

2021

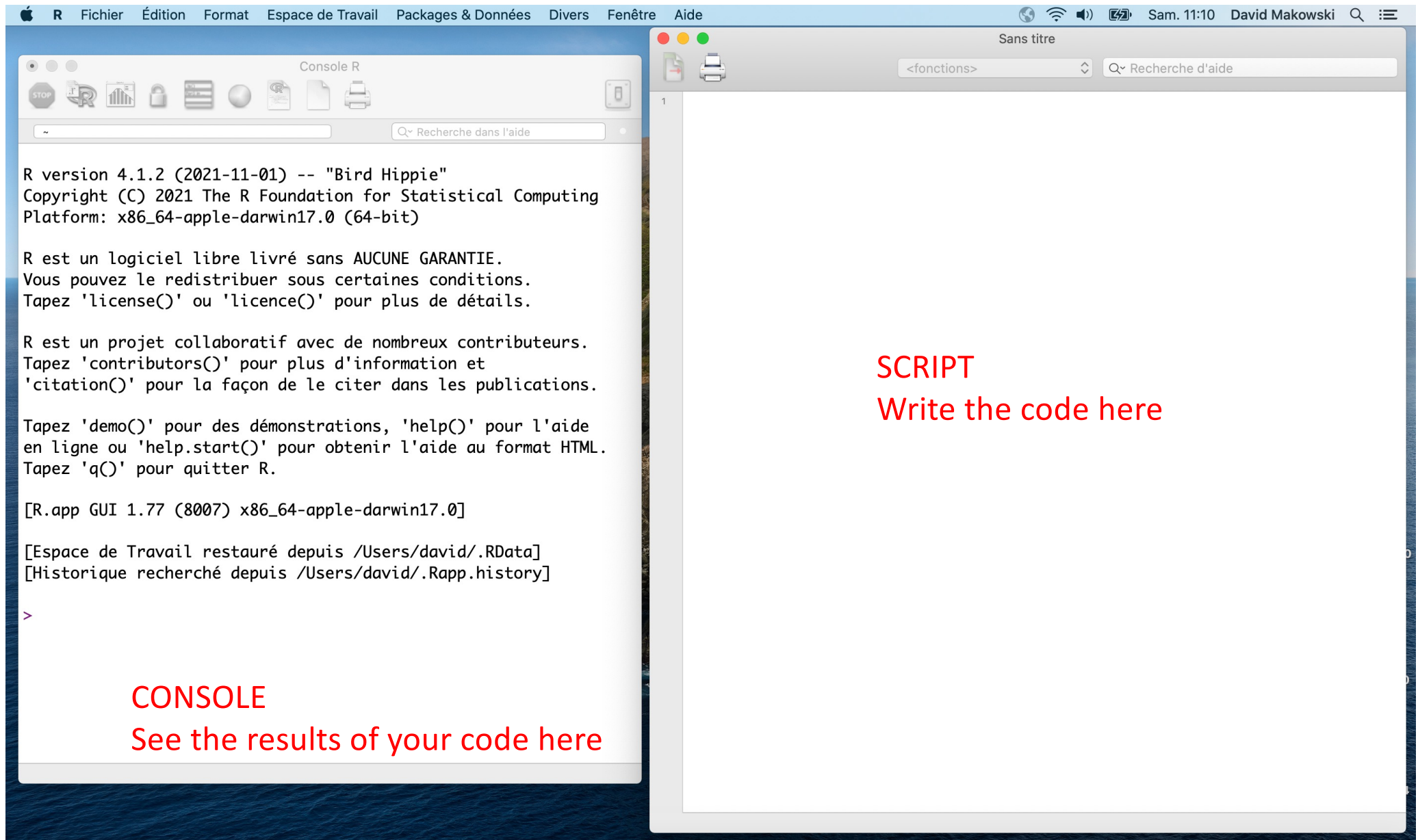
# A very short introduction to R

David Makowski

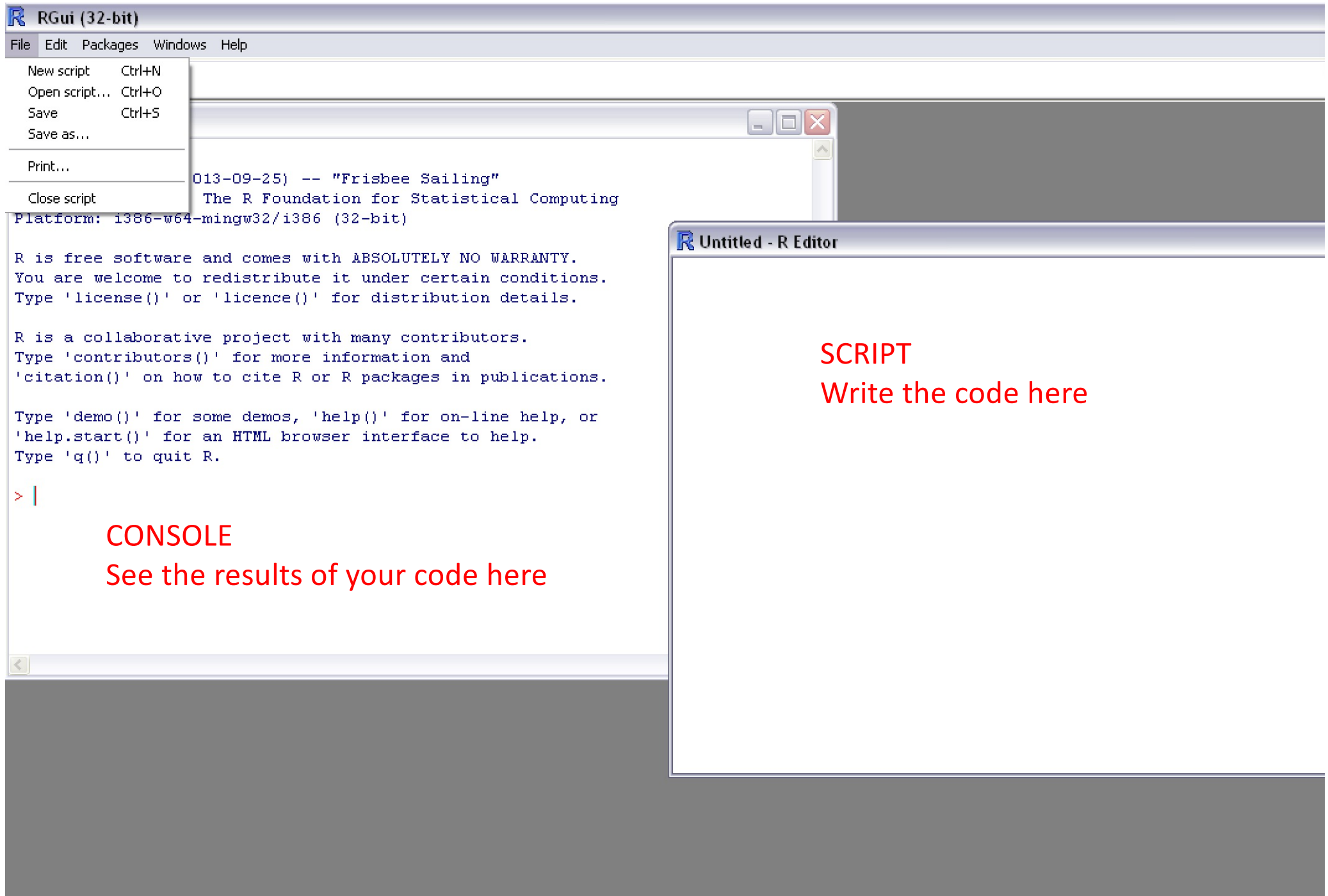
# R

- Derived from the language S
- Created in 1993 and developed by *R development core team* ([www.r-project.org](http://www.r-project.org))
- High level language
- Interpreted (no compilation)
- Free
- Very active community with new packages every year
- Useful for
  - Statistical analysis
  - Machine learning
  - Graphics
  - Modelling
  - Text mining

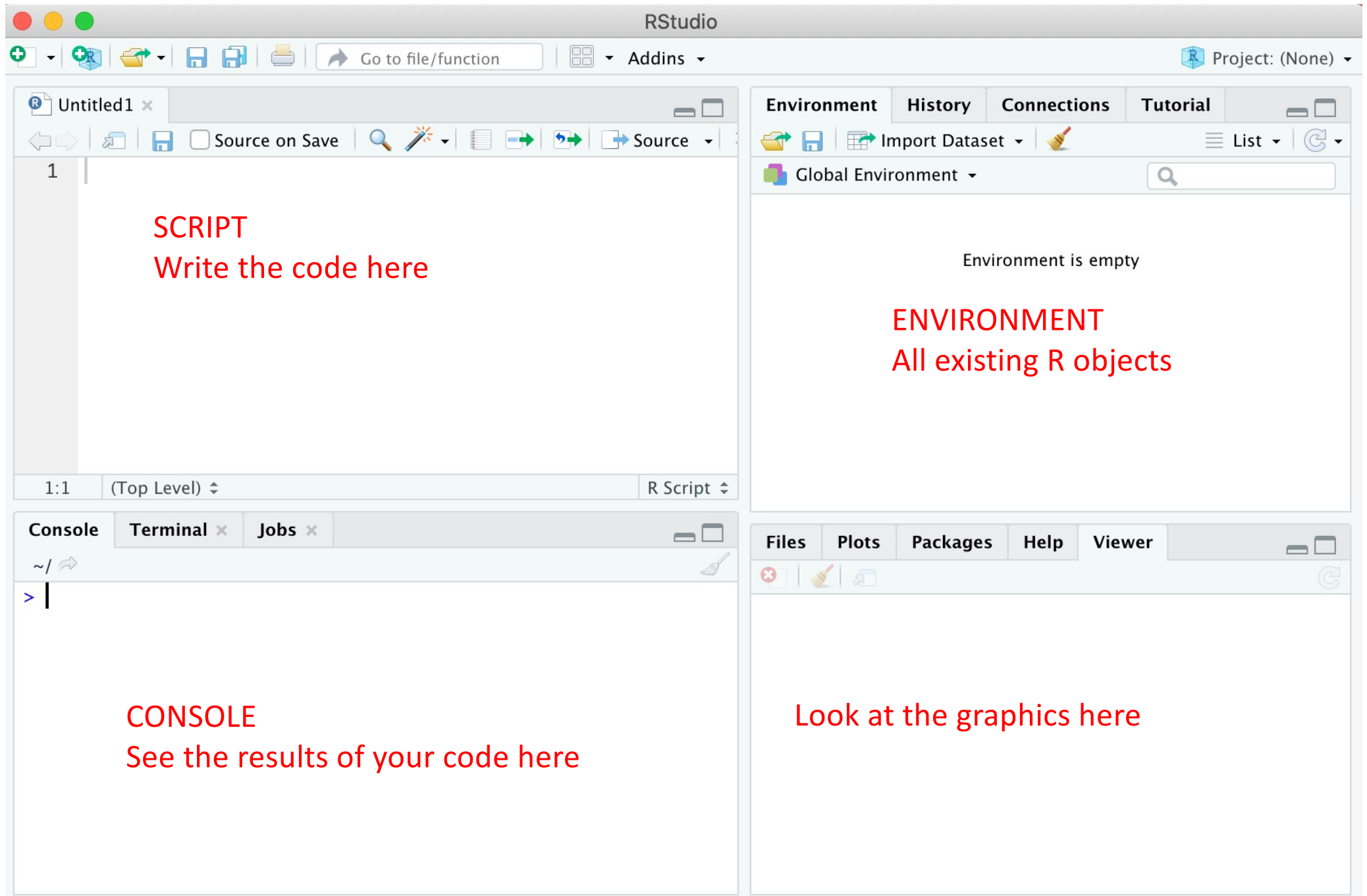
# Standard R interface when using a Mac



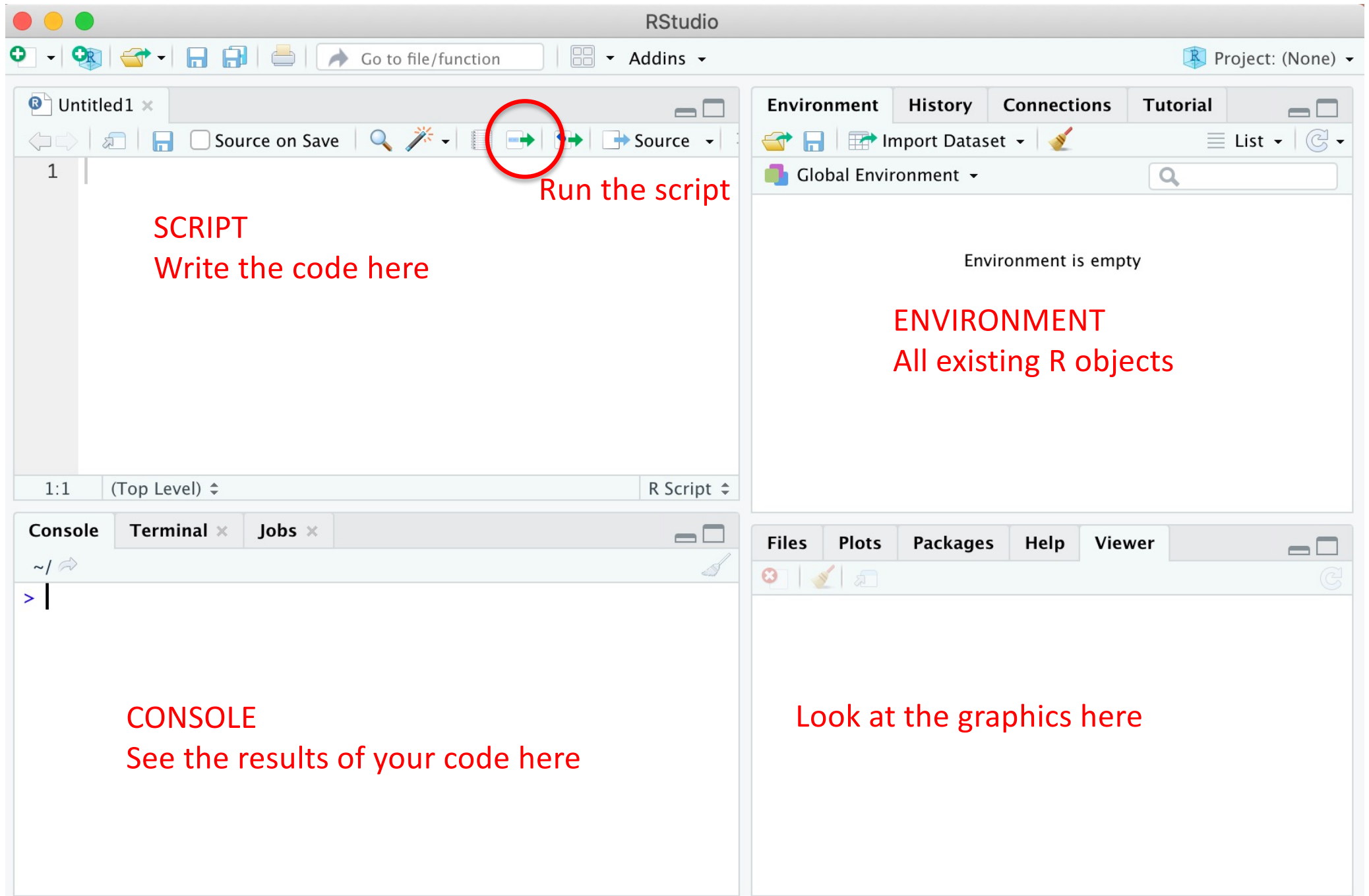
# Standard R interface when using Windows



# R studio interface



# R studio interface



# R studio interface

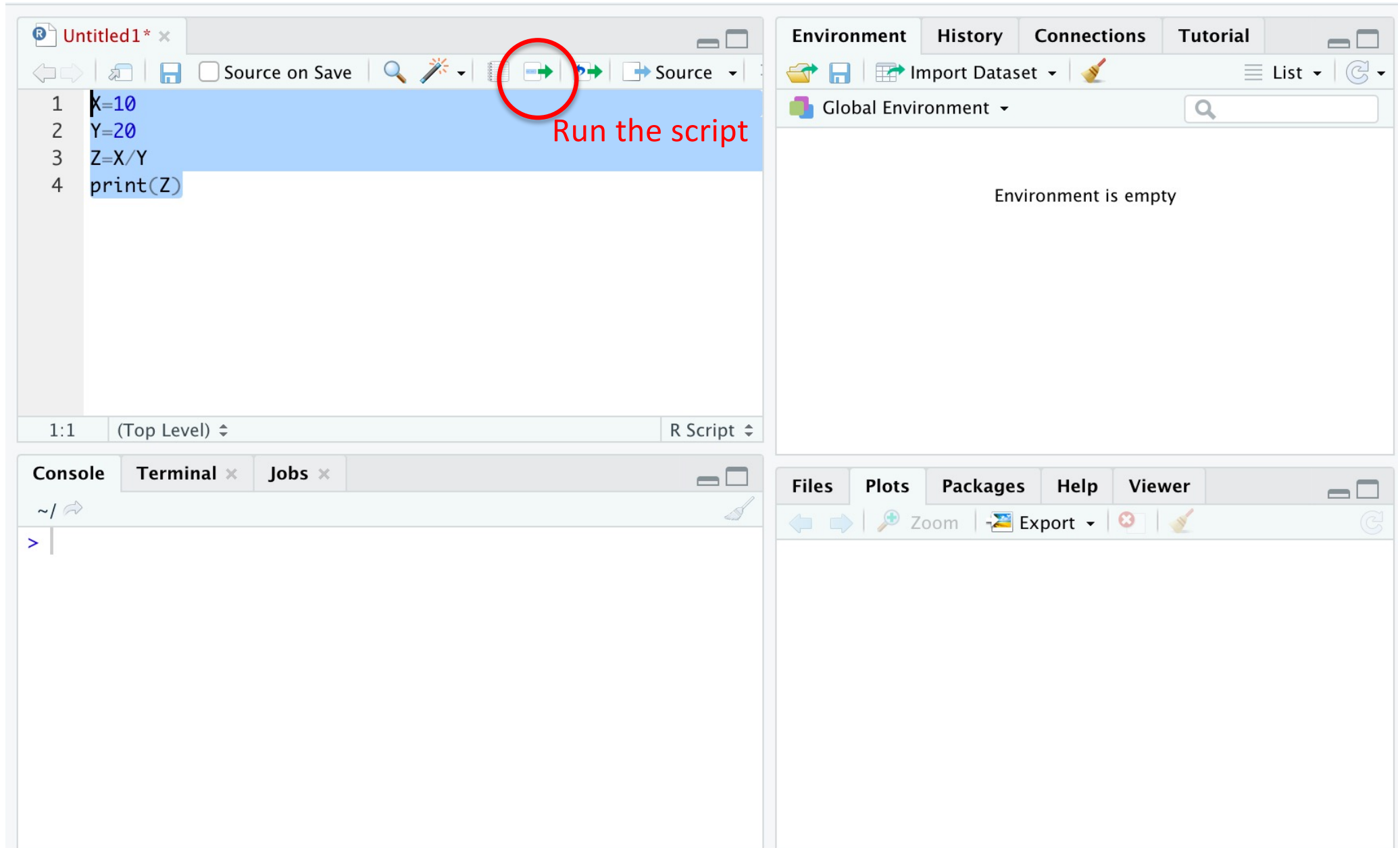
The image displays the R Studio interface, which is organized into several panes:

- Top Bar:** Contains icons for file operations (new, open, save, print) and a search bar labeled "Go to file/function". On the right, it shows "Project: (None)".
- Source Editor:** The central area for writing R code. It shows a file named "Untitled1\*" with the following code:

```
1 X=10
2 Y=20
3 Z=X/Y
4 print(Z)
```

Below the code, it indicates the current line is 4:9 at the (Top Level) in an R Script file.
- Environment Pane:** Located on the right side, it shows the "Global Environment" and states "Environment is empty". It includes a search bar and a "List" button.
- Console:** Located at the bottom left, it shows the prompt ">" and a tilde symbol (~) with a refresh icon.
- Files Pane:** Located at the bottom right, it shows a search bar and a "Zoom" button.
- Plots Pane:** Located at the bottom right, it shows a search bar and a "Export" button.
- Packages Pane:** Located at the bottom right, it shows a search bar and a "Help" button.
- Viewer Pane:** Located at the bottom right, it shows a search bar and a "Viewer" button.

# R studio interface





# R studio interface

The screenshot displays the R Studio interface with the following components:

- Source Editor:** Contains a script named "Untitled1\*" with the following code:

```
1 X=10
2 Y=20
3 Z=X/Y
4 print(Z)
```
- Environment Pane:** Shows the "Global Environment" with the following values:

Values	
X	10
Y	20
Z	0.5
- Console:** Shows the execution output:

```
> X=10
> Y=20
> Z=X/Y
> print(Z)
[1] 0.5
>
```
- Files Pane:** Empty.
- Plots Pane:** Empty.
- Packages Pane:** Empty.
- Help Pane:** Empty.
- Viewer Pane:** Empty.

# R

- Vector
- Matrix
- Data.frame
- List
- Functions
- Loops
- Conditions
- Graphics
- Linear regression

# Vector

```
> X<-c(1,2,3,4)
```

```
> X
```

```
[1] 1 2 3 4
```

```
> X<-1:4
```

```
> X
```

```
[1] 1 2 3 4
```

```
> X<-seq(1,4,by=1)
```

```
> X
```

```
[1] 1 2 3 4
```

```
> X<-seq(1,4,by=0.1)
```

```
> X
```

```
[1] 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0 2.1 2.2 2.3 2.4 2.5 2.6 2.7 2.8 2.9 3.0 3.1  
3.2 3.3 3.4 3.5 3.6 3.7 3.8 3.9 4.0
```

```
>
```

# Vector

```
> X<-c(10,20,50,99,-123)
```

```
> X[3]
```

```
[1] 50
```

```
> X[3:5]
```

```
[1] 50 99 -123
```

```
> X[c(1,4)]
```

```
[1] 10 99
```

# Vector

```
X<-1:10
```

```
> X
```

```
[1] 1 2 3 4 5 6 7 8 9 10
```

```
> Y<-seq(10,1, by=-1)
```

```
> Y
```

```
[1] 10 9 8 7 6 5 4 3 2 1
```

```
> X+Y
```

```
[1] 11 11 11 11 11 11 11 11 11 11
```

```
> X*Y
```

```
[1] 10 18 24 28 30 30 28 24 18 10
```

```
> X%*%Y
```

```
[,1]
```

```
[1,] 220
```

# Vector of characters

- Vector of texts
- Use `` or ‘

```
x<- c("I","like","Lille","very","much")  
x  
[1] "I"      "like"    "Lille"   "very"    "much"
```

# Matrix

```
> MyNiceMatrix<-matrix(0,nrow=2,ncol=2)
```

```
> MyNiceMatrix
```

```
  [,1] [,2]
```

```
[1,]  0  0
```

```
[2,]  0  0
```

```
> MyBeautifulMatrix<-matrix(3, nrow=2, ncol=2)
```

```
> MyBeautifulMatrix
```

```
  [,1] [,2]
```

```
[1,]  3  3
```

```
[2,]  3  3
```

```
> MyWonderfulMatrix<-MyNiceMatrix+MyBeautifulMatrix+12
```

```
> MyWonderfulMatrix
```

```
  [,1] [,2]
```

```
[1,] 15 15
```

```
[2,] 15 15
```

# data.frame

```
> Location<-c("London","Paris","Madrid")
> Temperature<-c(10,15,22)
> Rating<-c("Awful", "Medium", "OK")
> Weather<-data.frame(Location,Temperature,Rating)
> Weather
>
```

	Location	Temperature	Rating
1	London	10	Awful
2	Paris	15	Medium
3	Madrid	22	OK



# data.frame

**Important: A data.frame can be created from an external file (.txt, .csv, .xls) using specific functions.**

**Example:**

```
TAB=read.table('filename.txt', header=T).
```

# list

```
> MyList<-list(Location, Temperature, Rating)
```

```
> MyList
```

```
[[1]]
```

```
[1] "London" "Paris" "Madrid"
```

```
[[2]]
```

```
[1] 10 15 22
```

```
[[3]]
```

```
[1] "Awful" "Medium" "OK"
```

```
> MyList[[2]]
```

```
[1] 10 15 22
```

```
> MyList[[2]][3]
```

```
[1] 22
```

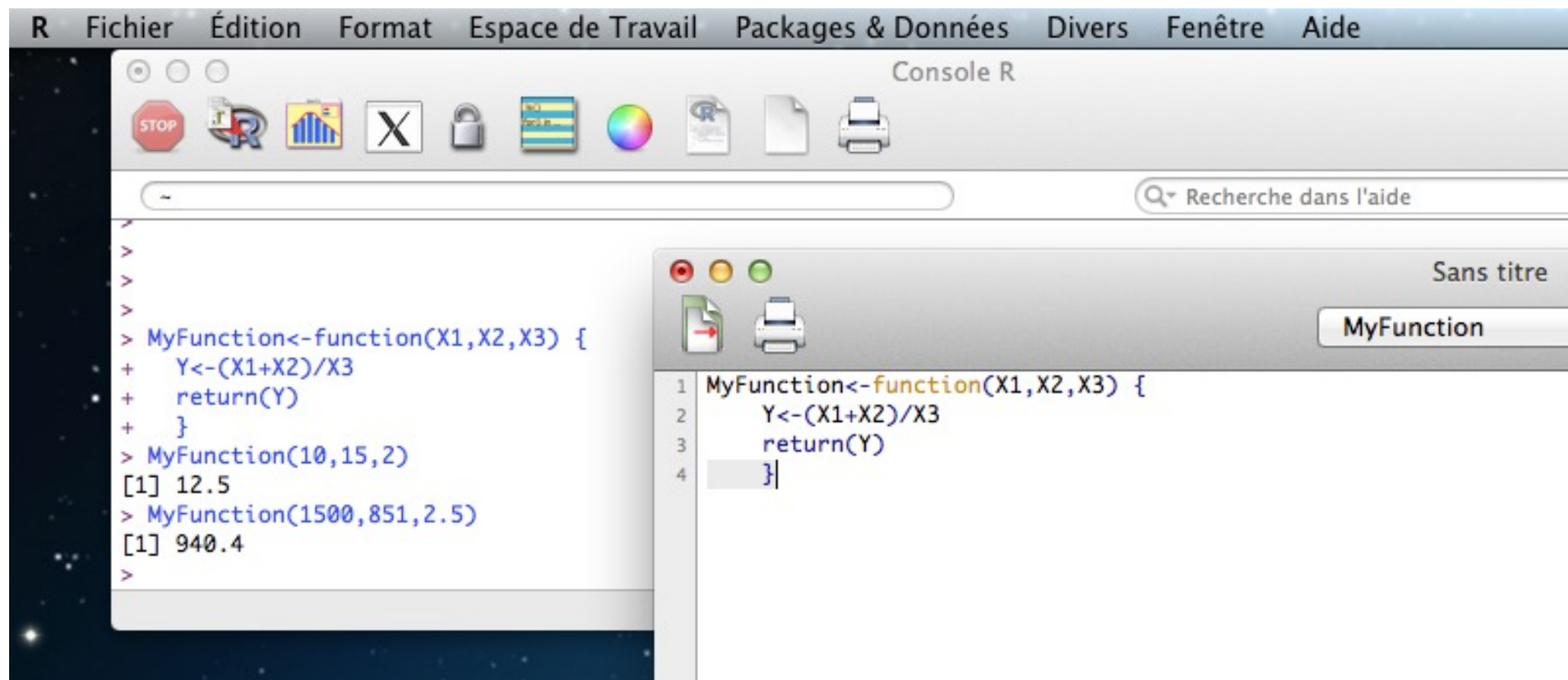
```
>
```

# fonctions

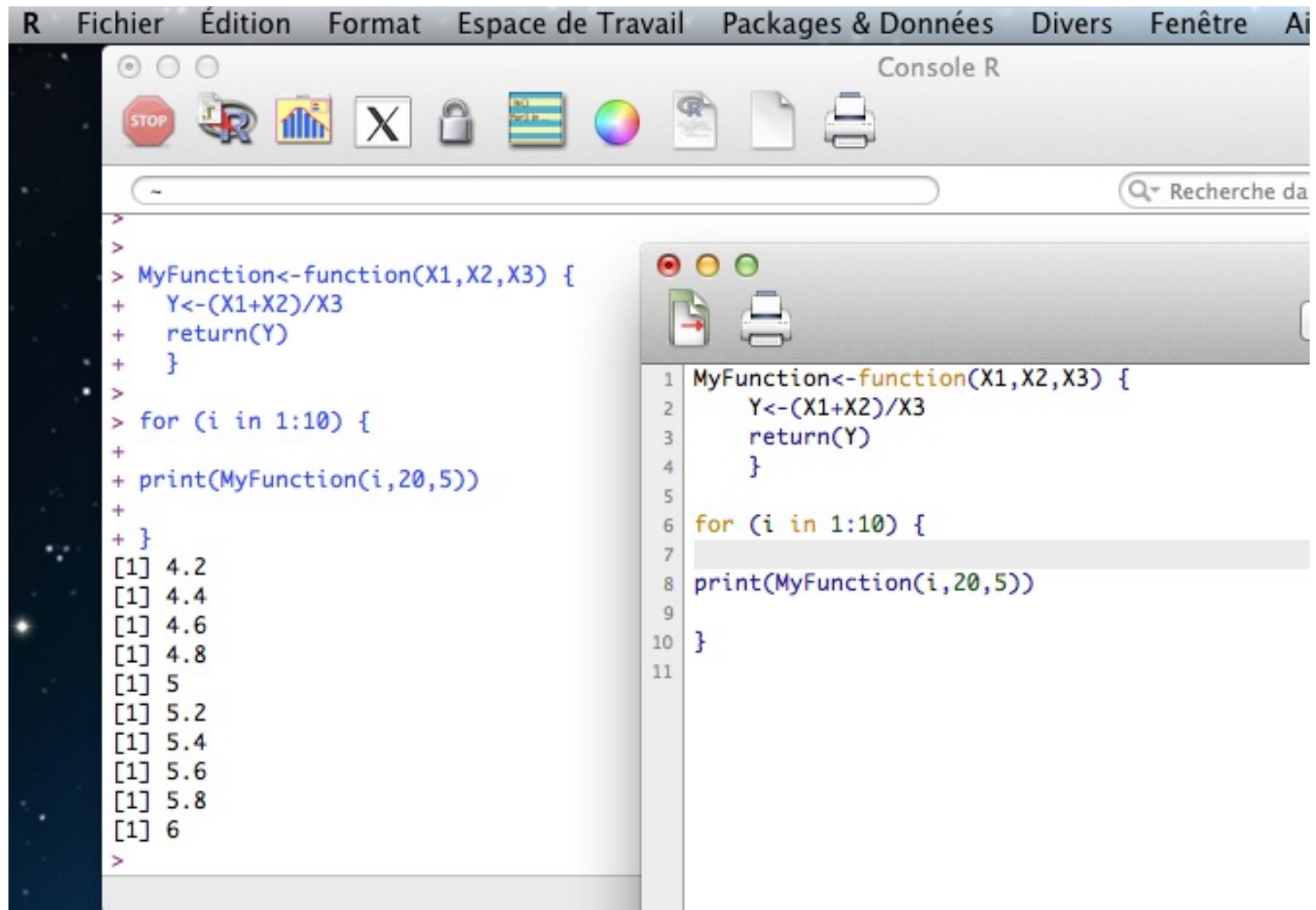
- Existing functions

`print()`, `log()`, `exp()`, `plot()`, `lm()` etc.

- New functions



# Loop: for (i in 1:N) {}



The screenshot displays the RStudio environment. The top menu bar includes 'R', 'Fichier', 'Édition', 'Format', 'Espace de Travail', 'Packages & Données', 'Divers', 'Fenêtre', and 'Ai'. Below the menu is a toolbar with icons for stopping, running, saving, and other functions. The main console window on the left shows the execution of a for loop that calls a custom function. The output of the loop is displayed line by line. The script editor on the right shows the definition of the function 'MyFunction' and the loop structure.

```
> 
> 
> MyFunction<-function(X1,X2,X3) {
+   Y<-(X1+X2)/X3
+   return(Y)
+ }
> 
> for (i in 1:10) {
+   print(MyFunction(i,20,5))
+ }
[1] 4.2
[1] 4.4
[1] 4.6
[1] 4.8
[1] 5
[1] 5.2
[1] 5.4
[1] 5.6
[1] 5.8
[1] 6
> 
```

```
1 MyFunction<-function(X1,X2,X3) {
2   Y<-(X1+X2)/X3
3   return(Y)
4 }
5 
6 for (i in 1:10) {
7 
8   print(MyFunction(i,20,5))
9 
10 }
11 
```



~ Recherche

```
> MyFunction<-function(X1,X2,X3) {  
+   Y<-(X1+X2)/X3  
+   return(Y)  
+ }
```

```
> X<-matrix(nrow=10,ncol=1)
```

```
> X[,1]<-1:10
```

```
> apply(X,2,MyFunction,20,5)
```

```
      [,1]  
[1,] 4.2  
[2,] 4.4  
[3,] 4.6  
[4,] 4.8  
[5,] 5.0  
[6,] 5.2  
[7,] 5.4  
[8,] 5.6  
[9,] 5.8  
[10,] 6.0
```

```
>
```

```
>
```



```
1 MyFunction<-function(X1,X2,X3) {  
2   Y<-(X1+X2)/X3  
3   return(Y)  
4   }
```

```
5  
6 X<-matrix(nrow=10,ncol=1)
```

```
7 X[,1]<-1:10
```

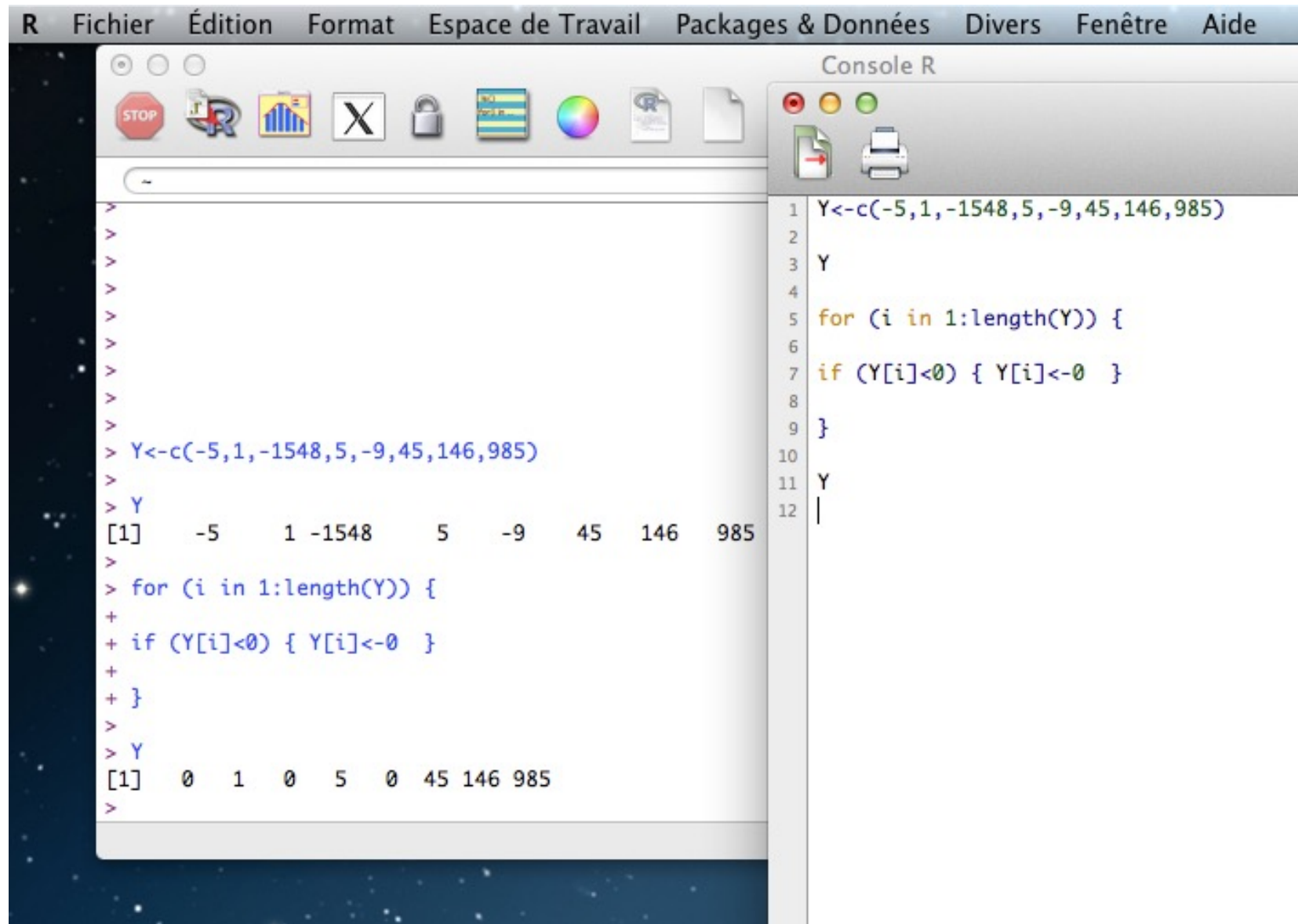
```
8
```

```
9 apply(X,2,MyFunction,20,5)
```

```
10
```

```
11 |
```

# conditions: if (test) {}



The screenshot shows the R console interface with a menu bar (R, Fichier, Édition, Format, Espace de Travail, Packages & Données, Divers, Fenêtre, Aide) and a toolbar. The main editor area on the left contains the following R code:

```
>  
>  
>  
>  
>  
>  
>  
>  
>  
>  
> Y<-c(-5,1,-1548,5,-9,45,146,985)  
>  
> Y  
[1] -5 1 -1548 5 -9 45 146 985  
>  
> for (i in 1:length(Y)) {  
+ if (Y[i]<0) { Y[i]<-0 }  
+ }  
>  
> Y  
[1] 0 1 0 5 0 45 146 985  
>
```

The console output on the right shows the execution of the code:

```
1 Y<-c(-5,1,-1548,5,-9,45,146,985)  
2  
3 Y  
4  
5 for (i in 1:length(Y)) {  
6  
7 if (Y[i]<0) { Y[i]<-0 }  
8  
9 }  
10  
11 Y  
12 |
```

R   Fichier   Édition   Format   Espace de Travail   Packages & Données   Divers   Fenêtre   Aide



```
>
>
>
>
>
>
>
>
>
>
> Y<-c(-5,1,-1548,5,-9,45,146,985)
>
> Y
[1] -5 1 -1548 5 -9 45 146 985
>
> Y[Y<0]<-0
>
> Y
[1] 0 1 0 5 0 45 146 985
>
>
```

```
1 Y<-c(-5,1,-1548,5,-9,45,146,985)
2
3 Y|
4
5 Y[Y<0]<-0
6
7 Y
8
```

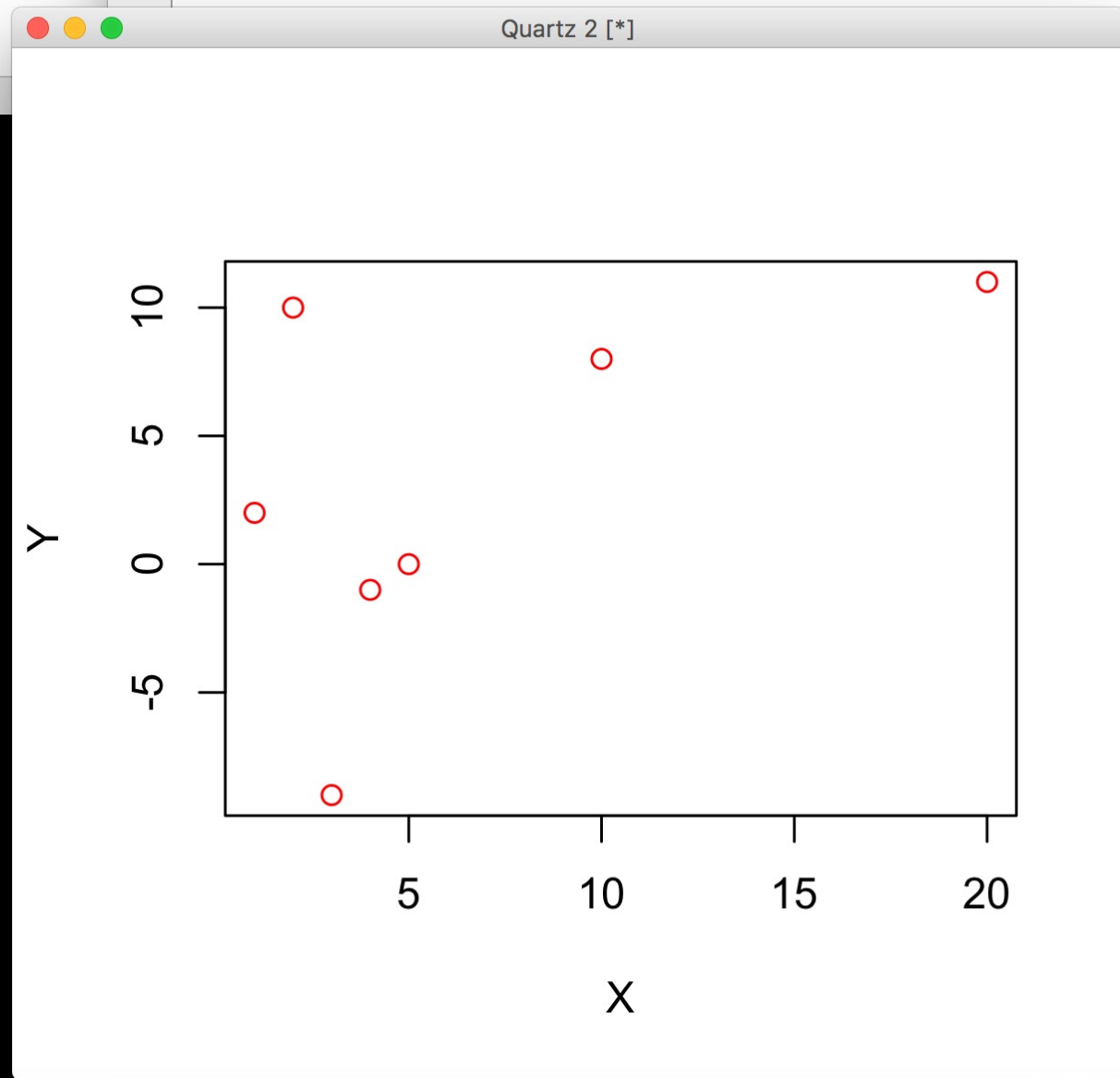


# graphics

- Functions for building a new graph  
plot, hist, barplot, etc.
- Functions to add new components to an existing graph  
lines, points, text, abline etc.

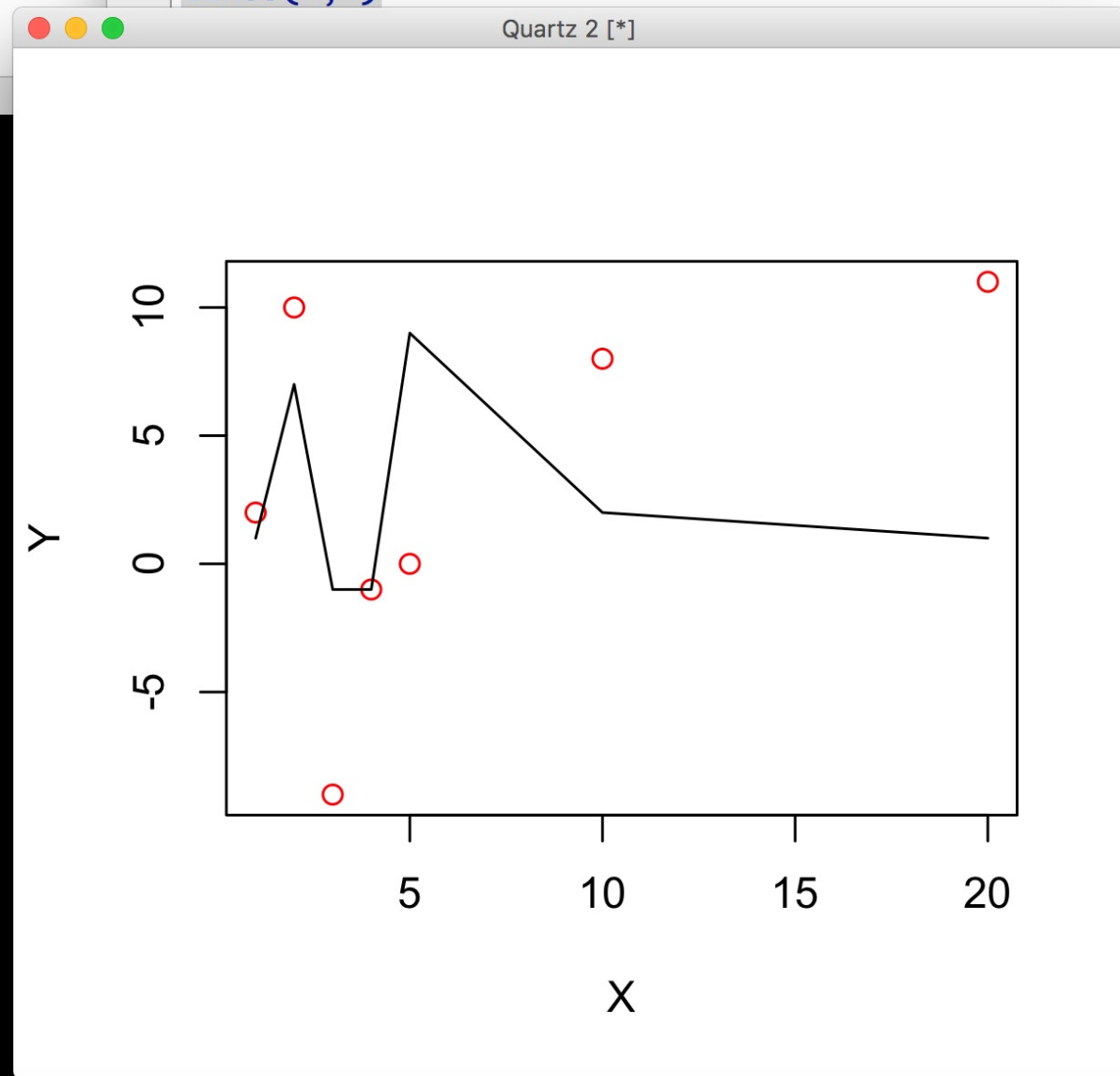
```
~  
>  
>  
>  
> X<-c(1,2,3,4,5,10,20)  
> Y<-c(2,10,-9,-1,0,8,11)  
> plot(X,Y, col="red")  
>
```

```
1 X<-c(1,2,3,4,5,10,20)  
2 Y<-c(2,10,-9,-1,0,8,11)  
3 plot(X,Y, col="red")
```



```
> X<-c(1,2,3,4,5,10,20)
> Y<-c(2,10,-9,-1,0,8,11)
> plot(X,Y, col="red")
> Z<-c(1,7,-1,-1,9,2,1)
> lines(X,Z)
>
```

```
1 X<-c(1,2,3,4,5,10,20)
2 Y<-c(2,10,-9,-1,0,8,11)
3 plot(X,Y, col="red")
4 Z<-c(1,7,-1,-1,9,2,1)
5 lines(X,Z)
```



# Simple mathematical functions

- - + / \*
- sum()
- ^2, ^3,...
- log(), exp()
- mean()
- median()
- quantile()
- min(), max()

```
> X<-c(1,2,3,4,5,6,7,8,9,10,11)
```

```
> sum(X)
```

```
[1] 66
```

```
> mean(X)
```

```
[1] 6
```

```
> median(X)
```

```
[1] 6
```

```
> quantile(X,0.25)
```

```
25%
```

```
3.5
```

```
> log(X)
```

```
[1] 0.0000000 0.6931472 1.0986123 1.3862944 1.6094379 1.7917595 1.9459101 2.0794415 2.1972246 2.3025851 2.3978953
```

```
> exp(X)
```

```
[1] 2.718282 7.389056 20.085537 54.598150 148.413159 403.428793 1096.633158 2980.957987 8103.083928  
22026.465795 59874.141715
```

```
> X^3
```

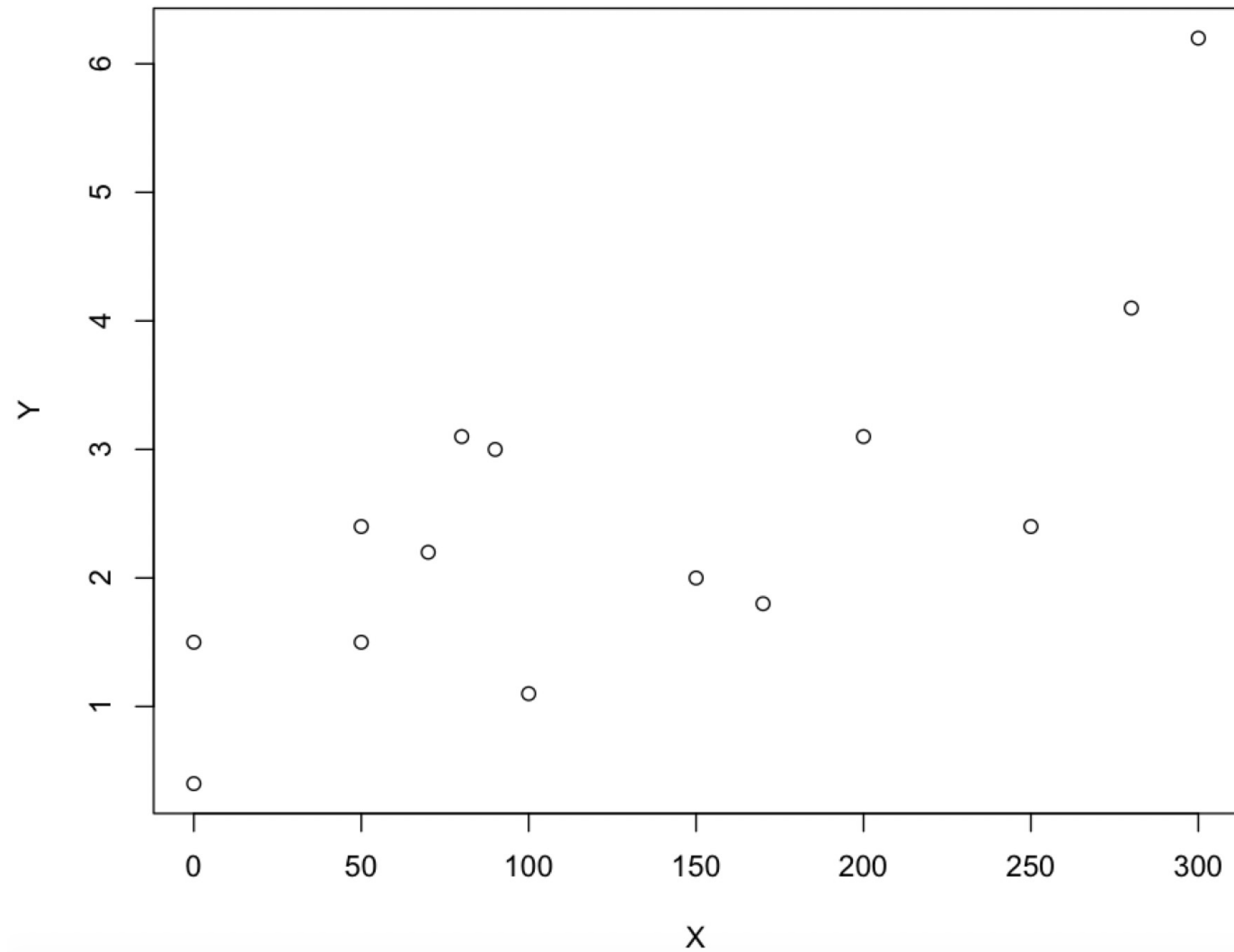
```
[1] 1 8 27 64 125 216 343 512 729 1000 1331
```

```
> X*10
```

```
[1] 10 20 30 40 50 60 70 80 90 100 110
```

```
>
```

# Linear regression



```
#Data
```

```
X<-c(0,250,100,50,70,170,300,50,80,90,0,280,200,150)
```

```
Y<-c(1.5,2.4,1.1,1.5,2.2,1.8,6.2,2.4,3.1,3.0,0.4,4.1,3.1,2)
```

```
#Graphic
```

```
plot(X,Y)
```

```
#Regression
```

```
LinReg<-lm(Y~X)
```

```
summary(LinReg)
```

```
#Regression line
```

```
D<-1:300
```

```
Fit<-1.12+0.0106*D
```

```
lines(D,Fit,col="red",lwd=2)
```

```
> summary(LinReg)
```

Call:

```
lm(formula = Y ~ X)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-1.38665	-0.72333	-0.08014	0.65167	1.88080

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	1.123915	0.445230	2.524	0.02670 *
X	0.010651	0.002789	3.818	0.00245 **

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9973 on 12 degrees of freedom

Multiple R-squared: 0.5485, Adjusted R-squared: 0.5109

F-statistic: 14.58 on 1 and 12 DF, p-value: 0.002446

