Pumpkin CaRving: Basic data visualisation with R

Leeds Data Science Society

Workshop Aims

The workshop aims to introduce you to the R package ggplot2 which is a powerful data visualisation tool. You will:

- Install packages
- Read in .csv data files
- Produce a scatter plot with ggplot2
- Modify the colour of points in the scatter plot
- Modify the theme of the scatter plot and individual theme elements

Getting started with R

The first thing to do is open R or RStudio. On a Windows computer, R has a basic graphical user interface, but RStudio provides a more sophisticated one.

Click on File > New File > R Script to open a new R Script in which you can type your code.

In the first line of code, set your working directory using the command below (substitute file location with a file location of your choice – if you're using Windows, make sure to change any back slashes (\) to forwards slashes(/))

```
setwd("file/location")
```

This is where any files you produce will be stored. We then need to download, install and load the package ggplot2 using the following commands.

```
install.packages("ggplot2")
library(ggplot2)
```

You may be asked after the first command to choose a CRAN mirror. This is a location where the R package files are stored and it is best to pick one relatively nearby (for example, in the UK if you are in Leeds).

Importing the pumpkin data

We are going to import the data stored in the file pumpkin.csv. Download this file from https://github.com/leedsdatascience/Resources and move it to the working directory you specified in the previous step. We can read in a .csv file using the `read_csv()` function like this:

```
read.csv("pumpkin.csv")
```

However, this command just prints the data out and does not let you use it again. To keep the data in your R session you must assign it to an object. You do this using the assignment operator <-

```
pumpkin<-read.csv("pumpkin.csv")
```

Now there is an object in your R environment called pumpkin. You can look at the first few rows of it by typing:

```
head(pumpkin)
```

```
##      x        y label
## 1 1.75 5.178246     1
## 2 1.85 5.275590     1
## 3 1.95 5.370181     1
## 4 2.05 5.461073     1
## 5 2.15 5.547358     1
## 6 2.25 5.628174     1
```

It has three variables: x, y and label. X specifies the location of a data point along the horizontal axis of a plot, y specifies the same along the vertical axis. Label tells us which part of the pumpkin this data corresponds to.

Plotting the data

There are many different tools for plotting data in R. This workshop focuses on ggplot2 which has a unique syntax which can be spread across multiple lines. The first part of this is to open a ggplot plot using the command `ggplot()`

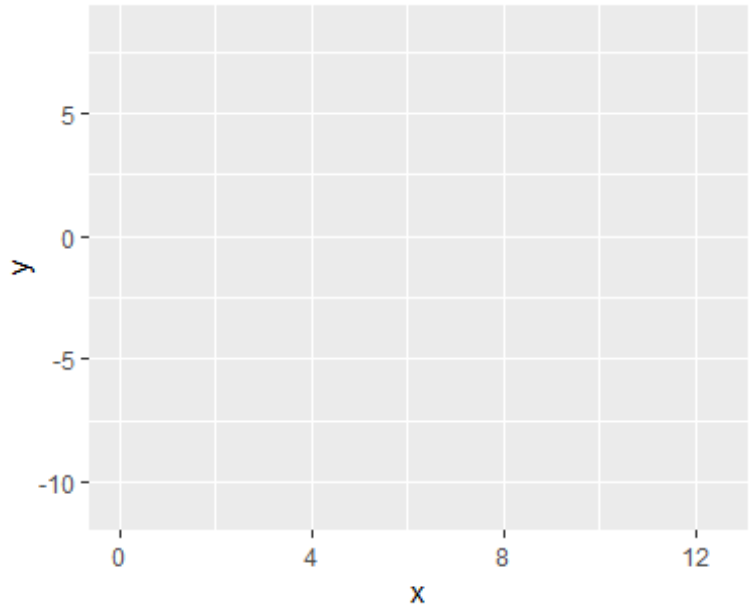
```
ggplot()
```

This brings up a blank space. We then need to tell ggplot what data we are using, so we specify the argument data=pumpkin. This tells the function to use our pumpkin data frame.

```
ggplot(data=pumpkin)
```

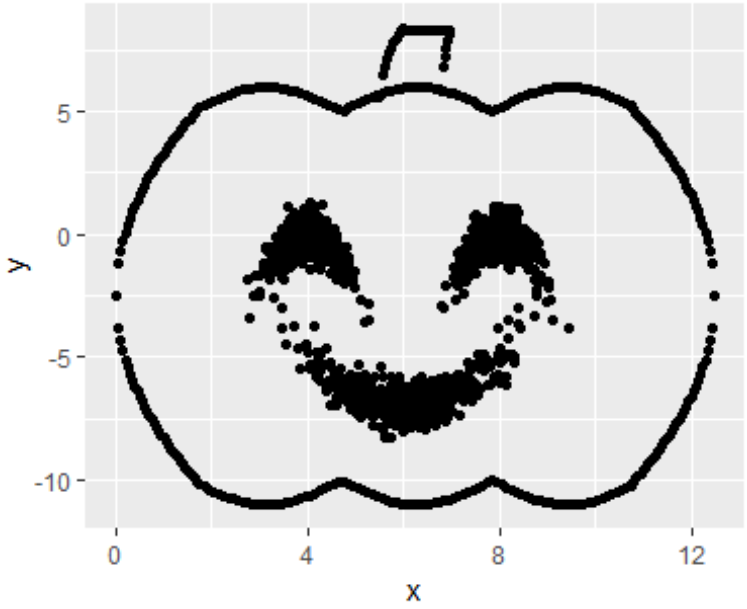

This still brings up a blank space at the moment! We also want to tell ggplot which variables to use from our data. We want our variable x to be plotted on the x axis, and y on the y axis. We specify this using the function `aes()` within our `ggplot()` command.

```
ggplot(data=pumpkin, aes(x=x, y=y))
```



We now have gridlines and axis labels, but no points on our plot. This is because the first line in the `ggplot()` command really only tells R which data to use, but not what to do with it. To tell `ggplot()` to make a scatter plot, we then add the following:

```
ggplot(data=pumpkin, aes(x=x, y=y))+
geom_point()
```



You should now see a smiling pumpkin! `geom_point()` specifically plots points, but there are other "geom" functions too. You can find out about these here: https://ggplot2.tidyverse.org/reference/.

Modifying the plot

There are many ways we could modify our pumpkin scatter plot. Today we will focus on two of these: changing the colour of points and changing the background theme.

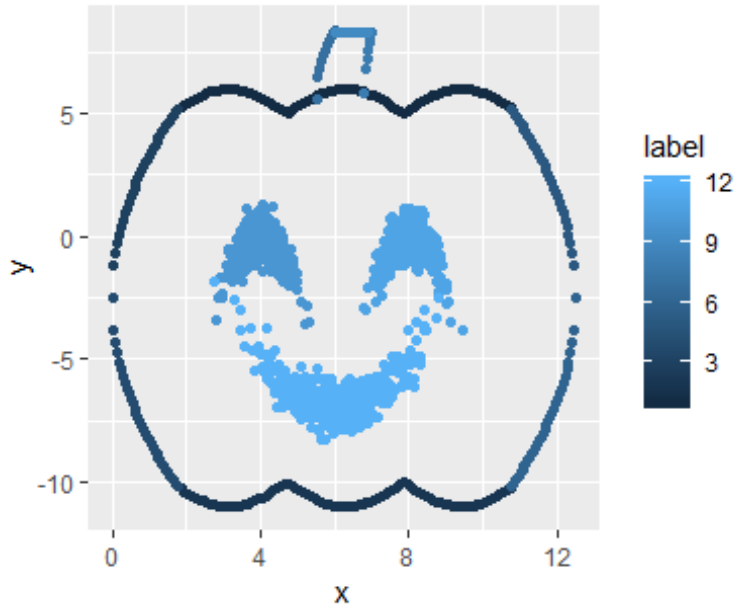
We can change the colour of all the points by specifying the colour option within `geom_point()`

```
ggplot(data=pumpkin, aes(x=x, y=y))+
geom_point(colour="red")
```

This turns our pumpkin red. There are a large range of colours to choose from in R, you can see them and their names at http://www.stat.columbia.edu/~tzheng/files/Rcolor.pdf. Try making the plot again in a different colour.

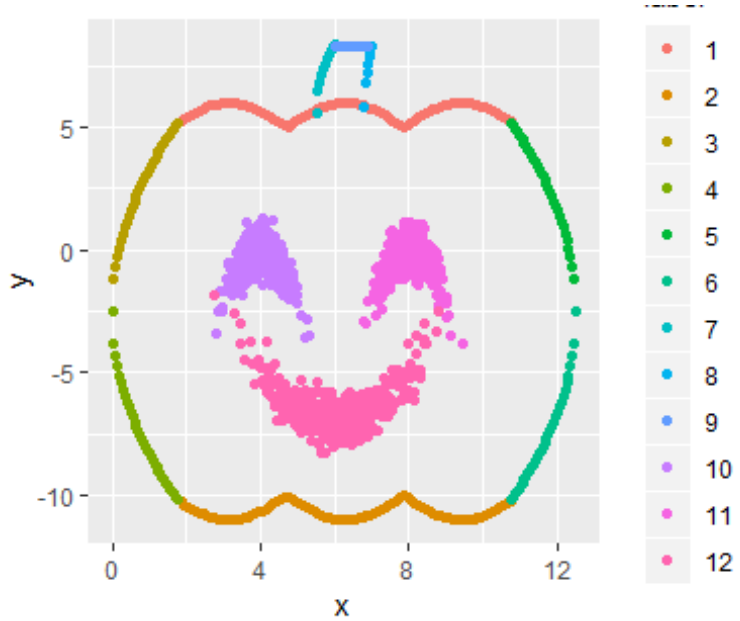
We might want the colour the be different in the different parts of the pumpkin. To do this, we might use the label variable to set the colour. If we want the colour to vary for different points, we must specify it with the `aes()` section of the `ggplot()` command like this.

```
ggplot(data=pumpkin, aes(x=x, y=y, colour=label))+
geom_point()
```



We can see that the colour is different for each label, but they are all shades of blue. This because our variable label is being treated as a continuous number. We want it to be treated as a categorical variable and for each section to be given a distinct colour. For this we must convert it to a factor and plot it again.

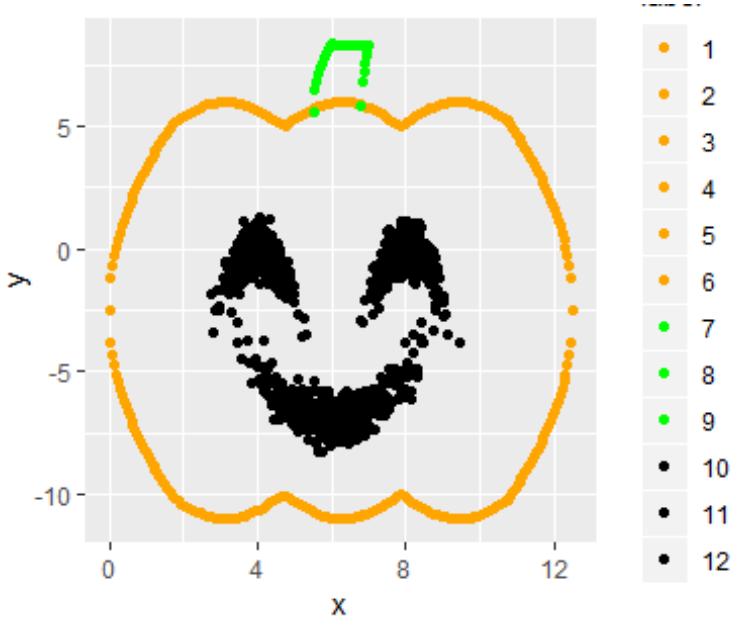
```
pumpkin$label<-as.factor(pumpkin$label)
ggplot(data=pumpkin, aes(x=x, y=y, colour=label))+
geom_point()
```



Great! Now we can see all twelve different sections of the pumpkin. But R has chosen the colours for us and they are not very spooky. We can specify the colours we want by adding the

command `scale_colour_manual()` to our plot. We specify the colours we want for sections 1-12 in order.

```
ggplot(data=pumpkin, aes(x=x, y=y, colour=label))+
  geom_point()+
  scale_colour_manual(values=c("orange","orange","orange","orange","orange","
orange","green","green","green","black","black","black"))
```



This makes a truly Halloween-y pumpkin! However, it did take a lot of time to type out and we might not want to repeat this. We won't have to do this again if we take the time to make a new variable showing which part of the pumpkin each label belongs to: the body (orange), stalk (green) or the face (black). We do this by using the function `recode()` which is in the package dplyr.

```
install.packages("dplyr") #install the package
library(dplyr) #load the package dplyr

pumpkin$part<-recode(pumpkin$label, `1`="body", `2`="body", `3`="body", `4`="
body", `5`="body", `6` ="body", `7`="stalk", `8`="stalk", `9`="stalk", `10`="
face", `11`="face", `12`="face")
```

We can then colour our plot using the part variable and a much shorter colour scale command:

```
ggplot(data=pumpkin, aes(x=x, y=y, colour=part))+
  geom_point()+
  scale_colour_manual(values=c("orange","green","black"))
```

Ggplot has helpfully included a legend so we can see which part of the plot is which colour. If we want to hide the legend, we can do this by adding the following line to our `ggplot()` command.

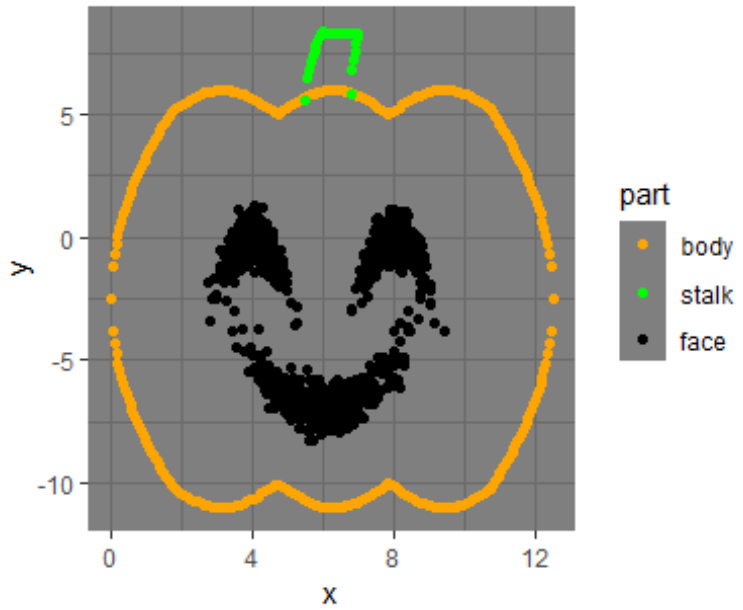
```
ggplot(data=pumpkin, aes(x=x, y=y, colour=part))+
  geom_point()+
  scale_colour_manual(values=c("orange","green","black"))+
  theme(legend.position="none")
```

Now that you can see how to change the plot colours, try setting your own for the body, stalk and face of the pumpkin. You might want a darker stalk, for example, or glowing eyes! Try

recoding the label variable in a different way to make the eyes and mouth a different colour or to make the top and bottom of the pumpkin body different colours.

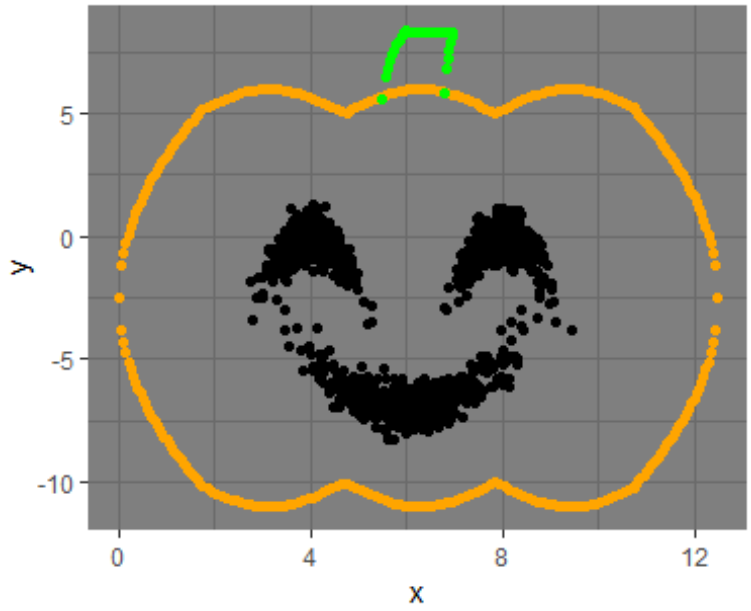
Now you have coloured your pumpkin how you want you might want to change the background of the plot to fit with our Halloween theme. You can check out the available themes in ggplot2 at this blug: https://www.datanovia.com/en/blog/ggplot-themes-gallery/#use-themes-in-ggplot2. We are going to start with the dark theme by adding a fifth line to our plot.

```
ggplot(data=pumpkin, aes(x=x, y=y, colour=part))+
  geom_point()+
  scale_colour_manual(values=c("orange","green","black"))+
  theme(legend.position="none")+
  theme_dark()
```



The background of our plot is now much darker, but our legend has appeared even though we asked it to be hidden. This is because the command `theme_dark()`, which sets the whole plot theme, overrides `theme(legend.position="none")` which just modifies a small part of the theme. To stop this we can change their order.

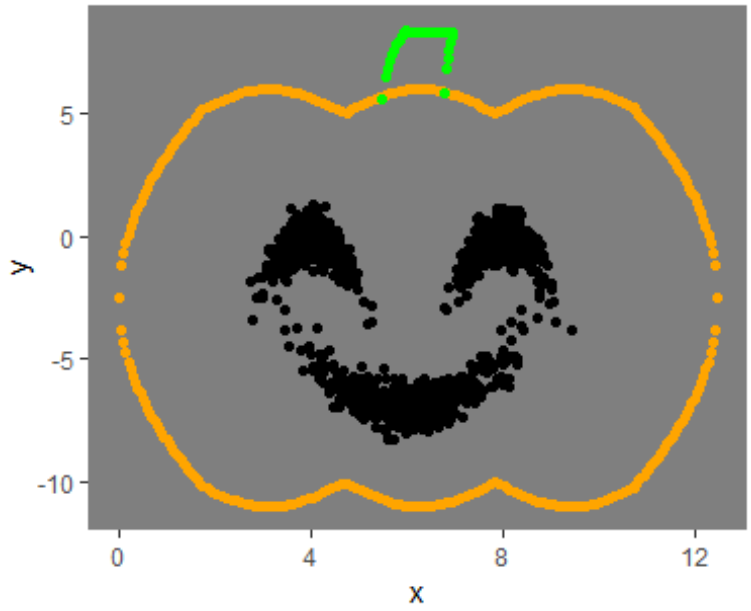
```
ggplot(data=pumpkin, aes(x=x, y=y, colour=part))+
  geom_point()+
  scale_colour_manual(values=c("orange","green","black"))+
  theme_dark()+
  theme(legend.position="none")
```



If we don't like this theme we can modify things about it in the last line of our ggplot command. Some of the elements we can modify are shown in the picture below, but you can find more here: https://ggplot2.tidyverse.org/reference/theme.html.

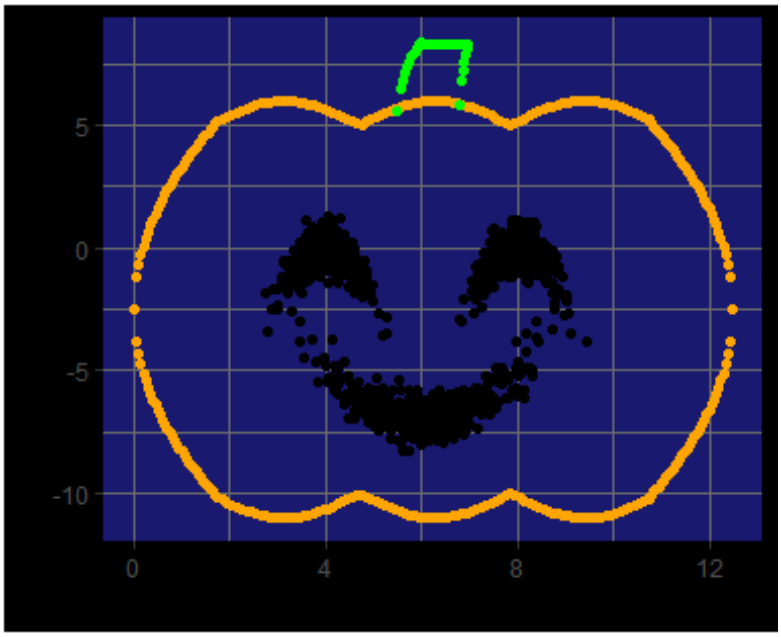
The code below draws a plot with no gridlines

```
ggplot(data=pumpkin, aes(x=x, y=y, colour=part))+
  geom_point()+
  scale_colour_manual(values=c("orange","green","black"))+
  theme_dark()+
  theme(legend.position="none", panel.grid=element_blank())
```



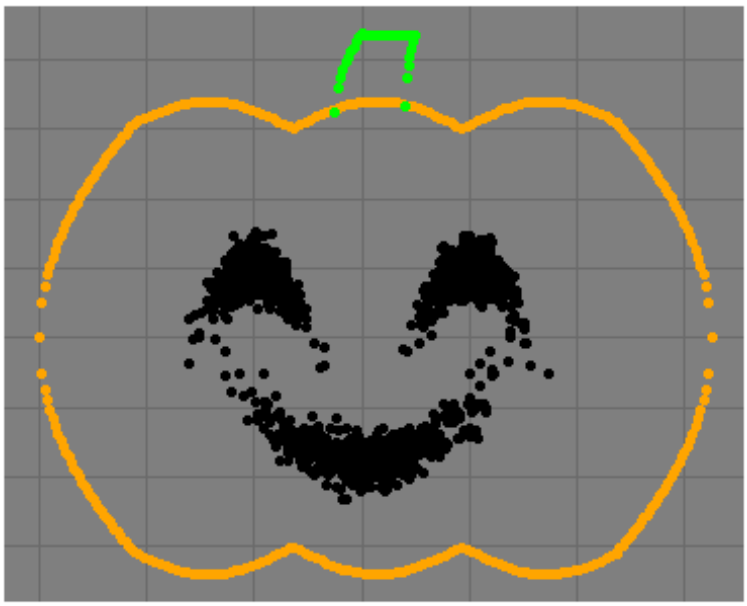
The plot window is divided into the background and panel. We can change the colour of the plot background to black and the panel to dark blue like this:

```
ggplot(data=pumpkin, aes(x=x, y=y, colour=part))+
  geom_point()+
  scale_colour_manual(values=c("orange","green","black"))+
  theme_dark()+
  theme(legend.position="none", plot.background=element_rect(fill="black"), p
anel.background = element_rect(fill="midnightblue"))
```



If we want to remove the axis ticks, text and titles we can use the following command:

```
ggplot(data=pumpkin, aes(x=x, y=y, colour=part))+
  geom_point()+
  scale_colour_manual(values=c("orange","green","black"))+
  theme_dark()+
  theme(legend.position="none", axis.ticks = element_blank(), axis.text = ele
ment_blank(), axis.title = element_blank())
```



Use these options to modify the theme of your plot however you like.

Exporting your plot

Your plot appears in R when you use the `ggplot()` command but you may want to save it. To do this you can use the `png()` command before you use `ggplot()` and then the line `dev.off()` afterwards. This tells R to plot in a png file rather than the plot window. For example:

```
png(filename="my_pumpkin.png")

ggplot(data=pumpkin, aes(x=x, y=y, colour=part))+
  geom_point()+
  scale_colour_manual(values=c("orange","green","black"))+
  theme_dark()+
  theme(legend.position="none", panel.grid=element_blank())

dev.off()

## png
##   2
```

If you want to change the size or resolution of your image you can modify the `png()` command. This produces an image of 5 by 5 inches and 300 dpi resolution.

```
png(filename="my_pumpkin.png", width=5, height=5, units="in", res=300)

ggplot(data=pumpkin, aes(x=x, y=y, colour=part))+
  geom_point()+
  scale_colour_manual(values=c("orange","green","black"))+
  theme_dark()+
  theme(legend.position="none", panel.grid=element_blank())

dev.off()

## png
##   2
```

Save your pumpkin picture and email it to leedsdatascience@gmail.com to feature in the slideshow at the end of the session.

Data visualisation resources

Here are some useful resources if you are interested in doing some more data visualisation with R.

R Graph Gallery

https://www.r-graph-gallery.com/

This is a useful site with a wide range of reproducible examples of data visualisations.

The ggplot cheat sheet

https://rstudio.com/wp-content/uploads/2016/11/ggplot2-cheatsheet-2.1.pdf

This cheat sheet will help guide you through the different graphs you can make with ggplot depending on what type of data you have

The Data Visualisation Catalogue Resource Page

https://datavizcatalogue.com/resources.html

This website has a substantial list of resources for those interested in data visualisation, from data sources and visualisation tools to blogs and podcasts.