# **R in SQL, Power BI, SSRS, AML – time series - 5 of 9**

**Time series and forecasting**

 \*\*\* I hope to add further algorithms about once every couple of weeks \*\*\*

# To peer into the future

Ah, if only I could perfect this for stock market prices. In R there are many algorithms to predict how a time series will extend into the future, in SSAS just the one. It’s just one of the many algorithms available through R and DMX in the Microsoft BI stack. Here are lots of ways to do it (all files can be downloaded as a zip file - under Queries, R queries, Time Series on my Resources page):

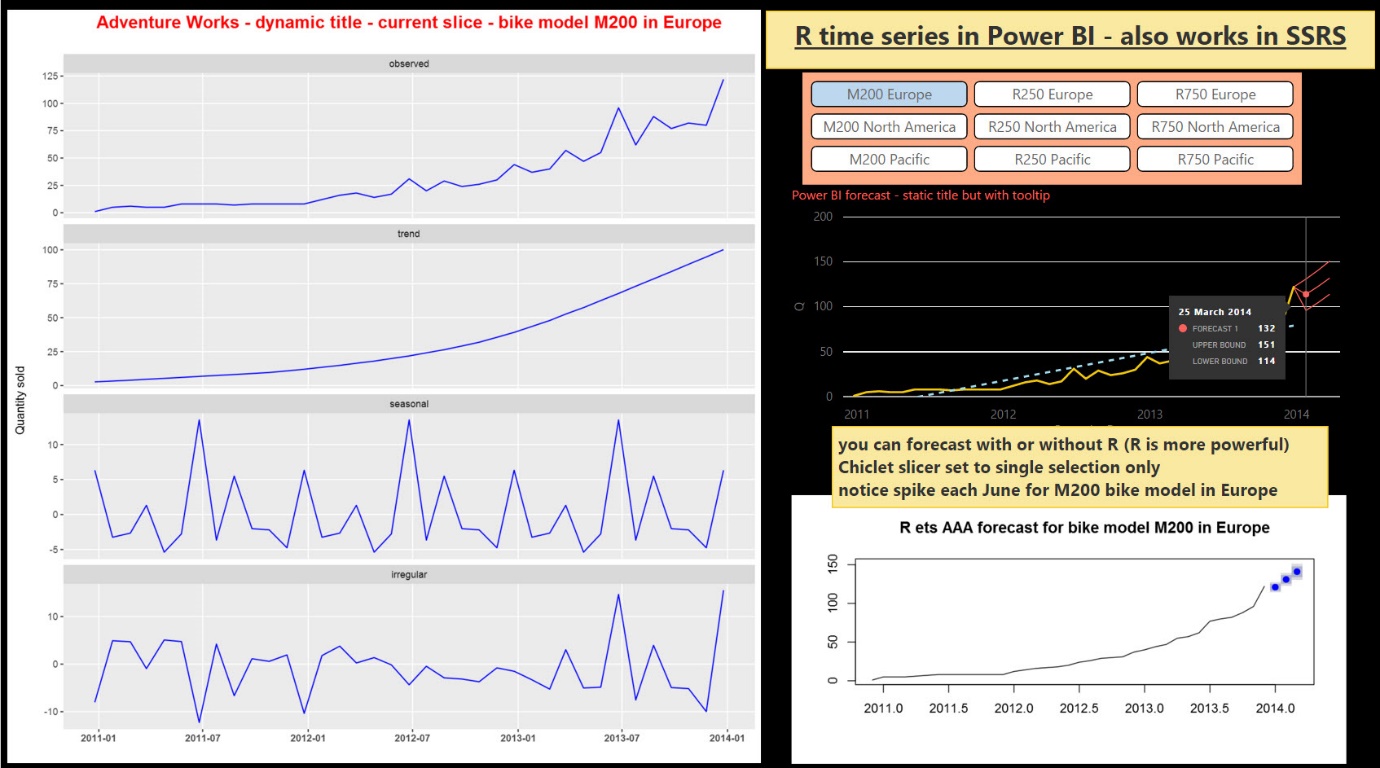
<http://www.mrcube.net/pages/Resources.aspx>

All the data is from the vTimeSeries view in Adventure Works DW relational. The zip file contains the R script, an SQL query, a DMX query, this document, screenshots, and a Power BI Desktop pbix file, so you can try all of these:

## Power BI

(TimeSeries.pbix)

This uses one table import and two R visual scripts – it also uses the built-in Power BI forecasting (make sure you have the September 2016 update at least). If you don’t have the latest Chiclet, then make sure you choose one option after opening – the recent version that I used should remember the single selection that I last used before saving the file. If you receive errors for one or both of the two R visuals, then you will need the necessary packages to support the library() functions.



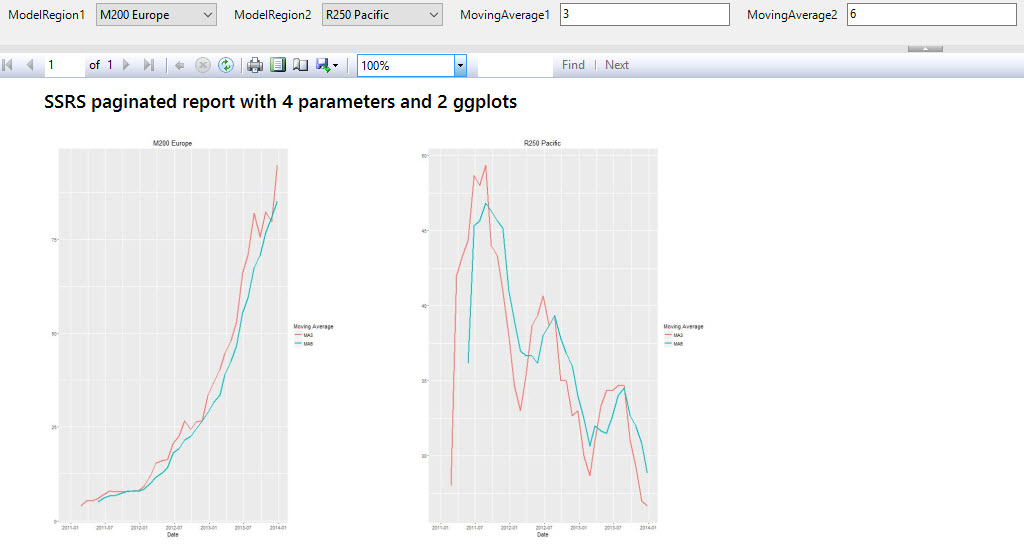
## Azure Machine Learning

Connect back to your on-prem SQL Server. Paste in and adapt the R code provided (see below). Please see my first article on the Association algorithm, if you are unsure of how to adapt the R code using input and output ports. You can also use a CSV version of the data if you can’t connect back to on-prem SQL Server – copy the query from the R code (see below) and use SSIS to create the CSV.

## SQL Server and SSRS Paginated

(SQLGraphicR.sql)

The graphic query is for SSRS Paginated, not supported yet in SSRS Mobile. You need SQL Server to do this! The graphics in Power BI are even easier and uses R directly rather than R embedded in SQL – see the pbix file above.

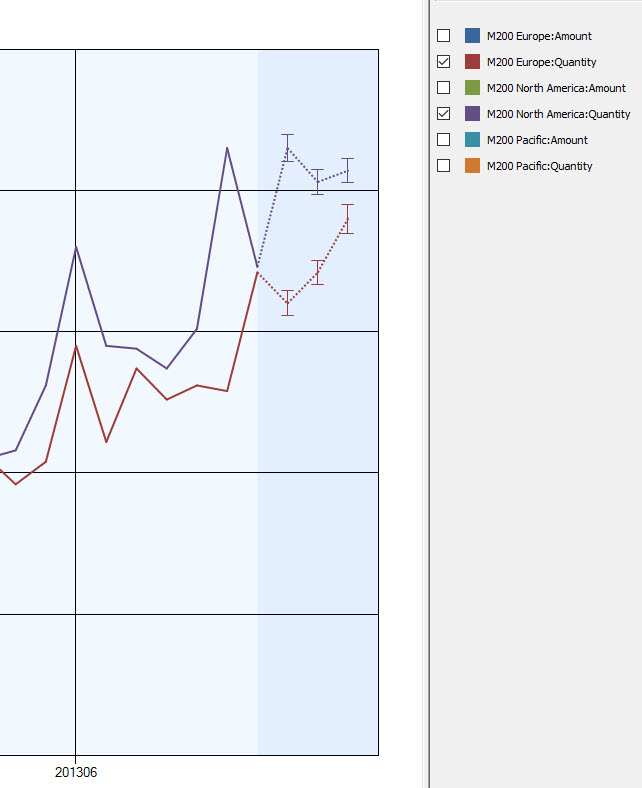


## SSAS and Excel

(SSASDataset.dmx)

You need SSAS multidimensional with the Adventure Works cube and the Excel Data Mining Add-In to show the graphic directly (I am not classing SSMS as an end-user tool). Choose the Forecasting model. The DMX dataset can be fed into Power BI, SSRS mobile, or SSRS paginated to produce a report on projected sales.

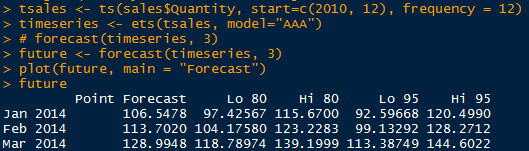




## R Studio or R Tools for Visual Studio

(TimeSeries.R)

Always prototype and debug in R Studio or R Tools for Visual Studio first. Copy and paste the R into AML, Power BI, or SQL when finished.



## Just one of the queries for now (all are in the zip file)

(SQLGraphicR.sql) – to show forecast graphics as two parameterised ggplots in SSRS paginated:

create proc [dbo].[TimeSeriesGraphic]

/\* M200 Europe is hard-coded for now \*/

/\* set up a parameter in SSRS with 'select distinct modelregion from vTimeSeries' and add parameter to this proc\*/

/\* see @params towards the end \*/

@ModelRegion nvarchar(50) = N'M200 Europe',

/\* the @X and @Y parameters for moving averages show a different way of using parameters - these should be parameterised in SSRS too \*/

/\* @X, @Y are handled outside sp\_execute\_external\_script for the R, @Modelregion is handled inside for the SQL\*/

/\* you could use @X and @Y to change the legend labels too \*/

@X int = 3,

@Y int = 6

as

declare @R nvarchar(max)

set @R = N'library(forecast);

library(ggplot2);

sales$ReportingDate <- as.Date(sales$ReportingDate);

tsales <- ts(sales$Quantity, start=c(2010, 12), frequency = 12);

ma3 <- rollmean(tsales,' + cast(@X as nchar(1)) + ', fill = NA, align = "right");

ma6 <- rollmean(tsales,' + cast(@Y as nchar(1)) + ', fill = NA, align = "right");

salescopy <- sales;

sales$maName <- "MA3";

sales$maValue <- ma3;

salescopy$maName <- "MA6";

salescopy$maValue <- ma6;

sales$maValue = as.numeric(sales$maValue);

salescopy$maValue = as.numeric(salescopy$maValue);

sales <- rbind(sales, salescopy);

image\_file = tempfile();

jpeg(filename = image\_file, width = 600, height = 800);

print(ggplot(sales, aes(x = ReportingDate, y = maValue, colour = maName)) + geom\_line(size = 1) + ggtitle(sales$ModelRegion) + xlab("Date") + ylab("") + guides(colour = guide\_legend((title = "Moving Average"))));

dev.off();

OutputDataSet <- data.frame(data=readBin(file(image\_file, "rb"), what=raw(), n=1e6));'

exec sp\_execute\_external\_script

@language = N'R'

, @script = @R

,

@input\_data\_1 = N'select TimeIndex, ReportingDate, Quantity, ModelRegion from vTimeSeries where modelregion = @ModelRegion order by TimeIndex',

@input\_data\_1\_name = N'sales',

@params = N'@ModelRegion nvarchar(50)',

@ModelRegion = @ModelRegion

with result sets((plot varbinary(max)));