DATA SOCIETY®

Interactive visualization with R - Part 1

One should look for what is and not what he thinks should be.
-Albert Einstein.

Rising Data Science Interest

- With the rise of COVID-19, the need for fast visualizations to put data into perspective has also risen
- Have a look at this ***Visualization article*** while we wait to start class to get a peek at what's ahead.

Recap

- In the last module, we:
  - Transformed data using `tidyverse`
  - Plotted the transformed and tidy data to produce complex visualizations
- In this module, we will create plots using the `highcharter` package

Module completion checklist

Objective	Complete
Introduce using the highcharter package to build interactive visualizations	
Use highcharter with tidy data to create a scatterplot	
Visualize a correlation plot with hchart	
Build a column plot with hchart	
Create a boxplot with hchart	
Save interactive plots with htmlwidgets	

Directory settings

- In order to maximize the efficiency of your workflow, you may want to encode your directory structure into `variables`
- Let the `main_dir` be the variable corresponding to your `skillsoft` folder

```
# Set `main_dir` to the location of your `skillsoft` folder (for Mac/Linux).
main_dir = "~/Desktop/skillsoft"
# Set `main_dir` to the location of your `skillsoft` folder (for Windows).
main_dir = "C:/Users/[username]/Desktop/skillsoft"
# Make `data_dir` from the `main_dir` and remainder of the path to data directory.
data_dir = paste0(main_dir, "/data")
# Make `plots_dir` from the `main_dir` and remainder of the path to plots directory.
plot_dir = paste0(main_dir, "/plots")
# Set directory to data_dir.
setwd(data_dir)
```

Interactive visualizations with highcharter

- Similar to `ggplot2` and other plotting libraries (including base R itself), `highcharter` allows us to build complex, customized, and meaningful visualizations with the help of **layers** of graphical elements
- `Highcharter` is an R wrapper that allows R users to tap into one of the most comprehensive data visualization JavaScript-based libraries: ***Highcharts***
- `Highcharts` is free for individual research and non-profit purposes, but there are cases and restrictions to its use and you may need to obtain a license if you decide to integrate it into software or company-wide products

```
> library(highcharter)
Highcharts (www.highcharts.com) is a Highsoft software product which is
not free for commercial and Governmental use
>
```

Interactive visualizations: highcharter

Let's install the package and check its documentation

```
# Install `highcharter` package.
install.packages("highcharter")

# Load the library.
library(highcharter)

# View documentation.
library(help = "highcharter")
```

Documentation for package 'high...

```
                Information on package 'highcharter'

Description:

Package:            highcharter
Type:               Package
Version:            0.5.0
Title:              A Wrapper for the 'Highcharts' Library
Description:        A wrapper for the 'Highcharts' library including shortcut functions t
                    <http://www.highcharts.com/> is a charting library offering numerous
                    configuration syntax.
Authors@R:          c( person("Joshua", "Kunst", role = c("aut", "cre"), email = "jbkunst
                    "Agostinho", role = c("ctb"), comment = "hchart.survfit, densities an
                    person("Eduardo", "Flores", role = c("ctb"), comment = "First version
                    person("Dean", "Kilfoyle", role = c("ctb"), comment = "First version
                    person("Adline", "Dsilva", role = c("ctb"), comment = "First version
Date:               2016-09-05
URL:                http://jkunst.com/highcharter
BugReports:         https://github.com/jbkunst/highcharter/issues
Maintainer:         Joshua Kunst <jbkunst@gmail.com>
License:            MIT + file LICENSE
RoxygenNote:        5.0.1
Depends:            R (>= 2.10)
Imports:            htmlwidgets, magrittr, purrr, rlist, assertthat, zoo, dplyr, tibble (
                    quantmod, tidyr, htmltools, jsonlite, igraph, lubridate
Suggests:           knitr, rmarkdown, survival, ggplot2, httr, viridisLite
VignetteBuilder:    knitr
```

Highcharter main function: highchart

```
?highchart
```

- Similar to `ggplot2`, in order to create a plot, we need to call the main plotting function `ggplot()`
- In `highcharter` it is `highchart()`
- The function doesn't need any required arguments, and all of the graphic parameters and plotting options can be specified within layers themselves

Files | Plots | Packages | Help | Viewer

R: Create a Highcharts chart widget Find in Topic

highchart {highcharter} R Documentation

Create a Highcharts chart widget

Description

This function creates a Highchart chart using **htmlwidgets**. The widget can be rendered on HTML pages generated from R Markdown, Shiny, or other applications.

Usage

```
highchart(hc_opts = list(), theme = getOption("highcharter.theme"),
  type = "chart", width = NULL, height = NULL, elementId = NULL)
```

Arguments

hc_opts	A list object containing options defined as http://api.highcharts.com/highcharts.
theme	A hc_theme class object
type	A character value to set if use Highchart, Highstock or Highmap. Options are "chart", "stock" and "map".
width	A numeric input in pixels.
height	A numeric input in pixels.
elementId	Use an explicit element ID for the widget.

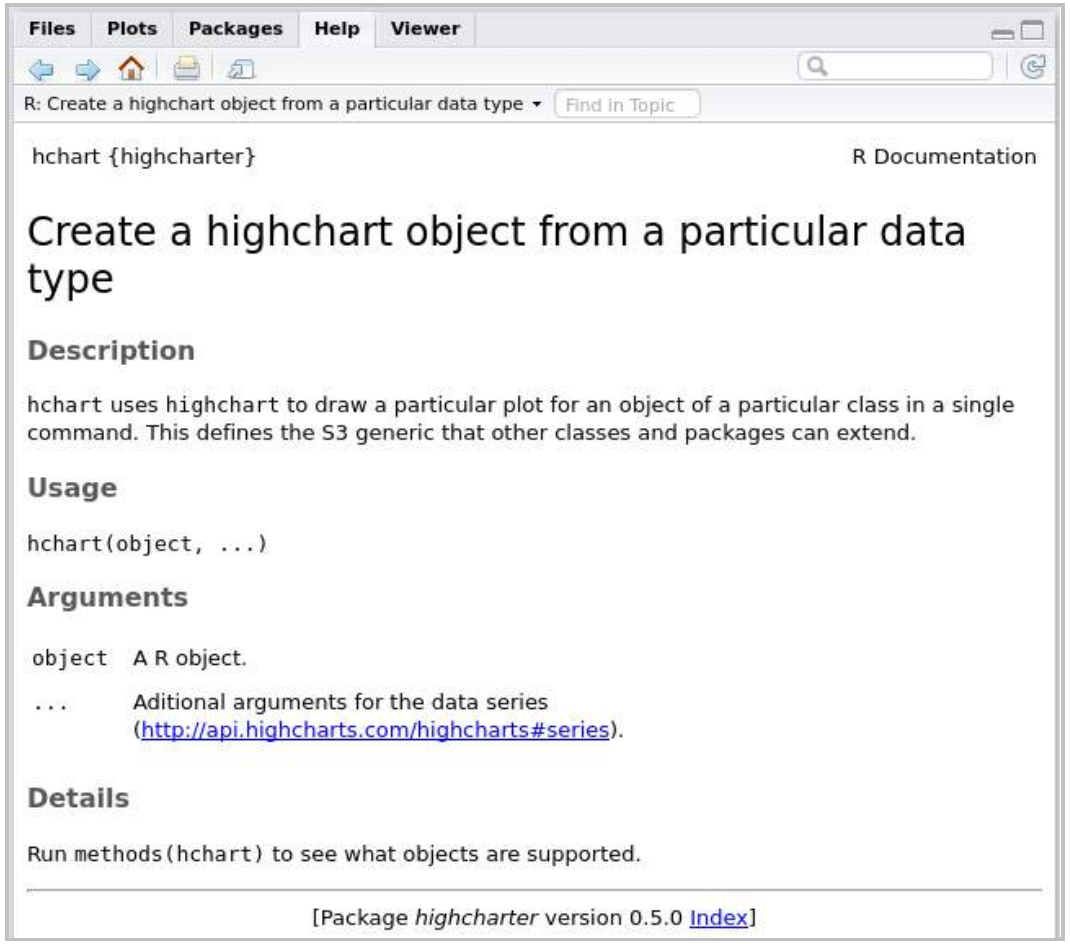
[Package *highcharter* version 0.5.0 Index]

Highcharter: `hchart` vs `highchart`

```
?hchart

hchart(Some_data,        #<- dataset to use
       "plot_type",      #<- plot type to use
       hcaes(x = variable1, #<- x-axis mapping
             y = variable2, #<- y-axis mapping
             group = variable3, #<- group by
             ...))
```

`hchart` is a shorthand version of the `highchart` function, which takes a few key arguments to plot:

1. Data to use
2. Type of plot to create (e.g., `scatter`, `bar`, `column`, `line`, etc.)
3. `hcaes` (i.e., highcharts aesthetics) mapping of variables (works exactly the same way as with `ggplot2`!)



Files | Plots | Packages | Help | Viewer

R: Create a highchart object from a particular data type Find in Topic

hchart {highcharter} R Documentation

Create a highchart object from a particular data type

Description

hchart uses highchart to draw a particular plot for an object of a particular class in a single command. This defines the S3 generic that other classes and packages can extend.

Usage

```
hchart(object, ...)
```

Arguments

object	A R object.
...	Aditional arguments for the data series (http://api.highcharts.com/highcharts#series).

Details

Run methods(hchart) to see what objects are supported.

[Package *highcharter* version 0.5.0 Index]

Layers in Highcharts: series

- Just like `ggplot2`, the `highcharter` library has its own vocabulary
- Each new data / graphic layer in `highcharter` is called a **series**
- Each series can be of different `type`. Here are some widely used ones:

Highcharter series type	Plot type
scatter	scatterplot
line	line graph
boxplot	boxplot
column	bar plot
bar	horizontal bar plot
histogram	histogram
area	density

Module completion checklist

Objective	Complete
Introduce using the highcharter package to build interactive visualizations	✔
Use highcharter with tidy data to create a scatterplot	
Visualize a correlation plot with hchart	
Build a column plot with hchart	
Create a boxplot with hchart	
Save interactive plots with htmlwidgets	

Set up: load & prepare data

- Let's load the `CMP_subset` dataset from our `data_dir` into R's environment

```
# Set working directory to where we store
data.
setwd(data_dir)

library(tidyverse)

# Read CSV file
CMP_subset = read.csv("CMP_subset.csv",
            header = TRUE)
```

Now, tidy the data as before and transform it from wide to long for easy visualization

```
# Prep data for univariate plots
CMP_subset_long = CMP_subset %>%
  gather(key = "variable",
         value = "value")

# Make names of processes and materials more user
friendly and readable.
CMP_subset_long = CMP_subset_long %>%
     mutate(variable =
            str_replace(variable,
                        "Biological", "Bio ")) %>%
     mutate(variable =
            str_replace(variable,
                        "Manufacturing", "Man. "))
%>%
     mutate(variable =
            str_replace(variable,
                        "0", " ")) %>%
    group_by(variable) %>%              #<- normalize
    mutate(norm_value =
           value/max(value, na.rm = TRUE))
```

Set up: prepare data

```
# Prep data for scatterplot
CMP_subset_long2 = CMP_subset %>%
  gather(BiologicalMaterial01:ManufacturingProcess03,
         key = "variable",
         value = "value") %>%
  # All other transformations we've done before.
  mutate(variable = str_replace(variable, "Biological", "Bio ")) %>%
  mutate(variable = str_replace(variable, "Manufacturing", "Man. ")) %>%
  mutate(variable = str_replace(variable, "0", " ")) %>%
  group_by(variable) %>%
  mutate(norm_value = value/max(value, na.rm = TRUE))

head(CMP_subset_long2,3)
```

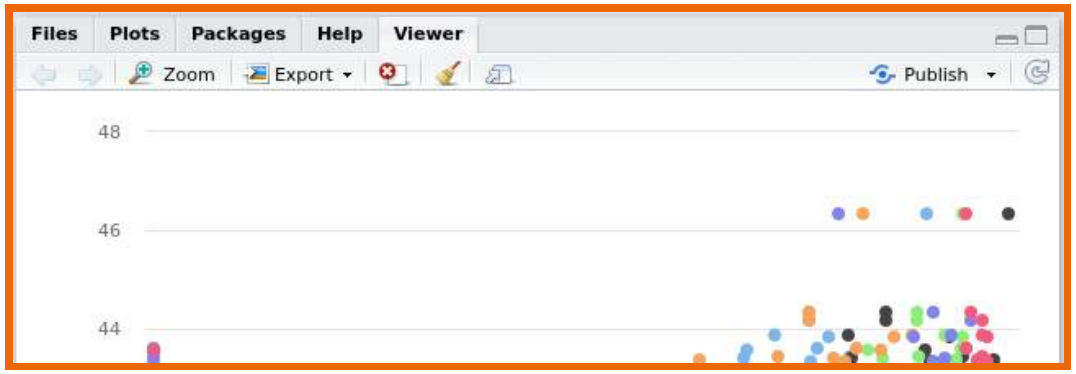
```
# A tibble: 3 x 4
# Groups:   variable [1]
  Yield variable       value norm_value
  <dbl> <chr>          <dbl>      <dbl>
1  38   Bio Material 1  6.25      0.709
2  42.4 Bio Material 1  8.01      0.909
3  42.0 Bio Material 1  8.01      0.909
```

hchart: scatterplot

To construct a scatterplot with `highcharter`, we use the `hchart()` function and pass the **data**, **plot type** (`scatter`), and **aesthetics** to it as arguments

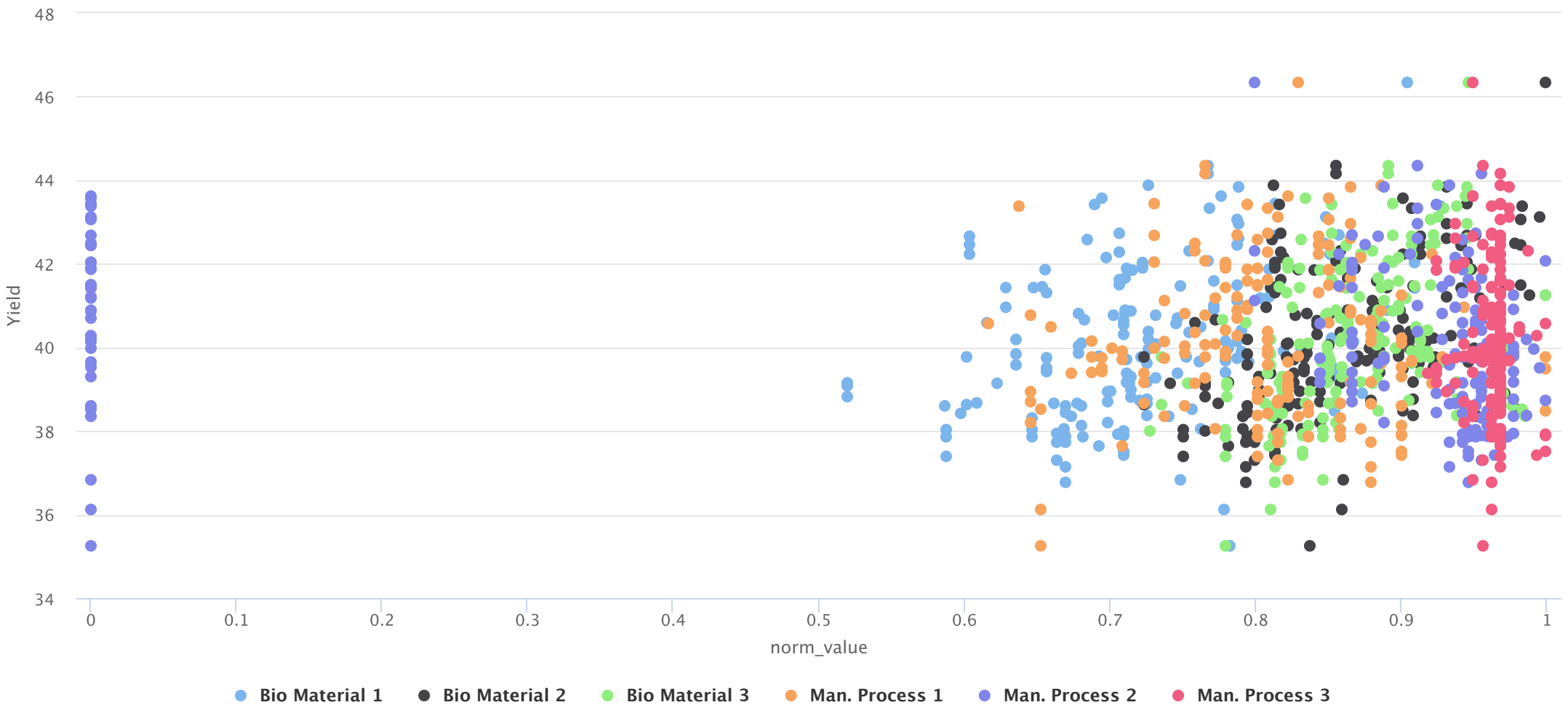
```
# Construct an interactive scatterplot.
scatter_interactive =                #<- name the
plot
  hchart(CMP_subset_long2,           #<- set data
         "scatter",                  #<- plot type
"scatter"
          hcaes(x = norm_value,      #<- set
aesthetics to map x-axis
                y = Yield,           #<- set
aesthetics to map y-axis
                group = variable)) #<- group by
```

- In R, interactive charts appear in the `Viewer` pane, right next to the `Help` tab



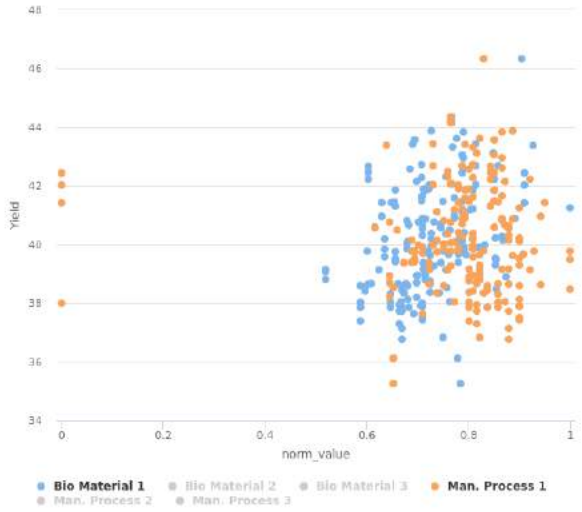
hchart: scatterplot (cont'd)

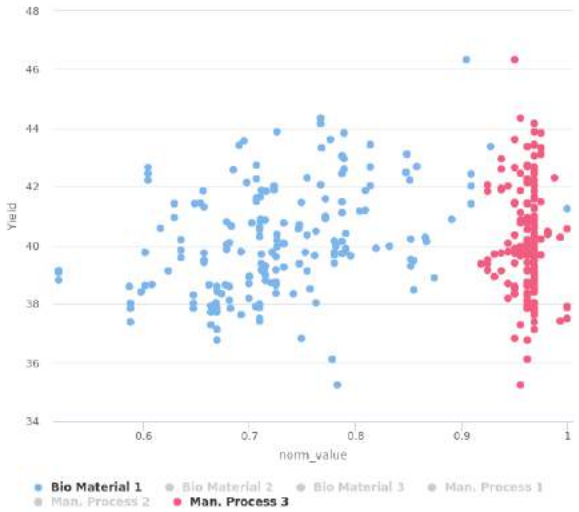
```
scatter_interactive
```



Scatterplot: selecting categories

- In `highcharts` every plotted category seen in the legend is called a `series`
- Highcharts auto-colors series when it detects more than 1 category
- We can interactively select and de-select which series to display by clicking on the series names in the legend

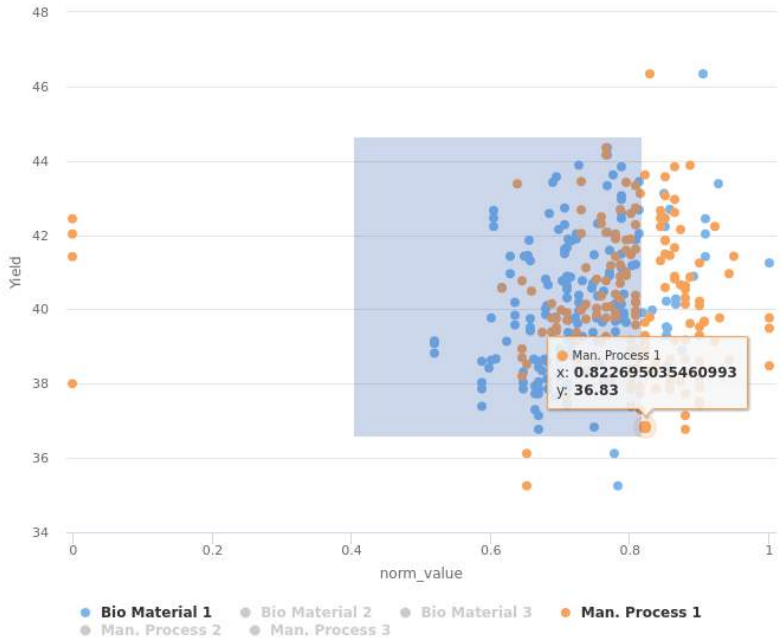




Scatterplot: `pipe` and customize

- In `highcharts` every new option or layer can be added using the already familiar `pipe` operator (as opposed to the + operator in `ggplot2`)
- The `hc_chart` function is responsible for global chart options like zoom, size, and theme
- Let's set a zoom for our plot by passing the `zoomType` argument to `hc_chart`
- `xy` zoom allows to zoom across both `x` and `y` axes

```
# Pipe chart options to original chart.
scatter_interactive = scatter_interactive %>%
  # Use chart options to specify zoom.
  hc_chart(zoomType = "xy")

scatter_interactive
```



Scatterplot: add title

A title can be added to `highcharter` plots using the `hc_title()` function

```
# Pipe chart options to original chart.
scatter_interactive = scatter_interactive %>%
 # Add title to the plot.
 hc_title(text = "CMP data: Yield vs. other variables")

scatter_interactive
```

Module completion checklist

Objective	Complete
Introduce using the highcharter package to build interactive visualizations	✔
Use highcharter with tidy data to create a scatterplot	✔
Visualize a correlation plot with hchart	
Build a column plot with hchart	
Create a boxplot with hchart	
Save interactive plots with htmlwidgets	

hchart: correlation matrix

- `hchart` recognizes the type of data being given to it
- If we pass a correlation matrix, it will create a correlation plot
- No other arguments are necessary to create a basic plot!

```
# Compute a correlation matrix for the first
# 4 variables in our data.
cor_matrix = cor(CMP_subset[, 1:4])

# Construct a correlation plot by
# simply giving the plotting function
# a correlation matrix.
correlation_interactive = hchart(cor_matrix) %>%
 # Add title to the plot.
 hc_title(text = "CMP data: correlation")
```

hchart: correlation matrix (cont'd)

```
correlation_interactive
```

Knowledge check 1

Exercise 1

Module completion checklist

Objective	Complete
Introduce using the highcharter package to build interactive visualizations	✔
Use highcharter with tidy data to create a scatterplot	✔
Visualize a correlation plot with hchart	✔
Build a column plot with hchart	
Create a boxplot with hchart	
Save interactive plots with htmlwidgets	

hchart: column plot

Let's now create an interactive plot to visualize the summary of our data

- To do that, we first need to get the summary statistics and save them as a dataframe

```
# Create data summary.
CMP_summary = summary(CMP_subset)

# Save it as a data frame.
CMP_summary = as.data.frame(CMP_summary)

# Inspect the data.
head(CMP_summary)
```

```
  Var1      Var2            Freq
1          Yield Min.   :35.25
2          Yield 1st Qu.:38.75
3          Yield Median :39.97
4          Yield Mean   :40.18
5          Yield 3rd Qu.:41.48
6          Yield Max.   :46.34
```

```
# Remove an empty variable.
CMP_summary$Var1 = NULL

# Rename remaining columns.
colnames(CMP_summary) = c("Variable",
                          "Summary")
# Inspect updated data.
head(CMP_summary)
```

```
   Variable         Summary
1     Yield Min.   :35.25
2     Yield 1st Qu.:38.75
3     Yield Median :39.97
4     Yield Mean   :40.18
5     Yield 3rd Qu.:41.48
6     Yield Max.   :46.34
```

Column plot: prepare data

- Let's separate the summary values from the summary statistics
- We can utilize the `separate` function in `tidyr`

```
# Separate `Summary` column into 2 columns.
CMP_summary = CMP_summary %>%                #<- set original data
  separate(Summary,                          #<- separate `Summary` variable
           into = c("Statistic", "Value"),   #<- into 2 columns: `Statistic`, `Value`
           sep = ":",                        #<- set separating character
           convert = TRUE)                   #<- where applicable convert data (to numeric)

# Inspect the first few entries in data.
head(CMP_summary)
```

```
   Variable Statistic Value
1     Yield   Min.    35.25
2     Yield   1st Qu. 38.75
3     Yield   Median  39.97
4     Yield   Mean    40.18
5     Yield   3rd Qu. 41.48
6     Yield   Max.    46.34
```

```
# Inspect total number of rows in data including NAs.
nrow(CMP_summary)
```

```
[1] 49
```

Column plot: create plot

```
# Inspect `Value` column for `NAs`.
which(is.na(CMP_summary$Value) == TRUE)
```

```
[1]  7 14 21 28
```

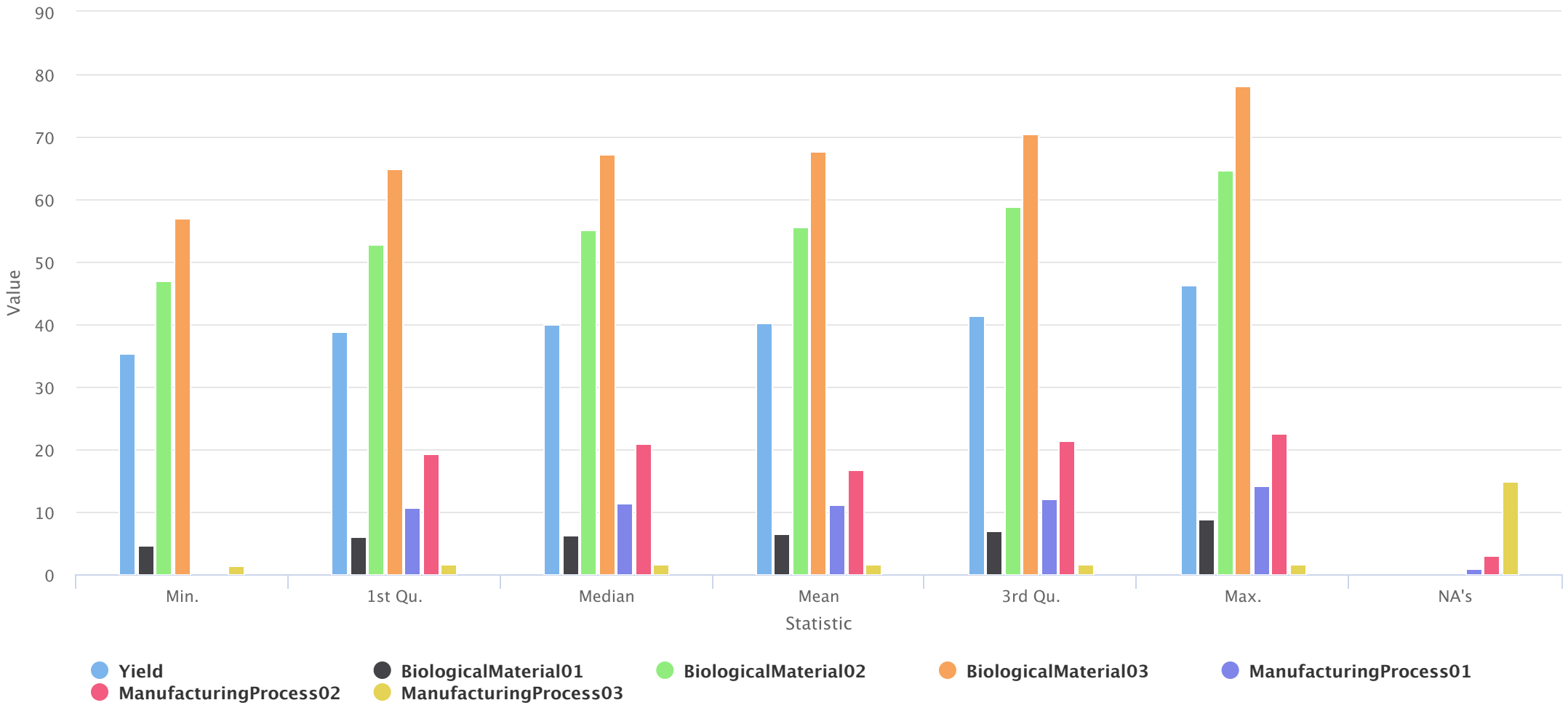
```
# Subset only rows where `Value` is not NAs.
CMP_summary = subset(CMP_summary, !is.na(Value))

# Now the number of rows should be 4 less.
nrow(CMP_summary)
```

```
[1] 45
```

```
# Construct the summary chart.
CMP_summary_interactive =
  hchart(CMP_summary,              #<- set data
         "column",                 #<- set type (`column` in highcharts)
         hcaes(x = Statistic,      #<- arrange `Statistics` across x-axis
               y = Value,          #<- map `Value` of each `Statistic` to y-axis
               group = Variable)) #<- group columns by `Variable`
```

Column plot: display plot

```
CMP_summary_interactive
```



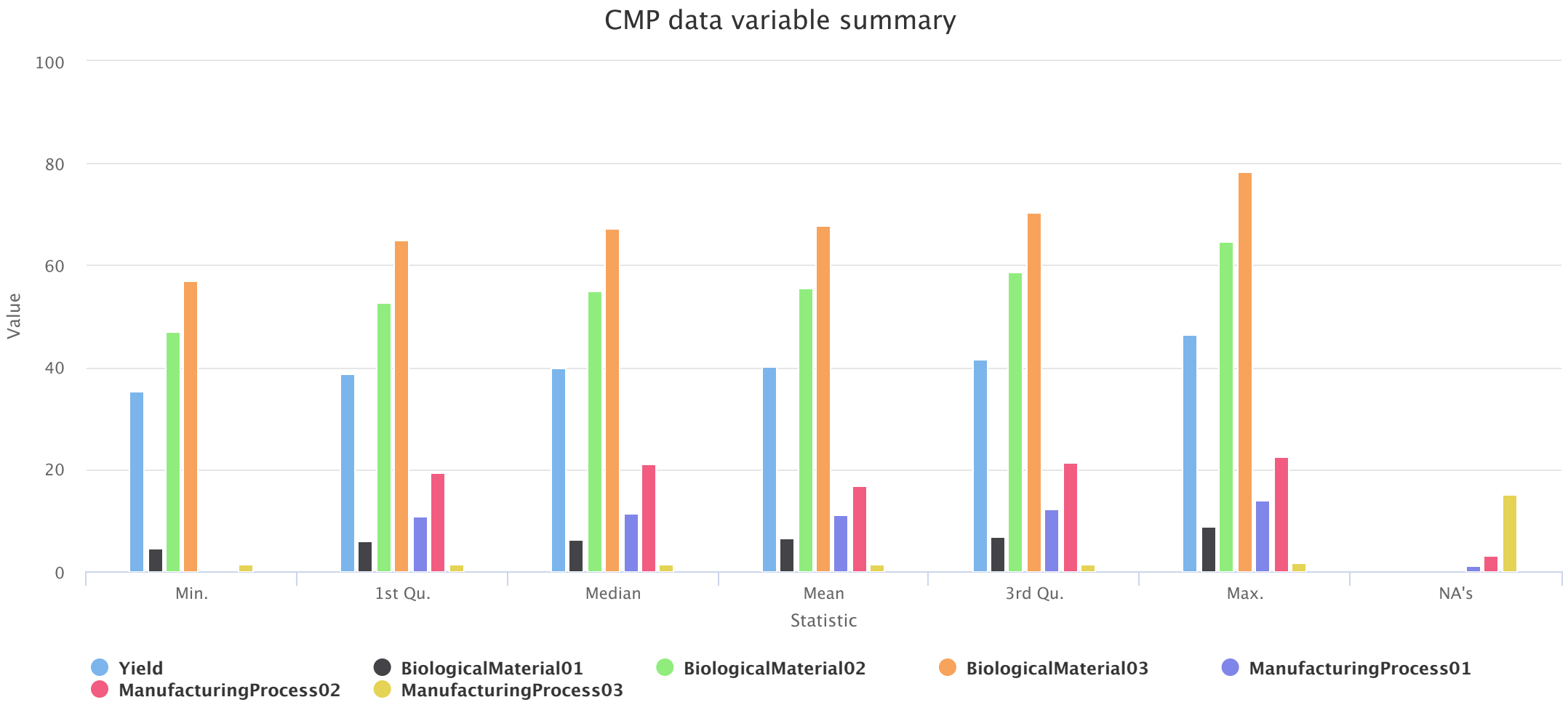
Column plot: customize tooltip

- Since we are comparing variables' summary `statistics`, it would be convenient for the `tooltip` to contain information about the `group` rather than the individual columns within the `group`
- We can control different `tooltip` options of the chart using the `hc_tooltip` option
- The `shared` option is often used to share a tooltip between members of a `group`

```
# Adjust tooltip options by piping `hc_tooltip` to base plot.
CMP_summary_interactive = CMP_summary_interactive %>%
  hc_tooltip(shared = TRUE)  %>%                #<- `shared` needs to be set to `TRUE`
  hc_title(text = "CMP data variable summary") #<- add title to your plot
```

Column plot: customize tooltip (cont'd)

```
CMP_summary_interactive
```



Module completion checklist

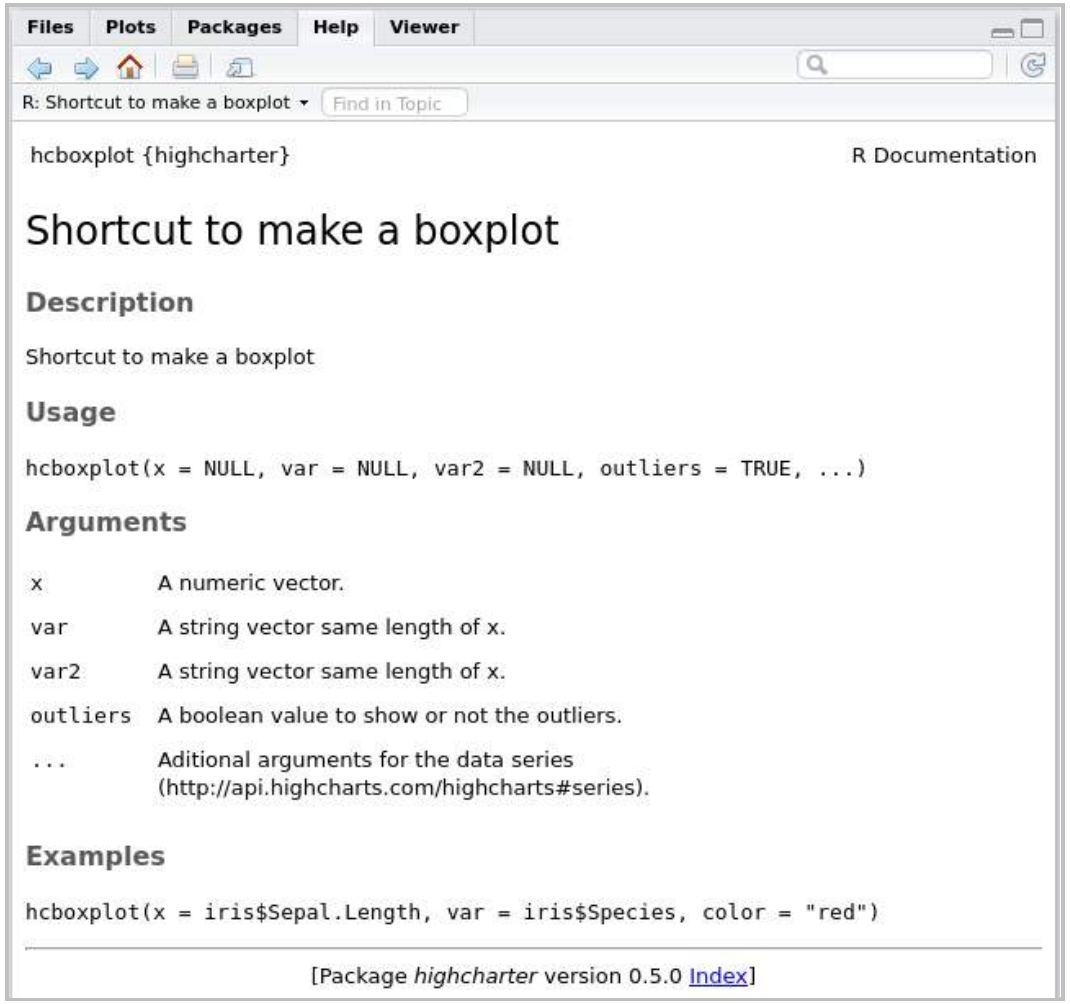
Objective	Complete
Introduce using the highcharter package to build interactive visualizations	✔
Use highcharter with tidy data to create a scatterplot	✔
Visualize a correlation plot with hchart	✔
Build a column plot with hchart	✔
Create a boxplot with hchart	
Save interactive plots with htmlwidgets	

Highcharts boxplot: hcboxplot

`hcboxplot` allows us to create an interactive boxplot. It needs two arguments:

- `x` requires the numeric data to be plotted along the x-axis (`boxplot` in `highcharts` is horizontal by default)
- `var` requires categorical data to be plotted along the y-axis

```
?hcboxplot

hcboxplot(x = Numeric_data_vector,
          var = Categorical_data_vector,
          ...)
```



Files | Plots | Packages | Help | Viewer

R: Shortcut to make a boxplot | Find in Topic

hcboxplot {highcharter} R Documentation

Shortcut to make a boxplot

Description

Shortcut to make a boxplot

Usage

```
hcboxplot(x = NULL, var = NULL, var2 = NULL, outliers = TRUE, ...)
```

Arguments

x	A numeric vector.
var	A string vector same length of x.
var2	A string vector same length of x.
outliers	A boolean value to show or not the outliers.
...	Aditional arguments for the data series (http://api.highcharts.com/highcharts#series).

Examples

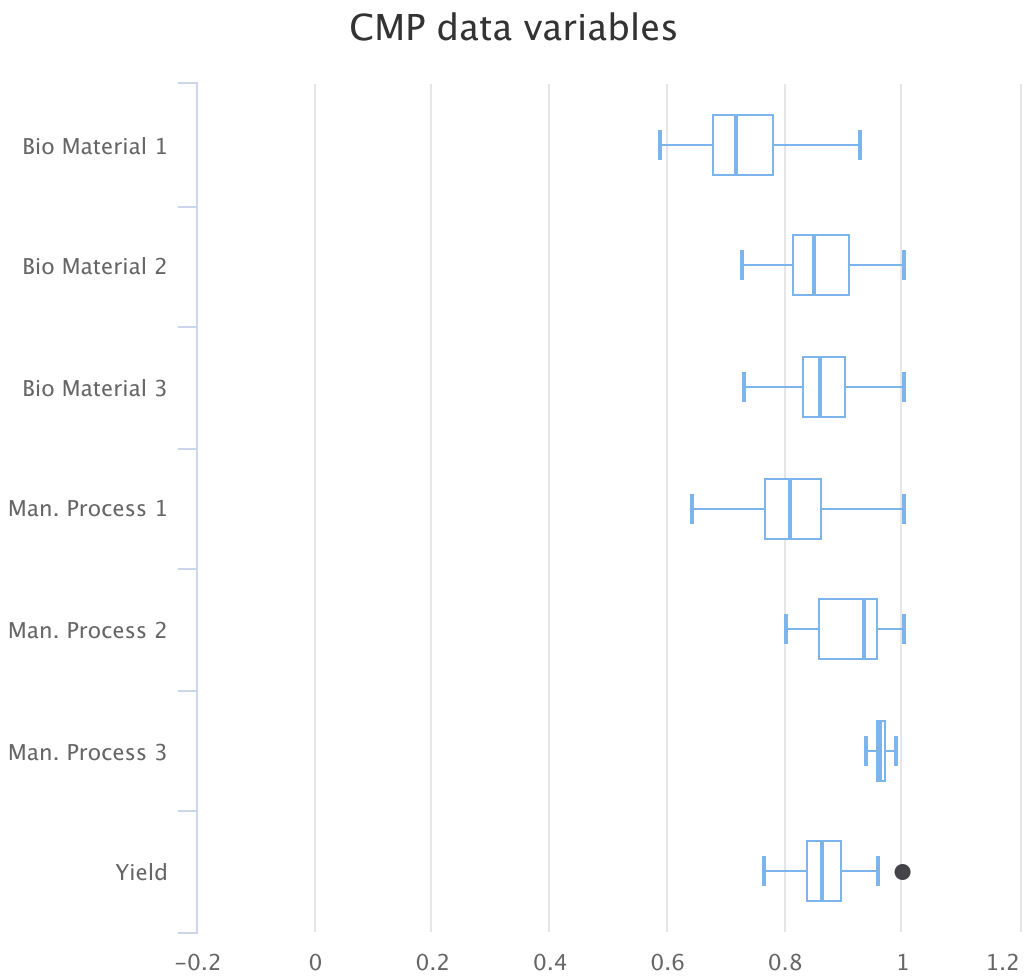
```
hcboxplot(x = iris$Sepal.Length, var = iris$Species, color = "red")
```

[Package *highcharter* version 0.5.0 Index]

Highcharts boxplot: hcboxplot (cont'd)

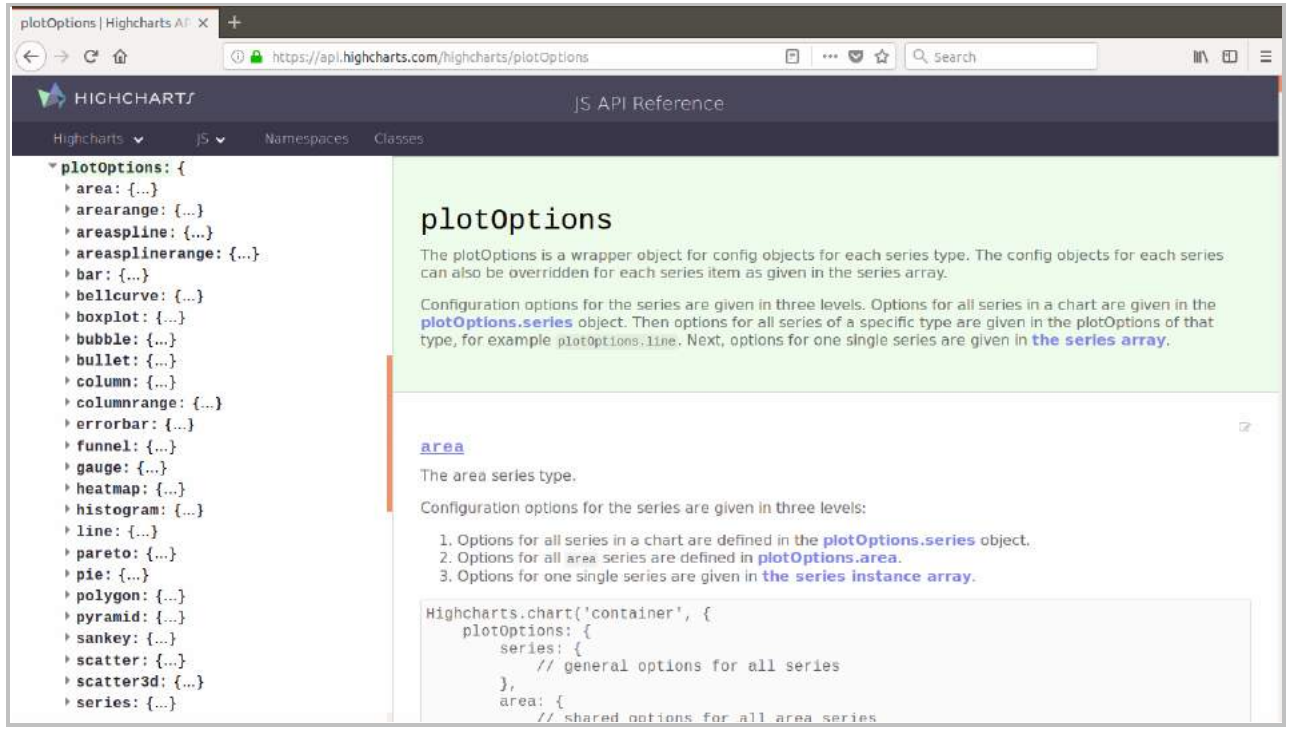
```
# Construct an interactive boxplot.
boxplot_interactive =
  hcboxplot(x = CMP_subset_long$norm_value,
            var = CMP_subset_long$variable,
            name = "Normalized value") %>%
  hc_title(text = "CMP data variables")
```

```
boxplot_interactive
```



Highcharts plotOptions

- To control individual layer/`series` options for various plot types, we use the `hc_plotOptions` function
- It can be **piped** to the original chart to enhance our base plot
- **Each series type** represented in the chart **can be given a unique set of options**
- **Every set of options is a list** of micro-level adjustments



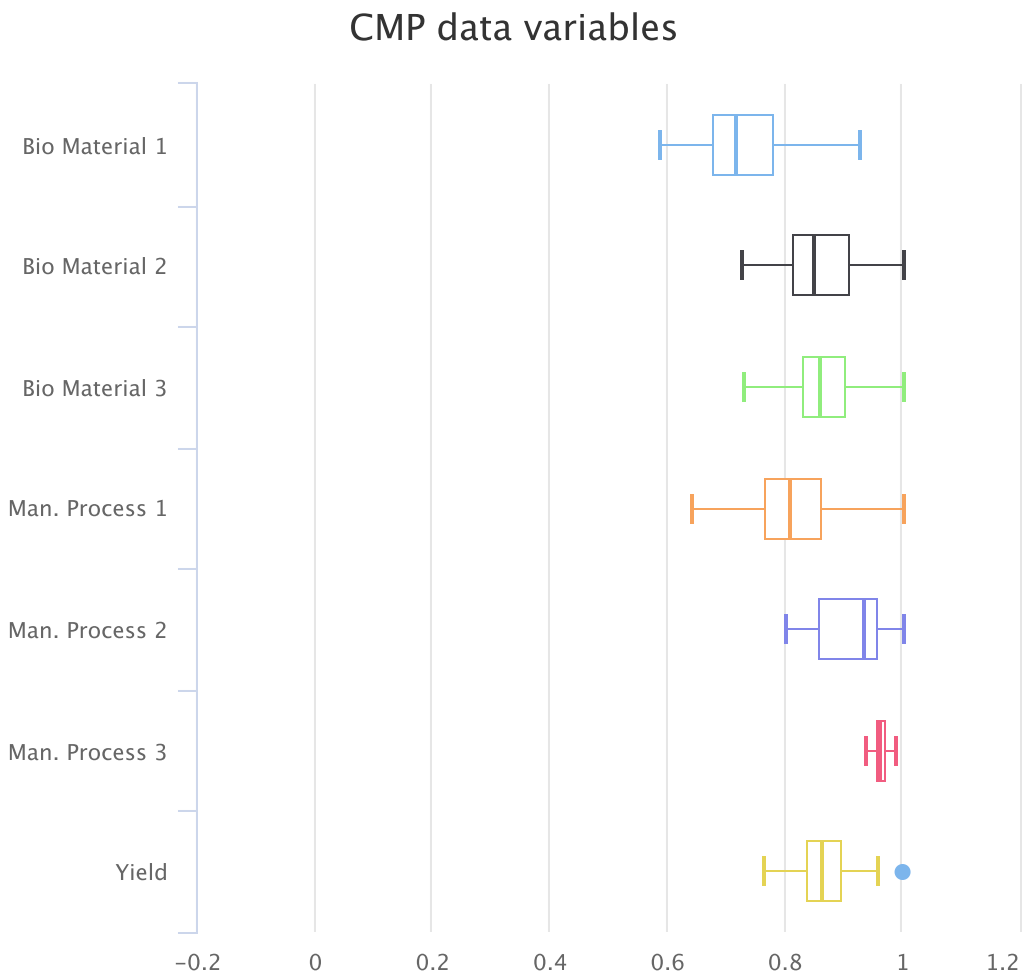
- The full list of plot options can be found in `Highcharts API` documentation ***https://api.highcharts.com/highcharts/plotOptions***

hcboxplot: customize with `hc_plotOptions`

Now we will use the `hc_plotOptions()` function to customize color for our boxplot

```
# Enhance original boxplot with color options.
boxplot_interactive = boxplot_interactive %>%
  hc_plotOptions(   #<- plot options
    boxplot = list(     #<- for boxplot
      colorByPoint = TRUE)) #<- color each box
```

```
boxplot_interactive
```



Module completion checklist

Objective	Complete
Introduce using the highcharter package to build interactive visualizations	✔
Use highcharter with tidy data to create a scatterplot	✔
Visualize a correlation plot with hchart	✔
Build a column plot with hchart	✔
Create a boxplot with hchart	✔
Save interactive plots with htmlwidgets	

Save interactive plots: htmlwidgets

- The `htmlwidgets` package lets us use JavaScript visualization libraries in R console
- We can embed widgets in R markdown and Shiny web applications
- Explore `htmlwidgets` ***here***

```
# Install `htmlwidgets` package.
install.packages("htmlwidgets")

# Load the library.
library(htmlwidgets)

# View documentation.
library(help = "htmlwidgets")
```

Documentation for package 'html...

```
                Information on package 'htmlwidgets'

Description:

Package:            htmlwidgets
Type:               Package
Title:              HTML Widgets for R
Version:            0.9
Date:               2017-07-10
Authors@R:          c( person("Ramnath", "Vaidyanathan", role = c("aut", "cph")), person("Yihui
                    c("aut")), person("JJ", "Allaire", role = c("aut", "cre"), email = "jj@rstu
                    person("Joe", "Cheng", role = c("aut")), person("Kenton", "Russell", role =
                    person(family = "RStudio", role = "cph") )
Description:        A framework for creating HTML widgets that render in various contexts inclu
                    Markdown' documents, and 'Shiny' web applications.
License:            MIT + file LICENSE
VignetteBuilder:    knitr
Imports:            htmltools (>= 0.3), jsonlite (>= 0.9.16), yaml
Suggests:           knitr (>= 1.8)
Enhances:           shiny (>= 0.12)
URL:                https://github.com/ramnathv/htmlwidgets
BugReports:         https://github.com/ramnathv/htmlwidgets/issues
RoxygenNote:        6.0.1
NeedsCompilation:   no
Packaged:           2017-07-10 10:49:02 UTC; jjallaire
Author:             Ramnath Vaidyanathan [aut, cph], Yihui Xie [aut], JJ Allaire [aut, cre], Jo
                    Russell [aut, cph], RStudio [cph]
```

Save interactive plots: htmlwidgets (cont'd)

```
# Set working directory to where you save plots.
setwd(plot_dir)

# Save desired interactive plot to an HTML file.
saveWidget(scatter_interactive,        #<- plot object to save
           "interactive_scatterplot.html", #<- name of file to where the plot is to be saved
           selfcontained = TRUE)              #<- set `selfcontained` to TRUE, so that
                                              #   all necessary files and scripts are embedded
                                              #   into the HTML file itself
```

Knowledge check 2

Exercise 2

Module completion checklist

Objective	Complete
Introduce using the highcharter package to build interactive visualizations	✔
Use highcharter with tidy data to create a scatterplot	✔
Visualize a correlation plot with hchart	✔
Build a column plot with hchart	✔
Create a boxplot with hchart	✔
Save interactive plots with htmlwidgets	✔

Summary

In this module, we:

- Outlined the `highcharter` package for creating interactive visualizations
- Built an interactive scatterplot, column plot, and boxplot using the `hchart()` function of `highcharter`
- Customized plots using various `highchart` options and parameters
- Learned to save interactive plots using `htmlwidgets` so that they can be embedded in R markdown and R shiny applications

In the next module, we will:

- Introduce how to build an interactive map with a json file
- Discuss best practices for highcharter

This completes our module

Congratulations!