1. Getting Started
3. Basic Graphs
4. Basic customization
5. Changing themes & Geoms
6. Multiple graphs
facets =
7. Save output

R Crash Course Part 2 – *qplot()*

Rob Colautti

1. Getting Started

Install the `ggplot2` package

```
library(ggplot2)
source("http://bit.ly/theme_pub")
theme_set(theme_pub())
```

The *ggplot2* package includes two main functions, quickplot `qplot()` for fast graphs and the `ggplot()` function for more detailed, customizable graphs.

The package was created by the same group that also made:

RStudio, *RMarkdown*, *Shiny*, *ggvis*

and many other cool packages. These are available on the RStudio website:

https://www.rstudio.com/products/rpackages/ (https://www.rstudio.com/products/rpackages/)

Full ggplot2 Documentation

http://docs.ggplot2.org/current/ (http://docs.ggplot2.org/current/)

This tutorial focuses on ***qplot()***, a quick and simple yet versatile plotting function. It is much more intuitive than R's default graphics commands.

For more advanced graphics, see the ggplot tutorial (./3_ggplot.html)

General format:

```
qplot(x=my.xVariable, y=my.yVariable, data=my.data.frame)
```

ppt slides

1. Graphical concepts
2. ggplot grammar
3. Explanation of selection meta-analysis data
4. anatomy of a graph

Slides (Graphics_small.pdf)

Data setup

We will again be working with the FallopiaData.csv dataset, which can be downloaded here (FallopiaData.csv), and saved to your project folder to follow along.

```
MyData<-read.csv("FallopiaData.csv",header=T)
```

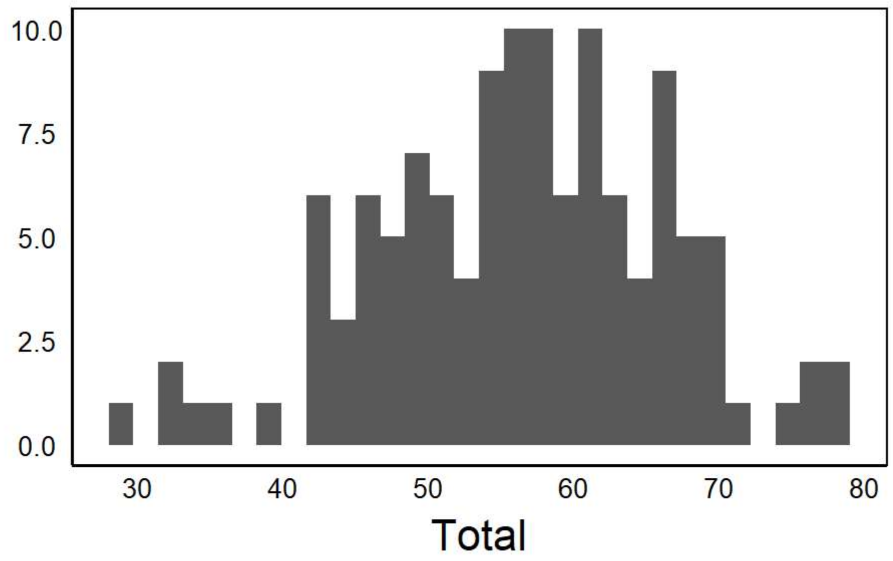
3. Basic Graphs

One continuous

Produces a histogram by default

```
qplot(x=Total,data=MyData)
```

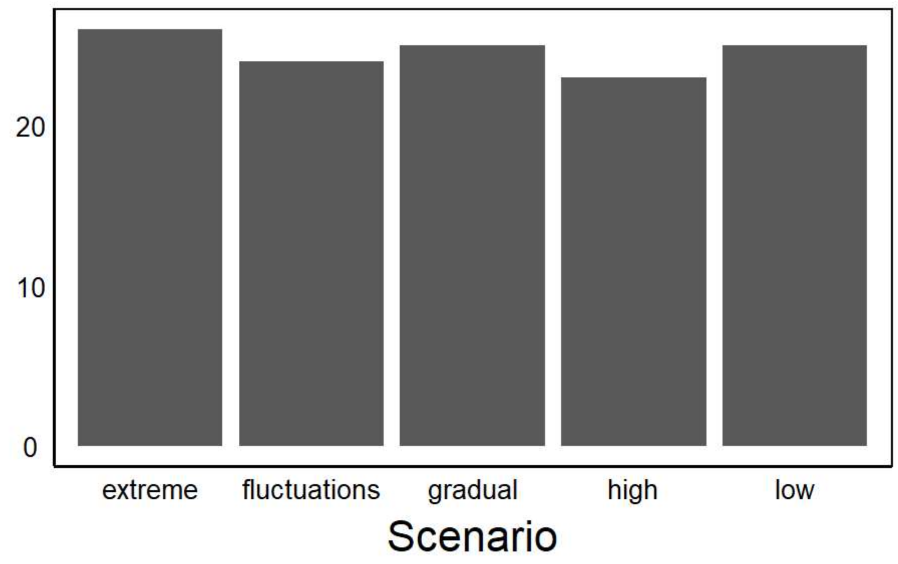
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



One categorical

Bar graph of counts for each category

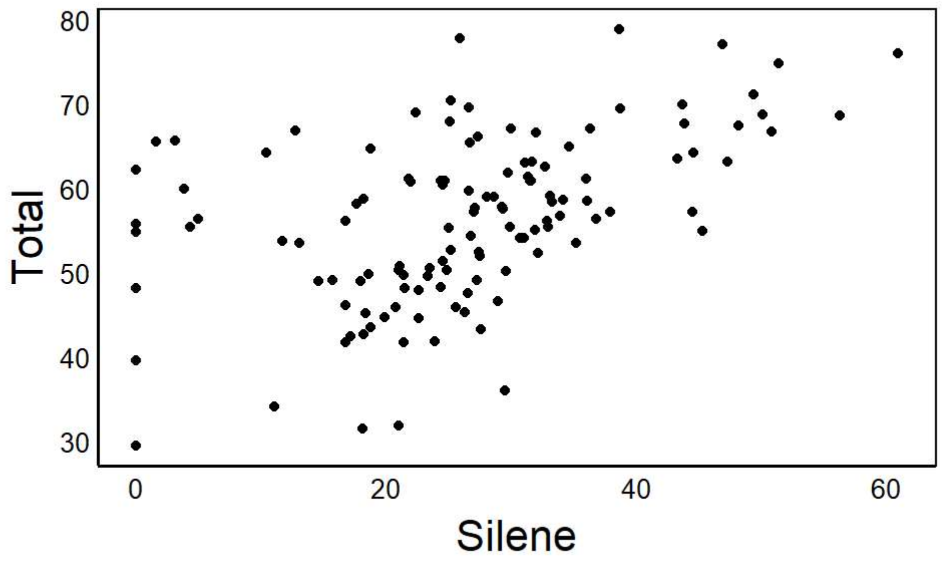
```
qplot(x=Scenario,data=MyData)
```



Two continuous

Bivariate plot

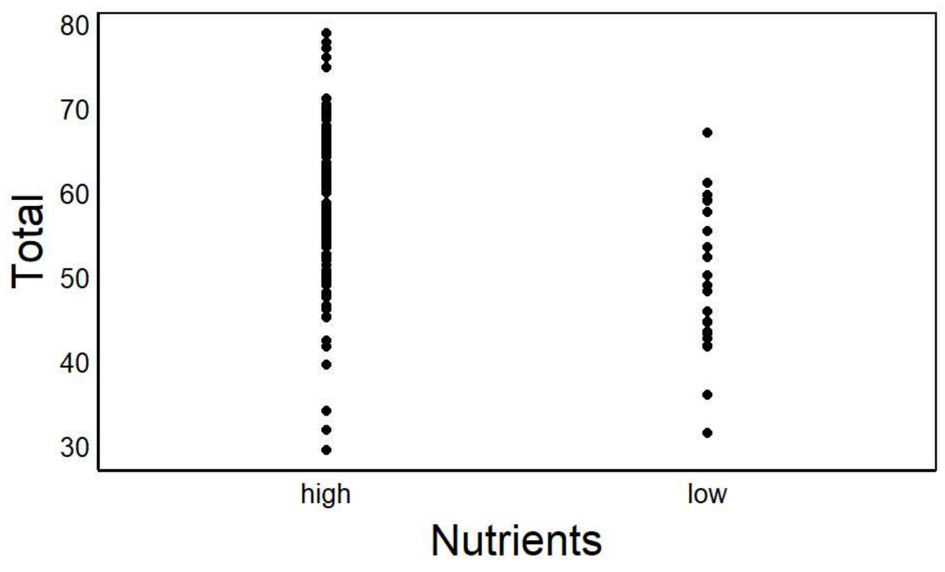
```
qplot(x=Silene,y=Total,data=MyData)
```



Categorical by continuous

Categorical scatter plot

```
qplot(x=Nutrients,y=Total,data=MyData)
```

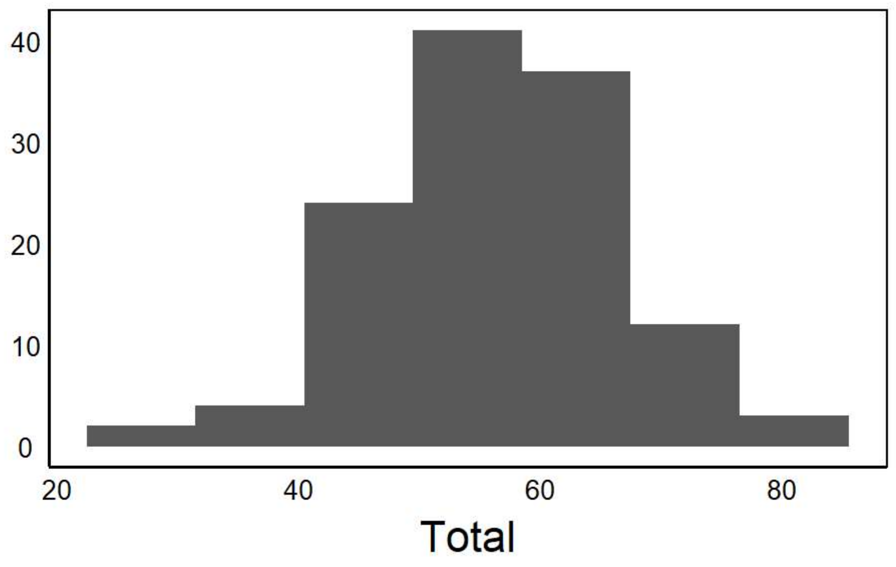


4. Basic customization

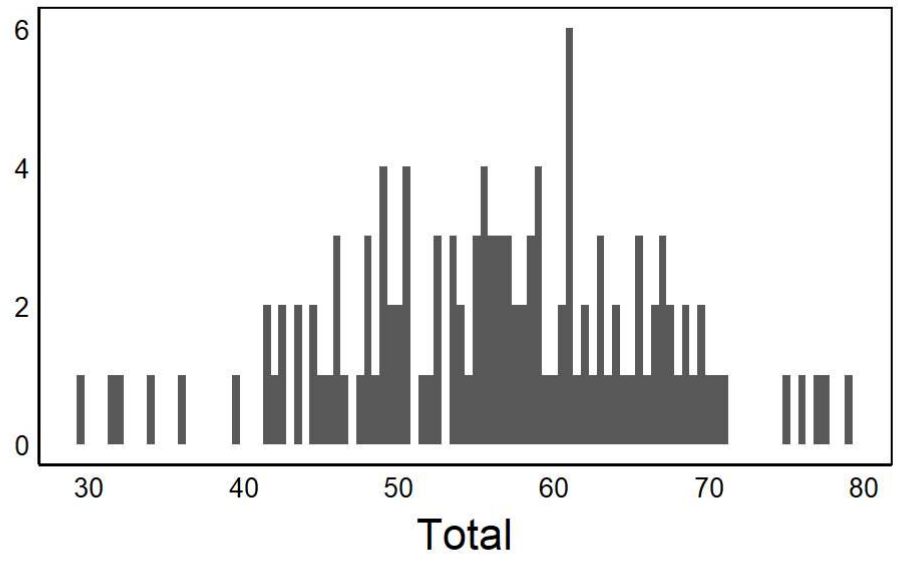
binwidth =

Change bin width of a histogram

```
qplot(x=Total,data=MyData,binwidth=9)
```



```
qplot(x=Total,data=MyData,binwidth=0.5)
```

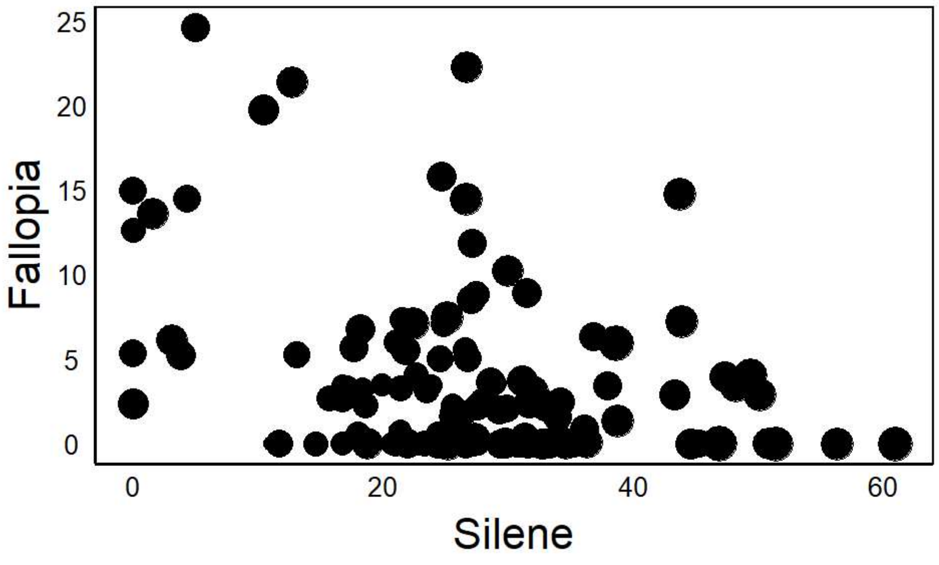


size =

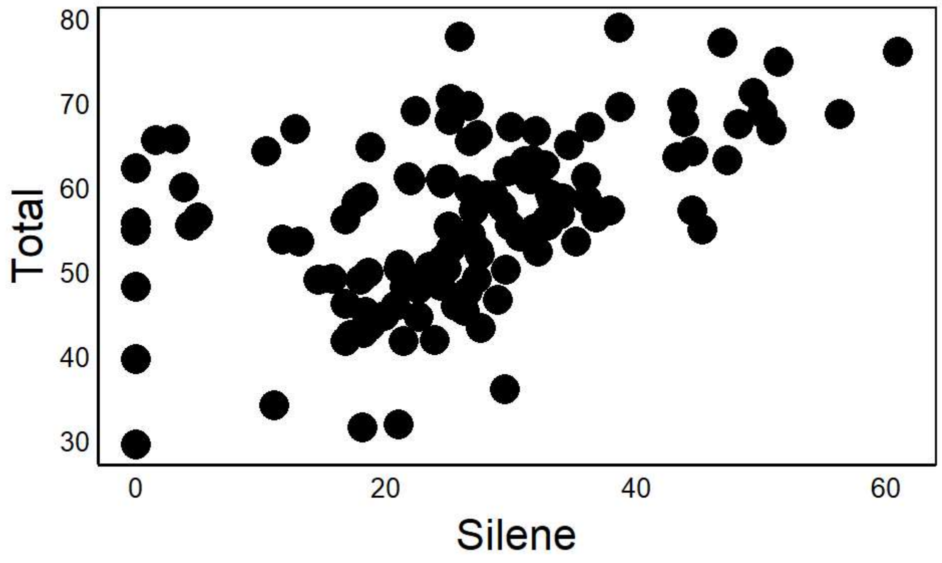
Controls size of points, lines, etc.

Google “pch shapes in R” http://vis.supstat.com/2013/04/plotting-symbols-and-color-palettes/ (http://vis.supstat.com/2013/04/plotting-symbols-and-color-palettes/)

```
qplot(x=Silene,y=Fallopia,data=MyData,size=Total) # Scale by a variable
```



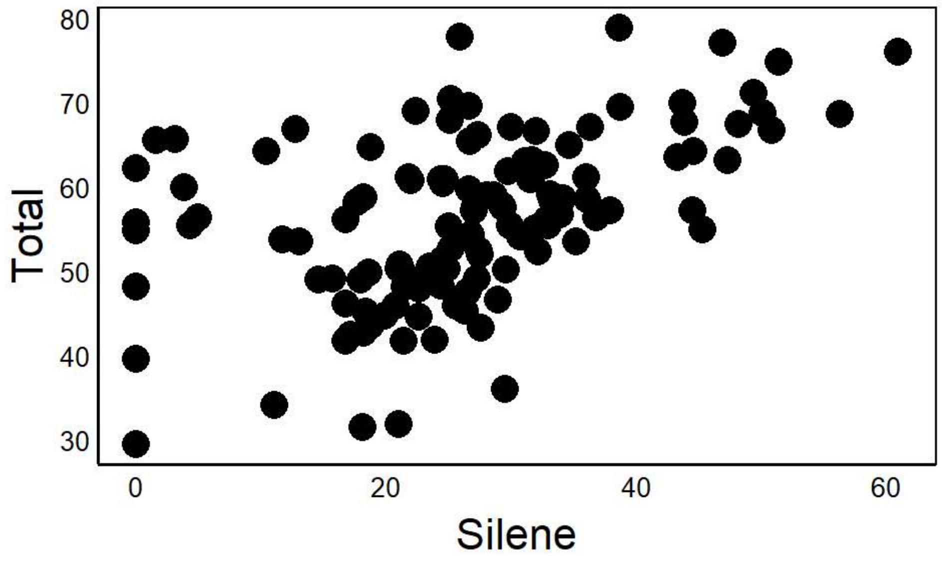
```
qplot(x=Silene,y=Total,data=MyData,size=I(5)) # Scale by a constant
```



NOTE: use of identity function: ***I()*** for constant

Compare to:

```
qplot(x=Silene,y=Total,data=MyData,size=5)
```



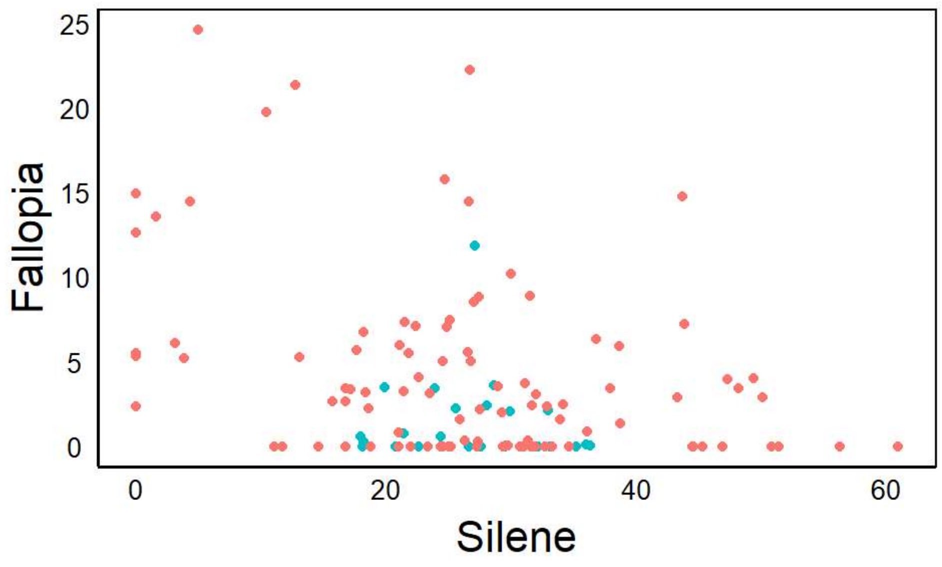
Without I(), 5 is interpreted as a factor

colour = or *color* =

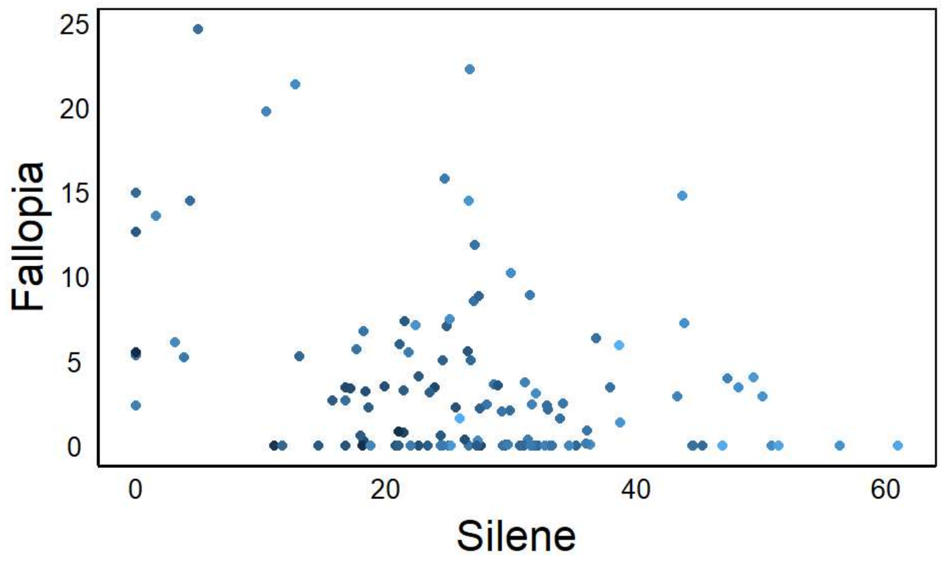
Colours points based on a factor or

Again, note use of ***I()*** for constants vs variables

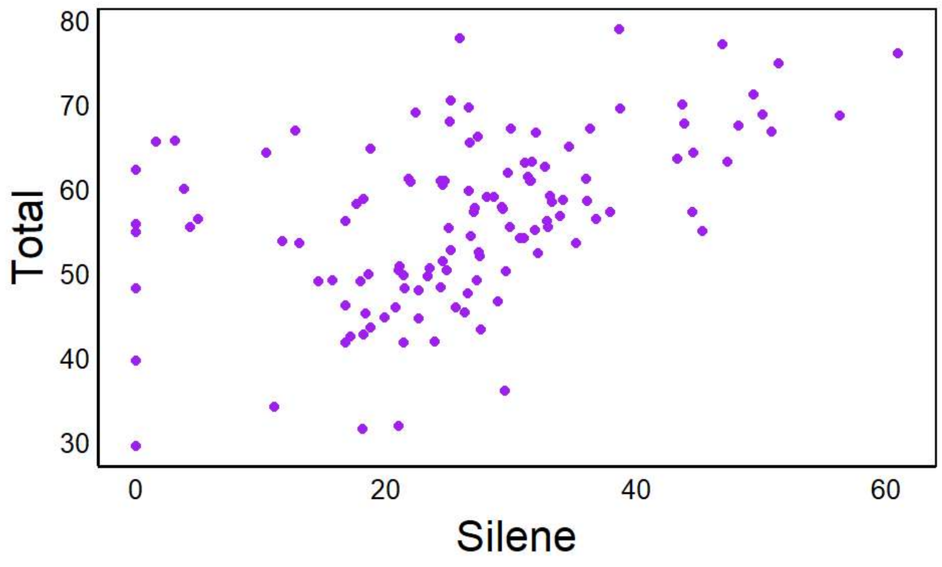
```
qplot(x=Silene,y=Fallopia,data=MyData,colour=Nutrients) # Categorical colour
```



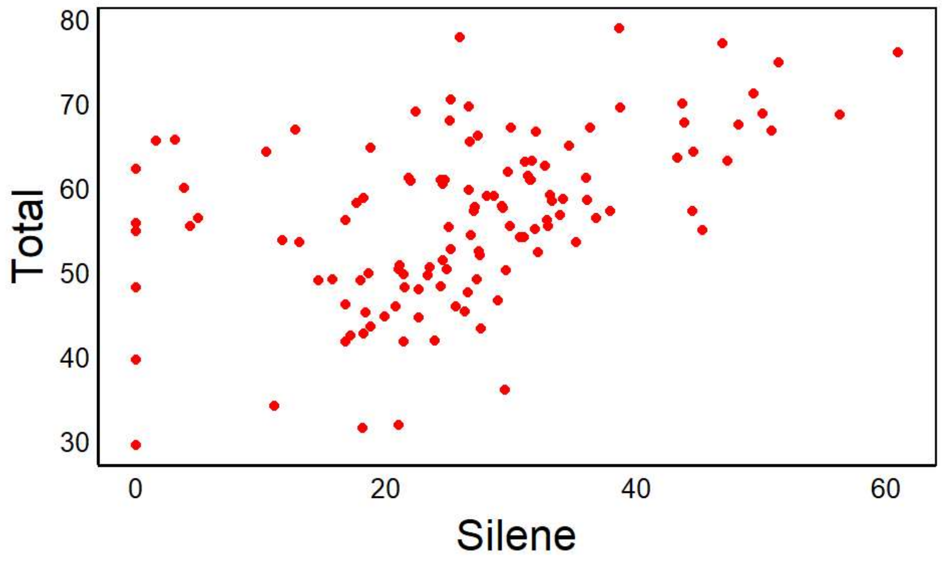
```
qplot(x=Silene,y=Fallopia,data=MyData,colour=Total) # Continuous colour
```



```
qplot(x=Silene,y=Total,data=MyData,colour=I("purple")) # basic colour
```



```
qplot(x=Silene,y=Total,data=MyData,colour=I(rgb(1,0,0))) # rgb = red/green/blue
```

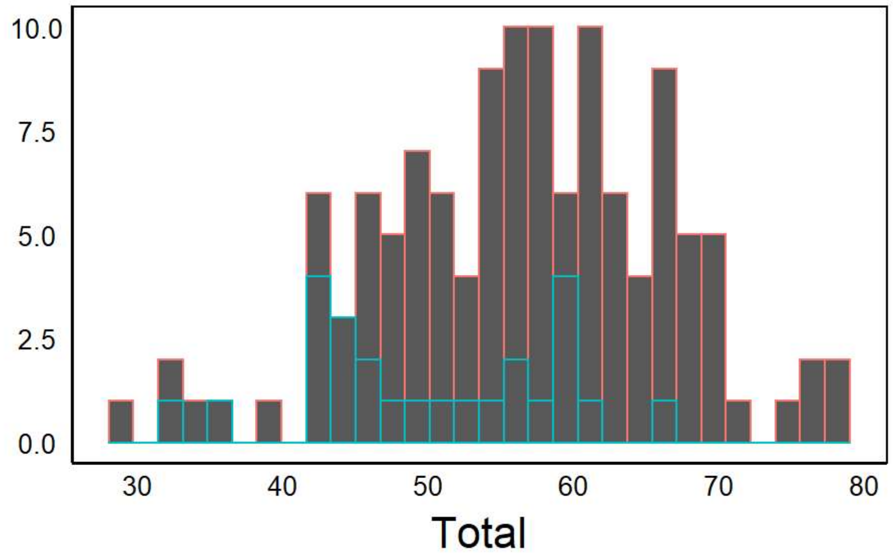


Note for histograms:

Colour applies to outlines, not so helpful

```
qplot(x=Total,data=MyData,group=Nutrients,colour=Nutrients)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

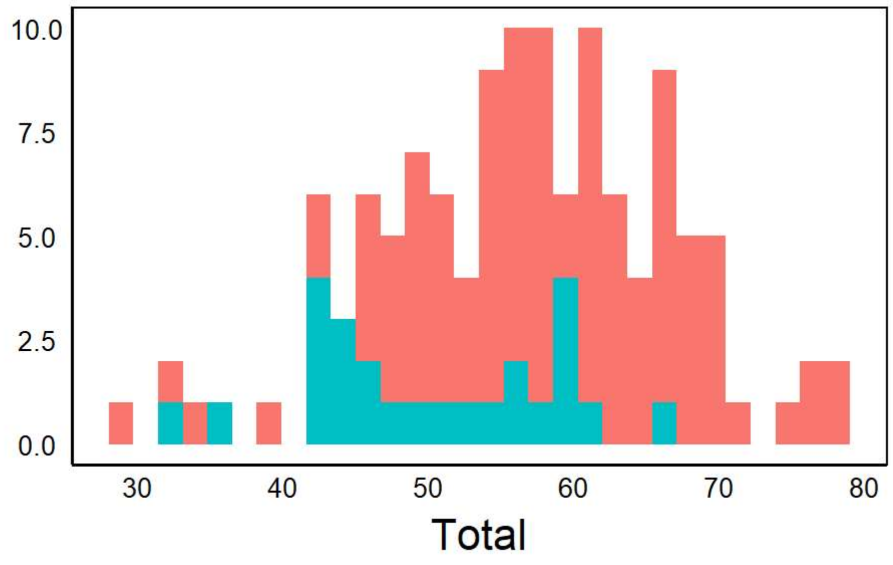


fill =

More useful for histograms

```
qplot(x=Total,data=MyData,group=Nutrients,fill=Nutrients)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

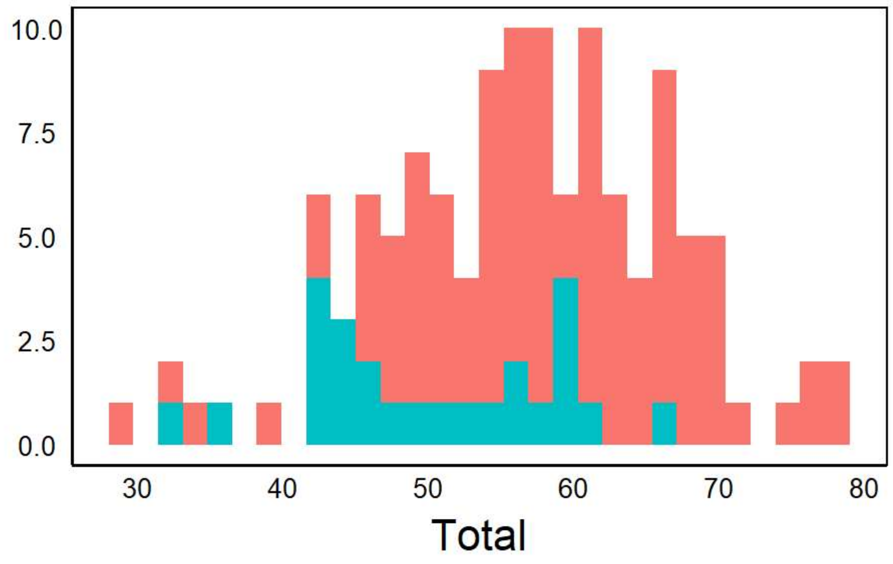


posit =

Avoid stacking histogram bars

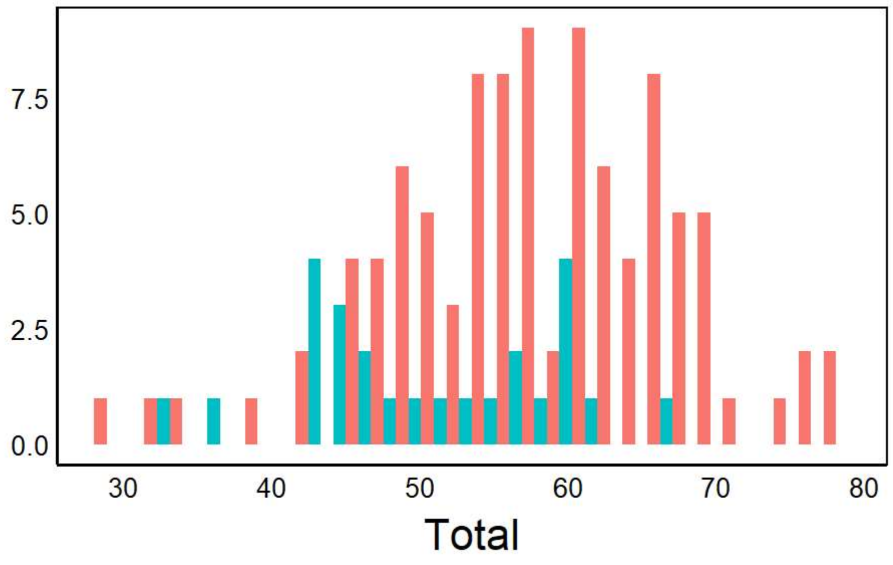
```
qplot(x=Total,data=MyData,group=Nutrients,fill=Nutrients)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
qplot(x=Total,data=MyData,group=Nutrients,fill=Nutrients,posit="dodge")
```

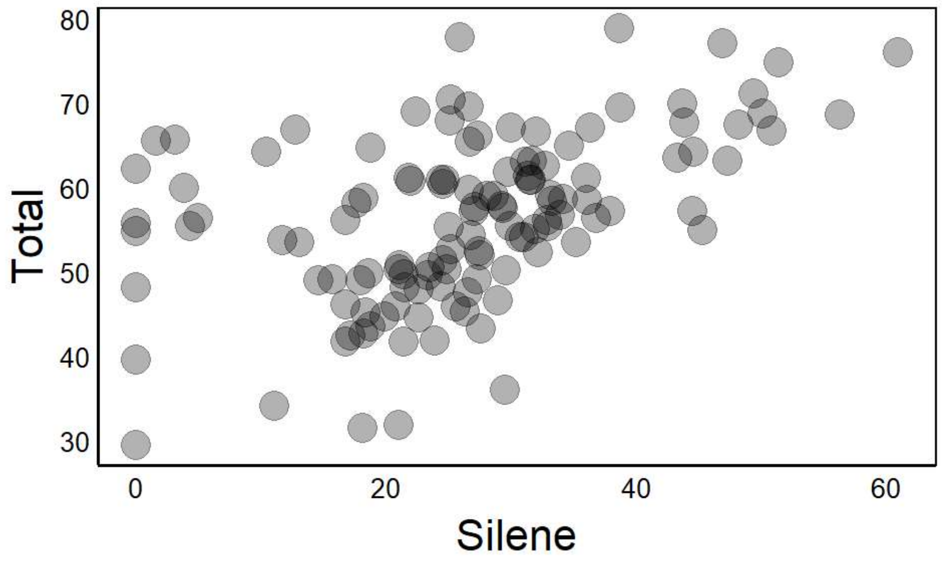
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



alpha =

Adjust transparency for overlapping points

```
qplot(x=Silene,y=Total,data=MyData,size=I(5),alpha=I(0.3))
```

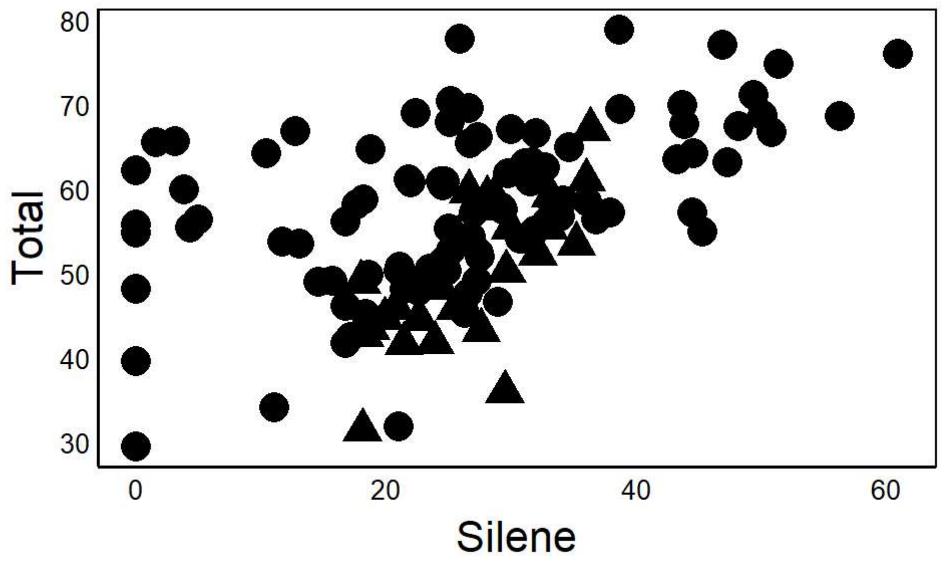


shape =

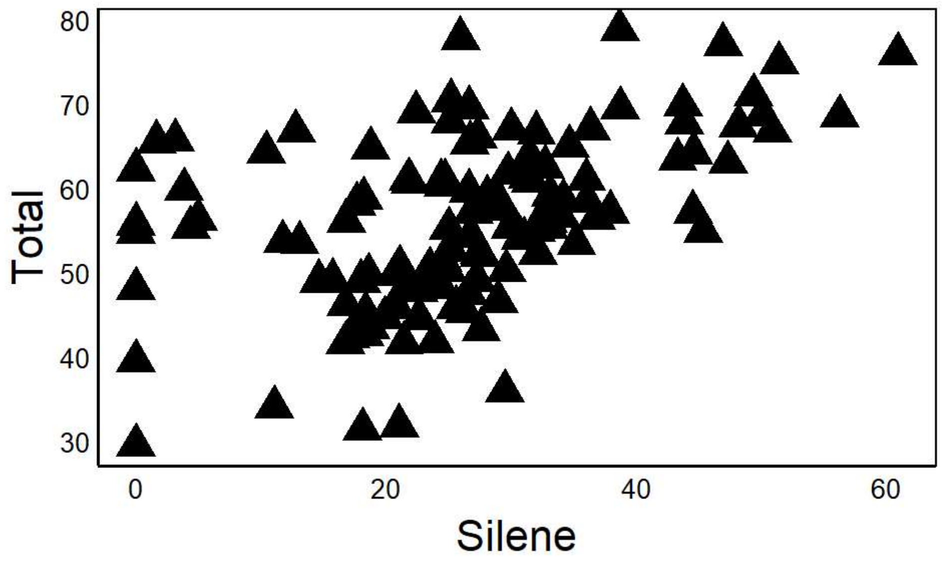
Use different shapes

Most common shape codes: 0-25

```
qplot(x=Silene,y=Total,data=MyData,size=I(5),shape=Nutrients)
```



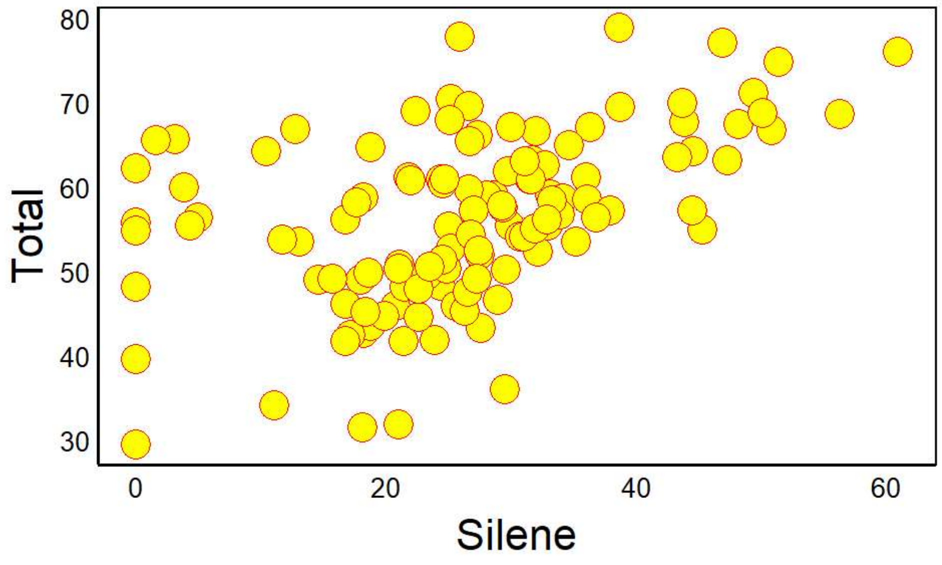
```
qplot(x=Silene,y=Total,data=MyData,size=I(5),shape=I(17))
```



Some shape types have outlines

You can use ***fill =*** and ***colour =*** to customize

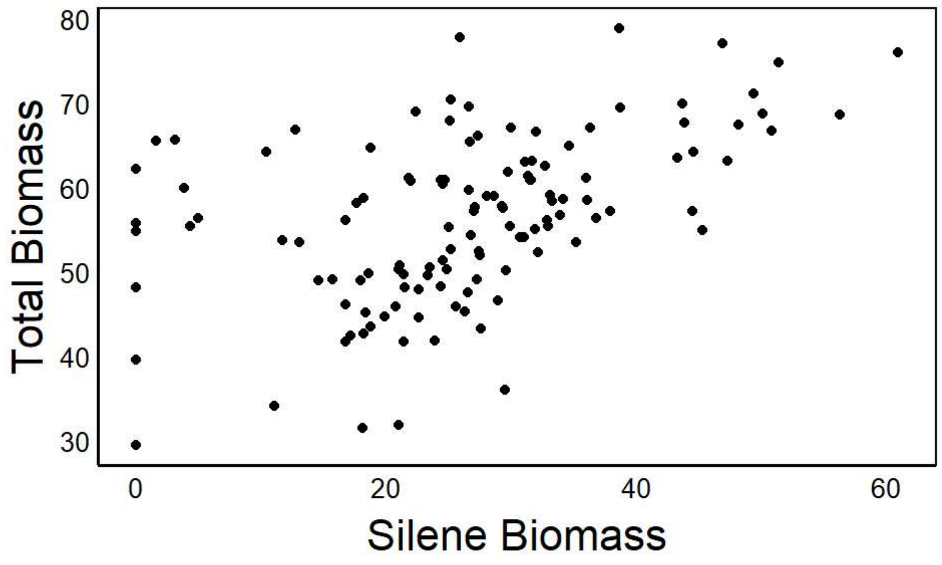
```
qplot(x=Silene,y=Total,data=MyData,size=I(5),shape=I(21),fill=I("yellow"),colour=I("re
        d"))
```



xlab = and *ylab* =

Custom axis labels

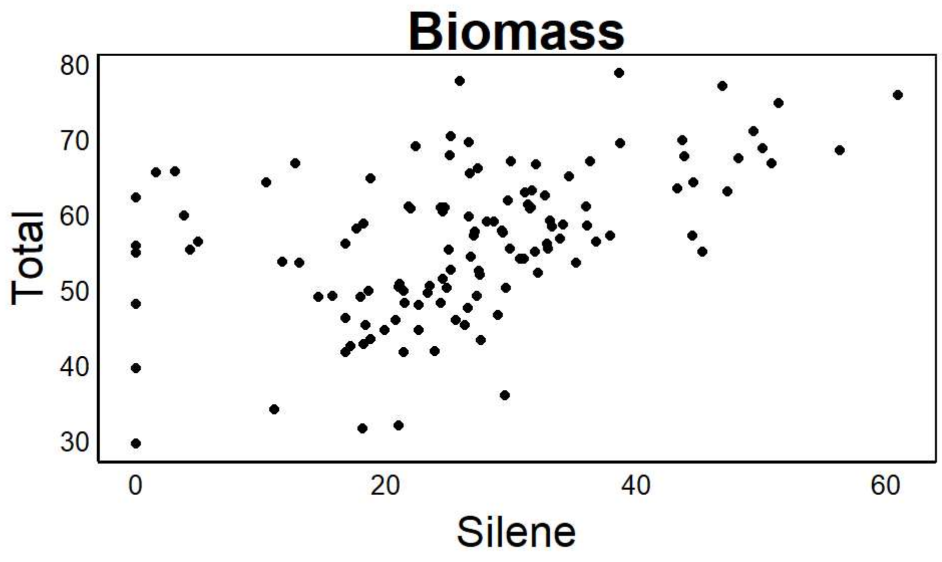
```
qplot(x=Silene,y=Total,data=MyData,xlab="Silene Biomass",ylab="Total Biomass")
```



main =

Add a title

```
qplot(x=Silene,y=Total,data=MyData,main="Biomass")
```



5. Changing themes & Geoms

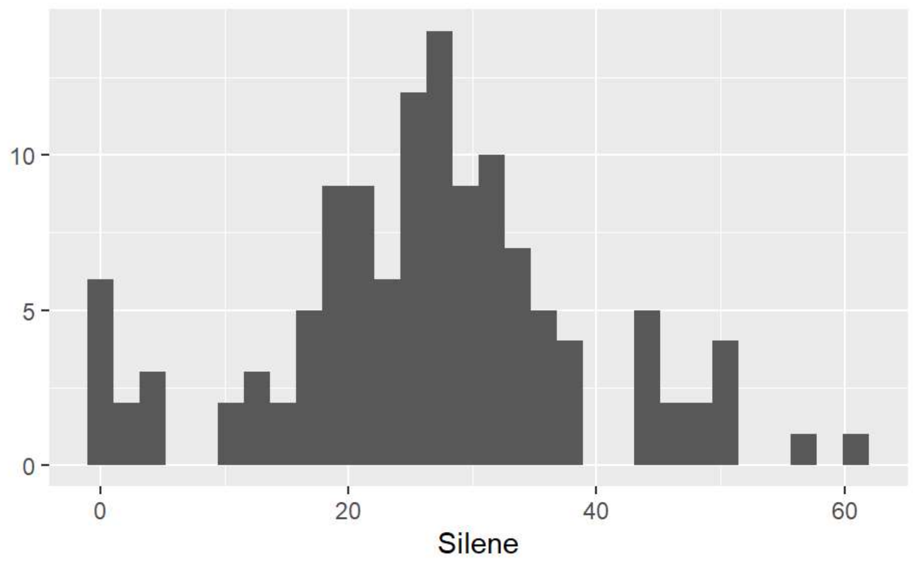
+ theme_NAME()

Modify basic format and appearance

These can be completely customized, but there are several pre-cut options:

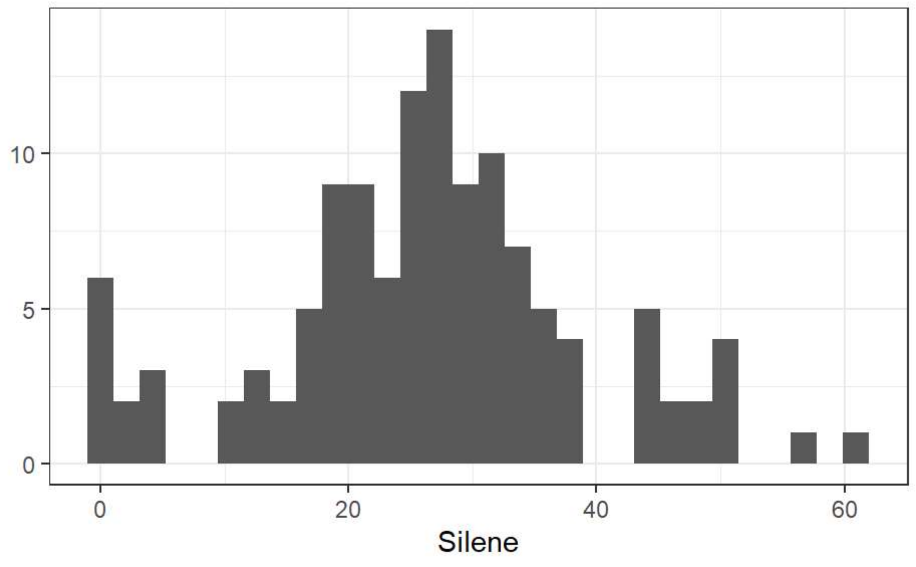
```
qplot(x=Silene,data=MyData) + theme_grey() # default
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



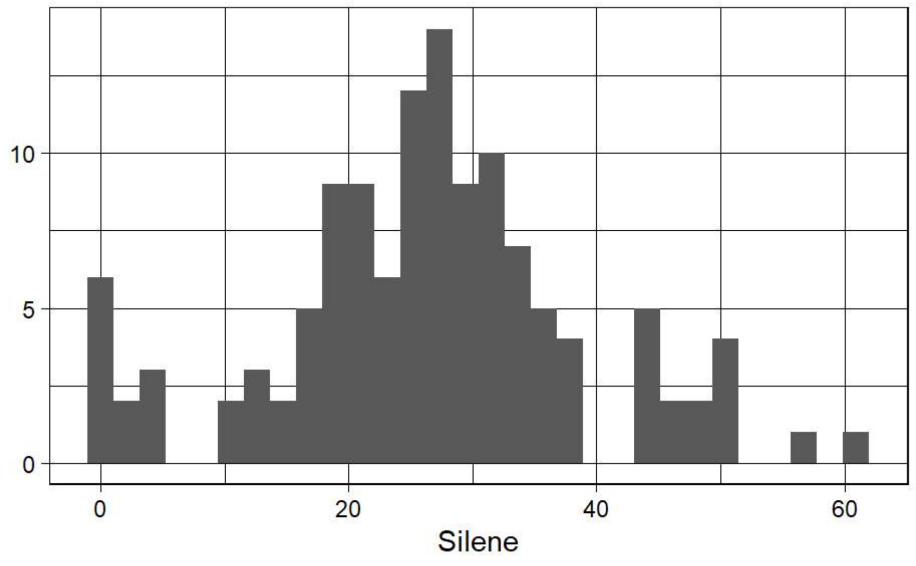
```
qplot(x=Silene,data=MyData) + theme_bw() # cleaner, better contrast
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



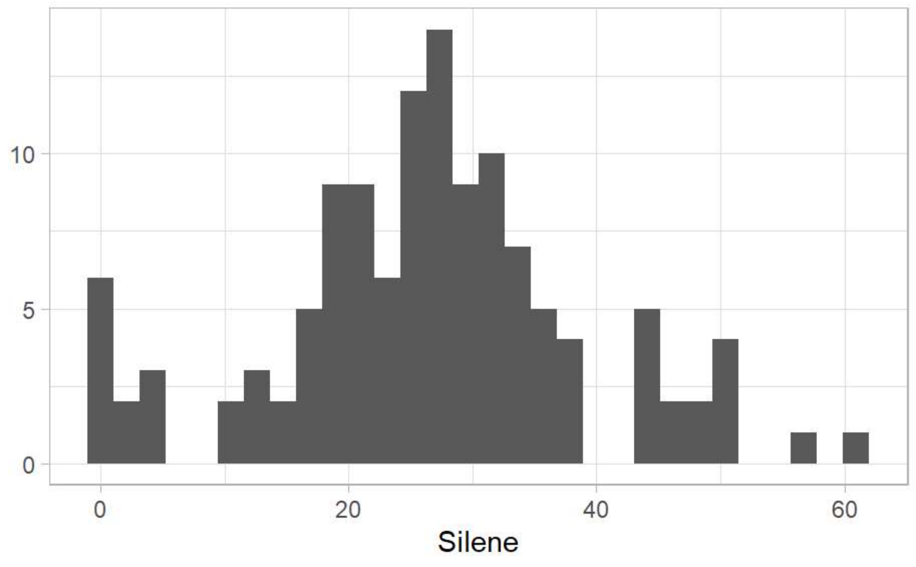
```
qplot(x=Silene,data=MyData) + theme_linedraw() # thicker grid lines
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



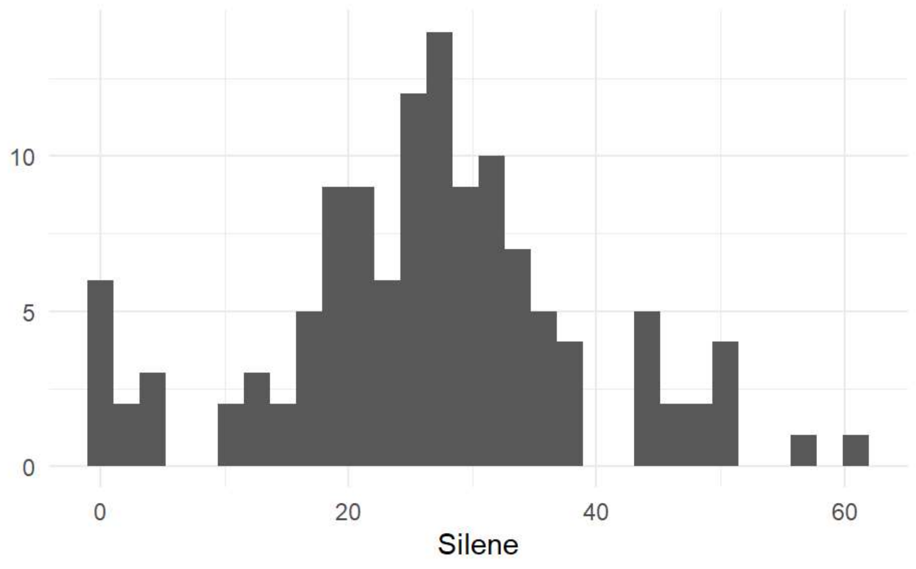
```
qplot(x=Silene,data=MyData) + theme_light() # fainter border and axis values
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



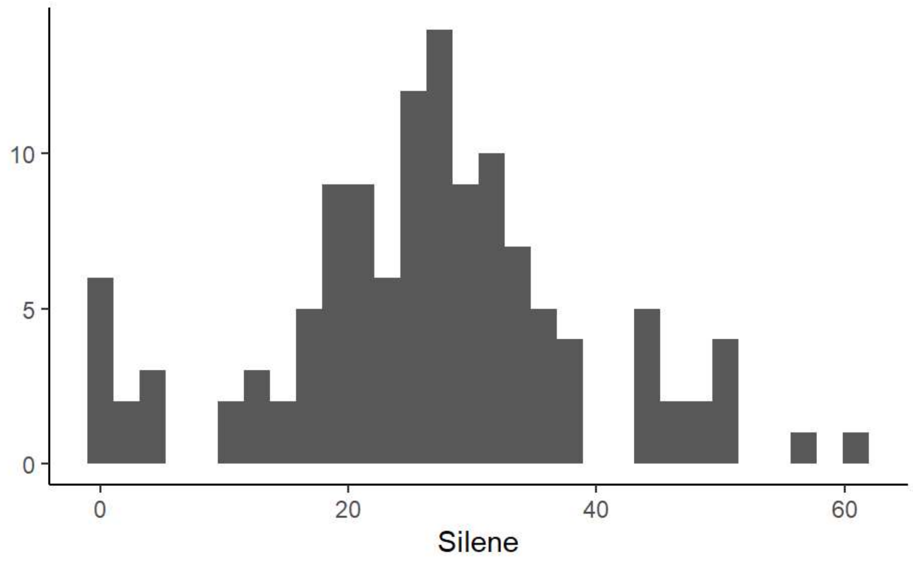
```
qplot(x=Silene,data=MyData) + theme_minimal() # no borders
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
qplot(x=Silene,data=MyData) + theme_classic() # x and y lines only, no tick marks
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Or create your own custom theme:

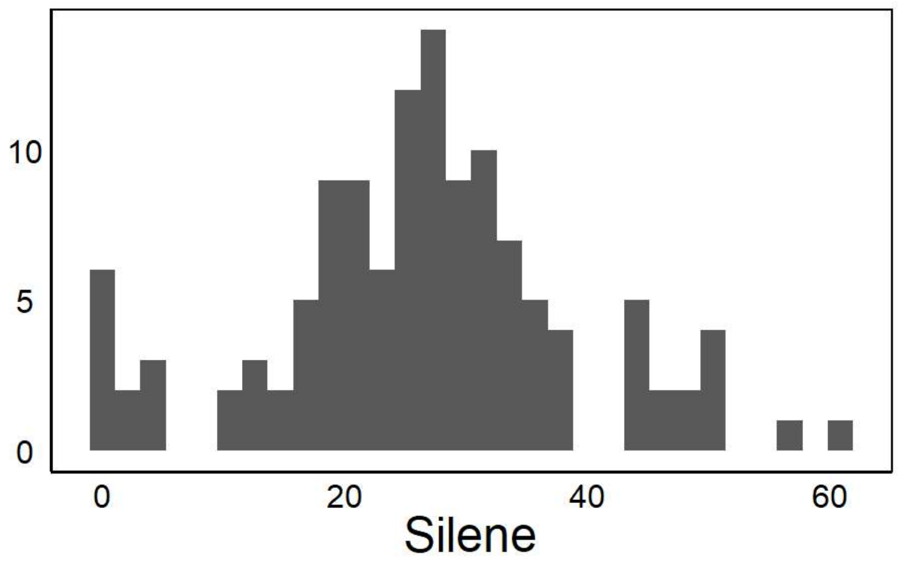
Simplified/clean version of theme_classic with bigger axis labels more suitable for presentation or publication:

```
# Clean theme for presentations & publications used in the Colautti Lab
theme_pubworthy <- function (base_size = 12, base_family = "") {
  theme_classic(base_size = base_size, base_family = base_family) %+replace%
    theme(
      axis.text = element_text(colour = "black"),
      axis.title.x = element_text(size=18),
      axis.text.x = element_text(size=12),
      axis.title.y = element_text(size=18,angle=90),
      axis.text.y = element_text(size=12),
      axis.ticks = element_blank(),
      panel.background = element_rect(fill="white"),
      panel.border = element_blank(),
      plot.title=element_text(face="bold", size=24),
      legend.position="none"
    )
}
```

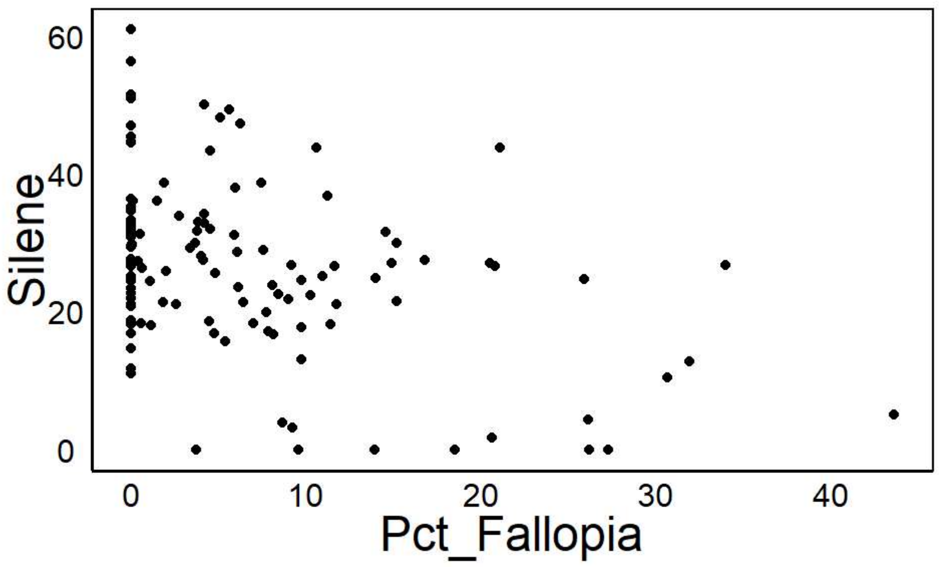
What it looks like:

```
qplot(x=Silene,data=MyData) + theme_pubworthy() # A clean format with bigger axis label
         s for publication
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
qplot(x=Pct_Fallopia,y=Silene,data=MyData) + theme_pubworthy() # Bivariate plot
```



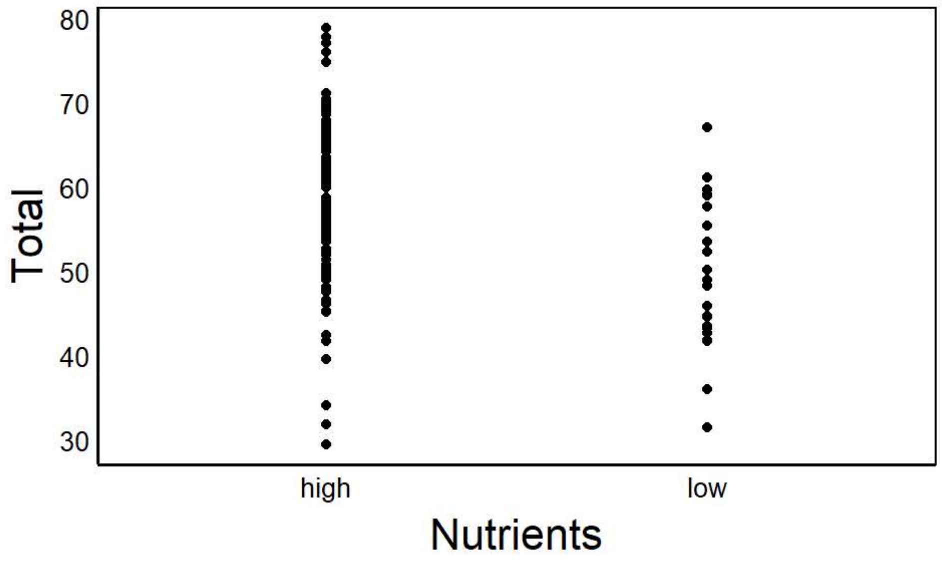
geom =

See website for list of geoms

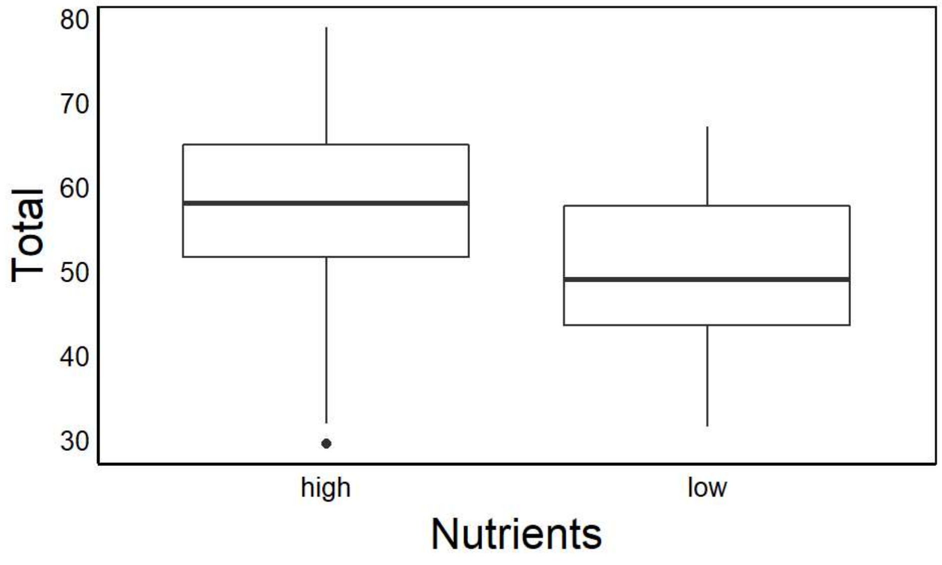
http://docs.ggplot2.org/ (http://docs.ggplot2.org/)

A couple of examples:

```
qplot(x=Nutrients,y=Total,data=MyData) # Basic
```

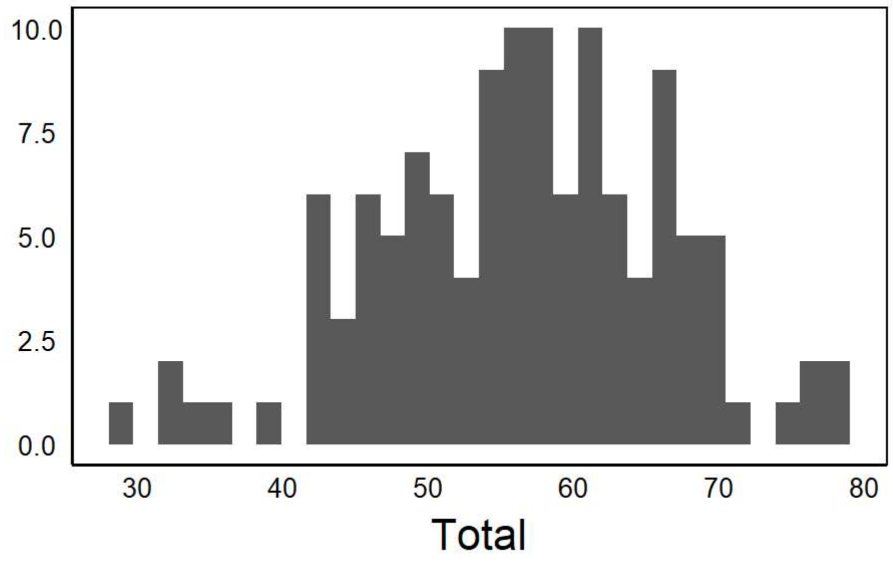


```
qplot(x=Nutrients,y=Total,data=MyData,geom="boxplot") # Geom
```

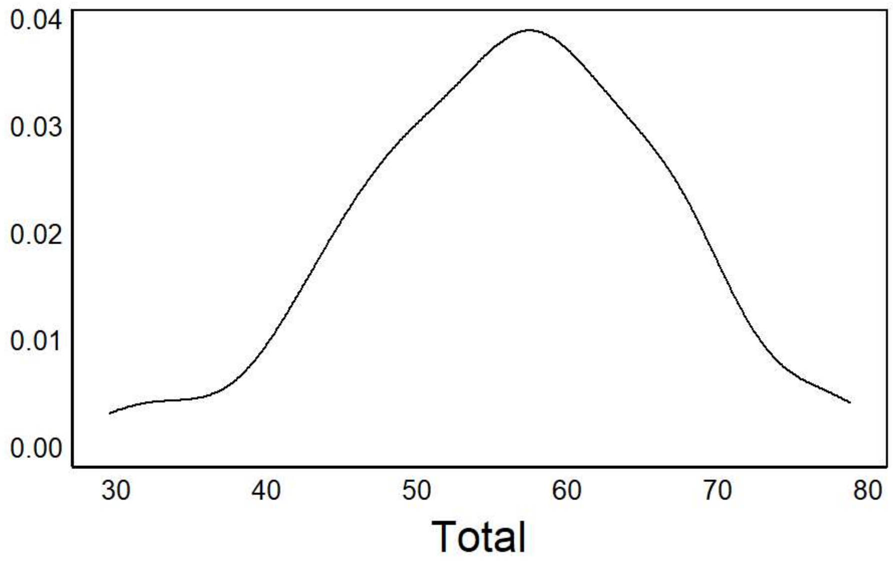


```
qplot(Total,data=MyData) # Basic
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
qplot(Total,data=MyData,geom="density") # Geom
```



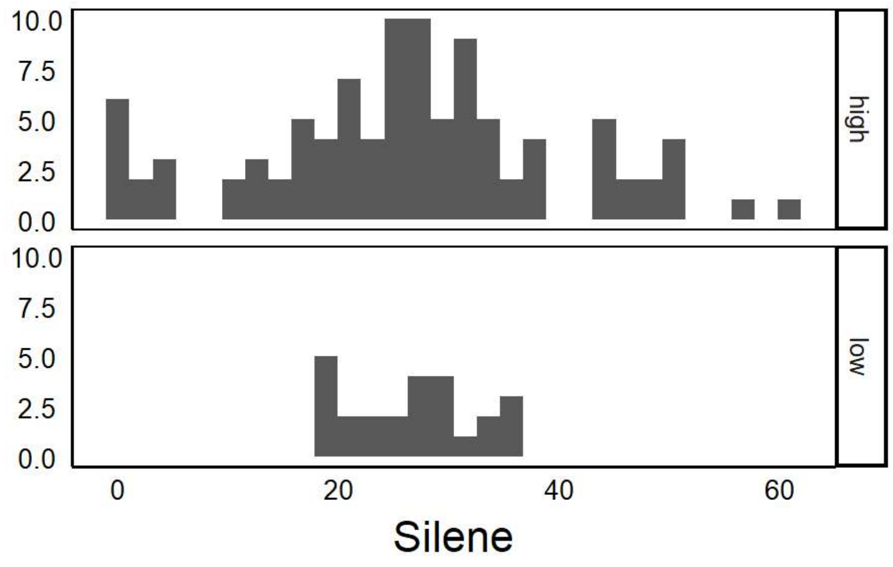
6. Multiple graphs

***facets* =**

General form: facets=Vertical~Horizontal

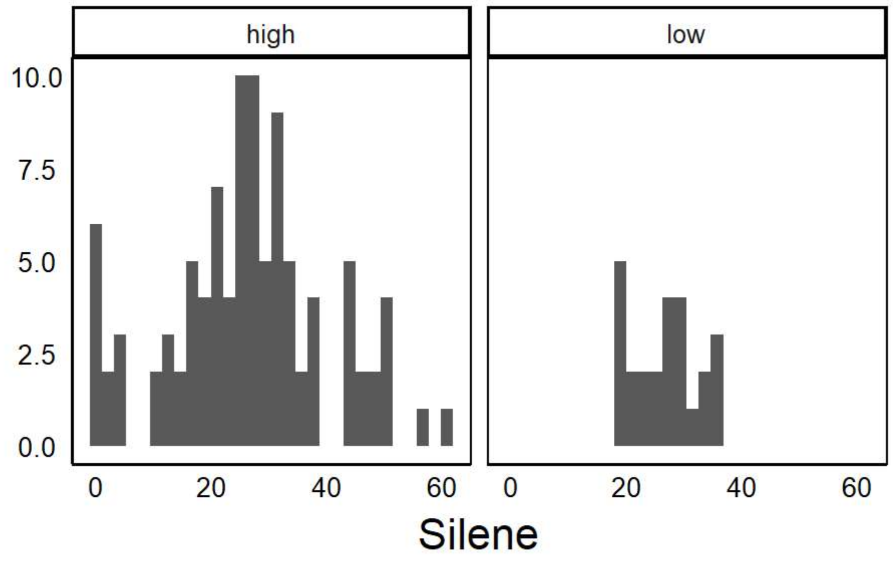
```
qplot(x=Silene,data=MyData,facets=Nutrients~.) # Vertical stacking
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



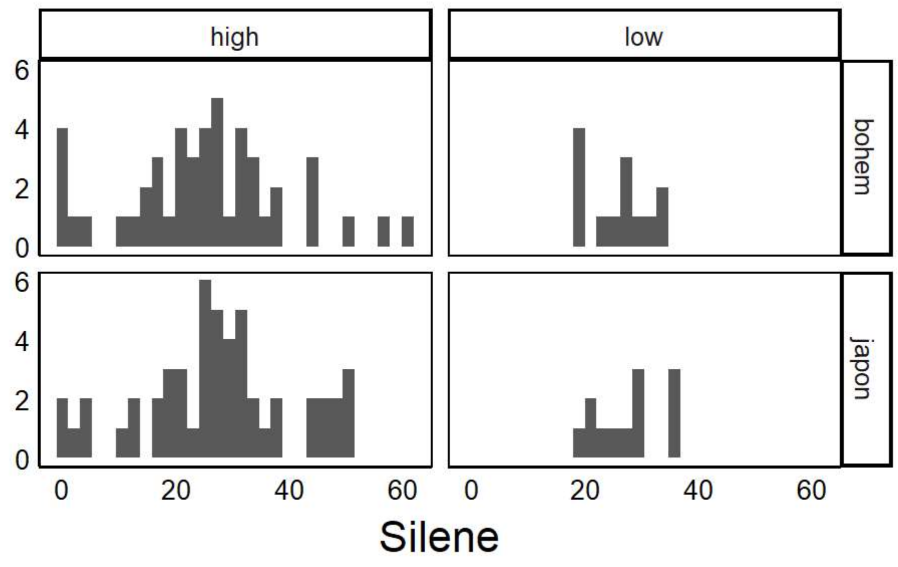
```
qplot(x=Silene,data=MyData,facets=.~Nutrients) # Horizontal stacking
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
qplot(x=Silene,data=MyData,facets=Taxon~Nutrients) # Both (2 column stacking)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



7. Save output

Saving graphs/images requires three steps

1. Open a file (e.g. pdf, svg, png)
2. Create the graph inside the file
3. Close the file
    - Important: If you don't close the file, it is unusable.

```
pdf("SileneHist.pdf") # 1. Create and open a file
  qplot(x=Silene,data=MyData,facets=Taxon~Nutrients) # 2. Write the graph info
dev.off() # 3. Close the file
```

Note how the qplot command does not open in the plots window

This is because the info is sent to "SileneHist.pdf" instead