INTEGRATING R AND GGPLOT2 IN POWER BI

OVERVIEW AND REVIEW OF R

R is a ‘real’ programming language but it’s sometimes used as a scripting language.

In this respect, what you can do with R is essentially what can you do using programming more generally.

It all depends on how comfortable you are with **writing algorithms**. If you can write an algorithm, you can implement that algorithm in R (or any other language).

BUT If you don’t like writing algorithms, there are many pre-packaged algorithms you can run as **a series of commands**. R has a lot of these – it’s a strength of R.

PROGRAMMING VS SCRIPTING

As the name suggests, **programming** languages (e.g. Python, Java, C) are generally intended for writing (substantial) programs.

Scripting languages (e.g. bash, R, SAS) may technically have the same capabilities, but be more geared to **running a sequence of commands** (mini pre-packaged programs)

Some scripting languages 'graduate' to becoming more of a full-fledged programming language (e.g. JavaScript).

BASE R VS R WITH PACKAGES VS POWER BI

Power BI:

- GUI, designed to make certain tasks easy to carry out ‘at the push of a button’
- if you **can** do it in Power BI, it will **probably** be easier to do it in Power BI

R + Popular Packages:

- packages lessen your need to be a full-fledged programmer – to some extent
- you still need to understand how to program

Base R:

- if you can write algorithms, you can do anything you need to in Base R, but…
- **it probably won’t be super easy at first!** (you need to understand programming quite a bit)

GOING BEYOND SCRIPTS AND PACKAGES

If you understand algorithms/programming, you can go beyond pre-packaged code.

The term ‘algorithm’ doesn’t have a super rigorous description (strangely!).

An algorithm is an abstract series of steps or instructions.

The same algorithm can be implemented in different programming languages.

A good algorithm should have clear stopping conditions.

EXAMPLE OF AN ALGORITHM

1. Pour ½ cup flour into bowl
2. Break one egg into bowl
3. Pour 3 tablespoons oil into bowl
4. Pour 1 teaspoon baking powder into bowl
5. Mix with spoon until smooth
6. Pour mixture into muffin tins
7. Bake for 15 minutes at 350 degrees Fahrenheit

GENERAL ELEMENTS OF COMPUTER CODE/PROGRAMMING

Variables

Data Structures

Operators

Statements and Expressions

Blocks (and Scope)

Functions

Logical (Control) Flow

Libraries/Packages/Modules

Inputs/Outputs

Interpreters/Compilers

load additional functions (package/module/ library) from outside the current code

variable

user-defined function with three arguments

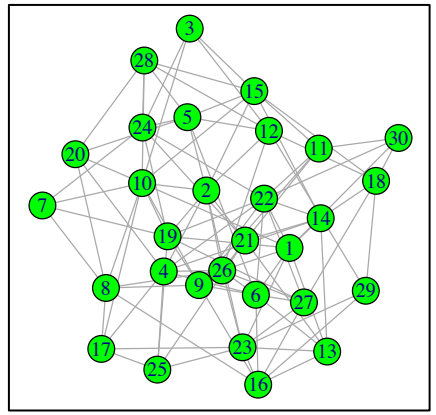
block

```
library(igraph)

my_graph_function <- function(my_number_nodes, my_colour, my_density)
{
my_graph <- sample_gnp(my_number_nodes, my_density, directed = FALSE, loops = FALSE)
if(ecount(my_graph) >= my_number_nodes){V(my_graph)$color <-  my_colour}
plot(my_graph, layout=layout.fruchterman.reingold, vertex.color=V(my_graph)$color)
}
my_graph_function(30,"green",0.3)
```

creating a data structure/ object (a graph)

conditional logic statement (control flow)



calling the user-defined function

generating output (a visualization of the graph)

EMBEDDING R IN POWER BI – 3 PLACES

You can run R scripts in 3 places in Power BI:

1. when loading the data
2. in a visualization tile
3. in Power Query

SOME IMPORTANT CAVEATS AND QUIRKS

From https://docs.microsoft.com/en-us/power-bi/desktop-r-scripts :

- **Only data frames are imported, so make sure the data you want to import to Power BI is represented in a data frame**
- Columns that are typed as Complex and Vector are not imported, and are replaced with error values in the created table
- Values that are N/A are translated to NULL values in Power BI Desktop

SOME IMPORTANT CAVEATS AND QUIRKS

From https://docs.microsoft.com/en-us/power-bi/desktop-r-scripts (cont.):

- Any R script that runs longer than 30 minutes times out
- Interactive calls in the R script, such as waiting for user input, halt the script's execution
- **When setting the working directory within the R script, you *must* define a full path to the working directory, rather than a relative path**

ANOTHER KEY QUIRK – 'MAGIC' VARIABLES

To allow R code to interface with Power BI, Power BI uses '**magic**' variables.

When you use these variables, Power BI knows to get or send data from and to certain parts of Power BI.

Magic variables include:

- '**dataset**' : Power BI creates this variable and connects it to the dataset you have loaded into Power BI. Use it in your R code when you want to interact with the loaded dataset.
- '**output**' : In Power Query, assign the result (data frame) of your R code to this variable so that Power BI can 'catch' the result and apply it to the loaded dataset.

ENABLING R IN POWER BI

To use R in Power BI, you need to point Power BI at your R installation:

- File > Options and settings and then Options > R scripting

R should offer a detected R home directory option (e.g. C:\Program Files\R\R-3.50)

OPENING R STUDIO OR A TERMINAL

When using R within Power BI, test the code directly in an R interpreter first.

This lets you debug your R code without causing issues in your dashboard, which might be tricky to resolve.

EXERCISE 1 - Do:

- open your R interpreter
- Run one or more of the code samples provided (see accompanying code snippets file)

EXERCISE 1 – OPTION 1: ENTER THIS PRACTICE CODE

```
> data() #show a list of available datasets
> mydataframe <- trees #copy the tree dataset to mydataframe
> mytreetype <- sample (c("A","B","C"),nrow(trees),replace=TRUE)
> mytreelocation <- sample(c("mountain","meadow"),nrow(trees),
   replace=TRUE)
> mytreeID <- c(1:nrow(trees))
> mydataframe$type <- mytreetype
> mydataframe$location <- mytreelocation
> mydataframe$ID <- mytreeID
```

EXERCISE 1 – OPTION 2: ENTER THIS PRACTICE CODE

```
> getwd()

# you will need to determine the correct path for your data
location.

> path <- "C:/Users/User/Desktop/PowerBI_demo_files/CityTables"
# NOTE: forward slashes

> setwd(path)

> filename <- "myfilename.csv"

> mydataframe <-  read.csv(filename)
# read in file as a dataframe

> head(mydataframe) # print the first few lines of the dataframe
```

EXERCISE 1 – OPTION 3: ENTER THIS PRACTICE CODE

```
> getwd()
> customer <- c("Jen","Steve","Deepthi","Gilles")
> purchaseamount <- c(3100, 3340, 5800,45)
> purchasetype <- c("fun","practical","fun","both")
> purchasedate <- as.Date(c('2010-11-1','2008-3-25',
   '2007-3-14','2012-01-01'))
> customer_data <- data.frame(customer, purchaseamount, purchasedate)
> mydataframe <- customer_data
```

EXERCISE 2: RUNNING AN R SCRIPT WHEN GETTING DATA

Exercise 2 - Do:

- Get Data > More..., then select Other > R script > Connect
- Paste in your chosen sample code from Exercise 1
- Run the code (click Okay)
- Click the checkbox beside the name of the dataframe(s) you would like loaded into PowerBI (may be more than one dataframe available)

EXERCISE 3: RUNNING AN R SCRIPT WHEN USING A VISUALIZATION TILE

Exercise 3 - Do:

- Select the tile with the large R icon

You will see a new tile, with an R script editor at the bottom of the workspace

- Select **several fields** (or drag fields onto the values area in the tile)

Power BI dynamically creates some R code for you to start working with.

Enter your visualizaiton code (replace blue text with your own details):

```
> plot(dataset$mycolumname)#pick a categorical variable if possible
```

Run the code by pressing the little arrow icon.

EXERCISE 3 – PART 1 : RUN THIS CODE

```
# this code assumes you have loaded the modified trees dataset
# from Exercise 1 Option 1 into the dataset variable
# if you are using a different dataset, you will need to change
variable names


> plot(dataset$Girth) #single variable simple visualization
# try with both a categorical variable and numeric variable if your
dataset has a categorical variable
```

EXERCISE 3 – PART 2 : RUN THIS CODE

```
# this code assumes you have loaded the modified trees dataset

# from Exercise 1 Option 1 into the dataset variable

# if you are using a different dataset, you will need to change
variable names

> plot(dataset$Girth, dataset$Height)
# two variable simple visualization

# try with a categorical variable and numeric variable if your
dataset has these variables
```

IMPORTANT NOTE: INSERTED UNIQUE COMMAND

Power BI always runs '**unique**' on the dataset you create in your tile R script.

If you only select the particular column in which you are interested, this may cause problems

Optional Do:

- If your datasets has a categorical variable and an ID variable, try creating a chart with just the categorical variable selected vs the categorical variable and an ID variable selected

USING SLICERS WITH R TILES

Power BI **slicers** work with R visualizations as that they work with other tiles

EXERCISE 4 - Do:

- Create a chart in the R tile that uses a categorical variable (e.g plot with two numeric variables)
- Create a second chart using a regular Power BI tile, that uses a categorial variable.
- Create a slicer that allows you to select the categories displayed in the Power BI tile
- Does the R visualization change in the same way?

RUNNING AN R SCRIPT WHEN WORKING IN POWERQUERY

Notes:

1. All R data source settings and other steps in a Power Query Editor query must be set to **Public**.
2. Output of code must be a data frame.

Exercise 5 - Do:

- Edit Query > Transform > Run R Script

EXERCISE 5 – PART 1 – CODE TO RUN

```
> dataset$season <- sample(c("summer","winter"),nrow(trees),
   replace=TRUE)

> output <- dataset
#if you don't have this, you will overwrite the data with an empty
dataset
```

EXERCISE 5 – PART 2 – CODE TO RUN

```
# This is going to replace your existing data with new data
> customer <- c("Jen","Steve","Deepthi","Gilles")
> purchaseamount <- c(3100, 3340, 5800,45)
> purchasetype <- c("fun","practical","fun","both")
> purchasedate <- as.Date(c('2010-11-1','2008-3-25','2007-3-14',
   '2012-01-01'))
> output <- data.frame(customer, purchaseamount, purchasedate)
```

RESHAPING DATA

Power BI:

- Somewhat assumes or requires data be in a database OR flat file format?

R:

- Can extensively reshape data

RESHAPING DATA EXAMPLE

```
> L_to_w <- reshape(data=customer_data,idvar="customer",
      v.names = "purchaseamount",
      timevar = "purchasetype",
      direction="wide")
```

READING IN A FILE

Power BI:

- Navigate to file using file browser
- Read in using Get Data

R:

```
> setwd("C:/Users/User/Desktop/PowerBI_demo_files/CityTables")
> df <- read.csv("myfunfile.csv",fileEncoding="latin1",
   header=TRUE, sep=",", stringsAsFactors=FALSE,
   colClasses=c("character",rep("numeric",5)))
```

STRING MANIPULATION: GSUB, GREP, STRSPLIT

Power BI:

- DAX text functions, similar to Excel

R:

- Fine grained text matching and manipulation, BUT you must have a strong grasp of regular expressions

EXAMPLE STRING MANIPULATION

```
> customer_data$purchasetype[grep('f',customer_data$purchasetype)]
   <- "FUN"
> customer_data$purchasetype <- gsub(x =
   customer_data$purchasetype, pattern = "F", replacement = "fff")
```

MERGING (JOINING) DATA FILES

Power BI:

- Has all of the join options – Merge query in Power Query
- [https://www.powerbi-pro.com/en/power-bi-seven-types-of-table-joins/]

R:

- Has all of the join options using the merge function (available in base R)
- Tidyverse also has some more user-friendly functions on this front
- [http://www.datasciencemadesimple.com/join-in-r-merge-in-r/]

CARRYING OUT OPERATIONS ON ROWS, COLUMNS, SUBSETS

Power BI:

- You can do this in M

R

- Many different ways to do this

EXAMPLE IN R: SUBSETTING, APPLYING COLUMN FUNCTION

```
> mean(customer_data$purchaseamount[customer_data$purchaseamount > 50])
```

WORKING WITH FILES

Power BI:

May have some constraints here? Expects files to be data files

R:

Can work with files as lines, strings and even characters

Using libraries, can manipulate different file types, read from databases, etc.

Example – issue with extra carriage returns – if you can think of an algorithm that deals with this, you can write it.

CONCLUDING THOUGHTS

Lots of overlap!

If you *really* can't do it in Power BI, you likely can still do it in R, somehow...

BUT it may not be *super* easy!

HOW TO INCORPORATE GGPLOT2 IN POWER BI

https://docs.microsoft.com/en-us/power-bi/visuals/service-r-visuals

https://www.red-gate.com/simple-talk/sql/bi/power-bi-introduction-working-with-r-scripts-in-power-bi-desktop-part-3/

https://medium.com/@Konstantinos_Ioannou/how-to-create-an-r-custom-visual-html-for-powerbi-7f2d2e44e453

https://www.c-sharpcorner.com/article/create-power-bi-visual-using-r-script-visual-bar-chart2/ Installing R for Power BI