_1 の beta1 が β_1 の推定値である。

次は ARCH(6)モデルを推定してみよう。先ほどと同様に平均は定数とする。まず ARCH(6)の定式化をする。

```
> arch6=ugarchspec(variance.model=list(model="sGARCH",garchOrder=
+ c(6,0)),mean.model=list(armaOrder=c(0,0),include.mean=TRUE))
```

次にこの定式化に沿って推定をする。

```
> arch6result=ugarchfit(spec=arch6,data=topixrate)
```

最後に結果を見るために

```
> arch6result
```

と入力する。

`ugarchspec()`関数の `garchOrder` を変更する事によっていろいろなオーダーの GARCH モデルが推定できる。また `include.mean=FALSE` とすれば μ は推定されない。

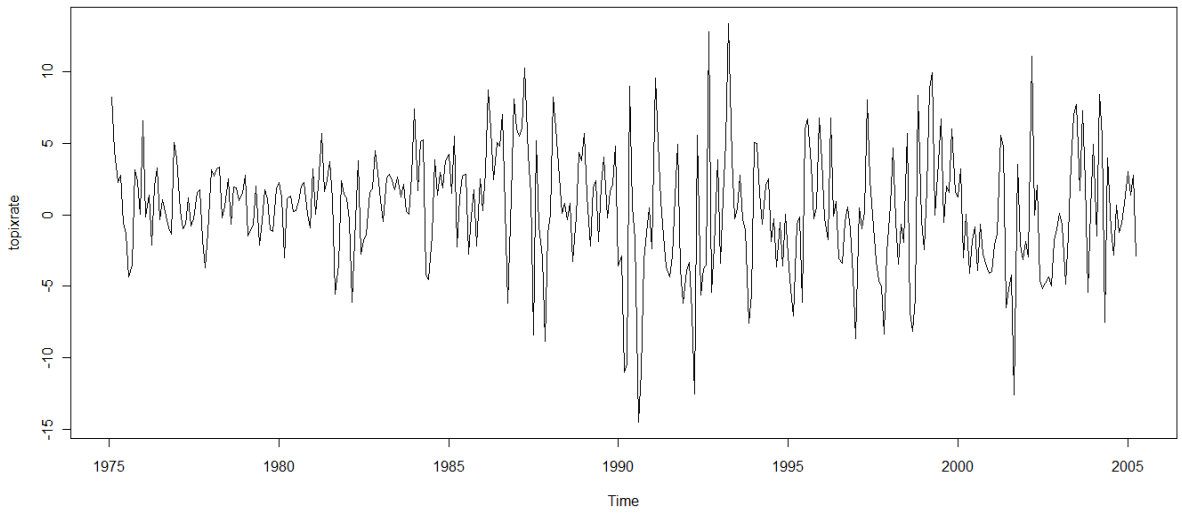
2. ボラティリティの定式化 に外生変数(ダミー変数)を入れる。

あるイベントが起きた時にそれがボラティリティにどのような影響を与えるかを見たいようなときはボラティリティの定式化の部分にそのイベントによる影響があったと思われる期間だけダミー変数を入れてみるという方法が考えられる。

TOPIX の収益率に対して、`plot()`関数でグラフを書いてみると

```
> topixrateTS=ts(topixrate, start=c(1975,2), frequency=12)
> plot(topixrateTS, type="l")
```

とすると(`ts()`データを関数は時系列オブジェクトに直す関数、詳しくは他の資料参照)



という図が出力される。これを見ると 1980 年代の後半から TOPIX の収益率は変動が大きくなったように見える。歴史的には 1988 年に TOPIX の先物取引が 1989 年にはオプション取引が導入されている。これらのイベントが TOPIX のボラティリティにどのような影響が見たかを調べるために次のようなモデルを推定してみよう。(ここでは先物の導入に関してのみ調べる)

$$r_t = \mu + \sqrt{h_t}\varepsilon_t, \quad \varepsilon_t \sim i.i.d.N(0, 1)$$
$$h_t = \omega + \beta_1 h_{t-1} + \alpha_1 r_{t-1}^2 + \delta dum_t,$$

ここで dum_t は 1988 年の 9 月より前は dum_t =0, 以降は dum_t =1 を取る変数だとする。δ の係数が有意に正であれば、導入後にボラティリティが恒常的に上がったという事になる。この変数を取り入れたデータ topixdata2.txt を読み込もう。topixdata2.txt を作業ディレクトリに保存し、

```
> topixdata2=read.table("topixdata2.txt",header=T)
```

によって読み込む。最初の 7 行は

```
> head(topixdata2, 7)
date  topixrate dum
1 Feb-75  8.2422041   0
2 Mar-75  4.4650419   0
3 Apr-75  2.2301301   0
4 May-75  2.7930824   0
5 Jun-75 -0.5810874   0
6 Jul-75 -1.3114237   0
7 Aug-75 -4.3147168   0
```

となっている。必要なのはダミー変数(dummy variable)についてのデータだけなので(topixrate のデータは先ほどと同じ)

```
> dum=topixdata2$dum
```

と入力して取り出す。さらにこの dum という変数はこのままだと R はベクトルとして認識しているのだが、のちに ugarchspec() 関数で外生変数として扱う際には R には行列として認識してもらわないといけないので(ここは何の事かわからないと思うが、とりあえずこのようにする)、

```
> dum = as.matrix(dum)
```

として R に行列として認識させる。次にこのダミー変数を GARCH モデルのボラティリティの中に入れたいので ugarchspec() 関数によって

```
> garch11wd=ugarchspec(variance.model=list(model="sGARCH",
+ garchOrder=c(1,1),external.regressors=dum),mean.model=
+ list(armaOrder=c(0,0), include.mean=TRUE))
```

とする(wd は with dummy の略)（青色にしたのは見やすいようにしただけで実際にはもちろん青色にならないし、する必要もない事に注意）。あとは先ほどと同じように

```
> garch11wdresult=ugarchfit(spec=garch11wd,data=topixrate)
```

によって推定する。結果は

```
> garch11wdresult
```

によって出力され、例えば

```
…
Optimal Parameters
------------------------------------
       Estimate  Std. Error  t value Pr(>|t|)
mu      0.71122    0.182457   3.8980 0.000097
omega   0.62848    0.375690   1.6729 0.094352
alpha1  0.18762    0.077130   2.4325 0.014995
beta1   0.78532    0.068334  11.4924 0.000000
vxreg1  0.00000    0.512826   0.0000 1.000000

Robust Standard Errors:
       Estimate  Std. Error  t value Pr(>|t|)
mu      0.71122    0.227581   3.1251 0.001777
omega   0.62848    0.504041   1.2469 0.212439
alpha1  0.18762    0.092323   2.0322 0.042132
beta1   0.78532    0.084396   9.3052 0.000000
vxreg1  0.00000    0.544069   0.0000 1.000000

LogLikelihood : -1001.735
…
```

のように出力される。vxreg1 というのが dum の係数である。これを見ると先ほどのグラフの直観とは異なり、先物市場の導入はボラティリティ変動への恒常的な効果をもっていない事になる。

恒常的な効果があるというは効果を過大に評価していたのかもしれない。では次に一時的な効果があったかどうかを見る事にする。このためダミー変数を導入直後にのみしばらく入れてその係数の値によって見ることにしよう。導入後 1 年間のみダミー変数を入れてみる(1 年間というのはかなり恣意的な決め方ではあるが)。つまり d_t=1 (1988 年の 9 月から 1999 年の 8 月まで)、$d_t = 0$ (それ以外の期間)として、先ほどと同じモデルを推定する。ここでは topixdata3.txt にある dumtemp というダミー変数を使う。読み取り方、定式化、推定の仕方等は先程同じなので(dum を dumtemp にするだけ)推定結果だけ見てみよう。

```
…
Estimate  Std. Error  t value Pr(>|t|)
mu      0.71122    0.183038   3.8857 0.000102
omega   0.62848    0.377483   1.6649 0.095926
alpha1  0.18762    0.061517   3.0499 0.002289
beta1   0.78532    0.065185  12.0476 0.000000
vxreg1  0.00000    1.618795   0.0000 1.000000

Robust Standard Errors:
       Estimate  Std. Error  t value Pr(>|t|)
mu      0.71122     0.23183   3.0678 0.002156
omega   0.62848     0.48391   1.2988 0.194028
alpha1  0.18762     0.06879   2.7274 0.006383
beta1   0.78532     0.07496  10.4765 0.000000
```

```
vxreg1   0.00000    1.94969   0.0000 1.000000

LogLikelihood : -1001.735
```
…

これを見てみると短期的にも何の影響もなかったという事になる。

また平均の部分にダミー変数を入れることもできる。ugarchspec()関数の中の mean.model の部分を

```
mean.model=list(armaOrder=c(0,0), include.mean=TRUE,
external.regressors = dum)
```

のようにすればよい。さらにここで armaOrder=c(2,1)などとすれば、mean は ARMA(2,1)モデルであり、その誤差項が GARCH モデルに従うようなモデルを推定する事もできる。条件付き分散の方にはダミー変数を入れないとすると、この場合

$$(1-\phi_1 L-\phi_2 L^2)r_t = \mu + \delta\, dum_t + u_t + u_{t-1},\ u_t = \sqrt{h_t}\,\varepsilon_t,\ \varepsilon_t \sim i.i.d.N(0,1)$$

$$h_t = \omega + \beta_1 h_{t-1} + \alpha_1 u_{t-1}^2,$$

というモデルを推定していることになる。

さらに同様の事を GJR モデルで行うには ugarchspec()関数の中で variance.model の部分を

```
variance.model=list(model="gjrGARCH",garchOrder=c(1,1),
external.regressors=dum)
```

とすればよく、EGARCH モデルで行うには ugarchspec()関数の中で variance.model の部分を

```
variance.model=list(model="eGARCH",garchOrder=c(1,1),
external.regressors=dum)
```

などとすればよい(EGARCH の場合は garchOrder の部分は GARCH モデルとは少し意味が異なる)。より詳しくは help(ugarchspec)を参照の事。