



南開大學



第一讲 R语言及其基本数据类型

赵宏

提 纲

- ◆ 1. 背景
- ◆ 2. R语言介绍
- ◆ 3. R语言的基本数据类型



1. 背景

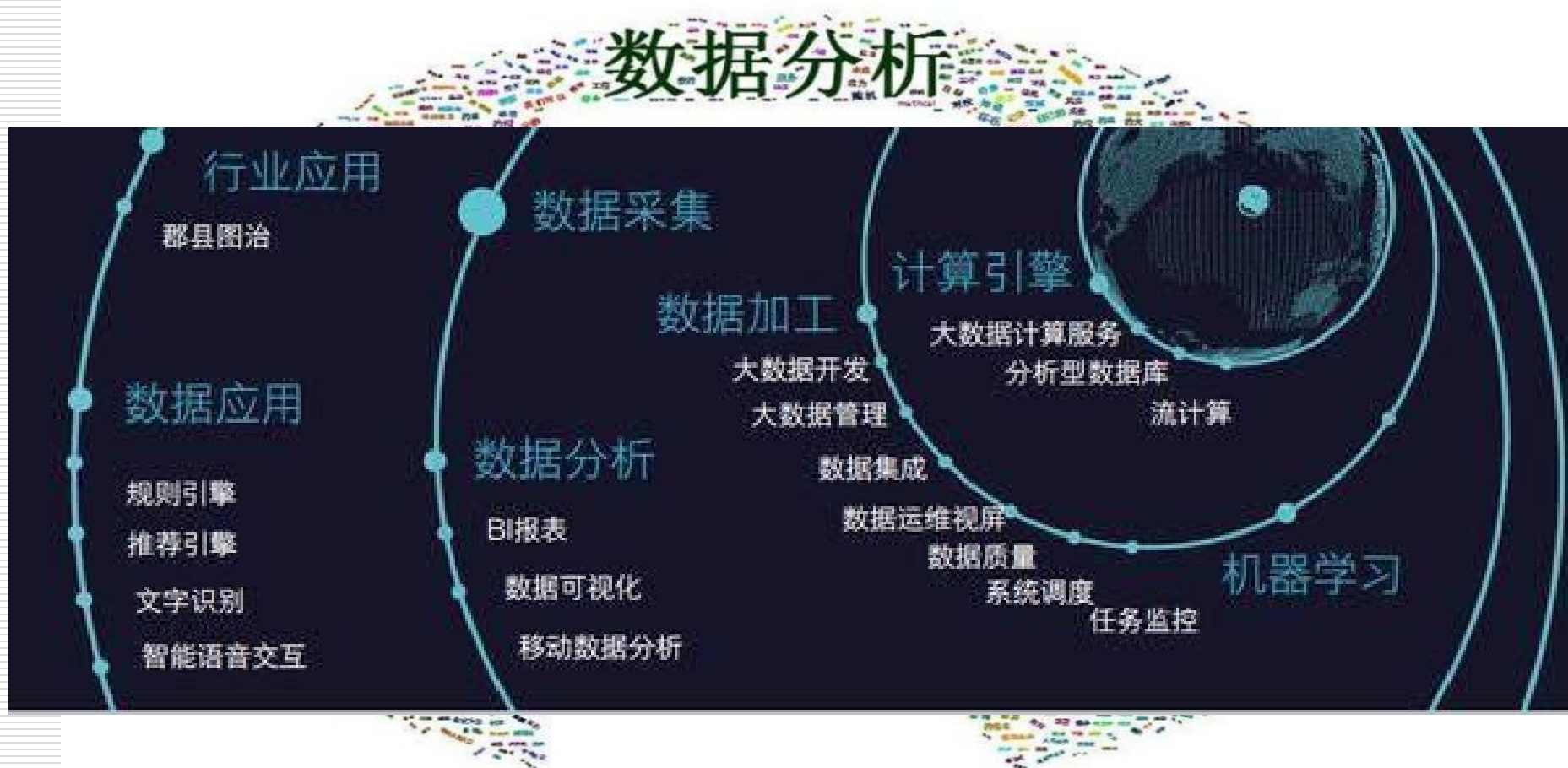
很多人还没搞清楚
什么是PC互联网，
移动互联来了
我们还没搞清楚移动
互联的时候，
大数据时代
又来了



按顺序给出所有数据的单位：**bit、Byte、KB、MB、GB、TB、PB、EB、ZB、YB、BB、NB、DB**。大型数据集，一般在**10TB**规模左右。



1. 背景



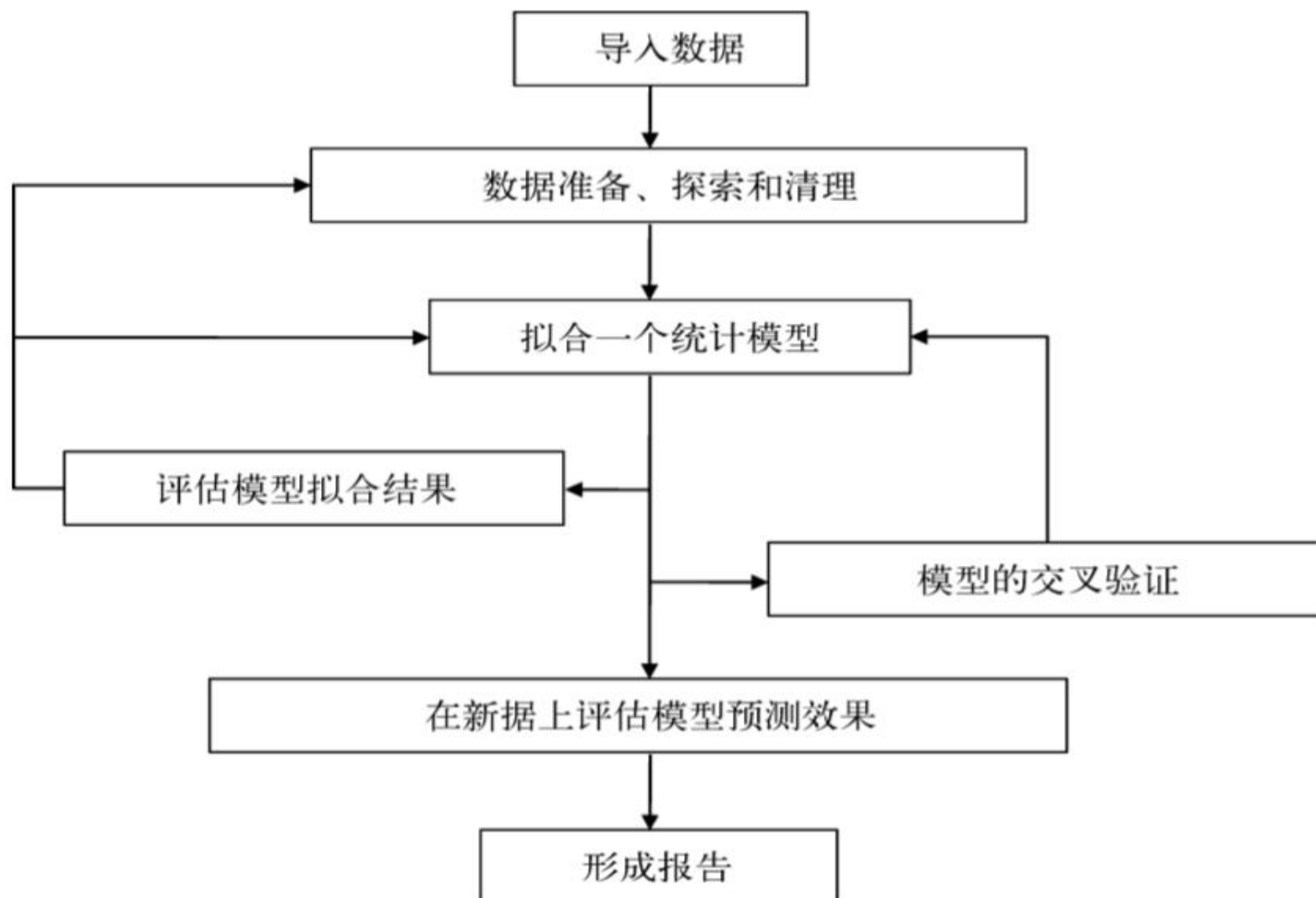
1. 背景

以数据科学家的角度学习R:

- 获取数据（从各种数据源将数据导入程序）；
- 整理数据（编码缺失值、修复或删除错误数据、将变量转换成更方便的格式）；
- 注释数据（记住每段数据的含义）；
- 总结数据（通过描述性统计量了解数据的概况）；
- 数据可视化（一图胜千言）；
- 数据建模（解释数据间的关系，检验假设）；
- 整理结果（创建具有出版水平的表格和图形）。



1. 背景



2. R语言介绍

为什么是R:

- 多数商业统计软件价格不菲，投入成千上万美元都是可能的。而R是免费的！
- R是一个全面的统计研究平台，提供了各式各样的数据分析技术。几乎任何类型的数据分析工作皆可在R中完成。
- R拥有顶尖水准的制图功能。如果希望复杂数据可视化，那么R拥有最全面且最强大的一系列可用功能。
- R是一个可进行交互式数据分析和探索的强大平台。



2. R语言介绍

- **R**可以轻松地从各种类型的数据源导入数据，包括文本文件、数据库管理系统、统计软件，乃至专门的数据仓库。
- **R**是一个无与伦比的平台，在其上可使用一种简单而直接的方式编写新的统计方法。
- **R**囊括了在其他软件中尚不可用的、先进的统计计算例程。
- 有各式各样的**GUI**（**Graphical User Interface**，图形用户界面）工具通过菜单和对话框提供了与**R**语言同等的功能。
- **R**可运行于多种平台之上，包括**Windows**、**UNIX**和**Mac OS X**。这基本上意味着它可以运行于你所能拥有的任何计算机上。



2. R语言介绍

- 本课程主要参考书

游皓麟

R语言预测实战

电子工业出版社

2016年10月

**Robert I.
Kabacoff**
著

高涛 肖楠
陈刚译

R语言实战

人民邮电出版社

2013年



2. R语言介绍

- 考核方式：通过/不通过
 - ❑ 出勤及参与讨论 20%
 - ❑ 平时练习20%
 - ❑ 实际问题求解60%

根据自己的兴趣，发现学科或生活中的大数据，进行性数据分析（回归、聚类、人工神经网络等方法不限），挖掘出新的信息，最后提交一个研究报告或基于**R-Shiny**的应用程序。并进行成果展示。



2. R语言介绍

R的获取和安装

- R的获取: <https://cran.r-project.org/>
- IDE: RStudio (<https://www.rstudio.com/>)
(推荐)

R语言版本的升级——利用Rstudio

```
install.packages("installr")
```

```
library(installr)
```

```
updateR()
```



一些库文件安装时无法下载，修改下面的文件：

C:\Program Files\R\R-4.2.1\etc\Rprofile.site

.....

set a CRAN mirror

local({r <- getOption("repos")

r["CRAN"] <-

"http://mirrors.tuna.tsinghua.edu.cn/CRAN"

options(repos=r))

.....



2. R语言介绍

【操作练习】（10分钟） 在自己的笔记本上安装R和RStudio。

提示：

1) 下载安装（<https://cran.r-project.org/> 和 <https://www.rstudio.com/>）

方法参考：

<https://baijiahao.baidu.com/s?id=1731504073327371440&wfr=spider&for=pc>

或

2) 用已下载好的安装包（学堂云上已提供）。



2. R语言介绍

R的工作目录：

当前的工作目录是R用来读取文件和保存结果的默认目录。

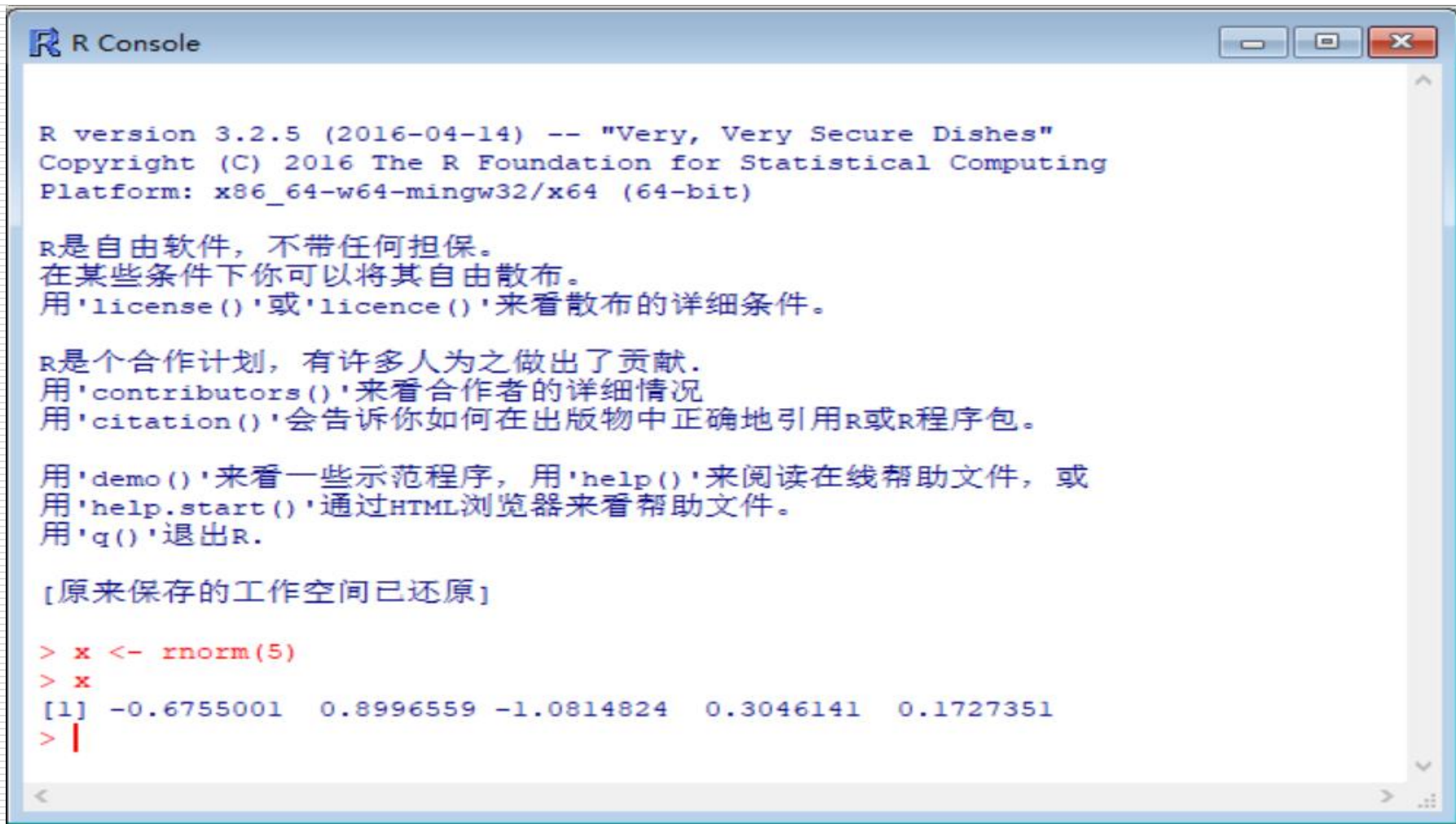
- 函数`getwd()`：查看当前的工作目录
- 函数`dir.create()`来创建新目录
- 函数`setwd()`：设置新的工作目录（存在的目录）

【操作练习】（5分钟） 在R下查看当前工作目录，并建立和设置自己的工作目录（例如`setwd("D:/R")`）。



2. R语言介绍

R的运行方式——命令行（刚才的练习）



```
R Console

R version 3.2.5 (2016-04-14) -- "Very, Very Secure Dishes"
Copyright (C) 2016 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.

[原来保存的工作空间已还原]

> x <- rnorm(5)
> x
[1] -0.6755001  0.8996559 -1.0814824  0.3046141  0.1727351
> |
```

2. R语言介绍

R的运行方式（推荐使用）——在Rstudio下单行运行或运行脚本（`source('E:/Rtest/Test.R')`）。

脚本名称：Test.R

内容：

```
x=seq(0,20,by=0.5)
```

```
y=dchisq(x,3)
```

```
plot(x,y,type="l",col="blue")
```

```
text(15,0.15,"d.f.=3",cex=2,col="red")
```

【操作练习】（15分钟）在Rstudio下编辑Test.R并运行Test.R (包括单步运行和整体运行)。



2. R语言介绍



Break Time
10 minutes

www.aijia.com

By: xiaomuzilyx No: 201402220448B29588088



2. R语言介绍

R包含海量**packages**，囊括了其他软件中尚不可用的、先进的统计计算程序。

- 默认包: **base, datasets, utils, grDevices, graphics, stats, methods**
- 查看已加载包: **search()**
- 安装包: **install.packages("packages name")**
- 载入包: **library("package name")**
- 查看包: **help(package="package name")**



2. R语言介绍

【操作练习】(10分钟)使用一个新的包，学习R安装、载入和查看包的方法。

help.start() #打开帮助文档首页，并查阅其中的
“Introduction to R”

install.packages("vcd") #安装 vcd 包(可视化类别数据)

help(package = "vcd") #列出此包中可用的函数和数据集

library("grid") #vcd依赖grid包

library ("vcd") # 载入vcd包

help(Arthritis) # 阅读数据集 **Arthritis** 的描述

Arthritis #显示数据集 **Arthritis** 的内容（直接输入一个对象的名称将列出它的内容）

example(Arthritis) # 运行数据集 **Arthritis** 自带的示例。



2. R语言介绍

- R语言常用的包

实例： 爬取网页数据



2. R语言介绍

获取帮助:

- 函数**help.start()**会打开一个浏览器窗口，可以查看入门和高级的帮助手册、常见问题集，以及参考材料。
- 函数**RSiteSearch()**可在在线帮助手册和**R-Help**邮件列表的讨论存档中搜索指定主题，并在浏览器中返回结果。
- 由函数**vignette()**函数返回的**vignette**文档一般是**PDF**格式的实用介绍性文章。不过，并非所有的包都提供了**vignette**文档。可用“?”代替“**help**”命令。
- 百度



2. R语言介绍

R的其他特点:

- 区分大小写
- 赋值符号 (`<-`) 或者 (`->`)
- 命令提示符 (`>`)
- 小巧而精悍
- 全面的统计研究平台, 几乎任何类型的数据分析需求都可以完成
- 画图功能十分强大

【操作练习】（5分钟） 体验R语言的绘图功能 `demo(graphics)`。对R语言的绘图功能有一个基本概念。



2. R语言介绍

- 跨平台，且易于扩展(C++/python/spark等)
- 没有多行注释。单行注释使用“#”
- 将一个值赋给某个向量、矩阵、数组或列表中一个不存在的元素时，R将自动扩展这个数据结构以容纳新值
- 下标从1开始。
- R中没有标量。标量以单个向量的形式出现。
- 变量不需声明。它们在首次被赋值时自动生成。



2. R语言介绍

【操作练习】（5分钟） 认识一下R语言中的变量

```
x <- c(8,6,4)
```

```
x [7] <- 10
```

#自动扩充

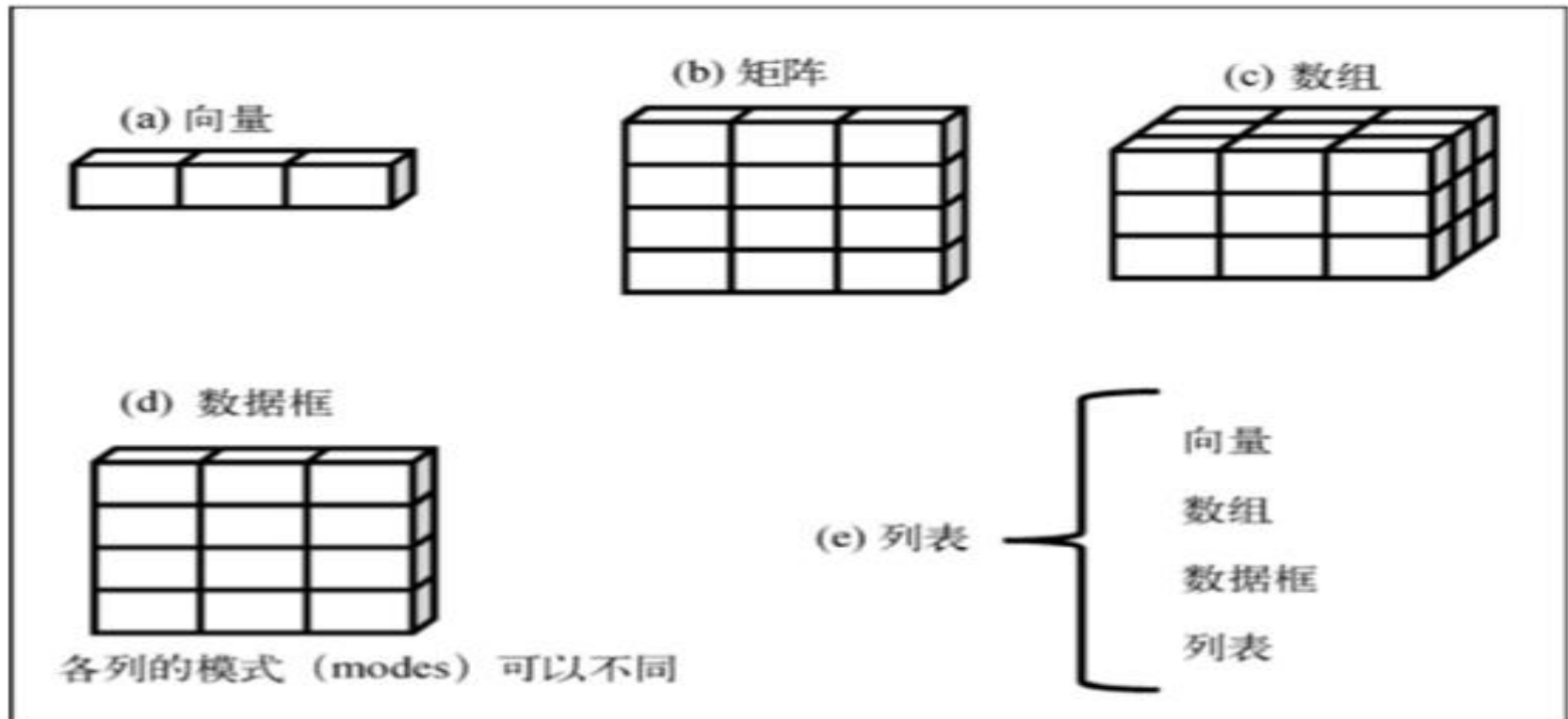
```
x
```



3. R语言的基本数据类型

在R中，对象（**object**）是指可以赋值给变量的任何事物，包括常量、数据结构、函数，甚至图形。

下图是R中的数据结构。



3. R语言的基本数据类型

(a) 向量

向量是用于存储数值型、字符型或逻辑型数据的一维数组。执行组合功能的函数 **c()** 可用于创建向量。标量是只含一个元素的向量，例如 **f <- 3**、**g <- "US"** 和 **h <- TRUE**。它们用于保存常量。



3. R语言的基本数据类型

【操作练习】（5分钟） 变量的创建及访问

```
a <- c(1, 2, 5, 3, 6, -2, 4)
```

```
a
```

```
b <- c("one", "two", "three " )
```

```
b
```

```
d <- c(TRUE, TRUE, TRUE, FALSE, TRUE)
```

```
a[3]
```

```
a[c(1,3,5)] #访问第一、三、五的元素
```

```
a[2:6]      #访问第二到第六的元素
```



3. R语言的基本数据类型

(b) 矩阵

矩阵是一个二维数组，只是每个元素都拥有相同的数据类型（数值型、字符型或逻辑型）。一般创建矩阵的格式为：

```
mymatrix <- matrix(vector,  
                     nrow=number_of_rows,  
                     ncol=number_of_columns,  
                     byrow=logical_value,  
                     dimnames=list(char_vector_rownames,  
                                     char_vector_colnames))
```

其中**vector**包含了矩阵的元素，**nrow**和**ncol**用以指定行和列的维数，**dimnames**包含了**可选的**、以字符型向量表示的行名和列名。选项**byrow**则表明矩阵应当按行填充（**byrow=TRUE**）还是按列填充（**byrow=FALSE**），默认情况下按**列**填充。



3. R语言的基本数据类型

【操作练习】（5分钟） 矩阵的创建及访问

#创建一个5*4的矩阵, 值为1-20,默认按列填充

```
y <- matrix(1:20, nrow = 5, ncol = 4)
```

```
y      #访问矩阵y
```

```
y[1,c(3,4)]  #访问矩阵第一行, 第3、4列
```



3. R语言的基本数据类型

【操作练习】（5分钟） 矩阵的创建及访问

```
cells <- c(1,2,3,4)#
```

```
rnames <- c("R1", "R2")
```

```
cnames <- c("C1", "C2")
```

```
mymatrix <- matrix(cells, nrow = 2, ncol = 2,  
byrow = TRUE, dimnames = list(rnames,  
cnames))#创建2*2的矩阵，并按行填充cells的元素，  
行名为rnames，列名为cnames
```

```
mymatrix #访问整个矩阵
```

```
mymatrix[2,] #访问矩阵的第2行
```

```
mymatrix[2,2] #访问矩阵的第2行，第2列的元素
```



3. R语言的基本数据类型

(c) 数组

数组 (**array**) 与矩阵类似，但是维度可以大于2。创建数组的形式如下：

```
myarray <- array(vector, dimensions, dimnames)
```

其中 **vector** 包含了数组中的数据， **dimensions** 是一个数值型向量，给出了各个维度下标的最大值，而 **dimnames** 是可选的、各维度名称标签的列表。



3. R语言的基本数据类型

【操作练习】（5分钟） 数组的创建及访问

```
dim1 <- c("A1", "A2")
```

```
dim2 <- c("B1", "B2", "B3")
```

```
dim3 <- c("C1", "C2", "C3", "C4")
```

```
z <- array(1:24, c(2, 3, 4), dimnames = list(dim1,  
dim2, dim3))
```

```
z
```

```
z[1,2,3]
```

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



3. R语言的基本数据类型

(d) 数据框

由于数据有多种数据类型，无法将此数据集放入一个矩阵。在这种情况下，使用需要数据框。

例 病人数据

PatientID (病人编号)	1	2	3	4
AdmDate (入院时间)	10/15/2009	11/01/2009	10/21/2009	10/28/2009
Age(年龄)	25	34	28	52
Diabetes (糖尿病类型)	Type1	Type2	Type1	Type1
Status (病情)	Poor	Improved	Excellent	Poor

3. R语言的基本数据类型

(d) 数据框

创建数据框：

```
mydata <- data.frame(col1, col2, col3,...,  
row.names=)
```

其中的列向量 **col1, col2, col3,...** 可为任何类型（如字符型、数值型或逻辑型）。每一列数据的类型必须唯一。

在R中，实例标识符（**case identifier**）可通过数据框操作函数中的**rowname**选项指定。在病例数据中，病人编号（**patientID**）用于区分数据集中不同的个体——即**关键字**。



3. R语言的基本数据类型

【操作练习】（5分钟） 数据框的创建及访问

```
patientID <- c(1, 2, 3, 4)
```

```
age <- c(25, 34, 28, 52)
```

```
diabetes <- c("Type1", "Type2", "Type1", "Type1")
```

```
status <- c("Poor", "Improved", "Excellent", "Poor")
```

```
patientdata <- data.frame(patientID, age, diabetes,  
status, row.names=patientID)
```

```
patientdata
```

```
patientdata[1:2] #显示前两列
```

```
patientdata[c("diabetes", "status")]
```

```
patientdata$age
```



3. R语言的基本数据类型

【操作练习】（5分钟） 理解函数table()、attach()和函数detach()

统计，类似于交叉表

```
table(patientdata$diabetes, patientdata$status)
```

或

```
attach(patientdata) #patientdata被隐藏
```

```
table(diabetes, status) #直接使diabetes, status
```

```
detach(patientdata) # 去掉对patientdata的隐藏
```



3. R语言的基本数据类型

(e) 列表

列表允许整合若干（可能无关的）对象到单个对象名下。即某个列表中可能是若干向量、矩阵、数据框，甚至其他列表的组合。使用函数 **list()** 创建列表：

```
mylist <- list(object1,object2,...)
```

还可以为列表中的对象命名：

```
mylist <- list(name1=object1,name2=object2,...)
```



3. R语言的基本数据类型

【操作练习】（5分钟） 列表的创建及访问

```
g <- "My First List"
```

```
h <- c(25, 26, 18, 39)
```

```
j <- matrix(1:10, nrow = 5)
```

```
k <- c("one", "two", "three")
```

```
mylist <- list(title = g, ages = h, j, k)
```

```
mylist
```

```
mylist[[2]]
```

```
mylist[["ages"]]
```

#与下面的访问形式等价

```
mylist$ages
```



3. R语言的基本数据类型

(f) 因子

- **名字型变量**是没有顺序之分的类别变量，例如糖尿病类型 **Diabetes** (**Type1**、**Type2**)
- **有序型变量**表示一种顺序关系，而非数量关系。病情 **Status** (**poor**, **improved**, **excellent**就是有序型变量。
- **连续型变量**可以呈现为某个范围内的任意值，并同时表示了顺序和数量。

类别（名字型）变量和有序类别（有序型）变量在R中称为**因子（factor）**。因子决定了分析和呈现方式。



3. R语言的基本数据类型

【操作练习】（10分钟）因子的创建及访问

```
patientID <- c(1, 2, 3, 4)
```

```
age <- c(25, 34, 28, 52)
```

```
diabetes <- c("Type1", "Type2", "Type1", "Type1")
```

```
status <- c("Poor", "Improved", "Excellent", "Poor")
```

```
diabetes <- factor(diabetes)
```

```
status <- factor(status, order = TRUE, levels=c("Poor",  
"Improved", "Excellent"))
```

```
patientdata <- data.frame(patientID, age,  
diabetes,status)
```

```
str(patientdata) #即struc
```

```
summary(patientdata)
```

函数 **factor()** 以一个整数向量的形式存储类别值，整数的取值范围是 $[1 \dots k]$ （其中 k 是名义型变量中唯一值的个数）；对于字符型向量，因子的水平默认依字母顺序创建，可以通过指定 **levels** 选项来覆盖默认排序。