ethans.little.function <- function(G=50000000, gamma= 0, sigma2.theta=1, sigma2.epsilon=1, sigma2.xstar=1, sigma2.rw=0) {

  # first level simulation
  thetaL <- rnorm(G, mean=0, sd=sqrt(sigma2.theta))
  thetaR <- rnorm(G, mean=0, sd=sqrt(sigma2.theta))
  epsilonL <- rnorm(G, mean=0, sd=sqrt(sigma2.epsilon))
  epsilonR <- rnorm(G, mean=0, sd=sqrt(sigma2.epsilon))
  etaL <- rnorm(G, mean=gamma, sd=sqrt(sigma2.xstar))
  lambdaL1 <- rep((sigma2.theta / (sigma2.theta + sigma2.epsilon), G)
  SL1 <- thetaL + epsilonL
  SR1 <- thetaR + epsilonR
  mL1 <- lambdaL1 * SL1
  mR1 <- lambdaL1 * SR1
  
  reelectL <- sum(mL1 - mR1 >= etaL) / (sum(mL1 - mR1 >= etaL) + sum(mL1 - mR1 < etaL))

  # yank off the first case but have no selection
  thetaL2 <- thetaL
  thetaR2 <- rnorm(length(thetaL2), mean=0, sd=sqrt(sigma2.theta))
  epsilonL2 <- rnorm(length(thetaL2), mean=0, sd=sqrt(sigma2.epsilon))
  epsilonR2 <- rnorm(length(thetaL2), mean=0, sd=sqrt(sigma2.epsilon))
  eta2 <- rnorm(length(thetaL2), mean=gamma, sd=sqrt(sigma2.xstar))
  lambdaL2 <- lambdaL1
  lambdaL2 <- (lambdaL2 * sigma2.epsilon + sigma2.rw) / (lambdaL2 * sigma2.theta + sigma2.rw + sigma2.epsilon)
  lambdaR2 <- rep((sigma2.theta / (sigma2.theta + sigma2.epsilon), length(thetaL2))
  SL2 <- thetaL2 + epsilonL2
  SR2 <- thetaR2 + epsilonR2
  mL2 <- lambdaL2 * SL2 + (1 - lambdaL2) * mL1
  mR2 <- lambdaR2 * SR2

  #Probability of left wing guy achieving reelection, conditioning on incumbency
  reelect2 <- sum(mL2 - mR2 >= eta2) / (sum(mL2 - mR2 >= eta2) + sum(mL2 - mR2 < eta2))

  #Now calculate the increased probability of winning given incumbency
  iaL <- reelect2 - reelectL

  #now I need to define the normal vote

  sigma <- sqrt(((2*(sigma2.theta^2))/(sigma2.theta + sigma2.epsilon)) + sigma2.xstar)
  nv <- 1 - pnorm((gamma / sigma))

  cat("iaL ", iaL, ", nv", nv, ", "n", ", reelect1", reelectL, ", "n", ", reelect2", reelectL, ", "n")
  return(c(iaL, nv, reelect1, reelect2))
}

ruler<- seq(-4,4,.25)
storage.matrix <- matrix(NA, length(ruler), 4)
count <- 1
for(i in ruler) {
  storage.matrix[count,1:4] <- ethans.little.function(G=50000, gamma=1)
  count <- count + 1
}
count <- count + 1

# create the final incumbency advantage matrix
storage.matrix2 <- matrix(NA, length(seq(ruler)), 4)

# now put back in the original stuff
storage.matrix2[,1:2] <- storage.matrix[,1:2]

# now add the right wing incumbency advantage
for(j in seq(1,length(ruler)))(storage.matrix2[j,3]<-storage.matrix2[34-j,1])

# now add the average per district incumbency advantage
for(k in seq(1,length(ruler)))(storage.matrix2[k,4]<-
        storage.matrix2[k,2] * storage.matrix2[k,1] + (1-storage.matrix2[k,2]) *
        storage.matrix2[k,3])

# Now make a matrix with the left-wing component parts
storage.matrix3<-matrix(NA, length(seq(ruler)), 5)

# 1. Add in the Probability of winning an open seat
storage.matrix3[,1]<-storage.matrix[,3]

# 2. Add in the probability of winning an incumbent seat
storage.matrix3[,2]<- storage.matrix[,4]

# 3. Now subtract these two to get the additional probability of winning given that you are an incumbent
storage.matrix3[,3]<-storage.matrix3[,2]-storage.matrix3[,1]

# 4. Add in 1- the normal vote
storage.matrix3[,4]<- 1-storage.matrix[,2]

# 5. Calculate the left wing contribution to the IA

# Now plot the unconditional probability of winning the open seat and the incumbent seat as a function of the NV

postscript(file = "c:/latex/incumbency/figuresR/unconditioned_pr_win.eps", horizontal = FALSE, paper = "letter")
par(cex=2)
plot(storage.matrix3[,4],storage.matrix3[,1],
type="l",xlab="right-wing normal vote", ylab="probability left-wing candidate wins",
        xlim=c(0,1.0), main=expression(paste((sigma^2) [theta] == 1, ", ",
        [sigma^2] [epsilon] == 1, ", ", [sigma^2] [eta] == 1)))
points(storage.matrix3[,4],storage.matrix3[,1], pch =19)
lines(storage.matrix3[,4],storage.matrix3[,2],type="1")
points(storage.matrix3[,4],storage.matrix3[,2], pch=23)
legend(.35, 1.0, legend=c("open seat", "left-wing incumbent"),pch=c(19,23),cex=.85)
dev.off()
ylab="left-wing incumbency advantage without selection",
    xlim=c(0,1.0), ylim=c(-0.05,0.15), main=expression(paste({sigma}^2 [theta] == 1, ", ", 
    {sigma}^2 [epsilon] == 1, ", ", (sigma^2) [eta] == 1)))
dev.off()