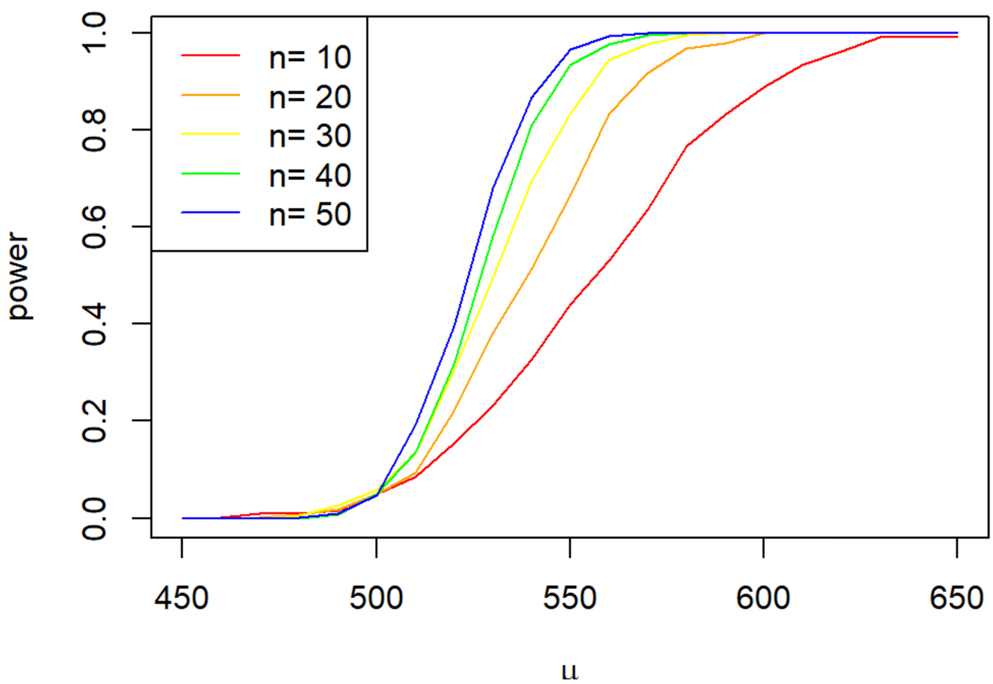
```r
#####   6.3   ###############
n <- c(seq(10, 50, 10))
m <- 1000
mu0 <- 500
sigma <- 100
mu <- c(seq(450, 650, 10))  #alternatives 不同备择参数
M <- length(mu)
power <- numeric(M)
color <- c('red','orange', 'yellow','green','blue')  #设置颜色
label <- c('','','','','')   ##图例空值
for (j in 1:length(n))
{
  n1 <- n[j]
  for (i in 1:M) {
    mu1 <- mu[i]
    pvalues <- replicate(m, expr = {
      #simulate under alternative mu1
      x <- rnorm(n1, mean = mu1, sd = sigma)
      ttest <- t.test(x,
                      alternative = "greater", mu = mu0)
      ttest$p.value  } )
    power[i] <- mean(pvalues <= .05)
  }
  if (j==1) {plot(mu, power, xlab = bquote(mu),col=color[j],'l')} #n=10 用plot
  lines(mu, power, col=color[j],'l')  #n=20..50 用lines
  label[j] <- paste('n=',n[j]) #图例赋值
}
legend("topleft",label, col=color,lty=1) #将图例置于左上角

# 对正态样本做均值t检验(eg6.7)，估计功效并绘制经验功效曲线(eg6.9)，
# 经验功效在接近mu0=500时较小，当mu远离mu0时开始增大，逐渐趋近于1。
# 若改变样本量的大小，则样本量越大，功效增大越快。
```



```r
#####   6.4   ###############
# 使用MC方法得到经验置信水平，理论上应接近1-α=0.95
n <- 20
m <- 1000
alpha <- .05
mean <- 0
sd <- 1
CL <- replicate(m, expr={
  # 生成对数正态分布的样本
  x <- rlnorm(n, mean, sd)
  # 构造均值μ的置信区间上下界
  max <- sum(log(x))/n+qt(1-alpha/2, n-1)*sqrt(var(log(x)))/sqrt(n)
  min <- sum(log(x))/n-qt(1-alpha/2, n-1)*sqrt(var(log(x)))/sqrt(n)
  if(0 > min & 0 < max){beta <- 1}  else{beta <- 0}
  beta
  })
mean(CL) #计算经验置信水平(接近0.95)
```

```r
#####   6.5   ###############
n <- 20
m <- 1000
alpha <- .05
df <- 2
CL <- replicate(m, expr={
  x <- rchisq(n, df)
  mu <- mean(x)
  S <- var(x)
  max <- mu+qt(1-alpha/2, n-1)*S/sqrt(n)
  min <- mu-qt(1-alpha/2, n-1)*S/sqrt(n)
  x>min&x<max
})
> mean(CL) #计算经验置信水平
[1] 0.6751
```

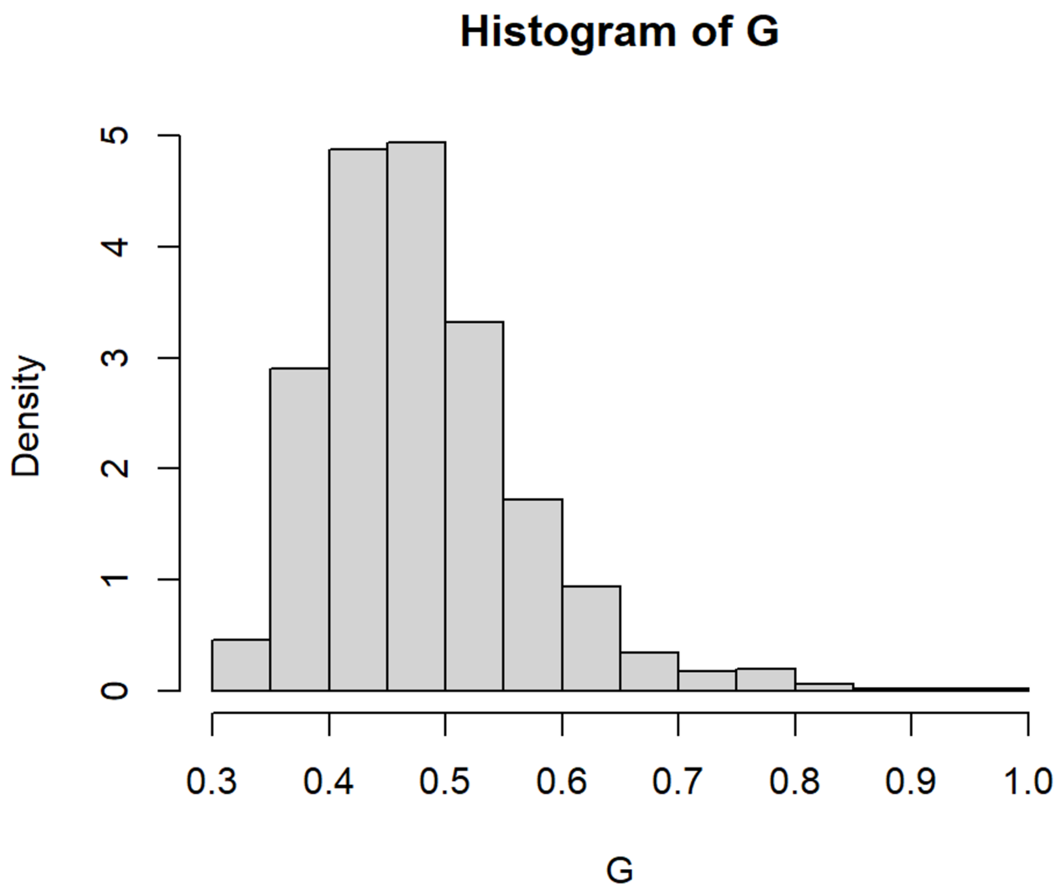
```r
#####   6.8   ###############
# 双样本等方差的Count5检验函数 返回值1(拒绝H0)或0(接受H0)
count5test <- function(x, y) {
  X <- x - mean(x)
  Y <- y - mean(y)
  outx <- sum(X > max(Y)) + sum(X < min(Y))
  outy <- sum(Y > max(X)) + sum(Y < min(X))
  # return 1 (reject) or 0 (do not reject H0)
  return(as.integer(max(c(outx, outy)) > 5))
}

m <- 1000
n <- c(10,20,100)
alpha <- .055
sigma1 <- 1
sigma2 <- 1.5
power.C5 <- numeric(3)
power.F <- numeric(3)
for (i in 1:length(n)) {
  n1 <- n2 <- n[i]
  rep <- replicate(m, expr={
  x <- rnorm(n1, 0, sigma1)
```

```r
  y <- rnorm(n2, 0, sigma2)
  C5 <- count5test(x, y)  ##C5-test
  Ftest <- var.test(x, y, ratio = 1, "two.sided", conf.level = 1-alpha)
  F <- Ftest$p.value < alpha  ##F-test
  c(C5, F)})
  power.C5[i] <- mean(rep[1,])
  power.F[i] <- mean(rep[2,])
}
> data.frame(n=c(10,20,100), power.C5, power.F )
    n power.C5 power.F
1  10    0.116   0.231
2  20    0.323   0.423
3 100    0.854   0.977
# 在样本量分别为10,20,100时，F检验的功效都高于C5检验
```

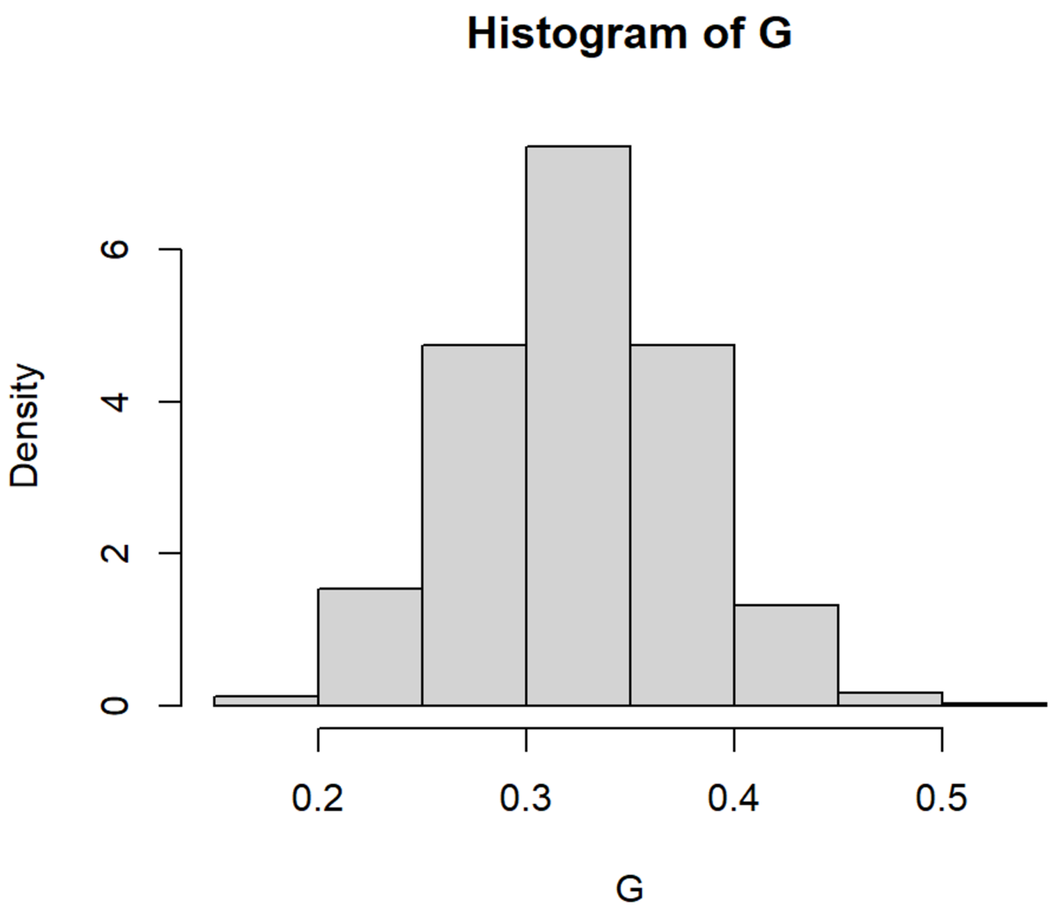
```r
> #####   6.9   ###############
> # 利用次序统计量表示基尼系数(分配公平程度)
> #基于标准对数正态分布
> n <- 20
> m <- 1000
> G <- numeric(m)
> G <- replicate(m,expr={
+   g <- numeric(n)
+   for (i in 1:n){
+     x <- sort(rlnorm(n, mean=0, sd=1)) #排序
+     mu <- mean(x)
+     g[i] <- (2*i-n-1)*x[i]/(n^2*mu)
+   }
+     sum(g)})
> G.mean <- mean(G)
> G.mean
[1] 0.4785041
> G.median <- median(G)
> G.median
[1] 0.4638238
> G.quantile <- quantile(G,c(seq(0.1,0.9,0.1)))
> G.quantile
      10%       20%       30%       40%       50% 
0.3858744 0.4108444 0.4311688 0.4482350 0.4638238 
      60%       70%       80%       90% 
0.4860891 0.5087767 0.5405828 0.5846074 
> hist(G, prob=TRUE)
> 
> #基于均匀分布
> n <- 20
> m <- 1000
> G <- numeric(m)
> G <- replicate(m,expr={
+   g <- numeric(n)
+   x <- sort(runif(n))
+   mu <- mean(x)
+   for (i in 1:n){
+     g[i] <- (2*i-n-1)*x[i]/(n^2*mu)
+   }
+   sum(g)})
> G.mean <- mean(G)
> G.mean
```

```
[1] 0.3217252
> G.median <- median(G)
> G.median
[1] 0.3212548
> G.quantile <- quantile(G,c(seq(0.1,0.9,0.1)))
> G.quantile
      10%       20%       30%       40%       50% 
0.2561058 0.2789920 0.2950767 0.3092741 0.3212548 
      60%       70%       80%       90% 
0.3333215 0.3481908 0.3641689 0.3865076 
> hist(G, prob=TRUE)
> 
> #基于伯努利分布
> n <- 20
> m <- 1000
> G <- numeric(m)
> G <- replicate(m,expr={
+   g <- numeric(n)
+   x <- sort(rbinom(n,1,0.1))
+   mu <- mean(x)
+   for (i in 1:n){
+     if (mu==0){g[i]=0}
+     else{g[i] <- (2*i-n-1)*x[i]/(n^2*mu)}
+   }
+   sum(g)})
> G.mean <- mean(G)
> G.mean
[1] 0.7926
> G.median <- median(G)
> G.median
[1] 0.9
> G.quantile <- quantile(G,c(seq(0.1,0.9,0.1)))
> G.quantile
 10%  20%  30%  40%  50%  60%  70%  80%  90% 
0.00 0.80 0.85 0.85 0.90 0.90 0.90 0.95 0.95 
> hist(G, prob=TRUE)
> ## 基于密度直方图，可以看出贫富差距悬殊。
> # 事实上由于x只取值于0或1，直观上可以推断密度直方图会趋于极端。
```

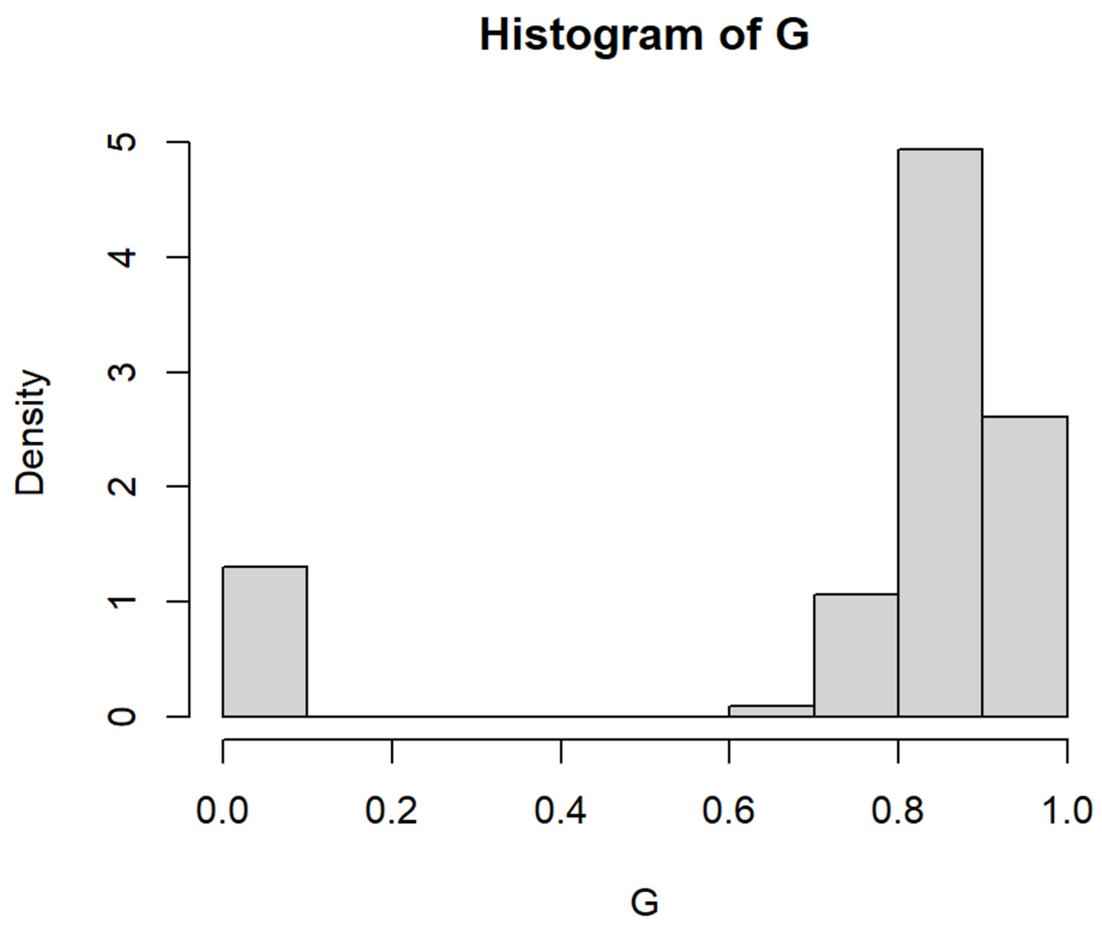
基于标准对数正态分布



基于均匀分布



基于伯努利分布




```
> #####   6A显著性水平的t-test   ################
> #用MC模拟研究 经验error I 是否约等于理论显著水平α
> #注: t-test对微小的正态性偏离是稳定的
>
> #基于卡方分布
> n <- 20
> alpha <- .05
> mu0 <- 1
> m <- 10000          #number of replicates
> p <- numeric(m)     #storage for p-values
> for (j in 1:m) {
+   x <- rchisq(n, df=1)
+   ttest <- t.test(x, alternative = "greater", mu = mu0)
+   p[j] <- ttest$p.value
+ }
> p.hat <- mean(p < alpha) #观测的第一类错误率 (理论上应该接α=0.05)
> se.hat <- sqrt(p.hat * (1 - p.hat) / m) #估计的标准误差
> print(c(p.hat, se.hat))
[1] 0.01300000 0.00113274
> # p-value < 0.02,和0.05差距较大
>
> #基于均匀分布
> n <- 20
> alpha <- .05
> mu0 <- 1
> m <- 10000          #number of replicates
> p <- numeric(m)     #storage for p-values
> for (j in 1:m) {
+   x <- runif(n, 0, 2)
+   ttest <- t.test(x, alternative = "greater", mu = mu0)
```

```
+   p[j] <- ttest$p.value
+ }
> p.hat <- mean(p < alpha) #观测的第一类错误率 (理论上应该接α=0.05)
> se.hat <- sqrt(p.hat * (1 - p.hat) / m) #估计的标准误差
> print(c(p.hat, se.hat))
[1] 0.048700000 0.002152401
> # p-value 接近0.05
>
> #基于指数分布
> n <- 20
> alpha <- .05
> mu0 <- 1
> m <- 10000          #number of replicates
> p <- numeric(m)     #storage for p-values
> for (j in 1:m) {
+   x <- rexp(n, 1)
+   ttest <- t.test(x, alternative = "greater", mu = mu0)
+   p[j] <- ttest$p.value
+ }
> p.hat <- mean(p < alpha) #观测的第一类错误率 (理论上应该接α=0.05)
> se.hat <- sqrt(p.hat * (1 - p.hat) / m) #估计的标准误差
> print(c(p.hat, se.hat))
[1] 0.018900000 0.001361719
> # p-value < 0.02,和0.05差距较大
```

```
> #####   6B   ################
> library(MASS) #加载MASS包
> #二元正态下，pearson比另两种非参数检验得到的pho更接近
> mean<-c(1, 1) #指定均值向量
> sigma<-matrix(c(1, 0.5, 0.5 ,1), nrow=2, ncol=2) #指定协方差矩阵
> mydata <- mvrnorm(100000, mean, sigma)
> cor.test(mydata[,1],mydata[,2],method = "pearson")
      cor
0.5018238

> cor.test(mydata[,1],mydata[,2],method = "spearman")
      rho
0.4856505

> cor.test(mydata[,1],mydata[,2],method = "kendall")
      tau
0.3355287


>
> #在数据非线性单调情况下，非参检验功效更高
> x <- runif(1000,0,1)
> y <- x^6
> cor.test(x,y,method = "pearson")
      cor
0.7786169

> cor.test(x,y,method = "spearman")
rho
  1

> cor.test(x,y,method = "kendall")
sample estimates:
```

```
tau
  1
```