Introduction to Statistics and R

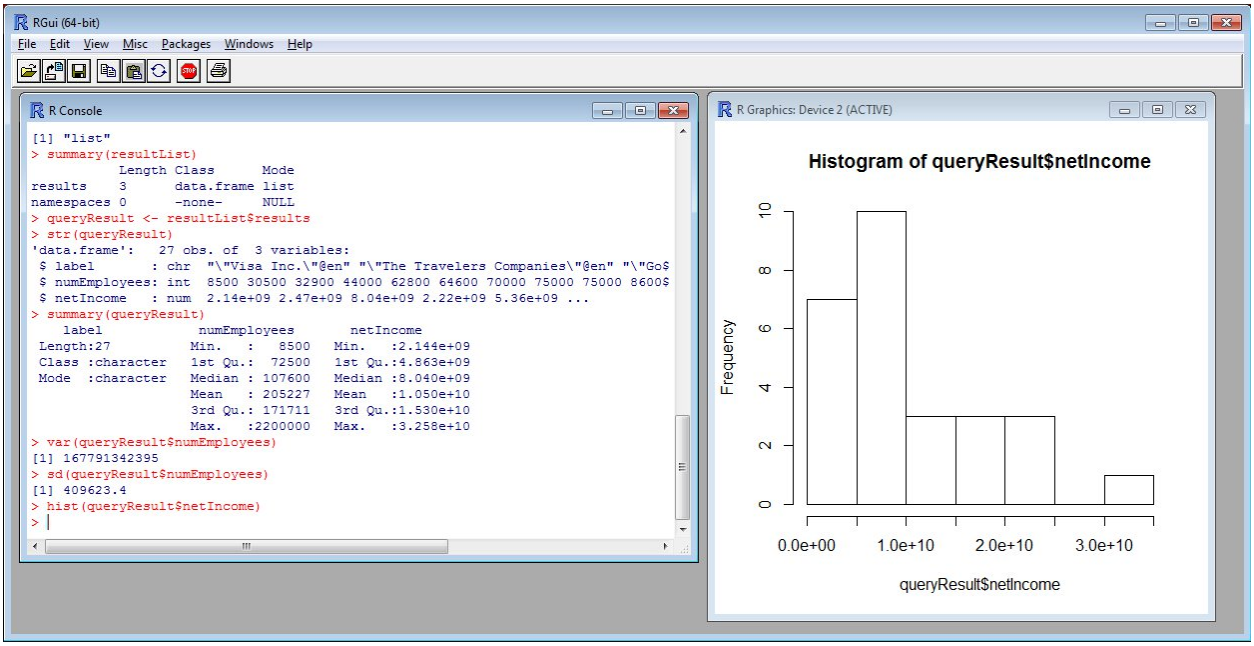
R Crash Course

Eric Stemmler

10.02.2021

Contents

1 What is R?



R бол хэл бөгөөд тухайн хэлний чөлөөт, нээлттэй эх сурвалж R орчуулагч ч юм. R нь олон зүйлийг хийх боломжийг олгодог програмчлалын хэл юм. Үүнийг анхандаа статистик тооцоолол, өгөгдлийг визуалчлах/дүрслэх зорилгоор боловсруулсан болно. Гэсэн хэдий ч өнөөдөр R-ийн чадварууд нь дүн шинжилгээ хийх, вэб дизайн хийх, их хэмжээний өгөгдөл боловсруулах, онлайн хяналт хийх самбарыг бий болгох гэх мэт олон зүйлийг хамарч байна. Тиймээс R-ийг судлаач хүний хувьд сурч мэдэх нь маш чухал юм Хэрэв та Windows үйлдлийн систем дээр R-ийг эхлүүлбэл танд иймэрхүү зүйл харагдах болно. Таны харж байгаа зүйл бол R Gui /график хэрэглэгчийн интерфэйс гэсэн үгний товчлол/ юм. Цэсний бааранаас гадна энэ нь үндсэндээ консолоос бүрдэнэ. Консол нь командыг шууд R руу илгээхэд ашиглагддаг. Дараагийн слайдууд дээр би та бүхэнд янз бүрийн командуудыг танилцуулах болно. Илүү практик туршлагатай болохын тулд тэдгээрийг R консол дээрээ шууд бичиж оруулаад дагаад хийгээд явна уу.

2 R Commands

```
3+4*12
```

```
## [1] 51
```

- Place cursor into console
- Write command
- Press ↵
- R interprets the command and returns it's computes output
- Repeat previous command: ↑

За ингээд энгийн тооцооллыг команд болгож үзье. Курсорыг консол дээр байрлуулаад командыг оруулаад enter товчийг дарахад л хангалттай. Хэрэв та enter товчийг дарвал команд R орчуулагч руу илгээгдэж, үр дүн нь консол дээр дахин гарч ирнэ. Хэрэв бид хэд хэдэн командуудыг дараалан илгээсэн бөгөөд хэсэг хугацааны дараа өмнөх командыг давтахыг хүсч байвал дээшээ заасан суман товчийг дарж өмнөх бүх командыг үзэх боломжтой.

```
a <- 3+4*12
a
```

```
## [1] 51
```

Ихэвчлэн тооцооллын үр дүнгүүдийг хадгалах боломжтой хувьсагчуудтай ажиллах нь илүү амар байдаг. "a" хувьсагчийг тодорхойлж, утгыг оноохын тулд хувьсагч тохируулах оператор болох "-аас бага" болон "хасах" тэмдэг/ <- ийг бичнэ. Бид a хувьсагчийг тохируулсны дараа a-ийн утгыг хэвлээд амжилттай болсон эсэхээ мэдэж болно. Бид үүнийг консол дээр "a" гэж бичээд л амархан хийж болно.

```
b <- c(1, 2, 3, 4, 5)
d <- 1:5
b
```

```
## [1] 1 2 3 4 5
```

```
b == d
```

```
## [1] TRUE TRUE TRUE TRUE TRUE
```

```
a*b
```

```
## [1]  51 102 153 204 255
```

```
a^b
```

```
## [1]        51      2601    132651   6765201 345025251
```

Тэгэхээр дахиад хэдэн хувьсагчийг тодорхойлъё. Ерөнхийдөө R дахь тоон хувьсагчид нь үргэлж вектор хэлбэрээр хадгалагддаг. Хэрэв та хувьсагчийг дан ганц тоо гэж тодорхойлсон бол түүнийг бас

л вектор гэж үзнэ. Векторыг олон янзаар R-д хялбархан үүсгэж болно. Үүний нэг нь c() функцийг ашиглах явдал юм. c() нь нэгтгэх/concatenate/ гэсэн үгний товчлол юм. Тэгэхээр c() функц нь ердөө хаалтаар зааглагдсан оролтын параметрүүдийг аваад эдгээр параметрүүдийг агуулсан вектор үүсгэдэг. Ижил вектор үүсгэх өөр нэг арга бол тодорхойлох цэг/:/ийг ашиглах явдал юм. Эдгээр хоёр командын ялгаа нь эхнийх нь бүхэл тоо болон тоон утгуудын аль алинд нь ашиглагдах боломжтой бөгөөд дараагийнх нь зөвхөн дараалсан бүхэл тоон утгуудыг үүсгэдэг Хоёулаа тэнцүү эсэхийг харахын тулд R-ийн Boolean эсвэл логик операторуудын аль нэгийг ашиглан харьцуулж болно. Давхар тэнцүү тэмдэг/== нь хоёр объектыг харьцуулдаг бөгөөд тэдгээрийн утга нь тэнцүү бол TRUE, ялгаатай бол FALSE, эсвэл утга нь аль нэг нь боломжгүй байвал NA -ийг гаргаж ирдэг. Гэсэн хэдий ч b ба d нь ижил утгуудтай боловч гарч ирэх харьцуулалтын үр дүн нь b ба d-тэй ижил тоотой вектор байх бөгөөд векторын бүрэлдэхүүн хэсгүүдийн харьцуулалт бүрийн үр дүнг хэлж өгдөг болохыг анхаарна уу. Үүнийг векторжилт гэнэ. Векторжилт нь бусад үйлдлүүдэд бас ажилладаг бөгөөд тухайн үйлдэл нь векторийн бүрэлдэхүүн хэсэг тус бүрээр автоматаар хийгддэг.

3 Working Principles

- R commands can be send via console or script
- Console:
  - Experiment with commands
  - R documentation: e.g. `?range`
  - Look up history of commands (↑ or ↓)
  - Installing packages: `install.packages("ggplot2")`
- R-Scripts: File » New Script
  - Make your analysis shareable and replicable
  - Documentation of analysis
  - Using comments: `\#here is a comment`
  - Saving analysis: File » Save or Ctrl + s and save as `*.R`

R-ийг консолоор нь дамжин хэрхэн ашиглаж болохыг бид мэдэж авлаа. Тооцоолол хийх өөр нэг арга бол script-д комманд бичих явдал юм. Консол руу командыг шууд бичиж оруулах нь шинэ команд туршиж үзэх, баримт бичгийг хайж олох, командын түүхийг үзэх эсвэл шинэ команд өгдөг R багцуудыг суулгах боломжтой гэдгээрээ давуу талтай юм. Гэсэн хэдий ч та R-д ажиллах ихэнх ажлынхаа командийг R-script-д бичихийг уриалмаар байна. Та File цэсний New Script-ийг дарж шинэ R script нээх боломжтой. Script editor нь консолын хажууд гарч ирэх бөгөөд та R script бичиж эхлэх боломжтой болно Дүн шинжилгээний алхам бүрийг агуулсан script-ийг ашигласнаар та бусдад анализаа давтах боломжийг олгоно.

3.1 Using the RGui

- Chose clean windowing layout: Windows » Tile Vertically
- Select code with cursor and press Ctrl + R executes the code
- . . . or right-click on selection and chose “Run line or selection”
- Clear console: Ctrl + L

R хэрэглэгчийн интерфэйстэй ажиллах өөр хэдэн зөвлөмжийг энд авч үзье. Хэрэв та консол болон script хоёуланг нь ажиллуулж байгаа бол хоёр цонхыг цэгцтэй байдлаар байрлуулахыг хүсч магадгүй юм. Үүнийг хийхийн тулд та Windows цэсний Tile Vertically-ыг сонгож, хоёр цонхыг зэргэлдээ жигд байрлуулна. Хэрэв таны курсор editor дээр байгаа бол ctrl + R товчийг дарж одоогийн мөрний командыг гүйцэтгэнэ. Олон тооны мөрний кодыг гүйцэтгэхийн тулд та олон мөрийг сонгож идэвхжүүлээд дараа нь ctrl +R товчийг дарах эсвэл баруун товчийг дарж эхний сонголтыг сонгож гүйцэтгэнэ. Хэрэв таны консол дүүрсэн, жаахан эмх замбараагүй байгаа бол Ctrl + L товчийг дарж арилгаж болно.

4 R Data Types

```
s <- "Hello World!"
s
```

```
## [1] "Hello World!"
```

```
class(s)
```

```
## [1] "character"
```

R дээр үргэлжлүүлэн янз бүрийн өгөгдлийн төрөлтэй ажиллацгаая. Дээрх эхний командуудад бидний ашиглаж байсан хувьсагчууд нь тоон өгөгдөл байсан. R-д өөр өөр төрлийн өгөгдөл байдаг, жишээлбэл, та "character" гэсэн өгөгдлийн төрлөөр бичсэн текстийг харуулж болно. Бидэнд Hello World гэсэн хоёр үгийг агуулсан тэмдэгт векторыг хадгалсан s нэртэй хувьсагч байна. Бид s нэрийг бичих замаар s-ийн утгыг дахин харуулж болно. s-ийн өгөгдлийн төрлийг нь мэдэхийн тулд R-ийн class() функцийг ашиглаж болно. Энэ тохиолдолд бид s нь жинхэнэ character төрөл болохыг баталгаажуулна.

```
a
```

```
## [1] 51
```

```
class(a)
```

```
## [1] "numeric"
```

R нь бидний анхны a хувьсагчийн өгөгдлийн төрлийн талаар юу хэлэхийг харцгаая.

```
m <- matrix(data = c(1, 2, 3, 4),
        nrow = 2,
        ncol = 2)
m
```

```
##      [,1] [,2]
## [1,]    1    3
## [2,]    2    4
```

```
class(m)
```

```
## [1] "matrix"
```

Шугаман алгебр ашиглан тооцоо хийхдээ матриц ашиглахыг хүсч магадгүй юм. R-д матрицыг матрицын элементүүдийн утгыг өгч, матрицын хэмжээсийг мөр, баганын тоогоор тодорхойлдог matrix() функц-ийг дуудаж үүсгэж болно. Дахин хэлэхэд бид матриц m-ийн өгөгдлийн төрлийг class () -ийг дуудаж баталгаажуулж болно.

```
v <- c(2, 2)
m %*% v
```

```
##      [,1]
## [1,]    8
## [2,]   12
```

Хэрэв бид скаляр үржвэрээс өөр матрицыг m-ээр үржүүлэхийг хүсч байвал матрицын үржүүлэх оператор/ %*%/ыг ашигладаг. Жишээ болгон бид тохирох 2 тоотой вектор v-ыг тодорхойлоод дараа нь m-ээр үржүүлнэ.

```
a <- runif(15)
summary(a)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.03921 0.47017 0.59317 0.58597 0.73041 0.95299
```

Статистик програмчлалын өөр нэг ойлголт бол санамсаргүй тоо үүсгэгч ашиглах явдал юм. R-д хэд хэдэн суурилуулсан санамсаргүй тоо үүсгэгчид байдаг бөгөөд эдгээрийг мэдэгдэж байгаа тархалтаас үүсэх тооны дарааллыг үүсгэхэд ашиглаж болно. Ихэнхдээ эдгээр санамсаргүй тооны үүсгүүрүүд нь нэр нь "r" үсгээр эхэлсэн функцууд юм. Энэ жишээнд бид жигд тархсан 15 тоог үүсгэх ба тус бүр нь 0-ээс 1-ийн хооронд гарах тэнцүү магадлалтай 15 тоог гаргаж, үр дүнг a хувьсагчид хадгална. Энэ командын өмнө бид a-г аль хэдийн тодорхойлсон тул a-ийн өмнөх утгыг зүгээр л дарж бичдэг. A-д агуулагдсан өгөгдлийг шалгах өөр нэг үр дүнтэй арга бол R-ийн summary() функцийг дуудах явдал юм. Summary нь нэршлийнхээ дагуу a хувьсагч дахь утгуудын талаарх товч хураангуй статистикийг хэвлэн гаргаж ирдэг. Хэрэв энэ нь тоон хувьсагч бол бидний өмнөх хичээл дээр авч үзсэн хамгийн бага, хамгийн их, дундаж, голч, мөн 1, 3-р квартилуудыг энэ нь бэлтгэж өгөх болно.

```
length(a)
```

```
## [1] 15
```

Хэрэв та R объект дээр length() функцийг дуудвал энэ нь вектор байх тохиолдолд вектор доторх элементүүдийн тоог гаргана. Length() нь бусад объектуудад, жишээлбэл матрицад бас ажилладаг боловч үр дүн нь өөр утгатай байж болно.

```
violation <- a > 0.8
```

```
class(violation)
```

```
## [1] "logical"
```

```
summary(violation)
```

```
##    Mode   FALSE    TRUE
## logical      12       3
```

```
mean(violation)
```

```
## [1] 0.2
```

A-д агуулагдаж буй санамсаргүй байдлаар үүсгэсэн өгөгдлүүдтэй цаашаа ажиллацгаая. A нь голын усны 15 дээжийн хэмжилтийг агуулж байна гэж үзье. Жишээлбэл, бид дээж тус бүрийн фосфорын концентрацийг хэмжсэн гэж үзье. Зарим улс оронд усан сангаас авах фосфорын зөвшөөрөгдсөн хэмжээг хориглодог. Фосфорын хувьд 0.8 гэсэн хууль ёсны босго хэмжээ байна гэж үзье. Энэ босго хэмжээ нь хэтэрсэн дээжийг бид ердөө 0.8-аас их хэмжээтэй гэж бичээд үр дүнг нь violation гэж нэрлэдэг хувьсагчид хадгалах замаар тодорхойлж болно. R-ийн векторжилтын ачаар харьцуулах үйлдлийг а вектор дахь элемент тус бүрт автоматаар хийж байна. Хувьсагч violation-ний өгөгдлийн төрөл нь логик/“logical” юм. Энэ өгөгдлийн төрөл нь зөвхөн FALSE/ХУДАЛ ба TRUE/ҮНЭН гэж тэмдэглэгдсэн хоёртын өгөгдлийг хадгалах боломжтой. Энэ тохиолдолд TRUE/ҮНЭН нь тухайн дээж нь фосфорын босго хэмжээнээс хэтэрсэн гэсэн үг бөгөөд FALSE/ХУДАЛ нь тийм биш гэсэн үг юм. Хэрэв бид violation дээр summary функцийг дуудвал үр дүн нь өмнөхөөсөө өөр харагдаж байна. Мэдээжийн хэрэг, квартилийг зөвхөн хоёр боломжит үр дүнтэй өгөгдөл дээр тооцоолох нь утга учиртай юм. Тиймээс энэ хувьсагчийн summary нь илүү энгийн, хялбар байдаг. Гэсэн хэдий ч хэрэв бид violation хувьсагчийнхаа дундаж утгыг тооцоолох юм бол юу болох вэ? Бид тоон үр дүнг авах болно! R нь дотооддоо 0-ийг FALSE, 1-ийг TRUE гэж логик утгуудыг авч үздэг тул энэ нь болно. Тэгэхээр логик векторын дундажыг тооцоолох юм бол эхлээд энэ тохиолдолд 14-ийг 0-д үржүүлээд 0 гарах ба 1-ийг 1-д үржүүлээд 1 гаргаад нийлбэрийг тооцоолно. Тиймээс нийлбэрийн үр дүн нь 1 байна. Дараа нь үүнийг элементийн тоонд хуваах ба энэ тохиолдолд элементийн тоо нь 15 юм. Тиймээс үр дүн нь 1-ийг 15-д хуваана. Үүний үр дүн нь бас ач холбогдолтой юм, учир нь энэ нь дээжийн 7 орчим хувь болох босго хэмжээг давсан дээжийн хувийг бидэнд хэлж өгдөг.

```
my_list <- list(measurements = a,
        violations = a > 0.8,
        percentage = mean(a > 0.8))
```

```r
class(my_list)
```

```
## [1] "list"
```

R дахь өөр нэг өгөгдлийн төрөл бол list юм. List нь янз бүрийн тоотой хэд хэдэн хувьсагчдыг нэгтгэж нэг хувьсагч болгон бичих боломжтой цогц өгөгдлийн төрөл юм. list үүсгэхийн тулд бид R-ийн list() функцийг дуудаж, хэдэн тооны нэр болон хувьсагчийг параметр болгон бэлдэж өгөх боломжтой. Үр дүнгийн list-ийг бид шинэ хувьсагчид хадгалах бөгөөд үүнийг бид my_list гэж нэрлэдэг. Бид class()-ыг дуудаж өгөгдлийн төрлийг дахин баталгаажуулж, үнэхээр list үүсгэсэн болохыг харах боломжтой.

```r
# This is a comment: you can describe something here
my_list
```

```
## $measurements
##  [1] 0.89435281 0.76264325 0.03920702 0.59316960
##  [5] 0.20459158 0.95299083 0.56384652 0.67461965
##  [9] 0.55433377 0.94954836 0.69817750 0.48905335
## [13] 0.36196161 0.45128005 0.59977342
##
## $violations
##  [1]  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
##  [9] FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
##
## $percentage
## [1] 0.2
```

List-ийн агуулгыг бүхэлд нь хэвлэхийн тулд бид list-ийн нэрийг команд хэлбэрээр дахин бичиж болно.

```r
# You can access elements from a list by the $-operator
my_list$percentage
```

```
## [1] 0.2
```

```r
# ... or by enumeration
my_list[3]
```

```
## $percentage
## [1] 0.2
```

```r
# ... or by using names
my_list[["percentage"]]
```

```
## [1] 0.2
```

List-тай ажиллахыг хүсвэл түүний элементүүдэд тусгайлан хандах боломжтой байх шаардлагатай. Үүнийг хийх хэд хэдэн арга байдаг. Жишээлбэл, бид $ operator-ийг ашиглан жагсаалтын бие даасан элементэд хандах боломжтой. Мөн элементүүдийг үүсгэсэн дарааллаар нь авч дөрвөлжин хаалтанд өгөгдсөн индексээр нь хандаж болно. Ийм байдлаар list-ийн элементэд түүний нэрийг мэдэх шаардлагагүйгээр хандах боломжтой болно. Эсвэл бид элементэд нэрээр нь хандаж болох боловч үүнийг давхар дөрвөлжин хаалтанд тэмдэгт вектор болгон бэлтгэж өгөх боломжтой юм.

5 R Functions

R-ийн үндсэн инсталл нь тооцоолол хийх олон функц, бүтцийг агуулдаг. Жишээ нь: mean() функц Үүнээс гадна хэрэглэгч та өөрийн функцуудаа чөлөөтэй тодорхойлох боломжтой. Энэ нь жишээлбэл, кодын зарим хэсэг давтагдвал бичсэн кодын хэмжээг багасгах, эсвэл таны кодыг уншихад илүү хялбар болгоход тустай байж болох юм. Хувьсагчтай адилхан функцүүдийг суман операторын дараа түлхүүр үгийн функц, дараа нь хаалт () ашиглан үүсгэдэг. Хаалт дотор бид оролтын параметрүүдийг

тодорхойлж болох боловч заавал тэгэх шаардлагагүй юм. Дараа нь функцийн их биеийг зааглаасан хос долгионтой хаалт гарч ирнэ. Их биеийн дотор функцын үр дүнг функцын дуудагч руу буцааж гаргах зааврыг өгөхийн тулд return()-ийг дуудаж функцийг ихэвчлэн дуусгадаг.

```
greet <- function(n = 1) {
    return(paste0(rep("Hello!", times = n), collapse = " "))
}
```

```
greet()
```

```
## [1] "Hello!"
```

```
greet(4)
```

```
## [1] "Hello! Hello! Hello! Hello!"
```

Одоо бид ижилхэн функцийн илүү уян хатан хувилбарыг бичиж байна. Шинэ greet функц нь n параметрийг авах боломжтой. Функцийн тодорхойлолтон дахь n = 1 нь хэрэв функцийн дуудлагад ямар ч параметр өгөгдөхгүй бол n-ийг 1 болгож тохируулна гэсэн үг юм. Энэ нь ямар ч параметр бэлтгэж өгөхгүйгээр greet() функцийг дахин дуудаж ажилласныг бид харж байна. Хэрэв бид n параметрийн стандарт утгыг тодорхойлоогүй байсан бол энэ функцийн дуудлага нь R-аас дутуу параметр байна гэсэн алдааны мэдэгдлийг гаргахад хүргэх байсан. Энэ функцийн тодорхойлолт болон өмнөх хувилбар хоёрын хоорондын өөр нэг ялгаатай тал нь функцийг дуудах замаар илэрдэг: "Сайн уу!" гэсэн үгийг консол руу n удаа хэвлэнэ. Бидний функц нь өөр 2 функцийг дуудахаас хамаарна: paste0() ба rep(). Эдгээрийн утгыг өөрөө олж мэдэхийн тулд та R-ийн баримт бичгийг ашиглаж болно. Мөн түүнчлэн R-ийн баримт бичиг нь ихэвчлэн жишээнүүдийг агуулдаг. Эдгээр жишээг баримт бичиг дотор харж болох ба эсвэл R консол дээр example(...)-ийг дуудаж гүйцэтгэж болно.

6 Exercises

6.1 Using R as calculator

6.1.1 Circles

1. Calculate the area of a circle $A = 2\pi r^2$ with $r = 2$ (Qian, 2016)
2. Write the circle area formula as a function with named as `circle` with parameter `r` and and calculate the area for `r <- seq(0, 3, 0.1)`
3. **Extension:** Extend your function to return a named `list` that contains the area and the circumference for a given parameter vector `r`

6.1.2 Normal Probability Density Function

1. Calculate the density of the normal probability distribution function $x \sim \mathcal{N}(2, 1.25)$ (mean and standard deviation) at `x <- seq(0, 4, 0.5)` by using the normal probability density formula $\left(\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}\right)$, and verify your result by using the function `dnorm` (Qian, 2016).

6.2 Solutions

6.2.1 Circles

```
circle <- function(r) {
    result <- list(area = 2 * pi * r^2,
                   circumference = 2 * pi * r)
    return(result)
}
```

```r
circle(r = seq(0, 3, 0.5))
```

```
## $area
## [1]  0.000000  1.570796  6.283185 14.137167 25.132741
## [6] 39.269908 56.548668
##
## $circumference
## [1]  0.000000  3.141593  6.283185  9.424778 12.566371
## [6] 15.707963 18.849556
```

6.2.2 Normal Probability Density Function

```r
npdf <- function(x, avg, stdev) {
    result <- 1.0 / (sqrt(2*pi*stdev^2)) *
        exp(-(x - avg)^2 / (2.0 * stdev^2))
    return(result)
}

npdf(x = seq(0, 4, 0.5), avg = 2, stdev = 1.25)
```

```
## [1] 0.08873667 0.15534884 0.23175324 0.29461611
## [5] 0.31915382 0.29461611 0.23175324 0.15534884
## [9] 0.08873667
```

```r
    dnorm(x = seq(0, 4, 0.5), mean = 2, sd = 1.25)
```

```
## [1] 0.08873667 0.15534884 0.23175324 0.29461611
## [5] 0.31915382 0.29461611 0.23175324 0.15534884
## [9] 0.08873667
```

References

Song S Qian. *Environmental and ecological statistics with R*. CRC press, 2016.