

目录

一、 Intro	2
二、 Data manipulation.....	2
Lab1	3
三、	3
Lab2.....	4
Lab3	6
五、 Loops and Functions.....	6
Lab4.....	7
六、	7
七、 Distributions and Random Numbers.....	8
八、 loglikelihood	10
九、 R graphics	11
Lab 7.....	15
十、	15
十一、 Optimization	16
Lab 8.....	17
十二.....	17
十三.....	18
十四、 Logistic Regression	19
十五、	21
十六、 Nelder-Mead	22
十七、 Sampling	23
十八、 MCMC	24
十九、 Metroplis	24
二十、 Metroplis-Hastings	25
二十一、 Gibbs	25
二十二、 Bayesian Inference with MCMC	27
二十三、	28
二十四、 Summarize + Data-Import-Export	30
二十五、 Computational-Linear-Algebra	31
二十六、	32
期末作业.....	33

一、Intro

```
c()  
rep( ,each= )  
length()  
[ ]  
seq(,,)  
?  
??  
example()
```

二、Data manipulation

which() 提供角标
连接符 & |
判断 == != <>
ls() 列出变量
rm() 删除变量(找不回来)
rm(list=ls())
a+c

sqrt()
向量化 vectorization
12 维向量+3 维向量: 3 维向量循环加
12 维向量+5 维向量: 有问题但依然计算
↓
检验向量的特征:
length()
class() 类属性
typeof() 数据存储属性

矩阵 (按列排)
> matrix(c(22,31,17,38,16,7),3,2)
[, 1] [, 2]
[1,] 22 38
[2,] 31 16
[3,] 17 7

dim()维度 (对矩阵管用)
nrow()

ncol()

mat[1,]等价 mat[1,1:2]

byrow=TRUE 按行排列

which(arr.ind=TRUE)按坐标

t(mat) transpose 转置

%*%矩阵乘法

*对应元素相乘

solve() 求逆 (singular 奇异的→不可求逆)

det() determinant 求行列式

eigen() 特征值

prod() product 连乘

Lab1

all.equal()

as.numeric (转化)

is.numeric (检验)

qr()\$rank

cbind()

rbind()

点评

==只能比较 int 字符 逻辑

浮点型用 abs () 或 all.equal ()

三、

Ary=array(1:24, c(2, 3, 4))

Ary[, , 1]

Ary[, 1,]

Ary[1, ,]

Ary1*Ary2 对应元素乘法

数组只能存同一类型的数据

↓ 存储不同类型

数据框 data.frame

(与矩阵维度类似，但可以存储不同类型的数据)

(但是只能列与列之间类型不同，同一列类型相同)

```
Name=c("Zhang3", "Li 4", "Wang5", "Zhao6")
```

```
Age=c(19, 28, 17, 14)
```

```
Sal=c(2000, 180, 5000, 20000)
```

```
data=data.frame(Name, Age, Sal)
```

```
> Sex=c("M", "F", "F", "M")
```

```
> cbind(data, Sex)
```

```
data$Sal
```

```
data2=data.frame(a=c(1, 2), b=c(3, 4))
```

```
as.matrix(data2)
```

↓ 真 · 存储不同类型

list()

```
info=list(zhang3=17, li4=28, wang5=99)
```

```
length(info)
```

```
names(info)
```

info[[1]] 两个方括号

```
unlist(info)
```

```
lapply(info,log) list 仅处理第一级的数据
```

```
rapply(info,log) recursive 可递归的 每一级都处理 (默认还会 unlist, 除非 how="replace")
```

```
rapply(info, function(x) x^2, how="replace")
```

Lab2

```
a=c(1,3,4,5,6,7,8,9,3)
```

```
dim(a)=c(3,3)
```

```
matrix(b,nrow=2,ncol=2)
```

```
array(b,dim=c(2,2))
```

```
array(d,dim=c(2,2,3))
```

四、如何在复杂数据结构上作运算

rowMeans(mat) 只能对矩阵做操作

```
colMeans(mat)
```

`mean(a)`

`var(a)`

`median(a)`

`apply()` 函数 只作用在矩阵上

`apply(mat, 2, mean)` 和 `colMeans(mat)` 效果一样

`apply(ary, 3, mean)`

`apply(ary, c(1, 3), mean)`

`apply(ary, c(1, 3), max) - apply(ary, c(1, 3), min)`

↓

`mymaxmin=function(x) max(x)-min(x)` 定义函数

`apply(ary, c(1, 3), mymaxmin)`

↓

`apply(ary, c(1, 3), function(x) max(x) - min(x))` 匿名函数

对于数组

`sum(lst$a)`

`sum(lst$b)`

`sum(lst$c)`

↓

`lapply(lst, sum)`

(在矩阵中, `var()` 计算的是协方差矩阵, 因此对于矩阵, 要 `as.numeric()`)

`var(as.numeric(c))`

`apply(ary, 3, function(x) var(as.numeric(x)))`

`rapply()`

`lstd3=NULL` 去掉 `list` 中的某一项

`unlist(lst2)-unlist(lst3)` `list` 不能每一项直接相加减

复杂数据结构不能进行某种操作时, 转化为简单数据结构

或者:

`maply(function(x, y) x-y, lst2, lst3)`

或者:

`maply("-", lst2, lst3)` 减号也是函数

`"%+%" = function(x,y) maply("+",x,y)`

`lst2 %+% lst3` 左边是 `x`, 右边是 `y`

(补充)

把矩阵拉直：

```
matrix(x(mat, 24, 1))  
matrix(t(mat), 24, 1)
```

```
matrix(unlist(b), 4, 3)  
do.call(cbind, b)
```

Lab3

apply 函数族练习

```
# 自定义函数 myFUN, 第一个参数 x 为数据  
# 第二、三个参数为自定义参数, 可以通过 apply 的...进行传入。  
> myFUN<- function(x, c1, c2) { c(sum(x[c1],1), mean(x[c2])) }  
# 把数据框按行做循环, 每行分别传递给 myFUN 函数, 设置 c1,c2 对应 myFUN 的第二、三个参数  
> apply(x,1,myFUN,c1='x1',c2=c('x1','x2'))
```

set.seed()用于设定随机数种子，一个特定的种子可以产生一个特定的伪随机序列，这个函数的主要目的，是让你的模拟能够可重复出现，因为很多时候我们需要取随机数，但这段代码再跑一次的时候，结果就不一样了，如果需要重复出现同样的模拟结果的话，就可以用 set.seed()。在调试程序或者做展示的时候，结果的可重复性是很重要的，所以随机数种子也就很有必要。

(如果空，则是当前时间，起到了随机的效果)

(数字随意，但小数部分无效，同样的整数部分就会出现同样的结果)

(只对运行该命令后的第一次随机产生结果有效)

五、Loops and Functions

1、判断

```
tolower("FENG") == "feng"  
"FENG" == toupper("feng")
```

all.equal()

any(rec < 60) 判断至少一条满足条件，只输出一个 TRUE、FALSE
all(rec < 60) 所有

R 中 0, 1 对应 FALSE, TRUE

```
sum(rec<60)

' zhang3' %in% c("zhang4", "zhang5", "zhang3")
tolower(' zhang3') %in% tolower(c("zhang4", "zhang5", "zhang3"))
")
```

2、将以上判断用在 **if**, **else** 中

3、函数

%%整除

函数里不能有多个 return，并在 return 处结束

Lab4

检测输入类型

另一项作业：

上下四分位数、最大最小值、偏度、峰度、能想到的

1st Qu. 空格变点，前面有 x?

x1 x2 只能输出一个？

满足四原则（如果输入有问题，要提醒）

虚部也要考虑

作业：

修改作业

（避免其他类型

多个 abc 同时求根

a=0）

六、

对数稳定性强

对于很小的数，计算机告诉你连乘的结果是 0

所以我们用对数相加

pro(1: 500)

```
sum(log(1: 500))
```

```
a=rep(0.0001, 1000)
prod(a)
sum(log(a))
```

捕获异常：

```
try ()
```

第一个参数为调用的方法，第二个参数为是否显示异常消息

```
xTry = try(solve(x), silent = TRUE)
if(is(xTry, "try-error"))
{
  return(NA)
}
```

factorial 阶乘

递归：

```
mygamma=function(n)
{
  if(n>1)
  {
    out=mygamma(n-1)*(n-1)
  }
  else if(n ==1)
  {
    out=1
  }
  return(out)
}
```

log(gamma()) 和 lgamma()

算法不同

```
source("")
getwd()
setwd()
dir()
```

七、 Distributions and Random Numbers

```
set.seed(1)
```

runif(1)

随机数种子只能用一次，之后就被重置

以 d 开头，后面加分布的英文缩写，告诉你概率密度

dnorm()

```
x= seq(-5, 5, 0.01)
fx = dnorm(x)
plot(x, fx)
```

分布函数:

pnorm()

```
plot(x, Fxme, type="l", col = "purple")
```

Z=F(X)，那么 Z 是[0,1]上的均匀分布:

对比: x2=rnorm(1000) Fx2=pnorm(x2)

mean(fx1)

mean(fx2)

var(fx1)

var(fx2)

hist(fx1)

hist(fx2)

qnorm()

随机数、密度、分布、分位数

r d prob quantile

n q p

```
fx = runif(1000)
```

```
x=qnorm(fx)
```

直方图 hist

```
loglikelihood = function(x,miu,sigma){
  sum(dnorm(x,miu,sigma,log = TRUE))
}
```

求对数密度

sum(log(dnorm(x,miu,sigma)))也可以，但不推荐，因为先算密度

作业:

黑板上 4 个图

在 R 里面找 20 个密度函数，画 20 个四个图

连续: β 分布、柯西分布、卡方分布、指数分布、F 分布、Gamma 分布、对数正态分布、正态分布、t 分布、均匀分布、韦布尔分布

离散: 二项式分布、几何分布、超几何分布、负二项式分布、泊松分布

par()设定图形的参数

op=par(mfrow=c(2,2))

plot()

plot()

plot()

plot()

举例:

将图形参数的设定赋值给 op

par(op)

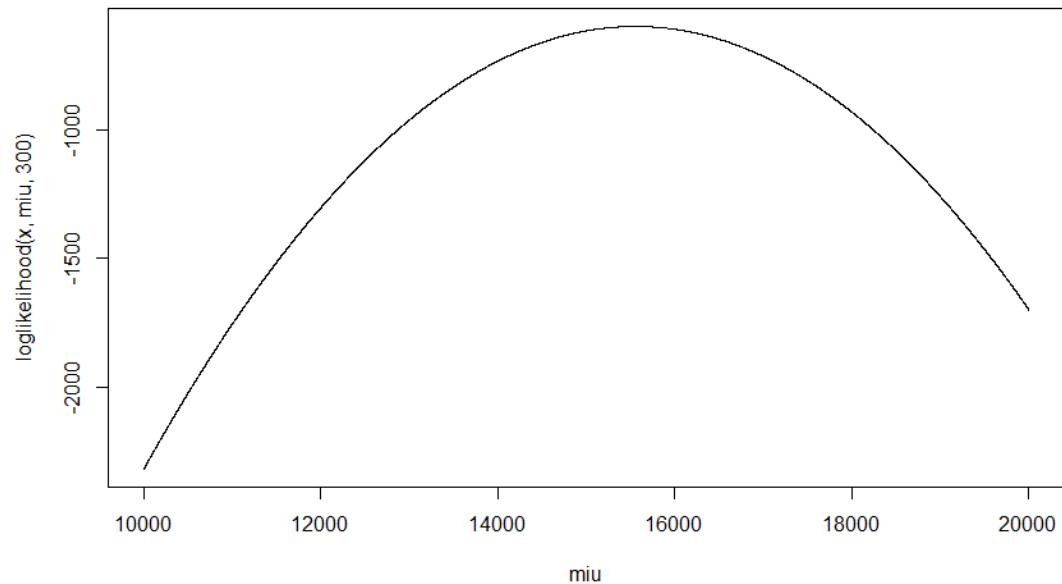
作用是把当前图形设备恢复到之前的设置,否则格式都是四张图放一起输出

plot(xlim = c(-5,5)) 设置横轴范围

八、 loglikelihood

循环

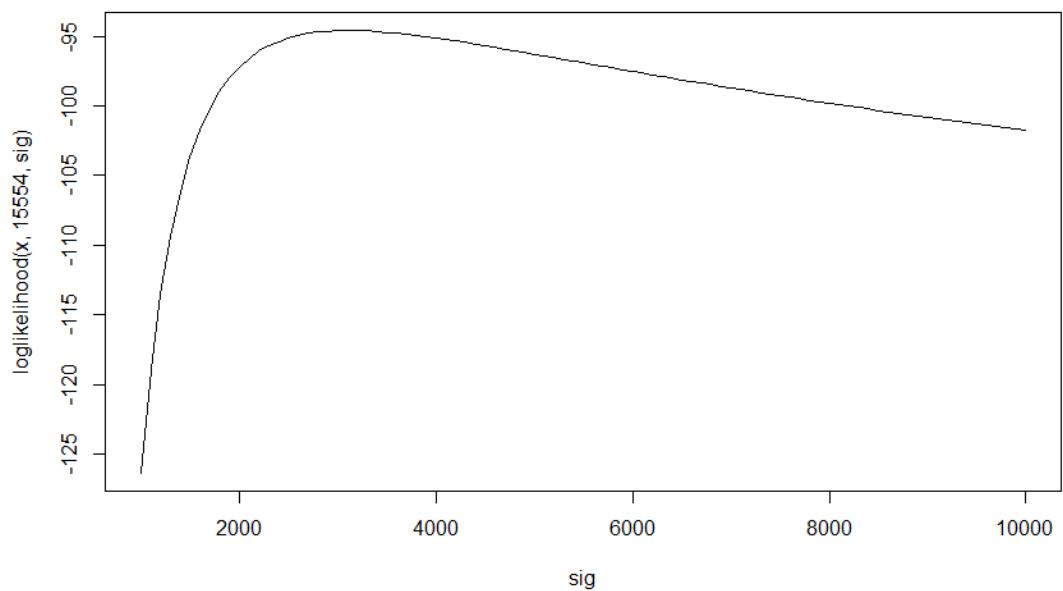
loglikelihood



which.max(LS)

muS(which.max(LS))

15554



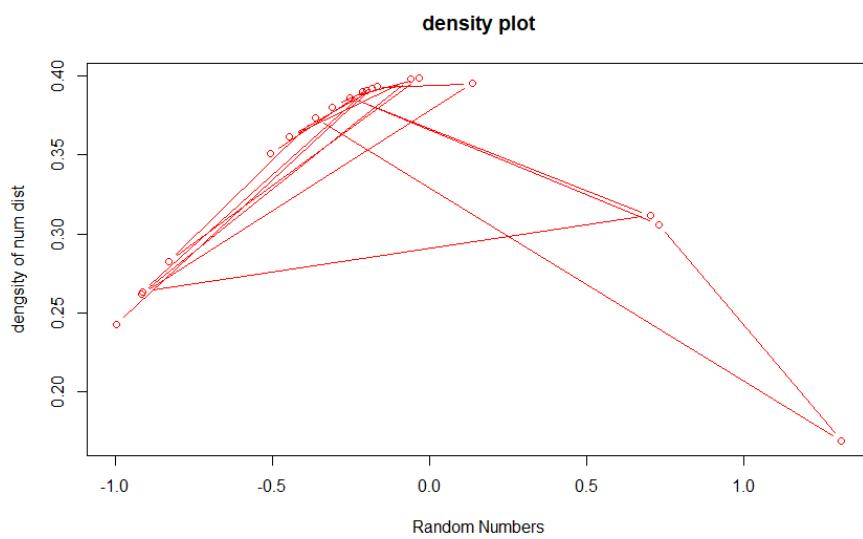
3100

九、R graphics

CRAN Task View

device axis horizontal axis vertical axis label curve

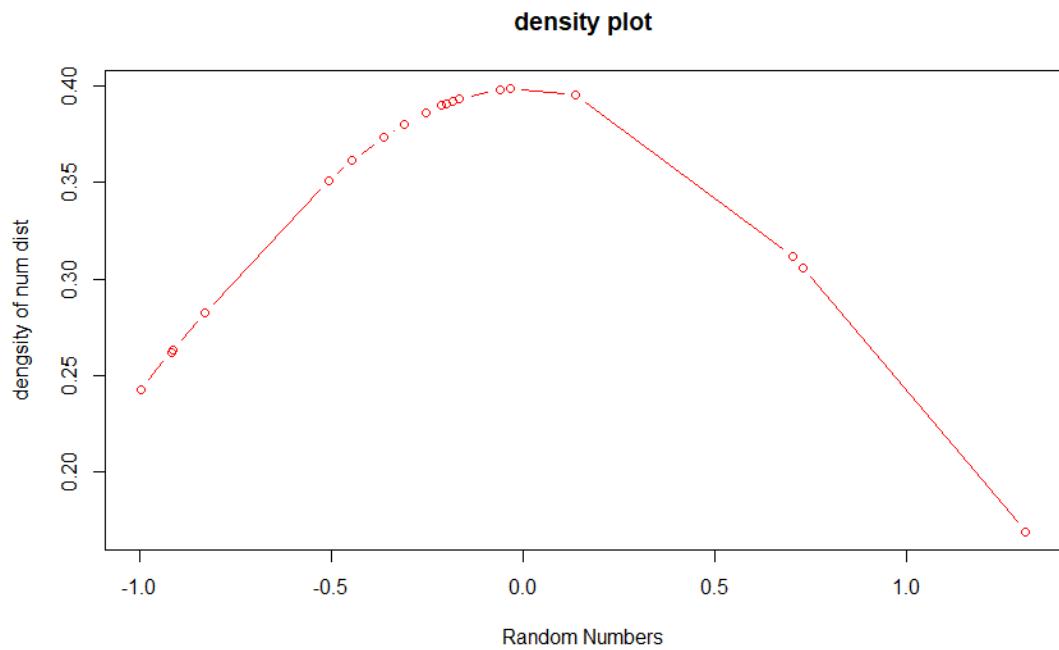
```
xlab ylab main col type = "l" ("b")
plot(x2, fx2, xlab = "Random Numbers", ylab = "density of num
dist", main = "density plot", col = "Red", type = "b")
```



按先后顺序连的线

`sort` — 排序 `order` — 排序后原来的下标位置

```
plot(sort(x2), fx2[order(x2)], xlab = "Random Numbers", ylab = "density of num dist", main = "density plot", col = "Red", type = "b")
```



`lwd` `xlim`

```
par(mfrow = c(1,2))  
dev.copy2pdf(file = "plot.pdf")
```

```
png(file="plot.png")
```

画图程序

```
dev.off()
```

`las` = 0 (1、2、3)

`label axis` 颈椎

`adj` = (0-1)

`adjust`

标签位置 0 最左 1 最右

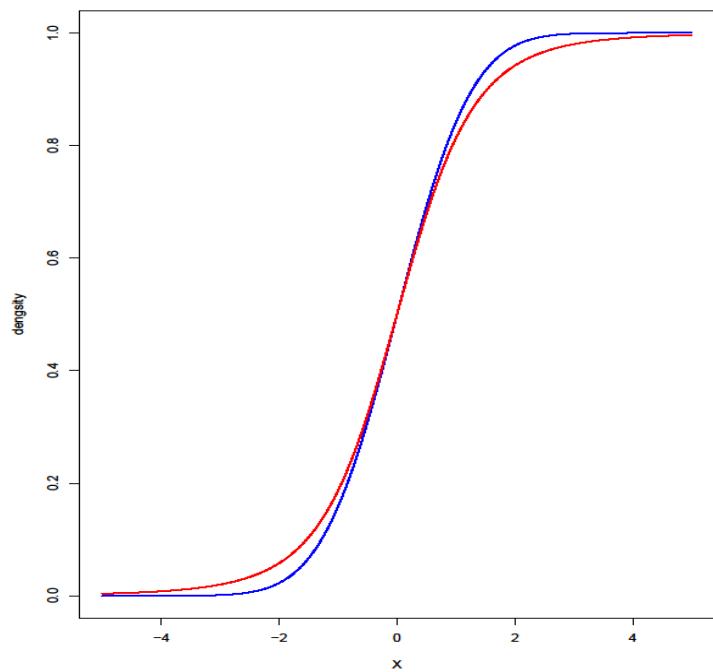
`bty` = "o"(l、7、n、c、u、])

`box type`

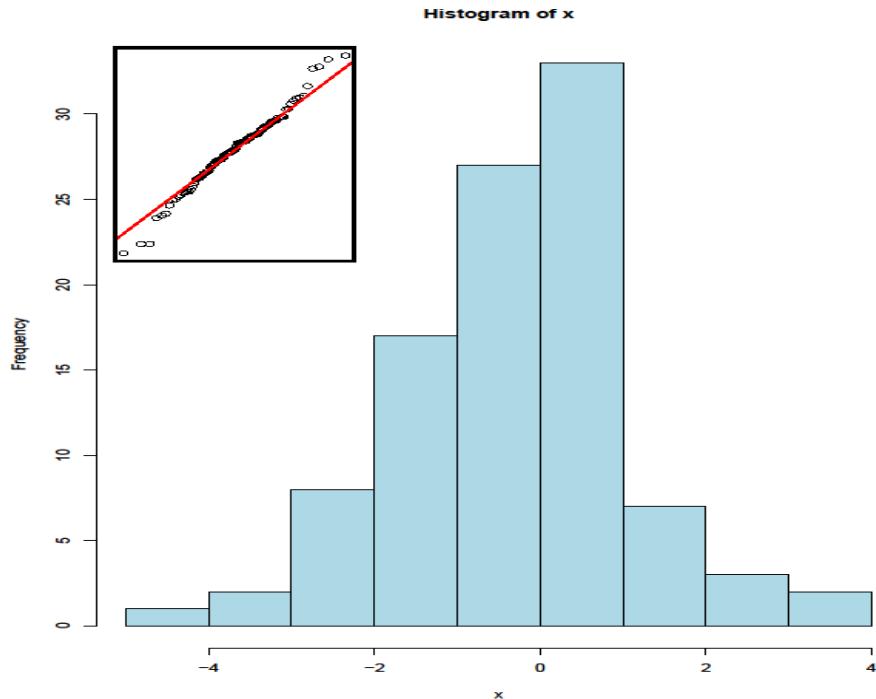
箱子类型(根据样子)

图层叠加

```
plot(x, ft, xlab = "X", ylab = "density", col = "blue", type = "l")  
points(x, fn, col = "red", type = "l", lwd = 1)
```



```
x = rt(100, 4)
hist(x, col = "light blue")
op = par(fig = c(0.02, 0.42, 0.53, 0.99), new = TRUE)
qqnorm(x, xlab = "", ylab = "", main = "", axes = FALSE)
qqline(x, col = "red", lwd = 2)
box(lwd=3)
```

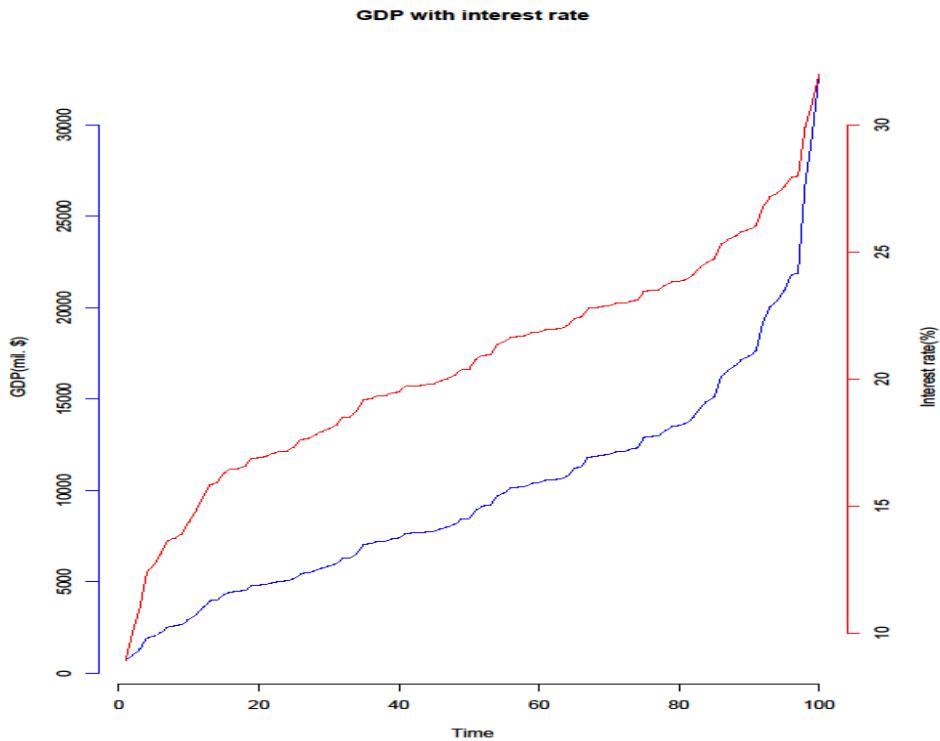


双纵轴

```

set.seed(1)
par(mar=c(5, 5, 5, 5))
x1<-sort(rnorm(100, mean=20, sd=5))
x2<-x1^3 # this is GDP
plot(x2, axes=FALSE, type="l", col="blue", xlab="", ylab="")
axis(1)
axis(2, col="blue")
par(new=TRUE)
plot(x1, , axes=FALSE, type="l", col="red", xlab="", ylab="")
axis(4, col="red")
mtext(c("Time", "GDP(mil. $)", "Interest rate(%)" ), side=c(1, 2, 4), line=3)
title("GDP with interest rate")

```



Lab 7

points () 和 lines () 效果差不多
 pie3D 不专业，因为视角容易误导
 要有人文关怀，即便黑白也应该能看懂（通过粗细）

十、

组织一个切分的图形

```
split.screen(c(2, 1))
split.screen(c(1, 2), 2)
screen(1)
plot
screen(3)
screen(4)
```

split 之后想还原设置，就 close，前面的设置就废掉了（window () 是再开一个窗，不开画不了）

增加数学符号:

```
text(x=2, y=.35, expression(paste(frac(1, sqrt(2*pi))*sigma), e  
xp(-frac(1, 2*sigma^2)*((x-mu)^2))))
```

expression 写法示例:

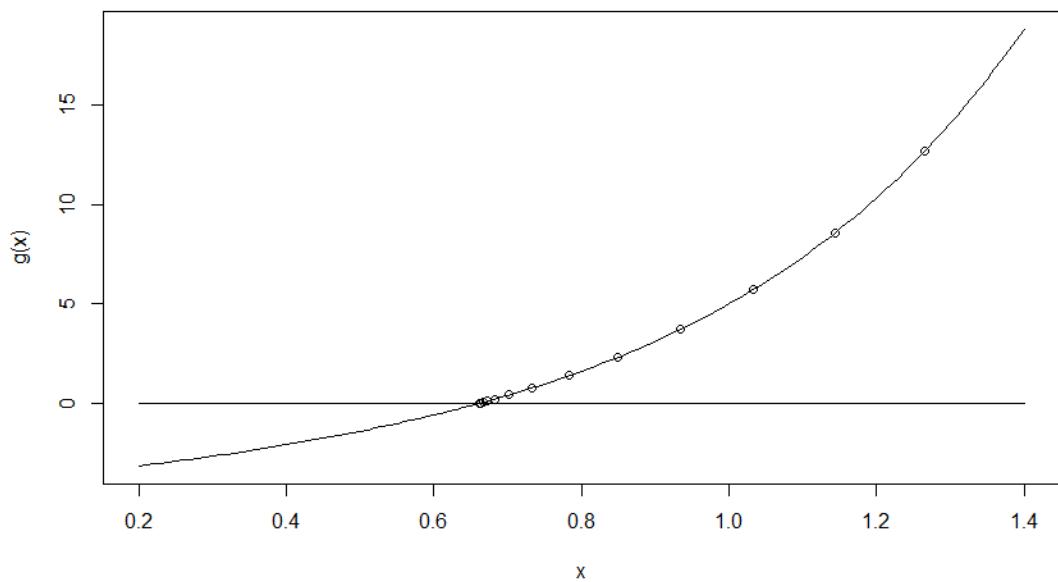
```
demo(plotmath)
```

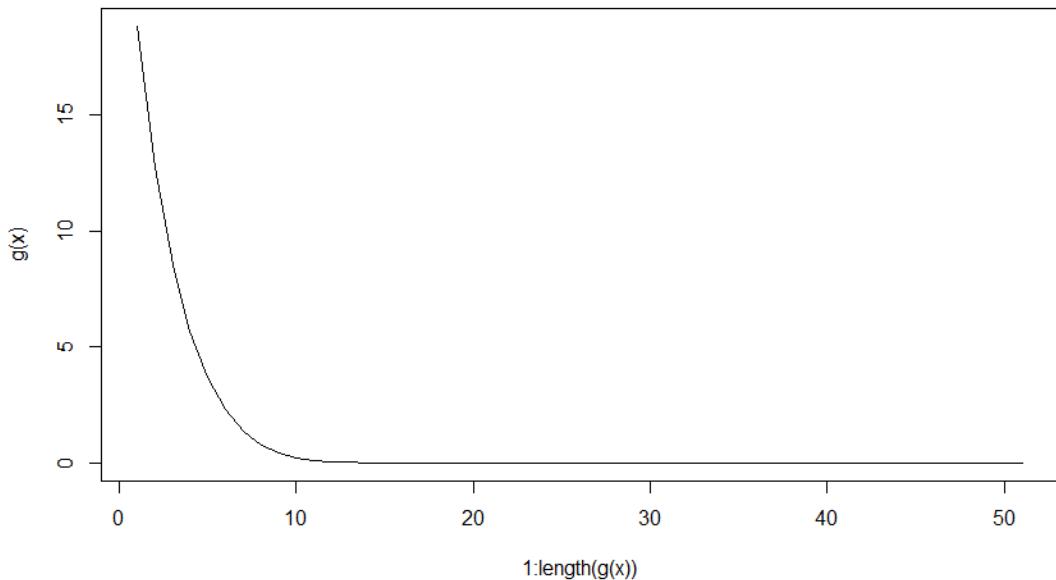
作业 1: 网格对齐数字

作业 2: 如何画

十一、 Optimization

1、牛顿迭代





2、运用牛顿迭代求函数最优化（local）

作业：
对数似然函数求极值

Lab 8

expression() 表达式（不是函数）
 deriv() 求导(输入表达式,输出函数)
 D 求导(输入表达式,输出表达式)

options(digits=10) 显示小数点后十位

browser() (断点) c continue(下一步,再迭代一次)

十二

多维牛顿迭代

十三

DGP (data generating process)

$$\begin{array}{l}
 \text{DGP} \\
 y_i = \alpha + \beta x_i + \varepsilon_i \\
 \begin{array}{c} y_i \ x_i \rightarrow \boxed{\alpha, \beta} \\ \uparrow \quad \uparrow \\ \hat{\alpha} \quad \hat{\beta} \end{array} \\
 \left[\begin{array}{l}
 1^{\circ} \alpha = 5, \beta = 3 \\
 2^{\circ} \varepsilon \sim N(0, 0.1) \\
 3^{\circ} x_i \sim \text{runif}(100) \\
 4^{\circ} y_i = \alpha + \beta x_i + \varepsilon_i \\
 5^{\circ} \{y_i, x_i\} \rightarrow \hat{\beta}, \hat{\alpha}
 \end{array} \right]
 \end{array}$$

$$\begin{array}{l}
 1^{\circ} \text{Find model } y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \dots + \beta_p x_{pi} + \varepsilon_i \\
 2^{\circ} \left\{ \begin{array}{l} \text{Data: } y \ x_1 \dots x_p \checkmark \\ \text{params: } \beta_0 \ \beta_p \\ \text{error: } \varepsilon_i \quad \varepsilon_i \sim N(0, \sigma^2) \end{array} \right. \\
 3^{\circ} ? \left\{ \begin{array}{l} 1^{\circ} \text{OLS } \arg \min \sum \| \hat{\varepsilon}_i \|^2 \\ 2^{\circ} \text{MLE} \\ 3^{\circ} \text{GMM} \end{array} \right.
 \end{array}$$

$$y_i \sim N\left(\underbrace{\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}}_{m_i}, \sigma^2\right)$$

$$L = \prod_{i=1}^n f(y_i; m_i, \sigma^2)$$

$$= \prod_{i=1}^n \frac{1}{\sqrt{2\pi}\sigma} \exp \left\{ -\frac{(y_i - [\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}])^2}{2\sigma^2} \right\}$$

$\arg \max_{\beta_0 \dots \beta_p} L \iff \arg \max_{\beta_0 \dots \beta_p} \log L$

十四、Logistic Regression

OLS	ML
$y_i = \beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip} + \varepsilon_i$ $\arg \min_{\beta_0 \dots \beta_p} \sum_{i=1}^n y_i - \hat{y}_i ^2$ OP $\hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \dots + \hat{\beta}_p x_{ip}$	$y_i \sim ?$ Data: y_i, x_i param: $\beta_0 \dots \beta_p$ Assumption: Gaussian-Markov $\arg \max_{\beta_0 \dots \beta_p} \log L = \arg \max \sum_{i=1}^n \log f(y_i; x_i, \beta_0 \dots \hat{\beta}_p, \sigma^2)$ $\frac{\partial \log L}{\partial \beta} = \sum_{i=1}^n \frac{\partial \log f(y_i; x_i, \beta, \sigma^2)}{\partial \beta} \rightarrow g(\beta, y, x)$ $\frac{\partial^2 \log L}{\partial \beta \partial \beta} = \dots \rightarrow H(\beta, y, x)$ $\beta^{(n+1)} = \beta^{(n)} - H^{-1}g$ Loop

OLS 与 ML 的区别

逻辑斯蒂回归

例子：垃圾邮件

h	y	x_1	x_2	x_3	x
1	1	1	40	1	
2	0	0	0	0	
3	1	0	20	0	
4	.	1	300	.	
5	.	0	.	.	
6	.	1	.	.	
7	0	0	0	1	

h : 样本 y : 是否为垃圾邮件 X_1 : 是否有真实姓名 X_2 : 邮件字数 X_3 : 发票

$$Y_i \sim \underbrace{X_{1i}\beta_1 + \dots + X_{Pi}\beta_P}_{\{0,1\}} = \underbrace{\beta_0 + \beta_1 X_{1i} + \dots + \beta_P X_{Pi}}_{(-\infty, +\infty)}$$

不能用线性回归模型

$$\begin{cases} P(Y_i=1) \\ P(Y_i=0) \end{cases} \quad \begin{aligned} P(Y_i|P) &= P^{y_i} (1-P)^{1-y_i} \\ L &= \prod_{i=1}^n P^{y_i} (1-P)^{1-y_i} \end{aligned}$$

使用逻辑斯蒂回归，巧妙地将连续与离散结合起来

$$P_i = \frac{1}{1 + \exp\{-\alpha_i\}} = \frac{1}{1 + \exp\{-x_i\beta\}}$$

$$L = \prod_{i=1}^n \left\{ \frac{1}{1 + \exp(-y_i \beta - x_i^\top \beta)} \right\}^{y_i} \left\{ \frac{1}{1 + \exp(y_i \beta + x_i^\top \beta)} \right\}^{1-y_i} \quad (y_i \in (-\infty, +\infty))$$

$$\frac{\partial \log L}{\partial \beta_j} = \sum_{i=1}^n \frac{\partial \log f(y_i, \beta_i, x_i)}{\partial \beta_j}$$

$$g(\beta, y, x) = H(\beta, y, x)$$

$$P(y_i | \beta_i, x_i) = P_{\beta_i}(y_i | \beta_i)$$

求偏导的方法，运用链式法则

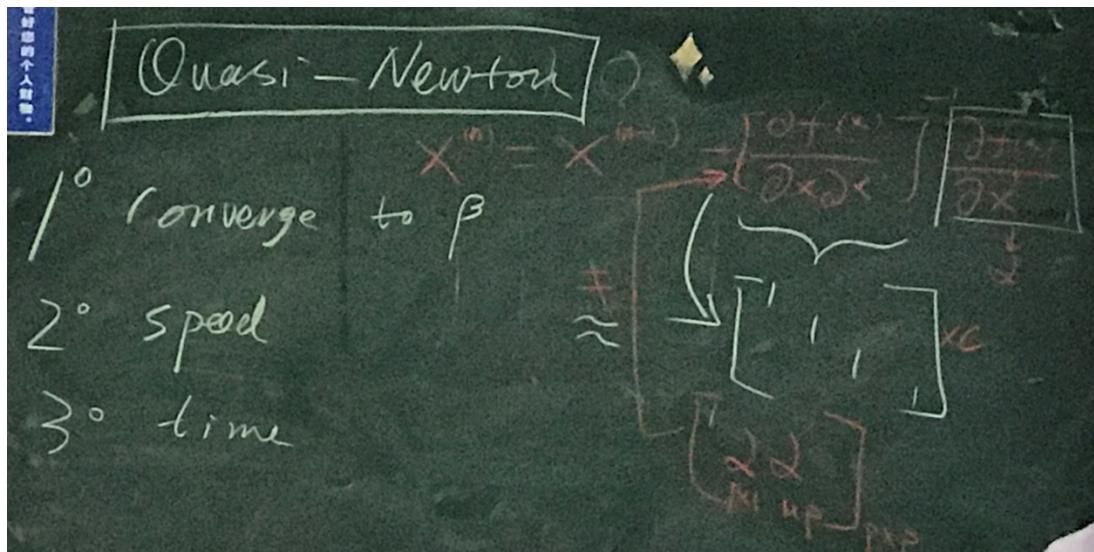
$$\frac{\partial \log L}{\partial \beta_j} = \sum \frac{\partial}{\partial \beta_j} \log f(y_i, \beta_i, x_i)$$

$$= \sum \left[\frac{\partial \log f(y_i, \beta_i, x_i)}{\partial \beta_i} \right] \times \left[\frac{\partial \beta_i}{\partial \beta_j} \right]$$

十五、

- 1、break things down
- 2、validating
- 3、visualizing

$f(x+dx)-f(x)/dx$ 估计一阶导
 “numdriv” grad(f,x0)



optim (一般是求最小值, 求最大用 control=list(fnscale=-1))

“optimx”—20 多种算法

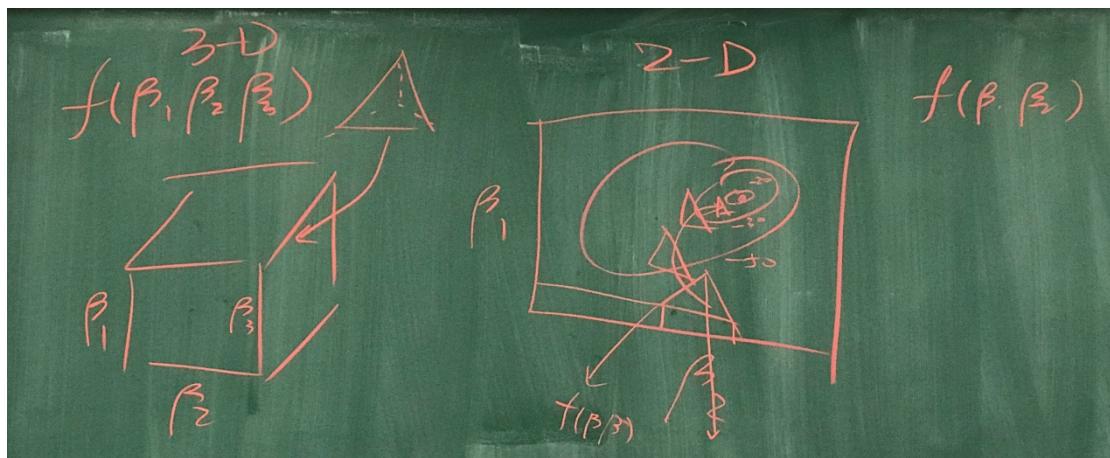
十六、Nelder-Mead

牛顿迭代的缺陷: 依赖于求导

下山单纯形法:

vertex 顶点 simplex 单纯形

(单纯形是三角形和四面体的一种泛化, 一个 k 维单纯形是指包含 $k+1$ 个节点的凸多面体)



作业:

用 Nelder-Mead 算 logistic 回归似然函数

迭代不得不用循环，其他地方能不用就不用循环

十七、Sampling

(伪随机数采样)

当均值、方差表达式难以计算时，只得退而求其次，生成服从某一分布的随机数，通过样本均值和样本方差来估计总体均值和总体方差。

一、Direct Methods

1、Discrete Case

(0, 1) 均匀 → 两点

2、Continuous Case

```
rmyexp = function(n,lambda)
{
  x = runif(n)
  -log(1-x)/lambda
}
```

二、Indirect Methods

reject and accept method:

Theorem

proof

normalizing constant (归一化常数) and kernel (核)

作业：

$f_v \sim N$

$f_y \sim t(4)$

考虑两种方法：1、用完整密度表达式 2、只用 kernel comparison

维基百科参考：

https://en.wikipedia.org/wiki/Pseudo-random_number_sampling

https://en.wikipedia.org/wiki/Inverse_transform_sampling

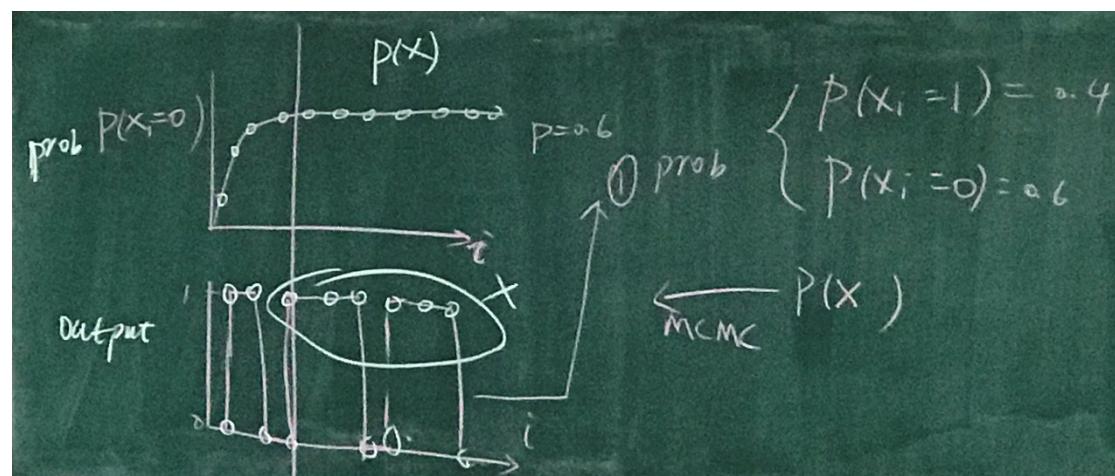
https://en.wikipedia.org/wiki/Rejection_sampling

十八、MCMC

Metroplis-Hastings Algorithm
蒙特卡洛性质

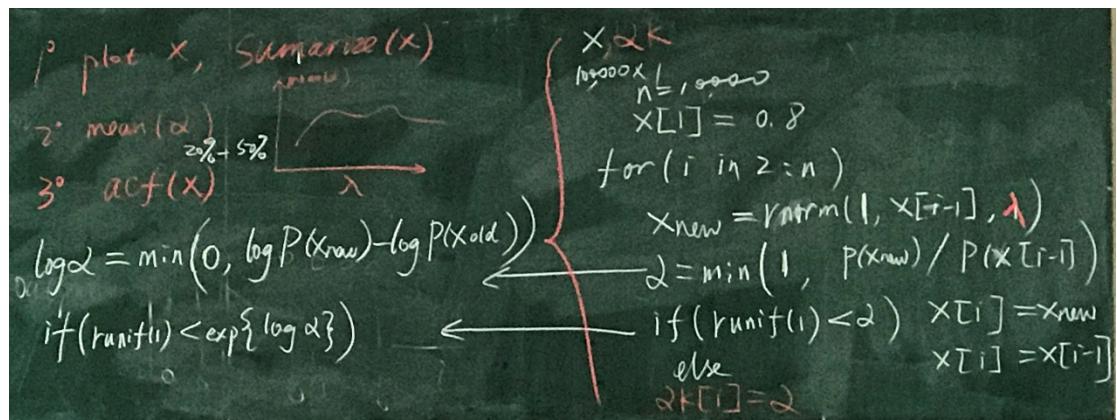
十九、Metropolis

马尔科夫性：



通过构建规则，建立马尔科夫链，再利用马尔科夫性的极限分布模拟目标分布：

$$P(x_i | x_{i-1}) \quad | \\ x^{(0)} \longrightarrow x^{(1)} = x^{(0)} + r\text{Norm}(0, 1)$$



必要的补充:

- 1、plot x , summarize(x)
- 2、mean(alpha) 0.2—0.5 比较合适
- 3、acf(x)

若 $x_{\text{new}} = \text{rnorm}(1, x[i], \lambda)$
则 mean(alpha) 与 lambda 有关

二十、Metropolis-Hastings

作业:

$$f(x) = 0.4 f_N(x, \mu=3, \sigma=1) + 0.3 f_N(x, \mu=5, \sigma=3) + 0.3 f_N(x, \mu=0, \sigma=2)$$

求这一分布的期望、方差、偏度、峰度、直方图

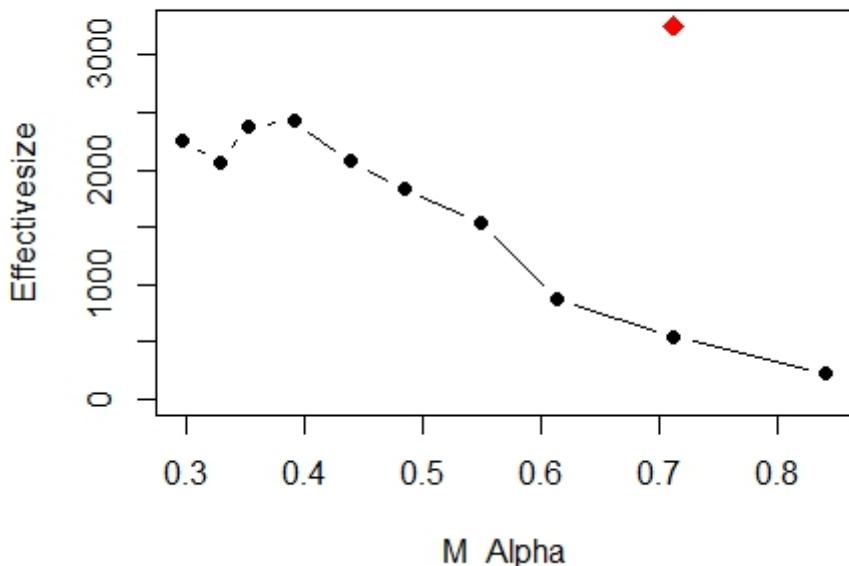
二十一、Gibbs

Metropolis 高维:
多元正态上的随机游走

$$\begin{aligned}
 & P(\theta_1, \theta_2, \dots, \theta_n) \\
 & \uparrow P(\theta) \\
 & \left(\begin{array}{c} \theta_1 \\ \theta_2 \end{array} \right)^{(new)} = \left(\begin{array}{c} \theta_1 \\ \theta_2 \end{array} \right)^{(old)} + MVN \left(\begin{array}{c} 0 \\ 0 \end{array} \right), \left[\begin{array}{cc} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{array} \right] \right) \\
 & \text{mvtnorm} := rMVN \left(\begin{array}{c} 0 \\ 0 \end{array} \right), \left[\begin{array}{cc} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{array} \right]
 \end{aligned}$$

n 越大，算法效率越低（空间上信息越稀疏）

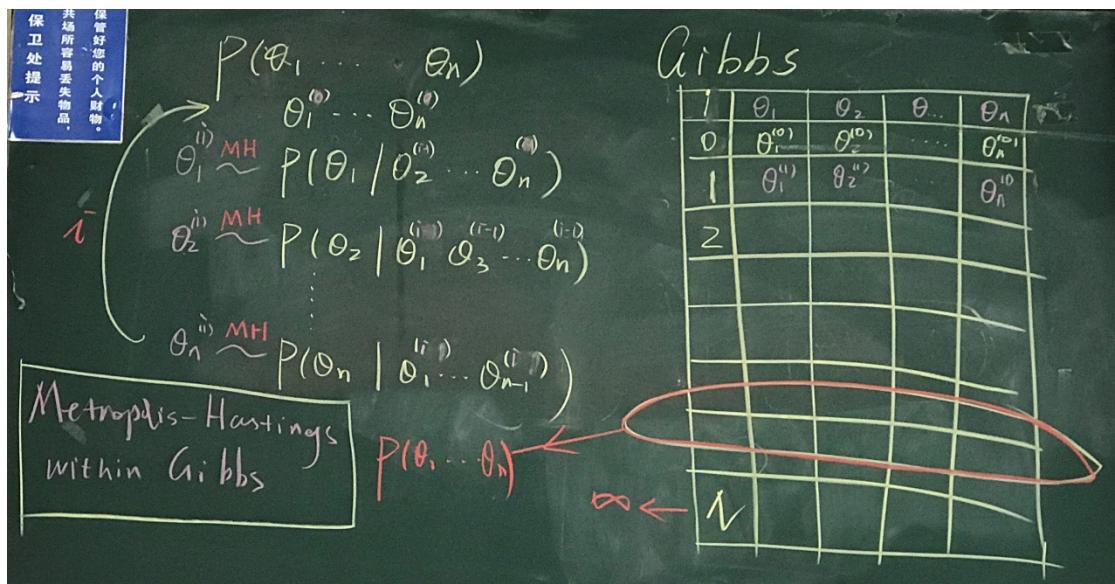
Metropolis—Hastings 同理，不过比 Metropolis 稍微好一些，因为有较高的接受概率



但 n 到十万，就过大了，直接做 Metropolis 算法很困难，因此需要新的方法：

条件概率（给定其余参数）

吉布斯抽样 (Gibbs)



其中 MH 只生成一个点，不循环，不 burn-in

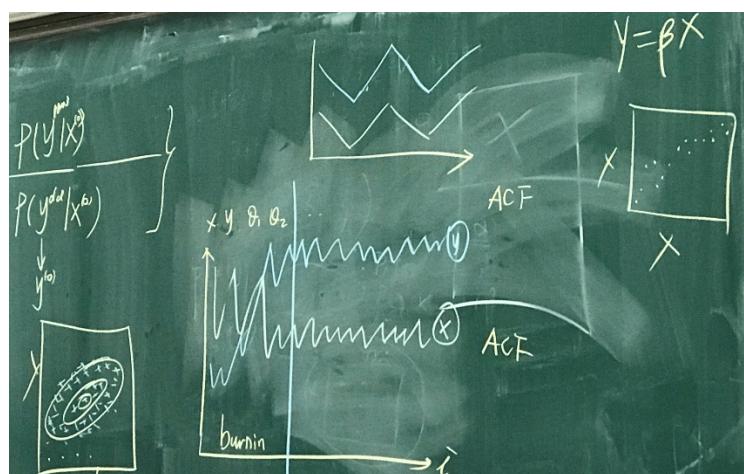
作业：

$$\begin{aligned} P(x, y) &= 0.4 P_1(x, y) \\ P_1 &\in \mathcal{N}\left(\begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 & 0.3 \\ 0.3 & 1 \end{bmatrix}\right) \\ P_2 &\in \mathcal{T}\left(\begin{bmatrix} -1 \\ -1 \end{bmatrix}, d=3, \begin{bmatrix} 1 \\ 0.7 \\ 1 \end{bmatrix}\right) \end{aligned}$$

用多维 Metropolis 和 Gibbs 抽样

二十二、 Bayesian Inference with MCMC

Gibbs 续：



Bayesian Inference:

$$y: \text{data} \quad \theta: \text{par} \quad P: \text{model}$$

$$P(y|\theta)$$

$$P(\theta)$$

$$\text{Frquentist}$$

$$\text{Classic}$$

$$\text{Inference}$$

$$Y = \theta X + \epsilon$$

$$\hat{\theta} \rightarrow \theta_{\text{true}}$$

$$E(\hat{\theta}|X) = \theta$$

$$\text{Var}(\hat{\theta}|X) = \frac{\sigma^2}{(x\bar{x})}$$

$$\text{Conditional dist}$$

$$P(\theta|y) = \frac{P(y|\theta)P(\theta)}{P(y)} = C \times P(y|\theta)P(\theta) \propto P(y|\theta)P(\theta)$$

$$P(y) = \int P(y|\theta)P(\theta)d\theta$$

$$= C$$

Y, X : data, observed, error

θ : unknown

$P(\theta) \rightarrow \text{prior}$

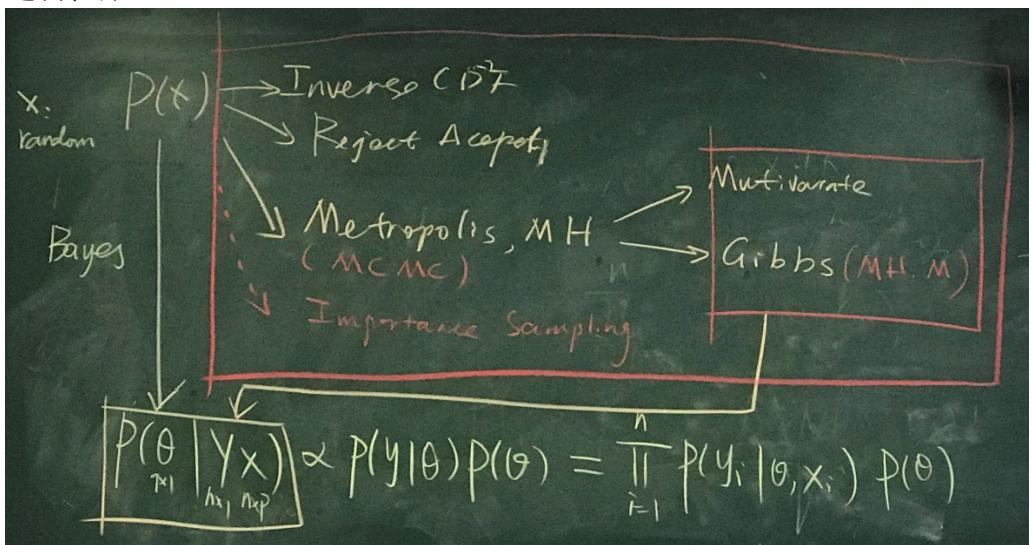
$P(y|\theta) \rightarrow \text{likelihood}$

$P(\theta|y) \rightarrow \text{posterior}$

作业：贝叶斯推断

二十三、

逻辑框架：



逻辑斯蒂:

$$\begin{aligned}
 Y_i &= \beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + \dots + \beta_k x_{k,i} \\
 &\log \frac{P}{1-P} = \beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + \dots + \beta_k x_{k,i} \\
 P(Y_i=1) &= P \\
 P(Y_i=0) &= 1-P \\
 P(Y_i=1 | X_i, \beta) &= \frac{P}{1+e^{-\beta^T X_i}} \\
 P(Y_i=0 | X_i, \beta) &= \frac{1-P}{1+e^{-\beta^T X_i}} \\
 P(Y_i | X_i, \beta) &= \prod_{i=1}^n P(Y_i | X_i, \beta_i)^{y_i} (1-P_i)^{1-y_i} \\
 &= \prod_{i=1}^n \left\{ \frac{1}{1+e^{-\beta_i^T X_i}} \right\}^{y_i} \left\{ \frac{1}{1+e^{-\beta_i^T X_i}} \right\}^{1-y_i} \\
 P(\beta) &\stackrel{\text{iid}}{\sim} \prod_{i=1}^n N(0, I) \\
 &\stackrel{\text{MVN}}{\sim} MVN\left(\begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & & & \\ \vdots & \ddots & & \\ 0 & & 1 & \\ & & & \ddots & 1 \end{bmatrix}\right)
 \end{aligned}$$

考虑 sig2 估计的贝叶斯:

$$\begin{aligned}
 Y_i &= \beta_0 x_{0,i} + \beta_1 x_{1,i} + \dots + \beta_k x_{k,i} + \varepsilon_i \\
 \varepsilon_i &\sim N(0, \sigma^2) \\
 P(\beta, \sigma^2 | Y, X) &\propto P(Y | X, \beta, \sigma^2) \\
 P(\beta, \sigma^2) &= P(\beta | \sigma^2) P(\sigma^2) \\
 &\stackrel{\text{MVN}(0, \Sigma)}{\sim} \stackrel{\text{Inverse Chi}^2}{\sim}
 \end{aligned}$$

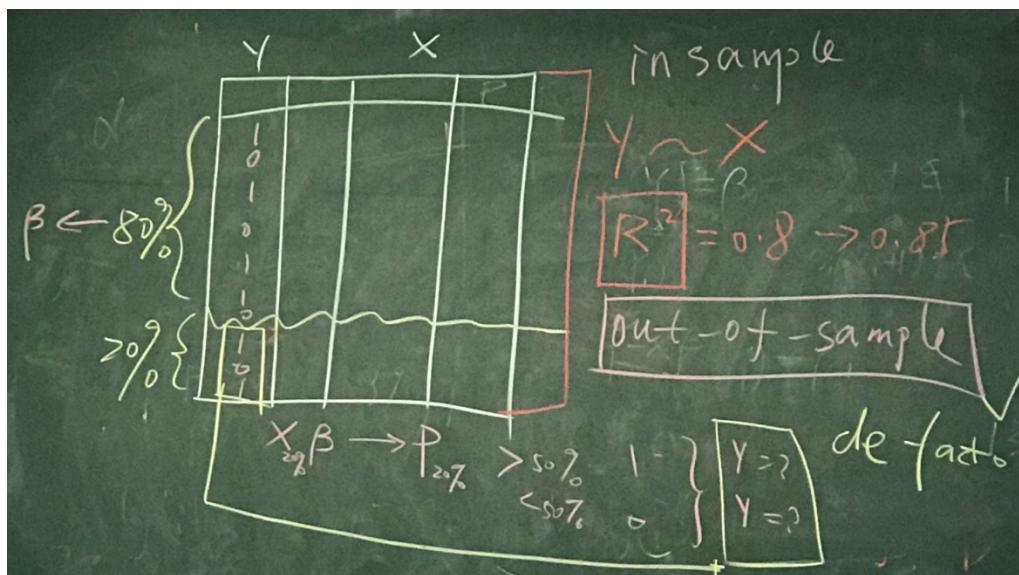
https://en.wikipedia.org/wiki/Inverse-chi-squared_distribution

二十四、Summarize + Data-Import-Export

作业点评：

如果 logit 函数得到的 P 接近 0 或 1，就导致 0 的 0 次幂不存在，对数又趋近于负无穷。因此可以做一个判断，如果 P 足够解决 0 或 1，就把 P 替换成一个足够解决 0 或 1 的常数。

R^2 —— in sample (内样本) —— e.g. 只考之前的作业
out-of-sample (外样本) —— 还考作业以外的



总结：

贝叶斯的 3 个优点

$$\text{Prior} \quad P(\theta | y) \propto P(y | \theta) P(\theta)$$

1° General

2° Computational Intensive

3° $P(\theta | y) = \frac{P(y | \theta)}{1 - P} P(\theta | y)$

yesterday's posterior is today's prior

二十五、Computational-Linear-Algebra

1、克罗内克积

(不符合交换律)

$A \otimes B$ 如果 A 是一个 $m \times n$ 的矩阵, 而 B 是一个 $p \times q$ 的矩阵, 克罗内克积则是一个 $mp \times nq$ 的分块矩阵

$$A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{bmatrix}$$

更具体地可表示为

$$A \otimes B = \begin{bmatrix} a_{11}b_{11} & a_{11}b_{12} & \cdots & a_{11}b_{1q} & \cdots & \cdots & a_{1n}b_{11} & a_{1n}b_{12} & \cdots & a_{1n}b_{1q} \\ a_{11}b_{21} & a_{11}b_{22} & \cdots & a_{11}b_{2q} & \cdots & \cdots & a_{1n}b_{21} & a_{1n}b_{22} & \cdots & a_{1n}b_{2q} \\ \vdots & \vdots & \ddots & \vdots & & & \vdots & \vdots & \ddots & \vdots \\ a_{11}b_{p1} & a_{11}b_{p2} & \cdots & a_{11}b_{pq} & \cdots & \cdots & a_{1n}b_{p1} & a_{1n}b_{p2} & \cdots & a_{1n}b_{pq} \\ \vdots & \vdots & & \vdots & \ddots & & \vdots & \vdots & & \vdots \\ \vdots & \vdots & & \vdots & & \ddots & \vdots & \vdots & & \vdots \\ a_{m1}b_{11} & a_{m1}b_{12} & \cdots & a_{m1}b_{1q} & \cdots & \cdots & a_{mn}b_{11} & a_{mn}b_{12} & \cdots & a_{mn}b_{1q} \\ a_{m1}b_{21} & a_{m1}b_{22} & \cdots & a_{m1}b_{2q} & \cdots & \cdots & a_{mn}b_{21} & a_{mn}b_{22} & \cdots & a_{mn}b_{2q} \\ \vdots & \vdots & \ddots & \vdots & & & \vdots & \vdots & \ddots & \vdots \\ a_{m1}b_{p1} & a_{m1}b_{p2} & \cdots & a_{m1}b_{pq} & \cdots & \cdots & a_{mn}b_{p1} & a_{mn}b_{p2} & \cdots & a_{mn}b_{pq} \end{bmatrix}$$

2、QR 分解

3、Moore-Penrose 广义逆矩阵

存在条件

存在一个唯一的矩阵 M 使得下面三个条件同时成立:

(1) $AM\mathbf{A} = \mathbf{A}$;

(2) $\mathbf{M}\mathbf{A}\mathbf{M} = \mathbf{M}$;

(3) \mathbf{AM} 与 \mathbf{MA} 均为对称矩阵。

这样的矩阵 M 成为矩阵 A 的Moore-Penrose广义逆矩阵, 记作 $M = A^+$.

4、奇异值分解 (SVD)

$$\mathbf{M}_{m \times n} = \mathbf{U}_{m \times n} \Sigma_{n \times n} \mathbf{V}'_{n \times n}$$

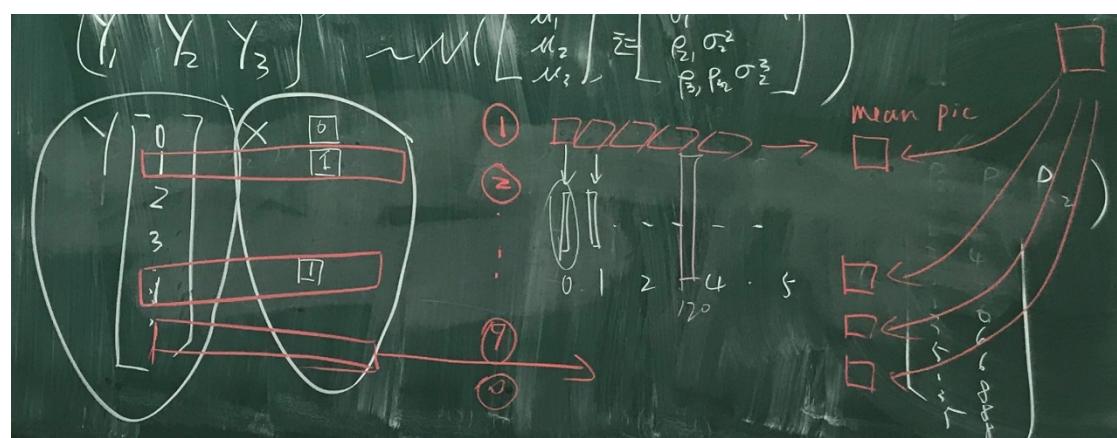
和条件数的关系

5、手写数字识别

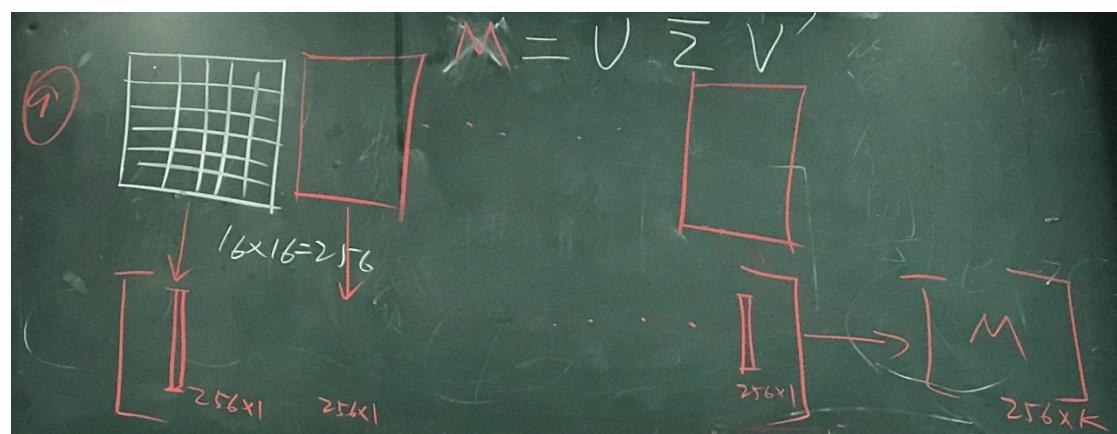
naive method —— mean pic → 比较新图片与每个 mean 矩阵间的距离

SVD method —— SVD 分解 → U 矩阵的每一分量 → 回归、看残差

Y(0:9) X(singular value)



二十六、



M 有 k 个图像组成，每个图像都是 M 中的 1 列

U 几乎包含了 M 的所有信息

以一个测试图像为 Y，选取的 U 的前几列为 X，做回归，看残差

期末作业

统计离不开数据分析

一、交作业时间: **7.17 当天发** (24: 00 之前)

二、邮件要求

Subject: SC_EXAM

三个文件: 1、Name_ID_SCEXAM.R

2、Name_ID_SCEXAM.pdf 报告, 写程序运行结果

3、Name_ID_SCEXAM.csv 如果有数据

不要压缩, 三个附件就行

正文: 写自己的想法, 不计入成绩, 可以多写点

三、作业要求

自学章节:

Part III: Topics in Matrices and Data Mining

Web Scraping with R

从网页中爬取数据

报告内容:

1、Data:Scraping 数据来源需要从自学章节中得到, 不用抓太多

2、Description,Motivation 描述数据, 以及这个数据能干什么

3、Modeling:

如果认为数据服从特定分布, 可以做密度估计 (健步走), **不建议用学过的模型**
cran task view:distribution 可以简单, 但不要学过

(1) regression,

假设误差项服从某一分布, 但不是高斯分布 e.g. gamma 回归

(2)implement, algorithms/model

自己的实现和 R 提供的做对比

(3)Algebra,Matrix

信科同学: 微分方程、数值计算

(4)image

如果对图像感兴趣, 可以做图像的应用, 但不要网上找技术博客, 抄过来

4、Comparison: Empirical study

如果别人也做过 R 包，可以一行跑一下，对比一下，看优劣势

5、Conclusion