



随机模拟方法与应用

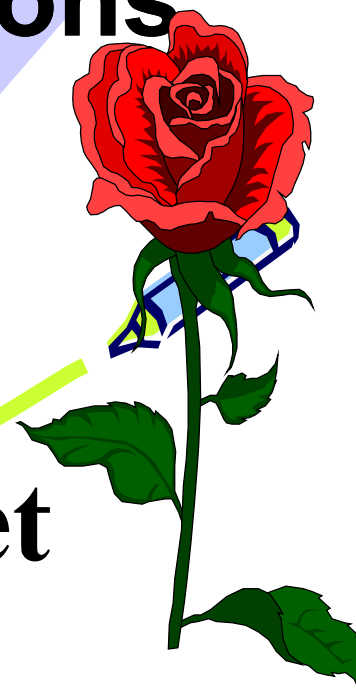
Stochastic Simulation
Methods and Its Applications

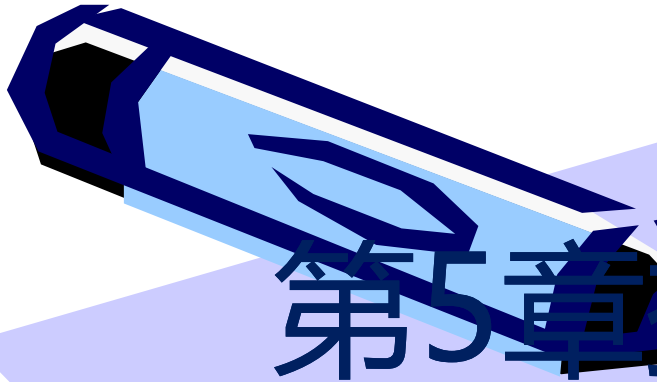
肖柳青 博士

lucyxiao@sjtu.edu.cn

PUB:SSMA_xiao@yeah.net

肖柳青 上海交通大学 数学系





第5章掷骰子的进阶： 特殊分布随机数的抽样

1.1 逆变换法

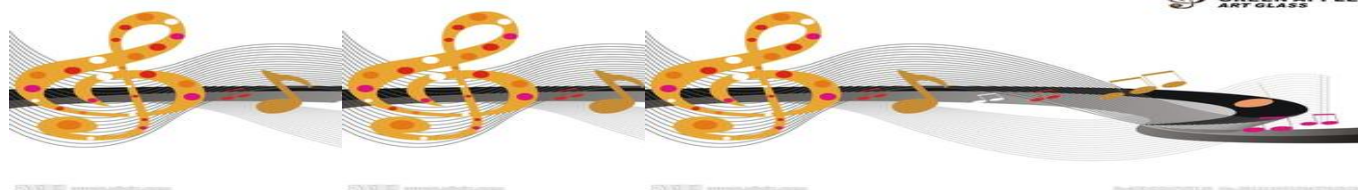
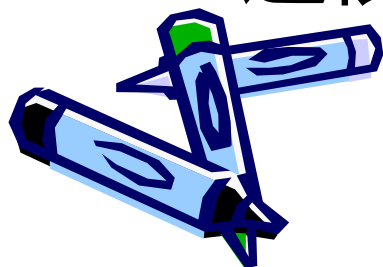
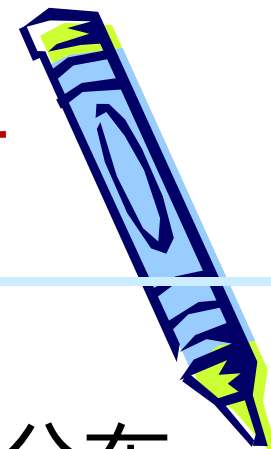
1.2 接受-拒绝法

1.3 多维分布的抽样方法



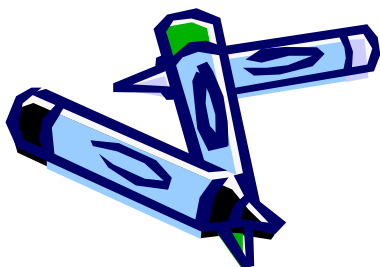
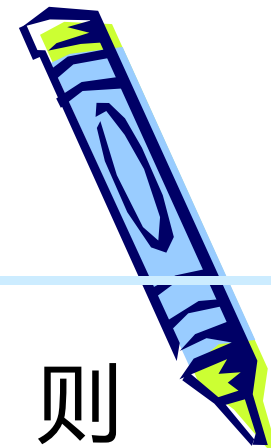
如何能够生成服从那些特殊分布的随机数呢？

- 人们在一些实际问题中所遇到的概率分布并不是通常介绍的几种常见分布，例如 Matlab 统计工具箱中没有列入的分布，我们不妨将它们称为**特殊分布**。
- 那么怎样才能生成服从那些特殊分布的随机数呢？
- 这很重要！



5.1 逆变换法

- 设 $F(x)$ 是某特定的一维概率分布函数，则生成服从该分布的随机数的逆变换法是：
- [1] 写出该分布函数的反函数 $F^{-1}(x)$;
- [2] 生成随机数 $u \sim U(0, 1)$;
- [3] 计算 $x = F^{-1}(u)$ ，则 x 就是服从该特定分布的随机数。



逆变换法数学原理:

- 设 $F(x)$ 是随机变量 X 的分布函数, 连续且严格单调上升的分布函数, 其反函数存在, 且记为 $F^{-1}(x)$ (即 $F[F^{-1}(x)] = x$)。
- ①若随机变量 X 的分布函数为 $F(x)$, 则 $F(X) \sim U(0,1)$;
- ②若随机变量 $R \sim U(0,1)$, 则 $F^{-1}(R)$ 的分布函数为 $F(x)$;

(证明)



• 证明:

- ①设随机变量 $F(X)$ 的分布函数为 $F_1(u)$,

当 $u \in [0,1]$ 时,

$$F_1(u) = P\{F(X) \leq u\} = P\{X \leq F^{-1}(u)\} = F[F^{-1}(u)] = u$$

当 $u < 0$ 时, $F_1(u) = 0$;

当 $u > 1$ 时, $F_1(u) = 1$;

所以 $F(X) \sim U(0,1)$

均匀分布

$$F(x) = \begin{cases} 0, & x \leq a \\ \frac{x-a}{b-a}, & a < x \leq b \\ 1, & x > b \end{cases}$$

- ②设随机变量 $F^{-1}(R)$ 的分布函数为 $F_2(x)$,

$$F_2(x) = P\{F^{-1}(R) \leq x\} = P\{R \leq F(x)\} = F_R[F(x)] = F(x)$$

因为 $R \sim U(0,1)$, 对任意 $F(x) \in [0,1]$, 有 $F_R(F(x)) = F(x)$

所以 $F^{-1}(R)$ 的分布函数为 $F(x)$



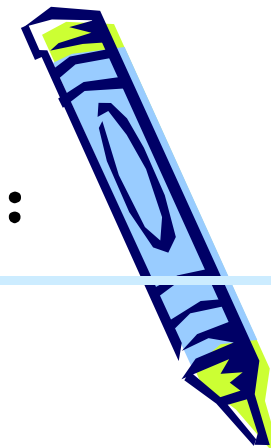
- 例5.1 标准柯西分布的概率分布函数是：

$$F(x) = \frac{1}{2} + \frac{\arctan(x)}{\pi}$$

- 现在，我们逆变换算法来生成服从柯西分布的随机数序列。
- 首先，容易写出它的反函数为

$$F^{-1}(y) = \tan\left(\pi\left(y - \frac{1}{2}\right)\right)$$

- 接着，使用rand函数生成区间(0, 1)上的均匀分布的随机数



- 我们可以利用Matalb的特点一次性地生成一组均匀分布的随机数序列（如10万个）：

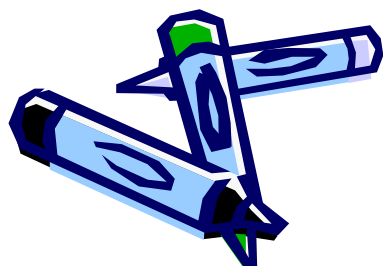
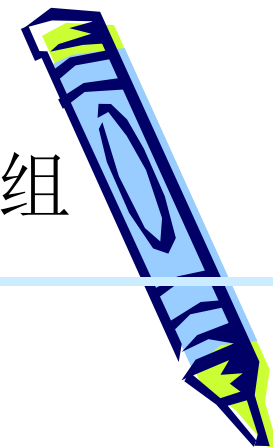
- $U = \text{rand}(1,100000);$

- 最后，将这组随机数序列代入上面的反函数即可得到我们需要的柯西分布的

- 随机数序列：

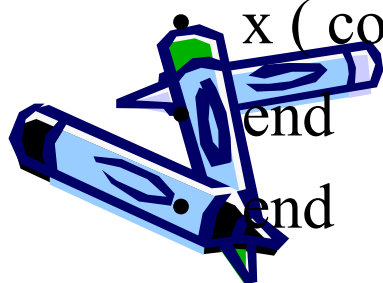
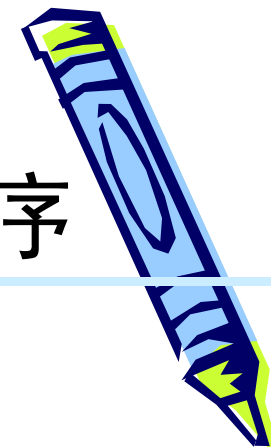
- $X = \tan(\pi*(U-1/2))$

- 程序如下，演示见图5.1



生成柯西分布随机数的Matlab程序

- `N=500000;`
- `x=zeros (1 ,N) ; count=0;`
- `for i =1:N`
- `U=unifrnd (0 , 1) ;`
- `tmp=tan(pi *(U-0 . 5)) ;`
- `if tmp>-20 && tmp<20`
- `count=count+1;`
- `x (count)=tmp ;`
- `end`
- `end`
- `hist (x , 6 0) ;`
- `hold on`
- `t = -20:0.1:20;`
- `y=1./(pi*(1+t . * t)) ;`
- `s c a l e=count * (4 0 / 6 0) ;`
- `plot (t , s c a l e *y , ' r ')`
- `hold o f f`



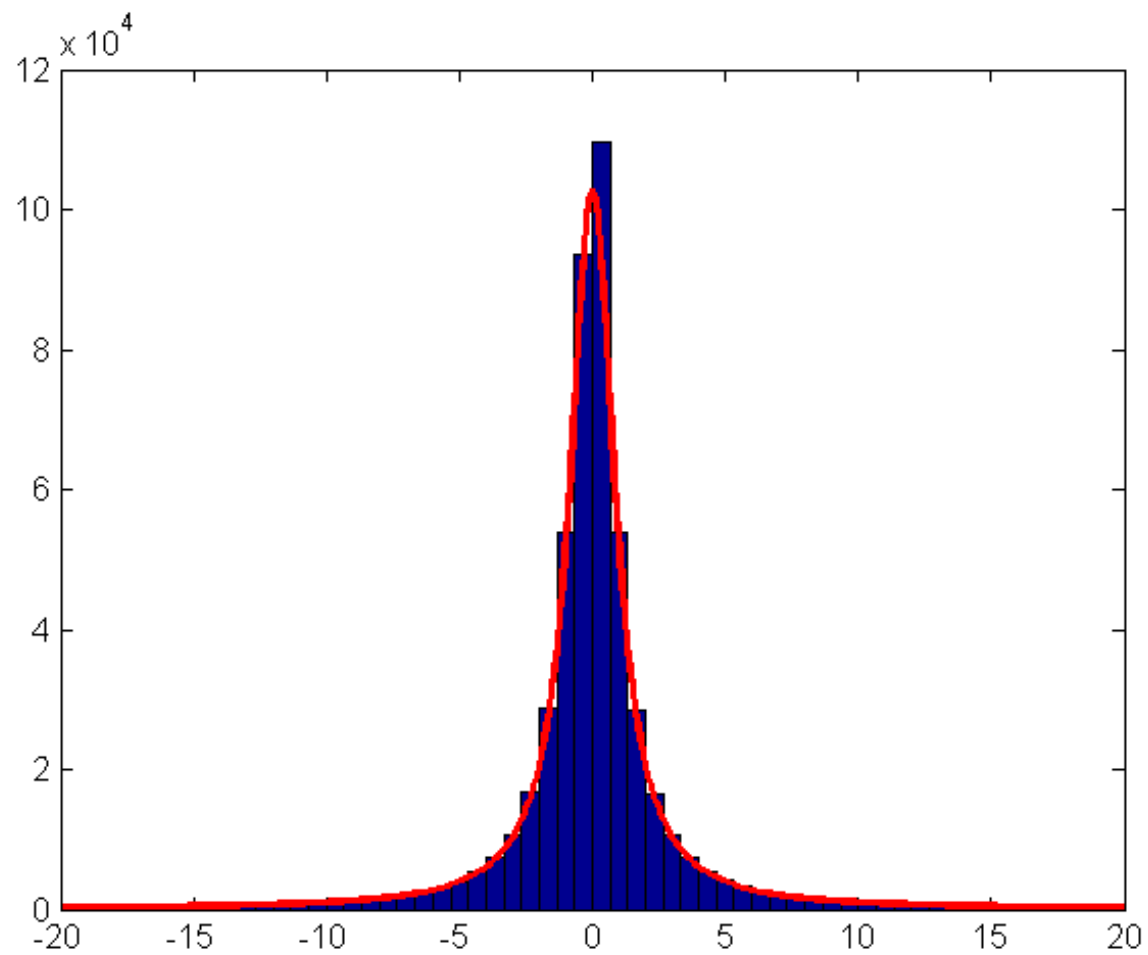
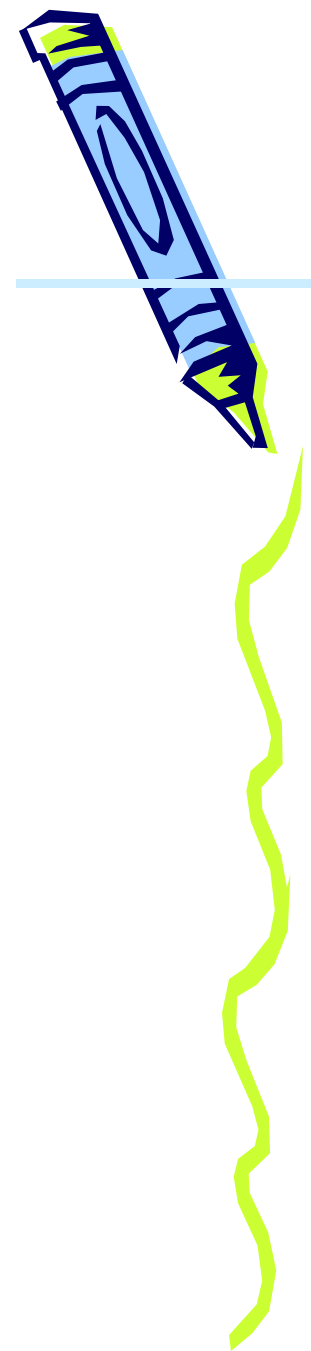
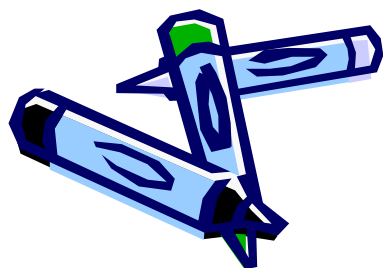
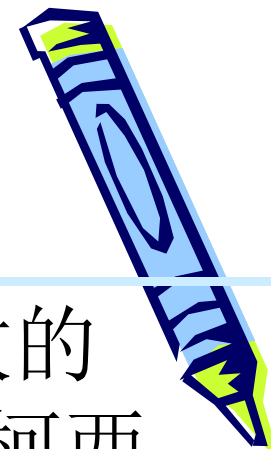
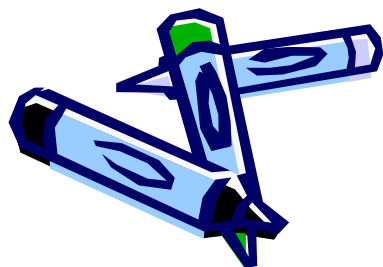


图5.1 由逆变换法生成柯西分布的随机数的直方图



逆变换的局限法

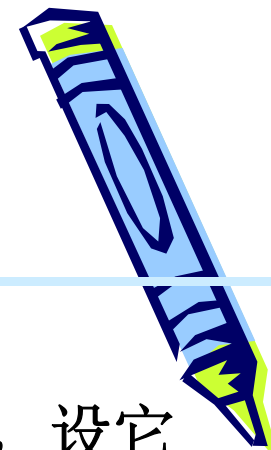
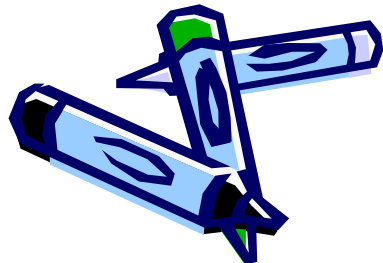
- (1) 程序运行有可能出错，有的分布函数的反函数在0或者1处的值是无穷大，例如柯西分布就是这样。
- (2) 分布函数必须严格单调递增，否则它的反函数不存在。
- (3) 即使分布函数的反函数存在，如果它太复杂或者无法解析地写出，则可能导致计算速度过慢，甚至无法计算的后果。



5.2 接受-拒绝法

- 接受-拒绝算法的具体步骤如下：
- (1) 首先选择一个容易抽样的某个分布作为建议分布，设它为 $g(x)$ ，接着要确定一个常数 $M > 1$ ，使得在 x 的定义域上均有 $f(x) \leq Mg(x)$ 成立；
- (2) 生成服从概率密度函数为 $g(x)$ 的建议随机数 y ；
- (3) 再生成一个服从均匀分布 $U(0, 1)$ 的随机数 u ；
- (4) 计算接受准则的概率函数 $h(x) = \frac{f(x)}{Mg(x)}$ ；如果 $u < h(y)$,

则接受所生成的该随机数 y ；反之，则丢弃该随机数 y 并转到第2步。



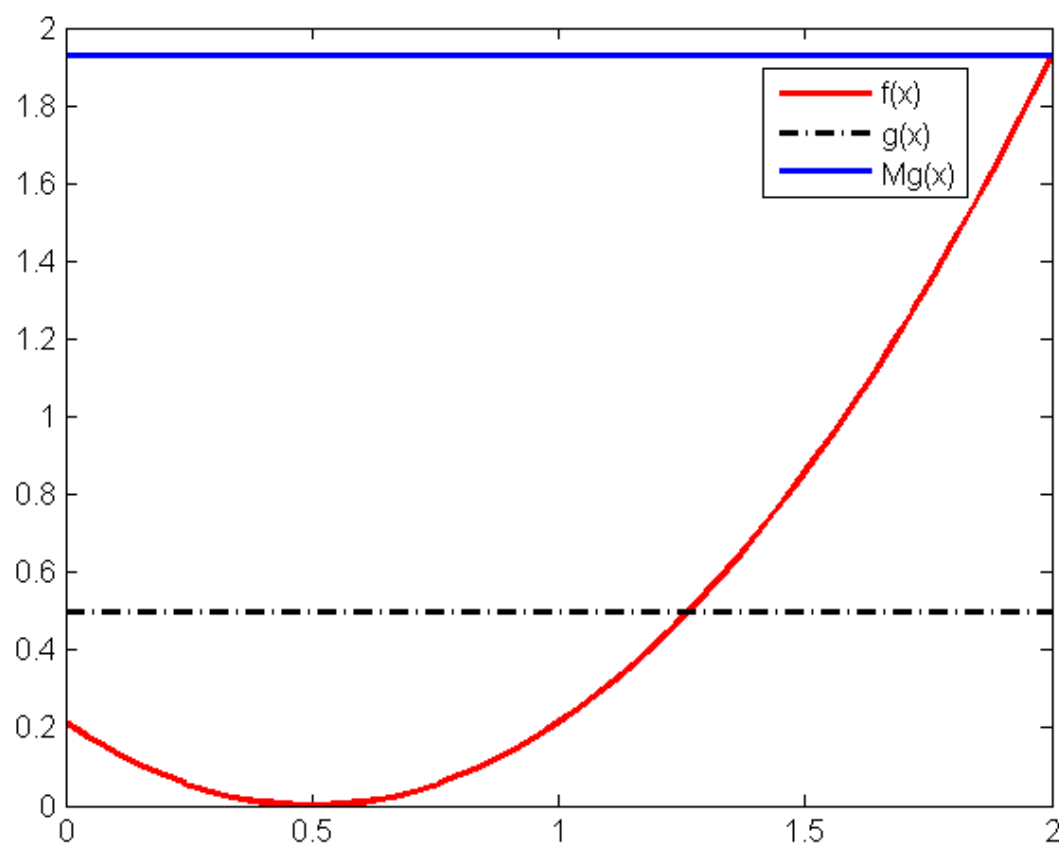
- 例5.2 设目标概率密度函数是：

$$f(x) = \frac{6(x-0.5)^2}{7}, x \in (0, 2)$$

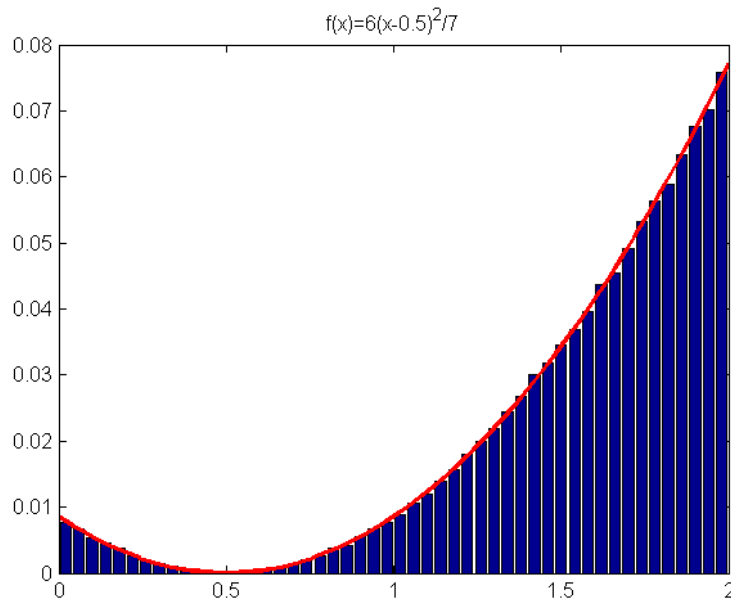
- (1) 我们选用在相同定义域上的均匀分布 $U(0, 2)$ 作为建议分布，即 $g(x)=0.5$ 。由于要求 $f(x) \leq Mg(x)$ 成立，不难求得 $\max_{x \in [0, 2]} f(x) = 1.9286$ ，则我们可以选择 $M = 3.858$ 。
- (2) 生成建议随机数 $y \sim U(0, 2)$ ；
- (3) 再生成一个随机数 $u \sim U(0, 1)$ ；
- (4) 计算接受准则的概率函数 $h(x) = \frac{f(x)}{Mg(x)}$ ；如果 $u < h(y)$ ，则接受所生成的该随机数 y ；反之，则丢弃该随机数 y 并转到第2步重新生成。



- 不断地重复步骤2-4就可以生成出一组服从目标概率密度函数 $f(x)$ 的随机数序列。



- 我们充分利用Matlab矩阵运算的优势，整批地产生一堆随机数，其程序代码如下：

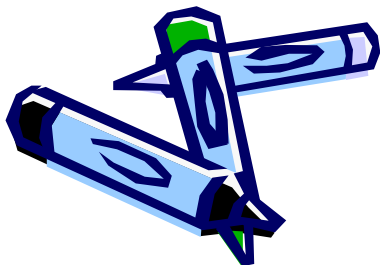


```
N=500000; M=3.858;
Y=unifrnd ( 0 , 2 , 1 ,N) ;
U=unifrnd ( 0 , 1 , 1 ,N) ;
gy=0.5;
fy=6*(Y-0.5) .*(Y-0.5)/7;
X=Y(U < fy ./ gy/M) ;
sample=length (X) ;
```

```
[Xnumber , Xcenters ]=hist (X, 50 ) ;
bar ( Xcenters ,Xnumber/ sample )
title ( ' f ( x ) = 6 ( x - 0 . 5 ) ^ 2 / 7 ' )
hold on
t = 0 : 0 . 0 4 : 2 ;
z = 6 * ( t - 0 . 5 ) . * ( t - 0 . 5 ) / 7 ;
scale = 2 / 50 ;
plot ( t , scale * z , ' r ' )
```

hold on f

肖柳青 上海交通大学 数学系



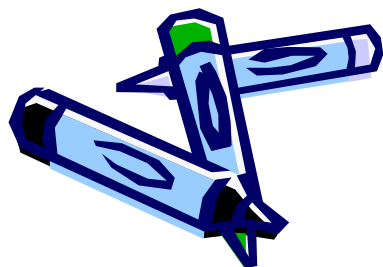
接受-拒绝法的数学原理

- 设 F 和 G 分别是对应于目标密度函数 f 和建议密度函数 g 的分布函数，我们只需要证明

$$P(Y \leq y | U \leq \frac{f(Y)}{Mg(Y)}) = F(y)$$

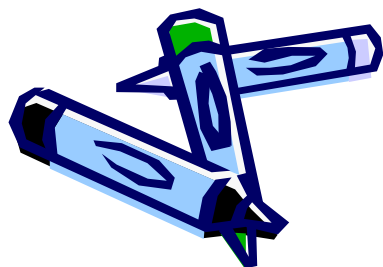
$$P(U \leq \frac{f(Y)}{Mg(Y)} | Y = y) = \frac{f(y)}{Mg(y)}$$

$$\begin{aligned} P(U \leq \frac{f(Y)}{Mg(Y)}) &= \int_{-\infty}^{\infty} P(U \leq \frac{f(Y)}{Mg(Y)} | Y = y) g(y) dy \\ &= \int_{-\infty}^{\infty} \frac{f(y)}{Mg(y)} g(y) dy = \frac{1}{M} \int_{-\infty}^{\infty} f(y) dy = \frac{1}{M} \end{aligned}$$



记事件 $A = \{Y \leq y\}$ 和 $B = \{U \leq \frac{f(Y)}{Mg(Y)}\}$,

$$\begin{aligned} P(Y \leq y | U \leq \frac{f(Y)}{Mg(Y)}) &= P(A|B) = \frac{P(B|A)P(A)}{P(B)} \\ &= \frac{P(U \leq \frac{f(Y)}{Mg(Y)} | Y \leq y)P(Y \leq y)}{1/M} \\ &= M \int_{-\infty}^y P(U \leq \frac{f(Y)}{Mg(Y)} | Y = w < y)g(w)dw \\ &= M \int_{-\infty}^y \frac{f(w)}{Mg(w)}g(w)dw = \int_{-\infty}^y f(w)dw = F(y) \end{aligned}$$



接受-拒绝法似乎更简单，不需求分布函数的反函数，但它也有如下缺陷：

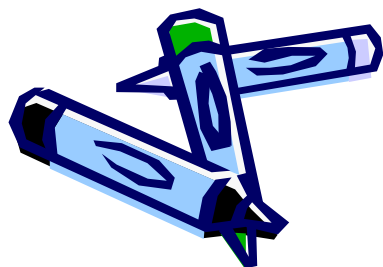
- (1) 由于算法要随机地拒绝掉许多建议的随机数 y ，根据算法效率，我们估计迭代 N 次后最终会得到随机数的数量大约是 N/M 。
- (2) 选择合适的建议密度函数 $g(x)$ 是算法的关键。在完全使 $Mg(x)$ 罩住 $f(x)$ 前提下， $g(x)$ 的选择原则是：
 - (a) M 要尽可能小；
 - (b) 建议密度函数 $g(x)$ 要容易被抽样，即要求它是很常用的分布或者可以应用上面的逆变换法来抽样；
 - (c) 在满足前两个要求基础上， $Mg(x)$ 尽可能与 $f(x)$ 形似，两者形状越相似，需要削去的部分就越少，这种算法的效率就越高。

接受-拒绝法的两种特殊情形

- 一、概率密度函数是无界的
- 例5.3 抽样无界的贝塔密度函数的分布Beta(α, β)。贝塔的概率密度函数如下：

$$f(x) = cx^{\alpha-1}(1-x)^{\beta-1}, x \in (0, 1)$$

- 其中 $\alpha, \beta > 0$ 是该分布的参数， c 是归一化常数。我们仅考虑当 $\alpha = \beta < 1$ 的情形，这时上面的密度函数在区间 $(0, 1)$ 的端点 $x = 0$ 或 $x = 1$ 是无界的。



肖柳青 上海交通大学 数学系

我们使用如下的建议密度函数：

$$g(x) = c'x^{\alpha-1}, \quad x \in (0, 1/2]$$

其中 c' 是相应的归一化常数。它的分布函数是

$$G(x) = \int_0^x g(t)dt = \int_0^x c't^{\alpha-1}dt = \frac{c'x^\alpha}{\alpha}$$

故 $c' = \alpha(1/2)^{-\alpha}$ 。容易写出该建议分布函数的反函数是

$$Y = \left(\frac{\alpha}{c'}U\right)^{1/\alpha} = \frac{1}{2}U^{1/\alpha}, \quad U \in (0, 1]$$

因为对于 $\alpha < 1$, $(1-x)^{\alpha-1}$ 在区间 $(0, 1/2]$ 上的最大值是在 $x = 1/2$ 处, 所以我们可以取 $M = (1/2)^{\alpha-1}c/c'$, 这样就有

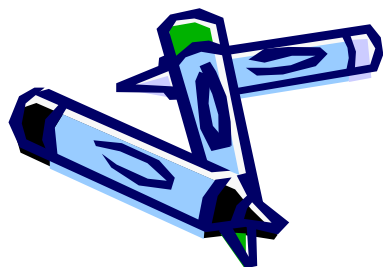
$$Mg(x) = (1/2)^{\alpha-1}cx^{\alpha-1} \geq c(1-x)^{\alpha-1}x^{\alpha-1} = f(x), \quad \forall x \in (0, 1/2]$$

因此, 我们得到了接受准则的概率函数为

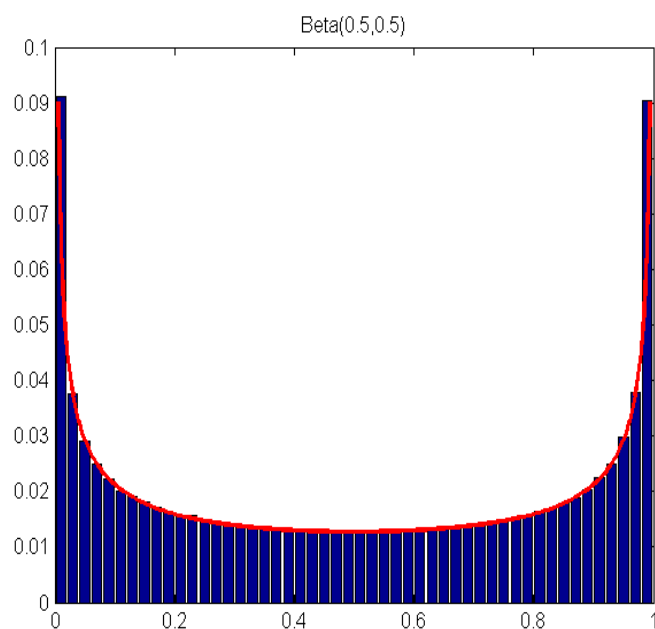
$$h(x) = \frac{f(x)}{Mg(x)} = (2(1-x))^{\alpha-1}, \quad x \in (0, 1/2]$$

抽样该贝塔分布 $\text{Beta}(\alpha, \alpha)$ 的算法:

- (1) 生成随机数 $u \sim U(0, 1)$ 并计算 $y = 1/2u^{\alpha-1}$;
- (2) 再生成随机数 $r \sim U(0, 1)$; 如果 $r < h(y)$, 则接受 y , 否则转到步骤1;
- (3) 还要再生成随机数 $s \sim U(0, 1)$; 如果 $s \leq 1/2$, 则输出 $x = y$, 否则输出 $x = 1 - y$ 。



下面这Matlab程序给出抽样贝塔分布Beta(0.5, 0.5)的算法:



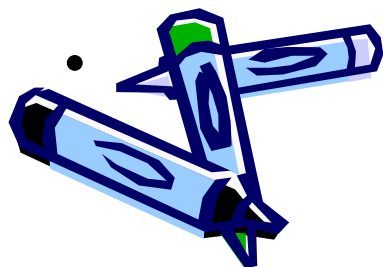
```
N=500000; alpha =0.5;
U = unifrnd ( 0 , 1 , 1 ,N);
R = unifrnd ( 0 , 1 , 1 ,N);
Y=0.5*U. ^ ( 1 / alpha );
h=(2*(1-Y) ) . ^ ( alpha -1);
X=Y(R<h );
sample = length (X);
S = unifrnd ( 0 , 1 , 1 , sample );
S=(S>0.5);
X=(1-S ) . *X+S.*(1-X);
[Xnumber , Xcenter s ]=hist (X, 50 );
bar ( Xcenters ,Xnumber/ sample )
t i t l e ( ' Beta ( 0 . 5 , 0 . 5 ) ' )
hold on
dt =0.005; t=dt : dt :1-dt ;
z = ( t . ^ ( alpha -1)).*((1-t ) . ^ ( alpha -1))/pi ;
s c a l e = 1/50; plot ( t , s c a l e *z , ' r ' )
hold o f f
```

肖柳青 上海交通大学 数学系

5.3 抽样多维联合分布的方法

- 5.3.1 最简单的情形—各分量独立
- 例5.4 生成一个在正方形区域 ($0 \leq x \leq 2, 0 \leq y \leq 2$) 上的二维均匀分布。二维均匀分布在每个维度上的分量都是均匀分布（即两个分量的边缘分布都是(0, 2)上的均匀分布），且两个分量互相独立。我们生成2500个二维均匀分布的随机向量，其Matlab程序如下：

- `N = 2500;`
- `X = unifrnd (0 , 2 , N, 2) ;`
- `scatter (X(:, 1) , X(:, 2) , 'k . ')`



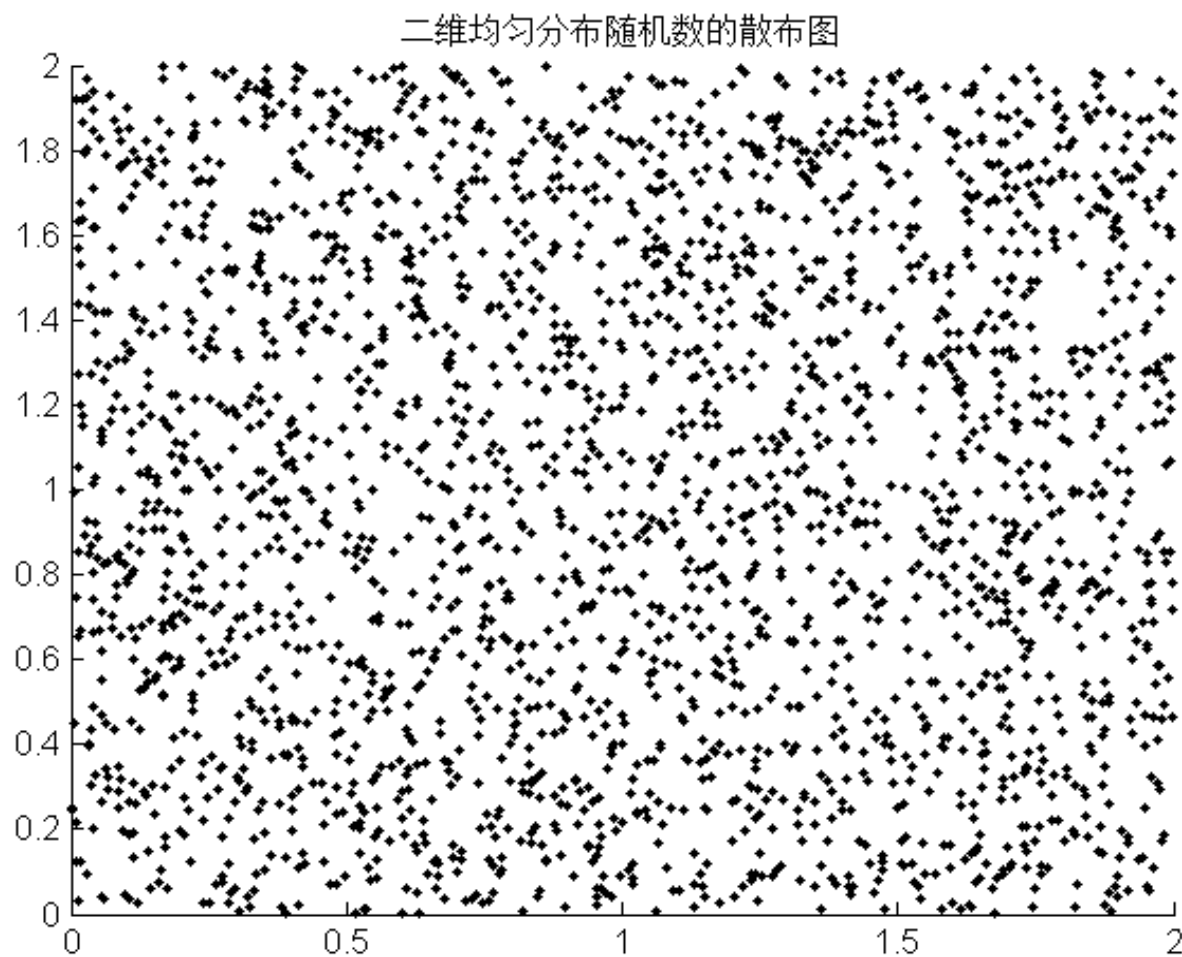
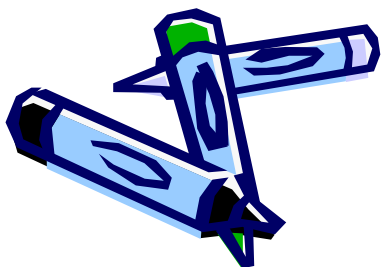
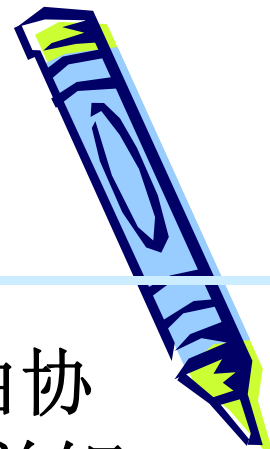


图5.6 二维均匀分布随机数的散布图

5.3.2 基于乔莱斯基分解的 多元正态分布的抽样法

- 一个 n 维的随机向量，各分量之间的相关性由协方差矩阵或者相关系数矩阵来表示。其协方差矩阵为一个 $n \times n$ 阶的矩阵。该矩阵对角线上的元素是随机向量各个分量的方差，矩阵其他位置的元素是各维分量两两之间的协方差。
- 它的相关系数矩阵也是一个 $n \times n$ 阶的矩阵，这矩阵对角线上的元素都是1，其它位置的元素是各维分量两两之间的相关系数。



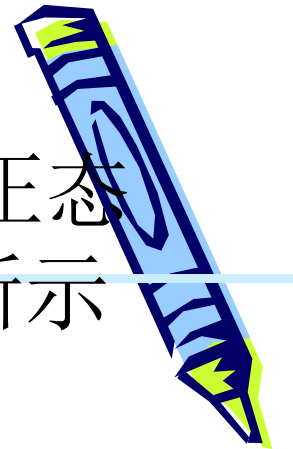
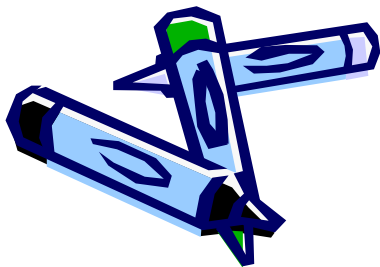
- 例5.6 假定我们要生成一个三维的多元正态分布。各个维度均值和标准差如表5.3所示

表 5.3 一个三维的多元正态分布

维度	均值	标准差
(1)	2	3
(2)	-1	2
(3)	0	1

- 其相关系数矩阵为

$$\rho = \begin{pmatrix} 1 & 0.3 & 0.4 \\ 0.3 & 1 & 0.2 \\ 0.4 & 0.2 & 1 \end{pmatrix}$$

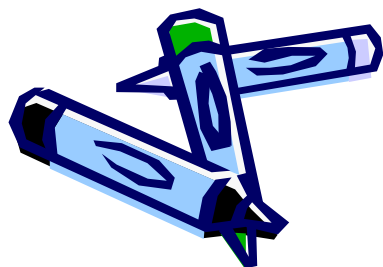


- 设这多维正态分布的随机向量为 X ，其相关系数矩阵为 ρ ，将它作乔莱斯基(Cholesky)分解： $\rho = L L^T$

- 其中 L 是下三角矩阵。令 $Y = L^{-1}X$ ，则容易证明随机向量 Y 的各分量是互不相关的。

$$\text{cov}(Y) = \text{cov}(XL^{-1}) = L^{-1} \text{cov}(XX^T)(L^{-1})^T = L^{-1} \rho (L^{-1})^T = I$$

- 这样我们只要首先生成独立的多维正态分布的随机向量 Y ，然后再做变换 $X = LY$ 即得所要求的随机向量。




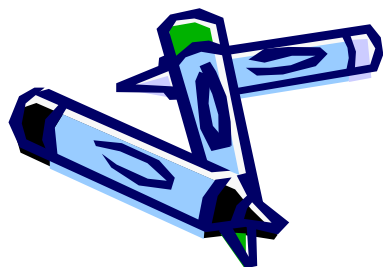
多维正态分布随机向量的生成算法:

- (1) 依照给定的边缘分布的均值和标准差，分别独立地生成各个维度上的服从正态分布的随机数，并按照分量的位置合成一个向量 Y ；
- (2) 将相关系数矩阵 ρ 作乔莱斯基分解： $\rho = LL^T$ ；
- (3) 用分解得到的矩阵乘以第1步中所生成的向量即 $X = LY$ ，这 X 就是我们需要的随机向量。



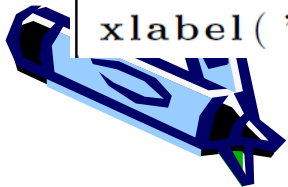
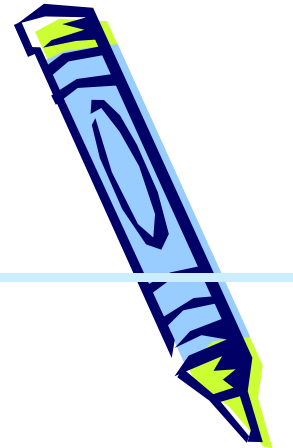
肖柳青 上海交通大学 数学系

- 
- 例5.6的抽样算法步骤如下：
 - (1) 生成各维度上的独立的正态分布随机数，将它们合成向量 Y ；
 - (2) 将相关系数矩阵 ρ 做Cholesky分解得到矩阵 L ($\rho = LL^T$) ；
 - (3) 计算 $X = LY$ ，即可得到服从上述要求的多元正态分布随机向量 X 。



Matlab程序

```
N = 10000;  
Y = [normrnd(2,3,1,N); normrnd(-1,2,1,N); normrnd(0,1,1,N)];  
rho = [1,0.3, 0.4; 0.3, 1, 0.2; 0.4, 0.2, 1];  
L = chol(rho, 'lower');  
X = L*Y;  
subplot(2,2,1)  
scatter(X(1,:), X(2,:), 'marker', '.', 'sizedata', 1)  
xlabel('轴X'); ylabel('轴Y');  
subplot(2,2,2)  
scatter(X(2,:), X(3,:), 'marker', '.', 'sizedata', 1)  
xlabel('轴Y'); ylabel('轴Z');  
subplot(2,2,3)  
scatter(X(3,:), X(1,:), 'marker', '.', 'sizedata', 1)  
xlabel('轴Z'); ylabel('轴X');  
subplot(2,2,4)  
scatter3(X(1,:), X(2,:), X(3,:), 'marker', '.', 'sizedata', 1)  
xlabel('轴X'); ylabel('轴Y'); zlabel('轴Z');
```



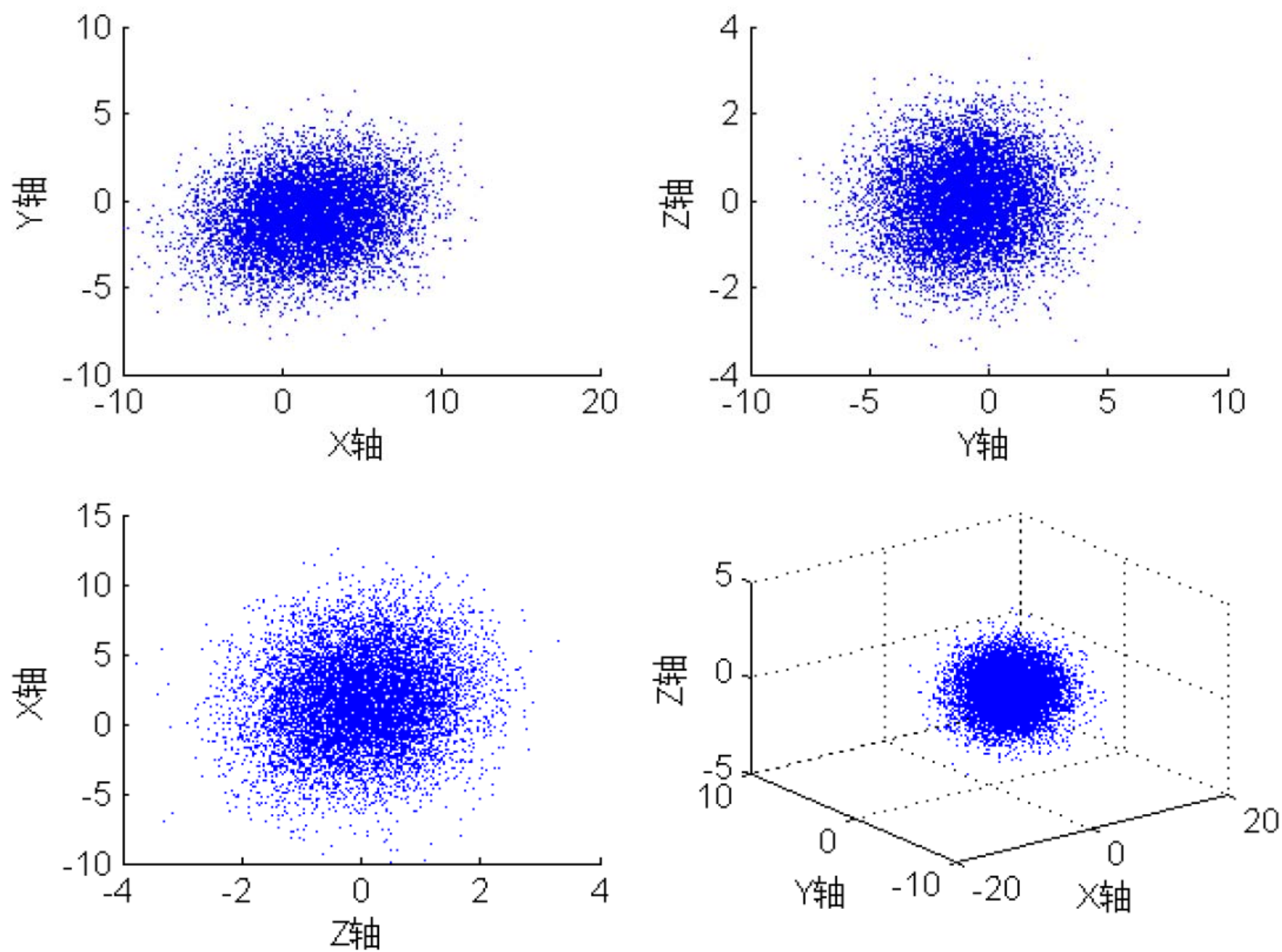


图 5.7 三维正态分布随机向量的散布图

5.4 关于伪随机数的问题的注释

- 1. 由于同一编程语言（在同台计算机上）生成的随机数有固定的算法，总会有一定的周期性规律。所以如果模拟程序由同一个随机数种子来生成很长的随机数序列，则会陷入周期性的境况，那将导致结果出现偏歧。
- 2. 在不同的计算机上运行同一随机模拟程序结果可能会略有差别，但随着重复试验次数的趋于很大，则平均意义上的结果差别应该趋于及其微小。否则，如果结果悬殊，则说明模拟程序存在问题。

