

1.10 Exercise: Install and start using R

Using R for *Data to Insight*

This is a late breaking initiative to make analyses in **R** available in this run of *Data to Insight*. The intended audience is people who already have had some experience with installing software and some experience of coding, or at least of using some command- driven system.

R is made up of a base system and many thousands of additional packages which give it enormous capability. Almost all of our analyses will use functions from Tom Elliot's **R** package **iNZightPlots** which in turn draw on many other packages.

Warning: Coding is entirely unforgiving. If you get anything wrong, however small (e.g. missing a bracket, misspelling a name, using a lower-case letter when the name has an upper-case letter or vice versa as *R* is case sensitive), you will just get error messages. So be very careful, and even then expect to make mistakes.

1. **Install R:** Go to <https://cran.r-project.org/mirrors.html> and click on a CRAN mirror site near you. Download and install R (download versions available for Windows, Mac and Linux).

Warning: *If you are a Mac user* and may also want to use **iNZight** just use the version of R installed while installing **iNZight**. Installing a second version is likely to create problems for you. The **iNZight** installation will already have installed the packages in number **3.** below.

2. **Start up R.**

3. **Install the R packages** we will be using in the course by copying the following 2 lines of code and pasting them into the **R Console** window (not in R Studio).

```
install.packages(c('iNZightPlots', 'FutureLearnData'), dependencies = TRUE,  
repos = c('http://r.docker.stat.auckland.ac.nz/R', 'https://cran.rstudio.com'))
```

If it asks you, “**Would you like to create a personal library to install packages into?**”, say, “**Yes**”.

4. When that has completed, **paste the following 2 lines into the R Console window**

library(iNZightPlots)

library(FutureLearnData)

You will get error messages if these packages have not installed.

5. **Now try the following:** (Paste lines of code, or even several lines of code at a time, into the **R Console** window. See what they do.

# R CODE	COMMENTARY
	<i>These first 2 lines have to be run every time you start up R and want to use the functionality in iNZightPlots, or the data in FutureLearnData.</i>
library(iNZightPlots) library(FutureLearnData)	<i>Load the iNZightPlots package</i> <i>Load the FutureLearnData package (contains all the data sets for the course)</i>
data(package="FutureLearnData")	<i>Tell me about all the data sets in FutureLearnData</i>
data(nhanes_1000)	<i>Make the data set nhanes_1000 in FutureLearnData available for analysis</i>
nhanes_1000[1:10, 1:8]	<i>Show me the first 10 rows and 8 columns of nhanes_1000</i>
head(nhanes_1000) tail(nhanes_1000)	<i>Show me the top rows of nhanes_1000</i> <i>Show me the bottom rows of nhanes_1000</i>
names(nhanes_1000)	<i>Give me the names of all of the variables in nhanes_1000</i>
iNZightPlot(Race3, data=nhanes_1000)	<i>Plot the variable named Race3 in nhanes_1000</i>
iNZightPlot(Height, data=nhanes_1000) getPlotSummary(Height, data=nhanes_1000)	<i>Plot the variable named Height in nhanes_1000</i> <i>Get me a Summary of the variable named Height in nhanes_1000</i>

4. **Ask** for plots and summaries of other variables whose names you can see in the names list.

5. **When you have finished, close R.** When it asks “**Save Workspace image?**”, click, “**No**”.

To discuss issues related to this Exercise,

go to <https://gitter.im/iNZightVIT/d2i-R-discussion>

*To be able to post to the list you will have to set up a (free) account on **Github***
<https://github.com/login>

If your question relates to an Exercise, say which one you are talking about!

1.15 Exercise: Import data into R

(R version of Exercise 1.15)

From Exercise 1.10 (R version) you have already seen how to make data sets in the FutureLearnData package available for analysis but we will reiterate the general pattern soon.

The # character in R: If you type or paste a line into the R Console window, R will ignore everything that comes after a “#” character. So # tells R that what follows is a comment left for human readers, not an instruction for R itself.

We will use this in the following as we talk about the pattern for making the data in a package, in our case the FutureLearnData package available for analysis.

```
library(FutureLearnData) # Load the package FutureLearnData

data(package= "FutureLearnData") # give me info about the data in the package FutureLearnData
#                               I can copy and paste from this to get the names of data sets exactly right

data(olympics100m) # data(dataset name) makes it available for use

olympics100m # saying the name of something causes it to display
              # OK to do here as this particular dataset is small
              # Otherwise use commands from Exercise 1.10 for displaying small parts of the data set

Olympics100m # this name is wrong because of the capital "O" so will give an error

data(package= "FutureLearnData") # curly quotation marks " from Word, not straight ones, ", so error
# This whole block of lines can be copied and pasted as code. Try it
```

Reading csv and tab-separated text files into R

It is simple to read rectangular data sets in csv or tab-separated text file formats into R. We will do it now.

1. **Download** the file ***Census at School-500.csv*** from <https://www.stat.auckland.ac.nz/~wild/d2i/FutureLearn/>
2. **Download** the file ***olympics100m.txt*** from https://www.stat.auckland.ac.nz/~wild/d2i/FutureLearn_TabTxt/
3. **Now try the following:** (Paste lines of code, or even several lines of code at a time, into the R Console window. See what they do.

# R CODE	COMMENTARY
<pre># Import the file <i>Census at School-500.csv</i> cas_500 = read.csv(file.choose(), header = TRUE) cas_500[1:5, 1:9] names(cas_500) library(iNZightPlots) iNZightPlot(armspan, data= cas_500) # Now import the file <i>olympics100m.txt</i> Olymp_imp = read.table(file.choose(), header = TRUE, sep="\t") names(Olymp_imp)</pre>	<p><i>read.csv</i> is asking R to read a csv file file.choose() is telling R to throw up a browser window that will allow you to navigate to wherever you have stored <i>Census at School-500.csv</i> and open the file header = TRUE tells R that this file has a header line containing the names of the variables cas_500 = tells R to store the result as <i>cas_500</i></p> <p>Show me the first 5 rows and 9 columns of <i>cas_500</i></p> <p>Give me the names of all of the variables in <i>cas_500</i></p> <p>Need to load <i>iNZightPlots</i> package if not already done this session Plot the variable named <i>armspan</i> in <i>cas_500</i></p> <p>As above but to read the tab-separated text file we use read.table, not read.csv. We include sep="\t" to tell R to look for tab characters as the separators between data fields We store the result as store it as <i>Olymp_imp</i></p> <p>Give me the names of all of the variables in</p>

iNZightPlot(YEAR, TIME, data= Olymp_imp)	Olymp_imp Plot YEAR, TIME in Olymp_imp (gives a scatter plot of y=TIME versus x=YEAR)
?read.table	Show me the help file for the function read.table
?read.csv	Show me the help file for the function read.csv . In this case the same help file covers both of these closely related functions

[Note: Most actions in R are invoked by calling an R function. Function calls in R are of the form:

function.name(*list of function parameters separated by commas*)

When you look at help files you will note in the “Usage” paragraph that a function will often have a large number of parameters. You do not need to include any parameters in your call to a function if that parameter is set equal to a value in this paragraph. That assigned value is the **default value**. You do not need to include any parameter that has a default in your call unless you want to change its value from the default to something else.]

4. **Try some variations of the above**, e.g. plotting new variables, reading another data file.
5. **When you have finished, close R**. When it asks “**Save Workspace image?**”, click, “**No**”.

To discuss issues related to this Exercise,

go to <https://gitter.im/iNZightVIT/d2i-R-discussion>

To be able to post to the list you will have to set up a (free) account on **Github**
<https://github.com/login>

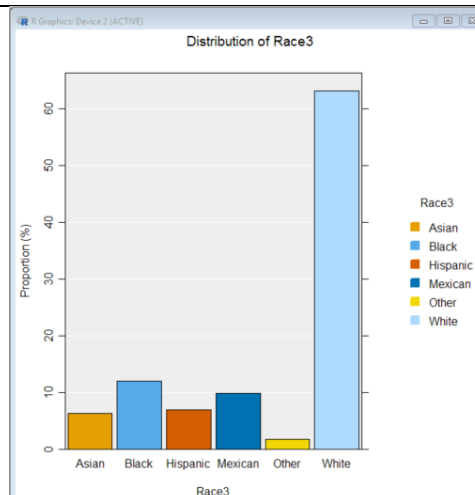
If your question relates to an Exercise, say which one you are talking about!

2.5 Exercise: Categorical variables – R version

# R code	Output																																																																
<pre># Setup library(iNZightPlots) library(FutureLearnData) data(nhanes_1000)</pre>																																																																	
<pre># Plot the variable Race3 # Because Race3 is <i>categorical</i> we get a <i>bar chart</i> iNZightPlot(Race3, data=nhanes_1000)</pre>																																																																	
<pre># Get a summary for a variable (Race3) # Equiv. of Get Summary in iNZight getPlotSummary(Race3, data=nhanes_1000)</pre>	<pre>===== iNZight Summary ----- Primary variable of interest: Race3 (categorical) Total number of observations: 1000 ===== Summary of the distribution of Race3: =====</pre> <table><tr><td></td><td>Asian</td><td>Black</td><td>Hispanic</td><td>Mexican</td><td>Other</td><td>White</td><td>Total</td></tr><tr><td>Count</td><td>63</td><td>120</td><td>70</td><td>98</td><td>17</td><td>632</td><td>1000</td></tr><tr><td>Percent</td><td>6.3%</td><td>12.0%</td><td>7.0%</td><td>9.8%</td><td>1.7%</td><td>63.2%</td><td>100%</td></tr></table> <pre>=====</pre>		Asian	Black	Hispanic	Mexican	Other	White	Total	Count	63	120	70	98	17	632	1000	Percent	6.3%	12.0%	7.0%	9.8%	1.7%	63.2%	100%																																								
	Asian	Black	Hispanic	Mexican	Other	White	Total																																																										
Count	63	120	70	98	17	632	1000																																																										
Percent	6.3%	12.0%	7.0%	9.8%	1.7%	63.2%	100%																																																										
<pre># Equivalent of Get Inference in iNZight getPlotSummary(Race3, data=nhanes_1000, summary.type="inference", inference.type="conf")</pre>	<pre>===== iNZight Inference using Normal Theory ----- Primary variable of interest: Race3 (categorical) Total number of observations: 1000 ===== Inference of the distribution of Race3: ===== Estimated Proportion with 95% Confidence Interval</pre> <table><tr><td></td><td>Lower</td><td>Estimate</td><td>Upper</td></tr><tr><td>Asian</td><td>0.04794</td><td>0.063</td><td>0.0781</td></tr><tr><td>Black</td><td>0.09986</td><td>0.120</td><td>0.1401</td></tr><tr><td>Hispanic</td><td>0.05419</td><td>0.070</td><td>0.0858</td></tr><tr><td>Mexican</td><td>0.07957</td><td>0.098</td><td>0.1164</td></tr><tr><td>Other</td><td>0.00899</td><td>0.017</td><td>0.0250</td></tr><tr><td>White</td><td>0.60211</td><td>0.632</td><td>0.6619</td></tr></table> <pre> Chi-square test for equal proportions X^2 = 1595.5, df = 5, p-value < 2.22e-16 Null Hypothesis: true proportions in each category are equal Alternative Hypothesis: true proportions in each category are not equal ### Differences in proportions of Race3 (col group - row group) Estimates</pre> <table><tr><td></td><td>Asian</td><td>Black</td><td>Hispanic</td><td>Mexican</td><td>Other</td></tr><tr><td>Black</td><td>-0.057</td><td></td><td></td><td></td><td></td></tr><tr><td>Hispanic</td><td>-0.007</td><td>0.050</td><td></td><td></td><td></td></tr><tr><td>Mexican</td><td>-0.035</td><td>0.022</td><td>-0.028</td><td></td><td></td></tr><tr><td>Other</td><td>0.046</td><td>0.103</td><td>0.053</td><td>0.081</td><td></td></tr><tr><td>White</td><td>-0.569</td><td>-0.512</td><td>-0.562</td><td>-0.534</td><td>-0.615</td></tr></table> <pre> 95% Confidence Intervals </pre>		Lower	Estimate	Upper	Asian	0.04794	0.063	0.0781	Black	0.09986	0.120	0.1401	Hispanic	0.05419	0.070	0.0858	Mexican	0.07957	0.098	0.1164	Other	0.00899	0.017	0.0250	White	0.60211	0.632	0.6619		Asian	Black	Hispanic	Mexican	Other	Black	-0.057					Hispanic	-0.007	0.050				Mexican	-0.035	0.022	-0.028			Other	0.046	0.103	0.053	0.081		White	-0.569	-0.512	-0.562	-0.534	-0.615
	Lower	Estimate	Upper																																																														
Asian	0.04794	0.063	0.0781																																																														
Black	0.09986	0.120	0.1401																																																														
Hispanic	0.05419	0.070	0.0858																																																														
Mexican	0.07957	0.098	0.1164																																																														
Other	0.00899	0.017	0.0250																																																														
White	0.60211	0.632	0.6619																																																														
	Asian	Black	Hispanic	Mexican	Other																																																												
Black	-0.057																																																																
Hispanic	-0.007	0.050																																																															
Mexican	-0.035	0.022	-0.028																																																														
Other	0.046	0.103	0.053	0.081																																																													
White	-0.569	-0.512	-0.562	-0.534	-0.615																																																												

Colour by a variable (*Race3*) (default colour palette)

```
iNZightPlot(Race3, data=nhanes_1000, colby=Race3)
```



Create a new variable *Race3.reord* to re-order *Race3*
with the categories in frequency order

```
levels(nhanes_1000$Race3)
```

```
nhanes_1000$Race3.reord =  
  factor(nhanes_1000$Race3, levels = c("White",  
    "Black", "Mexican", "Hispanic", "Asian", "Other") )
```

```
iNZightPlot(Race3.reord, data=nhanes_1000)
```

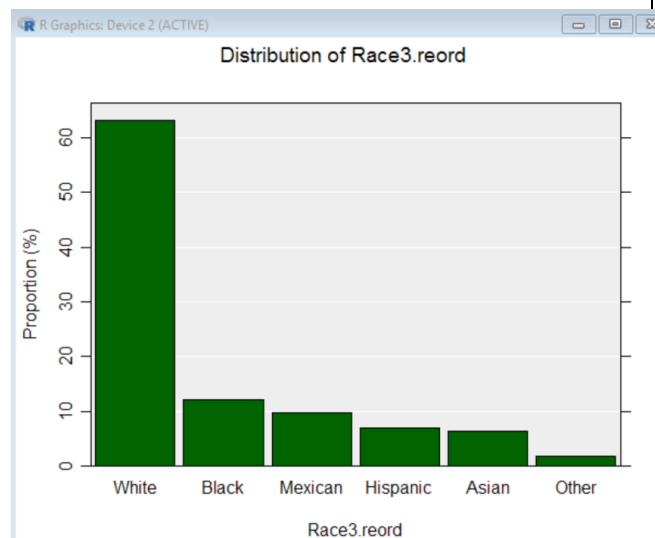
COMMENTARY

R calls a *categorical* variable a “factor”

Show me the *levels* of *Race3* (I can also see in the graph). Output is ...

```
[1] "Asian" "Black" "Hispanic" "Mexican" "Other"  
"White"
```

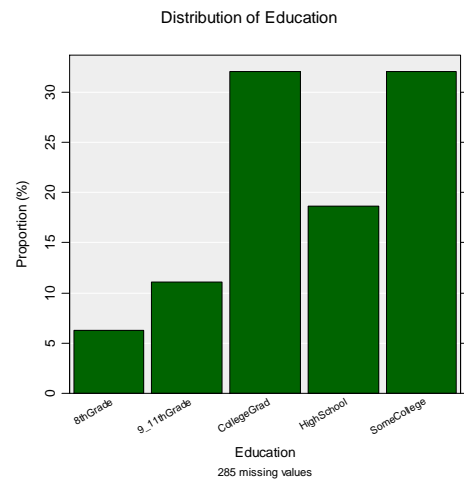
I can see what the frequency order should be from the graph. (This can be done generally with code but the code is too complex to do at this stage) So I'll make *Race3.reord* from *Race3* and put them in the order I want. (Getting the number of levels and spelling exactly right is crucial)



We'll do this again *putting the levels of Education into a sensible order*

```
iNZightPlot(Education, data=nhanes_1000)
```

```
levels(nhanes_1000$Education)
```



```
[1] "8thGrade" "9_11thGrade" "CollegeGrad"
     "HighSchool" "SomeCollege"
```

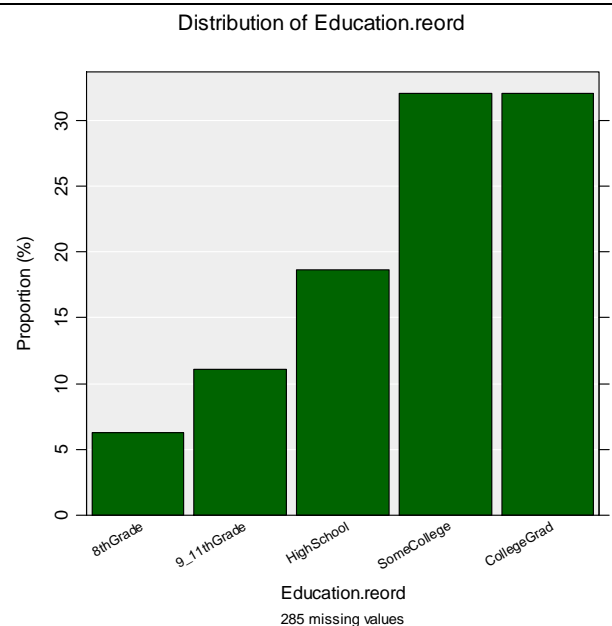
Create a new variable to re-order Education

```
nhanes_1000$Education.reord =  
  factor(nhanes_1000$Education, levels =  
    c("8thGrade", "9_11thGrade", "HighSchool", "SomeCollege", "CollegeGrad"))
```

```
levels(nhanes_1000$Education.reord)
```

```
[1] "8thGrade" "9_11thGrade" "HighSchool"  
     "SomeCollege" "CollegeGrad"
```

```
iNZightPlot(Education.reord, data=nhanes_1000)
```



```
iNZightPlot(Education.reord,  
data=nhanes_1000,colby=Education.reord)
```

Now change the colour palette to rainbow colours

```
iNZightPlot(Education.reord,  
data=nhanes_1000,colby=Education.reord,  
col.fun=rainbow)
```

COMMENTARY

Colour by Education.reord

*Col.fun has to be a colour palette function
There are lots of colour palette functions in R,
many you have to install other packages to get.
rainbow() is a generally available colour palette*

```
library(colorspace)
iNZightPlot(Education.reord,
  data=nhanes_1000,colby=Education.reord,
  col.fun=rainbow_hcl)
```

Using the **rainbow_hcl** colour function from the **colorspace** package

- Try repeating the above using other choices for variables and settings

If you want to try installing some other R packages, in the R menus

Go **Packages > Install packages** . You will probably be asked to choose a CRAN mirror site.

Then you will be shown a list of packages to choose from.

Installing the package **viridis** and then loading it [via *library(viridis)*] will give you access to the colour functions: **viridis**, **magma**, and **inferno**

To discuss issues related to this Exercise,

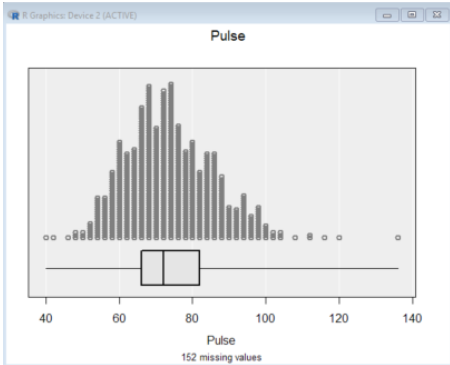
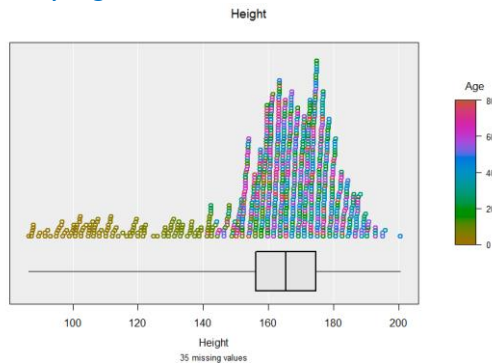
go to <https://gitter.im/iNZightVIT/d2i-R-discussion>

To be able to post to the list you will have to set up a (free) account on **Github**

<https://github.com/login>

If your question relates to an Exercise, say which one you are talking about!

2.10 Exercise: Numeric variables – R version

#R Code	Commentary																
<pre># Plot a numeric variable (Pulse) iNZightPlot(Pulse, data=nhanes_1000) # Changing the size of the dots iNZightPlot(Pulse, data=nhanes_1000, cex.dotpt=.4) iNZightPlot(Pulse, data=nhanes_1000, cex.dotpt=2)</pre>	<p>Because <i>Pulse</i> is numeric we get a dot plot</p> 																
<pre># Get a Summary for Pulse getPlotSummary(Pulse, data=nhanes_1000) # Equivalent of Get Inference for Pulse getPlotSummary(Pulse, data=nhanes_1000, summary.type="inference", inference.type="conf")</pre>	<div>iNZight Summary</div> <div>-----</div> <div>Primary variable of interest: Pulse (numeric)</div> <div>Total number of observations: 1000 Number omitted due to missingness: 152 Total number of observations used: 848</div> <div>-----</div> <div>Summary of Pulse:</div> <div>-----</div> <table><thead><tr><th>Min</th><th>25%</th><th>Median</th><th>75%</th><th>Max</th><th>Mean</th><th>SD</th><th>Sample Size</th></tr></thead><tbody><tr><td>40</td><td>66</td><td>72</td><td>82</td><td>136</td><td>73.73</td><td>12.03</td><td>848</td></tr></tbody></table> <div>-----</div>	Min	25%	Median	75%	Max	Mean	SD	Sample Size	40	66	72	82	136	73.73	12.03	848
Min	25%	Median	75%	Max	Mean	SD	Sample Size										
40	66	72	82	136	73.73	12.03	848										
<pre>iNZightPlot(Height, data=nhanes_1000) # Colour points by Age iNZightPlot(Height, data=nhanes_1000, colby=Age) # Change colour palette to rainbow iNZightPlot(Height, data=nhanes_1000, colby=Age, col.fun=rainbow)</pre>	<p>Coloured by Age</p> 																

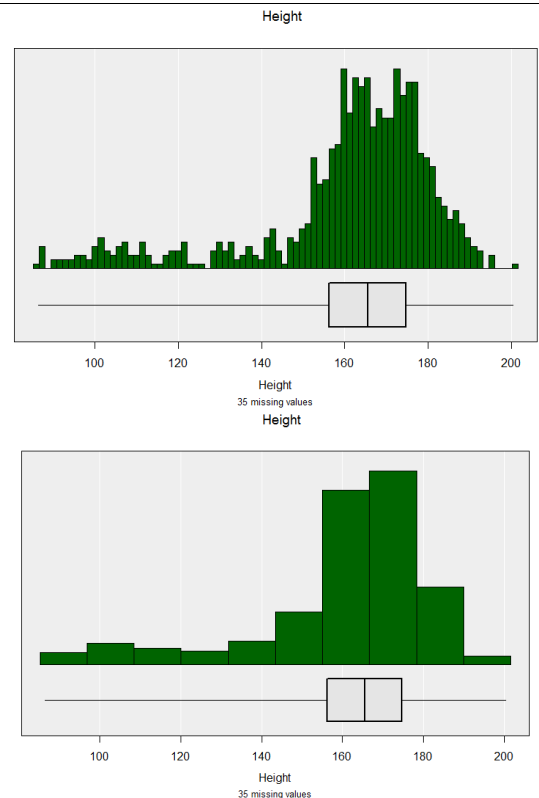
```
iNZightPlot(Height, data=nhanes_1000)
```

```
# Change Plot type to histogram
```

```
iNZightPlot(Height, data=nhanes_1000, plottype="hist")
```

```
# Control the number of bins (class intervals)
```

```
iNZightPlot(Height, data=nhanes_1000, plottype="hist", hist.bins=10)
```



```
# Get a list of all the other things that can be changed in plots
```

```
inzpar() # This list is complete
```

```
?inzpar # This documentation is not complete
```

- Try doing more things like the above but using other variables and settings

To discuss issues related to this Exercise,

go to <https://gitter.im/iNZightVIT/d2i-R-discussion>

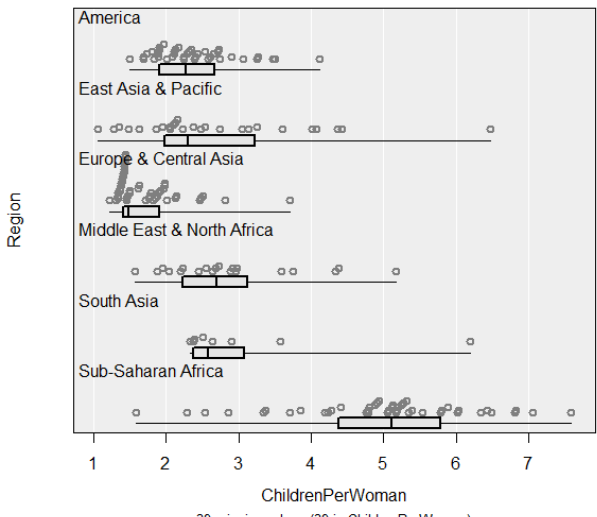
To be able to post to the list you will have to set up a (free) account on **Github**

<https://github.com/login>

If your question relates to an Exercise, say which one you are talking about!

2.13 Exercise: Comparing groups – R version

We are going to use the **gapminder_2008** data in the **FutureLearnData** package and look at the values of **ChildrenPerWoman** for each of the countries broken out by **Region**.

# R CODE	COMMENTARY or OUTPUT
<pre> # Setup library(iNZightPlots) library(FutureLearnData) data(gapminder_2008) names(gapminder_2008) iNZightPlot(ChildrenPerWoman, data=gapminder_2008) </pre>	<p><i>Commentary</i></p> <p>Make gapminder_2008 inside FutureLearnData available for analysis</p> <p>Useful for checking on the spellings of variable names</p> <p>Dot plot for ChildrenPerWoman</p>
<pre> # Now break out by Region iNZightPlot(ChildrenPerWoman, Region, data=gapminder_2008) </pre>	<p>ChildrenPerWoman by Region</p>  <p>39 missing values (39 in ChildrenPerWoman)</p>

Get Summary of ChildrenPerWoman broken out by Region

**getPlotSummary(ChildrenPerWoman, Region,
data=gapminder_2008)**

```
-----
Primary variable of interest: ChildrenPerWoman (numeric)
Secondary variable: Region (categorical)

Total number of observations: 225
Number omitted due to missingness: 39 (39 in ChildrenPerWoman)
Total number of observations used: 186
-----

Summary of ChildrenPerWoman by Region:
-----
```

	Min	25%	Median	75%	Max	Mean	SD	Sample Size
America	1.498	1.908	2.264	2.668	4.119	2.377	0.5779	40
East Asia & Pacific	1.050	1.979	2.295	3.226	6.479	2.685	1.2292	26
Europe & Central Asia	1.218	1.410	1.482	1.898	3.703	1.713	0.4700	48
Middle East & North Africa	1.570	2.223	2.703	3.125	5.163	2.893	0.9310	20
South Asia	2.323	2.381	2.574	3.069	6.196	3.113	1.3125	8
Sub-Saharan Africa	1.579	4.377	5.117	5.785	7.588	4.990	1.2674	44

```
-----
```

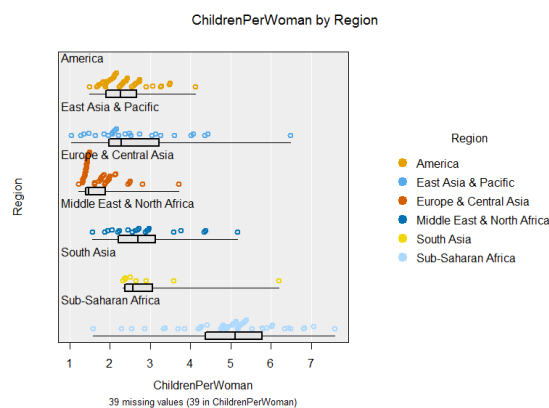
Colour by Region

**iNZightPlot(ChildrenPerWoman, Region,
data=gapminder_2008, colby=Region)**

Try also

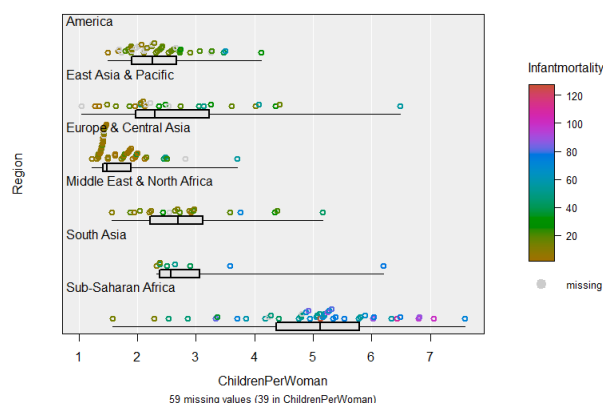
**iNZightPlot(ChildrenPerWoman, Region, data=gapminder_2008,
colby=Region, cex.text=.3)**

**iNZightPlot(ChildrenPerWoman, Region, data=gapminder_2008,
colby=Region, hide.legend = TRUE)**



Colour by Infantmortality

**iNZightPlot(ChildrenPerWoman, Region,
data=gapminder_2008, colby=Infantmortality)**



- What do you see in the last graph?
- Also try colouring by other variables you think might help explain the Regional differences.

To discuss issues related to this Exercise,

go to <https://gitter.im/iNZightVIT/d2i-R-discussion>

To be able to post to the list you will have to set up a (free) account on **Github**

<https://github.com/login>


If your question relates to an Exercise, say which one you are talking about!

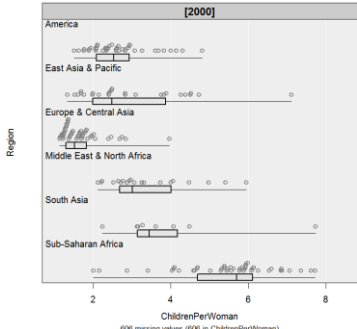
2.15 Exercise: Time travel – R version

In this Exercise we are going to use the **gapminder** data set that contains data for all years whereas **gapminder_2008** (that we used in the last Exercise) only had the data for 2008.

We will ...

1. Start by looking at the plot of *ChildrenPerWoman* by *Region*
 - This is not sensible as we are getting lots of points for the same country sometimes going as far back as 1952
2. Make a separate graph like this for every year (as a tiled set of plots)
3. See how to look at just one of these plots
4. See how to “play through the years” by displaying the graph for each year in turn

#R Code	Commentary OR Output
<pre># Setup library(iNZightPlots) library(FutureLearnData) data(gapminder) names(gapminder) iNZightPlot(ChildrenPerWoman, Region, data=gapminder)</pre>	<p><i>Commentary</i></p> <p>Use gapminder NOT gapminder_2008</p> <p>Plot ChildrenPerWoman by Region</p>
<pre># Plot ChildrenPerWoman by Region subset by Year_cat iNZightPlot(ChildrenPerWoman, Region, g1=Year_cat, data=gapminder)</pre>	

<pre> levels(gapminder\$Year_cat) # Display just the plot for the year 2000 iNZightPlot(ChildrenPerWoman, Region, g1=Year_cat, g1.level="[2000]", data=gapminder) # Do it again for 2004 iNZightPlot(ChildrenPerWoman, Region, g1=Year_cat, g1.level="[2004]", data=gapminder) </pre>	<p><i>Remind ourselves how the levels of Year_cat are represented</i></p> <pre>[1] "[1952]" "[1956]" "[1960]" ...</pre> <p><i>Choose the level value corresponding to the year 2000</i></p> 
<pre> # Now put it in a loop and do it for every year, i.e. for every level of Year_cat for (k in levels(gapminder\$Year_cat)) iNZightPlot(ChildrenPerWoman,Region, g1=Year_cat, g1.level=k, data=gapminder) </pre>	
<pre> # Do not display a new plot UNTIL you have clicked on the on plot window old.value = devAskNewPage(TRUE) # save current plotting behaviour and ask for new behaviour for (k in levels(gapminder\$Year_cat)) iNZightPlot(ChildrenPerWoman,Region, g1=Year_cat, g1.level=k, data=gapminder) devAskNewPage(old.value) # Reset the plotting behaviour back to the way it was before </pre>	
<pre> # Play the plots, but with a 2 second delay between plots for (k in levels(gapminder\$Year_cat)) { iNZightPlot(ChildrenPerWoman,Region, g1=Year_cat, g1.level=k, data=gapminder) Sys.sleep(2) } </pre>	<p>Commentary</p> <p><i>This time there are 2 lines of code to be run at each step so we have to put them in "{ .. }" brackets so that both lines get run</i></p>

To discuss issues related to this Exercise,

go to <https://gitter.im/iNZightVIT/d2i-R-discussion>

To be able to post to the list you will have to set up a (free) account on **Github**

<https://github.com/login>

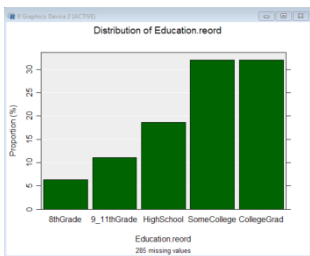
If your question relates to an Exercise, say which one you are talking about!

3.5 Exercise: Relationships between categorical variables – R version

This exercise will enable you to construct graphs of two categorical variables as discussed in the previous video. The skills addressed are:

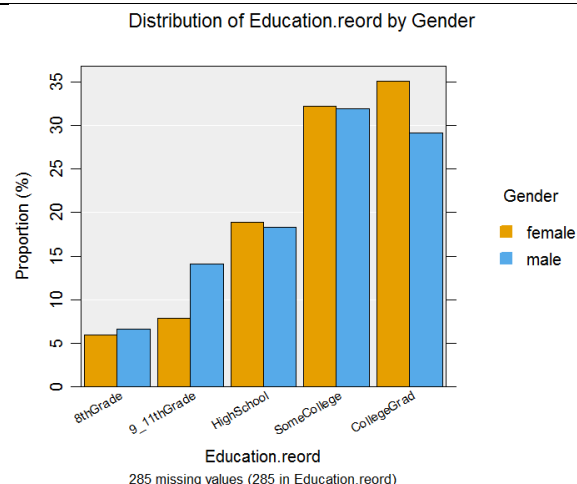
1. Creating plots of two categorical variables (when the predictor variable has only 2 groups).
2. Making a **side-by-side** bar charts and **separate** bar charts for two categorical variables.
3. Filtering out unwanted groups

We will use the **nhanes_1000** dataset from the **FutureLearnData** package.

#R Code	Output and/or Commentary												
<pre># Setup library(iNZightPlots) library(FutureLearnData) data(nhanes_1000)</pre>													
<pre># Recreate Education.reord nhanes_1000\$Education.reord = factor(nhanes_1000\$Education, levels = c("8thGrade", "9_11thGrade", "HighSchool", "SomeCollege", "CollegeGrad")) iNZightPlot(Education.reord, data=nhanes_1000)</pre>	<p><i>Education with levels in a sensible order</i></p> <p><i>(This was done in Exercise 2.5)</i></p>  <table border="1"> <caption>Data for Education.reord Distribution</caption> <thead> <tr> <th>Education Level</th> <th>Proportion (%)</th> </tr> </thead> <tbody> <tr> <td>8thGrade</td> <td>5</td> </tr> <tr> <td>9_11thGrade</td> <td>10</td> </tr> <tr> <td>HighSchool</td> <td>18</td> </tr> <tr> <td>SomeCollege</td> <td>25</td> </tr> <tr> <td>CollegeGrad</td> <td>25</td> </tr> </tbody> </table>	Education Level	Proportion (%)	8thGrade	5	9_11thGrade	10	HighSchool	18	SomeCollege	25	CollegeGrad	25
Education Level	Proportion (%)												
8thGrade	5												
9_11thGrade	10												
HighSchool	18												
SomeCollege	25												
CollegeGrad	25												

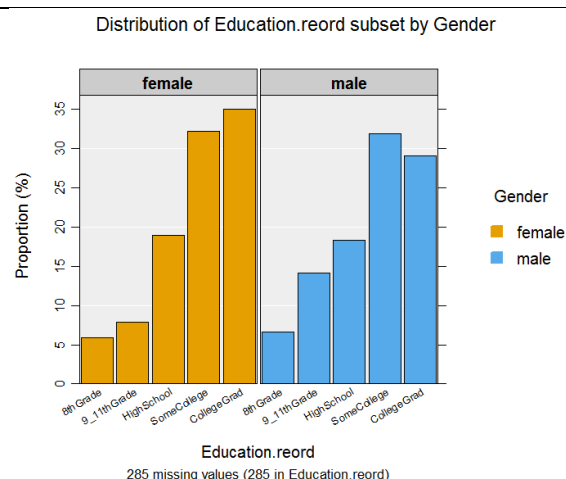
Education.reord by Gender (side by side barcharts)

```
iNZightPlot(Education.reord, Gender, data=nhanes_1000)
```



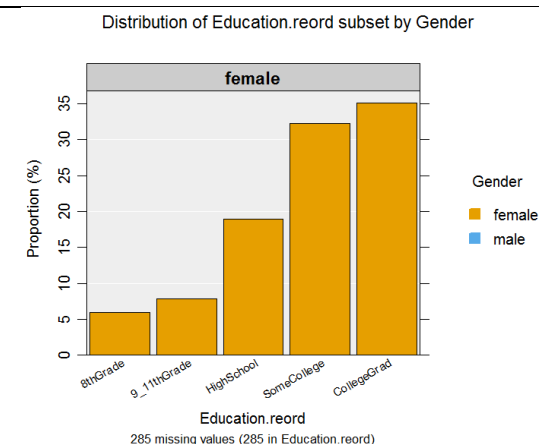
Education subset by Gender (also coloured by Gender)

```
iNZightPlot(Education.reord, g1=Gender, data=nhanes_1000, colby=Gender)
```



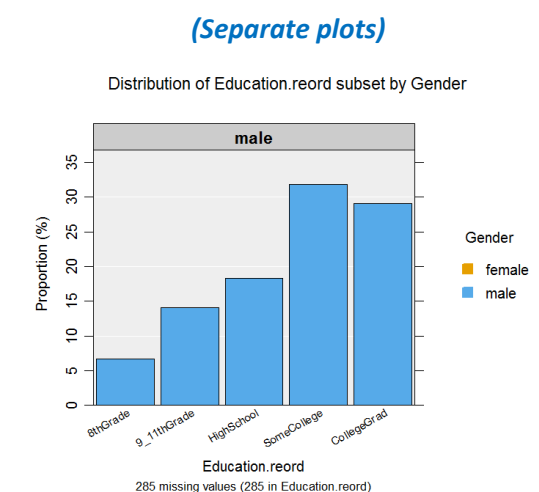
Display just the female plot

```
iNZightPlot(Education.reord, g1=Gender, g1.level="female", data=nhanes_1000, colby=Gender)
```



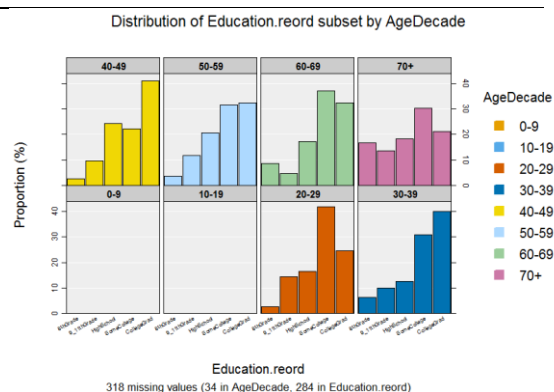
Display just the male plot

```
iNZightPlot(Education.reord, g1=Gender, g1.level="male", data=nhanes_1000, colby=Gender)
```



Plot Education.reord subset by AgeDecade

```
iNZightPlot(Education.reord, g1=AgeDecade, data=nhanes_1000,
colby=AgeDecade)
```



Filter out unhelpful data (Under 20s)

Make a subset of the data without “under 20s” & call it Temp

(This is an *example* of “filtering” the data)

```
Temp = subset(nhanes_1000, AgeDecade!=" 0-9" & AgeDecade
!=" 10-19") # “!=” is read as “is not equal to”
```

```
table(Temp$AgeDecade)
```

This will remove the empty levels

```
Temp$AgeDecade=factor(Temp$AgeDecade)
```

Replot the data using the subset of the data called Temp

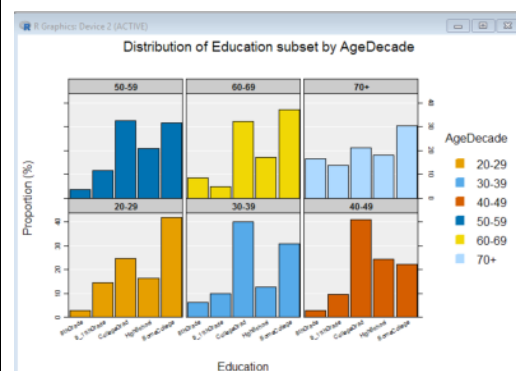
```
iNZightPlot(Education.reord, g1=AgeDecade, data=Temp,
colby=AgeDecade)
```

(Warning: there is a leading space on all of AgeDecade’s level names– a data bug)

Output

```
0-9 10-19 20-29 30-39 40-49 50-59 60-69 70+
0      0    146   110   144   111    66
```

It still has the empty levels



- Now construct similar graphics using other categorical variables whose behaviour you may be interested in

To discuss issues related to this Exercise,

go to <https://gitter.im/iNZightVIT/d2i-R-discussion>

To be able to post to the list you will have to set up a (free) account on **Github**

<https://github.com/login>

If your question relates to an Exercise, say which one you are talking about!

3.9 Exercise: Relationships between numeric variables – R version

This exercise will enable you to construct graphs of two numeric variables as discussed in the previous video. The skills addressed are:

1. Creating a scatterplot of two numeric variables (when the predictor variable has more than groups).
2. Adding a trend line (if suitable) and label any interesting points.
3. Train your eyes to observe and draw an envelope around the scatter.

We will use the **gapminder_2008** dataset in the **FutureLearnData** package.

Creating a scatterplot of two numeric variables

When we call **iNZightPlot(x,y)** using two numeric variables *x* and *y* we will get a scatterplot. The 1st variable (*x*) gets plotted against the horizontal axis and the 2nd variable (*y*) is plotted against the vertical axis. So we should put the ***predictor variable first*** and the ***outcome variable second***.

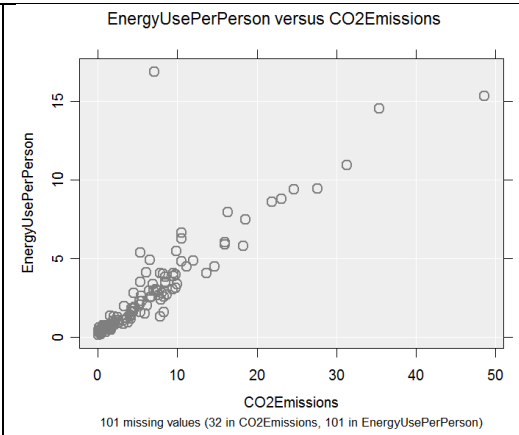
[**Note:** I've just seen that have reversed the order of the variables in this exercise from the way it was in the (older) iNZight version. The order below is more natural. The change affects nothing that matters in the exercise but the graphs are “the other way around” from the iNZight version.]

#R Code	Commentary
<pre># Setup library(iNZightPlots) library(FutureLearnData) data(gapminder_2008)</pre>	

Plot $y=EnergyUsePerPerson$ by $x=CO2emissions$

```
iNZightPlot(CO2Emissions, EnergyUsePerPerson,
data=gapminder_2008)
```

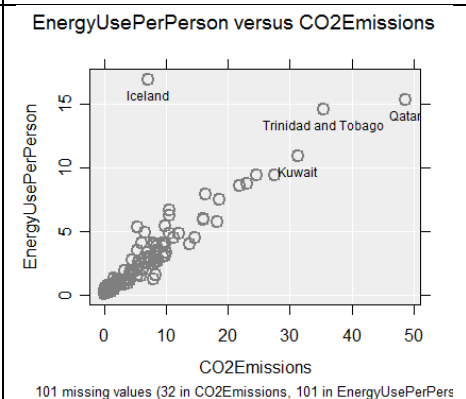
Note the reversal in the order of the names between the call and ordinary English usage for what we want to do!



Recall: the plot call is in the order $iNZightPlot(x,y,...)$

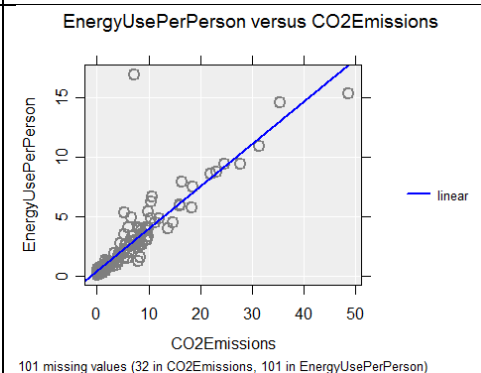
Label the 4 most extreme points by the variable Country

```
iNZightPlot(CO2Emissions, EnergyUsePerPerson,
data=gapminder_2008, locate.extreme=4, locate=Country)
```



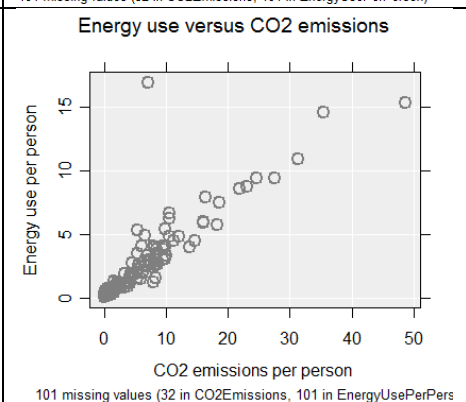
Add a trend line

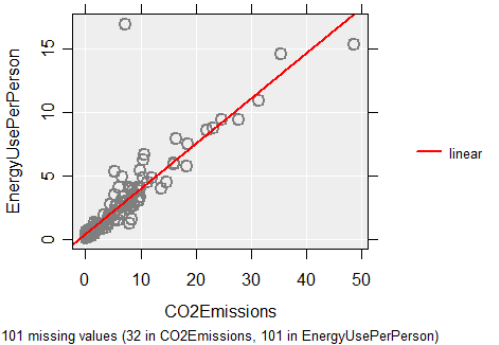
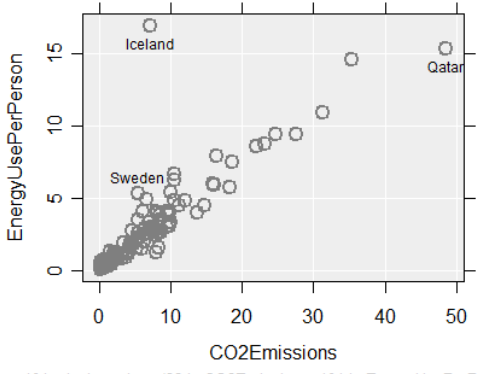
```
iNZightPlot(CO2Emissions, EnergyUsePerPerson,
data=gapminder_2008, trend="linear")
```



Changing from the default Axes labels and title

```
iNZightPlot(CO2Emissions, EnergyUsePerPerson,
data=gapminder_2008, xlab="CO2 emissions per person ",
ylab="Energy use per person", main="Energy use versus CO2
emissions")
```



<pre># Put on a trend line but coloured red, instead of the default (blue) iNZightPlot(CO2Emissions, EnergyUsePerPerson, data=gapminder_2008, trend="linear", col.trend=list(linear="red"))</pre>	<p>EnergyUsePerPerson versus CO2Emissions</p>  <p>101 missing values (32 in CO2Emissions, 101 in EnergyUsePerPerson)</p>
<pre># Label specified data points ids = (1:nrow(gapminder_2008))[gapminder_2008\$Country %in% c("Qatar","Iceland","Sweden")] iNZightPlot(CO2Emissions, EnergyUsePerPerson, data=gapminder_2008, locate.id=ids, locate=Country)</pre>	<p>Commentary</p> <p><i>Tell me the row numbers of all the rows of gapminder_2008 for which the value of Country is in the set listed in c(...).</i></p> <p><i>Store these row numbers in ids</i></p> <p><i>“locate.id” is used to tell iNZightPlot the row numbers of the points it should label</i></p> <p><i>“locate=Country” tells iNZightPlot to use the contents of the variable Country to label those points</i></p> <p>EnergyUsePerPerson versus CO2Emissions</p>  <p>101 missing values (32 in CO2Emissions, 101 in EnergyUsePerPerson)</p>

The options added to the plots above can all be used together. Try putting several of them together in the same call to iNZightPlot. There are lots of others you will find out about over time, e.g. if you are putting a line on your plot then adding “, lwd=2” to your call (without the “”) will double the thickness of the line.

Try to identify more or other countries (*spelling and lower/upper case is critical*).

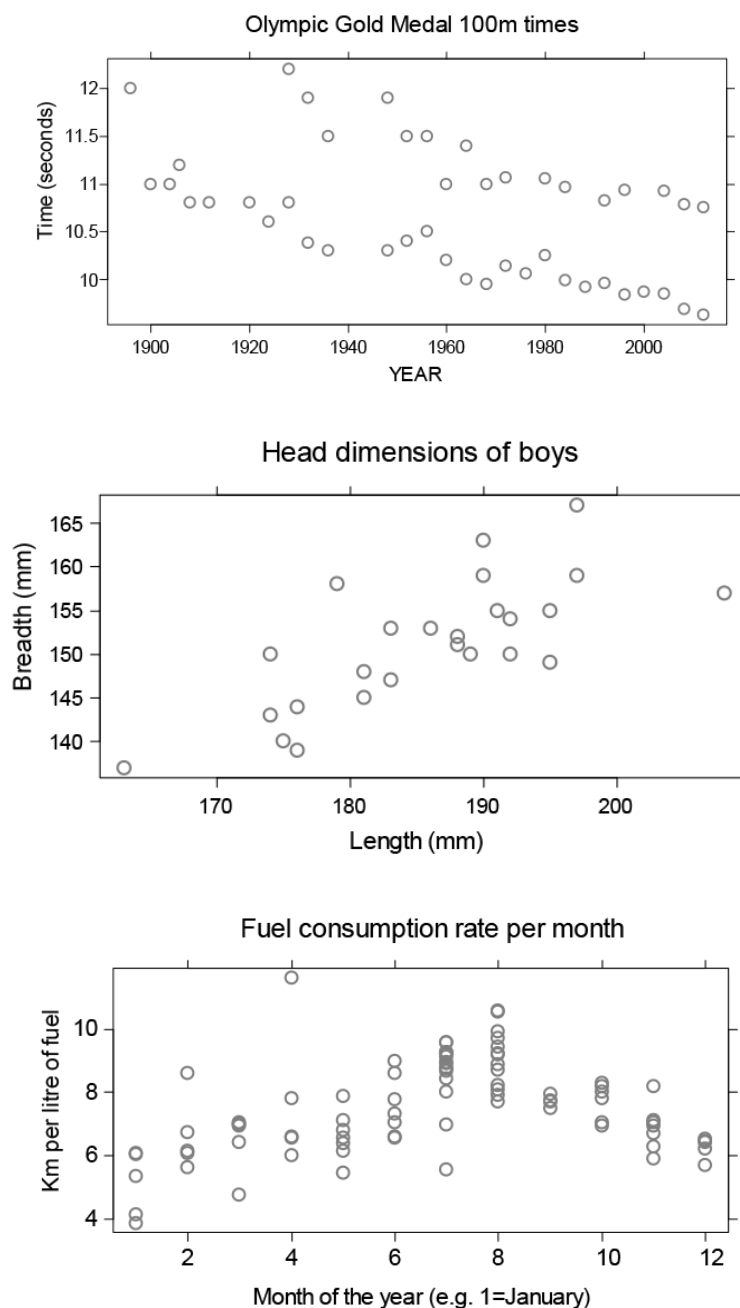
Try working with other variables.

Observations of scatterplots

Not all scatterplots show obvious trends with uniform scatter.

For each scatterplot below, train your eye to look for trends, clusters, outliers and draw an envelope around the scatter. You may want to print them out so that you can draw on the above graphs.

For each graph, what do you notice? Post one of your findings on the discussion.



To discuss R issues related to this Exercise,

go to <https://gitter.im/iNZightVIT/d2i-R-discussion>

If your question relates to an Exercise, say which one you are talking about!

4.7 Exercise: Techniques for scatterplots

– R version

This exercise will enable you to become more proficient in creating scatterplots with iNZightPlot. You will learn how to apply the most suitable trend line and use techniques to overcome perceptual problems.

The skills addressed are:

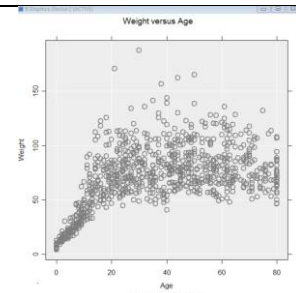
1. Create a scatterplot of two numeric variables and apply a suitable trend line.
2. Use techniques such as jittering, transparency and running quartiles to deal with overprinting.

We are going to explore the relationship between variables **Age** and **Weight** of people in the **nhanes_1000** dataset in the **FutureLearnData** package.

# R code	Output and/or Commentary
<pre># Setup library(iNZightPlots) library(FutureLearnData) data(nhanes_1000)</pre>	

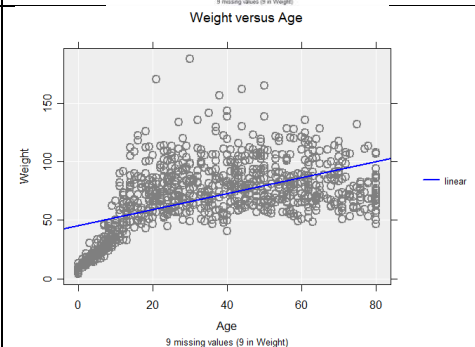
Plot Weight vs Age

```
iNZightPlot(Age, Weight, data=nhanes_1000)
```



Add a trend line

```
iNZightPlot(Age, Weight, data=nhanes_1000, trend="linear")
```



Get some summary information (result depends on trends fitted)

```
getPlotSummary(Age, Weight, data=nhanes_1000, trend="linear")
```

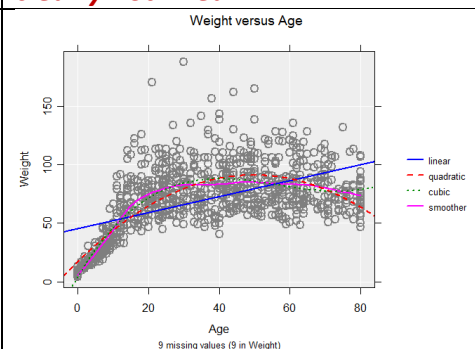
```
getPlotSummary(Age, Weight, data=nhanes_1000, trend="linear",  
summary.type="inference")
```

```
> getPlotSummary(Age, Weight, data=nhanes_1000, trend="linear")
iNZight Summary
-----
Response/outcome variable: Weight (numeric)
Predictor/explanatory variable: Age (numeric)
Total number of observations: 1000
Number omitted due to missingness: 9 (9 in Weight)
Total number of observations used: 991
-----
Summary of Weight versus Age:
-----
Linear trend:
  Weight = 45.17 + 0.6527 * Age
  Linear correlation: 0.53
Rank correlation: 0.52 (using Spearman's Rank Correlation)
-----
```

The inference results make no sense here as the real trend is clearly not linear

Add more trend curves and a smoother

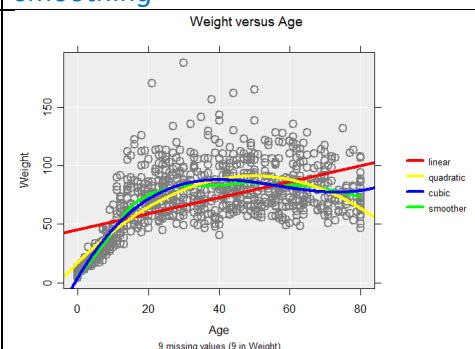
```
iNZightPlot(Age, Weight, data=nhanes_1000, trend=c("linear",  
"quadratic", "cubic"), smooth=.25)
```



The value given for the smooth (between 0 and 1) controls the level of smoothing. Bigger numbers correspond to more smoothing

Make all the lines thicker, all solid lines, and change line colours

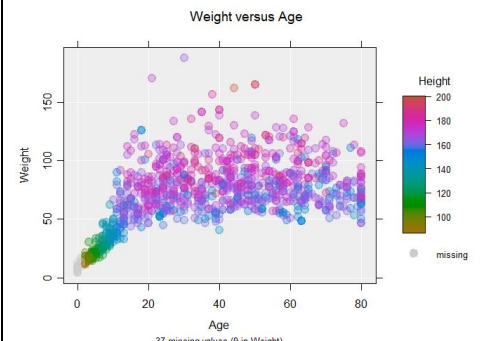
```
iNZightPlot(Age, Weight, data=nhanes_1000, trend=c("linear",  
"quadratic", "cubic"), smooth=.25, lwd=2,  
lty.trend=list(linear=1, quadratic=1, cubic=1),  
col.trend=list(linear="red", quadratic="yellow", cubic="blue"),  
col.smooth="green")
```



Scatter plot coloured by Height and with Transparency

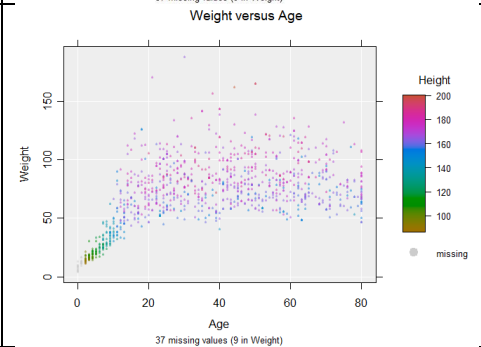
```
iNZightPlot(Age, Weight, data=nhanes_1000, colby=Height, alpha=.3)
```

alpha (0 to 1) controls transparency. Smaller for more transparent



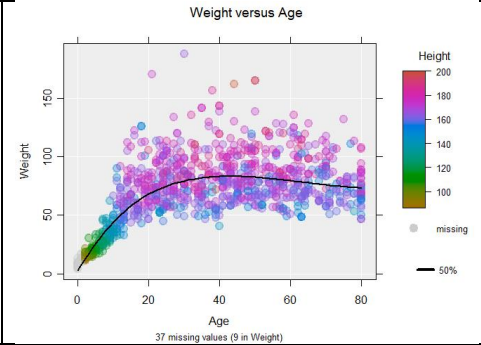
Make the points smaller

```
iNZightPlot(Age, Weight, data=nhanes_1000, colby=Height, alpha=.3,
  cex.pt=.2)
```



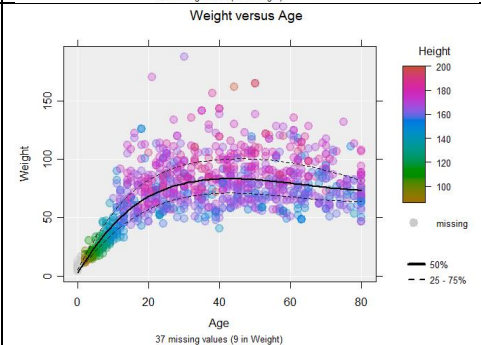
Create a median smoother in black

```
iNZightPlot(Age, Weight, data=nhanes_1000, colby=Height, alpha=.3,
  quant.smooth=.5, col.smooth="black")
```

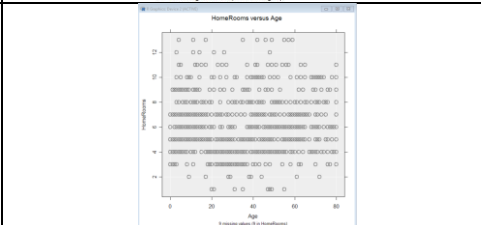


Create a median and quartile smoothers all in black

```
iNZightPlot(Age, Weight, data=nhanes_1000, colby=Height, alpha=.3,
  quant.smooth=c(.25,.5,.75), col.smooth="black")
```

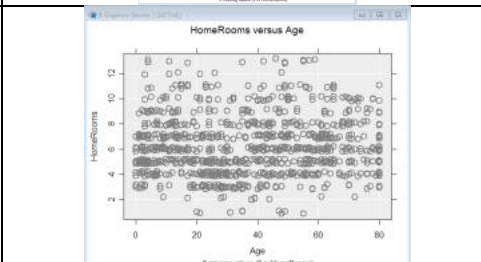


```
iNZightPlot(Age, HomeRooms, data=nhanes_1000)
```



Jitter in the vertical (y) direction

```
iNZightPlot(Age, HomeRooms, data=nhanes_1000, jitter="y")
```



- **Play some more with these settings and try other variables**
 - For even more settings, type **?inzpar** into R to get help on the **inzpar**, or type **inzpar** to just get a complete list (last time I looked the help file wasn't entirely complete)
-

To discuss issues related to this Exercise,

go to <https://gitter.im/iNZightVIT/d2i-R-discussion>

*To be able to post to the list you will have to set up a (free) account on **Github***

<https://github.com/login>

If your question relates to an Exercise, say which one you are talking about!

4.11 Exercise: Advanced scatterplots for deeper analysis – R version

This exercise will enable you to explore more complicated relationships between variables and the effects of a third and fourth variable, enabling you to view changes over time.

The skills addressed are:

1. Create a scatterplot of two numeric variables, subset by a 3rd variable.
2. Explore the effect of a third and fourth variable using colour and size.

We will use the **gapminder** dataset (but **not** *gapminder_2008*).

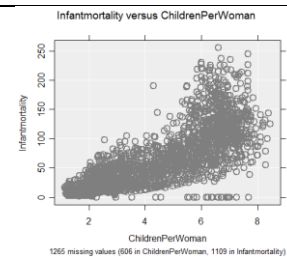
Create a scatterplot of two numeric variables, subset by a 3rd variable

We are going to explore the relationship between the variables **Infantmortality** and **ChildrenPerWoman** of countries in the **Gapminder** dataset over time.

#R Code	Output and/or Commentary
<pre># Setup library(iNZightPlots) library(FutureLearnData) data(gapminder)</pre>	

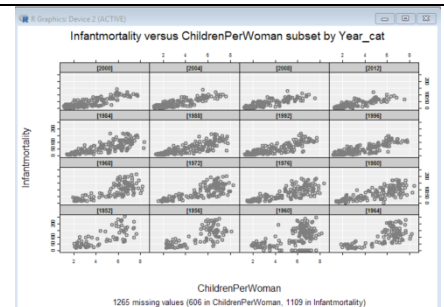
Scatterplot of *Infantmortality* against *ChildrenPerWoman*

```
iNZightPlot(ChildrenPerWoman,Infantmortality ,
data=gapminder)
```



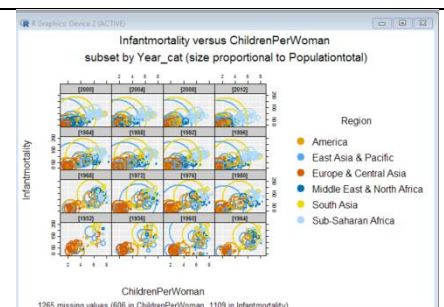
Subset by *Year_cat*

```
iNZightPlot(ChildrenPerWoman,Infantmortality, g1=Year_cat,
data=gapminder)
```



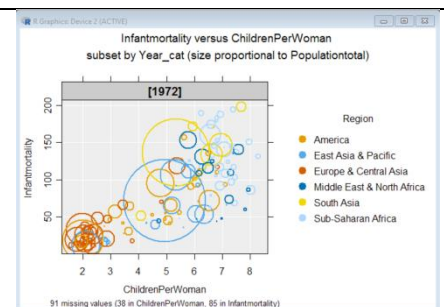
Change size and colour of points

```
iNZightPlot(ChildrenPerWoman,Infantmortality,g1=Year_cat,
data=gapminder, colby=Region, sizeby=Populationtotal)
```



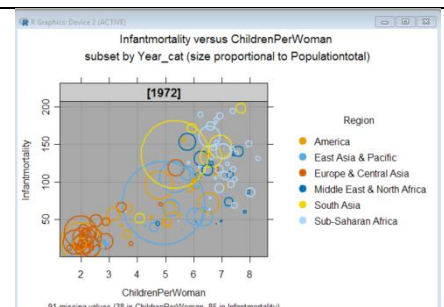
Show results for 1972 only

```
iNZightPlot(ChildrenPerWoman,Infantmortality,g1=Year_cat,
g1.level="[1972]",data=gapminder, colby=Region,
sizeby=Populationtotal)
```



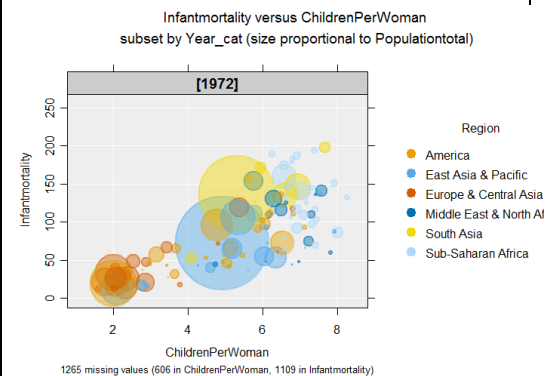
Darker background (often easier to see some of the lighter dots)

```
iNZightPlot(ChildrenPerWoman,Infantmortality, g1=Year_cat,
g1.level="[1972]",data=gapminder, colby=Region,
sizeby=Populationtotal, bg="darkgray")
```



Try transparency and smaller points (removed garkgray)

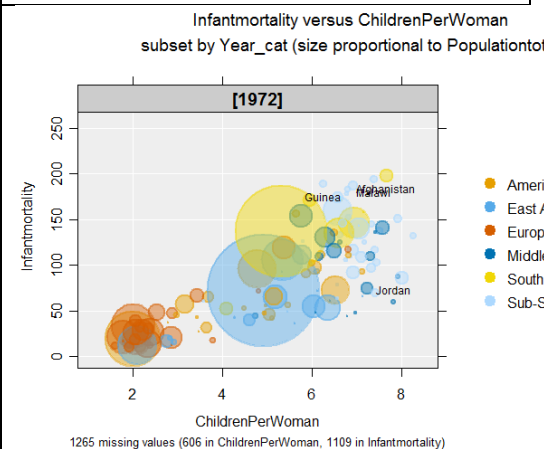
```
iNZightPlot(ChildrenPerWoman,Infantmortality,g1=Year_cat,
  g1.level="[1972]",data=gapminder,colby=Region,
  sizeby=Populationtotal,alpha=.45,cex.dotpt=.5)
```



Try subsetting by different years, e.g. g1.level="[1976]",

Label some of the extreme points (ask for 4)

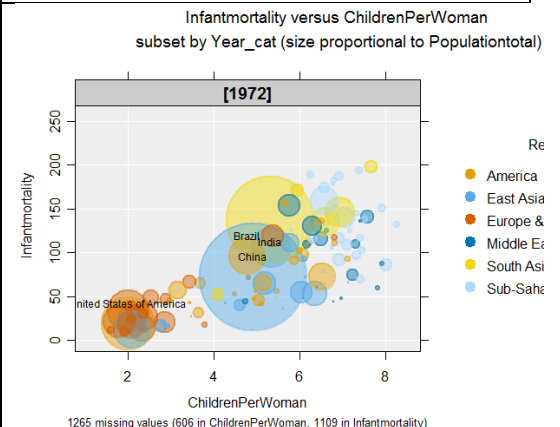
```
iNZightPlot(ChildrenPerWoman,Infantmortality,g1=Year_cat,
  g1.level="[1972]",data=gapminder,colby=Region,
  sizeby=Populationtotal,alpha=.45,cex.dotpt=.5,
  locate.extreme=4,locate=Country)
```



Label some specific countries

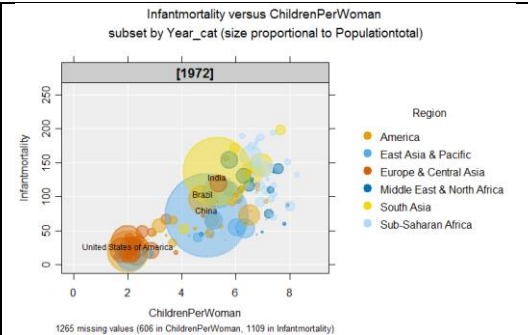
```
ids = (1:nrow(gapminder))[gapminder$Country %in% c("United States of America", "China", "Brazil", "India")]
```

```
iNZightPlot(ChildrenPerWoman,Infantmortality,g1=Year_cat,
  g1.level="[1972]",data=gapminder,colby=Region,
  sizeby=Populationtotal,alpha=.45,cex.dotpt=.5,
  locate.id=ids,locate=Country)
```



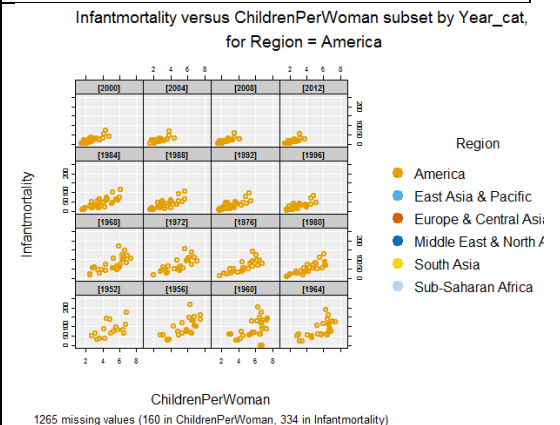
Allow a little more room on left to accommodate label

```
iNZightPlot(ChildrenPerWoman, Infantmortality, g1=Year_cat,
             g1.level="[1972]", data=gapminder, colby=Region,
             sizeby=Populationtotal, alpha=.45, cex.dotpt=.5,
             locate.id=ids, locate=Country, xlim=c(0,9))
```



Subset by a fourth variable (Region)

```
iNZightPlot(ChildrenPerWoman, Infantmortality, g1=Year_cat,
             g2=Region, g2.level="America", data=gapminder, colby=Region)
```



Play through the years

```
for (k in levels(gapminder$Year_cat)) {
  iNZightPlot(ChildrenPerWoman, Infantmortality, g1=Year_cat,
              g1.level=k, data=gapminder, colby=Region,
              sizeby=Populationtotal, alpha=.45, cex.dotpt=.5,
              locate.id=ids, locate=Country)
  Sys.sleep(1)
}
```

- Play some more with these settings and try other variables
- For even more settings, type `?inzpar` into R to get help on the `inzpar`, or type `inzpar` to just get a complete list (last time I looked the help file wasn't entirely complete)

To discuss issues related to this Exercise,

go to <https://gitter.im/iNZightVIT/d2i-R-discussion>

To be able to post to the list you will have to set up a (free) account on **Github**

<https://github.com/login>

If your question relates to an Exercise, say which one you are talking about!

6.12 Exercise: Inference with iNZight – R version

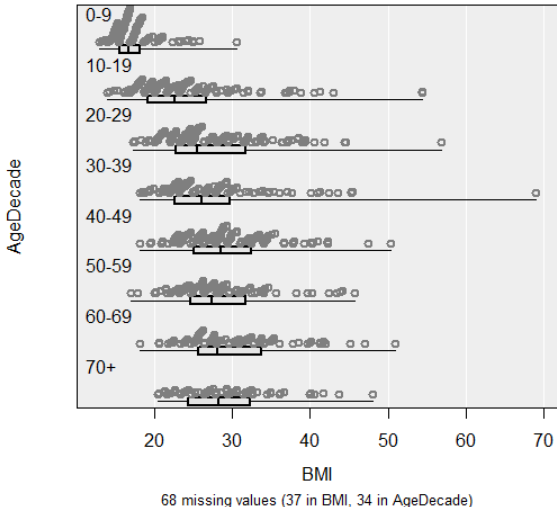
This exercise will enable you to make comparisons between sub-groups allowing for sampling error.

Background understanding: see **Step 6.9**.

The skills addressed in this Exercise are:

- Use iNZightPlot to get inferential mark-ups of plots so that you can make visual comparisons between sub-groups allowing for sampling error.
- To obtain numerical confidence limits for true between-group differences.

We will use the **nhanes_1000** dataset from the **FutureLearnDatasets** package.

#R code	Output and/or Commentary
<pre> # Setup library(iNZightPlots) library(FutureLearnData) data(nhanes_1000) names(nhanes_1000) </pre>	
<pre> # Plot BMI by AgeDecade iNZightPlot(AgeDecade, BMI, data=nhanes_1000) </pre>	<p>BMI by AgeDecade</p>  <p>68 missing values (37 in BMI, 34 in AgeDecade)</p>

Add inference information

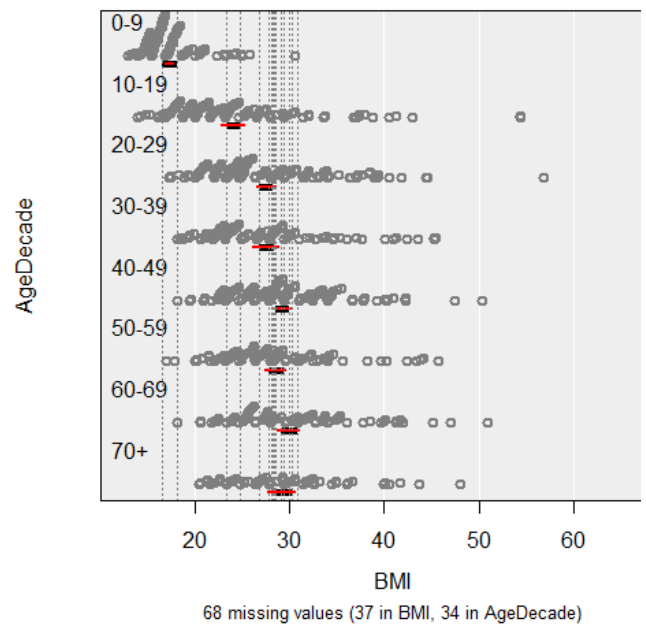
```
iNZightPlot(AgeDecade, BMI, data=nhanes_1000,  
  inference.type=c("comp","conf"),  
  inference.par="mean")
```

Commentary (on this side to save space)

You can **squash** your plot window **vertically** so that it is **easier to see** how much **overlap** there is between each age group.

What do you see here? The **thick black lines** are called '**comparison intervals**' and are the lines that we look at when are the lines that we **look at** when observing any **overlap**. The **thin red lines** are the **individual confidence intervals** for each mean/median.

BMI by AgeDecade



View detailed inferential information (Normal Theory)

```
getPlotSummary(AgeDecade, BMI, data=nhanes_1000,  
  summary.type="inference", inference.type="conf")
```

```
-----  
iNZight Inference using Normal Theory  
-----  
Primary variable of interest: AgeDecade (categorical)  
Secondary variable: BMI (numeric)  
  
Total number of observations: 1000  
Number omitted due to missingness: 68 (34 in AgeDecade, 37 in BMI)  
Total number of observations used: 932  
-----  
Inference of BMI by AgeDecade:  
-----  
  
Group Means with 95% Confidence Intervals  
  
      Lower   Mean   Upper  
0-9      16.86   17.36   17.85  
10-19    22.78   24.05   25.31  
20-29    26.49   27.57   28.64  
30-39    26.16   27.58   29.01  
40-49    28.35   29.28   30.21  
50-59    27.46   28.58   29.71  
60-69    28.74   29.98   31.21  
etc
```

We'll now do this with a pair of Categorical variables

Filter out under 20s

```
Temp=subset(nhanes_1000, AgeDecade != "0-9" &  
  AgeDecade != "10-19")  
Temp$AgeDecade=factor(Temp$AgeDecade)
```

Reorder the HealthGen variable

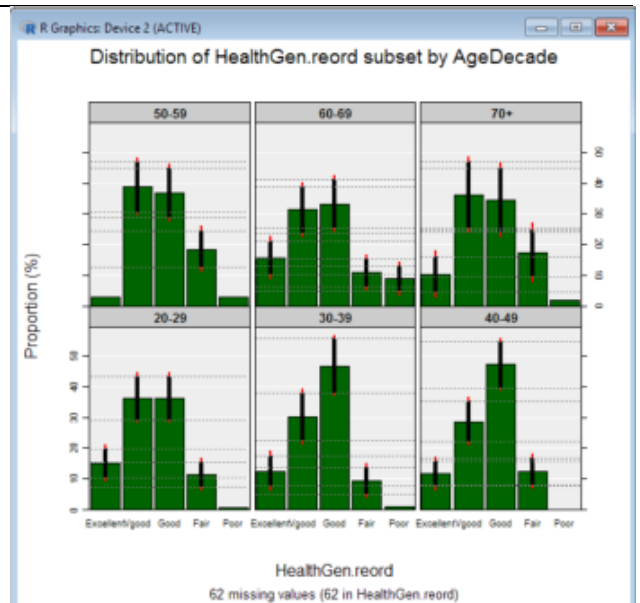
```
Temp$HealthGen.reord = factor(Temp$HealthGen, levels  
  = c("Excellent", "Vgood", "Good", "Fair", "Poor") )
```

First need to do some filtering to get rid of decades with no General Health data

and create HealthGen.reord with the levels in a sensible order

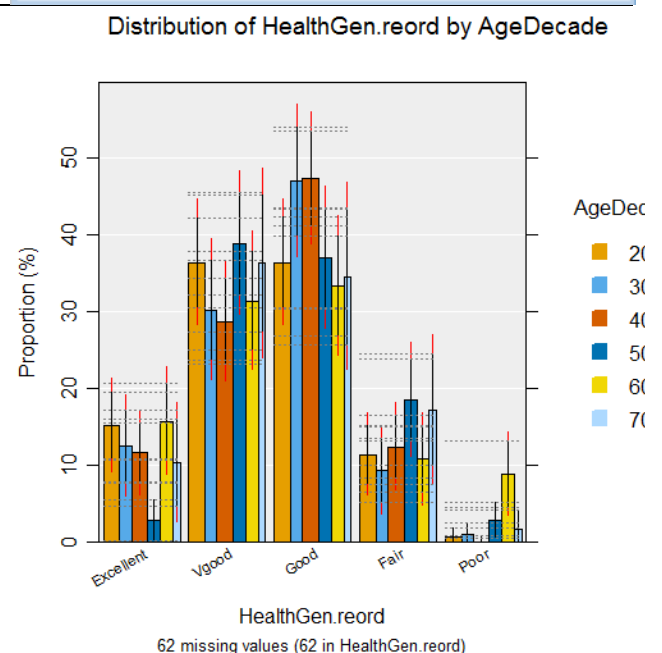
Plot as separate bar charts with inference information

```
iNZightPlot(HealthGen.reord, g1=AgeDecade, data=Temp,
  inference.type=c("comp", "conf") )
```



Plot as side by side without inference comparison

```
iNZightPlot(HealthGen.reord, AgeDecade, data=Temp,
  inference.type=c("comp", "conf") )
```



- Play some more with these settings and try other variables

To discuss issues related to this Exercise,

go to <https://gitter.im/iNZightVIT/d2i-R-discussion>

To be able to post to the list you will have to set up a (free) account on **Github**

<https://github.com/login>

If your question relates to an Exercise, say which one you are talking about!

Exercise: Time Series for a single variable – *R* version

In this exercise we will use **iNZightTS** package to create a Time Series plot and get the Additive and Multiplicative Decomposition for it.

We will use the **Week8_AverageVisitorsQuarterly** dataset from the **FutureLearnData** package.

The skills addressed are to use the **iNZightTS** package to:

1. Generate a Time Series plot and a Seasonal plot for a single numeric variable.
2. Get an Additive and Multiplicative Decomposition.
3. Make a forecast.

Generate a Time Series plot and a Seasonplot for a single numeric variable

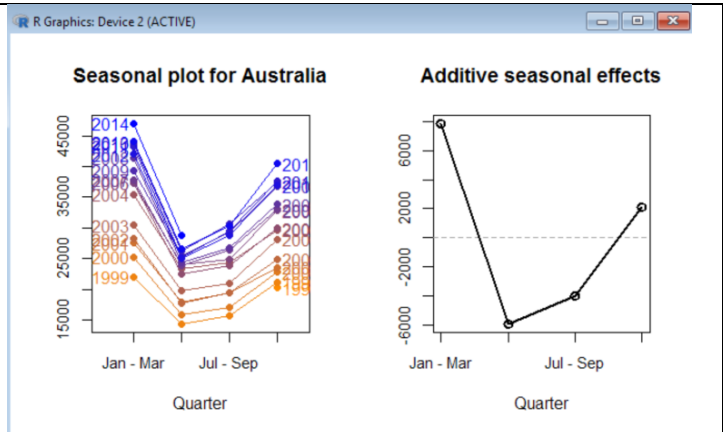
The data we are using shows us the number of visitors from different countries who are currently staying in New Zealand. We will investigate the changes in the number of Australian visitors over time.

8.6 Time Series for a single variable

# R Code	Output and/or Commentary
<pre># Install the iNZightTS package install.packages(c('iNZightTS'), dependencies = TRUE, repos = c('http://r.docker.stat.auckland.ac.nz/R', 'https://cran.rstudio.com')) # Load the Time Series Library library(iNZightTS) library(FutureLearnData) # List the course datasets and select dataset data(package="FutureLearnData") data(week8_AverageVisitorsQuarterly) head(week8_AverageVisitorsQuarterly)</pre>	<p>Commentary</p> <p><i>This installation only has to be done once</i></p> <p><i>Load packages in every new R session in which you want to use them</i></p> <p><i>Look at summary of the data sets to ensure you have the name exactly correct</i></p> <p><i>Load the data set (which is in the FutureLearnData package)</i></p> <p><i>List the top few rows</i></p> <pre>> head(week8_AverageVisitorsQuarterly) Time Australia China.PR Japan Rep.Korea Germany UK Canada USA 1 1998Q4 20288 1089 5938 1357 4376 13831 2196 7465 2 1999Q1 22047 1492 6925 2189 6591 23271 3846 10969 3 1999Q2 14362 1450 4353 1287 1787 9756 1285 5498 4 1999Q3 15775 1551 6855 1767 1169 7899 1210 4811 5 1999Q4 21209 2020 6216 2339 4998 15778 2748 9568 6 2000Q1 25261 2364 7061 4075 7740 25362 4147 13700 > </pre>
<pre>Australia = iNZightTS (week8_AverageVisitorsQuarterly, var=2) # Plot the data -- t controls smoothing rawplot(Australia , t=25) # Now experiment with different values of t # If you are on Windows and using R alone (not RStudio) try ... rawplot(Australia , t=25,animate=TRUE)</pre>	<p>Commentary</p> <p><u>Create Time Series object for the Australian series</u> <i>Australia is the 2nd variable in the dataset</i></p> <p><i>t controls the smoothing (t must be betw. 0 and 100)</i></p>  <p>Time series plot for Australia</p>

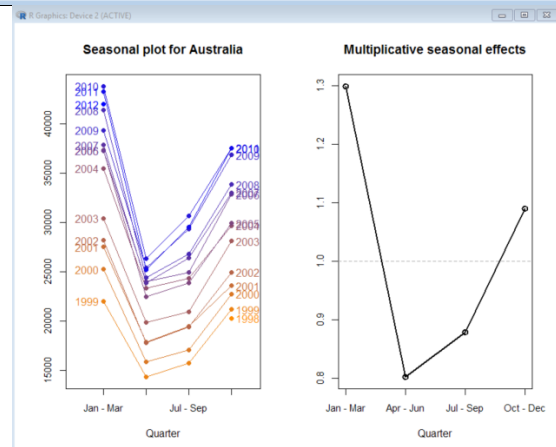
Seasonal plot – additive

`seasonplot(Australia)`



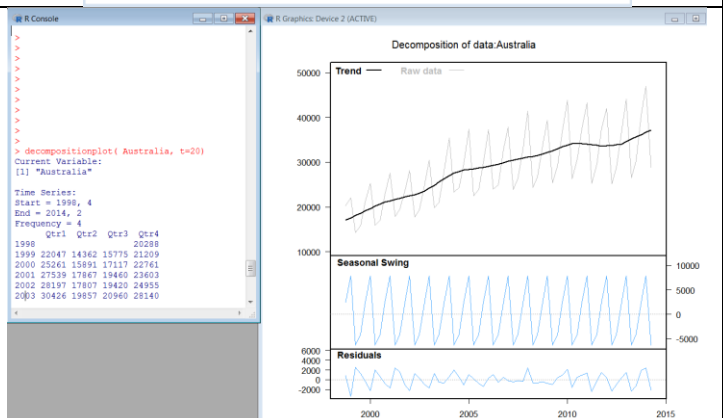
Seasonal plot – multiplicative

`seasonplot(Australia , multiplicative=TRUE)`



Decomposition plot

`decompositionplot(Australia, t=20)`

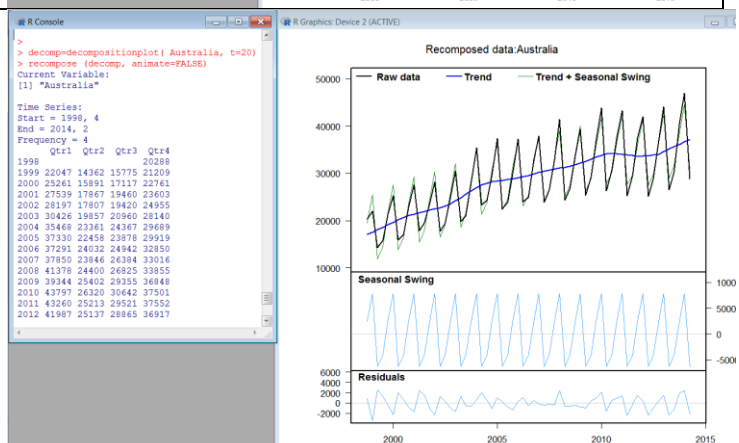


Recomposed plot

`decomp=decompositionplot(Australia, t=20)`
`recompose(decomp, animate=FALSE)`

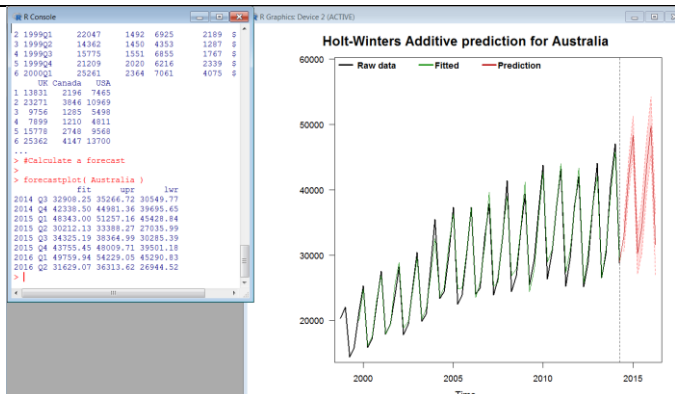
If you are on Windows and using R alone (not RStudio) try ...

`recompose(decomp, animate=TRUE)`



Calculate a forecast

forecastplot(Australia)



Let's establish this pattern for another country

China = iNZightTS (week8_AverageVisitorsQuarterly, var=3)

rawplot(China , t=20)

decompositionplot(China, t=20)

etc ...

Commentary

Create Time Series object for the China series. China is the 3rd variable in the dataset week8_AverageVisitorsQuarterly

Now you can start plotting ...

- Repeat what we have done above for any other country that interests you and try to interpret the patterns you see as has been done in the video
 - Skim-read the **iNZight version** for the **commentary** that is missing here. (This document just concentrates on how the code works)

[In the next Exercise we will start comparing series from different countries.]

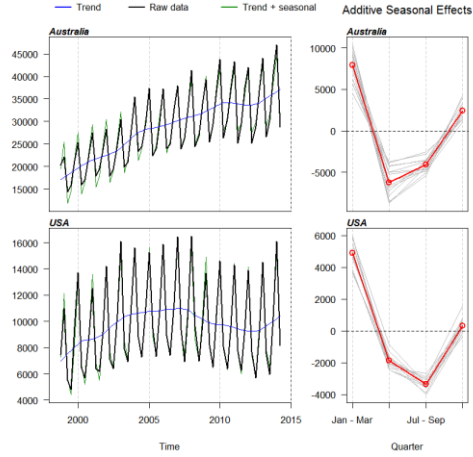
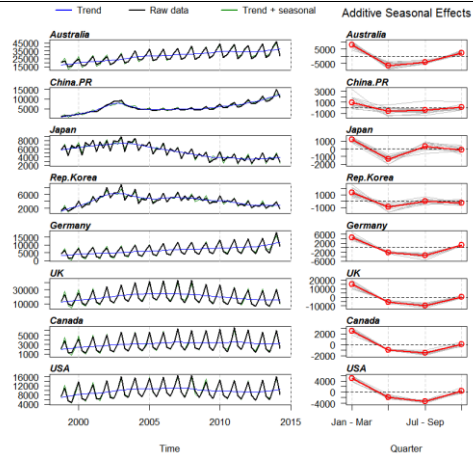
Exercise: Time Series analysis for more than one series – R version

In the previous exercise we only looked at the visitors from one country alone. Now we want to see the graphs for more than one, to be able to compare their visitor numbers.

This exercise will enable you to use iNZight to compare several time series by viewing them simultaneously in two different ways.

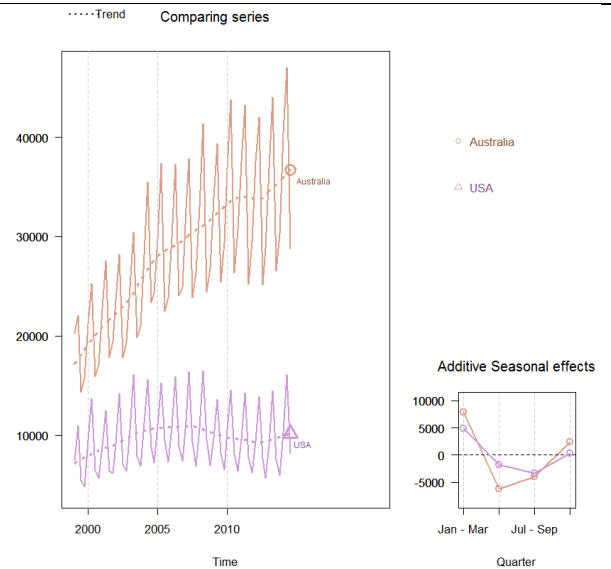
We will again use the **Week8_AverageVisitorsQuarterly** dataset from the **FutureLearnData** package.

# R Code	Output and/or Commentary
<pre># Set up library(iNZightTS) library(FutureLearnData) data(week8_AverageVisitorsQuarterly)</pre>	
<pre># See the variables in this dataset head(week8_AverageVisitorsQuarterly)</pre>	<pre>> head(week8_AverageVisitorsQuarterly) Time Australia China.FR Japan Rep.Korea Germany UK Canada USA 1 1998Q4 20288 1089 5938 1357 4376 13831 2196 7465 2 1999Q1 22047 1492 6925 2189 6591 23271 3846 10969 3 1999Q2 14362 1450 4353 1287 1787 9756 1285 5498 4 1999Q3 15775 1551 6855 1767 1169 7899 1210 4811 5 1999Q4 21209 2020 6216 2339 4998 15778 2748 9568 6 2000Q1 25261 2364 7061 4075 7740 25362 4147 13700</pre>
<pre># See how these work ... c(2,5,9) c(2,4:6, 8)</pre>	<p><i>We are going to use this idea to specify the column numbers corresponding to the countries we want to look at</i></p>

<pre>Aus_USA = iNZightTS(week8_AverageVisitorsQuarterly, var=c(2,9))</pre>	<p><u>Create Time Series object for the set of countries we want to look at</u></p> <p><u>selecting columns 2 and 9 (which correspond to Australia and USA).</u></p> <p><u>Let's call it Aus_USA</u></p>
<p><i># Separate plots for Aus_USA</i></p> <pre>multiseries(Aus_USA, t=20)</pre>	
<pre>ALL = iNZightTS(week8_AverageVisitorsQuarterly, var=c(2:9))</pre>	<p><u>Create Time Series object for the whole set of countries we want to look at</u></p> <ul style="list-style-type: none"> <u>selecting all columns from 2 to 9.</u> <u>Let's call it ALL</u>
<p><i># Separate plots for ALL</i></p> <pre>multiseries(ALL, t=20)</pre>	

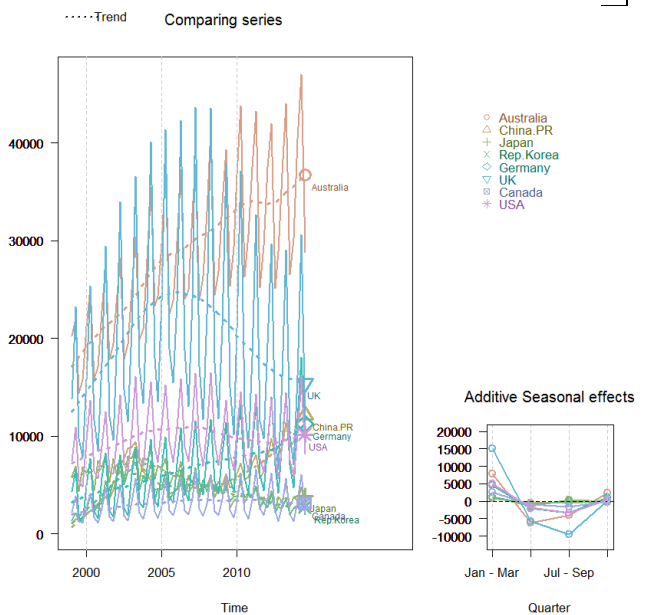
Separate plots for Aus_USA

compareplot(Aus_USA, t=30)



Separate plots for ALL

compareplot(ALL, t=30)



- Repeat what we have done above for any other *combinations of countries* that interest you and try to interpret the patterns you see as has been done in the video
 - Skim-read the **iNZight version** for the **commentary that is missing here**. (This document just concentrates on how the code works)
 - and for **exploration questions**