The R Trader

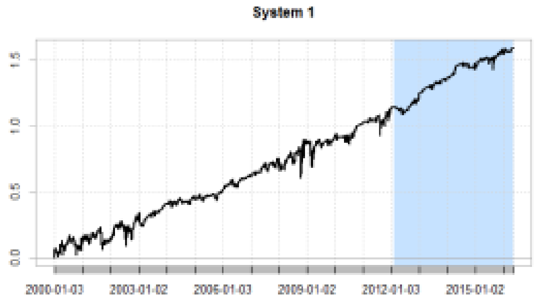
Using R and related tools in Quantitative Finance

Trading strategy: Making the most of the out of sample data

August 19, 2016, 9:26 am
12

When testing trading strategies a common approach is to divide the initial data set into **in sample** data: the part of the data designed to calibrate the model and **out of sample** data: the part of the data used to validate the calibration and ensure that the performance created in sample will be reflected in the real world. As a rule of thumb around 70% of the initial data can be used for calibration (i.e. in sample) and 30% for validation (i.e. out of sample). Then a comparison of the in and out of sample data help to decide whether the model is robust enough. This post aims at going a step further and provides a statistical method to decide whether the out of sample data is in line with what was created in sample.

In the chart below the blue area represents the out of sample performance for one of my strategies.



A simple visual inspection reveals a good fit between the in and out of sample performance but what degree of confidence do I have in this? At this stage not much and this is the issue. What is truly needed is a measure of similarity between the in and out of sample data sets. In statistical terms this could be translated as the likelihood that the in and out of sample performance figures coming from the same distribution. There is a non-parametric statistical test that does exactly this: the **Kruskall-Wallis Test**. A good definition of this test could be found on R-Tutor "A collection of data samples are independent if they come from unrelated populations and the samples do not affect each other. Using the Kruskal-Wallis Test, we can decide whether the population distributions are identical without assuming them to follow the normal distribution." The added benefit of this test is not assuming a normal distribution.

It exists other tests of the same nature that could fit into that framework. The **Mann-Whitney-Wilcoxon** test or the **Kolmogorov-Smirnov** tests would perfectly suits the framework describes here however this is beyond the scope of this article to discuss the pros and cons of each of these tests. A good description along with R examples can be found here.

Here's the code used to generate the chart above and the analysis:

```
#################################################
## Making the most of the OOS data
##
## thertrader@gmail.com - Aug. 2016
#################################################
library(xts)
library(PerformanceAnalytics)

thePath <- "myPath" #change this
theFile <- "data.csv"
```

```
data <- read.csv(paste0(thePath,theFile),header=TRUE,sep=",")
data <- xts(data[,2],order.by=as.Date(as.character(data[,1]),format = "%d/%m/%

##----- Strategy's Chart
par(mex=0.8,cex=1)
thePeriod <- c("2012-02/2016-05")
chart.TimeSeries(cumsum(data),
 main = "System 1",
 ylab="",
 period.areas = thePeriod,
 grid.color = "lightgray",
 period.color = "slategray1")

##----- Kruskal tests
pValue <- NULL
i <- 1
while (i < 1000){
 isSample <- sample(isData,length(osData))
 pValue <- rbind(pValue,kruskal.test(list(osData, isSample))$p.value)
 i <- i + 1
}

##----- Mean of p-values
mean(pValue)
```

In the example above the in sample period is longer than the out of sample period therefore I randomly created 1000 subsets of the in sample data each of them having the same length as the out of sample data. Then I tested each in sample subset against the out of sample data and I recorded the p-values. This process creates not a single p-value for the Kruskall-Wallis test but a distribution making the analysis more robust. In this example the mean of the p-values is well above zero (0.478) indicating that the null hypothesis should be accepted: there are strong evidences that the in and out of sample data is coming from the same distribution.

As usual what is presented in this post is a toy example that only scratches the surface of the problem and should be tailored to individual needs. However I think it proposes an interesting and rational statistical framework to evaluate out of sample results.

This post is inspired by the following two papers:

Vigier Alexandre, Chmil Swann (2007), "Effects of Various Optimization Functions on the Out of Sample Performance of Genetically Evolved Trading Strategies", Forecasting Financial Markets Conference

Vigier Alexandre, Chmil Swann (2010), « An optimization process to improve in/out of sample consistency, a Stock Market case», JP Morgan Cazenove Equity Quantitative Conference, London October 2010

◀ 1 2

Category: Data Science, Portfolio Management, R, Statistics, Trading Strategies | Comment (RSS) | Trackback

page: 2

BERT: a newcomer in the R Excel connection

November 30, 2016, 12:34 pm

A few months ago a reader point me out this new way of connecting R and Excel. I don't know for how long this has been around, but I never came across it and I've never seen any blog post or article about it. So I decided to write a post as the tool is really worth it and before anyone asks, I'm not related to the company in any way.

BERT stands for Basic Excel R Toolkit. It's free (licensed under the GPL v2) and it has been developed by Structured Data LLC. At the time of writing the current version of BERT is 1.07. More information can be found here. From a more technical perspective, BERT is designed to support running R functions from Excel spreadsheet cells. In Excel terms, it's for writing User-Defined Functions (UDFs) in R.

In this post I'm not going to show you how R and Excel interact via BERT. There are very good tutorials here, here and here. Instead I want to show you how I used BERT to build a "control tower" for my trading.

How do I use BERT?

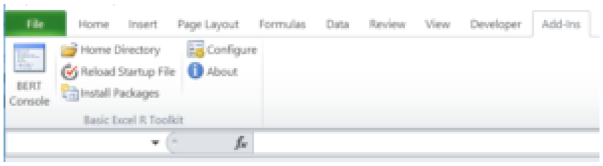
My trading signals are generated using a long list of R files but I need the flexibility of Excel to display results quickly and efficiently. As shown above BERT can do this for me but I also want to tailor the application to my needs. By combining the power of XML, VBA, R and BERT I can create a good looking yet powerful application in the form of an Excel file with minimum VBA code. Ultimately I have a single Excel file gathering all the necessary tasks to manage my portfolio: database update, signal generation, orders submission etc… My approach could be broken down in the 3 steps below:

1. Use XML to build user defined menus and buttons in an Excel file.
2. The above menus and buttons are essentially calls to VBA functions.
3. Those VBA functions are wrapup around R functions defined using BERT.

With this approach I can keep a clear distinction between the core of my code kept in R, SQL and Python and everything used to display and format results kept in Excel, VBA & XML. In the next sections I present the prerequisite to developed such an approach and a step by step guide that explains how BERT could be used for simply passing data from R to Excel with minimal VBA code.

Prerequisite

1 – *Download and install BERT from this link*. Once the installation has completed you should have a new Add-Ins menu in Excel with the buttons as shown below. This is how BERT materialized in Excel.



2 – *Download and install Custom UI editor*: The Custom UI Editor allows to create user defined menus and buttons in Excel ribbon. A step by step procedure is available here.

Step by step guide

1 – R Code: The below R function is a very simple piece of code for illustration purposes only. It calculates and return the residuals from a linear regression. This is what we want to retrieve in Excel. Save this in a file called **myRCode.R** (any other name is fine) in a directory of your choice.

```
myFunction <- function(){
 aa <- rnorm(200)
 bb <- rnorm(200)
 res <- lm(aa~bb)$res
 return(res)
}
```

2 – functions.R in BERT: From Excel select **Add-Ins -> Home Directory** and open the file called **functions.R**. In this file paste the following code. Make sure you insert the correct path.

```
source("D:\\myPath\\myRCode.R")
```

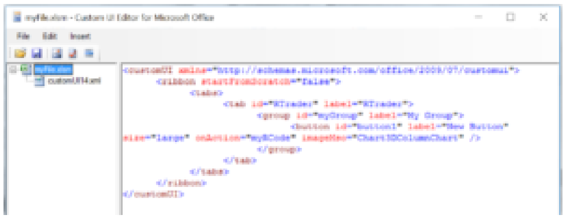
This is just sourcing into BERT the R file you created above. Then save and close the file **functions.R.** Should you want to make any change to the R file created in step 1 you will have to reload it using the BERT button "Reload Startup File" from the Add-Ins menu in Excel

3 – In Excel: Create and save a file called myFile.xslm (any other name is fine). This is a macro-enabled file that you save in the directory of your choice. Once the file is saved close it.

4 – Open the file created above in Custom UI editor: Once the file is open, paste the below code.

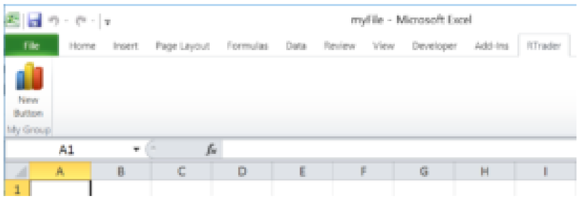
```
<customUI xmlns="http://schemas.microsoft.com/office/2009/07/customui">
 <ribbon startFromScratch="false">
 <tabs>
 <tab id="RTrader" label="RTrader">
 <group id="myGroup" label="My Group">
 <button id="button1" label="New Button" size="large" onAction="myRCode" imageMso="Chart3DColumnChart" />
 </group>
 </tab>
 </tabs>
 </ribbon>
</customUI>
```

You should have something like this in the XML editor:



```
<customUI xmlns="http://schemas.microsoft.com/office/2009/07/customui">
        <ribbon startFromScratch="false">
                <tabs>
                        <tab id="RTrader" label="RTrader">
                                <group id="myGroup" label="My Group">
                                        <button id="button1" label="New Button"
size="large" onAction="myRCode" imageMso="Chart3DColumnChart" />
                                </group>
                        </tab>
                </tabs>
        </ribbon>
</customUI>
```

Essentially this piece of XML code creates an additional menu (RTrader), a new group (My Group) and a user defined button (New Button) in the Excel ribbon. Once you're done, open myFile.xslm in Excel and close the Custom UI Editor. You should see something like this.

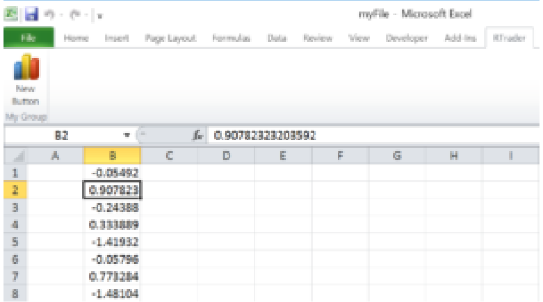


5 – Open VBA editor: In myFile.xlsm insert a new module. Paste the code below in the newly created module.

```
Sub myRCode(control As IRibbonControl)
   Dim a As Variant
   Dim theLength As Integer
   ThisWorkbook.Sheets("Sheet1").Range("B1:B10000").ClearContents
   a = Application.Run("BERT.Call", "myFunction")
   theLength = UBound(a, 1) + 1
   ThisWorkbook.Sheets("Sheet1").Range("B1:B" & theLength).Value = a
End Sub
```

This erases previous results in the worksheet prior to coping new ones.

6 – Click New Button: Now go back to the spreadsheet and in the RTrader menu click the "New Button" button. You should see something like the below appearing.



B2 0.907823233203592

	A	B	C	D	E	F	G	H	I
1		-0.05492							
2		0.907823							
3		-0.24388							
4		0.333889							
5		-1.41932							
6		-0.05796							
7		0.773284							
8		-1.48104							

You're done!

The guide above is a very basic version of what can be achieved using BERT but it shows you how to combine the power of several specific tools to build your own custom application. From my perspective the interest of such an approach is the ability to glue together R and Excel obviously but also to include via XML (and batch) pieces of code from Python, SQL and more. This is exactly what I needed. Finally I would be curious to know if anyone has any experience with BERT?

Category: Data Science, Excel, Portfolio Management, R, Trading Strategies | Comment (RSS) | Trackback

5 Comments

1. *Quantocracy's Daily Wrap for 08/19/2016 | Quantocracy* says:
   August 20, 2016 at 5:16 am

   […] Trading strategy: Making the most of the out of sample data [R Trader] When testing trading strategies a common approach is to divide the initial data set into in sample data: the part of the data designed to calibrate the model and out of sample data: the part of the data used to validate the calibration and ensure that the performance created in sample will be reflected in the real world. As a rule of thumb around 70% of the initial data can be used for calibration […]

2. *Jstan* says:
   August 20, 2016 at 5:31 pm

   Great post! Wouldn't this test applied to the underlying assets that the strategy trades be better for assessing robustness? The fact the strategy returns aren't from different processes is good, the strategy was robust to any process change in the underlying during the sample periods. How impressive this is depends on if the underlying process changed which is what the test on the underlying would tell you.

   It occurs to me this could be used to categorize regimes – if two samples (with resampling) are significantly different processes it implies that a regime shift occurred. If many assets show the same result for the period this would be compelling, especially if you saw a bifurcation in the performance of different strategy types (e.g. trend and mean reversion) coinciding with the change. Identifying predictors of these changes would be ultimate goal.

   Question: if you start with period a and b that are adjacent and each have say 2000 length. then resample from them as you did to generate your probability distribution. Then shift a and b forward with a rolling window, would the evolution of the test p distribution over time be meaningful? Seems like a trough in the distribution could indicate when you've isolated the process change point.

   - *The R Trader* says:
     August 21, 2016 at 6:52 am

     Hi,

     Thank you for reaching out.
     As you mentioned this methodology could be used in many, many ways and regime identification is one of them.
     Regarding your question, if I understand well you want to shift in and out of sample period and then resample them at every iteration. In this situation if your sample is large enough I don't see why the p-value would be unstable. It's actually just a variation of the method described in the post.
     As mentioned in the post there are other tests that could suit this framework. They all have their pros and cons and if I were you I would have a closer look at this. In some situation, it can significantly affect the result.

     Hope this helps.

     Arnaud

   - *KC* says:
     August 21, 2016 at 7:20 am

     Great post, and this actually sounds like a great suggestion. Although in the author's example he finds no difference between the two, I suspect that is less common that the opposite case. Knowing whether the difference is due to substantially different strategy behavior (raising the question of overfitting) or due to the fact that the underlying distribution has itself changed is the key to whether strategy needs more work or better left alone.

I have a question about resampling: although not mentioned in the post, am I correct in assuming that all drawn 1000-long samples are contiguous?

- *The R Trader* says:
  August 22, 2016 at 1:04 pm

  Hi Karen,

  To answer your question (if this is your question) the 1000 in-sample subsets are not contiguous. They are simply randomly selected trading days from the in-sample period. Have a look at the sample function in R for more details.

  HTH

  Arnaud