CAViaR モデルを用いたボラティリティの予測

東京理科大学 工学部 経営工学科

鈴木 直明

平成 17 年 11 月 7 日

目次

1. はじめに

近年，金融工学では，他の工学の分野で広く使用されていた時系列解析の手法を取り入れて，「金融時系列解析」という分野が発展した[5].

金融時系列解析の発展の過程で，ボラティリティ・クラスタリング（株価の変動において，大きい変動があった後は，しばらく大きい変動が続き，小さい変動があった後は，小さい変動が続く現象のこと）の発見により，金融時系列の分野は急速な進歩を見せた.

このような背景の中で，金融時系列分析の中で最も重要なモデルの一つである GARCH（generalized autoregressive conditional hetroskedasticity：分散不均一）モデルは提唱された．GARCH モデルは株式市場の株価のボラティリティの予測に現在も広く使用されている．ボラティリティとは，株価の収益率の変動を表す指標である．ボラティリティは株価の収益率の時系列データの分散または標準偏差で与えられる．ボラティリティが大きいということは，株価の収益率が時の経過とともに大きく変化することを表し，ボラティリティが小さいということは，株価の収益率が時の経過とともに小さく変化することを表す．ボラティリティはあくまでも変動の大きさを表す指標なので，ボラティリティは単独で株価の収益率が上がったのか，それとも下がったのかといった情報は提供しない.

GARCH モデルはボラティリティの予測をする際に，分布を仮定する必要があることが知られている[1]．GARCH モデルによる予測は，仮定した分布が実際の分布とは異なっていたり，時間とともに実際の分布は変わっていってしまったりなどの原因で大きく外れてしまう危険にさらされている．そこで，本研究では，論文[1]で発表された，自己回帰という，過去の実測値を将来の予測のための変数として組み込んだものを使用して，ボラティリティの予測に分布の仮定を必要としない，CAViaR（conditional autoregressive VaR：条件付自己回帰 VaR）[2]モデルを用いて予測をする手法を用いてボラティリティを推測する.

2. モデル

2．1. GARCH(p,q)モデル

GARCH モデルは，標準偏差を求める式（式（1.3））の中に，標準偏差の自己回帰が組み込まれているところに特徴がある．株価の t 期の収益率を r_t とすると，GARCH モデルは次の式で表される.

$$r_t = \mu + \varepsilon_t \qquad (1.1)$$

$$\varepsilon_t = \sigma_t u_t \qquad (1.2)$$

$$\sigma_t^2 = \alpha + \sum_{i=1}^{p} \alpha_i \varepsilon_{t-i}^2 + \sum_{j=1}^{q} \beta_i \sigma_{t-j}^2 \qquad (1.3)$$

ただし

$\mu = E[r_t]$，　ε_t：誤差項，　σ_t：ボラティリティ　，

u_t：互いに独立で平均　0，分散 1の分布を仮定する

$\alpha, \alpha_i, \beta_i$：パラメータ

2．2.　GJR GARCH(Glosten=Jagannathan=Runkle GARCH)モデル

ボラティリティの変動は，収益率が上がった後よりも，収益率が下がった後のほうが大きい．この事実を GARCH モデルに反映させているのが GJR GARCH である．GARCH モデルの第 3 式（1.3）の条件付分散を次のように表現しなおしたモデルである．

$$\sigma_t^2 = \alpha + \sum_{i=1}^{p} \alpha_i \varepsilon_{t-i}^2 + \sum_{i=1}^{p} \gamma_i S_{t-i} \varepsilon_{t-i}^2 \sum_{j=1}^{q} \beta_i \sigma_{t-j}^2 \qquad (2)$$

S_{t-i} :もし$\varepsilon_{t-i} < 0$ならば$S = 1$，その他のとき$S = 0$となるダミー変数

γ_i :パラメータ

2．3.　CAViaR モデル

CAViaR（conditional autoregressive value at risk：条件付自己回帰 VaR）モデルは従来の与えられた確率のもとで，最低の収益率の値を分位点（$Q(\theta)$で表現する：数値を小さい順に並べたときに小さいほうから数えて総度数のθ%に当たる値）を用いて表現する VaR（value at risk）という手法を発展させたもので，過去の収益率の実現値を条件としたモデルである．

CAViaR は数種類紹介されているが，本研究では Asymmetric Slope CAViaR（以下 AS CAViaR）を用いる．

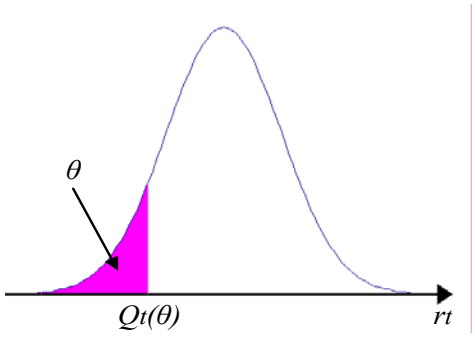

図 1：VaR の概念図

2．4.　AS CAViaR モデル

2．4．1.　AS CAViaR モデルの概要

AS CAViaR の特徴としては，株価の変動の予測をする際に，収益率のプラスの変動とマイナスの変動では，別の振る舞いをするように設計されている点が挙げられる．

AS CAViaR モデルは次の式で表現される．

$$Q_t(\theta) = \omega + \alpha Q_{t-1}(\theta) + \beta_1(\varepsilon_{t-1})^+ + \beta_2(\varepsilon_{t-1})^- \qquad (3)$$

ただし

$Q_t(\theta)$: t 時点における，確率 θ の分位点

$\omega, \alpha, \beta_1, \beta_2$: パラメータ

$(x)^+ = \max(x,0), \quad (x)^- = -\min(x,0)$

(3)式右辺の第 3 項と第 4 項の ε_t は t 時点における条件付期待値（*t-1* 時点までの情報に基づいて計算される期待値で $E(r_t \mid I_{t-1})$ と表現）を計算した上で，実測値と条件付期待値の差として次の式で定義される．

$$\varepsilon_t = r_t - E(r_t \mid I_{t-1}) \qquad (4)$$

AS CAViaR モデルの特徴は，このモデルよって，次の時点における収益率を予測する際，現時点で条件付期待値との差がプラスの（条件付期待値よりも実現値が上回る）ときと，マイナスの（条件付期待値よりも実現値が下回る）ときの振る舞いが異なるように設計されていることが挙げられる．特にその差がマイナスになるときのほうが，予測に大きく影響することが挙げられる[1].

2．4．2．　AS CAViaR モデルのパラメータ推定法

分位点回帰最小法[3]を用いる．分位点回帰は次の式で定義される．

$$\sum_{t \mid y_t \geq Q_t(\theta)} \theta \mid y_t - Q_t(\theta) \mid + \sum_{t \mid y_t < Q_t(\theta)} (1-\theta) \mid y_t - Q_t(\theta) \mid \qquad (5)$$

$t : 1, \cdots, t-1$

$\{y_t\}$: 実現値の集合

パラメータ推定用のデータを用いて，実際に分位点回帰を計算して，分位点回帰が最も小さくなるパラメータを使用する[3].

3.　事例研究

3．1.　分析対象データ

本研究で用いたデータの概要を以下に示す.

- 内容：JASDAQIndex 終値（日次）および，米国 S&PIndex の終値（日次）の時系列データ
- 期間：1997 年 4 月 1 日～2003 年 8 月 31 日
- いずれも 1997 年 4 月 1 日から 1001 営業日分のデータをモデルの学習用データとする.

3．2.　ヒストリカルボラティリティ

3．2．1.　ヒストリカルボラティリティの概要と利用意義

過去の株価の収益率の動きから変動度合いを表したもので，一般に年率で表現される.

本研究では，予測の結果とヒストリカルボラティリティを比較する．ヒストリカルボラティリティと比較する理由は，ヒストリカルボラティリティが実際のデータに基づいて計算されるもので，現実の市場の動向を反映していると考えられるためである.

3．2．2.　ヒストリカルボラティリティの算出法

1 年 250 営業日として，年率換算したヒストリカルボラティリティの算出方法は，次の通りである.

$$\left(\log\left(\frac{r_t}{r_{t-1}}\right)\right)^2 \times (250)^{\frac{1}{2}} \times 100 \quad (\%) \qquad (6)$$

3．2．3.　ヒストリカルボラティリティ計算結果

S-PLUS を用いてヒストリカルボラティリティを出力した結果を図 2 に示す(script は付録 B 参照).図 2 は横軸が営業日，縦軸がボラティリティを表しており，山が大きいほど，その時点で株価の収益率に大きな変化があったといえる．JASDAQ と S&P の縦軸を見てみると，JASDAQ のほうが最大値が大きいことが分かる．これは，JASDAQIndex の方が S&PIndex よりも株価の変動が大きい企業が多く含まれていることを表す.

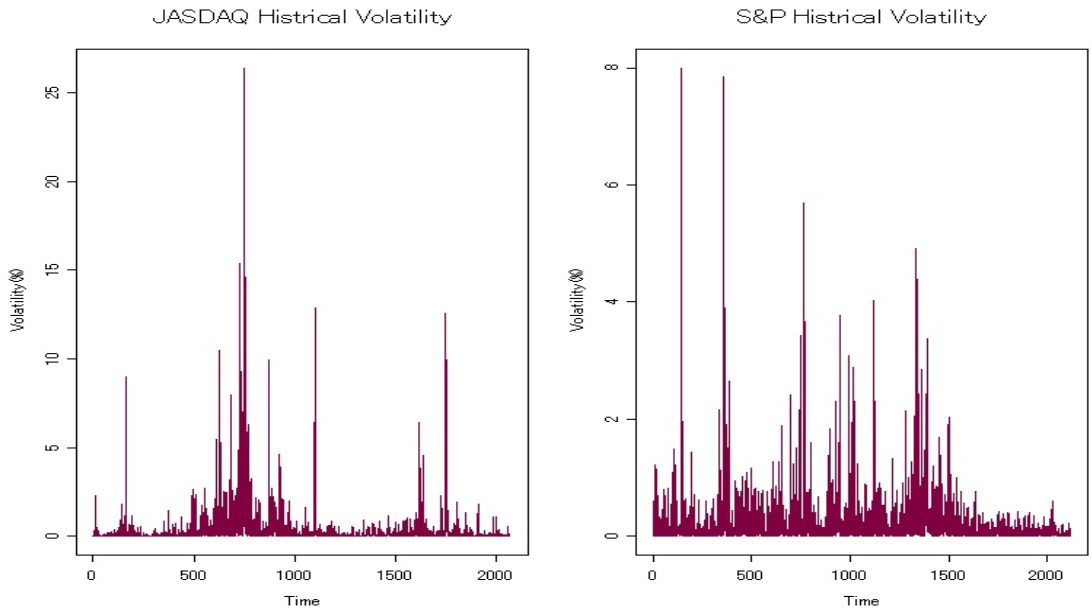

図 2：ヒストリカルボラティリティ

3. 3. GARCH 型モデルを用いたボラティリティの予測

3. 3. 1. GARCH モデルを用いたボラティリティの予測

S-PLUS によって, GARCH モデルを用いたボラティリティの予測を行った（script は付録 C 参照）. S-PLUS では GARCH モデルのパラメータの推定には garch という function を，GARCH モデルを用いた予測には simulate という function をそれぞれ用いる．GARCH モデルを用いた予測を S-PLUS を用いて行った理由は，元になるデータを時系列データに直すのが非常に容易に行える点，1 つデータセットを準備すれば様々なタイプの GARCH モデルを適用するのが容易な点，それに，出力したデータがきれいにグラフ化して表示できる点が挙げられる．function である garch を用いてモデルのパラメータを推定し，function である simulate を用いて，ボラティリティを予測した結果を出力したものを図 3 に示す．なお，縦軸第 1 軸はボラティリティの大きさを，単位を%として表している，この第 1 軸に関連するのが，緑線と赤線で，緑線がボラティリティの予測値，赤線がヒストリカルボラティリティである．これらの値は株価の収益率に、正であれ負であれ大きな変化があった時に，大きな値を示す．株価の収益率の分散であるため，値は必ず正である．縦軸第 2 軸は，黒線が収益率から収益率の平均を引いたものであり，この値が株価の収益率の実測値である．赤線のヒストリカルボラティリティは，黒線の実測値を元に計算されているので，赤線と黒線は互いに関連を持って推移しているのが見られる．例えば，100 日付近で，黒線が大きく下がっているので，この時期に株価の収益率が大きく下がったことが分かる．それに伴い，赤線が大きく上がっている，これは収益率が大きく（マイナスに）変化したため，ボラティリティが大きく上がったことを示している．緑線のボラティリティの予測値は，前半部分を大まかに観察すると，単純に減少しているように見えるが，詳細を観察すると細かい上下の変動を繰り返していることが分かる．

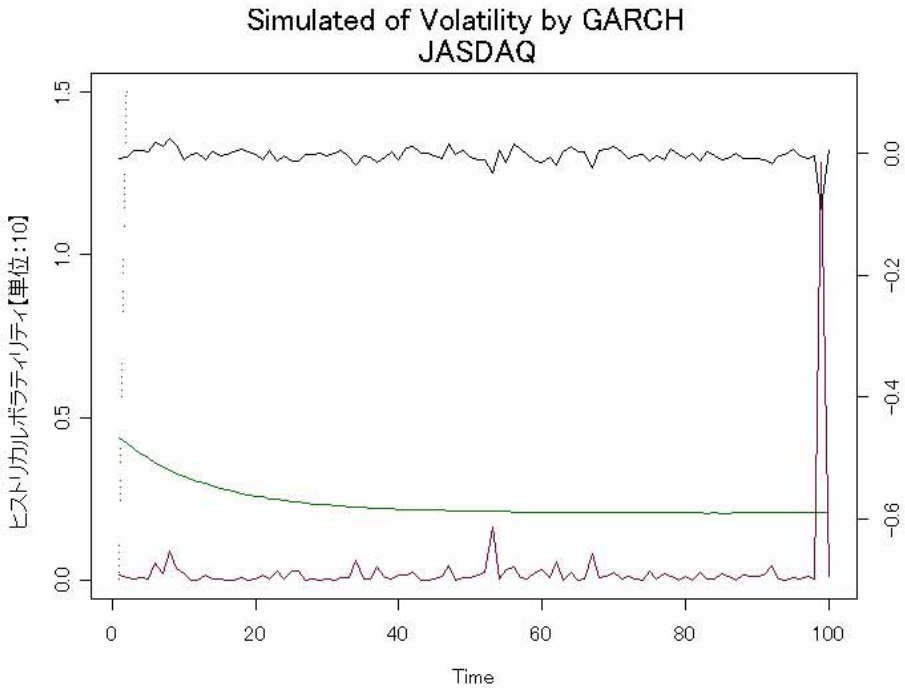

図 3. 1 ：GARCH を用いた JASDAQ のボラティリティの予測

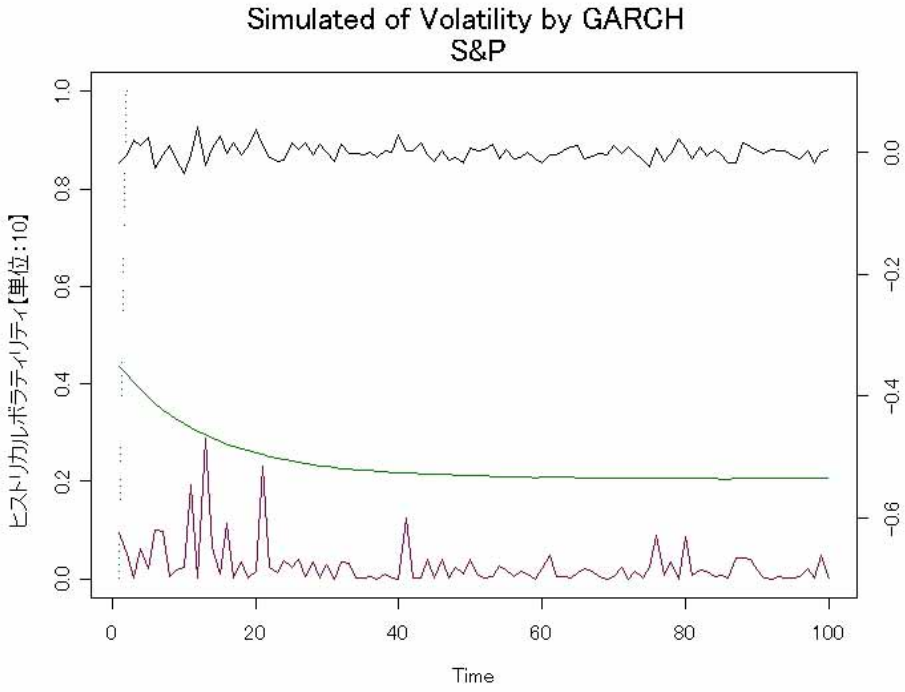

図 3. 2：GARCH モデルを用いた S&P のボラティリティの予測

3．3．2．　GJRGARCH モデルを用いたボラティリティの予測

S-PLUS によって，GJRGARCH モデルを用いてボラティリティの予測を行った（script は付録 C 参照）. S-PLUS では GJRGARCH モデルのパラメータの推定には tgarch という function を, GJRGARCH モデルを用いた予測には simulate という function をそれぞれ用いる．3．3．1．で S-PLUS を用いた理由でも触れたが, GJRGARCH の予測は, GARCH モデルの script を数行変更するだけで実現できる．function である tgarch を用いてモデルを推定し，function である simulate を用いてボラティリティを予測した結果を出力したものを図 4 に示す．なお，各線の説明は図 3 のものと同様である．出力した結果を GARCH モデルの出力と比較してみると，概形に大きな差異は見当たらない.

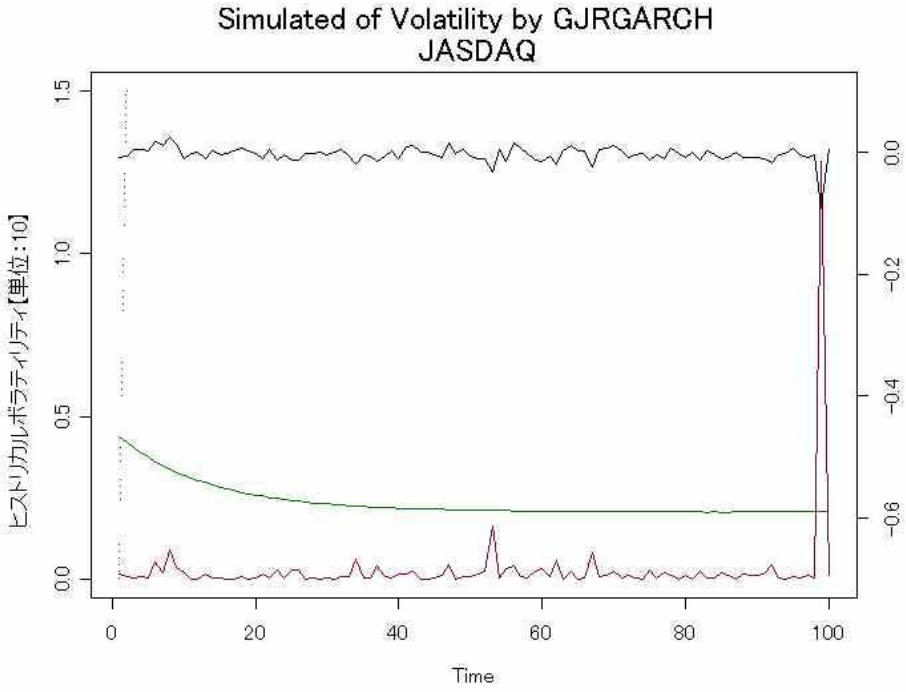

図 4．1：TGARCH を用いた JASDAQ のボラティリティの予測

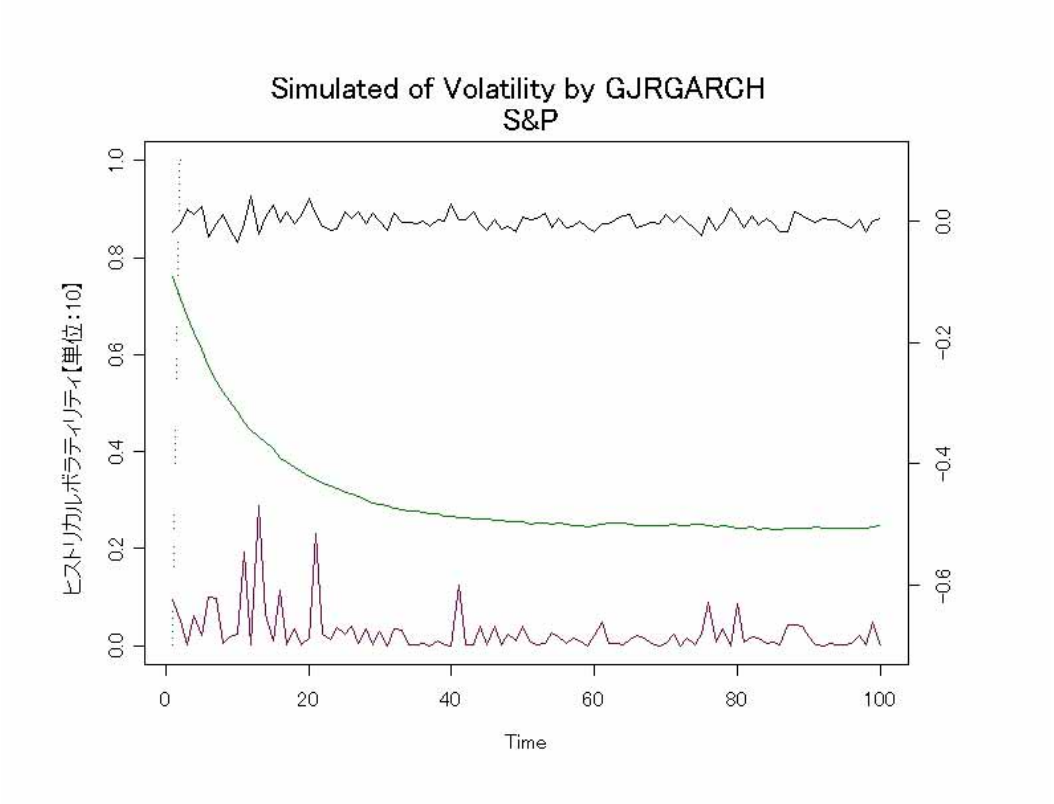

図 4．2：TGARCH を用いた S&P のボラティリティの予測

3．3．3．　寄与率

予測の精度のよさを寄与率を使って表現する．本研究では，ヒストリカルボラティリティを，株価の収益率のボラティリティの実測値と定義をした．モデルによって予測されたボラティリティの値とヒストリカルボラティリティの値の相関係数を計算し，相関係数の 2 乗を寄与率として，モデルの当てはまりのよさを検討する．モデルの寄与率を計算したところ，表 1 のような結果を得た（script は付録 C 参照）．寄与率が高い方がモデルの当てはまりはよいといえる．

表 1．1：GARCH モデルを用いた予測の寄与率

	10 日先まで予測	20 日先まで予測	100 日先まで予測
JASDAQ	27.4%	3.9%	0.3%
S&P	8.7%	0.0%	12.1%

表 1．2：GJRGARCH モデルを用いた予測の寄与率

	10 日先まで予測	20 日先まで予測	100 日先まで予測
JASDAQ	27.5%	3.5%	0.3%
S&P	8.2%	0.0%	11.6%

3．3．4．　寄与率に関する考察

まず, GARCH モデルと GJRGARCH モデルの寄与率を比較してみたところ, 大きく値は変わらないことが分かった．このことから今回，利用した 2 つのデータでは GARCH モデルを用いた予測と GJRGARCH モデルを用いた予測で，予測の精度は大きく変わらないことが分かる．

次に，寄与率を比べてみたところ，どちらのモデルを用いた予測でも，JASDAQ，S&P ともに短期間の予測の寄与率のほうが，長期間の予測の寄与率よりも大きいことが分かる．これは，GARCH 型モデルを用いた予測は短期間における予測にしか使えず，長期間の予測には使えないことを示している．

S&P の 100 日先の寄与率が高かった原因は特定できなかった．GARCH 型モデルは長期的な予測には向かないことは，論文[1]でも触れられているので，おそらくこの値は適当ではない．この点については今後検討する必要があると考える．

3．4．　CAViaR モデルを用いたボラティリティの予測

3．4．1．　AS CAViaR モデルのパラメータの推定

S-PLUS によって，AS CAViaR モデルを用いてボラティリティの予測を行うために，AS CAViaR に学習用データを用いてパラメータの推定を行う（script は付録 E 参照）．AS CAViaR モデルは caviar という function を用いる（caviar については付録 D 参照）. caviar にデータセットを入れて実行すると，リスト形式で出力される．リストの中には，$percentile，$beta，$theta，$model が含まれる．それぞれ，$percentile には求められたパラメータによって計算された分位点，$beta にはモデルの各パラメータ，$theta は設定した確率，$model は設定した CAViaR の種類が入っている．実際の株価の収益率と関数 caviar によって得られた 95%分位点，5%分位点をあわせて出力したものを図 5 に示す．なお，青線が実際の株価の収益率，緑線が 95%分位点，赤が 5%分位点である．95%分位点は実際の株価の収益率の上側に沿って，5%分位点は実際の株価の収益率の下側に沿って推移している様子が分かる．

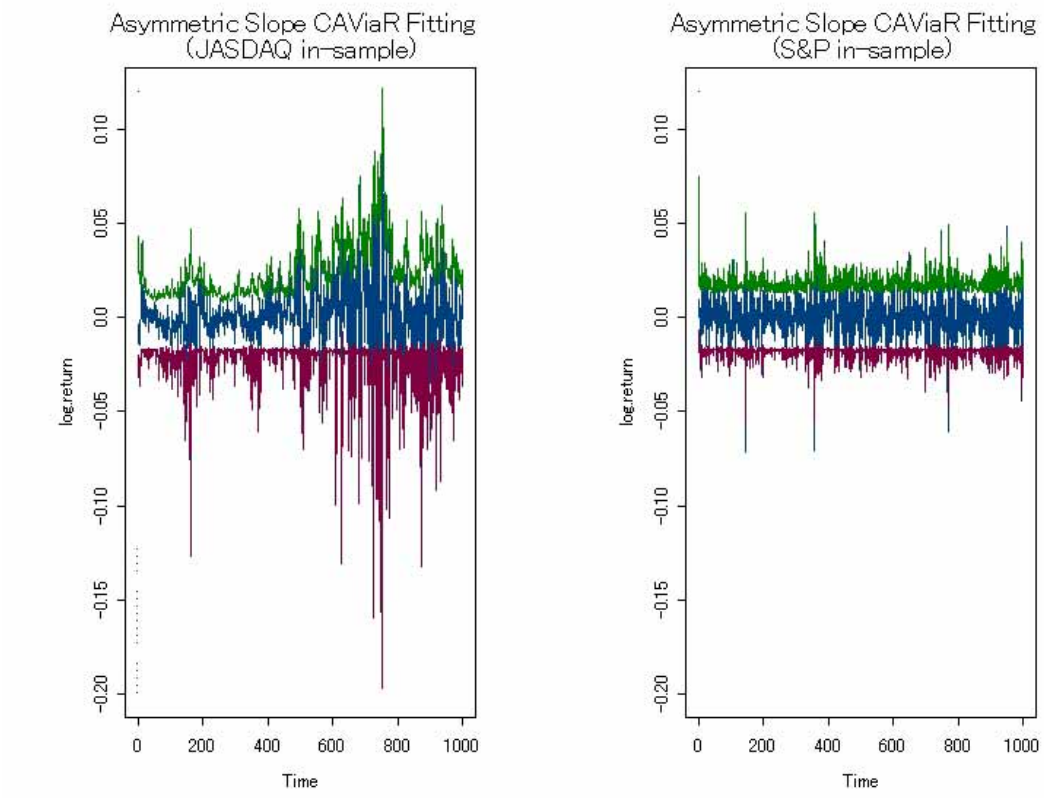

図 5：AS CAViaR モデルによって得られた学習データの分位点

3．4．2．　AS CAViaR を用いたボラティリティの予測

S-PLUS によって, AS CAViaR モデルを用いたボラティリティの予測を行う（script は付録 F 参照）．分位点によってボラティリティの平方根（σ）の近似を求める式は次の式で表される[1],

$$\sigma \approx \frac{Q(0.95) - Q(0.05)}{3.25} \qquad (7)$$

caviar によって出力された$beta を用いて，5%，95%の確率下の分位点を順次求めて，(7)式に代入していくことによって，ボラティリティの予測値が計算されていく．このようにして，ボラティリティを予測した結果を図 6 に示す．図 3 と同様，緑線がボラティリティの予測値，赤線がヒストリカルボラティリティ，黒線が株価の収益率の実測値を表す．

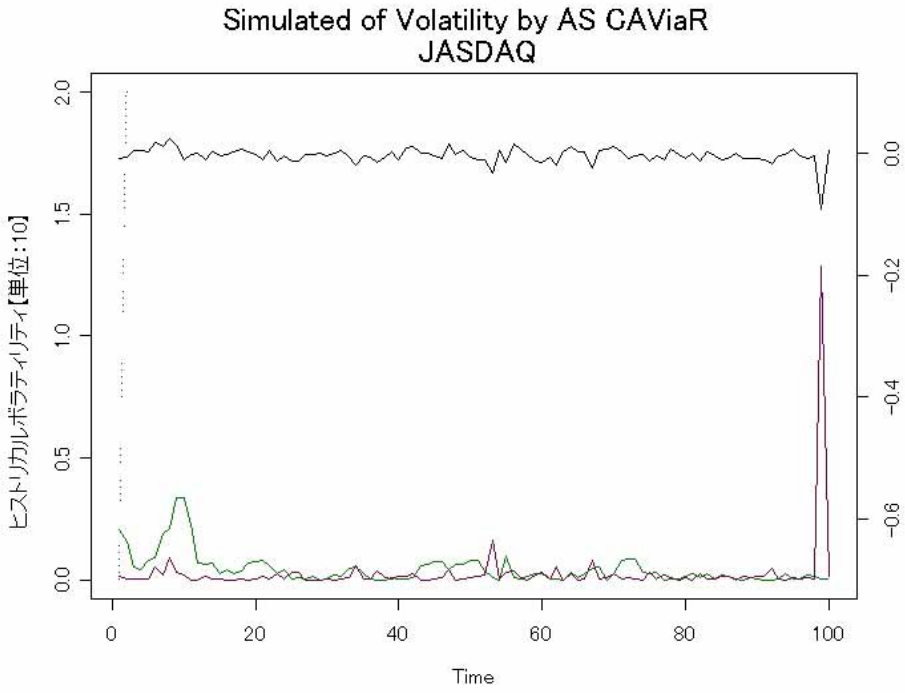

図 6. 1：AS CAViaR を用いた JASDAQ のボラティリティの予測

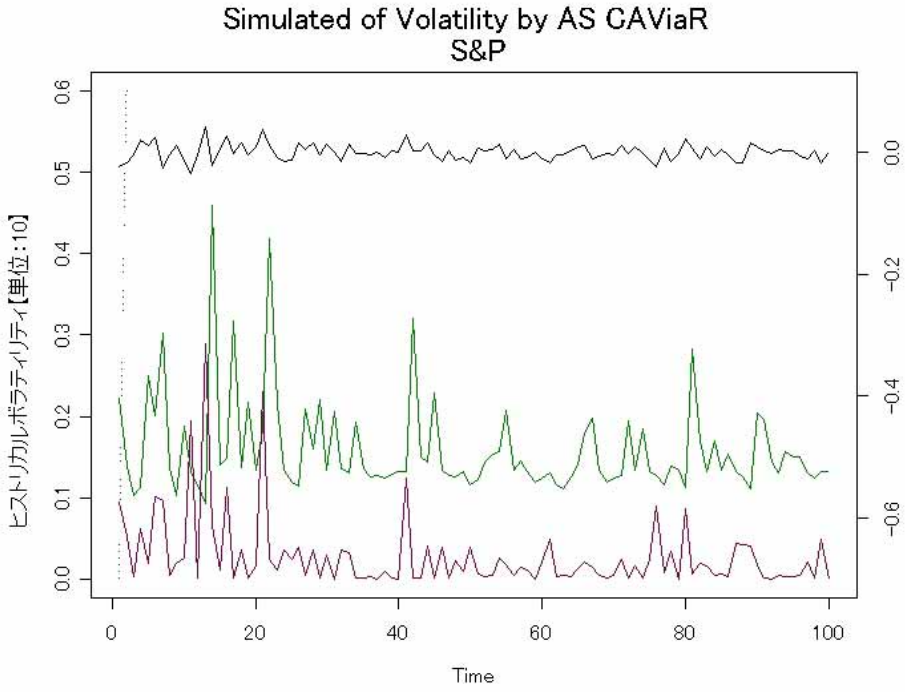

図 6. 2：AS CAViaR を用いた S&P のボラティリティの予測

3．4．3．　寄与率

モデルの寄与率を計算したところ，次の表 2 のような結果を得た．なお，GARCH 型モデルによって計算された寄与率の値よりも高かった値については，赤字で示した．

表 2 ：AS Slope モデルを用いた予測の寄与率

	10 日先まで予測	20 日先まで予測	100 日先まで予測
JASDAQ	8.5%	19.1%	0.2%
S&P	31.8%	1.2%	0.0%

3．4．4．　寄与率の考察

まず，JASDAQ と S&P ともに，短期間の予測の寄与率の方が大きく，長期間の予測の寄与率はほぼ 0 に近い．これは，AS CAViaR モデルを用いた予測も，GARCH モデルを用いた予測同様に，短期間の予測にしか使えず，長期間の予測には使えないことを示している．

次に，GARCH 型モデルを用いた予測の寄与率と比較してみる．短期間の予測においては，GARCH 型モデルよりも寄与率の値は大きかった．これは，AS CAViaR モデルを使った予測の方が，GARCH 型モデルを用いた予測の方が、精度が高いことを表している．

4.　　考察

事例研究の結果より，次のことが分かる．

（1）　ボラティリティ予測では，GARCH モデルも AS CAViaR モデルも短期予測は可能だが，長期予測は不可能である．

（2）　短期の予測に関しては，AS CAViaR モデルの方が，GARCH モデルよりも高い精度が期待できる．

まず（1）に関して，実務の場でボラティリティの予測を行う際は，定期的にデータを更新し，つぎつぎと新しく短期的な予測をしていく．モデルさえ構築されていれば予測を行う手間も大きなものではなく，明らかに遠い過去に予測したボラティリティより，近い過去に予測されたボラティリティのほうが信頼性が高い．したがって長期的な予測はあまり重要ではない．したがって，（1）に関しては大きな問題ではないといえる．

次に（2）に関して，本研究では 2 つの株価の Index で予測の精度の比較を行った．JASDAQ は日本のベンチャー企業が多く上場し，株価の収益率は高いボラティリティを示す．また，S&P は米国においてニューヨーク証券取引所, アメリカン証券取引所, NASDAQ のいずれかに上場している企業のうち，

代表的な 500 銘柄によって算出されるもので有名企業が多く, 株価の収益率は低いボラティリティを示す．事例研究より，低いボラティリティを示す市場の Index においても，高いボラティリティを示す市場の Index においても，AS CAViaR モデルを用いた予測の方が高い精度を期待できるという結果は有用であるといえる．

5. まとめ

本研究では，S-PLUS を用いてボラティリティの予測を行った．今回は 2 つのデータでしか扱わなかったが，データセットさえ作ってしまえば，script を少し書き換えるだけで予測を実行することができる．

S-PLUS は金融関係の function が充実しており，収益率の計算から GARCH モデルのあてはめまで容易に行え，また計算した結果を function 一つできれいに図として出力できることが，魅力的である．今回の実験においては，S-PLUS のベクトルの概念に慣れておらず，for 文を使ったプログラムを組んでしまったので, たいして大きなデータを扱ってはいなかったのだが, 実行に時間がかかってしまった．さらに，容量が大きなデータを用いた場合はかなりの時間を要することが予想される．S-PLUS はベクトル演算が早いのが特徴であるので，ベクトルを中心に記述された，もっとスマートで実行の早いプログラムを作るのも今後の課題の一つである．

6. 参考文献

[1] James W. Taylor (2005), Generating Volatility Forecasts from Value at Risk Estimates, MANAGEMENT SCIENCE, 51: 712-725

[2] Engle, R. F. and S. Manganelli (2004), CAViaR: Conditional autoregressive value at risk by regression quantiles, J. Bus. Econm. Statist, 22: 367-381

[3] Roger Koenker and Gilbert Bassett, Jr (1978), Regression Quantiles, Econometrica, Vol 46, No.1: 33-50

[4] 刈屋武明・矢島美寛・田中勝人・竹内啓(2003)，金融時系列の統計．岩波書店

[5] 廣松毅・浪花貞夫(1990)，経済時系列分析．朝倉書店

[6] R.A.ベッカー・J.M.チェンバース・A.R.ウィルクス(1991)，S 言語Ⅱ．共立出版株式会社

[7] Eric Zivot and Jiahui Wang(2001), Modeling Finacial Time Series with S-PLUS. Insightful

[8] Yahoo!ファイナンス，株価・投信時系列データ，http://table.yahoo.co.jp/t ，
最終閲覧日 9 月 30 日

[9] JASDAQ，JASDAQ INDEX 算出要領,
http://www.jasdaq.co.jp/data/JASDAQ_INDEX_Summary170204.pdf ，
最終閲覧日 10 月 1 日

付録

A. データセット

今回用いた, データセットである JASDAQ および S. P は, 日付を示す position, 始値を示す start, 高値を示す high, 安値を示す low, 終値を示す end で構成されたデータフレームである.

B. ヒストリカルボラティリティの算出および図の出力

```
## JASDAQ
tmp.J = ts(JASDAQ[,5])                                    # データをセット
HV.J = vector(mode = "double",length = length(tmp.J) - 1)   # 代入用のベクトルを準備
for(i in 1:length(tmp.J) - 1){
   a = tmp.J[i+1]
   b = tmp.J[i]
   c = (log(a/b))^2
   HV.J[i] = c * sqrt(250) *100
}                                                  # ヒストリカルボラティリティを計算し、順次代入
HV.J = ts(HV.J)                                        # 値を時系列化

## S&P
tmp.S = ts(S.P[,5])
HV.S = vector(mode = "double",length = length(tmp.J) - 1)
for(i in 1:length(tmp.S) - 1){
   a = tmp.S[i+1]
   b = tmp.S[i]
   c = (log(a/b))^2
   HV.S[i] = c * sqrt(250) *100
}
HV.S = ts(HV.S)

## plot
par(mfrow=c(1,2))
tsplot(HV.J, col=3)
title(main="JASDAQ Histrical Volatility", xlab="Time", ylab="Volatility(%)")
tsplot(HV.S, col=3)
title(main="S&P Histrical Volatility", xlab="Time", ylab="Volatility(%)")
```

C. GARCHモデル型によるボラティリティの予測

なお, このscriptはデータセットを変えるだけで様々なデータに適用可能. TGARCHモデルを適用するには、garchをtgarchに変更することで実現できる.

```
## データを用意

tmp = JASDAQ[,5]
data1.J = ts(tmp[1:1001])      # 学習用データ
data2.J = ts(tmp)              # 全データ
data3.J = ts(getReturns(data1.J))    # 学習用データの収益率（標本数：1000）
data4.J = ts(getReturns(data2.J))    # 全データ収益率
eps.J = ts(getReturns(data2.J) - mean(getReturns(data2.J)[1:1000]))   # 評価用データの偏差
# 学習用データでGARCHモデルの当てはめ
# ~garch を ~tgarch に変更することによってGJRGARCHに変更可能
data.g.J = garch(data3.J~1,~garch(1,1), trace=F, leverage=T)

## ヒストリカルボラティリティの計算

data.s.J = ts(JASDAQ[,5])
HV.1.J = 1:length(data.s.J - 1)
for(i in 1:length(data.s.J - 1)){
   a = data.s.J[i+1]
   b = data.s.J[i]
   c = (log(a/b))^2
   HV.1.J[i] = c * sqrt(250) *100
}

## 以下、ボラティリティの推定　（参照 -> [5] P255）

sigma.start.J = as.numeric(data.g.J$sigma.t[1000])       # 学習用データの最後の標準偏差
eps.start.J = as.numeric(data.g.J$residuals[1000])       # 学習用データの最後の残差
eps.start.J = matrix(eps.start.J, 1, 1000)           # 以下シミュレーションのパスを作成
error.J = rbind(eps.start.J, matrix(rnorm(100*1000), 100))
set.seed(4989)
data.t.pred.J = simulate(data.g.J, n=100, n.rep=1000, sigma.start=sigma.start.J,
   etat=error.J)$sigma.t  # シミュレーションをする
vol.mean.J = rowMeans(data.t.pred.J)    # 作成されたパスの平均を取ってボラティリティの平方根とする
vol.stdev.J = rowStdevs(data.t.pred.J)
```

```
sigma.2.J = (vol.mean)^2 * sqrt(250) * 100   # 1年250営業日として，ボラティリティを年率で表現する

## 作図

HV.data.J = ts(HV.1.J[1001:1100])                    # 評価期間のヒストリカルボラティリティを抽出

par(mar=rep(6,4))                                    # 余白を設定する
tsplot(sigma.2.J, col=4, range(0,1.5))
tslines(HV.data.J/10, col=3,axes=F)
par(new=TRUE)                                      # 新しい高水準であると宣言
tsplot(eps.J[1001:1100],axes=F,range(-0.7,0.1))          # 別の縦軸でplot
axis(side=4)
title(main="Simulated of Volatility by GARCH ¥n JASDAQ", xlab="Time", ylab="ヒストリカルボラ
    ティリティ【単位：10】")   # 緑線が予測値，赤線が実測値

## 寄与率

corr.100.J = cor(HV.data.J,sigma.2.J)                # 100日先までの予測の寄与率
r.2.100.J = (corr.100.J)^2
r.2.100.J

corr.20.J = cor(HV.data.J[1:20],sigma.2.J[1:20])     # 20日先までの予測の寄与率
r.2.20.J = (corr.20.J)^2
r.2.20.J

corr.10.J = cor(HV.data.J[1:10],sigma.2.J[1:10])     # 10日先までの予測の寄与率
r.2.10.J = (corr.10.J)^2
r.2.10.J
```

D. function：caviar

```
###QRSum 最小化モデル

##Asymmetric Slope

QRSum.ASCAViaR = function(y,eps.p,eps.m,theta){

   Time <- Set()
   Slope <- Set(1:4)

   length.time <- length(y)

   i <- Element(set = Slope)
   t <- Element(set = Time)

   y <- Parameter(list(1:length.time,y), index = t)
   eps.p <- Parameter(list(1:length.time,eps.p),index = t)
   eps.m <- Parameter(list(1:length.time,eps.m),index = t)

   beta <- Variable(index = i)
   Q <- Variable(index = t)

   obj = Objective(type="minimize")

   Q[t+1,t<length.time] == beta[1] + beta[2] * Q[t] + beta[3] * eps.p[t] + beta[4] * eps.m[t]

   obj ~ Sum(ife(y[t] >= Q[t],theta * (y[t] - Q[t]), (theta - 1) * (y[t] - Q[t])),t)
}

##Symmetric Absolute Slope

QRSum.SASCAViaR = function(y,eps,theta){

   Time <- Set()
   Slope <- Set(1:3)

   length.time <- length(y)
   eps <- abs(eps)
```

```
    i <- Element(set = Slope)
    t <- Element(set = Time)

    y <- Parameter(list(1:length.time,y), index = t)
    abseps <- Parameter(list(1:length.time,eps),index = t)

    beta <- Variable(index = i)
    Q <- Variable(index = t)

    obj = Objective(type="minimize")

    Q[t+1,t<length.time] == beta[1] + beta[2] * Q[t] + beta[3] * abseps[t]

    obj ~ Sum(ife(y[t] >= Q[t],theta * (y[t] - Q[t]), (theta - 1) * (y[t] - Q[t])),t)
}

##Adaptive CAViaR

QRSum.AdptCAViaR = function(y,eps,theta){

    Time <- Set()
    Slope <- Set(1:1)

    length.time <- length(y)

    i <- Element(set = Slope)
    t <- Element(set = Time)

    y <- Parameter(list(1:length.time,y), index = t)
    eps <- Parameter(list(1:length.time,eps),index = t)

    beta <- Variable(index = i)
    Q <- Variable(index = t)

    obj = Objective(type="minimize")

    Q[t+1,t<length.time] == Q[t] + beta[1] * (theta - ife(eps[t] <= Q[t],1,0))
```

```
    obj ~ Sum(ife(y[t] >= Q[t],theta * (y[t] - Q[t]), (theta - 1) * (y[t] - Q[t])),t)
}

###CAViaRモデル

caviar = function(x,model="Asymmetric",prob=0.05,graph=T,scaling="on",ep=1e-3){
    #モジュール呼び出し
    module(nuopt)

    #元データ編集
    eps = diff(log(x))

    #パラメータ推定開始

    if(model == "Asymmetric"){
        nuopt.options(maxitn = 200,method="auto",scaling=scaling,eps=ep)

        #epsilon^+とepsilon^-を計算
        eps.p = ifelse(eps>0,eps,0)
        eps.m = ifelse(eps<0,abs(eps),0)

        #最適化
        result.problem = System(model=QRSum.ASCAViaR,eps,eps.p,eps.m,prob)
        result.solution = solve(result.problem)

    }else if(model == "Symmetric"){
        nuopt.options(maxitn = 200,method="auto",scaling=scaling,eps=ep)

        #最適化
        result.problem = System(model=QRSum.SASCAViaR,eps,eps,prob)
        result.solution = solve(result.problem)

    }else if(model == "Adaptive"){
        nuopt.options(maxitn = 150,method="auto",scaling=scaling,eps=ep)

        #最適化
        result.problem = System(model=QRSum.AdptCAViaR,eps,eps,prob)
        result.solution = solve(result.problem)
```

```
    }else{
        print("モデル名、違うから。。。")
    }

    Q = result.solution$variable$Q$current
    beta = result.solution$variable$beta$current
    rslt = list(percentile = Q, beta = beta, theta = prob,model=model)

    if(graph){
        tsplot(eps)
        lines(Q,col=6)
    }

    rslt
}
```

E．CAViaRモデルのあてはめ

```
##JASDAQ当てはめ

data.J = JASDAQ[,5]     #データの準備
asyca.05.J = caviar(data.J[1:1001], prob=0.05) #確率5%のAS CAViaRを学習用データによって推定
asyca.95.J = caviar(data.J[1:1001], prob=0.95) #確率95%のAS CAViaRを学習用データによって推定
quant.asyca.05.J = ts(asyca.05.J$percentile[2:1000]) #確率5%の分位点を時系列化
quant.asyca.95.J = ts(asyca.95.J$percentile[2:1000]) #確率95%の分位点を時系列化
eps.J = ts(getReturns(data.J) - mean(getReturns(data.J)[1:1000])) #実測値の偏差

##S&P当てはめ

data.S = S.P[,5]
asyca.05.S = caviar(data.S[1:1001], prob=0.05)
asyca.95.S = caviar(data.S[1:1001], prob=0.95)
quant.asyca.05.S = ts(asyca.05.S$percentile[2:1000])
quant.asyca.95.S = ts(asyca.95.S$percentile[2:1000])
eps.S = ts(getReturns(data.S) - mean(getReturns(data.S)[1:1000]))

##plot

par(mfrow=c(1,2))   #2つのグラフを一度に表示
tsplot(eps.J[2:1000], col=2, range(-0.2,0.12))
tslines(quant.asyca.95.J, col=4)
tslines(quant.asyca.05.J, col=3)
title(main="Asymmetric Slope CAViaR Fitting ¥n (JASDAQ in-sample)", xlab="Time",
   ylab="log.return")
tsplot(eps.S[2:1000], col=2, range(-0.2,0.12))
tslines(quant.asyca.95.S, col=4)
tslines(quant.asyca.05.S, col=3)
title(main="Asymmetric Slope CAViaR Fitting ¥n (S&P in-sample)", xlab="Time",
   ylab="log.return")
```

F．CAViaRモデルによるボラティリティの予測

```
##predict quantile


epsp.J = ifelse(eps.J > 0, eps.J, 0)     #(3)式右辺第3項のepsilon-plusを計算する
epsm.J = ifelse(eps.J < 0, abs(eps.J), 0)#(3)式右辺第3項のepsilon-minusを計算する
epsp.S = ifelse(eps.S > 0, eps.S, 0)
epsm.S = ifelse(eps.S < 0, abs(eps.S), 0)

pred.quant.05.J = vector(mode = "double",length=1100) #ベクトルを用意する
pred.quant.95.J = vector(mode = "double",length=1100)
pred.quant.05.S = vector(mode = "double",length=1100)
pred.quant.95.S = vector(mode = "double",length=1100)

#分位点を先に予測した係数によって計算する
#ベクトル演算にしないとS-PLUSの魅力を半減させる部分
for(i in 1:100){
    if(i == 1){
        pred.quant.05.J[i+1000] = asyca.05.J$beta[1] + asyca.05.J$beta[2] * asyca.05.J$percentile[1000]
    + asyca.05.J$beta[3] * epsp.J[i+999] + asyca.05.J$beta[4] * epsm.J[i+999]
        pred.quant.95.J[i+1000] = asyca.95.J$beta[1] + asyca.95.J$beta[2] * asyca.95.J$percentile[1000]
    + asyca.95.J$beta[3] * epsp.J[i+999] + asyca.05.J$beta[4] * epsm.J[i+999]
        pred.quant.05.S[i+1000] = asyca.05.S$beta[1] + asyca.05.S$beta[2] * asyca.05.S$percentile[1000]
    + asyca.05.S$beta[3] * epsp.S[i+999] + asyca.05.S$beta[4] * epsm.S[i+999]
        pred.quant.95.S[i+1000] = asyca.95.S$beta[1] + asyca.95.S$beta[2] * asyca.95.S$percentile[1000]
    + asyca.95.S$beta[3] * epsp.S[i+999] + asyca.05.S$beta[4] * epsm.S[i+999]
    }
    else{
        pred.quant.05.J[i+1000] = asyca.05.J$beta[1] + asyca.05.J$beta[2] * pred.quant.05.J[i+999] +
    asyca.05.J$beta[3] * epsp.J[i+999] + asyca.05.J$beta[4] * epsm.J[i+999]
        pred.quant.95.J[i+1000] = asyca.95.J$beta[1] + asyca.95.J$beta[2] * pred.quant.95.J[i+999] +
    asyca.95.J$beta[3] * epsp.J[i+999] + asyca.05.J$beta[4] * epsm.J[i+999]
        pred.quant.05.S[i+1000] = asyca.05.S$beta[1] + asyca.05.S$beta[2] * pred.quant.05.S[i+999] +
    asyca.05.S$beta[3] * epsp.S[i+999] + asyca.05.S$beta[4] * epsm.S[i+999]
        pred.quant.95.S[i+1000] = asyca.95.S$beta[1] + asyca.95.S$beta[2] * pred.quant.95.S[i+999] +
    asyca.95.S$beta[3] * epsp.S[i+999] + asyca.05.S$beta[4] * epsm.S[i+999]
    }
}
pred.quant.05.J
```

```
pred.quant.05.J = pred.quant.05.J[1001:1100] #5%の分位点の予測値
pred.quant.95.J = pred.quant.95.J[1001:1100] #95%の分位点の予測値
pred.quant.05.S = pred.quant.05.S[1001:1100]
pred.quant.95.S = pred.quant.95.S[1001:1100]

##predict volatility

pred.vol.J = ts ((pred.quant.95.J - pred.quant.05.J)/3.25)^2 * sqrt(250) *100  #(7)式を用いてボラティリ
    ティを予測する
pred.vol.S = ts ((pred.quant.95.S - pred.quant.05.S)/3.25)^2 * sqrt(250) *100

##predict volatility plot

par(mfrow=c(1,1))
par(mar=rep(6,4))
tsplot(pred.vol.J, range(0,2),col=4)
tslines(HV.J[1001:1100]/10,col=3)
par(new=TRUE)
tsplot(eps.J[1001:1100],axes=F,range(-0.7,0.1))
axis(side=4)
title(main="Simulated of Volatility by AS CAViaR ¥n JASDAQ", xlab="Time", ylab="ヒストリカルボラティリテ
    ィ【単位：10】")

par(mar=rep(6,4))
tsplot(pred.vol.S, range(0,0.6),col=4)
tslines(HV.S[1001:1100]/10,col=3)
par(new=TRUE)
tsplot(eps.S[1001:1100],axes=F,range(-0.7,0.1))
axis(side=4)
title(main="Simulated of Volatility by AS CAViaR ¥n S&P", xlab="Time", ylab="ヒストリカルボラティリティ【単
    位：10】")

##寄与率

pred.corr.J.100 = cor(HV.J[1001:1100],pred.vol.J[1:100])# 100日先までの予測の寄与率
pred.r.2.J.100 = (pred.corr.J.100)^2
pred.r.2.J.100

pred.corr.J.20 = cor(HV.J[1001:1020],pred.vol.J[1:20])  # 20日先までの予測の寄与率
```

```
pred.r.2.J.20 = (pred.corr.J.20)^2
pred.r.2.J.20

pred.corr.J.10 = cor(HV.J[1001:1010],pred.vol.J[1:10])  # 10日先までの予測の寄与率
pred.r.2.J.10 = (pred.corr.J.10)^2
pred.r.2.J.10

pred.corr.S.100 = cor(HV.S[1001:1100],pred.vol.S[1:100])# 100日先までの予測の寄与率
pred.r.2.S.100 = (pred.corr.S.100)^2
pred.r.2.S.100

pred.corr.S.20 = cor(HV.S[1001:1020],pred.vol.S[1:20])  # 20日先までの予測の寄与率
pred.r.2.S.20 = (pred.corr.S.20)^2
pred.r.2.S.20

pred.corr.S.10 = cor(HV.S[1001:1010],pred.vol.S[1:10])  # 10日先までの予測の寄与率
pred.r.2.S.10 = (pred.corr.S.10)^2
pred.r.2.S.10
```