R 语言基础

R语言是一门常用于数据分析、统计建模的计算机语言，它与主流的C/C++、Java、Python等语言相比，支持更多的数据类型，例如向量、矩阵，同时提供了多种统计和数学计算方法。

可以前往 https://www.r-project.org/ 下载R语言解释器，并且推荐使用 RStudio 这个R语言的集成开发环境。RStudio 可以在 https://www.rstudio.com/ 下载。

四则运算

R语言的四则运算与大部分语言相同，使用 `+`, `-`, `*`, `/`, `^` 来表示加、减、乘、除和乘方。数值可以写成 `123`, `-123`, `123.45`, `1.23E-5` 这样的形式。其中 `1.23E-5` 表示 1.23×10^{-5} 。

字符串

用单引号 `'` 或双引号 `"` 包裹起来的文字内容为字符串，如 `'hello, world'` 或 `'123456'`。

向量

R 语言中可以通过 `c(...)` 来声明一个向量，例如 $\boldsymbol{x} = (1, 2, 3, 4, 5)$ 可以通过 `x <- c(1, 2, 3, 4, 5)` 来声明。

对两个长度相等的向量进行四则运算的效果是向量中的每一个元素都与另一个向量中的每一个元素进行四则运算。而一个向量与一个数进行四则运算的效果是该向量中的所有元素都与这个数进行四则运算。如：

```
> x <- c(1, 2, 3, 4, 5)
> y <- c(1, 2, 3, 4, 5)
> x + y # 输出: [1] 2 4 6 8 10
> x + 1 # 输出: [1] 2 3 4 5 6
```

矩阵

R 语言中可以通过 `matrix()` 函数来创建矩阵。`matrix()` 的原型为 `matrix(data=NA, nrow=1, ncol=1, byrow=FALSE, dinames=NULL)` 其中：

- `data`：矩阵的元素，通常为向量。
- `nrow` 和 `ncol`：设定矩阵的行数和列数，一般只需设定其一，另一个会根据数据长度算出。
- `byrow`：设定矩阵的填充方式，值为 `TRUE` 时按行填充。默认为 `FALSE`，即按列填充。

通过 `matrix()` 创建矩阵的例子如下：

```
> mat1 <- matrix(1:6, nrow=2) # 元素为1到6， 两行，按列填充
> mat1
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6

> mat2 <- matrix(1:6, nrow=2, byrow=TRUE)
> mat2
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
```

此外还可以使用 `dim()` 通过向量来创建矩阵，例如：

```
> mat3 <- 1:6
> dim(mat3) <- c(3, 2) # 将元素为1到6变为3行2列的矩阵
> mat3
     [,1] [,2]
[1,]    1    4
[2,]    2    5
[3,]    3    6
```

生成时间序列

通过 `ts()` 函数可以将一个向量或矩阵转成一个一元或多元的时间序列（ts）对象，`ts()` 函数的原型为：`ts(data = NA, start = 1, end = numeric(0), frequency = 1, deltat = 1, ts.eps = getOption("ts.eps"), class, names)`。其中：

- `data` 要生成时间序列的向量或矩阵。
- `start` 第一个观测值的时间。
- `end` 最后一个观测值的时间。
- `frequency` 单位时间内观测值的频数。
- `deltat` 两个观测值之间的时间间隔。
- `ts.eps` 序列之间的误差限。若序列之间的频率差异小于 `ts.eps` 则认为这些序列的频率相等。
- `class` 对象的类型。一元序列默认为 `ts`，多元序列默认为 `c("mts", "ts")`。
- `names` 给出多元序列中每个一元序列的名称，默认为 `Series 1, Series 2, ...`。

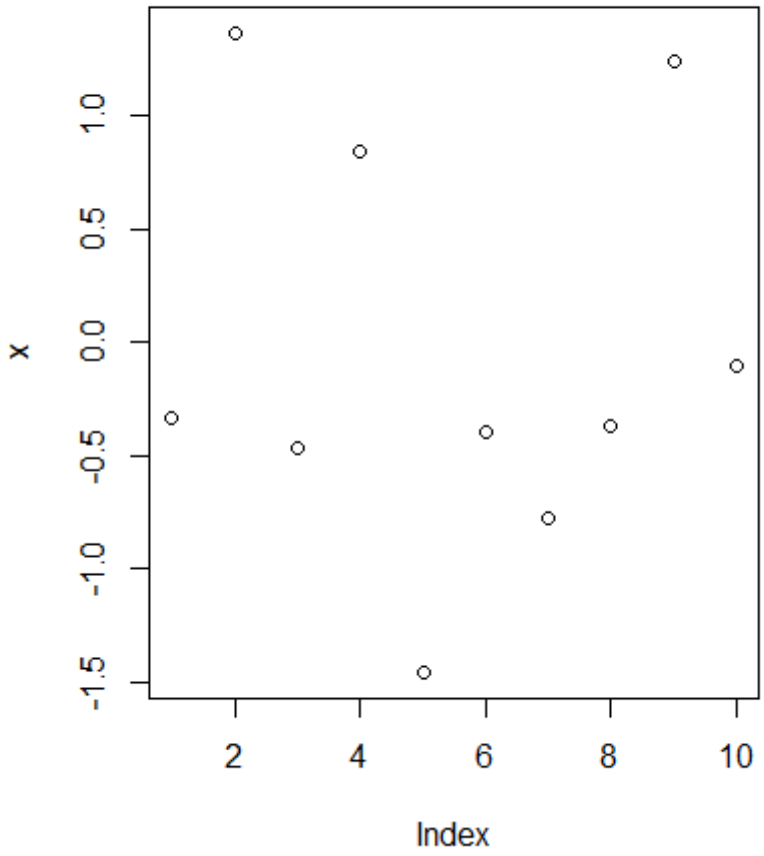
下面是一个例子：

```
> ts(1:26, start = 1986)
Time Series:
Start = 1986
End = 2011
Frequency = 1
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17
[18] 18 19 20 21 22 23 24 25 26
```

画图

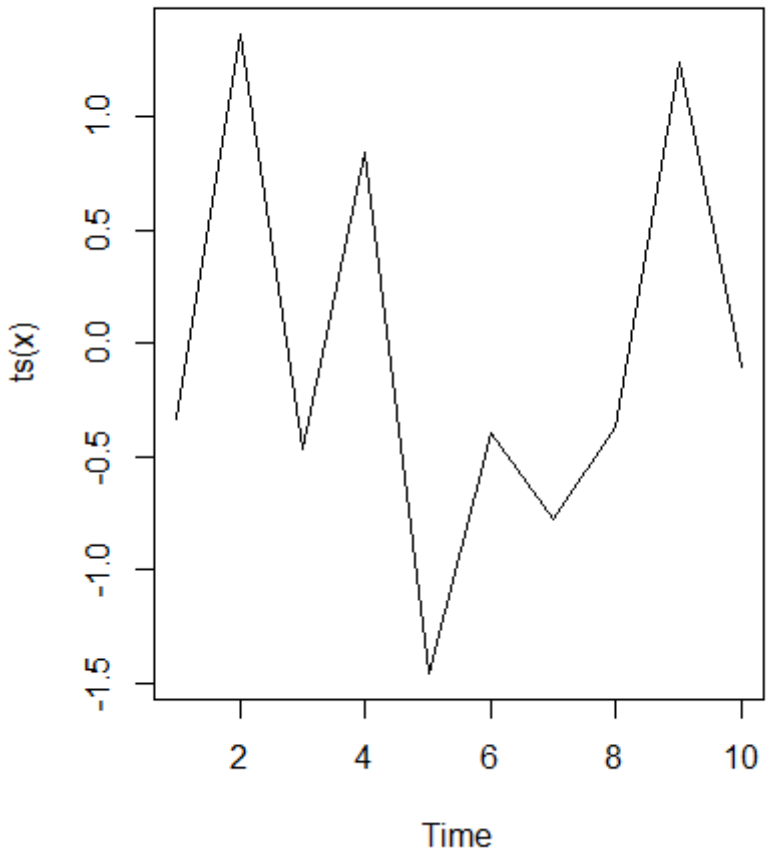
R 语言中可以直接使用 `plot()` 函数来绘制图像，例如：

```
> x <- rnorm(10)
> plot(x)
> x
 [1] -0.3329234  1.3631137 -0.4691473  0.8428756
 [5] -1.4579937 -0.4003059 -0.7764173 -0.3692965
 [9]  1.2401015 -0.1074338
```



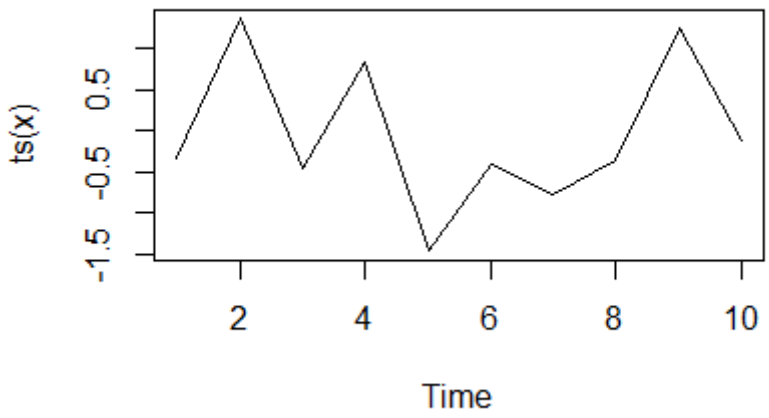
还可以直接绘制时间序列的图像：

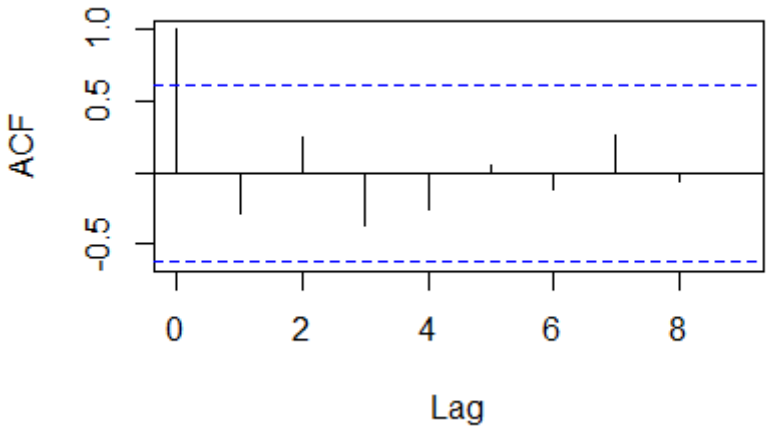
```
> plot(ts(x))
```



为了将多幅图画在一起，可以使用 `par()` 函数：

```
> op <- par(mfrow=c(2, 1), mar=c(5, 4, 2, 2) + .1) # mfrow 指定了图像矩阵为两行一列，即画两幅图，每行一幅；mar 指定了图像的边界，分别是下、左、上、右，可以根据喜好指定
> plot(ts(x))
> acf(z, main = "") # 计算 acf 函数
```


ts(x)
-1.5
-0.5
0.5
2
4
6
8
10
Time


ACF
-0.5
0.5
1.0
0
2
4
6
8
Lag