# Mini-Tutorials for Creating Graphs in R Scatterplots

The examples below are intended to instruct you to create statistical graphs in R with minimal initial training in R. You should be able to follow the example codes to obtain graphs by modifying the included code. Some examples (and a key) will be included at the end of the document for practice. The screenshots I will show of the environment use R Studio, which is a free program you can find online. Other R environments will look different, but probably have similar functionality.

R version 3.3.2 (2016-10-31) "Sincere Pumpkin Patch" Copyright (C) 2016 The R Foundation for Statistical Computing Platform: x86.64-w64-mings/X-64 (64-bit) R is free software and comes with ABSOLUTELY NO WARRANTY. You are welcome to redistribute it under certain conditions. Type 'idemcO)' or litence()' for distribution details. R is a collaborative project with many contributors. Type 'idemcO)' for some denos, 'help()' for on-line help, or 'help,start()' for an HTML browser interface to help. Type 'idem()' for some denos, 'help()' for on-line help, or 'help,start()' for an HTML browser interface to help. Type 'idem O' clut is immed to be interface to help. Type 'idem O' clut is immed to be interface to help. Type 'idem O' clut is immed to be interface to help. Type 'idem O' clut is immed to be interface to help. Type 'idem O' clut is immed to be interface to help. Type 'idem O' clut is immed to be interface to help. Type 'idem O' clut is immed to be interface to help. Type 'idem O' clut is immed to be interface to help. Type 'idem O' of the second to be interface to help. Type 'idem O' of the second to be interface to help. Type 'idem O' of the second to be interface to help. Type 'idem O' of the second to be interface to help. Type 'idem O' of the second to be interface to help. Type 'idem O' of the second to be interface to help. Type 'idem O' of the second to be interface to help. Type 'idem O' of the second to be interface to be interface to help. Type 'idem O' of the second to be interface to be interface to help. Type 'idem O' of the second to be interface tob be interface to be interface to be int	Console Terminal ×			
R version 3.3.2 (2016-10-31) "Sincere Pumpkin Patch" Copyright (C) 2016 The R Foundation for Statistical Computing Platform: x86.64-w64-mings/X646 (d-bit) R is free software and comes with ABSOLUTELY NO WARRANTY. You are welcome to redistribute it under certain conditions. Type 'idemcO)' or literation and 'citation(O)' on how to cite R or R packages in publications. Type 'qO' to guit R. >				- 0
R version 3.3.2 (2016-10-31) "Sincere Pumpkin Patch" Copyright (C) 2016 The R Foundation for statistical Computing Platform: x86_64-w6d-mingu32/x84 (64-bit) R is free software and comes with ABSOLUTELY NO WARRANTV. You are welcome to redistribute it under certain conditions. Type 'icense()' or 'licence()' for distribution details. R is a collaborative project with many contributors. Type 'icense()' or more information and 'citation()' on how to cite R or R packages in publications. Type 'a()' to quit R. Files Plots Packages Help Viewer New Yolder O Dotes R or mane Modified Files Plots Packages Help Viewer New Yolder O Dotes R or mane Modified Modified * Name Size Modified	~/R/training/ 🔗			🗏 List 🕶 🗌 🤅
You are welcome to redistribute it under certain conditions. Type 'license()' or 'licence()' for distribution details. R is a collaborative project with many contributors. Type 'contributors()' for more information and 'citation()' on how to cite R or R packages in publications. Type 'de()' for an HTML browser interface to help. Type 'de()' to quit R.	opyright (C) 2016 The R Foundation for Statistical Computing			Q,
Type 'contributors()' for more information and 'citation()' on how to cite R or R packages in publications. Type 'de()' for ann HTML browser interface to help. Type 'd()' to quit R.	ou are welcome to redistribute it under certain conditions.			
'help,start()' for an HTML browser interface to help. Type 'q()' to quit R. ► New Tolder O Delete Ramane @ More - ▲ Nome R. Training ▲ Nome R. Training ▲ Nome R. Training	ype 'contributors()' for more information and			
	help.start()' for an HTML browser interface to help.	Files Plots Packages Help Viewer		-0
A Name Size Modified		💁 New Folder 👂 Delete 📻 Rename 🔮 More 👻		
<b>1</b>		Arrow Home > R > training		8
		A Name	Size	Modified
Image: Straning Sprej         218 B         Dec 14. 2019.		1 L .		
		🔲 🔋 training.Rproj	218 B	Dec 14, 2019, 2:14 PM

When you open up a new project environment in R Studio, it looks like this.

The command line environment is on the left. Images when we construct them will appear on the bottom right. As we add variables, they will appear in the list at the top right (name, dimensions and samples will display, which is useful for checking that you didn't skip entries when entering data by hand).

We are going to start by creating a simple scatter plot graph from raw data. We'll add features as we go such as finding the trendline, making a residual plot and other related graphs.

Copy the commands shown into the command line.

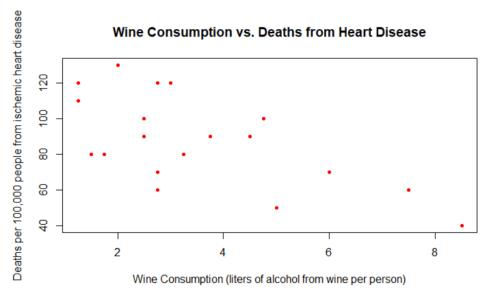
**Step 1.** Enter the data to be plotted in vector form. For a scatterplot, we need two sets of data to be correlated.

wine=c(3.25,4.75,2.75,1.5,4.5,3,8.5,3.75,1.25,2,7.5,2.75,2.5,1.75,5,2.5,6,2.75,1.25)

Use the window at the top right, if you are entering the data by hand especially, to ensure that the two sets have the same length.

## Step 2. Graph the data.

plot(wine, heart, main="Wine Consumption vs. Deaths from Heart Disease", xlab="Wine Consumption (liters of alcohol from wine per person)", ylab="Deaths per 100,000 peopl e from ischemic heart disease", col="red", pch=20)



The graph we get is shown. The default plot is a scatterplot when the variables are not sorted, and you can adjust the type of dot appearing at each value by changing the "pch" option, as well as the color. Good graphs should label both axes and have a descriptive title. When listing the variables to be graphed, be sure to list the x-variable first and the y-variable second.

**Step 3.** In order to add the regression line, we first need to get R to calculate it, and then obtain the coefficients we need to draw the line.

# reg1=lm(heart~wine)

For the regression function, "Im" stands for linear model, and the y-variable goes first. You can use the same function with multiple x's used to predict a single y. Use the summary function to obtain information about the analysis, including the coefficients. The command is shown along with the printed output. (The input is in blue as before, and the output is in black.)

```
summary(reg1)
```

Call:

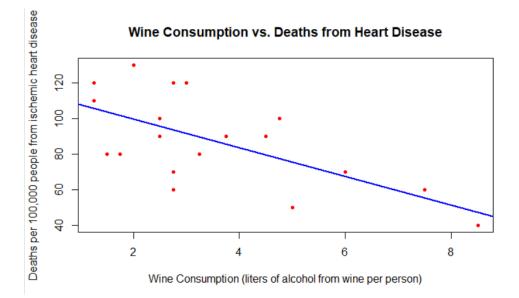
```
Im(formula = heart ~ wine)
Residuals:
  Min
         1Q Median
                       3Q
                             Max
-33.724 -15.736 4.202 12.283 30.239
Coefficients:
      Estimate Std. Error t value Pr(>|t|)
(Intercept) 115.861
                      9.399 12.327 6.67e-10 ***
          -8.050
                   2.314 -3.479 0.00287 **
wine
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 20.11 on 17 degrees of freedom
Multiple R-squared: 0.4159, Adjusted R-squared: 0.3815
F-statistic: 12.1 on 1 and 17 DF, p-value: 0.002871
```

I've highlighted the two coefficients we need for drawing the equation.

Add the equation to the graph with the command shown, using the coefficient obtained.

abline(a=115.861, b=-8.05, col="blue", lwd=2)

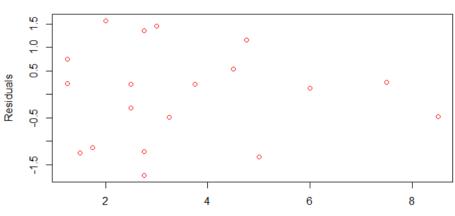
The equation graphed is y = a + bx, so a is the intercept and b is the slope. The graph produced is shown.



**Step 4.** To continue the analysis, we can obtain a residual plot. First, calculate the standardized residuals.

wine.stdres = rstandard(reg1)

plot(wine, wine.stdres, main="Residual Graph vs. Wine Consumption", xlab="Wine Cons umption (liters of alcohol from wine per person)", ylab="Residuals", col="red", pch=1)



Residual Graph vs. Wine Consumption

Wine Consumption (liters of alcohol from wine per person)

After finding the residuals, we replot them against the x-variable. We expect the residuals to be random if the assumptions of the linear regression are met. If you have multiple variables in your model, you should do this with each input variable.

You can also construct a normal probability plot for the residuals, but we'll leave that for a separate lesson.

# Practice.

You are welcome to practice with randomly generated data, but you should expect that the correlations will be extremely low.

 Two random sets. test1=c(runif(150, min=80, max=100)) test2=c(rnorm(150, mean=64, sd=3.1))

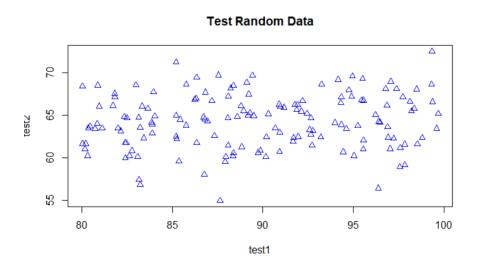
The first set is uniformly distributed, while the second is normally distributed numbers with the specified mean and standard deviation. The test sets should be the same size.

 A second set of real data. nickel=c(5.9, 2.8, 5.3, 4.6, 4.6, 3.5, 3.8, 5, 5.3, 4.2) tens\_streng=c(948, 859, 921, 909, 915, 876, 828, 964, 964, 900)

### Solutions.

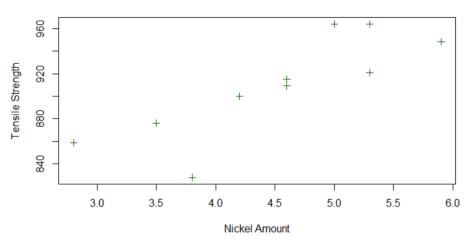
The random test data shows no correlation.

plot(test1,test2, main="Test Random Data", xlab="test1", ylab="test2", col="blue", pch= 2)



For the second set, there is a correlation.

plot(nickel,tens\_streng, main="Tensile Strength vs. Nickel Concentration", xlab="Nickel Amount", ylab="Tensile Strength", col="darkgreen", pch=3)



## Tensile Strength vs. Nickel Concentration

Let's find the regression line.

strength.lm = lm(tens\_streng ~ nickel)

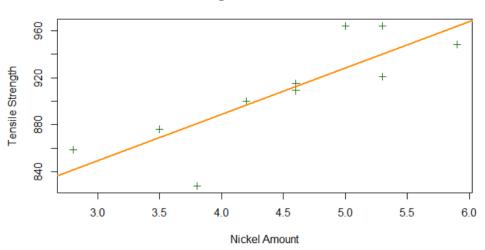
#### summary(strength.lm)

```
Call:
Im(formula = tens streng ~ nickel)
Residuals:
  Min
         1Q Median
                       3Q
                             Max
-52.882 -12.409 3.031 14.799 35.945
Coefficients:
      Estimate Std. Error t value Pr(>|t|)
(Intercept) 731.502 43.147 16.95 1.49e-07 ***
nickel
         39.311
                   9.405 4.18 0.00308 **
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 26.57 on 8 degrees of freedom
```

Multiple R-squared: 0.6859, Adjusted R-squared: 0.6467 F-statistic: 17.47 on 1 and 8 DF, p-value: 0.00308

Make our regression line.

abline(a=731.502, b=39.311,col="darkorange",lwd=2)

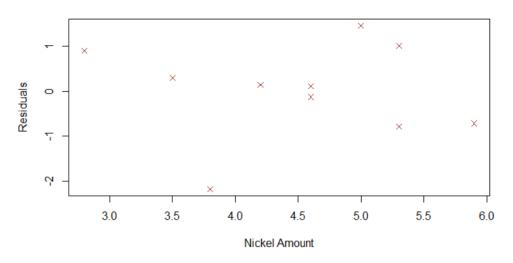


#### **Tensile Strength vs. Nickel Concentration**

And we can make our residual graph, too.

strength.stdres = rstandard(strength.lm)

plot(nickel,strength.stdres, main="Residuals vs. Nickel Concentration", xlab="Nickel Am ount", ylab="Residuals", col="brown", pch=4)



**Residuals vs. Nickel Concentration**