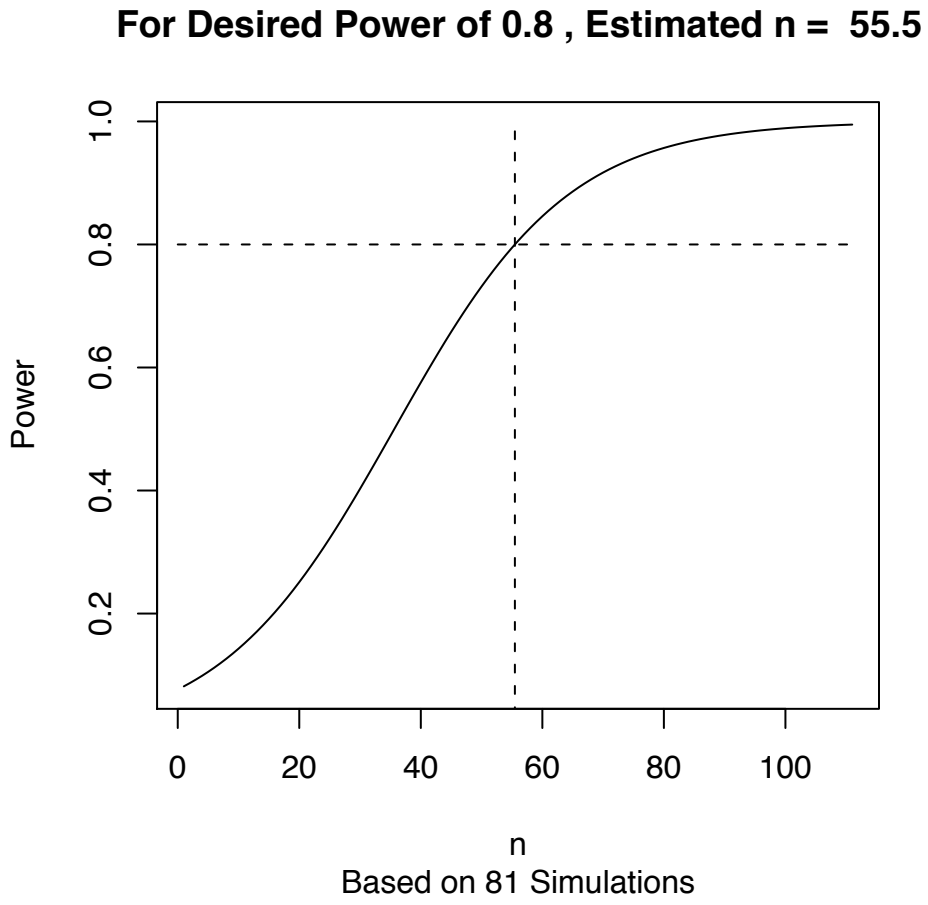
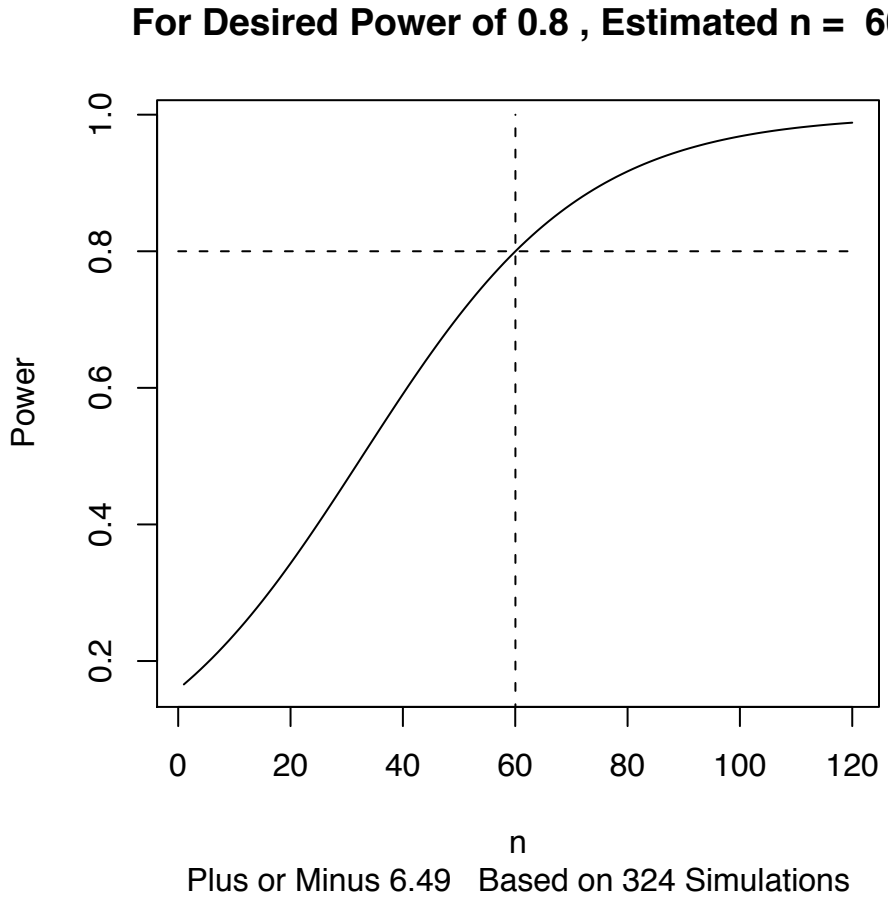
Logistic regression of simulated data

```
> ############################################################
> # Try for a good way to zero in on n
> # Power analysis by Monte Carlo
> ############################################################
>
> onesim <- function(nn) # Two-sample t-test with equal variances
+ # For detecting delta = |mu1-mu2|/sigma = 1/2 with prob .8, need
+ # n1 = n2 = 64 for n of 128
+     {
+     xx <- rnorm(nn); yy <- rnorm(nn)+.5 # Equal sample sizes
+     onesim <- as.numeric(t.test(xx,yy,var.equal=T)[3]<0.05)
+     names(onesim) <- NULL
+     onesim
+     } # End function onesim
>
> onesim(12)
[1] 0
> onesim(300)
[1] 1
> N = seq(from=20,to=100,by=5); sig=numeric(length(N))
> for(i in 1:length(N)) sig[i] = onesim(N[i])
> cbind(N,sig)
        N sig
 [1,]  20   0
 [2,]  25   1
 [3,]  30   0
 [4,]  35   0
 [5,]  40   1
 [6,]  45   1
 [7,]  50   0
 [8,]  55   1
 [9,]  60   0
[10,]  65   1
[11,]  70   1
[12,]  75   1
[13,]  80   1
[14,]  85   1
[15,]  90   1
[16,]  95   1
[17,] 100   1
>
> set.seed(4444); Pi = 0.80 # Desired power
> N = 20:100; M = length(N); sig=numeric(M)
> for(i in 1:M) sig[i] = onesim(N[i])
> mod = glm(sig~N,family=binomial)
> b0 <- coefficients(mod)[1]; b1 = coefficients(mod)[2]
> wantpow = 0.80; logodds = log( Pi/(1- Pi))
> n0hat = (logodds-b0)/b1
> names(n0hat) = "n0-hat" ; n0hat # 55.5
  n0-hat
55.48018
```

```
>
> # Plot
> top <- max(round(2*n0hat),max(N))
> plotxx <- 1:top
> exb <- exp(b0+plotxx*b1)
> phat <- exb/(1+exb)
> plot(plotxx,phat,type='l',xlab='n',ylab='Power')
> # Crosshairs
> lines(c(n0hat,n0hat),c(0,1),lty=2) ; lines(c(0,top),c(Pi,Pi),lty=2)
> kaption1 <- paste("For Desired Power of",round(Pi,3),
+ ", Estimated n = ",round(n0hat,1))
> kaption2 <- paste("Based on",M,"Simulations")
> title(main=kaption1,sub=kaption2)
>
```

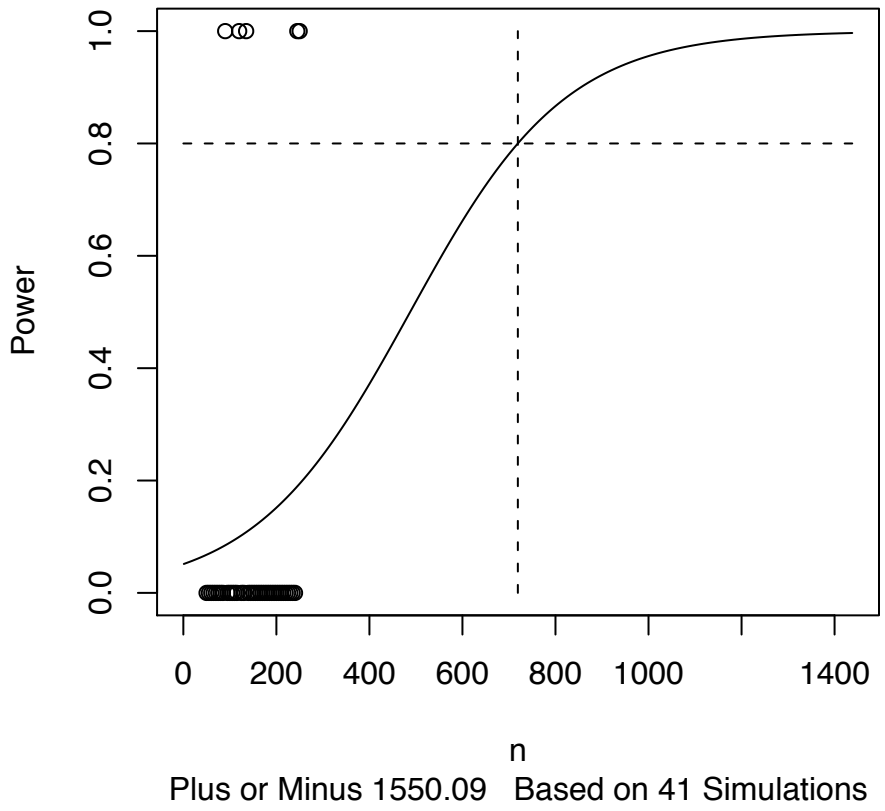


```
> # Tried again: got estimated n of 47.56008, 62.23321
> # Remember, true n needed per treatment is 64
>
> N = c(N,N,N,N) # And then paste in the same code ...

n0-hat
59.03287

> # Tried again and got n0=67. Need a confidence interval for n0-hat.
> # Delta method
> gdot1 <- -1/b1
> gdot2 <- (b0-logodds)/b1^2
> gdot <- cbind(gdot1,gdot2)
> V = vcov(mod)
> Vn0hat <- gdot %*% V %*% t(gdot)
> margin <- round(2*sqrt(Vn0hat),2)
> cat("Estimated n = ",round(n0hat,1)," plus or minus",margin)
Estimated n =  59  plus or minus 6.74

> # Do the whole thing
>
> set.seed(4444); Pi = 0.80 # Desired power
> N = 20:100; N = c(N,N,N,N)
> M = length(N); sig=numeric(M)
> for(i in 1:M) sig[i] = onesim(N[i])
> mod = glm(sig~N,family=binomial)
> b0 <- coefficients(mod)[1]; b1 = coefficients(mod)[2]
> wantpow = 0.80; logodds = log( Pi/(1- Pi))
> n0hat = (logodds-b0)/b1
> names(n0hat) = "n0-hat"
> # Get 55.5 the first time with N = 20:100
> # Delta method
> gdot1 <- -1/b1
> gdot2 <- (b0-logodds)/b1^2
> gdot <- cbind(gdot1,gdot2)
> Vn0hat <- gdot %*% vcov(mod) %*% t(gdot)
> margin <- round(2*sqrt(Vn0hat),2)
> cat("Estimated n = ",round(n0hat,1)," plus or minus",margin)
Estimated n =  60  plus or minus 6.49
> # Plot
> top <- max(round(2*n0hat),max(N))
> plotxx <- 1:top
> exb <- exp(b0+plotxx*b1)
> phat <- exb/(1+exb)
> plot(plotxx,phat,type='l',xlab='n',ylab='Power')
> # Crosshairs
> lines(c(n0hat,n0hat),c(0,1),lty=2)
> lines(c(0,top),c(Pi,Pi),lty=2)
> kaption1 <- paste("For Desired Power of",round(Pi,3),
+ ", Estimated n = ",round(n0hat,1))
> kaption2 <- paste("Plus or Minus",margin,
+ "  Based on",M,"Simulations")
> title(main=kaption1,sub=kaption2)
```



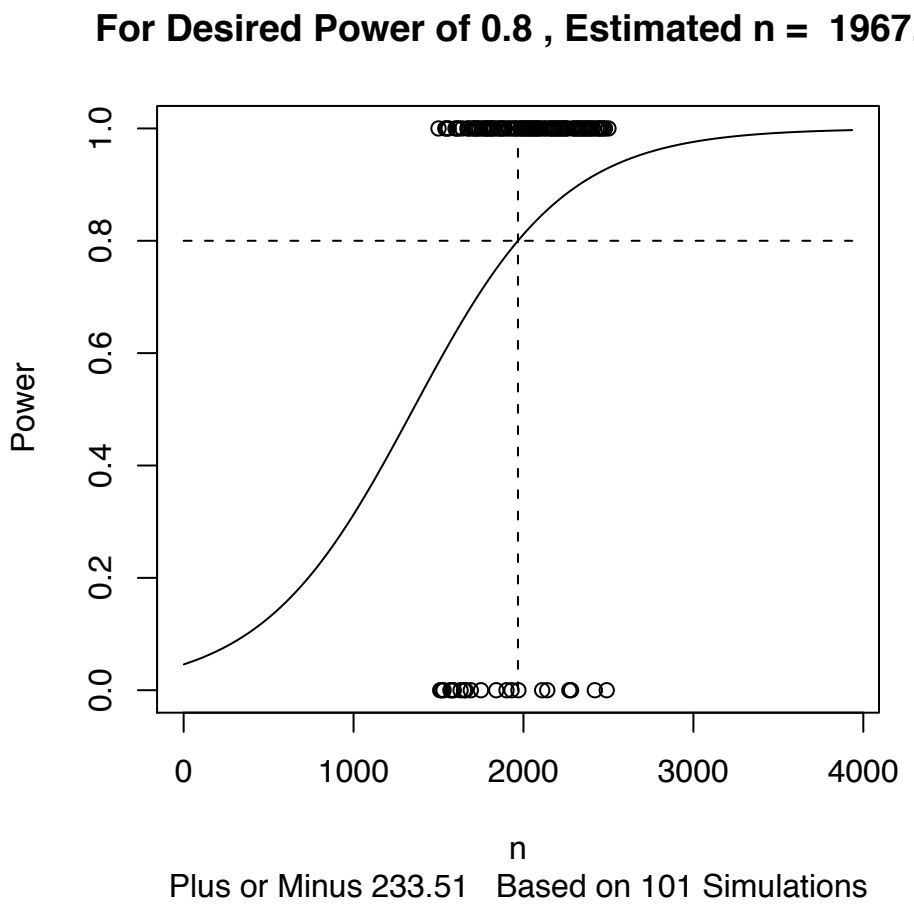

```
> # Try the deadly logistic regression
>
> rm(list=ls())
> critval <- qchisq(.95,1)
> # Calculate square root matrix A. A%*%Z will have var-cov Sigma
> sigma <- rbind( c(50,sqrt(2000)/2),
+                 c(sqrt(2000)/2,40) )
> spec <- eigen(sigma)
> A <- spec$vectors %*% diag(sqrt(spec$values))
> beta <- c(-11.09035489,0.11167961,0.02694983)
> # Another function onesim for this example
```

```
> onesim <- function(nn)
+     {
+       # Simulate X
+       Z <- rbind(rnorm(nn),rnorm(nn))
+       X <- cbind(matrix(1,nn),t(A%*%Z))
+       X[,2] <- X[,2]+70 ; X[,3] <- X[,3]+80
+       # Okay that's X. Now simulate Y
+       xb <- X %*% beta
+       pi <- exp(xb)/(1+exp(xb))
+       Y <- rbinom(nn,1,pi)
+       fullmod <- glm(Y ~ X[,2:3], family=binomial ) # Full model
+       redmod <- glm(Y ~ X[,2], family=binomial ) # Reduced model
+       onesim <- as.numeric(anova(redmod,fullmod)[2,4]>critval)
+       names(onesim) <- NULL
+       onesim
+     } # End function onesim

> # First pretend I don't know that the required sample size is huge.
            > set.seed(4444); Pi = 0.80 # Desired power
> N = seq(from=50,to=250,by=5)
> M = length(N); sig=numeric(M)
> for(i in 1:M) sig[i] = onesim(N[i])
> mod = glm(sig~N,family=binomial)
> b0 <- coefficients(mod)[1]; b1 = coefficients(mod)[2]
> wantpow = 0.80; logodds = log( Pi/(1- Pi))
> n0hat = (logodds-b0)/b1
> names(n0hat) = "n0-hat"
> # Delta method
> gdot1 <- -1/b1
> gdot2 <- (b0-logodds)/b1^2
> gdot <- cbind(gdot1,gdot2)
> Vn0hat <- gdot %*% vcov(mod) %*% t(gdot)
> margin <- round(2*sqrt(Vn0hat),2)
> cat("Estimated n = ",round(n0hat,1)," plus or minus",margin,"\n")
Estimated n =  719.1  plus or minus 1550.09
> cat("From ",round(n0hat-margin,1)," to ",round(n0hat+margin,1),"\n")
From  -831  to  2269.2
>
> # Plot (Include the points, too)
> top <- max(round(2*n0hat),max(N))
> plotxx <- 1:top
> exb <- exp(b0+plotxx*b1)
> phat <- exb/(1+exb)
> invisx = c(1,top); invisy = c(0,1)
> plot(invisx,invisy,type='n',xlab='n',ylab='Power')
> lines(plotxx,phat,type='l')
> points(N,sig)
> # Crosshairs
> lines(c(n0hat,n0hat),c(0,1),lty=2)
> lines(c(0,top),c(Pi,Pi),lty=2)
> kaption1 <- paste("For Desired Power of",round(Pi,3),
+ ", Estimated n = ",round(n0hat,1))
> kaption2 <- paste("Plus or Minus",margin,
+ "  Based on",M,"Simulations")
> title(main=kaption1,sub=kaption2)
```

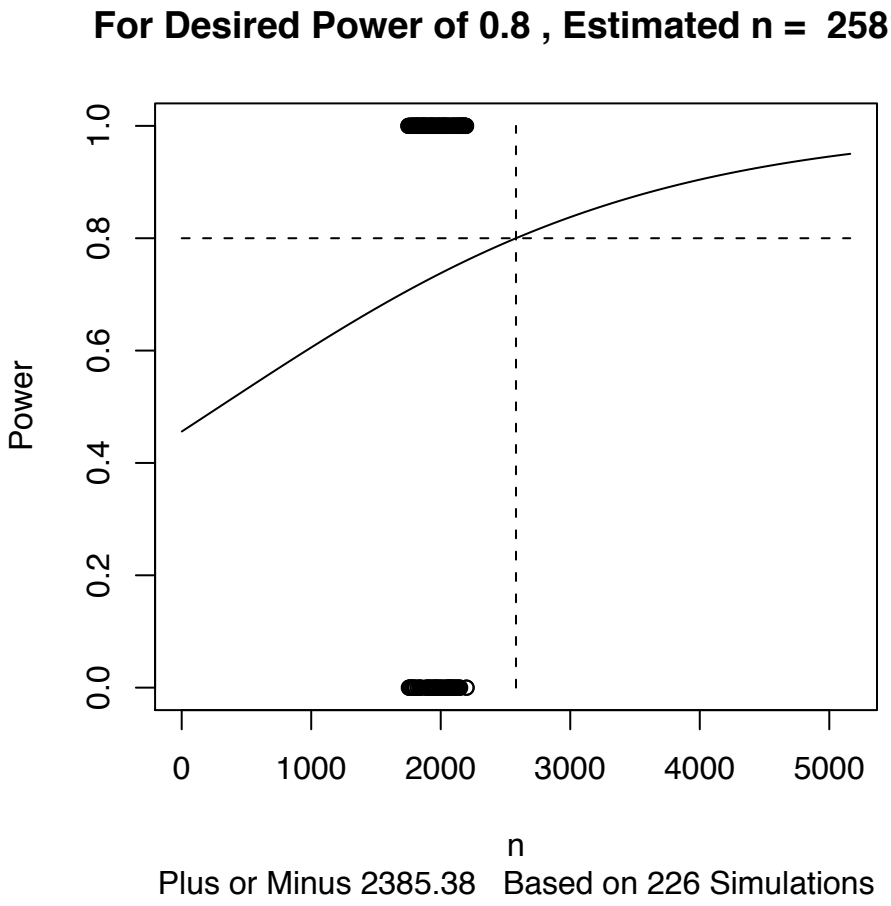



```
N = seq(from=300,to=3000,by=50)
Estimated n =  2014.3  plus or minus 531.57

N = seq(from=1500,to=2500,by=25)
Estimated n =  1755.9  plus or minus 475.71
From  1280.2  to  2231.6
```

```
N = seq(from=1500,to=2500,by=10) # M=101
Estimated n =  1967.5  plus or minus 233.51
From  1734  to  2201
```



```
N = seq(from=1750,to=2200,by=5) ; length(N) # 91
N = seq(from=1750,to=2200,by=2) ; length(N) # 226
Estimated n =  2581.2  plus or minus 2385.38
From  195.8  to  4966.6
```



Try *adding* points in an interval around the current estimate.