Interval Forecasts Evaluation: R programs for a new independence test

P. Araújo Santos
Instituto Politécnico de Santarém and CEAUL
paulo.santos@esg.ipsantarem.pt

May 25, 2010

Abstract: Interval forecasts evaluation can be reduced to examining the unconditional coverage and independence properties of the hit sequence. This work provides computer programs to apply a new independence test and to compare this test with existing procedures. The new test is suitable for detect models with a tendency to generate clusters of violations, is based on an exact distribution that does not depend on an unknown parameter and outperforms, in terms of power, existing procedures in realistic settings.

Keywords: hit sequence, backtesting.

1 Introduction

To test the forecasting methodology being applied, we consider the observed sample path, $\{r_t\}_{t=1}^T$, of the time series R_t, and the corresponding sequence of *out-of-sample* interval forecasts $\{(L_{t+1|t}(1-p), U_{t+1|t}(1-p))\}_{t=1}^T$, where $L_{t+1|t}(1-p)$ and $U_{t+1|t}(1-p)$ are the lower and upper limits of the interval forecast with probability $1-p$, for time $t+1$ made at time t. The hit function is defined as,

$$I_{t+1}(p) = \begin{cases} 1 & \text{if } R_{t+1} \notin (L_{t+1|t}(1-p), U_{t+1|t}(1-p)) \\ 0 & \text{if } R_{t+1} \in (L_{t+1|t}(1-p), U_{t+1|t}(1-p)). \end{cases} \tag{1.1}$$

For example, Value-at-Risk is an important application of interval forecasting, where the intervals are one-sided. In this context $L_{t+1|t}(1-p) = VaR_{t+1|t}(p)$ and $U_{t+1|t}(1-p) = +\infty$.

Christoffersen (1998) showed that evaluating interval forecasts can be reduced to examining whether the hit sequence,$\{I_t\}_{t=1}^T$, satisfies the unconditional coverage (UC) and independence (IND) properties. UC hypothesis means $P[I_{t+1}(p) = 1] = p$, $\forall_t$. IND hypothesis means that past violations do not hold information about future violations. When both properties are valid then we write $P[I_{t+1}(p) = 1|\Omega_t] = p, \forall_t$, and we say that forecasts have a correct conditional coverage (CC). In Lemma 1 of Christoffersen (1998) it is shown that condition CC is equivalent to

$I_{t+1}(p) \overset{iid}{\sim} Bernoulli(p)$.

There are several backtesting procedures for evaluating intervals forecasts; for a detailed review see Berkowitz *et al.* (2009). The Christoffersen (1998) Markov IND and CC likelihood ratio tests, are perhaps the most widely used in the literature. These tests, based on asymptotic distributions, are only sensible to one violation immediately followed by other, ignoring all other patterns of clustering.

In the same line as Engle and Manganelli (2004), Berkowitz *et al.* (2009) consider the autoregression

$$I_t = \alpha + \sum_{k=1}^{n} \beta_{1k} I_{t-k} + \sum_{k=1}^{n} \beta_{2k} g(I_{t-k}, I_{t-k-1}, ..., R_{t-k}, R_{t-k-1}) + \varepsilon_t \tag{1.2}$$

with $n = 1$ and $g(I_{t-k}, I_{t-k-1}, ..., R_{t-k}, R_{t-k-1}) = VaR_{t-k+1|t-k}(p)$. These authors proposed the logit model and test the CC hypothesis with a likelihood ratio test considering for the null $P(I_t = 1) = 1/(1+e^{-\alpha}) = p$ and the coefficients β_{11} and β_{21} equal to zero. For the the IND hypothesis, the test is adapted considering β_{11} and β_{21} equal to zero. We refer these tests as the CAViaR tests of Engle and Manganelli (CAViaR).

A *duration-based* approach emerged in the literature (e.g. Danielsson and Morimoto (2000), Christofferson and Pelletier (2004), Haas (2005)). In this set-up, let us define the duration between two consecutive violations as

$$D_i := t_i - t_{i-1} \tag{1.3}$$

where t_i denotes the day of violation number i and $t_0 = 0$, which implies that D_1 is the time until the first violation. If the IND hypothesis is valid then $I_{t+1}(p) \overset{iid}{\sim} Bernoulli(\pi)$, with $0 < \pi < 1$, and the common distribution of durations (1.3) is geometric with pmf

$$f_D(d;\pi) = (1-\pi)^{(d-1)}\pi,\ d \in \mathbb{N},\ 0 < \pi < 1. \tag{1.4}$$

The Generalised Method of Moments test framework suggested by Bontemps and Meddahi (2008) to test for distributional assumptions was extended by Candelon *et al.* (2008) to the case of VaR forecasts accuracy. In the group of previous duration-based tests it is shown that the proposed GMM tests are the best performers.

Recently, a new class of independence tests was proposed by Araújo Santos and Fraga Alves (2009). Therein, the following definition was proposed.

Definition 1.1 (Tendency to clustering of violations)**.** *A hit function (1.1) has a tendency to clustering of violations if the median of $D_{N:N}/D_{[N/2]:N}$ is higher than the median under the IND hypothesis.*

For explicitly testing the IND hypothesis versus tendency to clustering of violations, the following test statistic was proposed

$$T_{N,[N/2]} = \log 2\frac{D_{N:N} - 1}{D_{[N/2]:N}} - \log \mathrm{N}. \tag{1.5}$$

The new test is suitable for detect models with a tendency to generate clusters of violations, is based on an exact distribution and does not rely on an asymptotic distribution. Moreover, the test is pivotal in the sense that is based on a distribution that does not depend on an unknown parameter. The alternative independence test referred in this section are based on asymptotic distributions and suffer from small sample bias. Other drawback of these tests is that the distribution depends on an unknown parameter, the coverage probability p. Under IND this parameter is unknown. Moreover, the new test outperforms existing procedures in realistic settings in terms of power.

Subsection 2.1, presents a R program for the implementation of the test with any hit sequence. This program simulates the upper bound for the p-value. Subsections 2.2 and 2.3, presents R programs for a comparative simulation study with Markov, CAViaR and GMM independence tests, computing respectively, the empirical power of the tests and the empirical type I error rates. In subsection 2.4, we provide a R program to obtain the critical values by simulation. Finally, in subsection 2.5, the table needed for programs of subsections 2.2 and 2.3, is given.

2 R programs

2.1 R program for the new independence test

```
#### Use the following code to introduce a hit sequence
hit <- c(0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0,1,0)
#### Or use the following code to read the hit sequence from a text file
##  hit <- read.table("hit.txt")[,1]

tt <- length(hit)
#### Durations
 no_hit_duration <- 0
 j<-1
 zeros <- 0
 for(i in 1:tt) {
 if (hit[i]<1){
 zeros <- zeros+1
 }
 else {
  no_hit_duration[j]<- zeros+1
  zeros <- 0
  j <- j+1
  }
 }
 no <- no_hit_duration
 n <- length(no)

#### T[0.5] Independence Test
 no <- sort(no)
  observed_T <-log(2)*(no[n]-1)/no[floor(0.5*n)]-log(n)
  observed_T

#### Simulation of the upper bound for the p-value
 v <- 0
 replicas <- 250000
 v <- rep(0, times=replicas)
 print('wait for simulation of upper bound for the p-value')
 for(i in 1:replicas) {
```

```
 if((i/10000-floor(i/10000))==0){print(replicas-i)}
 u<- runif(n)
 y<- -log(1-u)
 no_simul <- sort(y)
 v[i] <- log(2)*(no_simul[n]/no_simul[floor(n/2)])-log(n)
 }

simulated_upper_bound_p_value <- length(v[v>=observed_T])/replicas
observed_T
simulated_upper_bound_p_value
```

2.2 R program for the empirical power

```
library(fGarch)
table <- read.table("table_T50.txt")[,2]

replicas <- 5000
tt <- 500 ## size of the hit sequence
ws <- 500 ## window size
coverage <- 0.01
v1<-0
v2<-0
v3<-0
v4<-0
var <- 0
reject_freq <- 0
failures <-0

for(t in 1:replicas) {
 print (t)
 ############################  MODEL1: Gaussian GARCH(1,1) ######################
   ##   model = garchSpec(model = list(omega = 0.05, alpha = 0.1, beta = 0.85))
   ##   a <- garchSim(model, n = tt+ws)
#################################################################################

 ############################  MODEL2: Skewed t APARCH(1,1) #####################
     model = garchSpec(model = list(mu = 0, omega = 0.03,
     alpha = c(0.086), gamma = c(0.64), beta = 0.91, delta = 1.15,
     shape = 10, skew=0.88), cond.dist = "sstd")
     a <- garchSim(model, n = tt+ws)
#################################################################################

#### Hit function
 hit <-runif(tt)
 for(i in 1:tt) {
 iws <- i+ws
 m_iws <- iws-1
 b <- a[i:m_iws]
 th <- quantile(b, probs=coverage)
 var[i] <- th
 if(a[i+ws]<th){hit[i]=1}
 else {hit[i]=0}
 }

#### Durations
 no_hit_duration <- 0
 j<-1
 zeros <- 0
 for(i in 1:tt) {
 if (hit[i]<1){ zeros <- zeros+1 }
 else {
 no_hit_duration[j]<- zeros+1
```

```
 zeros <- 0
 j <- j+1
  }
 }
 no <- no_hit_duration
 n <-length(no)

#### Exclude samples with size less than 2
 if (n<2){
 v1[t] <- -1
 v2[t] <- -1
 v3[t] <- -1
 v4[t] <- -1
 failures <- failures+1 }
 else{

#### T[0.5] Independence Test
 no <- sort(no)
 observed_T <-log(2)*(no[n]-1)/no[floor(0.5*n)]-log(n)
 if (observed_T  > table[n]){reject_freq <- reject_freq+1}


#### Markov Independence Test
 zz <- 0
 umz <- 0
 zum <- 0
 umum <- 0
 m_tt <- tt-1

 for(k in 1:m_tt) {
 i<-k+1
 if (hit[k]==0 & hit[i]==0){
 zz <- zz +1
 }
 else if (hit[k]==0 & hit[i]==1){
 zum <- zum +1
 }
 else if (hit[k]==1 & hit[i]==1){
 umum <- umum +1
 }
 else{
 umz <- umz +1
  }
 }

p00 <- zz/(zz+zum)
p01 <- zum/(zz+zum)
p10 <- umz/(umz+umum)
p11 <- umum/(umz+umum)
llp <- (zum+umum)/(zz+umz+zum+umum)
ll2 <- ((1-llp)^(zz+umz))*(llp^(zum+umum))
ll1 <- (p00^zz)*(p01^zum)*(p10^umz)*(p11^umum)
v1[t] <- -2*log(ll2/ll1)

#### Caviar Independence Test
hit1 <- hit[1:m_tt]
hit2 <- hit[2:tt]
var2 <- var[2:tt]
mylogit<- glm(hit2~hit1+var2, family=binomial(link="logit"), na.action=na.pass)
logLik(mylogit)

alpha <- -log(length(hit)/sum(hit)-1)
loglik1 <- -sum(1-hit2)*alpha-(tt-1)*log(1+exp(-alpha))

emv <- mylogit$coefficients
```

```
 emv1 <- emv[1]
 emv2 <- emv[2]
 emv3 <- emv[3]
 loglik2 <- -sum((1-hit2)*(emv1+emv2*hit1+emv3*var2))-sum(log(1+exp(-emv1-emv2*hit1-emv3*var2)))
 v2[t]<- -2*(loglik1-loglik2)

##### GMM Independence Tests
 pp <- n/tt
 m1 <- (1-pp*no)/sqrt(1-pp)
 m2 <- (3-pp-pp*no)*(1-pp*no)/(2-2*pp)-0.5
 m3 <-(5-2*pp-pp*no)/(3*sqrt(1-pp))*m2-(2/3)*m1
 m4 <-(7-3*pp-pp*no)/(4*sqrt(1-pp))*m3-(3/4)*m2
 m5 <-(9-4*pp-pp*no)/(5*sqrt(1-pp))*m4-(4/5)*m3
 mm1 <- sum(m1)/sqrt(n)
 mm2 <- sum(m2)/sqrt(n)
 mm3 <- sum(m3)/sqrt(n)
 mm4 <- sum(m4)/sqrt(n)
 mm5 <- sum(m5)/sqrt(n)
 v3[t] <- (mm1^2)+(mm2^2)+(mm3^2)
 v4[t] <-(mm1^2)+(mm2^2)+(mm3^2)+(mm4^2)+(mm5^2)
  }
  }

#### Empirical Power of Tests and Frequency of Excluded Samples
T_test <- (reject_freq)/(replicas-failures)
M_ind <- length(v1[v1>2.706])/(replicas-failures)  ### Asymptotic critical value
CAViaR <- length(v2[v2>4.605])/(replicas-failures)  ### Asymptotic critical value
J_ind3 <- length(v3[v3>4.605]) /(replicas-failures)  ### Asymptotic critical value
J_ind5 <- length(v4[v4>7.779]) /(replicas-failures)  ### Asymptotic critical value
FSE <-  failures/replicas

T_test
M_ind
CAViaR
J_ind3
J_ind5
FSE
```

2.3 R program for the empirical type I error rates

```
library(fGarch)
table <- read.table("table_T50.txt")[,2]

replicas <- 5000
tt <- 500 ## size of the hit sequence
ws <- 500 ## window size
coverage <- 0.01
v1<-0
v2<-0
v3<-0
v4<-0
var_ind <- 0
reject_freq <- 0
failures <-0

for(t in 1:replicas) {
 print (t)

   model = garchSpec(model = list(omega = 0.05, alpha = 0.1, beta = 0.85))
   aa <- garchSim(model, n = 1000)

#### Hit function under IND hypothesis
```

```
 hit <-rbinom(tt,1,coverage)
 for(i in 1:tt) {
 iws <- i+ws
 m_iws <- iws-1
 c <- aa[i:m_iws]
 var_ind[i] <- quantile(c, probs=(c(100-100*coverage)/100))
 }

#### Durations
 no_hit_duration <- 0
 j<-1
 zeros <- 0
 for(i in 1:tt) {
 if (hit[i]<1){ zeros <- zeros+1 }
 else {
 no_hit_duration[j]<- zeros+1
 zeros <- 0
 j <- j+1
  }
 }
 no <- no_hit_duration
 n <-length(no)

#### Exclude samples with size less than 2
 if (n<2){
 v1[t] <- -1
 v2[t] <- -1
 v3[t] <- -1
 v4[t] <- -1
 failures <- failures+1 }
 else{

#### T[0.5] Independence Test
 no <- sort(no)
 observed_T <-log(2)*(no[n]-1)/no[floor(0.5*n)]-log(n)
 if (observed_T  > table[n]){reject_freq <- reject_freq+1}


#### Markov Independence Test
 zz <- 0
 umz <- 0
 zum <- 0
 umum <- 0
 m_tt <- tt-1

 for(k in 1:m_tt) {
 i<-k+1
 if (hit[k]==0 & hit[i]==0){
 zz <- zz +1
 }
 else if (hit[k]==0 & hit[i]==1){
 zum <- zum +1
 }
 else if (hit[k]==1 & hit[i]==1){
 umum <- umum +1
 }
 else{
 umz <- umz +1
  }
 }

p00 <- zz/(zz+zum)
p01 <- zum/(zz+zum)
p10 <- umz/(umz+umum)
p11 <- umum/(umz+umum)
```

```
llp <- (zum+umum)/(zz+umz+zum+umum)
ll2 <- ((1-llp)^(zz+umz))*(llp^(zum+umum))
ll1 <- (p00^zz)*(p01^zum)*(p10^umz)*(p11^umum)
v1[t] <- -2*log(ll2/ll1)

#### Caviar Independence Test
hit1 <- hit[1:m_tt]
hit2 <- hit[2:tt]
var2 <- var_ind[2:tt]
mylogit<- glm(hit2~hit1+var2, family=binomial(link="logit"), na.action=na.pass)
logLik(mylogit)

alpha <- -log(length(hit)/sum(hit)-1)
loglik1 <- -sum(1-hit2)*alpha-(tt-1)*log(1+exp(-alpha))

emv <- mylogit$coefficients
 emv1 <- emv[1]
 emv2 <- emv[2]
 emv3 <- emv[3]
 loglik2 <- -sum((1-hit2)*(emv1+emv2*hit1+emv3*var2))-sum(log(1+exp(-emv1-emv2*hit1-emv3*var2)))
 v2[t]<- -2*(loglik1-loglik2)

##### GMM Independence Tests
 pp <- n/tt
 m1 <- (1-pp*no)/sqrt(1-pp)
 m2 <- (3-pp-pp*no)*(1-pp*no)/(2-2*pp)-0.5
 m3 <-(5-2*pp-pp*no)/(3*sqrt(1-pp))*m2-(2/3)*m1
 m4 <-(7-3*pp-pp*no)/(4*sqrt(1-pp))*m3-(3/4)*m2
 m5 <-(9-4*pp-pp*no)/(5*sqrt(1-pp))*m4-(4/5)*m3
 mm1 <- sum(m1)/sqrt(n)
 mm2 <- sum(m2)/sqrt(n)
 mm3 <- sum(m3)/sqrt(n)
 mm4 <- sum(m4)/sqrt(n)
 mm5 <- sum(m5)/sqrt(n)
 v3[t] <- (mm1^2)+(mm2^2)+(mm3^2)
 v4[t] <-(mm1^2)+(mm2^2)+(mm3^2)+(mm4^2)+(mm5^2)
  }
  }

#### Empirical type I error rates and Frequency of Excluded Samples
T_test <- (reject_freq)/(replicas-failures)
M_ind <- length(v1[v1>2.706])/(replicas-failures)  ### Asymptotic critical value
CAViaR <- length(v2[v2>4.605])/(replicas-failures)  ### Asymptotic critical value
J_ind3 <- length(v3[v3>4.605]) /(replicas-failures)  ### Asymptotic critical value
J_ind5 <- length(v4[v4>7.779]) /(replicas-failures)  ### Asymptotic critical value
FSE <-  failures/replicas

T_test
M_ind
CAViaR
J_ind3
J_ind5
FSE
```

2.4 R program for obtain critical values by simulation

```
replicas <- 1000000
n_first <- 2
n_last <- 200
crit_01 <- 0
crit_05 <- 0
crit_10 <- 0

for(j in n_first:n_last){
print(j)
n <-j
v <- rep(0,times=replicas)

for(i in 1:replicas) {
u<- runif(n)
y<- -log(1-u)
no <- sort(y)

v[i] <- log(2)*no[n]/no[floor(0.5*n)]-log(n)
}

crit_01[j] <- quantile(v,  probs=0.99)
crit_05[j] <- quantile(v,  probs=0.95)
crit_10[j] <- quantile(v,  probs=0.90)
}

crit_01
crit_05
crit_10
```

2.5 Table for the $table\_50.txt$ file

```
1   -1
2   12.46
3   28.23
4   6.72
5   10.54
6   5.33
7   7.4
8   4.7
9   6.1
10  4.3
11  5.37
12  4.04
13  4.9
14  3.84
15  4.58
16  3.7
17  4.34
18  3.59
19  4.14
20  3.5
21  3.98
22  3.41
23  3.86
24  3.34
25  3.75
26  3.28
27  3.66
28  3.23
29  3.58
30  3.18
31  3.51
32  3.14
33  3.46
34  3.11
35  3.4
36  3.07
37  3.35
38  3.04
39  3.31
40  3.02
```

```
41  3.27
42  2.99
43  3.23
44  2.96
45  3.2
46  2.94
47  3.16
48  2.93
49  3.14
50  2.91
51  3.11
52  2.88
53  3.08
54  2.87
55  3.06
56  2.85
57  3.04
58  2.84
59  3.01
60  2.82
61  3.0
62  2.81
63  2.98
64  2.8
65  2.96
66  2.79
67  2.95
68  2.77
69  2.93
70  2.77
71  2.92
72  2.76
73  2.9
74  2.75
75  2.88
76  2.74
77  2.88
78  2.73
79  2.87
80  2.72
81  2.85
82  2.71
83  2.84
84  2.71
85  2.84
86  2.7
87  2.82
88  2.69
89  2.8
90  2.68
91  2.8
92  2.68
93  2.79
94  2.67
95  2.78
96  2.67
97  2.78
98  2.66
99  2.77
100 2.65
101 2.76
102 2.65
103 2.75
104 2.65
105 2.74
106 2.63
107 2.74
108 2.64
109 2.73
110 2.62
111 2.72
112 2.62
113 2.72
114 2.62
115 2.71
116 2.62
117 2.7
118 2.61
119 2.7
120 2.6
121 2.69
122 2.6
123 2.69
124 2.6
125 2.68
126 2.59
127 2.68
128 2.59
```

```
129 2.67
130 2.58
131 2.67
132 2.58
133 2.66
134 2.58
135 2.66
136 2.57
137 2.65
138 2.57
139 2.66
140 2.57
141 2.65
142 2.56
143 2.65
144 2.56
145 2.63
146 2.56
147 2.63
148 2.55
149 2.63
150 2.55
151 2.62
152 2.55
153 2.62
154 2.55
155 2.62
156 2.55
157 2.62
158 2.54
159 2.61
160 2.54
161 2.61
162 2.53
163 2.61
164 2.53
165 2.6
166 2.53
167 2.6
168 2.51
169 2.59
170 2.52
171 2.59
172 2.53
173 2.59
174 2.52
175 2.58
176 2.51
177 2.58
178 2.51
179 2.58
180 2.5
181 2.57
182 2.5
183 2.57
184 2.51
185 2.57
186 2.51
187 2.56
188 2.5
189 2.57
190 2.5
191 2.57
192 2.49
193 2.56
194 2.5
195 2.56
196 2.5
197 2.55
198 2.5
199 2.55
200 2.49
```

References

[1] Araújo Santos, P. and Fraga Alves, M.I., 2010. A New Class of Independence Tests for Interval Forecasts Evaluation. Submitted to Computational Statistics & Data Analysis.

[2] Berkowitz, J., Christoffersen P., Pelletier D., 2009. Evaluating Value-at-Risk models with desk-level data, Management Science, Published online in Articles in Advance.

[3] Bontemps, C. and Meddahi, N., 2008. Testing distributional assumptions: A GMM approach, Working Paper.

[4] Candelon, B., Colletaz, G., Hurlin, C., and Tokpavi, S., 2008. Backtesting value-at-Risk: A GMM Duration-Based Test, HAL, Working Paper.

[5] Christoffersen, P. and Pelletier, D., 2004. Backtesting Value-At-Risk: A Duration-Based Approach, Journal of Financial Econometrics, 2,1,84-108.

[6] Christoffersen, P., 1998. Evaluating Intervals Forecasts, International Economic Review, 39, 841-862.

[7] Danielsson, J. and Morimoto, Y., 2000. Forecasting Extreme Financial Risk: A Critical Analysis of Practical Methods for the Japanese Market, Monetary and Economic Studies, 18(2), 25-48.

[8] Wuertz, D., Chalabi, Y. and Miklovic, M., 2008. fGarch: Rmetrics - Autoregressive Conditional Heteroskedastic Modelling, R package version 290.76. http://www.rmetrics.org.

[9] Engel, R.F. and Manganelli, S., 2004. CAViaR: Conditional Autoregressive Value-at-Risk by Regression Quantiles, Journal of Business and Economics Statistics, 22, 367-381.

[10] Haas, M., 2005. Improved duration-based backtesting of Value-at-Risk, Journal of Risk, 8(2), 17-36.

[11] R Development Core Team, 2008. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org.