**基本语法**

sum(x)#求和 barplot(x)#条形图

cat("1+1",1+1,sep="=") #命令行：1+1=2

cat(…,file="test.txt") #写入指定文件

cat(……,append=TRUE) #不覆盖原文件

sink()：把命令行结果保存到文件；保存文件的同时在命令行显示：split=TRUE

sink("test.txt",split=TRUE)#开始输出 sink()#停止

source()：运行.R文件中的代码

getwd()：获取工作目录

setwd()：改变当前工作目录

ls()：查看保存在工作空间的对象

rm()：删除对象 rm(list=ls())清空对象

默认的包：getOption("defaultPackages")

目前加载在环境中的包：(.packages())

从官网安装R包：install.packages("BeSS")

R包加载：library(BeSS)或require(BeSS)

>>>居中|右对齐|左对齐|默认

>>>:-:|-:|:-|-（换行） >>>A|B|C|D

输出：>>>

|  |  |  |  |
| --- | --- | --- | --- |
| 居中 | 右齐 | 左齐 | 默认 |
| A | B | C | D |

自动超链接：http://www.ustc.edu.cn

标准超链接：[中科大](http://…)

带鼠标标题：[中科大](http://…"南七")

插图图片：![](./figures/ustc.jpeg)

**数据类型**

整型：如1L； 字符型；

数值型/双整型：如1，1.1

逻辑型：TRUE和FALSE，缺失时为NA

复数类型：存储复数，如1+3i

日期时间类型(Date,POSIXct,POSIXlt)

因子型：存储分类数据，如factor("male")

特殊符号：NA，NULL，NaN，Inf(infinite)

typeof()判断数据类型

is.foo()判断是否属于某种类型foo

as.foo()强制转换成foo类型

2^2#幂运算 5%%2#取余运算

5%/%2#整除运算 log10(100)#对数

exp(1)#指数 round(3.14)#四舍五入

round(3.14159，2) #四舍五入到两位小数

floor(3.1)#向下取整 pi #圆周率

ceiling(-3.1) #向上取整，结果为-3

7!=5 ##[1]T !(5>7) ##T

(5>7)&(6\*7==42) ##[1]FALSE

(5>7)|(6\*7==42) ##[1]TRUE

判断两个浮点型对象是否完全相同，不能用==和identical(a,b)，应该用all.equal(a,b)

nchar()以字符个数为单位计算字符串长度

nchar("茉子",type="bytes")#以字节为单位

toupper()全部大写；tolower()全部小写

substr(x,start,stop)从字符串x中取出从第start个到第stop个的子串，开头为1

substring(x,start)可以从字符串x中取出从第start个到末尾的子串

substr(x,start,stop)=’y’替代特定位置

gsub(pattern,replacement,x):在x中找到pattern对应字符，替换成replacement

paste("常陆",…,"茉子",sep="-")连接两个以上字符型对象，默认用空格连接

paste0()默认直接连接没有间隔

strsplit(x,split="，")用"，"拆分x

[0-9]表示数字，等价于\d

[a-zA-Z0-9]表示数字和所有的英文字母，等价于\w和[:alnum:]

[:cntrl:]表示ASCII控制字符

[:punct:]表示不属于[:alnum:]、[:cntrl:]的

[:space:]表示任意空格

+一个或多个，?零个或一个，^x除x外任意字符

gsub('[[:space:]]+','',x)#删除x中多余空格

strsplit(x,split="[[:space:]]|[[:punct:]]")

#用空格和符号拆分x（会变成一个个单词）

Date为整数，值为从1970/1/1经过天数

POSIXct值为从1970/1/1零时经过秒数

POSIXlt保存从年到秒的列表

Sys.Date()##"2024-07-21"←Date

Sys.time()##"2024-07-21\_11:45:14\_CST"←POSI

as.Date()，as.POSIXlt()，as.POSIXct()从字符型数据生成日期型数据，指定格式：rmat="%m/%d/%Y"，默认分隔符为-或/ %Y年(四位)%y年(两位)

%m月(数字格式)%B月(英文全称)%b月(英文简称)

weekdays()#获取星期 quarters()#季度

as.Date("…07-21")+10 ##[1]"2021-07-31"

as.Date("…07-21")-21 ##[1]"2021-06-30"

as.POSIXlt("…18:47:22")-30 ##"…18:46:52"

difftime(as.Date("2021-9-10"),as.Date("2020-9-10"),units='days')##Time\_difference\_of\_365\_days

difftime(as.POSIXlt("2021-9-10\_18:47:22"), as.POSIXlt("2021-8-10\_10:47:22"), units='days') ##31.33333days

有序变量类之间有大小顺序关系，如打分“A”，“B"，仅知道“A”>“B"，不知道类之间差异是多少，即不能进行运算"A"-"B"

无序变量类之间没有顺序，如性别职业等

factor(LETTERS[1:3]) ##Levels:ABC

factor(LETTERS[1:3],ordered=TRUE)##Levels:A<B<C

factor(LETTERS[1:3],ordered=TRUE,levels=c("C","B","A")) ##Levels:C<B<A

**数据结构**

factor(c("男","女")) #因子型向量

c(7,21)>10#逻辑型，值为#FALSE，TRUE

c('茉子',1,NA)#字符型； c(7,21)#数值型

构建规则向量：1:10 #1到10

seq(from=5,to=25,by=5) #5,10,15,20,25

seq(as.Date('…7-21'),by='days',length=5)#连续五天

seq(as.Date('2021-07-21'),to=as.Date('2022-07-21'),by='2\_weeks') #间隔两周

运算：x<-c(7,21)，y<-c(0,0,1,2,3)

x\*1 ##[1]7,21 x>10 ##[1]FALSE,TRUE

x+y ##[1]7,21,8,23,10(重复用短的)

c(1,NA,3)>2 ##[1]FALSE,NA,TRUE

(x>10)&(x<30) ##[1]FALSE,TRUE

unique(c(1,2,3,2)) #找出唯一元素##[1]123

intersect(c(1,2,3,2),c(1,2)) #交集 ##[1]12

union(c(1,2,3,2),c(1,2)) #并集 ##[1]123

setdiff(c(1,2,3,2),c(1,2)) #差集 ##[1]3

setequal(c(1,2,3,2),c(1,2,3)) #是否相等##TRUE

order()返回排序序号； sort()返回排序结果

all(x>0) ##[1]TRUE

identical(x,y) 比较对应位置元素是否一致

all.equal(x,y) 比较x,y包含元素是否一致

match(c(1,3),c(2,3,4,3)) ##[1]NA,2

which(c(1,3)%in%c(2,3,3)) #找符合的下标#2

table(c(1,2,3,2)) 统计各元素个数返回表格

substr(x,1,2) 返回x每个元素的1-2字符

paste(x,y) 在x各元素尾追加y各元素

下标子集：x[2]访问2位置的元素和子集

x[-2]扣除2位置的元素后的子集

x[5]下标超界 ##NA x[0] ##numeric(0)

x[4]<-1 ##7,21,NA,1 x[x>7] ##21

matrix(1:6,nrow=2,ncol=3)2行3列先填列

rbind() 按列合并 cbind() 按行合并

diag(n) n阶单位阵 diag(c(3,4))对角线3,4

dim()返回列数行数nrow()行数ncol()列数

运算：A\*B对应元素相乘，不是矩阵乘法

A%\*%B矩阵乘法，要求A列数等于B行数

t()转秩 det()行列式

solve(A)A的逆 solve(A,b)Ax=b的解

crossprod(A,B)ATB tcrossprod(A,B)ABT

crossprod(A)ATA tcrossprod(A)AAT

apply(A,i,FUN)把矩阵A的每一列分别输入到函数FUN中，得到对应于每一维度的结果，其中i=1表示对行进行运算，i=2表示对列进行运算

summary() 函数按列输出汇总信息

rowMeans(),colMeans(),rowSums(),colSums()

下标子集：A[6] #同向量，先列后行依次取

A[1,2] #提取A的第一行第二列的元素

A[1,] #提取A的第一行，返回一个向量

A[1,,drop=FALSE] #保持原有行列数

A[1:2,c(1,3)] #提取指定行列对应的子矩阵

A[A>2] #A中所有大于2的元素

数组名<-array(数组元素,dim=c(第一下标个数,第二下标个数,...,第s下标个数))

arr<-array(1:24,dim=c(2,3,4))#2行3列4维，填充先列后行再维，数组是矩阵拓展，运算同矩阵

x<-list("茉子",7,TRUE) 列表元素类型可不同

names(x)<-c("name","age","love") #命名

list(name="茉子",age=18,love=TRUE) #初始命名

单个列表元素必须用两重方括号格式访问

x[[1]]##"茉子" x[[‘age’]]##18

若用单重方括号，结果是列表而非元素

或用$访问：x$love ##TRUE

直接给列表不存在的元素名定义元素值就添加了新元素：x$sex<-TRUE，赋值为NULL就删掉该元素

要把已经存在的元素改为NULL值而非删除，或给列表增加一个值为NULL的元素，要用单重方括号取子集，使子集保持其列表类型，给这样的子列表赋值为list(NULL)如dist['sex']<-list(NULL)

as.list()转换成列表unlist()列表转基本向量

d=data.frame(1:2,log=c(TRUE,FALSE),char=c("M","F"))2行3列，列间不同类

因为数据框的一行不一定是相同数据类型，固数据框的一行作为子集，结果还是数据框，而非向量

(d$列一+d$列二)/exp(d$列三)等价于with(d,(列一+列二)/exp(列三))

**数据处理**

load("cats.RData")#读取数据

read.csv()参数设置：

header:导入数据是否带列标题,默认TRUE

sep:列与列之间的文本分隔符

stringsAsFactor:导入数据时是否奖字符串数据转化成因子类型，默认是TRUE

na.strings=x指定字符串x为缺失值NA

row.names=x指定列名为x的列为行名

skip=x跳过前面x行

readLines()按行读取，每一行记为一个字符串

head(x,n)选择数据框x的前n行

tail(x,n)选择数据框x的倒数n行

View(x)：打开界面查看数据

subset(d,d$年龄<18)提出子集

d[d$年龄<18&d[,"性别"]=="男",c("性别","年龄","疗程")] #同时选择行和列的子集

order() #返回排序后各个元素的原位置信息

head(d[order(d[,"年龄"]),],6) #年龄最小的前6行

d[apply(is.na(d),1,any),] #所有有缺失值的行

如果行号都一样，那么使用data.frame(dat1,dat2)或者cbind(dat1,dat2)直接合并即可

如果行号不一样，用merge(dat1,dat2,by=x)按照列名为x来合并，只保留x元素相同的行，即同时在两个数据框中的行

用merge(dat1,dat2,by.x=x,by.y=y)把dat1中的x列和dat2中的y列作为合并的标准

scale()把每一列都标准化，即每一列都减去该列的平均值，然后除以该列的样本标准差

scale(x,center=TRUE,scale=FALS)仅中心化不标准化

总体信息：summary()，table()

位置度量：mean(),median()

分散程度（变异性）度量：sd(),IQR(),mad()

分位数：min(),max(),quantile()

mean(d$疗程,na.rm=TRUE)去除NA数值来计算

aggregate(d[,c(3:5,7)],by=d[c("分型","性别")],mean)对输入的数据框用指定的分组变量（或交叉分组）分组进行概括统计（不同分型、性别均值）

tapply(d[,"性别"],INDEX=d["分型"],table)对向量进行分组概括（不同分型下的性别数量），不计算NA，通过useNA="always"或useNA="ifany"来把NA计算在内，后者在NA为0个是不进行显示

table(d[,"分型"],d[,"性别"])对两个分类变量进行交叉分组计算频数（行为分型列为性别的2x2表格）

dplyr包

d<-readxl::read\_excel("covid.xlsx",col\_names=TRUE)

filter(d,d$年龄<18,性别=="男") 按行筛选数据

select(d,性别,年龄,疗程) 按名称选取变量/列

arrange(d,性别,desc(年龄)) 按两列对行排序数据

mutate(d,住院时间=出院时间-入院时间+1)创建新列变量，返回含有新变量以及原变量的新数据框

summarise(group\_by(d,分型,性别),mean(年龄),mean(入院时间),mean(出院时间))按照分型和性别分组汇总数据

%>%管道，x%>%f(y)转换为f(x,y)：

d %>%

mutate(住院时间=出院时间-入院时间+1) %>%

filter(年龄<18) #计算住院时间并筛选未成年

set.seed(seed,kind=NULL,normal.kind=NULL,sample.kind=NULL) #设置随机数种子

seed=k指定一个编号为k的种子，这样每次从k种子运行相同的模拟程序就可以得到相同的结果

kind=指定后续程序要使用的随机数发生器名称

normal.kind=指定用的正态分布随机数发生器名

runif(n) #产生n个标准均匀分布随机数

rnorm(n) #产生n个标准正态分布随机数

sample(x,size,replace=FALSE,prob=NULL)

x用以存储有限集合的向量，也可以为一个正整数（此时集合为1:x）

size指定抽样个数，即样本数

replace=指定是否为有放回抽样，TRUE是有放回抽样，FALSE是无放回抽样

prob=c(.2,.3,.5)指定各种权重抽取，默认等概率

**基本绘图**

graphics包的plot()

barplot()函数可以对这样的频数结果绘制条形图

main=：标题

col=：不同条形的颜色函数colors()可以返回R中定义的用字符串表示的六百多种颜色名字

horiz=是否需要横排，默认为FALSE竖排

beside=是否为并排条形图，默认为FALSE，即堆叠条形图

pie()饼图可以看成一种特殊的条形图，它是将条形图的y值变成了角度

boxplot(d[,"年龄"])箱线图可以简洁地表现变量分布，主要利用5个点来绘制,中间粗线是中位数,盒子上下边缘是和分位数,两条触须线延伸到取值区域的边缘

boxplot(年龄~性别,data=d)盒形图可以很容易地比较两组或多组数据的分布情况

hist(x)对x作频数直方图，直方图自动等距分段，纵坐标为每段的样本点个数

freq=FALSE改变y轴的计数为比例

main=：标题

xlab=，ylab=：x/y轴标签

xlim=，ylim=：x或y轴的绘图范围

add=TRUE：在已有的直方图上再叠加一个

density(x)对数据x进行核密度估计，需配合plot()来进行画图，如lines(density(d[,"年龄"]))

plot(x,y)或者plot(y~x,,data=d)散点图

pch：点的形状

col：点的颜色，输入数字或颜色的名字

col=ifelse(性别=='男','blue','red')#用颜色分类

cex：点的大小

cex=1+(入院时间-min(入院时间))/(max(入院时间)-min(入院时间))#用气泡大小表现第三维（入院时间的早晚）

plot函数中使用type='l'参数可以作折线图，lwd指定线宽度，lty指定虚线

qqnorm(d[,"年龄"])正态QQ点图

qqline(d[,"年龄"],lwd=2,col='blue')正态QQ线图

qqplot(x,y)判别两个数据是否来自于同一个分布

curve(expr,from,to)可以对以x为自变量的表达式expr做函数曲线，或者对某个函数作函数曲线

用persp函数作三维曲面图,contour作等值线图，image作色块图

坐标x和y构成一张平面网格，数据z是包含z坐标的矩阵，每行对应一个横坐标，每列对应一个纵坐标，persp(x,y,normal.mat)另两个同

rgl包里的plot3d函数可绘制三维散点图

低级图形也包含legend(),axis(),text(),mtext(),par()等给图形添加图例、坐标系、文字和改变图形参数

用abline()函数在图中增加直线

a指定截距 v指定横坐标画竖线

b指定斜率 h指定纵坐标画水平线

points()：在图中增加散点lines()：在图中增加折线

par()函数通过设置mfrow（按行）或者mfcol（按列）将页面分几个区域，每个区域对应一个图形

layout()函数提供一种更灵活的图形组合方式，允许有着不同尺寸的区域

**ggplot2画图**

library(ggplot2)

ggplot(data=mp,aes(x=displ,y=hwy))+geom\_point()

数据：mp

图形属性映射：aes(),d映射到x轴,h映射到y轴

几何对象：geom\_point()，散点图

ggplot2中的数据必须存储在data.frame格式中，通过aes()函数将数据映射到图形属性：

ggplot(data=mpg,aes(x=displ,y=hwy,color="blue"))+geom\_point()

x:映射到x轴的变量 y:映射到y轴的变量

color:轮廓颜色 fill:内部填充颜色

alpha:透明度 shape:点的形状

linetype:线型，如实线solid,虚线dashed,点线dotted size:图形属性的大小

ggplot2层层叠加，层间通过+连接，每层图形通过geom或stat产生；每层图形的映射可从ggplot()函数继承，也可重新定义；可把基础图层保存下来，后面在此基础上添加不同图层

几何对象实现图层实际渲染，控制创建图形类型：

geom\_bar():离散变量分布 geom\_line():折线图

geom\_boxplot():箱线图 geom\_point():散点图

geom\_histogram():直方图 geom\_ribbon():条带

geom\_smooth():最佳拟合平滑曲线

统计变换通过某种形式的概括来转换数据，伴随着几何对象使用，很少直接调用：

stat\_bin()==geom\_bar(),geom\_histogram()

stat\_boxplot()==geom\_boxplot()

stat\_contour()==geom\_contour()

stat\_sum()==geom\_count()

一些不能用geom\_\*\*\*函数创建的统计变换：

stat\_ecdf()：计算经验累计分布函数图

stat\_summary()：在不同的x值上概述y

stat\_qq()：Q-Q图的计算

stat\_unique()：去掉重复的行

标度控制着数据到图形属性的映射，将数据转化为视觉上可以看得到的东西，如大小，颜色，形状

标度提供了读图时所用的工具：坐标轴和图例

标度scale\_y\_log10()由三个分离的下划线”\_“组成：

scale；图形属性的名字如：colour,shape,x,y,fill;

标度的名字，如：

连续时间标度：用于映射整数、数值、时间等到x轴或y轴，如continuous

颜色标度：用来映射连续或离散数据到颜色

人工标度：用来映射离散数据到所选的尺寸、线性或颜色上，如discrete,manual

线性坐标系：保持了几何对象的形状

coord\_cartesian()：默认的笛卡尔坐标系

coord\_fixed()：宽高比固定的直角坐标系

coord\_flip()：x轴和y轴反转了的笛卡尔坐标系

非线性坐标系：可以改变形状，如直线不再是直线

coord\_map()/coord\_quickmap()：地图投影

coord\_polar：极坐标系

coord\_trans()：对数据进行统计变换之后，对x和y位置进行任意变换

coord\_trans(y="log10")#对y轴取log变换

分面通过切割数据生成一系列小连号图：

facet\_null():单个图像，默认情况

facet\_wrap():把1维面板条状封装在2维中

facet\_grid():生成2维面板网格,行列由变量组成

封装分面facet\_wrap()

ggplot(data=mp,aes(x=di,y=hw))+geom\_point()

facet\_wrap()函数把1维面板条状封装在2维中,在处理单个多水平变量时，特别有用

通过nrow,ncol,as.table,dir控制网格封装条块：

nrow,ncol控制有多少行/列(设置一个即可)

as.table控制分面的布局，TRUE最高值显示在右下角，反之则显示在右上角

dir控制封装的方向，h表横向，v表纵向

网格分面facet\_grid()在2维网格中展示图像：

.~a把a的值按列展开

b~.把b的值按行展开

b~a把a的值按列展开，把b的值按行展开

可以是多个的，如a+b~c+d

标度控制：通过调整scales来控制面板的位置标度

scales="fixed":x和y标度在所有面板中都固定

scales="free\_x":x的标度可变，y的标度固定

scales="free\_y":y的标度可变，x的标度固定

scales="free":x和y标度在每个面板都可以变化

主题对图像中的非数据元素进行精细调整，不影响几何对象和标度等数据元素

ggplot2自带了八种内置主题theme\_grey(),theme\_

bw(),theme\_linedraw(),theme\_light(),theme\_dark(),theme\_minimal(),theme\_classic(),theme\_void()

修改单个主题组件，则需要使用形如plot+theme(element.name=element\_function())

主题元素（element.name）制定了能控制的非数据元素，如plot.title控制了图像标题的外观，axis.ticks.x控制了x轴上的标签

元素函数（element\_function）描述的是元素的视觉属性，如element\_text()设定了字体的大小、颜色内置的元素函数有四种基本类型：

文字（element\_text）:绘制标签和标题，如控制字体的family(字体族)、face(字型)、colour(颜色)、size(大小)、hjust(横向对齐)等

线条（element\_line）:绘制线条，参数有colour(颜色)、size(大小)和linetype(线条类型)

矩形（element\_rect）:绘制（背景的）矩形，参数有fill(填充)的颜色、边缘的colour(颜色)、size(大小)和linetype(线条类型)

空白（element\_blank）:不绘制任何东西

**程序控制结构**

if(条件){表达式}else\_if(条件){表达式}else{表达式}

ifelse(x>=0,x,-x)根据逻辑向量条件，选择不同结果

一般使用&&或||来组合多个逻辑表达式，因为只返回一个逻辑型标量；||遇到第一个TRUE，则返回TRUE不再继续；&&遇到FALSE同理

&或|是向量化操作符，作用于向量时返回多个值；非要使用，用any()或all()将其转换为单个逻辑值

for(循环变量in序列){表达式}

如果要对向量元素遍历，采用下标访问，用seq\_along(x)取代1:length(x),避免出现长度为零时出现错误下标：for(i\_in\_seq\_along(x)){print(x[i])}

while(循环继续条件){表达式}

repeat{…if(循环推出条件)break}#等于while(TRUE)

replicate(重复次数,{要重复的表达式})

**函数和并行计算**

函数名<-function(形式参数表){函数体}

return(y)的方式在函数体的任何位置退出函数并返回y的值，还可以用函数体的最后一个计算表达式作为返回值

function(x,c=1)给定形参c的缺省默认值

stopifnot()检查输入值是否满足括号内表达式，如有不满足的则停止执行后续命令并返回错误信息

函数在调用执行时，除非用到某个形式变量的值才求出其对应实参的值。形参缺省值也是只有在函数运行时用到该形参的值时才求值

允许在函数体内定义函数，但在函数内部定义的函数只能在局部使用

在调用函数foo前输入debug(foo)命令，可在下面实际调用时进入调试模式

在程序中插入browser()函数，可进入跟踪调试状态，可以实时地查看甚至修改运行时变量的值

用browser()函数与if结构配合可以制作条件断点，如在调试带有循环的程序时，发现错误发生在循环变量i等于501的时候，就可以在循环内插入：if(i==501)browser()

library(snowfall)并行计算

f10<-function(k,n){…}

n<-5000000

nk<-20 #预先设置运算函数

sfCpus()查看本计算机的虚拟核心（线程）数

sfInit(parallel=TRUE,cpus=4)发现有4个节点，用sfInit()建立临时的有4个节点的单机集群

sfExport("f10","n")用sfExport()把计算所依赖的对象预先传送到每个节点

类似的函数有

sfLibrary：把计算所依赖的包导入到每个节点

sfSource：把依赖的源代码文件导入每个节点

sfExportAll：导入所有的全局变量

v.sf<-sfSapply(2:(nk+1),function(k)f10(k,n))此函数是sapply函数的并行版本，用它来对k并行地循环，类似的函数有sfLapply、sfAapply

sfStop()并行执行结束后解散临时集群，防内存泄漏

**RShiny**

Shiny应用有两个组成部分：用户界面对象(Userinterface,UI)、服务端函数(Serverfunction)

这两个部分被作为参数传输到ShinyApp功能函数，再根据UI/Server对生成一个Shiny应用网页

搭建RShiny

目的是搭建一个网页应用：基于faithful数据绘制直方图，并允许用户交互指定直方图窗宽的个数

第一步：定义Shiny应用后台的R代码

x<-faithful[,2]

input<-9 #↓9个柱子

bins<-seq(min(x),max(x),length.out=input+1)

hist(x,breaks=bins,col='darkgray',border='white',xlab='Waitingtimetonexteruption(inmins)',main='Histogramofwaitingtimes')

第二步：定义用户界面对象/前端

#DefineUIforapplicationthatdrawsahistogram

ui<-fluidPage(

#Applicationtitle

titlePanel("OldFaithfulGeyserData"),

#Sidebarwithasliderinputfornumberofbins

sidebarLayout(

sidebarPanel(

sliderInput("bins","Numberofbins:",min=1,

max=50,value=30)),

#Showaplotofthegenerateddistribution

mainPanel(plotOutput("distPlot"))))

UI使用fluidPage()函数来指定网页的布局：

标题栏：titlePanel()

网页的布局：通过一层层地设计，而且每一层布局都要在输出中有所对应

这里用的是sidebarLayout()来实现左右布局的

边栏：sidebarPanel()

主体部分：mainPanel()

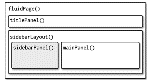
效果图：左边栏是用于摆放控件，右边空间用于展示图形、表格或者其他形式

fluidPage(#创建网页

titlePanel(), #标题栏

sidebarLayout( #常用的边栏布局，将输入布

局到左侧，输出布局到右侧设置position="right"

sidebarPanel(), #边栏

mainPanel())) #主体

所有输入控制函数都有相同的第一形参inputId，如果inputId为"name"，那么后台端会通过input$name来访问它

自由文本：textInput(),passwordInput(),textAreaInput().

数值输入：numericInput()，sliderInput().

日期输入：dateInput(),dateRangeInput().

单选：selectInput()，radioButtons().

多选：checkboxGroupInput().

上传文件：fileInput().

province<-c("中科大","清华","北大")

ui<-fluidPage(

textInput("变量名","显示问题"), #文本输入

numericInput("num","Numberone", value=0,min=0,max=100), #数值输入

selectInput("state","学校",province),#单选

checkboxGroupInput("animal","学校",province))

server<-function(input,output,session){}#↑多选

shinyApp(ui,server)

plotOutput()是对输出的控制，所有的输出函数都有相同第一形参outputId，如果outputId为"name"，那么后台端会通过output$name访问它；其余形参为具体的内容，如width等：

plotOutput()图形 tableOutput()静态表格

textOutput()带有格式的文本

verbatimTextOutput()代码和控制台输出

dataTableOutput()动态表格imageOutput()图片

ui<-fluidPage(textOutput("text"),verbatimTextOutput("code"),tableOutput("static"),dataTableOutput("dynamic"))#文本,代码,静态表格,动态表格

server<-function(input,output,session){

output$text<-renderText({"数据汇总如下"})

output$code<-renderPrint({summary(1:10)})

output$static<-renderTable(head(mtcars))

output$dynamic<-renderDataTable(mtcars,options=list(pageLength=5))}

shinyApp(ui,server)

第三步：配置后台端（服务器）

通过server()将前端的input输入到第一步的代码中，并将代码运行后的结果作为输出内容output和前端的输出匹配

这里的输出内容是图形，因此使用renderPlot()进行输出，定义输出的名称为distPlot，和第二步中的plotOutput("distPlot")相对应

通过调用前端的输入参数input$bins实现用户指定窗格的个数

server<-function(input,output){

output$distPlot<-renderPlot({

#generatebinsbasedoninput$binsfromui.R

x<-faithful[,2]

bins<-seq(min(x),max(x),length.out=input$bins+1)

#drawthehistogramwiththespecifiednumberofbins

hist(x,breaks=bins,col='darkgray',border=

'white',xlab='Waitingtimetonexteruption(in

mins)',main='Histogramofwaitingtimes')})}

每个前端输出函数都会在后端匹配一个render\*\*\*()函数，将R中某种形式的输出传递到网页上：

文本：textOutput()和renderText()

代码,控制台:verbatimTextOutput(),renderPrint()

表格：静态表格：tableOutput(),renderTable()

动态表格:dataTableOutput(),renderDataTable()

绘图：plotOutput()和renderPlot()

第四步：将前端和后台端连接，创建网页

shinyApp(ui=ui,server=server)

**R包开发**

第一步：创建一个空的R包，包内自动包括：

R/目录 man/目录

描述文档DESCRIPTION

命名空间文件NAMESPACE

RStudio项目文件pkgname.Rproj

接下来，删除以下文档：

NAMESPACE：自动产生

man/hello.Rd：自动产生

R/hello.R：修改R代码，因此先行删除

R包名称只能包含字母，数字和点号(.)，必须以一个字母开始，且不能以点号

结束

第二步：编辑描述文档，包括标题,描述,作者,版本,

依赖：可以通过Imports或Suggests来刻画新建的软件包所依赖的包，两者的区别在于依赖的程度

Imports中的包是必须安装的，在别人安装你的包的时候会同时把Imports中的包安装

Imports:stats,Matrix(>=1.2-6)#版本限制

Suggests是建议安装的，不是必须的

Suggests:stats,Matrix

第三步：将R代码保存到R/目录

将R代码保存到.R文件，并将文件保存在R/目录下

第四步：添加对象文档

文档通常保存在man/目录下，以.Rd作为后缀，语法大致基于LaTex，最后会被编译成HTML、纯文本和PDF格式的帮助文档以供查看

可手动编辑.Rd文档，但更推荐使用roxygen2包，因为可自动管理NAMESPACE和DESCRIPTION

大致的工作流程如下：

1.在.R文档中添加roxygen注释

2.运行devtools::document()或在RSudio中按Ctrl+Shift+D转化roxygen注释为.Rd文档

3.利用?预览文档，或点击来查看

4.修改注释，重复上面的步骤，直到满足要求

Roxygen注释总是#'开始，放在函数的前面。

第一句是文档的标题，应该占一行（80个字符以内），句首字母大写，并以句号结束

第二段是描述，用以简要说明函数的功能

第三段以及随后的段落进行细节的描述，通常可以分解成标签：

@param描述函数的输入或参数，及参数类型、用途，所有参数必须提供文档

@examples提供可执行代码,演示如何使用函数.用\dontrun{}包括无需运行的代码

@return描述函数的返回值

也可撰写帮助页面,描述包中最重要部分.方式为NULL函数添加@docType和@name

第五步：测试R包

检查R包，安装R包，测试R包是否安装正确

测试数据

将数据保存在新建文件夹data/中，并以.RData作为后缀，通过data(YourDataName)即可导入数据

如果DESCRIPTION文档中包含了LazyData:true，则数据集只有等到调用的时候才会读入内存

@format描述数据集；对于数据框，应包括一个描述每个变量的定义列表

@source提供数据来源的细节，通常是一个\url{}

**Rcpp**

library(Rcpp)

cppFunction("C代码，如定义double函数sumC")

sumC(3.5,6.5) #调用在R代码里嵌入C++函数

更好的方式是把程序保存在"\*.cpp"文件中，然后调用sourceCpp()编译使用：sourceCpp("….cpp")

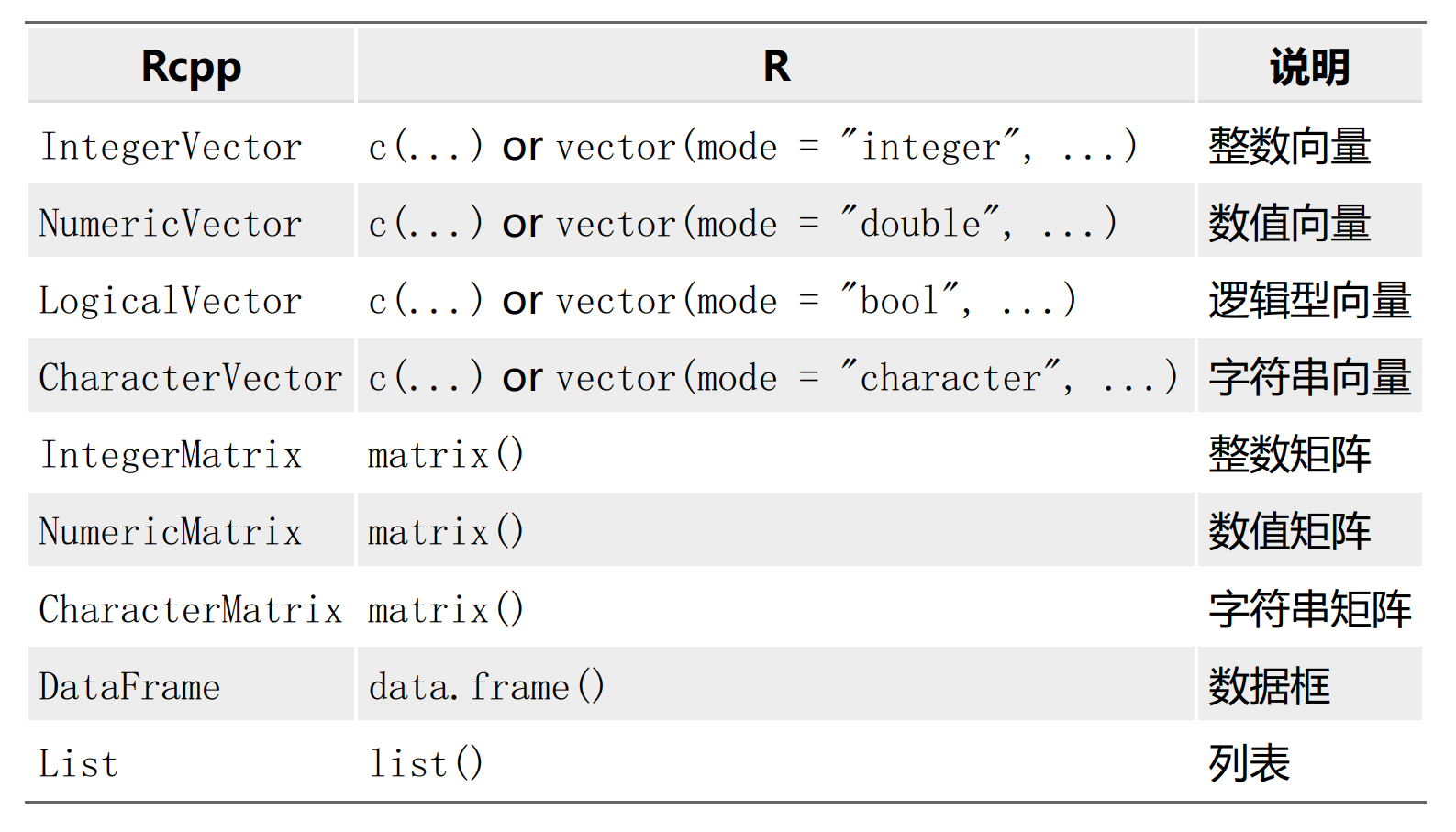
cpp文件开头应额外添加：

头文件：#include<Rcpp.h>

输出注释：//[[Rcpp::export]]

用wrap()把C++变量返回到R中。当C++中赋值运算的右侧表达式是R对象或R对象部分内容时，可隐含调用as()将其转换成左侧的C++类型

用as()函数把R变量转换为C++类型。当C++中赋值运算的左侧表达式是R对象或其部分内容时，可隐含调用wrap()将右侧C++类型转换成R类型



R版本：fooR<-function(x,y){ifelse(x<y,x\*x,-(y\*y))}

Rcpp糖：NumericVector\_fooRcpp(NumericVector\_

x,NumericVector\_y){returnifelse(x<y,x\*x,-(y\*y));}

逻辑判断函数：any,all

数学函数abs(),exp(),floor(),ceil(),pow()

产生糖表达式的函数：is\_na(),seq\_along(),seq\_len(),

pmax(),pmin(),ifelse()

apply族函数：sapply(),lapply(),mapply()

集合运算函数：setdiff(),union\_(),intersect(),unique(),

sort(),setequal()

统计函数：dnorm(),pnorm(),qnorm(),rnorm()

基于Rcpp创建R包

选择R包的类型为：R\_package\_using\_Rcpp，步骤一样。创建一个新的C++文件，包括一个基本的函数和有关开始的说明，保存在新目录src/下

Rcpp拓展包：

RcppArmadillo：使得线性代数的引入语法更加接近matlab

RcppEigen：优化过的线性代数运算

RInnside：实现在C++中调用R程序

RcppParallel：基于Rcpp实现并行计算

**Python**

函数的可变参数可以使传入函数的参数个数不限：

\*parameter可以接收任意多个参数，并放在一个元组中。参数传递时，按照位置传递

\*\*parameter接收任意多个参数并将参数放在一个字典中。在参数传递时，按照参数名传递

定义和调用时均需加\*号

map函数通过输入列表，然后对列表里的元素一一进行运算后输出一个运算后的列表，其实就是实现向量化运算：map(函数，列表)

匿名函数：lambda\_param1,param2,...:expression

filter函数通过输入一个判断函数和列表，输出使得判断函数为True的子列表：filter(func,list)