##################################################################

# gbinom is a function for graphing the binomial distribution. #

# two parameters are necessary: the number n of trials and the #

# probability p of succes. Other options are optional. #

# http://www.stat.wisc.edu/~larget/R/prob.R #

##################################################################

gbinom = function(n, p, low=0, high=n,scale = F, a=NA,b=NA,calcProb=!all(is.na(c(a,b))),quantile=NA,calcQuant=!is.na(quantile))

{

 sd = sqrt(n \* p \* (1 - p))

 if(scale && (n > 10)) {

 low = max(0, round(n \* p - 4 \* sd))

 high = min(n, round(n \* p + 4 \* sd))

 }

 values = low:high

 probs = dbinom(values, n, p)

 plot(c(low,high), c(0,max(probs)), type = "n", xlab = "Possible Values",

 ylab = "Probability",

 main = paste("Binomial Distribution \n", "n =", n, ", p =", p))

 lines(values, probs, type = "h", col = 2)

 abline(h=0,col=3)

 if(calcProb) {

 if(is.na(a))

 a = 0

 if(is.na(b))

 b = n

 if(a > b) {

 d = a

 a = b

 b = d

 }

 a = round(a)

 b = round(b)

 prob = pbinom(b,n,p) - pbinom(a-1,n,p)

 title(paste("P(",a," <= Y <= ",b,") = ",round(prob,6),sep=""),line=0,col.main=4)

 u = seq(max(c(a,low)),min(c(b,high)),by=1)

 v = dbinom(u,n,p)

 lines(u,v,type="h",col=4)

 }

 else if(calcQuant==T) {

 if(quantile < 0 || quantile > 1)

 stop("quantile must be between 0 and 1")

 x = qbinom(quantile,n,p)

 title(paste("The ",quantile," quantile = ",x,sep=""),line=0,col.main=4)

 u = 0:x

 v = dbinom(u,n,p)

 lines(u,v,type="h",col=4)

 }

 return(invisible())

}

#######################################################################

# gpois is a function for graphing the Poisson distribution. #

# One parameter is necessary: the mean mu. Other options are optional.#

#######################################################################

gpois = function(mu, a=NA,b=NA,calcProb=(!is.na(a) | !is.na(b)),quantile=NA,calcQuant=!is.na(quantile))

{

 sd = sqrt(mu)

 low = max(0, round(mu - 3 \* sd))

 high = round(mu + 5 \* sd)

 values = low:high

 probs = dpois(values, mu)

 plot(c(low,high), c(0,max(probs)), type = "n", xlab = "observable values",

 ylab = "Probability",

 main =substitute(paste("Poisson Distribution with ",mu==m), list(m=mu)))

 lines(values, probs, type = "h", col = 2)

 abline(h=0,col=3)

 if(calcProb) {

 if(is.na(a)){ a = 0 }

 if(is.na(b)){

 a = round(a)

 prob = 1-ppois(a-1,mu)

 title(paste("P(",a," <= Y ) = ",round(prob,6),sep=""),line=0,col.main=4)

 u = seq(max(c(a,low)),high,by=1)

 }

 else {

 if(a > b) {d = a; a = b; b = d;}

 a = round(a); b = round(b)

 prob = ppois(b,mu) - ppois(a-1,mu)

 title(paste("P(",a," <= Y <= ",b,") = ",round(prob,6),sep=""),line=0,col.main=4)

 u = seq(max(c(a,low)),min(c(b,high)),by=1)

 }

 v = dpois(u,mu)

 lines(u,v,type="h",col=4)

 }

 else if(calcQuant==T) {

 if(quantile < 0 || quantile > 1)

 stop("quantile must be between 0 and 1")

 x = qpois(quantile,mu)

 title(paste("The ",quantile," quantile = ",x,sep=""),line=0,col.main=4)

 u = 0:x

 v = dpois(u,mu)

 lines(u,v,type="h",col=4)

 }

 return(invisible())

}

#####################################################################

# gnorm plots normal curves and computes and displays probabilities #

# two parameters are necessary: the mean mu and standard #

# deviation sigma. Others options are available but optional only. #

#####################################################################

gnorm = function(mu, sigma,a=NA,b=NA,calcProb=!all(is.na(c(a,b))),quantile=NA,calcQuant=!is.na(quantile))

{

 values = seq(-1,1,.005) \* 4 \* sigma + mu

 probs = dnorm(values, mu, sigma)

 plot(values, probs, axes = F, type = "n", xlab = "Possible Values",

 ylab = "Probability Density",

 main = substitute(paste("Normal Distribution with ",mu == m,", ",sigma == s),list(m=mu,s=sigma)))

 axis(1, pos = 0)

 abline(0,0,col=1)

 lines(values, probs, col = 2)

 lo = mu - 4 \* sigma

 hi = mu + 4 \* sigma

 h = dnorm(mu,mu,sigma)

 cex=0.8

 if(calcProb==T)

 {

 if(!is.na(a) && !is.na(b) && a > b){

 d = a; a = b; b = d

 }

 if(is.na(a) || a <= lo){ ulo = lo }

 else if(a <= hi){ ulo = a }

 else { ulo = hi }

 if(is.na(b) || b >= hi){ uhi = hi }

 else if(b >= lo){ uhi = b }

 else { uhi = lo }

 u = seq(ulo,uhi,length=601)

 lines(u,dnorm(u,mu,sigma),type="h",col=2)

 if(!is.na(a) && !is.na(b)){

 text(mu - 3.9 \* sigma, 0.8 \* h,

 paste("P( ",a," < X < ",b," ) = ",

 round(pnorm(b,mu,sigma)-pnorm(a,mu,sigma),digits=4),sep=""),

 adj=0,col=4,cex=cex)

 text(mu - 3.9 \* sigma, 0.6 \* h,

 paste("P( X < ",a," ) = ",

 round(pnorm(a,mu,sigma),digits=4),sep=""),adj=0,col=4,cex=cex)

 text(mu + 3.9 \* sigma, 0.5 \* h,

 paste("P( X > ",b," ) = ",

 round(1-pnorm(b,mu,sigma),digits=4),sep=""),adj=1,col=4,cex=cex)

 }

 else if(!is.na(a) && is.na(b)){

 text(mu - 3.9 \* sigma, 0.6 \* h,

 paste("P( X < ",a," ) = ",

 round(pnorm(a,mu,sigma),digits=4),sep=""),adj=0,col=4,cex=cex)

 text(mu + 3.9 \* sigma, 0.5 \* h,

 paste("P( X > ",a," ) = ",

 round(1-pnorm(a,mu,sigma),digits=4),sep=""),adj=1,col=4,cex=cex)

 }

 else if(is.na(a) && !is.na(b)){

 text(mu - 3.9 \* sigma, 0.6 \* h,

 paste("P( X < ",b," ) = ",

 round(pnorm(b,mu,sigma),digits=4),sep=""),adj=0,col=4,cex=cex)

 text(mu + 3.9 \* sigma, 0.5 \* h,

 paste("P( X > ",b," ) = ",

 round(1-pnorm(b,mu,sigma),digits=4),sep=""),adj=1,col=4,cex=cex)

 }

 }

 else if(calcQuant==T)

 {

 zoffset = -0.02

 if( quantile <= 0 || quantile >= 1) quantile = 0.5

 x = qnorm(quantile,mu,sigma)

 if( x > lo && x < hi)

 {

 u = seq(lo,x,length=601)

 lines(u,dnorm(u,mu,sigma),type="h",col=2)

 text(x, zoffset \* h,

 paste("z = ",round(qnorm(quantile),2),sep=""),adj=0.5,col=4,cex=cex)

 }

 else if(x >= hi)

 {

 u = seq(lo,hi,length=601)

 lines(u,dnorm(u,mu,sigma),type="h",col=2)

 text(hi, zoffset \* h,

 paste("z = ",round(qnorm(quantile),2),sep=""),adj=0.5,col=4,cex=cex)

 }

 else if( x <= lo)

 {

 text(lo, zoffset \* h,

 paste("z = ",round(qnorm(quantile),2),sep=""),adj=0.5,col=4,cex=cex)

 }

 text(mu - 3.9 \* sigma, 0.5 \* h,

 paste("P( X < ",signif(x,4)," ) = ",

 round(quantile,digits=4),sep=""),adj=0,col=4,cex=cex)

 }

 return(invisible())

}

######################## next functions for fun ###################

#

# graph a skewed sampling distribution

gskew = function(n = 1, mu = 0, sd = 1, skew = 0)

{

 se = sd/sqrt(n)

 x = seq(mu - 4 \* se, mu + 4 \* se, length = 501)

 y = dskew(x, n, mu, sd, skew)

 plot(x, y, xlab = "", ylab = "density",

 main = paste("Distribution of x-bar, sample size =",n), type = "n")

 lines(x, y, type = "l", col = 4)

 abline(h = 0)

 lines(x, dnorm(x, mu, se), col = 5)

 return(invisible(y))

}

# density of skewed distribution

dskew = function(x, n = 1, mu = 0, sd = 1, skew = 0)

{

 se = sd/sqrt(n)

 if(skew == 0)

 y = dnorm(x, mu, se)

 else {

 a = (skew \* sd)/2

 b = mu - (2 \* sd)/skew

 alpha = 4/skew/skew

 if(skew > 0)

 y = (n \* dgamma((n \* (x - b))/a, n \* alpha))/a

 else y = ( - n \* dgamma((n \* (x - b))/a, n \* alpha))/a

 }

 return(y)

}

# density of symmetric bimodal distribution

dbimod = function(x,n=1,mu=0,sd=1,d=1)

{

 if(d >= sd)

 stop("sd must be larger than d")

 temp = matrix(0,nrow=n+1,ncol=length(x))

 for(i in 0:n){temp[i+1,] = dbinom(i,n,.5)\*dnorm(x,mu-d+2\*i\*d/n,sqrt((sd\*sd-d\*d)/n))}

 return(apply(temp,2,sum))

}

# graph sampling distribution of symmetric bimodal distribution

gbimod = function(n=1, mu=0, sd=1, d=0.5)

{

 if(d >= sd)

 stop("sd must be larger than d")

 se = sd/sqrt(n)

 x = seq(mu - 4 \* se, mu + 4 \* se, length = 201)

 y = dbimod(x, n, mu, sd, d)

 xx = c(mu - 4 \* se, mu + 4 \* se)

 yy = c(0, max(y, dnorm(mu, mu, se)))

 plot(xx, yy, xlab = "", ylab = "density",

 main = paste("Distribution of x-bar, sample size =",n), type = "n")

 lines(x, y, type = "l", col = 4)

 abline(h = 0)

 lines(x, dnorm(x, mu, se), col = 5)

 return(invisible(y))

}