Math 3070 § 1. **Design of a Binomial Experiment:** Name: 𝔈𝔵𝔞𝔪𝔭𝔩𝔢
Treibergs **Programming Conditional Statements** May 20, 2011

How big a sample should we test in order that the Type I and Type II error probabilities are less than some prescribed values? Our solution of this problem will illustrate some more **R**© programming ideas. We will show how a program uses conditional expressions to decide what action to take.

Consider my favorite problem from math 3070 (students work this problem in homework and exams several times during the semester in my class!) Suppose that a manufacturer is shipped a large number of widgets. Assume that the company is willing to accept shipments if the probability that any given widget is faulty is p_0. Your job is to design an experiment to determine whether to accept or reject the shipment depending on how many widgets test faulty. You decide to test a random sample of n of the widgets. If more than k of the widgets in your sample are faulty, then you will reject the sample. Suppose that you are given probabilities α and β of making a Type I or Type II error, and $p_1 > p_0$. The question is, what is the SMALLEST n can be in order that there is a k so that the probability of making a Type I error, of rejecting even though the proportion of faulty widgets is p_0 is at most α, and that the probability of making a Type II error, of accepting when the actual proportion of faulty widgets is p_1, is at most β?

To formulate this question mathematically, let X be the number of faulty widgets in the random sample, and p the shipment proportion of faulty widgets. The cumulative binomial probability is called $\mathrm{B}(x, n, p)$. The probability of rejecting the shipment, given that the proportion of faulty widgets is p_0 is at most α if

$$\mathrm{P}(X > k \,|\, p = p_0) = 1 - \mathrm{B}(k, n, p_0) \le \alpha.$$

The probability of accepting the shipment even though the fraction of faulty widgets is large, p_1, is at most β if

$$\mathrm{P}(X \le k \,|\, p = p_1) = \mathrm{B}(k, n, p_1) \le \beta.$$

Let $g(n)$ be the smallest m such that $1 - \mathrm{B}(m, n, p_0) \le \alpha$. If n is small, m may well be zero, but if it is positive, then $g(n)$ is our candidate for the critical $k(n)$. Then we find the smallest n, call it $h(n)$ so that $B(g(n), n, p_1) \le \beta$. The sample size and maximum tested not rejected, k, are $(h(n), g(n))$. How big should n be so that $g(n) > 0$? This will hold if $\mathrm{B}(0, n, p_0) = b(0, n, p_0) = (1-p_0)^n < 1-\alpha$ or

$$n > \frac{\log(1-\alpha)}{\log(1-p_0)}.$$

The typical student solution of this problem is to search the cumulative biomial tables starting with $n = 25$ (largest in the book) and finding the k's for which both inequalities hold. Then trying smaller n until no such k is possible. We will automate the process.

R Session:

```
R version 2.10.1 (2009-12-14)
Copyright (C) 2009 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale
```

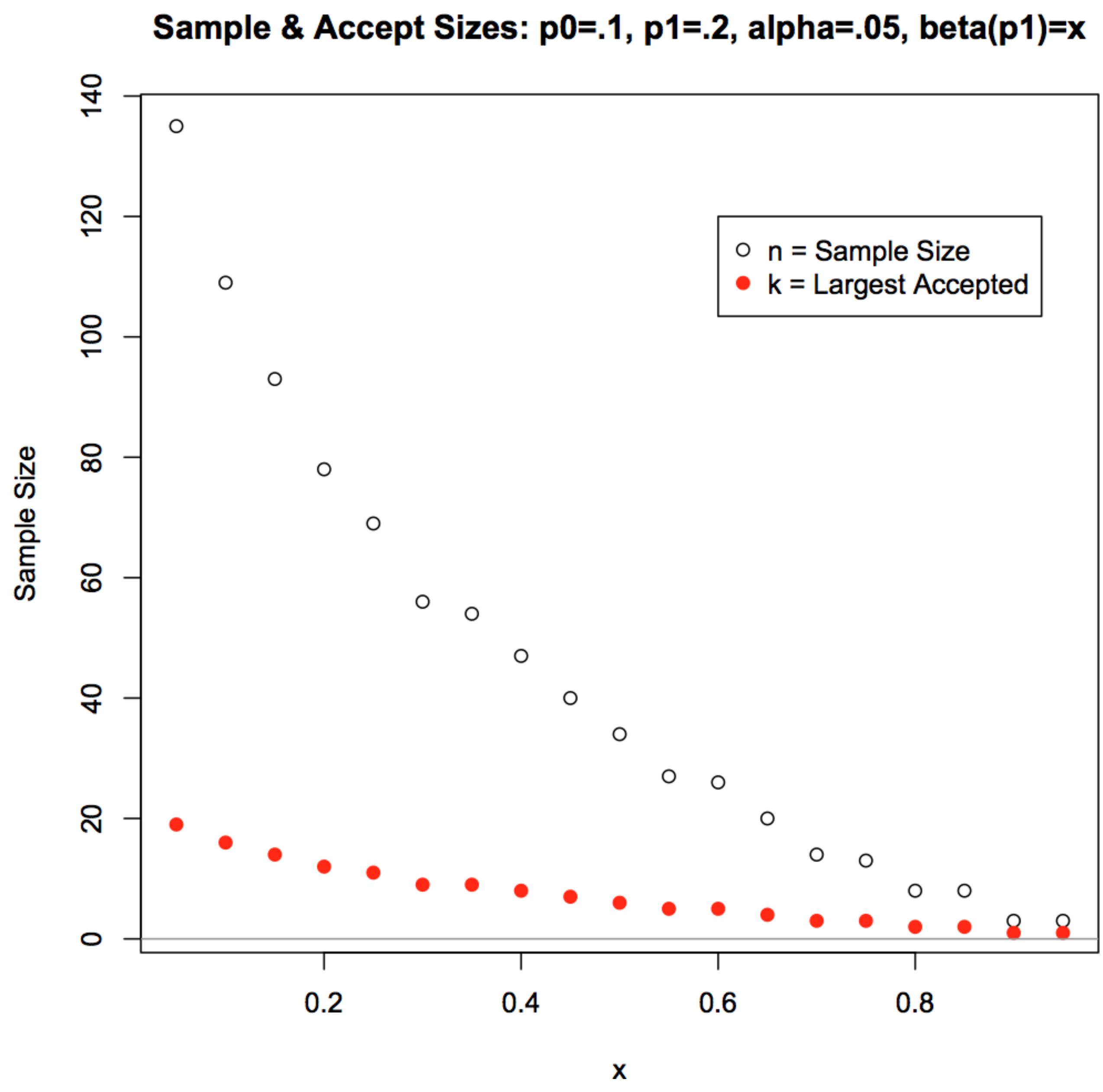
```
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[R.app GUI 1.31 (5538) powerpc-apple-darwin8.11.1]

[Workspace restored from /Users/andrejstreibergs/.RData]

> ############## FUNCTION TO GIVE LEAST n GIVEN p0 AND alpha ###########
>
> # The loop is constructed using while loop
> #             while(condition){code1}
> # Given p, a, we start with v=0 and if P(X>v | p)> a keep incrementing v
> # until it is not the case, and then output  v. In other words, v
> # is the smallest  n  such that P(X <= n | p=p0) <= a.
> # We know that the incrementing does not go beyond  n  because
> # 1-pbinom(n,n,p0) = 0 <= a.
> g <- function(n,p0,a){
+                       v<-0
+                       while(pbinom(v,n,p0)<1-a)
+                                                {
+                                                 v <- v+1
+                                                }
+                       return(v)
+                      }
>
> # This is BAD PROGRAMMING because there is no idiot proofing. With bad
> # choices of the inputs we might get an infinite loop, which would require
> # interrupting the execution (escape key on my MAC). However, since
> # anybody using this code will be careful to provide correct input
> # we shall not say more.
>
> # In fact, the critical value for a given probability is a canned
> # function in  R. A better way to define  g  would have been
> # g <- function(n,p0,a){qbinom(a,n,p0,lower.tail=F)}
>
> g(20,.3,.2)
[1] 8
>
> # k is zero until log(1-alpha)/log(1-p0) < n.
>
> # Fix p0 and alpha and tabulate over several n's
>
> alpha <- .05;  p0 <- .1
> cat(log(1-alpha)/log(1-p0)," alpha=",alpha," p0=",p0,"\n");
+ for(j in 1:20)
+               {
+                cat( " n=",j ,"  g=",g(j,p0,alpha),"1- P(g(k)-1)=",
+                1-pbinom(g(j,p0,alpha)-1,j,p0),"1-P(g(k))=",
+                1-pbinom((j,p0,alpha),j,p0),"\n")
+               }
0.486836  alpha= 0.05  p0= 0.1
 n= 1   g= 1 1- P(g(k)-1)= 0.1 1-P(g(k))= 0
 n= 2   g= 1 1- P(g(k)-1)= 0.19 1-P(g(k))= 0.01
 n= 3   g= 1 1- P(g(k)-1)= 0.271 1-P(g(k))= 0.028
 n= 4   g= 2 1- P(g(k)-1)= 0.0523 1-P(g(k))= 0.0037
 n= 5   g= 2 1- P(g(k)-1)= 0.08146 1-P(g(k))= 0.00856
 n= 6   g= 2 1- P(g(k)-1)= 0.114265 1-P(g(k))= 0.01585
 n= 7   g= 2 1- P(g(k)-1)= 0.1496944 1-P(g(k))= 0.0256915
 n= 8   g= 2 1- P(g(k)-1)= 0.1868953 1-P(g(k))= 0.03809179
 n= 9   g= 3 1- P(g(k)-1)= 0.05297214 1-P(g(k))= 0.008331094
 n= 10   g= 3 1- P(g(k)-1)= 0.07019083 1-P(g(k))= 0.01279520
 n= 11   g= 3 1- P(g(k)-1)= 0.08956185 1-P(g(k))= 0.01853476
 n= 12   g= 3 1- P(g(k)-1)= 0.1108700 1-P(g(k))= 0.02563747
 n= 13   g= 3 1- P(g(k)-1)= 0.1338828 1-P(g(k))= 0.03416072
 n= 14   g= 3 1- P(g(k)-1)= 0.1583600 1-P(g(k))= 0.04413292
 n= 15   g= 4 1- P(g(k)-1)= 0.05555563 1-P(g(k))= 0.01272048
 n= 16   g= 4 1- P(g(k)-1)= 0.06840617 1-P(g(k))= 0.01700400
 n= 17   g= 4 1- P(g(k)-1)= 0.08264062 1-P(g(k))= 0.02214422
 n= 18   g= 4 1- P(g(k)-1)= 0.09819684 1-P(g(k))= 0.02819386
 n= 19   g= 4 1- P(g(k)-1)= 0.1149976 1-P(g(k))= 0.03519416
 n= 20   g= 4 1- P(g(k)-1)= 0.1329533 1-P(g(k))= 0.04317450
```

```
> alpha<-.2
> cat(log(1-alpha)/log(1-p0)," alpha=",alpha," p0=",p0,"\n")
+ for(j in 1:20)
+               {
+                cat( " n=",j ,"  g=",g(j,p0,alpha),"1- P(g(k)-1)=",
+                1-pbinom(g(j,p0,alpha)-1,j,p0),"1-P(g(k))=",
+                1-pbinom((j,p0,alpha),j,p0),"\n")
+               }
2.117905  alpha= 0.2  p0= 0.1
 n= 1   g= 0 1- P(g(k)-1)= 1 1-P(g(k))= 0.1
 n= 2   g= 0 1- P(g(k)-1)= 1 1-P(g(k))= 0.19
 n= 3   g= 1 1- P(g(k)-1)= 0.271 1-P(g(k))= 0.028
 n= 4   g= 1 1- P(g(k)-1)= 0.3439 1-P(g(k))= 0.0523
 n= 5   g= 1 1- P(g(k)-1)= 0.40951 1-P(g(k))= 0.08146
 n= 6   g= 1 1- P(g(k)-1)= 0.468559 1-P(g(k))= 0.114265
 n= 7   g= 1 1- P(g(k)-1)= 0.5217031 1-P(g(k))= 0.1496944
 n= 8   g= 1 1- P(g(k)-1)= 0.5695328 1-P(g(k))= 0.1868953
 n= 9   g= 2 1- P(g(k)-1)= 0.225159 1-P(g(k))= 0.05297214
 n= 10   g= 2 1- P(g(k)-1)= 0.2639011 1-P(g(k))= 0.07019083
 n= 11   g= 2 1- P(g(k)-1)= 0.3026431 1-P(g(k))= 0.08956185
 n= 12   g= 2 1- P(g(k)-1)= 0.3409977 1-P(g(k))= 0.1108700
 n= 13   g= 2 1- P(g(k)-1)= 0.378655 1-P(g(k))= 0.1338828
 n= 14   g= 2 1- P(g(k)-1)= 0.4153709 1-P(g(k))= 0.1583600
 n= 15   g= 2 1- P(g(k)-1)= 0.450957 1-P(g(k))= 0.1840611
 n= 16   g= 3 1- P(g(k)-1)= 0.2107507 1-P(g(k))= 0.06840617
 n= 17   g= 3 1- P(g(k)-1)= 0.2382028 1-P(g(k))= 0.08264062
 n= 18   g= 3 1- P(g(k)-1)= 0.266204 1-P(g(k))= 0.09819684
 n= 19   g= 3 1- P(g(k)-1)= 0.2945552 1-P(g(k))= 0.1149976
 n= 20   g= 3 1- P(g(k)-1)= 0.3230732 1-P(g(k))= 0.1329533
```

```
> alpha<-1/9
> p0<-1/11
> cat(log(1-alpha)/log(1-p0)," alpha=",alpha," p0=",p0,"\n")
+ for(j in 1:20)
+               {
+                cat( " n=",j ,"  g=",g(j,p0,alpha),"1- P(g(k)-1)=",
+                1-pbinom(g(j,p0,alpha)-1,j,p0),"1-P(g(k))=",
+                1-pbinom((j,p0,alpha),j,p0),"\n")
+               }
1.235787  alpha= 0.1111111  p0= 0.0909091
 n= 1   g= 0 1- P(g(k)-1)= 1 1-P(g(k))= 0.09090909
 n= 2   g= 1 1- P(g(k)-1)= 0.1735537 1-P(g(k))= 0.008264463
 n= 3   g= 1 1- P(g(k)-1)= 0.2486852 1-P(g(k))= 0.02329076
 n= 4   g= 1 1- P(g(k)-1)= 0.3169865 1-P(g(k))= 0.04378116
 n= 5   g= 1 1- P(g(k)-1)= 0.3790787 1-P(g(k))= 0.06861802
 n= 6   g= 1 1- P(g(k)-1)= 0.4355261 1-P(g(k))= 0.09684171
 n= 7   g= 2 1- P(g(k)-1)= 0.1276312 1-P(g(k))= 0.01986799
 n= 8   g= 2 1- P(g(k)-1)= 0.1602867 1-P(g(k))= 0.02966465
 n= 9   g= 2 1- P(g(k)-1)= 0.1942145 1-P(g(k))= 0.04153938
 n= 10   g= 2 1- P(g(k)-1)= 0.2289134 1-P(g(k))= 0.05541894
 n= 11   g= 2 1- P(g(k)-1)= 0.2639628 1-P(g(k))= 0.07119117
 n= 12   g= 2 1- P(g(k)-1)= 0.2990122 1-P(g(k))= 0.08871586
 n= 13   g= 2 1- P(g(k)-1)= 0.3337719 1-P(g(k))= 0.1078337
 n= 14   g= 3 1- P(g(k)-1)= 0.1283735 1-P(g(k))= 0.03252097
 n= 15   g= 3 1- P(g(k)-1)= 0.1501582 1-P(g(k))= 0.04123484
 n= 16   g= 3 1- P(g(k)-1)= 0.1730093 1-P(g(k))= 0.05113697
 n= 17   g= 3 1- P(g(k)-1)= 0.1967506 1-P(g(k))= 0.06221627
 n= 18   g= 3 1- P(g(k)-1)= 0.2212114 1-P(g(k))= 0.07444667
 n= 19   g= 3 1- P(g(k)-1)= 0.2462282 1-P(g(k))= 0.08778892
 n= 20   g= 3 1- P(g(k)-1)= 0.2716462 1-P(g(k))= 0.1021925
```

```
> # Print a list of the values to be plotted.
> alpha <- .1; p0 <- .2
>
> cat(log(1-alpha)/log(1-p0)," alpha=",alpha," p0=",p0,"\n")
+ for(j in 1:20)
+              {
+               cat( " n=",j ,"  g=",g(j,p0,alpha), "1- P(g(k)-1)=",
+               1-pbinom(g(j,p0,alpha)-1, j, p0), "1-P(g(k))=",
+               1-pbinom(g(j,p0,alpha), j, p0), "\n")
+              }
2.054080  alpha= 0.1  p0= 0.05
 n= 1   g= 0 1- P(g(k)-1)= 1 1-P(g(k))= 0.05
 n= 2   g= 0 1- P(g(k)-1)= 1 1-P(g(k))= 0.0975
 n= 3   g= 1 1- P(g(k)-1)= 0.142625 1-P(g(k))= 0.00725
 n= 4   g= 1 1- P(g(k)-1)= 0.1854938 1-P(g(k))= 0.01401875
 n= 5   g= 1 1- P(g(k)-1)= 0.2262191 1-P(g(k))= 0.0225925
 n= 6   g= 1 1- P(g(k)-1)= 0.2649081 1-P(g(k))= 0.03277383
 n= 7   g= 1 1- P(g(k)-1)= 0.3016627 1-P(g(k))= 0.04438054
 n= 8   g= 1 1- P(g(k)-1)= 0.3365796 1-P(g(k))= 0.05724465
 n= 9   g= 1 1- P(g(k)-1)= 0.3697506 1-P(g(k))= 0.0712114
 n= 10   g= 1 1- P(g(k)-1)= 0.4012631 1-P(g(k))= 0.08613836
 n= 11   g= 2 1- P(g(k)-1)= 0.1018946 1-P(g(k))= 0.01523530
 n= 12   g= 2 1- P(g(k)-1)= 0.1183599 1-P(g(k))= 0.01956826
 n= 13   g= 2 1- P(g(k)-1)= 0.1354239 1-P(g(k))= 0.02450784
 n= 14   g= 2 1- P(g(k)-1)= 0.1529856 1-P(g(k))= 0.03005364
 n= 15   g= 2 1- P(g(k)-1)= 0.1709525 1-P(g(k))= 0.03620024
 n= 16   g= 2 1- P(g(k)-1)= 0.1892403 1-P(g(k))= 0.04293785
 n= 17   g= 2 1- P(g(k)-1)= 0.207772 1-P(g(k))= 0.05025298
 n= 18   g= 2 1- P(g(k)-1)= 0.2264774 1-P(g(k))= 0.05812893
 n= 19   g= 2 1- P(g(k)-1)= 0.2452928 1-P(g(k))= 0.06654635
 n= 20   g= 2 1- P(g(k)-1)= 0.2641605 1-P(g(k))= 0.07548367


>
> ############# FUNCTION TO FIND LEAST k THAT HAS P(II) <= b FOR OPTIMAL n ###########
>
> # Using the while loop, increment  w  until you satisfy P(II) <= beta
>
>  h<-function(q0,q1,a,b)
+                       {
+                        w <- 0
+                        while(pbinom(g(w,q0,a),w,q1)>b)
+                                                        {
+                                                         w<-w+1
+                                                        }
+                        return(w)
+                       }
>
> h(.1,.2,.075,.08)
[1] 106
```

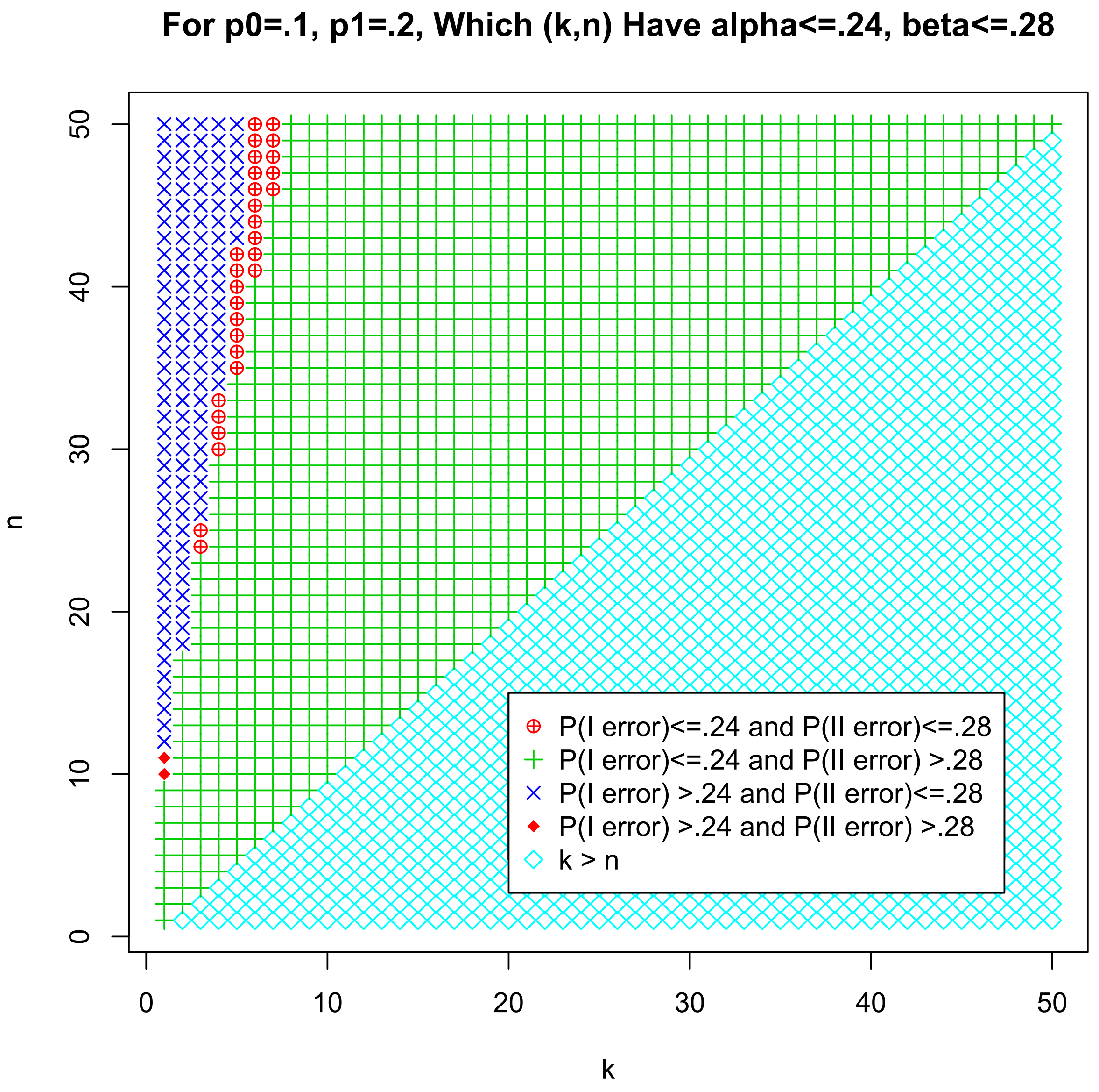
```
> # Table of (n,k) for p0=.1, p1=.2 and  beta  running from .05 to .95
>
> xs <- seq(from=.05,to=.95,by=.05)
> xs
 [1] 0.05 0.10 0.15 0.20 0.25 0.30 0.35 0.40 0.45 0.50 0.55 0.60 0.65 0.70
[15] 0.75 0.80 0.85 0.90 0.95

> for (be in xs)
+             {
+              cat(k, "(n,k) = (", h(.1,.2,.05,be), ",", g(h(.1,.2,.05,be),.1,.05), ") \n")
+             }
0.05 (n,k) = ( 135 , 19 )
0.1 (n,k) = ( 109 , 16 )
0.15 (n,k) = ( 93 , 14 )
0.2 (n,k) = ( 78 , 12 )
0.25 (n,k) = ( 69 , 11 )
0.3 (n,k) = ( 56 , 9 )
0.35 (n,k) = ( 54 , 9 )
0.4 (n,k) = ( 47 , 8 )
0.45 (n,k) = ( 40 , 7 )
0.5 (n,k) = ( 34 , 6 )
0.55 (n,k) = ( 27 , 5 )
0.6 (n,k) = ( 26 , 5 )
0.65 (n,k) = ( 20 , 4 )
0.7 (n,k) = ( 14 , 3 )
0.75 (n,k) = ( 13 , 3 )
0.8 (n,k) = ( 8 , 2 )
0.85 (n,k) = ( 8 , 2 )
0.9 (n,k) = ( 3 , 1 )
0.95 (n,k) = ( 3 , 1 )


> ######################## PLOT n And k FOR p0=.1, alpha=.05 p1=.2 VS b ###################
> length(xs)
[1] 19
>
> # Fill n vectors  nn  and  kk  with values of  n  and  k  as  b runs through  xs.
>
> nn <- 1:19; kk <- 1:19
> for (i in 1:19)
+               {
+                nn[i] <- h(.1,.2,.05,xs[i])
+                kk[i] <- g(h(.1,.2,.05,xs[i]),.1,.05)
+               }


> plot(xs, nn, main="Sample & Accept Sizes: p0=.1, p1=.2, alpha=.05, beta(p1)=x",
+ ylab="Sample Size", xlab="x")
> points(xs,kk,type="p",pch=19,col=2)
> legend(.6,120,legend=c("n = Sample Size","k = Largest Accepted"),pch=c(1,19),col=1:2)
> abline(h=0,col=grey(.5))
> # M3074ProgIf1.pdf
```



Sample & Accept Sizes: p0=.1, p1=.2, alpha=.05, beta(p1)=x
n = Sample Size
k = Largest Accepted
Sample Size
x
0
20
40
60
80
100
120
140
0.2
0.4
0.6
0.8

```
> ## PLOT n, k, ACTUAL alpha, beta FOR p1=.1,p2=.2 AND TARGET alpha=.05, beta FROM .05 TO .95 ##
>
> x1 <- x2 <- x3 <- x4 <- 1:19
> for(i in 1:19 ){
+                x1[i] <- h(.1,.2,.05,xs[i])
+                x2[i] <- kk(.1,.2,.05,xs[i])
+                x3[i] <- 1-pbinom(x2[i],x1[i],.1)
+                x4[i] <- pbinom(x2[i],x1[i],.2)
+               }
> matr <- cbind(xs,x1,x2,x3,x4)
> rownames(matr) <- rep("",19)
> colnames(matr) <-
+ c("Target beta","    n","    k"," actual alpha","actual beta")
> cat("\n\n Sample=n,  Accepted=k, Using p0=.1, p1=.2, target alpha=.05\n");matr


 Sample=n,  Accepted=k, Using p0=.1, p1=.2, target alpha=.05
 Target beta     n     k  actual alpha actual beta
        0.05   135    19    0.04835530  0.04896758
        0.10   109    16    0.04320808  0.09907705
        0.15    93    14    0.04287804  0.14283787
        0.20    78    12    0.04528574  0.19182084
        0.25    69    11    0.04001790  0.24956878
        0.30    56     9    0.04941135  0.29320357
        0.35    54     9    0.03977524  0.33960989
        0.40    47     8    0.04112599  0.38431734
        0.45    40     7    0.04190194  0.43714590
        0.50    34     6    0.04814433  0.46613988
        0.55    27     5    0.04705689  0.53865673
        0.60    26     5    0.03985931  0.57748639
        0.65    20     4    0.04317450  0.62964826
        0.70    14     3    0.04413292  0.69818988
        0.75    13     3    0.03416072  0.74732431
        0.80     8     2    0.03809179  0.79691776
        0.85     8     2    0.03809179  0.79691776
        0.90     3     1    0.02800000  0.89600000
        0.95     3     1    0.02800000  0.89600000


> ####### TABLE OF (n,k) FOR p0,p1 FIXED FOR SEVERAL alpha, beta ##########
>
> # Let us choose a range for alphas and betas.
> u<-(1:14)/25;u
 [1] 0.04 0.08 0.12 0.16 0.20 0.24 0.28 0.32 0.36 0.40 0.44 0.48 0.52 0.56
>
> # The table will put n above k in each cell. Thus every other row name is ""
>
> ucn <- c()
> for(i in 1:14){ucn<-c(ucn,u[i],"")}
> # Fill in the matrix with n=h(0 and k=kk().
>
> um <- matrix(1:392,ncol=14)
> for(i in 1:14)
+               {
+                for(j in 1:14)
+                              {
+                               um[2*i-1,j] <- h(.1,.2,u[i],u[j])
+                               um[2*i,j] <- kk(.1,.2,u[i],u[j])
+                              }
+               }
> colnames(um) <- u
> rownames(um) <- ucn
> cat("\n Sample Size and Max Accepted for Target alpha, beta Using p0=.1,
+ p2=.2\nalpha \\ beta              n\n                         k\n");um

 Sample Size and Max Accepted for Target alpha, beta Using p0=.1, p2=.2
alpha \ beta                        n
                                    k
     0.04 0.08 0.12 0.16 0.2 0.24 0.28 0.32 0.36 0.4 0.44 0.48 0.52 0.56
0.04  155  124  107   92  83   75   68   61   54  52   46   39   38   32
       22   18   16   14  13   12   11   10    9   9    8    7    7    6
0.08  126  101   84   75  61   59   52   45   38  37   30   29   23   22
       17   14   12   11   9    9    8    7    6   6    5    5    4    4
0.12  114   89   73   64  56   48   41   34   33  26   25   19   18   12
       15   12   10    9   8    7    6    5    5   4    4    3    3    2
0.16  102   77   61   52  44   37   35   28   27  21   20   14   13   12
       13   10    8    7   6    5    5    4    4   3    3    2    2    2
0.2    90   65   55   46  39   31   30   23   22  15   15   14    8    8
       11    8    7    6   5    4    4    3    3   2    2    2    1    1
0.24   84   59   49   41  33   25   24   17   16  15    9    9    8    8
       10    7    6    5   4    3    3    2    2   2    1    1    1    1
0.28   71   53   43   35  33   25   18   17   16  10    9    9    3    3
        8    6    5    4   4    3    2    2    2   1    1    1    0    0
0.32   71   46   37   35  27   19   18   17   11  10    9    9    3    3
        8    5    4    4   3    2    2    2    1   1    1    1    0    0
0.36   58   46   37   29  21   19   12   12   11  10    4    4    3    3
        6    5    4    3   2    2    1    1    1   1    0    0    0    0
0.4    58   40   31   22  21   13   12   12   11  10    4    4    3    3
        6    4    3    2   2    1    1    1    1   1    0    0    0    0
0.44   52   40   24   22  14   13   12   12    5   5    4    4    3    3
        5    4    2    2   1    1    1    1    0   0    0    0    0    0
0.48   45   34   24   22  14   13    6    6    5   5    4    4    3    3
        4    3    2    2   1    1    0    0    0   0    0    0    0    0
0.52   45   27   17   16  14   13    6    6    5   5    4    4    3    3
        4    2    1    1   1    1    0    0    0   0    0    0    0    0
0.56   38   27   17   16  14    7    6    6    5   5    4    4    3    3
        3    2    1    1   1    0    0    0    0   0    0    0    0    0
```

```
> ####################### TABLE PROBABILITIES FOR ALL (n,k) #############
>
> # We fix p0=.1, p1=.2, alpha=.24, beta=.28.
> # From the table above, the minimum sample size and max acceptable faulty
> # widgets is (n,k)=(24,3).
>
> # In this section we do a "bonehead" check that this is correct.
> # We will run through k,n from 1 to 50 and compute the probabilities:
> # In sample size n, p is prob widget is faulty, X is number faulty in sample.
> # Type I error: sample is rejected even though p0=.1
> # P(I error)=P(X>k | p=p0) = i-pbinom(k,n,p0)
> # Type II error: sample is accepted even though p0=.2
> # P(II error)=P(X<=k | p=p1) = pbinom(k,n,p1)
>
> # For each (k,n) we compute P(I) and P(II) and plot at (n,k)
> # different characters depending on condition:
>

> # In the first case,  k <= n. If in addition
> #     P(I) <= alpha  and P(II) <= beta  then  draw red o-plus;
> #     P(I) <= alpha  and P(II)  > beta  then  draw green plus;
> #     P(I)  > alpha  and P(II) <= beta  then  draw blue cross;
> #     P(I)  > alpha  and P(II)  > beta  then  draw red lozenge.
> # In the second case, k > n,  draw cyan diamond.
>
> # We make a list of all possible pairs (k,n) = (X[i],Y[i]).
> # For each pair, we use  if  statements n the form
> #       if( condition ) {code1} else {code2}
> # which means: if the  condition  is true then execute  code1 ,
> # otherwise execute  code2.
> # The nesting is as follows. Outside if asks if k <= n. If true
> # we go on to check if P(I) <= alpha. If true we check  P(II) <= beta.
> # If both are true we record o-plus. If the middle is true but the last
> # is false (the first  else) we record the plus.
> # But the middle could have been false.  Then we check P(II) <= beta
> # and record a cross if the third is true and a lozenge if third is false.
> # The last else corresponds to k <= n is false and we record a diamond.
>
> X <- rep(1:50,50)
> Y <- rep(1:50,rep(50,50))
> z <- 1:2500
> for(i in 1:2500)
+     {
+      if(X[i]<=Y[i])
+         {
+          if(1-pbinom(X[i],Y[i],.1)<=.24)
+                       {
+                        if(pbinom(X[i],Y[i],.2)<=.28)
+                                  z[i] <- 10
+                        else
+                                  z[i] <- 3
+                       }
+           else
+                       {
+                        if(pbinom(X[i],Y[i],.2)<=.28)
+                                  z[i] <- 4
+                        else
+                                  z[i] <- 18
+                       }
+           }
+       else
+           {
+            z[i]<-5
            }
+    }
> plot(X, Y, type="p", col=z, pch=z, xlab="k", ylab="n",
+ main="For p0=.1, p1=.2, Which (k,n) Have alpha<=.24, beta<=.28")
>  legend(20, 15, legend= c("P(I error)<=.24 and P(II error)<=.28",
+  "P(I error)<=.24 and P(II error) >.28", "P(I error) >.24 and P(II error)<=.28",
+  "P(I error) >.24 and P(II error) >.28","k > n"), bg="white",
+  pch=c(10,3,4,18,5), col=c(10,3,4,18,5))
> # M3074ProgIf2.pdf
```



For p0=.1, p1=.2, Which (k,n) Have alpha<=.24, beta<=.28
n
k
0
10
20
30
40
50
⊕ P(I error)<=.24 and P(II error)<=.28
+ P(I error)<=.24 and P(II error) >.28
× P(I error) >.24 and P(II error)<=.28
◆ P(I error) >.24 and P(II error) >.28
◇ k > n