

# R和Python简介

陈明杰

202410

# 为什么要学习编程语言？

- 生物信息学（Bioinformatics）是应用计算机科学和信息技术来管理、分析和解释生物数据的学科。随着生物技术的发展，生物数据的规模和复杂性都在不断增长，因此，掌握编程语言对于处理和分析这些数据变得至关重要。
- 学习R和Python可以帮助你更有效地处理和分析生物信息学数据，提高研究的效率和质量。此外，这些技能在学术界和工业界都是非常有价值的，可以为你的职业发展提供支持。

# R安装

The Comprehensive R Archive Network

## Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- [Download R for Linux \(Debian, Fedora/Redhat, Ubuntu\)](#)
- [Download R for macOS](#)
- [Download R for Windows](#)

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

R for Windows

<https://mirrors.e-ducation.cn/CRAN/>

约82M

Subdirectories:

[base](#)

Binaries for base distribution. This is what you want to **install R for the first time**.

[contrib](#)

Binaries of contributed CRAN packages (for R  $\geq$  4.0.x).

[old contrib](#)

Binaries of contributed CRAN packages for outdated versions of R (for R  $<$  4.0.x).

[Rtools](#)

Tools to build R and R packages. This is what you want to build your own packages on Windows, or to build R itself.

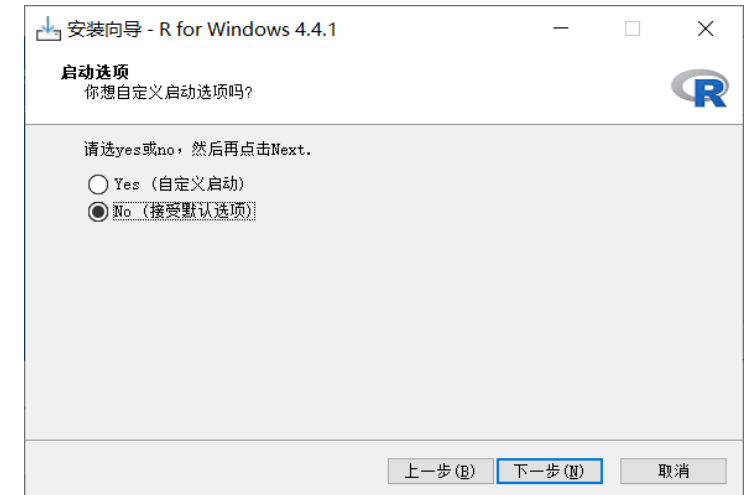
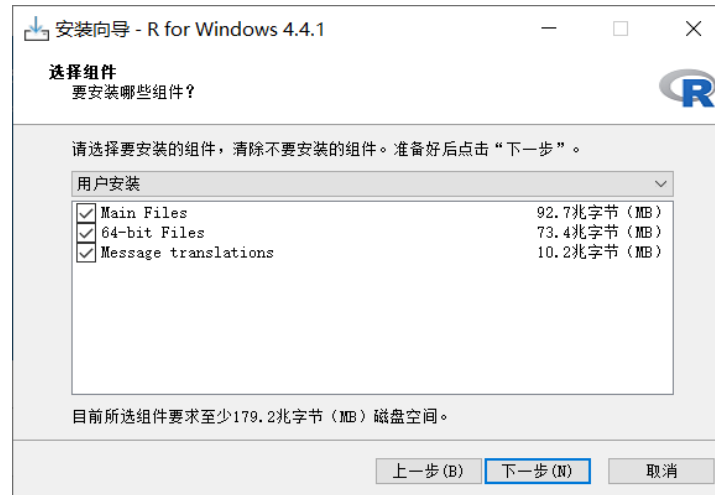
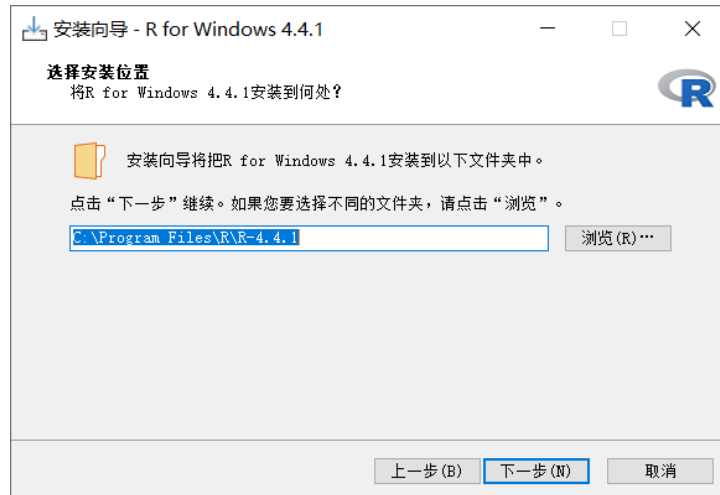
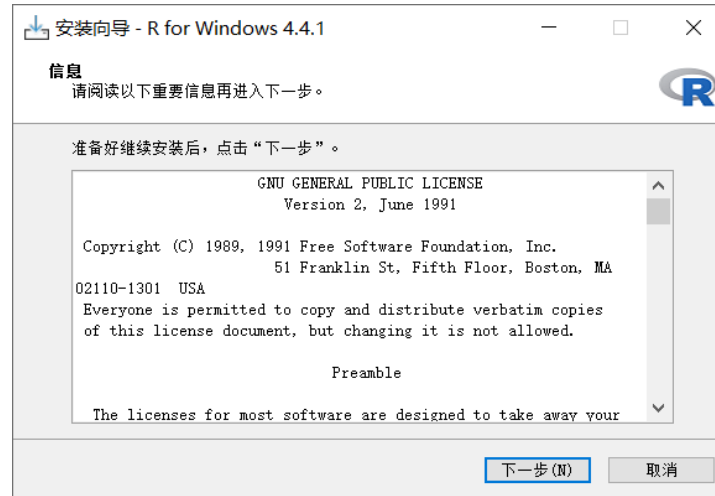
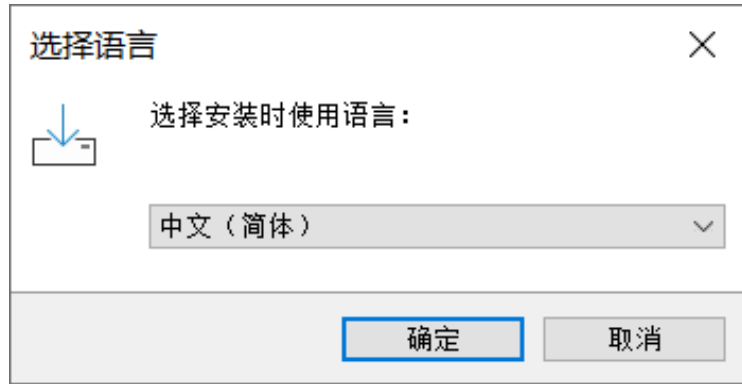
R-4.4.1 for Windows

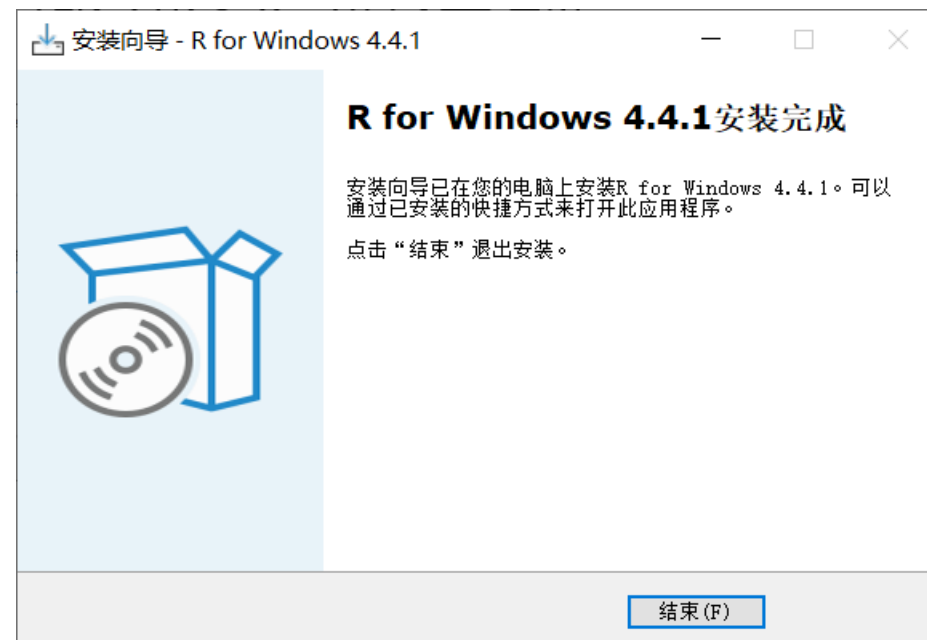
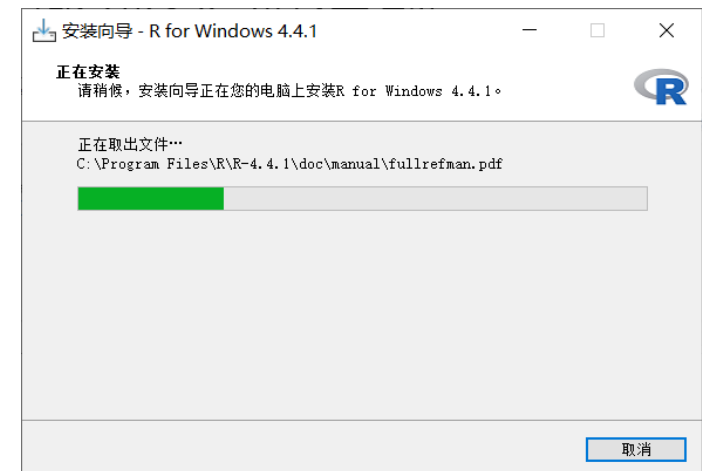
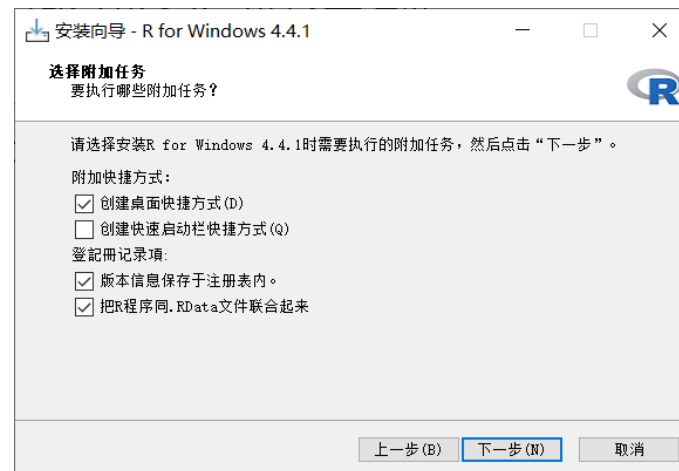
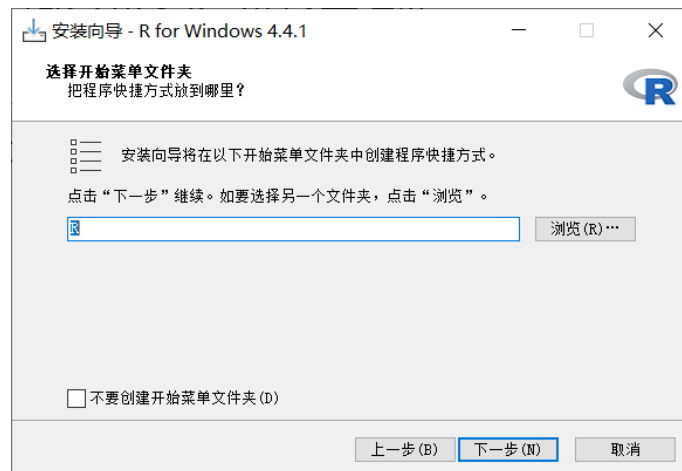
[Download R-4.4.1 for Windows](#) (82 megabytes, 64 bit)

[README on the Windows binary distribution](#)

[New features in this version](#)

This build requires UCRT, which is part of Windows since Windows 10 and Windows Server 2016. [here](#).





# Rstudio安装

posit [PRODUCTS](#) [SOLUTIONS](#) [LEARN & SUPPORT](#) [EXPLORE MORE](#) [PRICING](#)

## 1: Install R

RStudio requires R 3.6.0+. Choose a version of R that matches your computer's operating system.

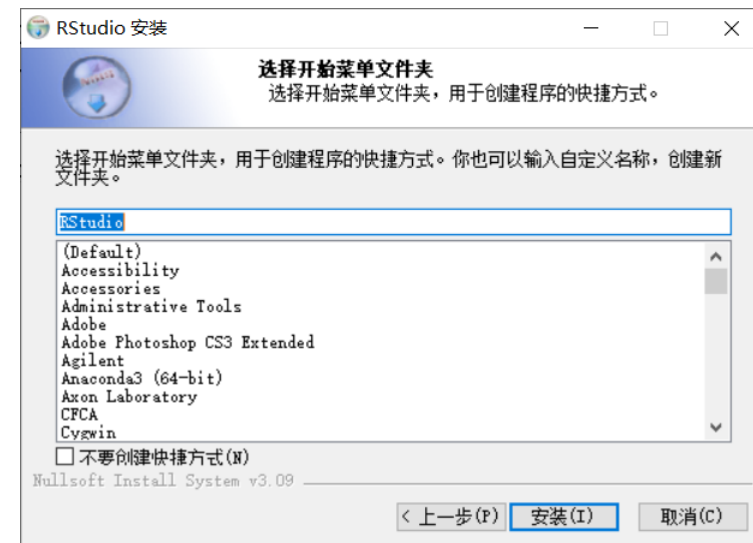
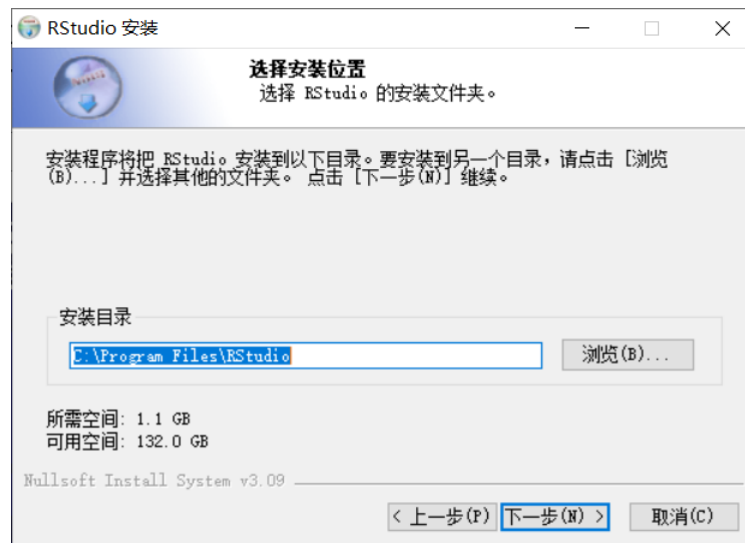
*R is not a Posit product. By clicking on the link below to download and install R, you are leaving the Posit website. Posit disclaims any obligations and all liability with respect to R and the R website.*

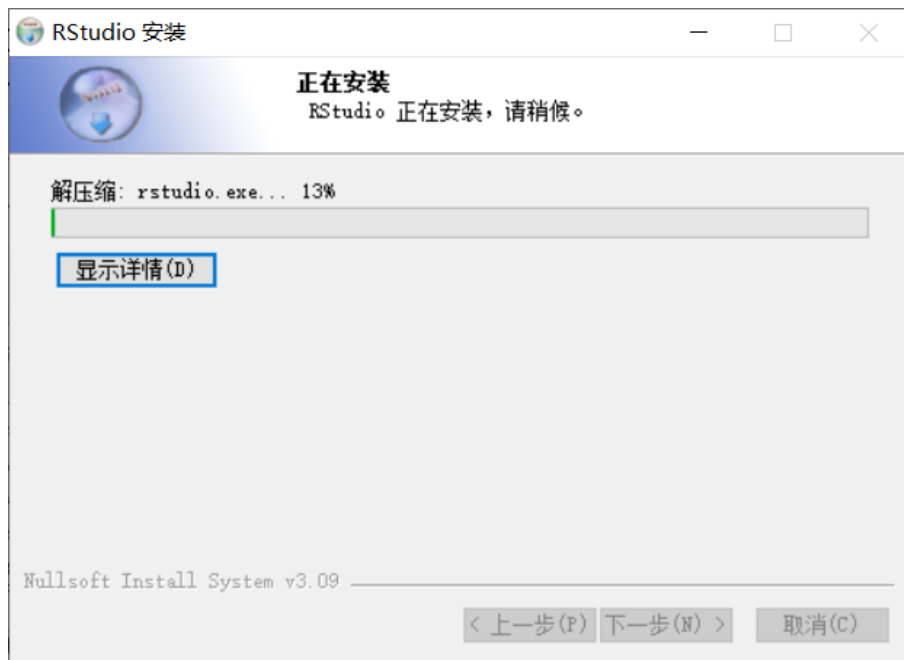
DOWNLOAD AND INSTALL R

## 2: Install RStudio

DOWNLOAD RSTUDIO DESKTOP FOR WINDOWS

Size: 262.79 MB | [SHA-256: 09E1E38A](#) | Version: 2024.04.2+764 | Released: 2024-06-10





```
11.R x  
1 |
```

写代码

Environment History Connections Tutorial  
Import Dataset 110 MiB  
R Global Environment  
Environment is empty  
查看变量

Console Terminal Background Jobs  
R 4.1.0 · ~/  
Copyright (C) 2021 The R Foundation for Statistical Computing  
Platform: x86\_64-w64-mingw32/x64 (64-bit)  
R是自由软件，不带任何担保。  
在某些条件下你可以将其自由散布。  
用'license()'或'licence()'来看散布的详细条件。  
R是个合作计划，有许多人人为之做出了贡献。  
用'contributors()'来看合作者的详细情况  
用'citation()'会告诉你如何在出版物中正确地引用R或R程序包。  
用'demo()'来看一些示范程序，用'help()'来阅读在线帮助文件，或  
用'help.start()'通过HTML浏览器来看帮助文件。  
用'q()'退出R。  
> |

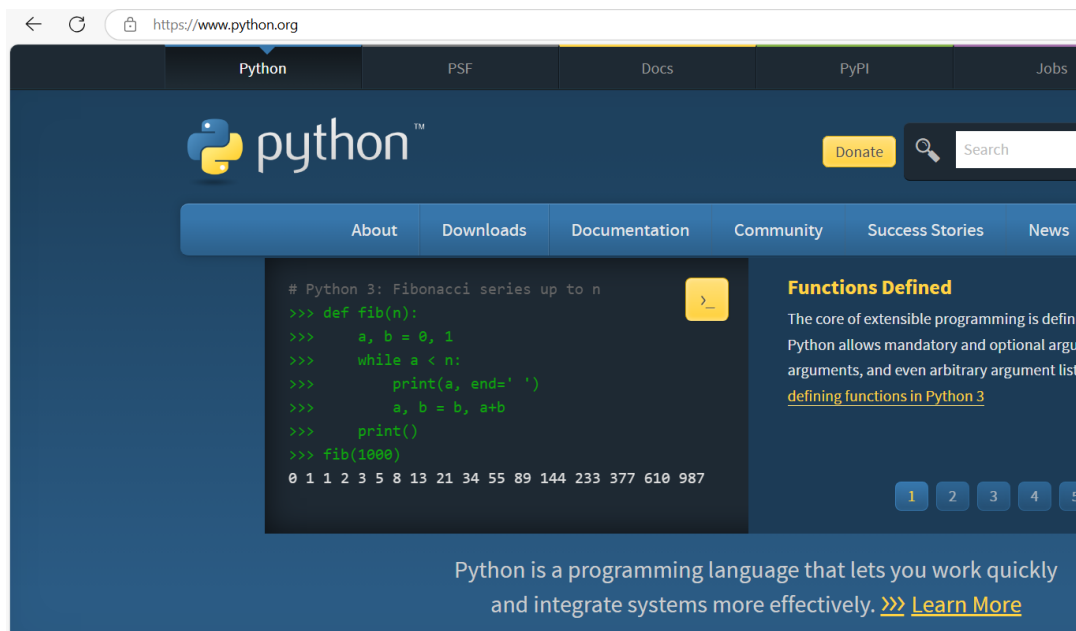
运行代码

Files Plots Packages Help Viewer Presentation  
New Folder New Blank File Delete Rename More  
Home  
Name Size Modified  
我的资源  
自定义 Office 模板  
apps  
Axure  
chinapubmed  
desktop.ini 402 B Jul 8, 2024, 9:24 PM  
Downloads  
HiSuite  
My eBooks  
My Music  
My Pictures  
My Videos  
NetFrameWork  
NetSarang Computer  
R  
rtc  
Sunlogin Files  
查看文件



# Python安装

<https://www.python.org/>



The screenshot shows the Python.org homepage. At the top, there is a navigation bar with links for Python, PSF, Docs, PyPI, and Jobs. Below this is a dark blue header with the Python logo and a search bar. A secondary navigation bar contains links for About, Downloads, Documentation, Community, Success Stories, and News. The main content area features a code editor with a Python script for calculating the Fibonacci series up to n. The script is as follows:

```
# Python 3: Fibonacci series up to n
>>> def fib(n):
>>>     a, b = 0, 1
>>>     while a < n:
>>>         print(a, end=' ')
>>>         a, b = b, a+b
>>>     print()
>>> fib(1000)
```

The output of the script is: 0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987. To the right of the code editor, there is a section titled "Functions Defined" which explains that the core of extensible programming is defining functions and allows for mandatory and optional arguments, and even arbitrary argument lists. It includes a link to "defining functions in Python 3". At the bottom of the page, there is a footer that says "Python is a programming language that lets you work quickly and integrate systems more effectively. >>> [Learn More](#)".

## Python Releases for Windows

- [Latest Python 3 Release - Python 3.12.7](#)

### Stable Releases

- [Python 3.12.7 - Oct. 1, 2024](#)

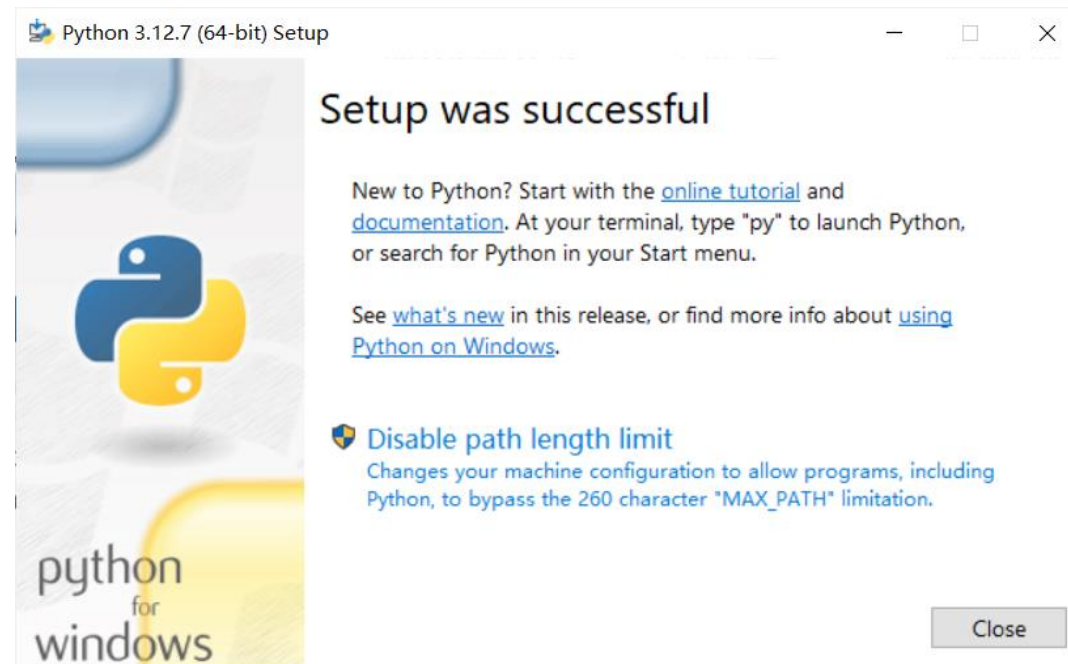
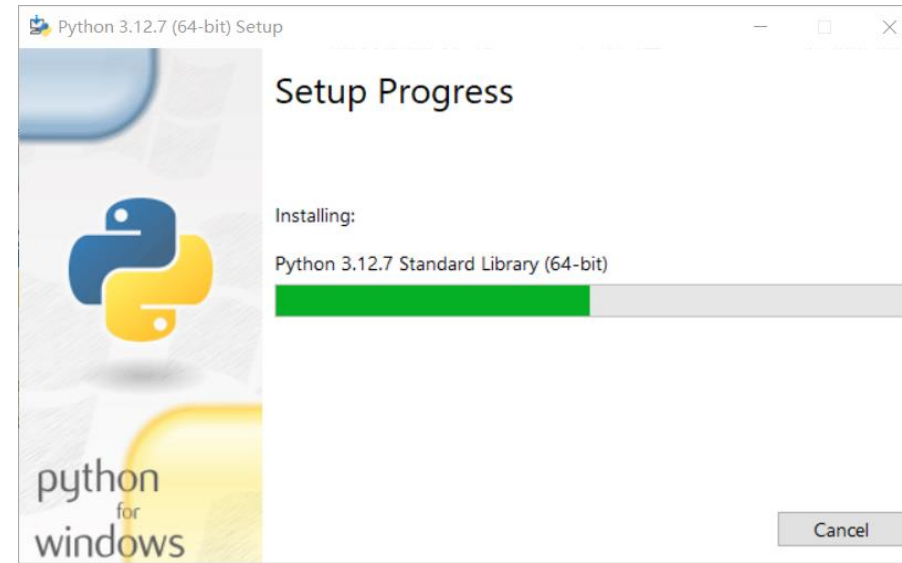
**Note that Python 3.12.7 cannot be used on Windows 7 or earlier.**

- [Download Windows installer \(64-bit\)](#)
  - [Download Windows installer \(32-bit\)](#)
  - [Download Windows installer \(ARM64\)](#)
  - [Download Windows embeddable package \(64-bit\)](#)
  - [Download Windows embeddable package \(32-bit\)](#)
  - [Download Windows embeddable package \(ARM64\)](#)
- Python 3.11.10 - Sept. 7, 2024

约25M

### Pre-releases

- [Python 3.13.0rc3 - Oct. 1, 2024](#)
  - [Download Windows installer \(64-bit\)](#)
  - [Download Windows installer \(32-bit\)](#)
  - [Download Windows installer \(ARM64\)](#)
  - [Download Windows embeddable package \(64-bit\)](#)
  - [Download Windows embeddable package \(32-bit\)](#)
  - [Download Windows embeddable package \(ARM64\)](#)
- [Python 3.13.0rc2 - Sept. 6, 2024](#)
  - [Download Windows installer \(64-bit\)](#)



# 安装R/Python包

```
options(repos=structure(c(CRAN="https://mirrors.tuna.tsinghua.edu.cn/CRAN/")))
```

```
r  
install.packages("ggplot2")
```

```
r  
install.packages("devtools")  
devtools::install_github("hadley/ggplot2")
```

```
r  
  
if (!requireNamespace("BiocManager", quietly = TRUE))  
  install.packages("BiocManager")  
BiocManager::install("BiocGenerics")
```

```
options(Bioc_mirror="https://mirrors.tuna.tsinghua.edu.cn/bioconductor")
```

```
bash  
conda install r-ggplot2
```

```
r 本地安装  
install.packages("ggplot2_4.0.0.tar.gz", repos=NULL, type="source")
```

`library('DESeq2')` 无报错, 安装成功\*

```
bash  
pip install matplotlib  
-i https://mirrors.tuna.tsinghua.edu.cn/pypi/web/simple
```

```
bash  
conda install matplotlib
```

```
bash 本地安装  
pip install /path/to/matplotlib-3.1.1-cp37-cp37m-manylinux1_x86_64.whl
```

`import matplotlib` 无报错, 安装成功\*

# 常用数据类型

| R 数据类型     | 描述     | R 示例                                     | Python 数据类型        | 描述   | Python 示例  |
|------------|--------|--|--------------------|------|--|
| Integer    | 整数     | 5  | int                | 整数   | 5  |
| Double     | 双精度浮点数 | 5.3                                      | float              | 浮点数  | 5.3  |
| Character  | 字符     | "R"                                      | str                | 字符串  | "Python"   |
| Logical    | 逻辑值    | TRUE, FALSE, T, F                        | bool               | 布尔值  | True, False  |
| Vector     | 向量     | c(1, 2, 3)                               | list               | 列表   | [1, 2, 3]  |
| Matrix     | 矩阵     | matrix(1:6, nrow=2)                      | list of list       | 二维列表 | [[1, 2], [3, 4]]                                   |
| Array      | 数组     | array(1:12, dim=c(3,4))                  | numpy.array        | 数组   | numpy.array([[1,2],[3,4],[5,6]])                   |
| Factor     | 分类变量   | factor(c("low", "medium", "high"))       | pandas.Categorical | 分类变量 | pandas.Categorical(["low", "medium", "high"])      |
| Data Frame | 数据框    | data.frame(x=c(1,2,3), y=c("a","b","c")) | pandas.DataFrame   | 数据框  | pandas.DataFrame({'x':[1,2,3], 'y':['a','b','c']}) |
| List       | 列表     | list(a=1, b="R")                         | list               | 列表   | [1, "Python"]                                      |

# 读写txt文件

读取文本文件:

```
r  
  
# 读取文本文件  
data <- read.table("test.txt", header = TRUE, sep = "")  
  
# 查看数据  
print(data)
```

写入文本文件:

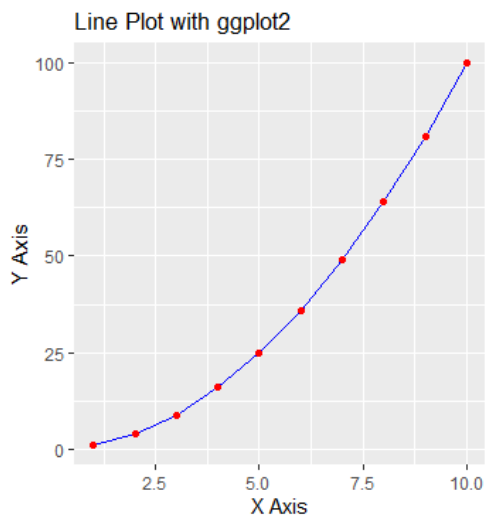
```
r  
  
# 创建一个数据框  
data_to_write <- data.frame(  
  Column1 = 1:5,  
  Column2 = letters[1:5]  
)  
  
# 写入文本文件  
write.table(data_to_write, "test.txt", row.names = FALSE, col.names = TRUE,  
  sep = "\t")
```

读取文本文件:

```
python  
  
# 读取文本文件  
with open('test.txt', 'r') as file:  
    content = file.read()  
  
# 打印文件内容  
print(content)
```

写入文本文件:

```
python  
  
# 要写入的内容  
content_to_write = "Hello, this is a test string."  
  
# 写入文本文件  
with open('test.txt', 'w') as file:  
    file.write(content_to_write)
```



r

```
# Install and load ggplot2
install.packages("ggplot2")
library(ggplot2)

# Create a data frame
df <- data.frame(x = 1:10, y = (1:10)^2)

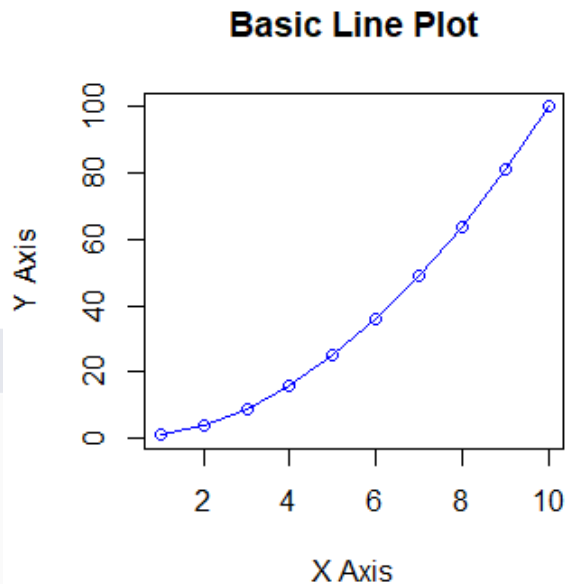
# Use ggplot2 to create a line plot
ggplot(df, aes(x = x, y = y)) +
  geom_line(color="blue") +
  geom_point(color="red") +
  ggtitle("Line Plot with ggplot2") +
  xlab("X Axis") +
  ylab("Y Axis")
```

# 折线图

r

```
# Create data
x <- 1:10
y <- x^2

# Create a line plot
plot(x, y, type="o", col="blue", main="Basic Line Plot", xlab="X Axis",
      ylab="Y Axis")
```

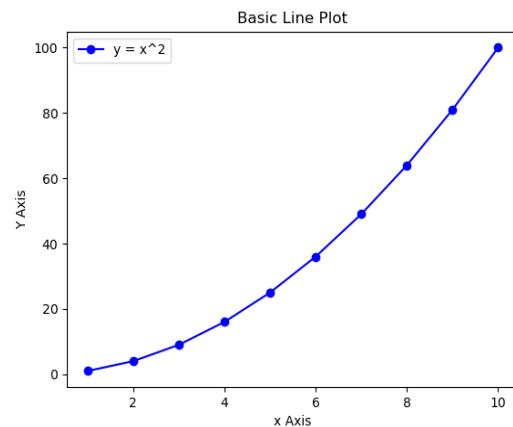


python

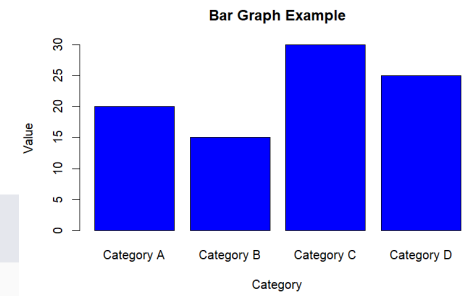
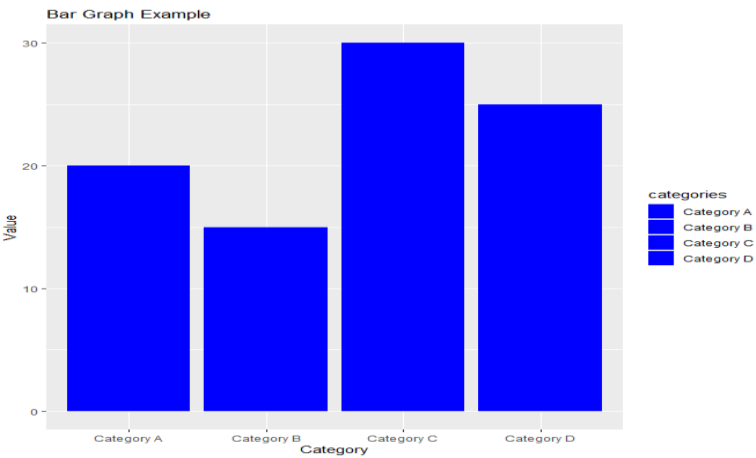
```
import matplotlib.pyplot as plt

# Create data
x = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
y = [i**2 for i in x]

# Create a line plot
plt.plot(x, y, marker='o', color='blue', label='y = x^2')
plt.title("Basic Line Plot")
plt.xlabel("X Axis")
plt.ylabel("Y Axis")
plt.legend()
plt.show()
```



# Bar图



r

```
# Create data
values <- c(20, 15, 30, 25)
categories <- c("Category A", "Category B", "Category C", "Category D")

# Plot bar graph
barplot(values, names.arg = categories, main = "Bar Graph Example", xlab =
"Category", ylab = "Value", col = "blue")
```

r

```
# Install and load the ggplot2 package if not already installed
if (!require(ggplot2)) {
  install.packages("ggplot2")
}
library(ggplot2)

# Create a data frame with the values and categories
data <- data.frame(
  categories = c("Category A", "Category B", "Category C", "Category D"),
  values = c(20, 15, 30, 25)
)

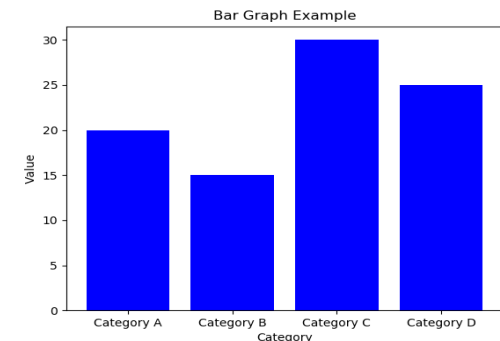
# Use ggplot2 to create a bar plot with the same blue color for all bars
ggplot(data, aes(x = categories, y = values, fill = categories)) +
  geom_bar(stat = "identity") +
  ggtitle("Bar Graph Example") +
  xlab("Category") +
  ylab("Value") +
  scale_fill_manual(values = c('blue', 'blue', 'blue', 'blue')) # Set the
fill color to blue for all bars
```

python

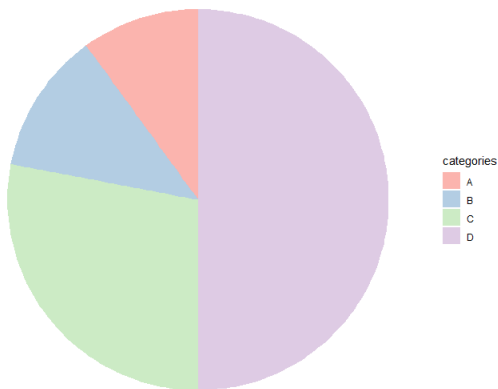
```
import matplotlib.pyplot as plt

# Create data
values = [20, 15, 30, 25]
categories = ['Category A', 'Category B', 'Category C', 'Category D']

# Plot bar graph
plt.bar(categories, values, color='blue')
plt.title('Bar Graph Example')
plt.xlabel('Category')
plt.ylabel('Value')
plt.show()
```

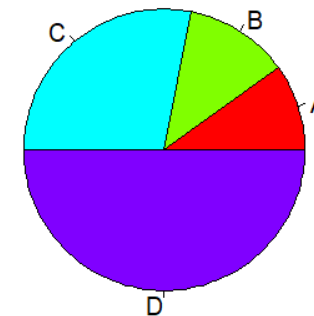


Pie Chart Example



# Pie 餅

Pie Chart Example



```
r

# Install and load the ggplot2 package if not already installed
if (!require(ggplot2)) {
  install.packages("ggplot2")
}
library(ggplot2)

# Create data
data <- data.frame(
  categories = c("A", "B", "C", "D"),
  values = c(10, 12, 28, 50)
)

# Use ggplot2 to create a pie chart
ggplot(data, aes(x = "", y = values, fill = categories)) +
  geom_bar(width = 1, stat = "identity") + # Create bars with a width of 1
  coord_polar("y", start = 0) + # Transform the bar chart into a pie chart
  ggtitle("Pie Chart Example") + # Add a title
  scale_fill_brewer(palette = "Pastel1") + # Set the fill color using a
  color palette
  theme_void() # Use a minimal theme
```

```
r

# Create data
slices <- c(10, 12, 28, 50)

# Create labels for the slices
labels <- c("A", "B", "C", "D")

# Plot pie chart
pie(slices, labels = labels, main = "Pie Chart Example", col =
rainbow(length(slices)))
```

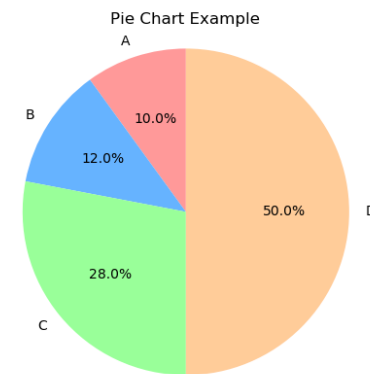
python

```
import matplotlib.pyplot as plt

# Create data
slices = [10, 12, 28, 50]

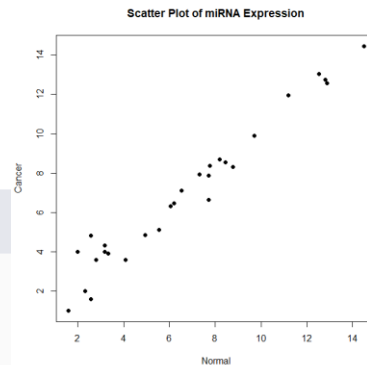
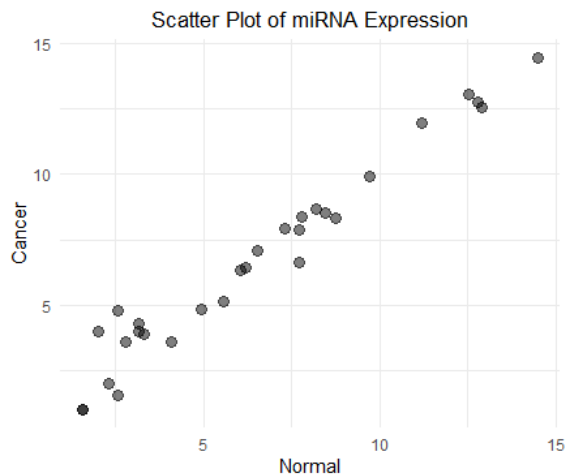
# Create labels for the slices
labels = ['A', 'B', 'C', 'D']

# Plot pie chart
plt.pie(slices, labels=labels, autopct='%1.1f%%', startangle=90, colors=
['#ff9999', '#66b3ff', '#99ff99', '#ffcc99'])
plt.axis('equal') # Equal aspect ratio ensures that pie is drawn as a
circle.
plt.title('Pie Chart Example')
plt.show()
```





# 散点图



```
r

# 读取数据
data <- read.table("4_data.txt", header = TRUE, sep = "\t",
stringsAsFactors = FALSE)

# 转换为data.frame
data <- data.frame(data)

# 使用ggplot2绘制散点图
library(ggplot2)

ggplot(data, aes(x = normal, y = cancer)) +
  geom_point(color = "#000000", size = 3, alpha = 0.5) +
  labs(title = "Scatter Plot of miRNA Expression", x = "Normal", y =
"Cancer") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5)) # 居中标题
```

```
r

# 读取数据
data <- read.table("4_data.txt", header = TRUE, sep = "\t",
stringsAsFactors = FALSE)

# 绘制散点图
plot(data$normal, data$cancer, main = "Scatter Plot of miRNA Expression",
xlab = "Normal", ylab = "Cancer", pch = 19, col = "#000000")
```

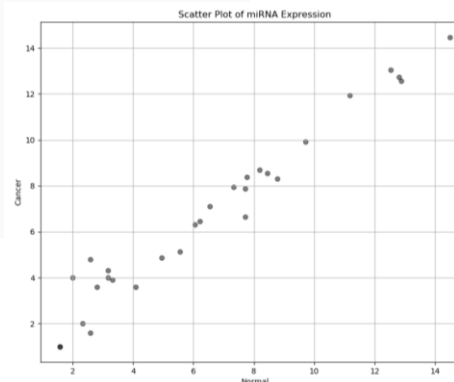
```
python

import pandas as pd
import matplotlib.pyplot as plt

# 读取数据
data = pd.read_csv("4_data.txt", sep="\t", index_col=0)

# 绘制散点图
plt.figure(figsize=(10, 8))
plt.scatter(data['normal'], data['cancer'], color='#000000', alpha=0.5)
plt.title('Scatter Plot of miRNA Expression')
plt.xlabel('Normal')
plt.ylabel('Cancer')
plt.grid(True)

plt.show()
```



# R vs Python

—————> 左手R, 右手Python

| 特性    | R语言                 | Python语言                    |
|-------|---------------------|-----------------------------|
| 语法简洁性 | 相对较复杂               | 语法简洁, 易于学习                  |
| 学习曲线  | 陡峭                  | 相对平缓                        |
| 统计分析  | 强大的统计分析功能           | 强大的库支持统计分析                  |
| 数据可视化 | ggplot2等库提供强大的可视化功能 | matplotlib、seaborn等库提供可视化功能 |
| 社区和支持 | 强大的统计学和生物统计学社区      | 广泛的社区支持, 适用于多种领域            |
| 包/库数量 | 大量专门针对统计分析的包        | 丰富的库, 适用于数据分析、机器学习、web开发等   |
| 性能    | 处理大型数据集时可能较慢        | 通常更快, 特别是在大数据集上             |
| 可扩展性  | 有限                  | 通过Cython、C++等可以提高性能         |
| 适用领域  | 统计分析、数据可视化、机器学习     | 数据分析、机器学习、web开发、自动化脚本等      |
| 生物信息学 | 有专门的Bioconductor项目  | 有Biopython等库                |
| 部署和应用 | 较少用于生产环境            | 易于集成到生产环境                   |
| 多语言支持 | 主要限于R               | 可以与其他语言 (如C、Java) 集成        |

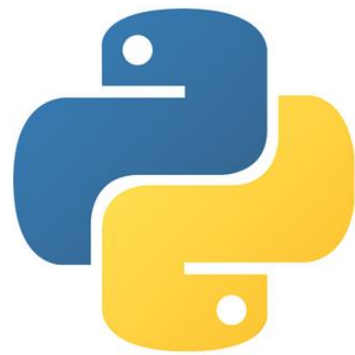
# 适合自己的才是最好的



when starting to build a new software system. The definition of the TIOBE index can be found [here](#).

| Sep 2024 | Sep 2023 | Change | Programming Language | Ratings | Change |
|----------|----------|--------|----------------------|---------|--------|
| 1        | 1        |        | Python               | 20.17%  | +6.01% |
| 2        | 3        | ▲      | C++                  | 10.75%  | +0.09% |
| 3        | 4        | ▲      | Java                 | 9.45%   | -0.04% |
| 4        | 2        | ▼      | C                    | 8.89%   | -2.38% |
| 5        | 5        |        | C#                   | 6.08%   | -1.22% |
| 6        | 6        |        | JavaScript           | 3.92%   | +0.62% |
| 7        | 7        |        | Visual Basic         | 2.70%   | +0.48% |
| 8        | 12       | ▲▲     | Go                   | 2.35%   | +1.16% |
| 9        | 10       | ▲      | SQL                  | 1.94%   | +0.50% |
| 10       | 11       | ▲      | Fortran              | 1.78%   | +0.49% |
| 11       | 15       | ▲▲     | Delphi/Object Pascal | 1.77%   | +0.75% |
| 12       | 13       | ▲      | MATLAB               | 1.47%   | +0.28% |
| 13       | 8        | ▼      | PHP                  | 1.46%   | -0.09% |
| 14       | 17       | ▲      | Rust                 | 1.32%   | +0.35% |
| 15       | 18       | ▲      | R                    | 1.20%   | +0.23% |

取决于需求、项目要求和个人偏好  
 \*R专用。统计分析和数据可视化  
 \*Python通用。更广泛的编程任务，  
 更复杂的应用程序和系统，人工智能



| 特性    | R语言                 | Python语言                    |
|-------|---------------------|-----------------------------|
| 语法简洁性 | 相对较复杂               | 语法简洁, 易于学习                  |
| 学习曲线  | 陡峭                  | 相对平缓                        |
| 统计分析  | 强大的统计分析功能           | 强大的库支持统计分析                  |
| 数据可视化 | ggplot2等库提供强大的可视化功能 | matplotlib、seaborn等库提供可视化功能 |
| 社区和支持 | 强大的统计学和生物统计学社区      | 广泛的社区支持, 适用于多种领域            |
| 包/库数量 | 大量专门针对统计分析的包        | 丰富的库, 适用于数据分析、机器学习、web开发等   |
| 性能    | 处理大型数据集时可能较慢        | 通常更快, 特别是在大数据集上             |
| 可扩展性  | 有限                  | 通过Cython、C++等可以提高性能         |
| 适用领域  | 统计分析、数据可视化、机器学习     | 数据分析、机器学习、web开发、自动化脚本等      |
| 生物信息学 | 有专门的Bioconductor项目  | 有Biopython等库                |
| 部署和应用 | 较少用于生产环境            | 易于集成到生产环境                   |
| 多语言支持 | 主要限于R               | 可以与其他语言 (如C、Java) 集成        |