简单梳理

第四章 产生随机数

逆方法

(1) 原理

连续的情况：为什么从均匀分布里面产生一个随机数U，然后将这个U带入分布函数的反函数就能得到所需要的分布？

$$\Downarrow$$

$$\begin{aligned}&X \sim F(x)\\&U \sim U(0,1)\\\\&X = F^{-1}(U) \sim F(x)\end{aligned}$$

$$\text{证明：}\quad \begin{aligned}F_X(x) &= P\{X \le x\}\\&= P\{F'(U) \le x\}\\&= P\{U \le F(x)\}\\&= F(x)\end{aligned}$$

离散的情况：为什么从均匀分布里面产生一个随机数U，然后将这个U带入$F(x_{i-1}) < U \le F(x_i)$就能得到所需要的分布？

$$\begin{aligned}&X \sim F(x) = \begin{pmatrix} x_1 & x_2 & \cdots & x_n \\ p_1 & p_2 & \cdots & p_n \end{pmatrix}\\&U \sim U(0,1)\end{aligned}$$

$$X = F^{-1}(U) \sim F(x) \qquad \Rightarrow \qquad F(X) = U$$

$$\begin{aligned}F_x(x) &= P(X = x_i)\\&= P(F^{-1}(U) = x_i)\\&= P(U = F(x_i))\end{aligned}$$

$$F_x(x) = P(U = F(x_i)) \Leftrightarrow F(x_{i-1}) < U \le F(x_i)$$

离散的情况：

$$P(X = x_i) = p_i; \quad x_0 < x_1 < x_2 < \ldots \quad \sum_i p_i = 1$$

产生随机数 U ，推导随机数 X 的公式如下：

$$X = x_i \quad \text{if} \quad F(x_{i-1}) < U \le F(x_i)$$

(2) 算法

算法----逆方法（连续）

1. Derive the expression for the inverse distribution function $F^{-1}(U)$.
   推导逆分布函数的表达式

2. Generate a uniform random number U.
   从均匀分布中产生随机数 U
3. Obtain the desired X from $X = F^{-1}(U)$.
   从 $X = F^{-1}(U)$ 中得到想要的 X.

算法----逆方法（离散）

1. Define a probability mass function for $x_i, i = 1, \ldots, k$. Note that k could grow infinitely.
2. Generate a uniform random number U.
3. If $U \le p_0$,deliver $X = x_0$
4. Else if $U \le p_0 + p_1$, deliver $X = x_1$.
5. Else if $U \le p_0 + p_1 + p_2$, deliver $X = x_2$.
6. ...
7. Else if $U \le p_0 + p_1 + p_2 + p_k$, deliver $X = x_k$.

(3) 程序

产生随机数种子

```
for i=1:3
  rand('state',i) % set the state
  x(i,:)=rand(1,5);
end
```

连续的逆方法

□

离散的逆方法

例子:

模拟一个离散的随机向量 X ，有概率分布列

$$\begin{aligned} P(X=0)=0.3\\ P(X=1)=0.2\\ P(X=2)=0.5 \end{aligned}$$

累积分布函数为

$$F(x)=\begin{cases} 0; & x<0\\ 0.3; & 0\le x<1\\ 0.5; & 1\le x<2\\ 1.0; & 2\le x \end{cases}$$

则根据式子，可以得到产生随机数的规则为：

$$X=\begin{cases} 0; & U\le 0.3\\ 1; & 0.3<U\le 0.5\\ 2; & 0.5<U\le 1 \end{cases}$$

```
n = 1000;
```

```
X=zeros(1,n);
% these are the x's in the domain
x = 0:2;
% these are the probability masses
pr = [0.3 0.2 0.5];
% generate 1000 random from the desired distribution
for i=1:n
  u=rand;  % generate the U
  if u<=pr(1)
    X(i)=x(1);
  elseif u<= sum(pr(1:2))
          % it has to be between 0.3 and 0.5
    X(i)=x(2);
 else
    X(i)=x(3); % it has to be between 0.5 and 1
  end
end

% find the proportion of each number
x0=length(find(X==0))/n
x1=length(find(X==1))/n
x2=length(find(X==2))/n
```

接受拒绝方法

(1) 原理

(2) 算法

算法----接受拒绝方法（连续）

1. Choose a density $g(y)$ that is easy to sample from.
2. Find a constant c such that satisfied $\frac{f(y)}{g(y)} \leq c$ for all y
3. Generate a random number γ from $g(y)$.
4. Generate a uniform random number U.
5. If $U \leq \frac{f(\gamma)}{cg(\gamma)}$, then accept $X = \gamma$,
   else go to step 3.

算法----接受拒绝方法（离散）

1. Choose a probability mass function q_i that is easy to sample from.
2. Find a constant c such that $p_\gamma < cq_\gamma$.
3. Generate a random number γ from the density q_i.
4. Generate a uniform random number U.
5. If $U \leq \frac{p(\gamma)}{cq(\gamma)}$, then deliver $X = \gamma$,
   else go to step 3.

(3) 程序

连续的接受拒绝方法

```
c = 2;   % constant
n=100;  % generate 100 rv's
% set up the arrays to store variates
x = zeros(1,n);                      % random variates
xy = zeros(1,n);                     % corresponding y values
rej = zeros(1,n);                    % rejected variates
rejy = zeros(1,n); % corresponding y values
irv=1;
irej=1;
while irv <= n
y = rand(1);  % random number from g(y)
u = rand(1);  % random number for comparison
if u <= 2*y/c
   x(irv)=y;
   xy(irv) = u*c;
   irv=irv+1;
else
   rej(irej)= y;
   rejy(irej) = u*c; % really comparing u*c<=2*y
   irej = irej + 1;
```

```
end
end

hold on
plot(x,xy,'o')
plot(rej,rejy,'*')
```

离散的接受拒绝方法

例4.5 根据概率质量函数，用离散的接受拒绝方法去产生随机变量

$$
\begin{aligned}
P(X=1)&=0.15\\
P(X=2)&=0.22\\
P(X=3)&=0.33\\
P(X=4)&=0.10\\
P(X=5)&=0.20
\end{aligned}
$$

令 q_γ 为离散的均匀分布，取值为$1,2,...,5$，则其概率质量函数为 $q_y=\frac{1}{5}$， $y=1,\ldots,5.$

The value for c is obtained as the maximum value of p_y/q_y, which is 1.65
获得c的值作为p_y/q_y的最大值，即1.65

This quantity is obtained by taking the maximum p_y, which is $P(X=3)=0.33$, and dividing by 1/5:
通过取最大p_y，即$P(X=3)=0.33$，再除以1/5，可以得出此数值：

$$\frac{max(p_y)}{1/5}=0.33\times 5=1.65$$

则产生变量的伪代码为:

1. 从离散的均匀分布中产生变量 γ.
   可以使用MATLAB中的函数 **randi**
2. 生成均匀分布的随机数 U.
3. 如果 $U\leq\frac{p_\gamma}{cq_\gamma}=\frac{p_\gamma}{1.65\cdot 1/5}=\frac{p_\gamma}{0.33}$，则令 $X=\gamma$，
   否则返回第1步

代码是一个练习，课后习题4.2

```
n = 500;  %产生500个随机数
X = zeros(1,n);  %定义X用来存放生成的随机数
x = 1:5;  %X的取值范围1，2，3，4，5
qy = 0.2;  %1/5
py = [0.15, 0.22, 0.33, 0.10, 0.20];  %各取值的概率
c = max(py)/qy;
i = 0;  %控制循环次数
while(i ~= n)
   Y = randi(5);  %随机产生1，2，3，4，5中的一个数
   u = rand;  %产生一个服从U(0,1)的随机数
   if u <= py(Y)/(c*qy)
      i = i+1;
      X(i) = Y;  %将满足条件的随机数录入X
   end
end

```

```
% 绘制X的直方图
% 后一个参数是范围，因为只有五个值，所以限定五个
[N,h]=hist(X,0:5);
bar(h,N/(n*(h(2)-h(1))),1,'w')
axis( [ 0 6 0 0.5 ] ) % 设定坐标轴范围为0<x<6,0<y<0.5
grid on % 画网格
phat = zeros(1,5);   %定义产生随机数的概率分布情况
for j = (1:5)
    phat(1,j) = length(find(X==j))/n;   %循环计算X=j时的概率
end
phat   %显示随机数的概率分布
```

多维正态

(1) 原理

多维正态：

z 是$d\times 1$维的标准正态分布向量，u是$d\times 1$维的标准正态分布向量的均值，R是$d\times d$维的矩阵，满足 $R^TR=\Sigma$

用公式 $x=R^Tz+\mu$ 来转化

MATLAB 函数 **csmvrnd** 产生多维标准正态随机变量 $X=ZR+\mu^T$

X 是 $x\times d$ 维的矩阵，d维随机变量
Z 是$n\times d$维矩阵，是标准正态分布

$$\begin{aligned}
&X\sim N_p(\mu,\Sigma)\quad \mu_{p\times 1}\ \Sigma_{p\times p}\\
&Z\sim N_p(0,I_p)\\
\Rightarrow &X=R^Tz+\mu\\
\text{prove}&\\
&E(X)=E(R^TZ+\mu)=R^TE(Z)+\mu=\mu
\end{aligned}$$

(2) 算法

(3) 程序

```
n = 500;
mu = [-2;3];
covm = [1 0.7 ; 0.7 1];
%  mu(:) 将行向量转化为列向量
% mu = mu(:); % Just in case it is not a column vector.
d = length(mu);
% get cholesky factorization of covariance
% 获得协方差的cholesky分解
% R = chol(A) 生成一个上三角矩阵 R 使得 R'*R = A.
% 如果 A 不是正，则报错
R = chol(covm);
% generate the standard normal random variables
% 生成标准正态随机变量
Z = randn(n,d);
X = Z*R + ones(n,1)*mu';
```

第五章 探索数据分析

直方图

1. ☑ 怎么画直方图：三种直方图的画法

    1.
       ```
       [N,h]=hist(forearm)
       bar(h,N,1,'w')
       ```
    2. `bar(h,N/140,1,'w')`
    3. `bar(x,n/( 140 * (h(2)-h(1)) ),1,'w')`

密度估计

1. ☑ 怎么估计密度估计，一维的密度估计怎么估计
   `bar(x,n/( 140 * (h(2)-h(1)) ),1,'w')`

Q-Q 图

1. ☑ Q-Q图怎么画，可以用插值函数 `interp1`
    1. ☑ 数据量相同
       ```
       xs = sort(x);
       ys = sort(y);
       plot(xs,ys,'o')
       ```
    2. ☑ 数据量不同
       ```
       m = 50;
       x = randn(1,75);
       y = randn(1,m);
       ys = sort(y);
       p = ((1:m) - 0.5)/m;
       xs=sort(x);
       qhat = zeros(size(p));
       n=length(x);
       phat = ((1:n)-0.5)/n;
       qhat=interp1(phat,xs,p); % 插值函数
       plot(qhat,ys,'ko')
       ```

泊松图

1. ☑ 给一个数据怎么认为是泊松分布，为什么k作为横坐标、...作为纵坐标形成直线就可以，说明理由，并且编程，画散点图
   理由：

   $$\frac{n_k}{N} = P(X=K) = \frac{\lambda^k}{k!}e^{-\lambda}$$

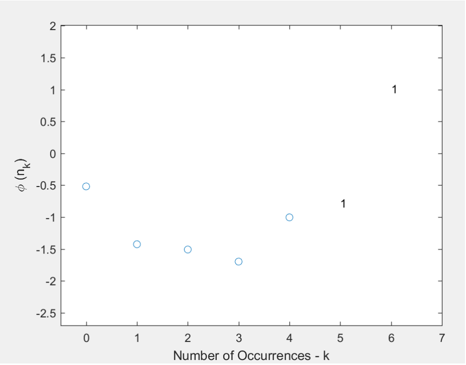
   $$k! \cdot \frac{n_k}{N} = \lambda^k e^{-\lambda}$$

   $$\varphi(n_k) = \ln\left(\frac{k! \cdot n_k}{N}\right) = k \cdot \ln\lambda - \lambda = a \cdot k + b$$

   上述公式的解释：用频率$\frac{n_k}{N}$来近似概率，然后把概率密度函数中的$k!$提取到左边，再对其求对数，就可以把关于n_k的方程转化为关于λ的线性方程，如果对于每一个k，所产生的n_k都服从泊松分布，则根据数据划出来的点是近似为一条直线，直线的斜率为 $\ln\lambda$，截距为 $-\lambda$.

   代码：

```
k=0:6;  % vector of counts
n_k = [156 63 29 8 4 1 1];
N=sum(n_k);
% get vector of factorials
fact=zeros(size(k));
for i=k
  fact(i+1)=factorial(i);  % 阶乘
end
% get phi(n_k) for plotting
phik=log(fact.*n_k/N);   % 公式
% find the counts that are equal to 1
% plot these with the symbol 1
% plot rest with a symbol
ind=find(n_k~=1);  % 将极端情况n_k=1找出来，极端情况容易使得样本异常
plot(k(ind),phik(ind),'o')
ind=find(n_k==1);
if ~isempty(ind)
  text(k(ind),phik(ind),'1') % 把空的指标的标签弄为1
end
% add some whitespace to see better
axis([-0.5 max(k)+1 min(phik)-1 max(phik)+1])
xlabel('Number of Occurrences - k')
ylabel('\phi (n_k)')
```


2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
ϕ (n_k)
0
1
2
3
4
5
6
7
Number of Occurrences - k
1
1

二项分布图

1. ☑ 给一个数据为什么是二项分布，为什么k作为横坐标、...作为纵坐标形成直线就可以，说明理由，编程

   理由：

$$X \sim B(n,p)$$

$$P(X=k)=C_n^k p^k(1-p)^{n-k}$$

$$\frac{n_k}{N}=C_n^k p^k(1-p)^{n-k}$$

$$\frac{n_k}{N\times C_n^k}=p^k(1-p)^{n-k}$$

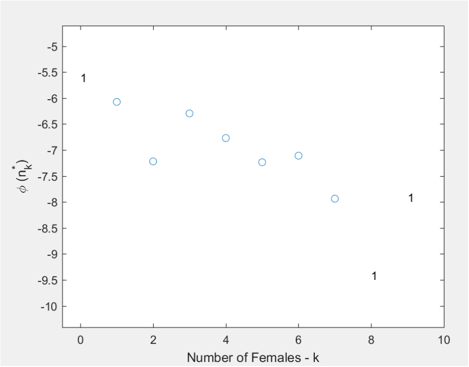
$$\begin{aligned}\varphi(n_k)&=\ln\left(\frac{n_k}{N\times C_n^k}\right)\\&=k\ln p+(n-k)\ln(1-p)\\&=(\ln p+\ln(1-p))k+n\ln(1-p)\\&=a\cdot k+b\end{aligned}$$

对上述公式的解释：如果一个随机变量 X 服从二项分布，则用频率 $\frac{n_k}{N}$ 来近似概率，把概率中的组合公式C_n^k挪到频率一边，然后两边取对数，则关于n_k的公式$\varphi(k)$就可以写成关于k的线性函数的形式，如果对每一个k，数据满足二项分布，则以k为横坐标，$\varphi(k)$为纵坐标的图像是一条直线。

代码：

```
k=0:9;
n=10;
n_k=[1 3 4 23 25 19 18 5 1 1];
N=sum(n_k);
nCk=zeros(size(k));
for i=k
  nCk(i+1)=nchoosek(n,i); % 从n里面找k个C_n^k
end
phat=n_k/N;
nkstar=n_k-0.67-0.8*phat;  % 纠偏
% Find the frequencies that are 1; nkstar=1/e.
ind1=find(n_k==1);
nkstar(ind1)= 1/2.718;  % 纠偏
% Get phi(n_k) for plotting.
phik=log(nkstar./(N*nCk));  % 计算phi(k)
% Find the counts that are equal to 1.
ind=find(n_k~=1);
plot(k(ind),phik(ind),'o') % 把不等于1 的点画上圆圈
if ~isempty(ind1)
  text(k(ind1),phik(ind1),'1') % 把等于1的点标上标签1
end
% Add some whitespace to see better.
axis([-0.5 max(k)+1 min(phik)-1 max(phik)+1]) % 规定坐标轴范围
```

```
xlabel('Number of Females - k')
ylabel('\phi (n^*_k)')
```



Andrew 曲线

- ☑ Andrew曲线怎么分类？给一个数据怎么画Andrew曲线，可能只要画三个线就行，四个线也可能，三维 $\sin 2t$等等

Andrew曲线的定义为：

$$f_x(t) = x_1/\sqrt{2} + x_2 \sin t + x_3 \cos t + x_4 \sin 2t + x_5 \cos 2t + \ldots \tag{5.9}$$

利用傅里叶展开，然后用线与线之间的形状来判定

- 相同的类其线的形状相似
- 不同的类其线的形状不同

程序

例 5.23

$$\begin{aligned} x_1 &= (2,6,4) \quad f_{x_1}(t) = 2/\sqrt{2} + 6\sin t + 4\cos t \\ x_2 &= (5,7,3) \quad f_{x_1}(t) = 5/\sqrt{2} + 7\sin t + 3\cos t \\ x_3 &= (1,8,9) \quad f_{x_1}(t) = 1/\sqrt{2} + 8\sin t + 9\cos t \end{aligned}$$

```
% Get the domain.
t = linspace(-pi,pi);
% Evaluate function values for each observation.
f1 = 2/sqrt(2)+6*sin(t)+4*cos(t);
f2 = 5/sqrt(2)+7*sin(t)+3*cos(t);
f3 = 1/sqrt(2)+8*sin(t)+9*cos(t);
plot(t,f1,'.',t,f2,'*',t,f3,'o')
legend('F1','F2','F3')
xlabel('t')
```

例 5.24

```
load iris
% 定义将要绘制的域。
theta=(-pi+eps):0.1:(pi-eps);
% 每一个特征的数目
n = length(setosa);
% 每一个特征的维数
p = 4;
ysetosa = zeros(n,p);
% 绘制n条曲线，每个数据点一条。
yvirginica=zeros(n,p);
% 取每行的点积进行观察。
ang = zeros(length(theta),p);
fstr = '[1/sqrt(2) sin(i) cos(i) sin(2*i)]';
k = 0;
```

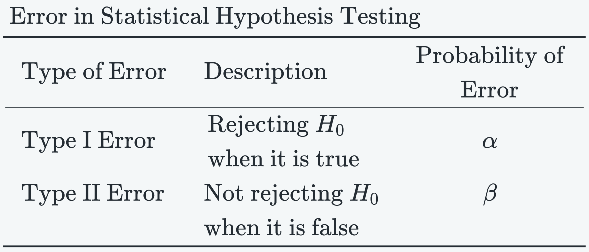
```
% 在每个角度theta上计算sin和cos函数。
for i=theta
k=k+1;
ang(k,:)=eval(fstr);
end
% 现在为每个观察值生成一个“ y”。
for i=1:n
 % 一共有50条线
 % 即每一个分类有50条线
for j=1:length(theta)
  % 通过观察找到点积。
  % 第i条线的第j个值
  % 遍历j，得到第i条线
  ysetosa(i,j)=setosa(i,:)*ang(j,:)';
  yvirginica(i,j)=virginica(i,:)*ang(j,:)';
end
end
hold
for i=1:n
plot(theta,ysetosa(i,:),'r',...
                theta,yvirginica(i,:),'b-.')
end
legend('Iris Setosa','Iris Virginica')
hold off
title('Andrews Plot')
xlabel('t')
ylabel('Andrews Curve')
```

第七章 蒙特卡洛方法

第一类错误和第二类错误，怎么计算?

第一类错误是弃真错误，原假设H_0为真却拒绝了

第二类错误是取伪错误，原假设H_0为伪却没有拒绝



Error in Statistical Hypothesis Testing

Type of Error	Description	Probability of Error
Type I Error	Rejecting H_0 when it is true	α
Type II Error	Not rejecting H_0 when it is false	β

计算第一类错误 **normcdf(ctv,mu,sigma)**

计算第二类错误

```
mualt = 40:60;
cv = norminv(0.95,0,1);
sig = 1.5;
ct = cv*1.5 + 45;
ctv = ct*ones(size(mualt));
beta = normcdf(ctv,mualt,sig);
```

计算power的程序

```
pow = 1 - beta;
plot(mualt,pow);
```

计算P值

```
mu = 45;
sig = 1.5;
xbar = 47.2;
% Get the observed value of test statistic.
zobs = (xbar - mu)/sig;
pval = 1-normcdf(zobs,0,1);
```

计算得出P值为0.071

算分位点

Monte Carlo方法怎么算第一类错误

第一类错误----伪代码

1. Determine the pseudo-population or distribution when the null hypothesis is true.
   当原假设为真时，确定伪总体或分布。
2. Generate a random sample of size n from this pseudo-population.
   从该伪总体生成大小为n的随机样本。
3. Perform the hypothesis test using the critical value.
   使用临界值执行假设检验。
4. Determine whether a Type Ⅰ error has been committed.
   确定是否发生了Ⅰ类错误。
   In other words, was the null hypothesis rejected?
   换句话说，原假设是否被拒绝？
   We know not be rejected because we are sampling from the distribution according to the null hypothesis.
   我们知道不会被拒绝，因为我们正在根据原假设从分布中进行抽样。
   Record the result for this trial as
   将该试验的结果记录为

$$I_i = \begin{cases} 1; & \text{Type I error is made} \\ 0; & \text{Type I error is not made.} \end{cases}$$

5. Repeat steps 2 through 4 for M trials.
   重复步骤2到4， M次
6. The probability of making a Type Ⅰ error is
   发生Ⅰ类错误的概率为

$$\hat{\alpha} = \frac{1}{M}\sum_{i=1}^{m} I_i \tag{7.9}$$

Matlab 代码

```
load mcdata
n = length(mcdata);
M = 1000;
alpha = 0.05;
sigma = 7.8;
sigxbar = sigma/sqrt(n);

% Get the critical value, using z as test statistic.
cv = norminv(alpha,0,1);
% Start the simulation.
Im = 0;

for i = 1:M
```

```
    % Generate a random sample under H_0.
    xs = sigma*randn(1,n) + 454;
    Tm = (mean(xs)-454)/sigxbar;
    if Tm <= cv % then reject H_0
        Im = Im +1;
    end
end
alphahat = Im/M;
```

Monte Carlo方法怎么算第二类错误、Power

第二类错误----伪代码

1. Determine the pseudo-population or distribution when the null hypothesis is false.
   当原假设为假时，确定伪总体或分布。
2. Generate a random sample of size n from this pseudo-population.
   从该伪总体生成大小为n的随机样本。
3. Perform the hypothesis test using the significance level α and corresponding critical value.
   使用显着性水平α和相应的临界值执行假设检验。
   这一点和下面的都和第一类错误不一样
4. Note whether a Type Ⅱ error has been committed;
   注意是否发生了Ⅱ型错误。
   i.e., was the null hypothesis not rejected?
   就是说，原假设不被拒绝？
   Record the result for this trial as
   将该试验的结果记录为

$$I_i = \begin{cases} 1; & \text{Type Ⅱ error is made} \\ 0; & \text{TypeⅡ error is not made.} \end{cases}$$

5. Repeat steps 2 through 4 for M trials.
   重复步骤2到4， M次
6. The probability of making a Type Ⅱ error is
   发生Ⅰ类错误的概率为

$$\hat{\beta} = \frac{1}{M}\sum_{i=1}^{m} I_i \tag{7.9}$$

Matlab 代码

```matlab
mualt = 445:458;
betahat = zeros(size(mualt));
for j = 1:length(mualt)
  Im = 0;
  % Get the true mean.
  mu = mualt(j);
  for i = 1:M
    % Generate a sample from H_1.
    xs = sigma*randn(1,n) + mu;
    Tm = (mean(xs)-454)/sigxbar;
    if Tm > cv % Then did not reject H_0.
      Im = Im +1;
    end
  end
  betahat(j) = Im/M;
```

```
end
% Get the estimated power.
powhat = 1-betahat;
```

☑ 重点是：怎么模拟P值，怎么计算第一类错误和第二类错误

第十四章 MH 方法 Gibbs 抽样

MH方法

$$\alpha(X_t,\Upsilon)=\min\left\{1,\frac{\pi(\Upsilon)q(X_t|\Upsilon)}{\pi(X_t)q(\Upsilon|X_t)}\right\} \tag{14.6}$$

(2) 算法

1. Initialize the chain to x_0 and set $t=0$.
   将链初始化为x_0并设置$t=0$。
2. Generate a candidate point Υ from $q(.|X_t)$.
   从$q(.|X_t)$生成候选点Υ。
3. Generate U from a uniform $(0,1)$ distribution.
   从均匀的$(0,1)$分布生成U。
4. If $U\le\alpha(X_t,\Upsilon)$, then set $X_{t+1}=\Upsilon$
   如果$U\le\alpha(X_t,\Upsilon)$，则设置$X_{t+1}=\Upsilon$
   else set $X_{t+1}=X_t$.
   否则设置$X_{t+1}=X_t$
5. Set $t=t+1$ and repeat steps 2 through 5.
   设置$t=t+1$并重复步骤2至5。

(3) 代码

例 14.2

$$f(x)\propto\frac{1}{1+x^2}$$

```
strg = '1./(1+x.^2)';  % 柯西分布
cauchy = inline(strg,'x');
% set up an inline function to evaluate the Normal pdf
strg = '1/sig*exp(-0.5*((x-mu)/sig).^2)'; % 正态分布
norm = inline(strg,'x','mu','sig');
% Generate 10000 samples in the chain.
% Set up the constants.
n = 10000; %一万个数据
sig = 2;
x = zeros(1,n); % 分配内存
% generate the starting point
% 初始值，理论上初始值对MH链没有影响
x(1) = randn(1);

```

```
for i = 2:n
    % generate a candidate from the proposal distribution
    % which is the normal in this case. This will be a
    % normal with mean given by the previous value in the
    % chain and standard deviation of 'sig'
    y = x(i-1) + sig*randn(1); % 用正态分布换算
    % generate a uniform for comparison
    u = rand(1);
    alpha = min([1, cauchy(y)*norm(x(i-1),y,sig)/...
       (cauchy(x(i-1))*norm(y,x(i-1),sig))]);
    if u <= alpha
        x(i) = y;
    else
        x(i) = x(i-1);
    end
end
```

```
[N,h]=hist(x);
bar(h,N/(n * (h(2)- h(1))),1,'w')
hold on
% xi = -30:2.5:30;
% yi = ksdensity(x,xi);
% bar(xi,yi,'w');
f = @(x) 1./(pi*(1+x.^2));
fplot(f,[-30,30],'k-')
hold off
```

Gibbs 抽样----二维beta

☑ Gibbs抽样，例14.7要会做，应该会把矩阵换掉

(2) 算法

1. Generate a starting point $X_0 = (X_{0,1}, X_{0,2})$. Set $t = 0$.
   生成起点$X_0 = (X_{0,1}, X_{0,2})$。设置$t = 0$。
2. Generate a point $X_{t,1}$ from $f(X_{t,1}|X_{t,2} = x_{t,2})$
   从$f(X_{t,1}|X_{t,2} = x_{t,2})$生成点$X_{t,1}$
3. Generate a point $X_{t,2}$ from $f(X_{t,2}|X_{t+1,1} = x_{t+1,1})$
   从$f(X_{t,2}|X_{t+1,1} = x_{t+1,1})$生成点$X_{t,2}$
4. Set $t = t + 1$ and repeat steps 2 through 4.
   设置$t = t + 1$并重复步骤2至4。

（3） 代码

例 14.6

联合分布 $f(x,y) \propto \binom{n}{x} y^{x+\alpha-1}(1-y)^{n-x+\beta-1}$ ，这是一个beta分布，$x = 0, 1, \ldots, n \quad 0 \le y \le 1$

条件分布 $f(x|y) \sim B(n,y)$

条件分布 $f(y|x) \sim Beta(x+\alpha, n-x+\beta)$

```
% Set up preliminaries.
% Here we use k for the chain length, because n
% is used for the number of trials in a binomial.
k = 1000; % generate a chain of size 1000
m = 500;  % burn-in will be 500
a = 2;   % chosen
b = 4;
x = zeros(1,k);
y = zeros(1,k);
n = 16;
```

```
% Pick a starting point.
% 给第一个分量的值为二项分布
x(1) = binornd(n,0.5,1,1);
% 给第二个分量的值为beta分布
y(1) = betarnd(x(1) + a, n - x(1) + b,1,1);
重复1000次
for i = 2:k
        x(i) = binornd(n,y(i-1),1,1);
        % x 抽取之后，beta分布就已知了
        % 所以抽取y的时候就可以
        % 直接从beta分布里面抽取
        y(i) = betarnd(x(i)+a, n-x(i)+b, 1, 1);
end
```

$$\hat{f}(x) = \frac{1}{k-m}\sum_{i=m+1}^{k} f(x|y_i)$$

```
% Get the marginal by evaluating the conditional.
% Use MATLAB's Statistics Toolbox.
% Find the P(X=x|Y's)
fhat = zeros(1,17);
for i = 1:17
        fhat(i) = mean(binopdf(i-1,n,y(500:k)));
end
```

☑ 为啥这里的是17次，为啥是i-1?

```
17是因为MATLAB计数从1开始

因为x的边际分布是二项分布，n=16，对于每一个不同的n，就有一个不同的分布。

这里的fhat(i)中i的含义是估计的第i个边际分布。
```

Gibbs

（2） 算法

例子 14.7 二维正态

PROCEDURE - GIBBS SAMPLER

1. Generate a starting point $X_0 = (X_{0,1}, X_{0,2}, \ldots, X_{0,d})$. Set $t = 0$.
   生成起点$X_0 = (X_{0,1}, X_{0,2}, \ldots, X_{0,d})$。设置$t = 0$。
2. Generate a point $X_{t,1}$ from $f(X_{(t+1),1} | X_{t,2} = x_{t,2}, \ldots, X_{t,d} = x_{t,d})$.
   Generate a point $X_{(t+1),2}$ from $f(X_{(t+1),1} | X_{t+1,1} = x_{t+1,1}, X_{t,3} = x_{t,3}, \ldots, X_{t,d} = x_{t,d})$.
   ......
   Generate a point $X_{(t+1),d}$ from $f(X_{(t+1),d} | X_{t+1,1} = x_{t+1,1}, \ldots, X_{t+1,d-1} = x_{t+1,d-1})$.
3. Set $t = t + 1$ and repeat steps 2 through 3.

(3) 代码

例 14.7 二维正态

$$\mu = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \quad \Sigma = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0.9 \\ 0.9 & 1 \end{bmatrix}$$

边际分布 $f(x_1|x_2) \sim N(\mu_1 + \rho(x_2 - \mu_2), (1-\rho)^2)$

边际分布 $f(x_2|x_1) \sim N(\mu_2 + \rho(x_1 - \mu_1), (1-\rho)^2)$

```
% Set up constants and arrays.
n = 6000;
xgibbs = zeros(n,2);
rho = 0.9;
y = [1;2]; % This is the mean.
sig = sqrt(1-rho^2);
% Initial point.
xgibbs(1,:) = [10 10];
% Start the chain.
for i = 2:n
  mu = y(1) + rho*(xgibbs(i-1,2)-y(2));
  % 边际分布为一元正态
  xgibbs(i,1) = mu + sig*randn(1);
  mu = y(2) + rho*(xgibbs(i,1) - y(1));
  % 边际分布为一元正态，且需要用到上一次更新的值
  xgibbs(i,2) = mu + sig*randn(1);
end
scatter(xgibbs(:,1),xgibbs(:,2))
```