ethans.little.function <- function(G=50000, gamma=0, kappa=0.001, sigma2.theta=10, 
sigma2.epsilon = 1, sigma2.rw=0, i) {
    sigma2.xstar=1
    sigma2.xstar1=1
    sigma2.xstar2=1
    sigma2.xstar3=1
    sigma2.xstar4=1
    sigma2.xstar5=1
    sigma2.xstar6=1
    
    #Define the first value of sigma2.epsilon21
    sigma2.epsilon21=1
    
    # 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
    
    reelect1 <- sum(mL1 - mR1 >= etal) / (sum(mL1 - mR1 >= etal) + sum(mL1 - mR1 < etal))
    
    # yank off the first case
    thetaL12 <- thetaL[mL1 - mR1 >= etal]
    thetaR12 <- rnorm(length(thetaL12), mean=0, sd=sqrt(sigma2.theta))
    epsilonL12 <- rnorm(length(thetaL12), mean=0, sd=sqrt(sigma2.epsilon21))
    epsilonR12 <- rnorm(length(thetaL12), mean=0, sd=sqrt(sigma2.epsilon21))
    etal2 <- rnorm(length(thetaL12), mean=gamma, sd=sqrt(sigma2.xstar))
    lambdaL12 <- lambdaL1[mL1 - mR1 >= etal]
    lambdaL12 <- (lambdaL12 * sigma2.epsilon + sigma2.rw) / (lambdaL12 * sigma2.epsilon + sigma2.rw + sigma2.epsilon21)
    lambdaR12 <- rep(sigma2.theta / (sigma2.theta + sigma2.epsilon21), length(thetaL12))
    sigma <- rep(sqrt(sigma2.epsilon21*(((sigma2.theta / (sigma2.theta + sigma2.epsilon)) * sigma2.epsilon) / ((sigma2.theta / (sigma2.theta + sigma2.epsilon)) * sigma2.theta) + sigma2.xstar + (sigma2.theta / (sigma2.theta + sigma2.epsilon)) * sigma2.theta), length(thetaL12))
    #sigma <- (lambdaL2)^2*sigma2.epsilon + sigma2.xstar + (sigma2.theta / (sigma2.theta + sigma2.epsilon)) * sigma2.theta
    SL12 <- thetaL12 + epsilonL12
    SR12 <- thetaR12 + epsilonR12
    mL12 <- lambdaL12 * SL12 + (1 - lambdaL12) * mL1[mL1 - mR1 >= etal]  
    
    #Assign mR12 to be the real value if there is a challenge and a value that loses for sure if there is not a real challenge
    mR12 <- ifelse(1-pnorm(mL1[mL1 - mR1 >= etal]/sigma) >= kappa, lambdaR12 * SR12, mL12 - etal2 - 1)
    
    #Probability of left wing guy achieving reelection, conditioning on incumbency
    reelect2 <- sum(mL12 - mR12 >= etal2) / sum(mL1 - mR1 >= etal)
    
    #Now calculate the increased probability of winning given incumbency
    iaL <- reelect2 - reelect1
Now do this all again for the next value of sigma2.epsilon2

    # Define the next value of sigma2.epsilon2
    sigma2.epsilon2=1.5
    # first level simulation
    thetaL2 <- rnorm(G, mean=0, sd=sqrt(sigma2.theta))
    thetaR2 <- rnorm(G, mean=0, sd=sqrt(sigma2.theta))
    epsilonL2 <- rnorm(G, mean=0, sd=sqrt(sigma2.epsilon2))
    epsilonR2 <- rnorm(G, mean=0, sd=sqrt(sigma2.epsilon2))
    etaL2 <- rnorm(G, mean=gamma, sd=sqrt(sigma2.xstar2))
    lambdaL2 <- rep(sigma2.theta / (sigma2.theta + sigma2.epsilon2), G)
    SL2 <- thetaL2 + epsilonL2
    SR2 <- thetaR2 + epsilonR2
    mL2 <- lambdaL2 * SL2
    mR2 <- lambdaL2 * SR2

    reelectL2 <- sum(mL2 - mR2 > etaL2) / (sum(mL2 - mR2 > etaL2) + sum(mL2 - mR2 < etaL2))
    # yank off the first case
    thetaL2 <- thetaL2[mL2 - mR2 > etaL2]
    thetaR2 <- rnorm(length(thetaL2), mean=0, sd=sqrt(sigma2.theta))
    epsilonL2 <- rnorm(length(thetaL2), mean=0, sd=sqrt(sigma2.epsilon2))
    epsilonR2 <- rnorm(length(thetaL2), mean=0, sd=sqrt(sigma2.epsilon2))
    etaL2 <- rnorm(length(thetaL2), mean=gamma, sd=sqrt(sigma2.xstar2))
    lambdaL2 <- lambdaL2[mL2 - mR2 > etaL2]
    lambdaL2 <- (lambdaL2 * sigma2.epsilon2 + sigma2.rw) / (lambdaL2 * sigma2.epsilon2 + sigma2.rw + sigma2.epsilon2)
    lambdaR2 <- rep(sigma2.theta / (sigma2.theta + sigma2.epsilon2), length(thetaL2))
    sigma2 <- rep(sigma2.epsilon2 * (((sigma2.theta / (sigma2.theta + sigma2.epsilon2)) * sigma2.epsilon2) + sigma2.epsilon2)^2 + sigma2.xstar2 + (sigma2.theta / (sigma2.theta + sigma2.epsilon2)) * sigma2.theta, length(thetaL2))
    # sigma2 <- (lambdaL2)^2 * sigma2.epsilon2 + sigma2.xstar2 + (sigma2.theta / (sigma2.theta + sigma2.epsilon2)) * sigma2.theta)
    SL2 <- thetaL2 + epsilonL2
    SR2 <- thetaR2 + epsilonR2
    mL2 <- lambdaL2 * SL2 + (1 - lambdaL2) * mL2[mL2 - mR2 > etaL2]
    # Assign mR22 to be the real value if there is a challenge and a value that loses for sure if there
    # is not a real challenge
    mR22 <- ifelse(1 - pnorm(mL2[mL2 - mR2 > etaL2] / sigma2) > kappa, lambdaR2 * SR2, mL2 - etaL2 - 1)
    # Probability of left wing guy achieving reelect, conditioning on incumbency
    reelect22 <- sum(mL22 - mR22 > eta22) / sum(mL22 - mR22 > eta22)

    # Now calculate the increased probability of winning given incumbency
    iaL2 <- reelect22 - reelectL2

    # Now I need to define the high xstar normal vote
    #
    sigma.nv2 <- sqrt(sigma2.theta + sigma2.epsilon + sigma2.xstar2)
    #
    nv2 <- 1 - pnorm((gamma / sigma.nv2))

    ###############################################################################
    # Do this all again for the next value of sigma2.epsilon2
    # Define the next value of sigma2.epsilon2
    sigma2.epsilon23=2
# first level simulation
thetaL3 <- rnorm(G, mean=0, sd=sqrt(sigma2.theta))
thetaR3 <- rnorm(length(thetaL23), mean=0, sd=sqrt(sigma2.theta))
epsilonL3 <- rnorm(length(thetaL23), mean=0, sd=sqrt(sigma2.epsilon2))
epsilonR3 <- rnorm(length(thetaL23), mean=0, sd=sqrt(sigma2.epsilon2))
eta3 <- rnorm(length(thetaL23), mean=gamma, sd=sqrt(sigma2.xstar3))
lambdaL3 <- rep(sigma2.theta / (sigma2.theta + sigma2.epsilon2), G)
SL13 <- thetaL3 + epsilonL3
SR13 <- thetaR3 + epsilonR3
mL13 <- lambdaL3 * SL13
mR13 <- lambdaL3 * SR13
reeelect13 <- sum(mL13 - mR13 >= eta13) / (eta13 + sum(mL13 - mR13 < eta13))

# yank off the first case
thetaL23 <- thetaL3[mL13 - mR13 >= eta13]
thetaR23 <- rnorm(length(thetaL23), mean=0, sd=sqrt(sigma2.theta))
epsilonL23 <- rnorm(length(thetaL23), mean=0, sd=sqrt(sigma2.epsilon2))
epsilonR23 <- rnorm(length(thetaL23), mean=0, sd=sqrt(sigma2.epsilon2))
eta23 <- rnorm(length(thetaL23), mean=gamma, sd=sqrt(sigma2.xstar3))
lambdaL23 <- lambdaL3[mL13 - mR13 >= eta13]
lambdaL23 <- (lambdaL23 * sigma2.epsilon2 + sigma2.rw) / (lambdaL23 + sigma2.epsilon2 + sigma2.rw + sigma2.epsilon2)
lambdaR23 <- rep(sigma2.theta / (sigma2.theta + sigma2.epsilon2), length(thetaL23))
sigma3 <- rep(sqrt(sigma2.epsilon2 * (sigma2.theta + sigma2.epsilon2)) + sigma2.xstar3 + sigma2.thetax23 * (sigma2.thetax23 + sigma2.epsilon2), length(thetaL23))

# Assign mR23 to be the real value if there is a challenge and a value that loses for sure if there is not a real challenge
mR23 <- ifelse(1 - pnorm(mL13[mL13 - mR13 >= eta13] / sigma3) == kappa, lambdaR23 * SR23, mL23 - eta23 - 1)

# Probability of left wing guy achieving reelection, conditioning on incumbency
reeelect23 <- sum(mL23 - mR23 >= eta23) / (eta23 + sum(mL23 - mR23 < eta23))

# Now calculate the increased probability of winning given incumbency
iaL3 <- reelect23 - reelect13

# Now do this all again for the next value of sigma2.epsilon2
sigma2.epsilon24 <- 2.5

# first level simulation
thetaL4 <- rnorm(G, mean=0, sd=sqrt(sigma2.theta))
thetaR4 <- rnorm(G, mean=0, sd=sqrt(sigma2.theta))
epsilonL4 <- rnorm(G, mean=0, sd=sqrt(sigma2.epsilon2))
epsilonR4 <- rnorm(G, mean=0, sd=sqrt(sigma2.epsilon2))
eta4 <- rnorm(G, mean=gamma, sd=sqrt(sigma2.xstar4))
lambdaL4 <- rep(sigma2.theta / (sigma2.theta + sigma2.epsilon2), G)
SL14 <- thetaL4 + epsilonL4
SR14 <- thetaR4 + epsilonR4
mL14 <- lambda14 * SL14
mR14 <- lambda14 * SR14

reelect14 <- sum(mL14 - mR14 >= eta14) / (sum(mL14 - mR14 >= eta14) + sum(mL14 - mR14 < eta14))

# yank off the first case
theta14 <- theta14[1][L14 - M14 >= eta14]
thetaR244 <- rnorm(length(thetaL244), mean=0, sd=sqrt(sigma2.theta))
epsilonL244 <- rnorm(length(thetaL244), mean=0, sd=sqrt(sigma2.epsilon))
epsilonR244 <- rnorm(length(thetaL244), mean=0, sd=sqrt(sigma2.epsilon))
eta24 <- rnorm(length(thetaL244), mean=gamma, sd=sqrt(sigma2.xstar4))
lambdaL1244 <- lambda144[ML14 - MR14 >= eta14]
lambdaL244 <- (lambdaL244 * sigma2.epsilon + sigma2.Rw) / (lambdaL1244 * sigma2.epsilon + sigma2.Rw + sigma2.epsilon)
lambdaR244 <- rep(sigma2.theta / (sigma2.theta + sigma2.epsilon), length(thetaL244))
sigma4 <- rep(sigma2.epsilon/((sigma2.theta / (sigma2.theta + sigma2.epsilon)^2) + sigma2.xstar4 + (sigma2.theta / (sigma2.theta + sigma2.epsilon)) * sigma2.xstar2, length(thetaL244))

# Assign m24 to be the real value if there is a challenge and a value that loses for sure if there is not a real challenge
m24 <- ifelse(1 - pnorm(mL14 - mR14 == eta14) / sigma4) == kappa, lambdaR244 * SR244, mL24 - eta24 - 1)

# Probability of left wing guy achieving reelection, conditioning on incumbency
reelect24 <- sum(mL14 - mR14 >= eta24) / sum(mL14 - mR14 >= eta14)

# Now calculate the increased probability of winning given incumbency
iaL4 <- reelect24 - reelect14

# Now do this all again for the next value of sigma2.epsilon2

# Define the next value of sigma2.epsilon2
sigma2.epsilon25 = 3

# First level simulation
thetaL5 <- rnorm(G, mean=0, sd=sqrt(sigma2.theta))
thetaR5 <- rnorm(G, mean=0, sd=sqrt(sigma2.theta))
epsilonL5 <- rnorm(G, mean=0, sd=sqrt(sigma2.epsilon))
epsilonR5 <- rnorm(G, mean=0, sd=sqrt(sigma2.epsilon))
eta15 <- rnorm(G, mean=gamma, sd=sqrt(sigma2.xstar5))
lambdaL15 <- rep(sigma2.theta / (sigma2.theta + sigma2.epsilon2, G))
SL15 <- thetaL5 + epsilonL5
SR15 <- thetaR5 + epsilonR5
mL15 <- lambda15 * SL15
mR15 <- lambda15 * SR15

reelect15 <- sum(mL15 - mR15 >= eta15) / (sum(mL15 - mR15 >= eta15) + sum(mL15 - mR15 < eta15))

# Yank off the first case
thetaL25 <- thetaL5[ML15 - MR15 >= eta15]
thetaR25 <- rnorm(length(thetaL25), mean=0, sd=sqrt(sigma2.theta))
epsilonL25 <- rnorm(length(thetaL25), mean=0, sd=sqrt(sigma2.epsilon2))
epsilonR25 <- rnorm(length(thetaL25), mean=0, sd=sqrt(sigma2.epsilon2))
eta25 <- rnorm(length(thetaL25), mean=gamma, sd=sqrt(sigma2.xstar5))
lambdaL125 <- lambda15[ML15 - MR15 >= eta15]
lambdaL25 <- (lambdaL25 * sigma2.epsilon2 + sigma2.Rw) / (lambdaL125 * sigma2.epsilon2)
# Define the next value of sigma2.epsilon2
sigma2.epsilon26=50
# first level simulation
thetaL6 <- rnorm(G, mean=0, sd=sqrt(sigma2.theta))
thetaL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.theta))
epsilonL6 <- rnorm(G, mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnorm(length(thetaL26), mean=0, sd=sqrt(sigma2.epsilon))
epsilonL6 <- rnor...
is not a real challenge

```r
mR26 <- ifelse(1 - pnorm(mL16[mL16 - mR16 >= eta16]/sigma6) >= kappa, lambdaR26 * SR26, mL26 - eta26 - 1)
```

#Probability of left wing guy achieving reelection, conditioning on incumbency

```r
reelect26 <- sum(mL26 - mR26 >= eta26) / sum(mL16 - mR16 >= eta16)
```

#Now calculate the increased probability of winning given incumbency

```r
iaL6 <- reelect26 - reelect16
```

```r
cat("iaL ", iaL, "n", "iaL2 ", iaL2, "n", "iaL3 ", iaL3, "n", "iaL4 ", iaL4, "n", "iaL5 ", iaL5, "n", "iaL6 ", iaL6, "n")
return(c(iaL, iaL2, iaL3, iaL4, iaL5, iaL6))
}
```

#create the final incumbency advantage matrix

```r
ruler <- seq(0, .5, .05)
storage.matrix <- matrix(NA, length(ruler), 7)
#storage.matrix3 <- matrix(NA, length(ruler), 1)
count <- 1
for(i in ruler) {
  storage.matrix[count, 1:6] <- ethans.little.function(G=50000, kappa=i)
  count <- count + 1
}
storage.matrix[, 7] <- seq(0, 0.5, 0.05)
```

```r
postscript(file = "c:/latex/incumbency/figuresR/kappa-epsilon/kappa-epsilon1.eps", horizontal = FALSE, paper = "letter")
par(cex=2)
plot(storage.matrix[, 7], storage.matrix[, 1], type="l", ylim=c(0.15, .35), xlab="Recruitment Cost", ylab="Incumbency Advantage",
main=paste("Recruitment Cost [eta] = 10", " ", (sigma^2) [eta] = 10", " ", (epsilon^2) [eta] = 10")
lines(storage.matrix[, 7], storage.matrix[, 6])
points(storage.matrix[, 7], storage.matrix[, 1], pch = 25)
points(storage.matrix[, 7], storage.matrix[, 6], pch = 20)
legend(0, .35, legend = c(expression((sigma^2) [epsilon] = "1"), expression((sigma^2) [epsilon] = "10")), pch = c(25, 20))
dev.off()
```