Code

binary.com 面试试题 I - GARCH模型中的*ARCH in Mean*

®γσ, Lian Hu ®

2018-10-17

1 简介

binary.com 面试试题 I - GARCH模型中的`ARIMA(p,d,q)`参数最优化添加了季节性和比较模型精准性。目前还测试下`archm=TRUE`是否会更精准，详情请参阅[问答] 请问怎样用R语言产生arch, arch-m, garch, garch-m的随机数？ 。

Hide

```
suppressPackageStartupMessages(require('BBmisc'))

## 读取程序包
pkg <- c('lubridate', 'plyr', 'dplyr', 'magrittr', 'stringr', 'rugarch', 'forecas
t', 'quantmod', 'memoise', 'microbenchmark', 'knitr', 'kableExtra', 'formattable')
suppressAll(lib(pkg))

funs <- c('task_progress.R') %>% paste0('function/', .)
l_ply(funs, source)
rm(pkg, funs)
```

2 数据

首先读取Binary.com Interview Q1 (Extention)的汇市数据。

Hide

```
cr_code <- c('AUDUSD=X', 'EURUSD=X', 'GBPUSD=X', 'CHF=X', 'CAD=X',
             'CNY=X', 'JPY=X')

#'@ names(cr_code) <- c('AUDUSD', 'EURUSD', 'GBPUSD', 'USDCHF', 'USDCAD',
#'@                      'USDCNY', 'USDJPY')

names(cr_code) <- c('USDAUD', 'USDEUR', 'USDGBP', 'USDCHF', 'USDCAD', 'USDCNY', 'U
SDJPY')

price_type <- c('Op', 'Hi', 'Lo', 'Cl')

## 读取雅虎数据。
mbase <- sapply(names(cr_code), function(x) readRDS(paste0('./data/', x, '.rds'))
 %>% na.omit)
```

数据简介报告。

Hide

```
sapply(mbase, summary) %>%
  kable %>%
  kable_styling(bootstrap_options = c('striped', 'hover', 'condensed', 'responsiv
e')) %>%
  scroll_box(width = '100%', height = '400px')
```

USDAUD	USDEUR	USDGBP	USDCHF	USDCAD	USDCNY	USDJPY
Min. :2012-01-02	Min. :2012-01-02	Min. :2012-01-02	Min. :2012-01-02	Min. :2012-01-02	Min. :2012-01-02	Min. :2012-01 02
1st Qu.:2013-05-31	1st Qu.:2013-05-31	1st Qu.:2013-05-30	1st Qu.:2013-06-03	1st Qu.:2013-05-30	1st Qu.:2013-05-29	1st Qu.:2013 05-29
Median :2014-10-31	Median :2014-10-31	Median :2014-10-31	Median :2014-11-03	Median :2014-10-30	Median :2014-10-30	Median :2014-10 30

Mean :2014-10-31	Mean :2014-10-31	Mean :2014-10-30	Mean :2014-11-01	Mean :2014-10-30	Mean :2014-10-30	Mean :2014-10 30
3rd Qu.:2016-03-30	3rd Qu.:2016-03-30	3rd Qu.:2016-03-30	3rd Qu.:2016-03-31	3rd Qu.:2016-03-29	3rd Qu.:2016-03-30	3rd Qu.:2016 03-30
Max. :2017-08-30	Max. :2017-08-30	Max. :2017-08-30	Max. :2017-08-30	Max. :2017-08-30	Max. :2017-08-30	Max. :2017-08 30
Min. :0.925	Min. :0.7180	Min. :0.5830	Min. :0.8540	Min. :0.968	Min. :6.031	Min. : 76.18
1st Qu.:1.037	1st Qu.:0.7580	1st Qu.:0.6250	1st Qu.:0.9220	1st Qu.:1.029	1st Qu.:6.189	1st Qu.: 97.86
Median :1.148	Median :0.8080	Median :0.6460	Median :0.9540	Median :1.127	Median :6.284	Median :103.91
Mean :1.176	Mean :0.8285	Mean :0.6702	Mean :0.9504	Mean :1.167	Mean :6.365	Mean :103.71
3rd Qu.:1.322	3rd Qu.:0.8980	3rd Qu.:0.6950	3rd Qu.:0.9780	3rd Qu.:1.308	3rd Qu.:6.524	3rd Qu.:114.
Max. :1.458	Max. :0.9620	Max. :0.8310	Max. :1.0300	Max. :1.458	Max. :7.478	Max. :125.60
Min. :0.927	Min. :0.7190	Min. :0.5830	Min. :0.8710	Min. :0.971	Min. :6.040	Min. : 76.20
1st Qu.:1.042	1st Qu.:0.7610	1st Qu.:0.6260	1st Qu.:0.9250	1st Qu.:1.032	1st Qu.:6.195	1st Qu.: 98.29
Median :1.153	Median :0.8130	Median :0.6490	Median :0.9570	Median :1.131	Median :6.295	Median :104.19
Mean :1.181	Mean :0.8318	Mean :0.6732	Mean :0.9539	Mean :1.171	Mean :6.375	Mean :104.07
3rd Qu.:1.327	3rd Qu.:0.9020	3rd Qu.:0.6990	3rd Qu.:0.9810	3rd Qu.:1.313	3rd Qu.:6.529	3rd Qu.:114.
Max. :1.464	Max. :1.3150	Max. :1.5690	Max. :1.0330	Max. :1.469	Max. :7.481	Max. :125.82
Min. :0.921	Min. :0.7150	Min. :0.5820	Min. :0.7330	Min. :0.963	Min. :2.201	Min. : 76.05
1st Qu.:1.031	1st Qu.:0.7560	1st Qu.:0.6230	1st Qu.:0.9180	1st Qu.:1.026	1st Qu.:6.185	1st Qu.: 97.46
Median :1.142	Median :0.8050	Median :0.6440	Median :0.9500	Median :1.123	Median :6.270	Median :103.54
Mean :1.171	Mean :0.8256	Mean :0.6681	Mean :0.9469	Mean :1.164	Mean :6.355	Mean :103.32
3rd Qu.:1.316	3rd Qu.:0.8940	3rd Qu.:0.6920	3rd Qu.:0.9730	3rd Qu.:1.303	3rd Qu.:6.515	3rd Qu.:113.
Max. :1.447	Max. :0.9600	Max. :0.8270	Max. :1.0280	Max. :1.449	Max. :6.945	Max. :124.97
Min. :0.9253	Min. :0.7178	Min. :0.5827	Min. :0.8544	Min. :0.9683	Min. :6.031	Min. : 76.18
1st Qu.:1.0369	1st Qu.:0.7582	1st Qu.:0.6247	1st Qu.:0.9216	1st Qu.:1.0286	1st Qu.:6.190	1st Qu.: 97.85
Median :1.1478	Median :0.8081	Median :0.6463	Median :0.9538	Median :1.1263	Median :6.285	Median :103.93
Mean :1.1759	Mean :0.8285	Mean :0.6702	Mean :0.9504	Mean :1.1673	Mean :6.365	Mean :103.71
3rd Qu.:1.3216	3rd Qu.:0.8981	3rd Qu.:0.6952	3rd Qu.:0.9775	3rd Qu.:1.3076	3rd Qu.:6.524	3rd Qu.:114.
Max. :1.4575	Max. :0.9624	Max. :0.8306	Max. :1.0302	Max. :1.4578	Max. :6.960	Max. :125.63

Min. :0	Min. :0	Min. :0	Min. :0	Min. :0	Min. :0	Min. :0
1st Qu.:0	1st Qu.:0	1st Qu.:0	1st Qu.:0	1st Qu.:0	1st Qu.:0	1st Qu.:0
Median :0	Median :0	Median :0	Median :0	Median :0	Median :0	Median :
Mean :0	Mean :0	Mean :0	Mean :0	Mean :0	Mean :0	Mean :0
3rd Qu.:0	3rd Qu.:0	3rd Qu.:0	3rd Qu.:0	3rd Qu.:0	3rd Qu.:0	3rd Qu.:0
Max. :0	Max. :0	Max. :0	Max. :0	Max. :0	Max. :0	Max. :0
Min. :0.9253	Min. :0.7178	Min. :0.5827	Min. :0.8544	Min. :0.9683	Min. :6.031	Min. : 76.18
1st Qu.:1.0369	1st Qu.:0.7582	1st Qu.:0.6247	1st Qu.:0.9216	1st Qu.:1.0286	1st Qu.:6.190	1st Qu.: 97.85
Median :1.1478	Median :0.8081	Median :0.6463	Median :0.9538	Median :1.1263	Median :6.285	Median :103.93
Mean :1.1759	Mean :0.8285	Mean :0.6702	Mean :0.9504	Mean :1.1673	Mean :6.365	Mean :103.71
3rd Qu.:1.3216	3rd Qu.:0.8981	3rd Qu.:0.6952	3rd Qu.:0.9775	3rd Qu.:1.3076	3rd Qu.:6.524	3rd Qu.:114.
Max. :1.4575	Max. :0.9624	Max. :0.8306	Max. :1.0302	Max. :1.4578	Max. :6.960	Max. :125.63

桌面2.1：数据简介。

3 统计建模

3.1 ARCH in Mean

Hide

```
opt_arma <- function(mbase){
  #ARMA Modeling minimum AIC value of `p,d,q`
  fit <- auto.arima(mbase)
  arimaorder(fit)
  }
```

再来就设置 `mean.model` 模型中的参数为 `archm = TRUE`。

Hide

```
calc_fx <- memoise(function(mbase, currency = 'JPY=X', ahead = 1, price = 'Cl') {

  source('function/filterFX.R')
  source('function/opt_arma.R')

  mbase = suppressWarnings(filterFX(mbase, currency = currency, price = price))
  armaOrder = opt_arma(mbase)

  spec = ugarchspec(
    variance.model = list(
      model = 'gjrGARCH', garchOrder = c(1, 1),
      submodel = NULL, external.regressors = NULL,
      variance.targeting = FALSE),
    mean.model = list(
      armaOrder = armaOrder[c(1, 3)],
      include.mean = TRUE, archm = TRUE,
      archpow = 1, arfima = TRUE,
      external.regressors = NULL,
      archex = FALSE),
    fixed.pars = list(arfima = armaOrder[2]),
    distribution.model = 'snorm')

  fit = ugarchfit(spec, mbase, solver = 'hybrid')

  fc = ugarchforecast(fit, n.ahead = ahead)
  res = tail(attributes(fc)$forecast$seriesFor, 1)
  colnames(res) = names(mbase)
  latestPrice = tail(mbase, 1)

  latestPrice <- xts(latestPrice)

  return(list(latestPrice = latestPrice, forecastPrice = res,
              AIC = infocriteria(fit)))
  })
```

4 模拟数据

4.1 回测数据

以下僕运行数据测试后事先储存，然后直接读取。首先过滤`timeID`时间参数，然后才模拟预测汇价。

Hide

```
timeID <- llply(mbase, function(x) as.character(index(x))) %>%
  unlist %>% unique %>% as.Date %>% sort
timeID <- c(timeID, xts::last(timeID) + days(1))
timeID0 <- ymd('2013-01-01')
timeID <- timeID[timeID >= timeID0]
```

模拟`calc_fx()`函数预测汇价数据。

Hide

```
## ------------ 模拟calc_fx()预测汇价 ---------------------
pred3 <- list()

for (dt in timeID) {

  for (i in seq(cr_code)) {

    smp <- mbase[[names(cr_code)[i]]]
    dtr <- xts::last(index(smp[index(smp) < dt]), 1)
    smp <- smp[paste0(dtr %m-% years(1), '/', dtr)]

    pred3[[i]] <- ldply(price_type, function(y) {
      df = calc_fx(smp, currency = cr_code[i], price = y)
      df = data.frame(Date = index(df[[1]][1]),
                      Type = paste0(names(df[[1]]), '.', y),
                      df[[1]], df[[2]], t(df[[3]]))
      names(df)[4] %<>% str_replace_all('1', 'T+1')
      df
    })

    if (!dir.exists(paste0('data/fx/', names(pred3[[i]])[3])))
      dir.create(paste0('data/fx/', names(pred3[[i]])[3]))

    saveRDS(pred3[[i]], paste0(
      'data/fx/', names(pred3[[i]])[3], '/pred3.',
      unique(pred3[[i]]$Date), '.rds'))

    cat(paste0(
      'data/fx/', names(pred3[[i]])[3], '/pred3.',
      unique(pred3[[i]]$Date), '.rds saved!\n'))

    }; rm(i)
  }
```

4.2 查询进度

查询模拟测试进度的函数`task_progress()`如下。

Hide

```
## ------------ 查询缺失文件 ---------------------
## 查询缺失文件。
dts <- sapply(mbase, function(x) {
  y = index(x)
  y[y >= timeID0]
  })

#'@ sapply(mbase, function(x) as.character(index(x)) %>% as.Date %>% sort)

fls <- sapply(names(cr_code), function(x) {
   fls <- list.files(paste0('./data/fx/', x), pattern = '^pred3') %>%
     str_extract_all('[0-9]{4}-[0-9]{2}-[0-9]{2}') %>%
     unlist %>% as.Date %>% sort
   dts[[x]][!dts[[x]] %in% fls] %>% sort
   })
  })

#'@ timeID <- sapply(fls, function(x) timeID[!timeID %in% x] %>% sort)

#'@ timeID <- llply(fls, function(x) timeID[!timeID %in% x] %>% sort) %>% unlist %
>% as.Date %>% sort
#'@ names(timeID) <- NULL
#'@ timeID %<>% unique
```

Hide

```
## ------------ 模拟calc_fx()预测汇价 ---------------------
pred3 <- list()

for (i in seq(cr_code)) {

  timeIDi <- fls[[names(cr_code)[i]]]
  for (dt in timeIDi) {

    smp <- mbase[[names(cr_code)[i]]]
    dtr <- xts::last(index(smp[index(smp) < dt]), 1)
    smp <- smp[paste0(dtr %m-% years(1), '/', dtr)]

    pred3[[i]] <- ldply(price_type, function(y) {
      df = calc_fx(smp, currency = cr_code[i], price = y)
      df = data.frame(Date = index(df[[1]][1]),
                      Type = paste0(names(df[[1]]), '.', y),
                      df[[1]], df[[2]], t(df[[3]]))
      names(df)[4] %<>% str_replace_all('1', 'T+1')
      df
    })

    if (!dir.exists(paste0('data/fx/', names(pred3[[i]])[3])))
      dir.create(paste0('data/fx/', names(pred3[[i]])[3]))

    saveRDS(pred3[[i]], paste0(
      'data/fx/', names(pred3[[i]])[3], '/pred3.',
      unique(pred3[[i]]$Date), '.rds'))

    cat(paste0(
      'data/fx/', names(pred3[[i]])[3], '/pred3.',
      unique(pred3[[i]]$Date), '.rds saved!\n'))

    }
  }; rm(i)
```

模拟完毕后，再来就查看数据结果。

Hide

```
## calc_fx()模拟数据误差率
task_progress(.pattern = '^pred3', .loops = FALSE)
```

```
## Current Tokyo Time : 2018-10-17 14:15:15
## ^pred3
##
##       .id    x    n progress
## 1 USDAUD 1088 1215   89.55%
## 2 USDEUR 1033 1215   85.02%
## 3 USDGBP 1037 1216   85.28%
## 4 USDCHF 1072 1215   88.23%
## 5 USDCAD 1033 1214   85.09%
## 6 USDCNY 1019 1215   83.87%
## 7 USDJPY  929 1215   76.46%
##
## =============== 84.79% ===============
```

以上结果显示，模拟后的数据的误差率非常渺小[1]。以下筛选 pred1 、 pred2 与 pred3 同样日期的有效数据。

Hide

```
##数据1
fx1 <- llply(names(cr_code), function(x) {
    fls <- list.files(paste0('data/fx/', x), pattern = '^pred1')
    dfm <- ldply(fls, function(y) {
        readRDS(paste0('data/fx/', x, '/', y))
    }) %>% data.frame(Cat = 'pred1', .) %>% tbl_df
    names(dfm)[4:5] <- c('Price', 'Price.T1')
    dfm
 })
names(fx1) <- names(cr_code)

##数据2
fx2 <- llply(names(cr_code), function(x) {
    fls <- list.files(paste0('data/fx/', x), pattern = '^pred2')
    dfm <- ldply(fls, function(y) {
        readRDS(paste0('data/fx/', x, '/', y))
    }) %>% data.frame(Cat = 'pred2', .) %>% tbl_df
    names(dfm)[4:5] <- c('Price', 'Price.T1')
    dfm
 })
names(fx2) <- names(cr_code)

##数据3
fx3 <- llply(names(cr_code), function(x) {
    fls <- list.files(paste0('data/fx/', x), pattern = '^pred3')
    dfm <- ldply(fls, function(y) {
        readRDS(paste0('data/fx/', x, '/', y))
    }) %>% data.frame(Cat = 'pred3', .) %>% tbl_df
    names(dfm)[4:5] <- c('Price', 'Price.T1')
    dfm
 })
names(fx3) <- names(cr_code)

#合并，并且整理数据。
fx1 %<>% ldply %>% tbl_df
fx2 %<>% ldply %>% tbl_df
fx3 %<>% ldply %>% tbl_df
fx <- suppressAll(bind_rows(fx1, fx2, fx3) %>% arrange(Date) %>%
  mutate(.id = factor(.id), Cat = factor(Cat)) %>%
  ddply(.(Cat, Type), function(x) {
    x %>% mutate(Price.T1 = lag(Price.T1, 1))
  }) %>% tbl_df %>%
    dplyr::filter(Date >= ymd('2013-01-01') & Date <= ymd('2017-08-30')))

rm(fx1, fx2, fx3)
```

Hide

```
## filter all predictive error where sd >= 20%.
notID <- fx %>% mutate(diff = abs(Price.T1/Price), se = ifelse(diff <= 0.8 | diff
 >= 1.25, 1, 0)) %>% dplyr::filter(se == 1)
ntimeID <- notID %>% .$Date %>% unique
notID %>%
  kable(caption = 'Error data') %>%
  kable_styling(bootstrap_options = c('striped', 'hover', 'condensed', 'responsiv
e')) %>%
  scroll_box(width = '100%', height = '400px')
```

Error data

.id	Cat	Date	Type	Price	Price.T1	Akaike
USDCHF	pred1	2015-07-28	USDCHF.Op	0.96200	-1.674244e+03	-6.8550360
USDCHF	pred1	2015-01-15	USDCHF.Lo	0.73300	1.016502e+00	-5.7998611
USDCNY	pred1	2014-07-10	USDCNY.Lo	2.20100	6.186397e+00	-2.7132659
USDCNY	pred1	2014-07-14	USDCNY.Lo	6.19600	1.294872e+00	-3.1503936
USDJPY	pred1	2013-06-27	USDJPY.Op	98.46800	7.839498e+01	4.7132866
USDJPY	pred1	2013-06-30	USDJPY.Op	99.41100	7.839644e+01	4.7356084

僕尝试运行好几次，USDCHF 都是获得同样的结果。然后将默认的 snorm 分布更换为 norm 就没有出现错误。至于 USDCNY 原始数据有误就不是统计模型的问题了。

Hide

```
## timeID which contains 3 prediction models.
ntimeID <- fx %>%
utimeID <- fx %>%
  ddply(.(Date), summarize,
        n = n()) %>%
  dplyr::filter(n == 84) %>%
  tbl_df %>% .$Date

fx %<>% dplyr::filter(Date %in% utimeID, !Date %in% ntimeID)
```

4.3 精准度

现在就比较下双方的MSE值与AIC值。

Hide

```
acc <- ddply(fx, .(Cat, Type), summarise,
             MSE = mean((Price.T1 - Price)^2, na.rm = TRUE),
             n = length(Price),
             Akaike.MSE = (-2*MSE)/n+2*4/n,
             Akaike = mean(Akaike, na.rm = TRUE),
             Bayes = mean(Bayes, na.rm = TRUE),
             Shibata = mean(Shibata, na.rm = TRUE),
             Hannan.Quinn = mean(Hannan.Quinn, na.rm = TRUE)) %>%
  tbl_df %>% mutate(MSE = round(MSE, 6)) %>%
  arrange(Type)

acc %>%
  kable(caption = 'Group Table Summary') %>%
  kable_styling(bootstrap_options = c('striped', 'hover', 'condensed', 'responsive')) %>%
  group_rows('USD/AUD Open', 1, 3, label_row_css = 'background-color: #e68a00; color: #fff;') %>%
  group_rows('USD/AUD High', 4, 6, label_row_css = 'background-color: #e68a00; color: #fff;') %>%
  group_rows('USD/AUD Low', 7, 9, label_row_css = 'background-color: #e68a00; color: #fff;') %>%
  group_rows('USD/AUD Close', 10, 12, label_row_css = 'background-color: #e68a00; color: #fff;') %>%
  group_rows('USD/EUR Open', 13, 15, label_row_css = 'background-color: #6666ff; color: #fff;') %>%
  group_rows('USD/EUR High', 16, 18, label_row_css = 'background-color: #6666ff; color: #fff;') %>%
  group_rows('USD/EUR Low', 19, 21, label_row_css = 'background-color:#6666ff; color: #fff;') %>%
  group_rows('USD/EUR Close', 22, 24, label_row_css = 'background-color: #6666ff; color: #fff;') %>%
  group_rows('USD/GBP Open', 25, 27, label_row_css = 'background-color: #339966; color: #fff;') %>%
  group_rows('USD/GBP High', 28, 30, label_row_css = 'background-color: #339966; color: #fff;') %>%
  group_rows('USD/GBP Low', 31, 33, label_row_css = 'background-color: #339966; color: #fff;') %>%
  group_rows('USD/GBP Close', 34, 36, label_row_css = 'background-color: #339966; color: #fff;') %>%
  group_rows('USD/CHF Open', 37, 39, label_row_css = 'background-color: #808000; color: #fff;') %>%
  group_rows('USD/CHF High', 40, 42, label_row_css = 'background-color: #808000; color: #fff;') %>%
  group_rows('USD/CHF Low', 43, 45, label_row_css = 'background-color: #808000; color: #fff;') %>%
  group_rows('USD/CHF Close', 46, 48, label_row_css = 'background-color: #808000; color: #fff;') %>%
  group_rows('USD/CAD Open', 49, 51, label_row_css = 'background-color: #666; color: #fff;') %>%
  group_rows('USD/CAD High', 52, 54, label_row_css = 'background-color: #666; color: #fff;') %>%
  group_rows('USD/CAD Low', 55, 57, label_row_css = 'background-color: #666; color: #fff;') %>%
  group_rows('USD/CAD Close', 58, 60, label_row_css = 'background-color: #666; color: #fff;') %>%
  group_rows('USD/CNY Open', 61, 63, label_row_css = 'background-color: #e60000; color: #fff;') %>%
  group_rows('USD/CNY High', 64, 66, label_row_css = 'background-color: #e60000; color: #fff;') %>%
  group_rows('USD/CNY Low', 67, 69, label_row_css = 'background-color: #e60000; color: #fff;') %>%
  group_rows('USD/CNY Close', 70, 72, label_row_css = 'background-color: #e60000; color: #fff;') %>%
  group_rows('USD/JPY Open', 73, 75, label_row_css = 'background-color: #ff3377; color: #fff;') %>%
  group_rows('USD/JPY High', 76, 78, label_row_css = 'background-color: #ff3377; c
olor: #fff;') %>%
  group_rows('USD/JPY Low', 79, 81, label_row_css = 'background-color: #ff3377; co
lor: #fff;') %>%
  group_rows('USD/JPY Close', 82, 84, label_row_css = 'background-color: #ff3377;
 color: #fff;') %>%
  scroll_box(width = '100%', height = '400px')
```

Group Table Summary

Cat	Type	MSE	n	Akaike.MSE	Akaike	Bayes	
USD/AUD Open							
pred1	USDAUD.Op	0.001485	694	0.0115231	-6.1003165	-5.9788352	-6
pred2	USDAUD.Op	0.000067	694	0.0115272	-6.8926533	-6.8017675	-6
pred3	USDAUD.Op	0.000069	694	0.0115272	-6.8224982	-6.7179646	-6
USD/AUD High							
pred1	USDAUD.Hi	0.001669	694	0.0115226	-6.2583349	-6.1291102	-6
pred2	USDAUD.Hi	0.000055	694	0.0115272	-7.0515969	-6.9619065	-7
pred3	USDAUD.Hi	0.000057	694	0.0115272	-5.6866132	-5.5832749	-5
USD/AUD Low							
pred1	USDAUD.Lo	0.000966	694	0.0115246	-6.5677746	-6.4385346	-6
pred2	USDAUD.Lo	0.000053	694	0.0115272	-7.0612497	-6.9695477	-7
pred3	USDAUD.Lo	0.000057	694	0.0115272	-7.0154288	-6.9100791	-7
USD/AUD Close							
pred1	USDAUD.Cl	0.001873	694	0.0115220	-6.0162413	-5.8954588	-6
pred2	USDAUD.Cl	0.000066	694	0.0115272	-6.8988588	-6.8077969	-6
pred3	USDAUD.Cl	0.000069	694	0.0115272	-6.8937281	-6.7890184	-6
USD/EUR Open							
pred1	USDEUR.Op	0.000368	694	0.0115263	-7.3338283	-7.2020527	-7
pred2	USDEUR.Op	0.000023	694	0.0115273	-7.9057246	-7.8131176	-7
pred3	USDEUR.Op	0.000026	694	0.0115273	-3.2266234	-3.1203687	-3
USD/EUR High							
pred1	USDEUR.Hi	0.000358	694	0.0115263	-7.1208509	-6.9949044	-7
pred2	USDEUR.Hi	0.000018	694	0.0115273	-8.1106028	-8.0198277	-8
pred3	USDEUR.Hi	0.000020	694	0.0115273	-7.7134003	-7.6089775	-7
USD/EUR Low							
pred1	USDEUR.Lo	0.000072	694	0.0115272	-7.8257307	-7.6860585	-7
pred2	USDEUR.Lo	0.000020	694	0.0115273	-8.0539791	-7.9609034	-8
pred3	USDEUR.Lo	0.000026	694	0.0115273	-6.3308221	-6.2240987	-6
USD/EUR Close							
pred1	USDEUR.Cl	0.000557	694	0.0115258	-7.3564598	-7.2230143	-7
pred2	USDEUR.Cl	0.000023	694	0.0115273	-7.9039691	-7.8117162	-7
pred3	USDEUR.Cl	0.000026	694	0.0115273	-5.7336380	-5.6277373	-5
USD/GBP Open							
pred1	USDGBP.Op	0.000106	694	0.0115271	-8.0967421	-7.9545923	-8
pred2	USDGBP.Op	0.000020	694	0.0115273	-8.4221013	-8.3242304	-8
pred3	USDGBP.Op	0.000020	694	0.0115273	-4.2024020	-4.0908996	-4
USD/GBP High							
pred1	USDGBP.Hi	0.000477	694	0.0115260	-7.1519345	-7.0420252	-7
pred2	USDGBP.Hi	0.000014	694	0.0115273	-8.5515114	-8.4560209	-8

pred3	USDGBP.Hi	0.000016	694	0.0115273	-6.8351756	-6.7260536	-6
USD/GBP Low							
pred1	USDGBP.Lo	0.000321	694	0.0115265	-7.6996366	-7.5775927	-7
pred2	USDGBP.Lo	0.000015	694	0.0115273	-8.6241601	-8.5288847	-8
pred3	USDGBP.Lo	0.000017	694	0.0115273	-6.6693291	-6.5604223	-6
USD/GBP Close							
pred1	USDGBP.Cl	0.000134	694	0.0115270	-7.8141090	-7.6833530	-7
pred2	USDGBP.Cl	0.000020	694	0.0115273	-8.4297825	-8.3355701	-8
pred3	USDGBP.Cl	0.000020	694	0.0115273	-5.3457457	-5.2379019	-5
USD/CHF Open							
pred1	USDCHF.Op	0.000099	694	0.0115271	-7.2484023	-7.1009514	-7
pred2	USDCHF.Op	0.000026	694	0.0115273	-7.5298151	-7.4369056	-7
pred3	USDCHF.Op	0.000027	694	0.0115273	-7.5027593	-7.3962012	-7
USD/CHF High							
pred1	USDCHF.Hi	0.000199	694	0.0115268	-7.2548707	-7.1099220	-7
pred2	USDCHF.Hi	0.000020	694	0.0115273	-7.6867609	-7.5900954	-7
pred3	USDCHF.Hi	0.000021	694	0.0115273	-4.9413177	-4.8310035	-4
USD/CHF Low							
pred1	USDCHF.Lo	0.000348	694	0.0115264	-6.8005222	-6.6817579	-6
pred2	USDCHF.Lo	0.000028	694	0.0115273	-7.6184651	-7.5252301	-7
pred3	USDCHF.Lo	0.000031	694	0.0115273	-7.5757600	-7.4688764	-7
USD/CHF Close							
pred1	USDCHF.Cl	0.000124	694	0.0115270	-7.2419650	-7.0947266	-7
pred2	USDCHF.Cl	0.000026	694	0.0115273	-7.5324416	-7.4378755	-7
pred3	USDCHF.Cl	0.000026	694	0.0115273	-3.9698805	-3.8616657	-3
USD/CAD Open							
pred1	USDCAD.Op	0.001082	694	0.0115243	-7.1202839	-6.9760318	-7
pred2	USDCAD.Op	0.000036	694	0.0115273	-7.5436211	-7.4514271	-7
pred3	USDCAD.Op	0.000039	694	0.0115273	-5.2476659	-5.1418206	-5
USD/CAD High							
pred1	USDCAD.Hi	0.000275	694	0.0115266	-7.4051971	-7.2539700	-7
pred2	USDCAD.Hi	0.000035	694	0.0115273	-7.6777426	-7.5748559	-7
pred3	USDCAD.Hi	0.000038	694	0.0115273	-3.4295198	-3.3129817	-3
USD/CAD Low							
pred1	USDCAD.Lo	0.001172	694	0.0115240	-7.0520764	-6.9033640	-7
pred2	USDCAD.Lo	0.000031	694	0.0115273	-7.6038382	-7.5119686	-7
pred3	USDCAD.Lo	0.000035	694	0.0115273	-4.6765398	-4.5710188	-4
USD/CAD Close							
pred1	USDCAD.Cl	0.000856	694	0.0115249	-7.1394047	-6.9954283	-7
pred2	USDCAD.Cl	0.000035	694	0.0115273	-7.5537405	-7.4609575	-7
pred3	USDCAD.Cl	0.000038	694	0.0115273	-3.8975181	-3.7910838	-3
USD/CNY Open							
pred1	USDCNY.Op	0.004577	694	0.0115142	-5.3262782	-5.1937738	-5
pred2	USDCNY.Op	0.001280	694	0.0115237	-6.0712923	-5.9714101	-6
pred3	USDCNY.Op	0.001739	694	0.0115224	-3.2805050	-3.1669812	-3
USD/CNY High							

pred1	USDCNY.Hi	0.006446	694	0.0115088	-5.5772166	-5.4386976	-5
pred2	USDCNY.Hi	0.001228	694	0.0115238	-5.8947789	-5.7889571	-5
pred3	USDCNY.Hi	0.000823	694	0.0115250	0.4844599	0.6039232	0
USD/CNY Low							
pred1	USDCNY.Lo	0.013831	694	0.0114875	-5.4334053	-5.3005227	-5
pred2	USDCNY.Lo	0.000449	694	0.0115261	-6.3281187	-6.2292034	-6
pred3	USDCNY.Lo	0.001241	694	0.0115238	-5.3583923	-5.2458355	-5
USD/CNY Close							
pred1	USDCNY.Cl	0.002978	694	0.0115188	-5.6155852	-5.4788612	-5
pred2	USDCNY.Cl	0.000185	694	0.0115268	-6.3803468	-6.2751058	-6
pred3	USDCNY.Cl	0.000223	694	0.0115267	-2.6275465	-2.5086639	-2
USD/JPY Open							
pred1	USDJPY.Op	0.849792	694	0.0090784	2.0366510	2.2006171	2
pred2	USDJPY.Op	0.463180	694	0.0101926	2.0178465	2.1131252	2
pred3	USDJPY.Op	0.596356	694	0.0098088	3.2665740	3.3754942	3
USD/JPY High							
pred1	USDJPY.Hi	5.072197	694	-0.0030899	2.3205292	2.4541433	2
pred2	USDJPY.Hi	0.350109	694	0.0105184	1.7719840	1.8733588	1
pred3	USDJPY.Hi	0.557077	694	0.0099220	5.4397869	5.5548033	5
USD/JPY Low							
pred1	USDJPY.Lo	2.765424	694	0.0035579	2.3864636	2.5342943	2
pred2	USDJPY.Lo	0.461756	694	0.0101967	1.9675156	2.0588424	1
pred3	USDJPY.Lo	0.695329	694	0.0095235	4.2116218	4.3165901	4
USD/JPY Close							
pred1	USDJPY.Cl	0.604835	694	0.0097843	2.0399015	2.2052227	2
pred2	USDJPY.Cl	0.468307	694	0.0101778	2.0249439	2.1205556	2
pred3	USDJPY.Cl	0.598571	694	0.0098024	2.9466791	3.0559323	2

Hide

```
acc <- ddply(fx, .(Cat, .id), summarise, 
             MSE = mean((Price.T1 - Price)^2, na.rm = TRUE), 
             n = length(Price), 
             Akaike.MSE = (-2*MSE)/n+2*4/n, 
             Akaike = mean(Akaike, na.rm = TRUE), 
             Bayes = mean(Bayes, na.rm = TRUE), 
             Shibata = mean(Shibata, na.rm = TRUE), 
             Hannan.Quinn = mean(Hannan.Quinn, na.rm = TRUE)) %>% 
  tbl_df %>% mutate(MSE = round(MSE, 6)) %>% 
  arrange(.id)

acc %>% 
  kable(caption = 'Group Table Summary') %>% 
  kable_styling(bootstrap_options = c('striped', 'hover', 'condensed', 'responsive')) %>% 
  group_rows('USD/AUD', 1, 3, label_row_css = 'background-color: #003399; color: #fff;') %>% 
  group_rows('USD/CAD', 4, 6, label_row_css = 'background-color: #003399; color: #fff;') %>% 
  group_rows('USD/CHF', 7, 9, label_row_css = 'background-color: #003399; color: #fff;') %>% 
  group_rows('USD/CNY', 10, 12, label_row_css = 'background-color: #003399; color: #fff;') %>% 
  group_rows('USD/EUR', 13, 15, label_row_css = 'background-color: #003399; color: #fff;') %>% 
  group_rows('USD/GBP', 16, 18, label_row_css = 'background-color: #003399; color: #fff;') %>% 
  group_rows('USD/JPY', 19, 21, label_row_css = 'background-color: #003399; color: #fff;') %>% 
  scroll_box(width = '100%', height = '400px')
```

Group Table Summary

Cat	.id	MSE	n	Akaike.MSE	Akaike	Bayes	Shib
USD/AUD							
pred1	USDAUD	0.001498	2776	0.0028808	-6.235667	-6.110485	-6.238
pred2	USDAUD	0.000060	2776	0.0028818	-6.976090	-6.885255	-6.977
pred3	USDAUD	0.000063	2776	0.0028818	-6.604567	-6.500084	-6.606
USD/CAD							
pred1	USDCAD	0.000846	2776	0.0028812	-7.179240	-7.032198	-7.182
pred2	USDCAD	0.000034	2776	0.0028818	-7.594736	-7.499802	-7.596
pred3	USDCAD	0.000038	2776	0.0028818	-4.312811	-4.204226	-4.314
USD/CHF							
pred1	USDCHF	0.000193	2776	0.0028817	-7.136440	-6.996840	-7.139
pred2	USDCHF	0.000025	2776	0.0028818	-7.591871	-7.497527	-7.593
pred3	USDCHF	0.000026	2776	0.0028818	-5.997429	-5.889437	-5.999
USD/CNY							
pred1	USDCNY	0.006958	2776	0.0028768	-5.488121	-5.352964	-5.491
pred2	USDCNY	0.000786	2776	0.0028813	-6.168634	-6.066169	-6.170
pred3	USDCNY	0.001034	2776	0.0028811	-2.695496	-2.579389	-2.697
USD/EUR							
pred1	USDEUR	0.000339	2776	0.0028816	-7.409217	-7.276508	-7.412
pred2	USDEUR	0.000021	2776	0.0028818	-7.993569	-7.901391	-7.994
pred3	USDEUR	0.000024	2776	0.0028818	-5.751121	-5.645295	-5.752
USD/GBP							
pred1	USDGBP	0.000260	2776	0.0028817	-7.690606	-7.564391	-7.693
pred2	USDGBP	0.000017	2776	0.0028818	-8.506889	-8.411176	-8.508
pred3	USDGBP	0.000018	2776	0.0028818	-5.763163	-5.653819	-5.765
USD/JPY							
pred1	USDJPY	2.323062	2776	0.0012082	2.195886	2.348569	2.192
pred2	USDJPY	0.435838	2776	0.0025678	1.945572	2.041470	1.944
pred3	USDJPY	0.613356	2776	0.0024399	3.966165	4.075705	3.964

Hide

```
acc <- ddply(fx, .(Cat), summarise,
             MSE = mean((Price.T1 - Price)^2, na.rm = TRUE),
             n = length(Price),
             Akaike.MSE = (-2*MSE)/n+2*4/n,
             Akaike = mean(Akaike, na.rm = TRUE),
             Bayes = mean(Bayes, na.rm = TRUE),
             Shibata = mean(Shibata, na.rm = TRUE),
             Hannan.Quinn = mean(Hannan.Quinn, na.rm = TRUE)) %>%
  tbl_df %>% mutate(MSE = round(MSE, 6))

acc %>%
  kable(caption = 'Group Table Summary') %>%
  kable_styling(bootstrap_options = c('striped', 'hover', 'condensed', 'responsiv
e'))
```

Group Table Summary

Cat	MSE	n	Akaike.MSE	Akaike	Bayes	Shibata	Hannan.Quinn
pred1	0.333308	19432	0.0003774	-5.563344	-5.426402	-5.566386	-5.508300

pred2	0.062397	19432	0.0004053	-6.126602	-6.031407	-6.128028	-6.088338
pred3	0.088525	19432	0.0004026	-3.879775	-3.770935	-3.881614	-3.836026

5 结论

结果证明`pred2`的`archm=FALSE`最为精准。目前正在编写着Q1App2自动交易应用。“商场如战场”，除了模式最优化以外，程序运作上分秒必争… `microbenchmark`测试效率，之前编写了个DataCollection应用采集实时数据以方便之后的高频率交易自动化建模[2]。欲知更多详情，请参阅Real Time FXCM。

6 附录

6.1 文件与系统资讯

以下乃此文献资讯：

- 文件建立日期：2018-10-14
- 文件最新更新日期：2018-10-17
- R version 3.5.1 (2018-07-02)
- R语言版本：3.5.1
- **rmarkdown** 程序包版本：1.10
- 文件版本：1.0.1
- 作者简历：®γσ, Eng Lian Hu
- GitHub：源代码
- 其它系统资讯：

Additional session information:

Category	session_info	Category	Sys.info
version	R version 3.5.1 (2018-07-02)	sysname	Windows
system	x86_64, mingw32	release	10 x64
ui	RTerm	version	build 17134
language	en	nodename	RSTUDIO-SCIBROK
collate	Japanese_Japan.932	machine	x86-64
tz	Asia/Tokyo	login	scibr
date	2018-10-17	user	scibr
Current time	2018-10-17 14:15:33 JST	effective_user	scibr

6.2 参考文献

1. [问答] 请问怎样用R语言产生arch, arch-m, garch, garch-m的随机数？ 🔥
2. binary.com 面试试题 I - GARCH模型中的`ARIMA(p,d,q)`参数最优化

[1] 一些数据模拟时，出现不知名错误。↩

[2] 不过数据量多就会当机，得继续提升才行。↩