AguiGomulkiWatts_SupplementaryMaterials_MarkdownFile

Jon Aguinaga

6/28/2021

Makes direct comparison between the accuracy of assessments based on personal information alone to that which combines both personal and social information. In the manuscript, this is the code used to generate **Figure 4**.

```
#The following parameters control the simulation.

#Personal sample size where K is greater than
#or equal to 1
K <- 3

#Social sample size where Ks is greater than
#or equal to one
Ks <- 5

#Social weight controls how much weight is
#placed on the social information relative
#to personal information (1 -w)
w <- 0.5


#Mean environmental state for a normally
#distributed environment
mu <- 0

#Standard deviation of the environmental state
STDV <- 0

#Group Size
N <- 20

#Number of replicates performed per
#value in INCREMENT. For example,
#the model is performed 10,000 times
#for each value in INCREMENT (50).
REPS <- 10000

#Simulation steps = INCREMENT
#the model is replicated (REPS) amount of
#times per each step in INCREMENT
INCREMENT <- 50
```

```r
#--- NORMALLY DISTRIBUTED ENVIRONMENT------#
#Values of personal information are continuous


#INNER LOOP data frame stores data from 10,000 replicates
#of group average personal assessments (PRIV) and
#combined assessments (COMB)
COLLECTIVE <- data.frame(PRIV = rep(0, REPS), COMB = rep(0, REPS))

#OUTER LOOP data frame stores data from 50 Increments of
#grand means across all replicates
#Model parameters are reported for GGPLOT2 plotting purposes
COLLECTIVE_TWO<- data.frame(K = rep(0, INCREMENT), Kds = rep(0, INCREMENT),
                            MU = rep(0, INCREMENT), AVGPRIV = rep(0, INCREMENT),
                            AVGCOMB = rep(0, INCREMENT),
                          THEORC = rep(0, INCREMENT), THEORP = rep(0, INCREMENT),
                          V.Sq = rep(0, INCREMENT))


#We introduce this resample function because, according to R documentation,
#the core function sample() can give 'undesired behaviour' when applied to
#sequences with only 1 element
resample <- function(x, Ks,...) x[sample.int(length(x), size=Ks,...)]


#-----------OUTER LOOP-----------#
for(t in 1:INCREMENT) {

  #Values in this section are incremented from their original values
  #(see above) by a specified amount (see immediately below) until
  #50 steps (INCREMENT) are completed.

  #Mean Environmental State
  mu <- mu + 0.0

  #Standard deviation of environmental state
  STDV <- STDV + 0.01

  #Environmental Variance
  V.SQ <- round(STDV^2, 5)

  #Theoretical expectation of combined assessment (EQN. 8)
  TheorC <- V.SQ*((1-w)^2 + (w^2)/Ks)/K

  #THeoretical expectation of personal assessment (EQN. 7)
  TheorP <- (V.SQ / K)


  #-----------INNER LOOP-------------#
  for( i in 1:REPS){
    #Each replicate (10,000 total) contains information
    #for mean group (N) personal and combined assessments.
```

```r
#Processed Personal information (AVG of K Personal samples)
Private <- c()


#------------PERSONAL ASSESSMENT--------#
#Loop performs personal information sampling for
#each group member in N.
for (j in 1:N){

  #Each group member independently samples from a
  #normally distributed environment with mean state mu
  PrivateSampling <- rnorm(K, mu, STDV)

  #Each individuals personal assessment is the average of
  #its personal samples
  AvgPI <- mean(PrivateSampling)

  #Stores personal samples into the vector specified above
  Private[j] <- AvgPI
}

#----------------SOCIAL ASSESSMENT-----------------#
#Loop performs social information sampling for
#each group member in N.

#Processed social information
Others <- c()

for(j in 1:N){

  #INTEGRATED SOCIAL INFORMATION
  #Social samples obtained from Ks randomly selected social partners
  SocialSampling <- resample(Private[-j], Ks)

  #Social assessment = average of social samples
  AvgSI <- mean(SocialSampling)

  #social assessment stored into vector above
  Others[j] <- AvgSI
}

#any values that = NOT A NUMBER are replaced with 0
Others[is.nan(Others)] <- 0

#COMBINED ASSESSMENT
Combined <- (((1 - w) * Private) + (w * Others))

#Mean squared deviation across all N individuals for
#Personal (PVAR) and combined (CVAR) assessments
PVAR <- mean((Private - mu)^2)
CVAR <- mean((Combined - mu) ^2)
```

```r
    #--------------STORES VALUES INTO DATA FRAME FOR INNER LOOP---------------#
    COLLECTIVE[i, "PRIV"] <- PVAR
    COLLECTIVE[i, "COMB"] <- CVAR
  }



  #Averages across 10,000 replicates for the personal and combined mean squared
  #Deviations
  AVGPVAR <- mean(COLLECTIVE$PRIV)
  AVGCVAR <- mean(COLLECTIVE$COMB)


  #For plotting we record the value of K, Ks, mu and also store
  #personal and combined mean squared deviations and the expected values of
  #personal and combined assessment accuracies (EQN 7 & 8)
  COLLECTIVE_TWO[t, "K"] <- K
  COLLECTIVE_TWO[t, "Ks"] <- Ks
  COLLECTIVE_TWO[t, "MU"] <- mu
  COLLECTIVE_TWO[t, "AVGPRIV"] <- AVGPVAR
  COLLECTIVE_TWO[t, "AVGCOMB"] <- AVGCVAR
  COLLECTIVE_TWO[t, "THEORC"] <- TheorC
  COLLECTIVE_TWO[t, "THEORP"] <- TheorP
  COLLECTIVE_TWO[t, "V.Sq"] <- V.SQ


}



#Specifications for plotting the segments (Green arrows)
#X_AXIS specifies where on the x-axis to expect a green arrow
#For example, when V.Sq = 0.0484, plot a green vertical arrow.

#Left most arrow
X_Axis_ONE <- COLLECTIVE_TWO[COLLECTIVE_TWO$V.Sq == 0.0484, "AVGPRIV"]

#Center arrow
X_Axis_TWO <- COLLECTIVE_TWO[COLLECTIVE_TWO$V.Sq == 0.1521, "AVGPRIV"]

#Rightmost arrow
X_Axis_THREE <- COLLECTIVE_TWO[COLLECTIVE_TWO$V.Sq == 0.25, "AVGPRIV"]



#Left most arrow
Y_end_ONE <- COLLECTIVE_TWO[COLLECTIVE_TWO$V.Sq == 0.0484, "AVGCOMB"]

#center arrow
Y_end_TWO <- COLLECTIVE_TWO[COLLECTIVE_TWO$V.Sq == 0.1521, "AVGCOMB"]

#Rightmost arrow
Y_end_THREE <- COLLECTIVE_TWO[COLLECTIVE_TWO$V.Sq == 0.25, "AVGCOMB"]
```

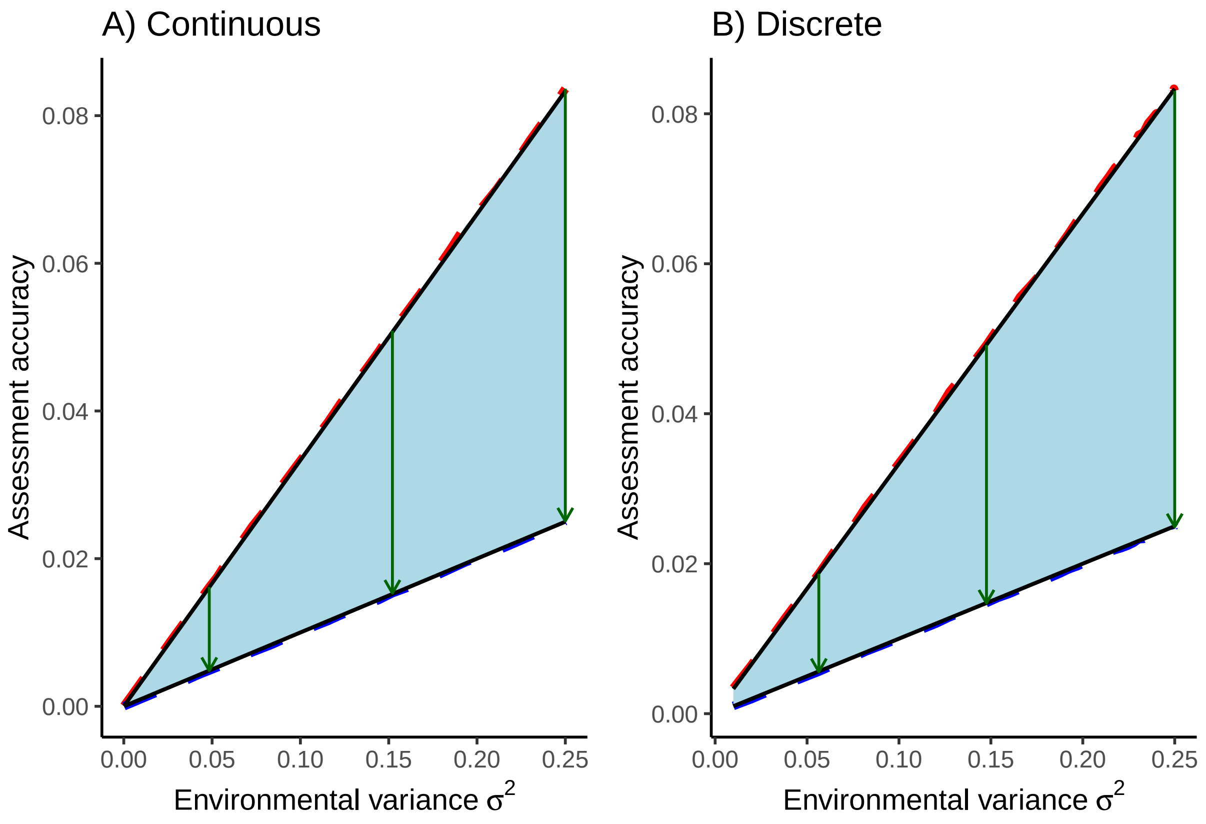
```r
#Dataframe for the 3 total arrows within one wedge plot
GRAPH_SEGMENTS1 <- data.frame(XONE = 0.0484, XTWO = 0.1521, XTHREE = 0.25,
                              YONE = X_Axis_ONE, YTWO = X_Axis_TWO,
                              YTHREE = X_Axis_THREE,
                             YEND_ONE = Y_end_ONE, YEND_TWO = Y_end_TWO,
                             YEND_THREE = Y_end_THREE)


#-----FIGURE 2 SPECIFICATIONS-----#

#Accuracy plot for the normally distributed environment
#Plots personal and combined mean squared deviations and their expected values
#which are extracted from COLLECTIVE_TWO

#X-axis = values of STDV^2 = V.Sq and all other values are layered on top of it
#using the aes (y = AVGPRIV) argument and geom_lines()

#geom_ribbon() shades in color between personal (TOP) and combined (BOTTOM)
#simulated and expected values

ACC_PLOT1 <- ggplot(COLLECTIVE_TWO, aes(x = V.Sq, y = AVGPRIV)) +
  geom_line(size = 1.5, color = "red", linetype = "dashed") +
  geom_line(aes(x = V.Sq, y = AVGCOMB), size = 1.5, color = "blue",
            linetype = "dashed") +
  geom_ribbon(aes(V.Sq, ymin = AVGCOMB, ymax = AVGPRIV), fill = "lightblue") + theme_classic() +
  labs(x = "Environmental variance"~sigma^2, y = "Assessment accuracy", title = "A) Continuous") +
  geom_line(aes(x = V.Sq, y = THEORP), size = 0.7, linetype = "solid",
            color = "black") +
  geom_line(aes(x = V.Sq, y = THEORC), size = 0.7, linetype = "solid",
            color = "black")


#SEGMENTS ARE ADDED ONTO THE PLOT
ACC_PLOT1 <- ACC_PLOT1 +
  geom_segment(data = GRAPH_SEGMENTS1,
               aes(x = XONE, xend = XONE, y = YONE, yend = YEND_ONE),
               size = 0.5, color = "darkgreen",
               arrow = arrow(length = unit(0.2, "centimeters"))) +
  geom_segment(data = GRAPH_SEGMENTS1, aes(x = XTWO, xend = XTWO,
                                            y = YTWO, yend = YEND_TWO),
               size = 0.5, color = "darkgreen",
               arrow = arrow(length = unit(0.2, "centimeters"))) +
  geom_segment(data = GRAPH_SEGMENTS1, aes(x = XTHREE, xend = XTHREE,
                                            y = YTHREE, yend = YEND_THREE),
               size = 0.5, color = "darkgreen",
               arrow = arrow(length = unit(0.2, "centimeters")))




#Bernoulli environment
#values of personal information is discrete (1 or 0)
```

```r
#mean environmental state mu = Epsilon
Epsilon <- 0

#INNER LOOP data frame stores data from 10,000 replicates
#of group average personal assessments (PRIV) and
#combined assessments (COMB)
COLLECTIVE <- data.frame(PRIV = rep(0, REPS), COMB = rep(0, REPS))

#OUTER LOOP data frame stores data from 50 Increments of
#grand means across all replicates
#Model parameters are reported for GGPLOT2 plotting purposes
COLLECTIVE_THREE<- data.frame(K = rep(0, INCREMENT), Ks = rep(0, INCREMENT),
                              AVGPRIV = rep(0, INCREMENT),
                              AVGCOMB = rep(0, INCREMENT),
                          THEORC = rep(0, INCREMENT),
                          THEORP = rep(0, INCREMENT), V.Sq = rep(0, INCREMENT))


for(t in 1:INCREMENT) {

  #Values in this section are incremented from their original values
  #(see above) by a specified amount (see immediately below) until
  #50 steps (INCREMENT) are completed.

  #Mean environmental state
  Epsilon <- Epsilon + 0.01

  #Environmental variance
  V.SQ <- Epsilon*(1 - Epsilon)

  #Theoretical expectation of combined assessment (EQN. 8)
  TheorC <- V.SQ*((1-w)^2 + (w^2)/Ks)/K

  #THeoretical expectation of personal assessment (EQN. 7)
  TheorP <- (V.SQ / K)


  #----------------INNER LOOP----------#
  for( i in 1:REPS){

    #Each replicate (10,000 total) contains information
    #for mean group (N) personal and combined assessments.

    #Processed Personal information (AVG of K Personal samples)
    Private <- c()


    #------------PERSONAL ASSESSMENT--------#
    #Loop performs personal information sampling for
    #each group member in N.
    for (j in 1:N){
```

```
    #Each group member independently samples from a
    #bernoulli environment with mean state epsilon
    PrivateSampling <- rbinom(K, 1, Epsilon)

    #Each individuals personal assessment is the average of
    #its personal samples
    AvgPI <- mean(PrivateSampling)

    #Stores personal samples into the vector specified above
    Private[j] <- AvgPI
  }

  #----------------SOCIAL ASSESSMENT-----------------#
  #Loop performs social information sampling for
  #each group member in N.

  #Processed social information
  Others <- c()


  for(j in 1:N){


    #INTEGRATED SOCIAL INFORMATION
    #Social samples obtained from Ks randomly selected social partners
    SocialSampling <- resample(Private[-j], Ks)

    #Social assessment = average of social samples
    AvgSI <- mean(SocialSampling)

    #social assessment stored into vector above
    Others[j] <- AvgSI
  }

  #any values that = NOT A NUMBER are replaced with 0
  Others[is.nan(Others)] <- 0

  #COMBINED ASSESSMENT
  Combined <- (((1 - w) * Private) + (w * Others))

  #Mean squared deviation across all N individuals for
  #Personal (PVAR) and combined (CVAR) assessments
  PVAR <- mean((Private - Epsilon)^2)
  CVAR <- mean((Combined - Epsilon) ^2)



  #--------------STORES VALUES INTO DATA FRAME FOR INNER LOOP---------------#
  COLLECTIVE[i, "PRIV"] <- PVAR
  COLLECTIVE[i, "COMB"] <- CVAR
}

#Averages across 10,000 replicates for the personal and combined mean squared
```

```
  #Deviations
  AVGPVAR <- mean(COLLECTIVE$PRIV)
  AVGCVAR <- mean(COLLECTIVE$COMB)

  #For plotting we record the value of K, Ks, Epsilon and also store
  #personal and combined mean squared deviations and the expected values of
  #personal and combined assessment accuracies (EQN 7 & 8)
  COLLECTIVE_THREE[t, "K"] <- K
  COLLECTIVE_THREE[t, "Ks"] <- Ks
  COLLECTIVE_THREE[t, "Epsilon"] <- Epsilon
  COLLECTIVE_THREE[t, "AVGPRIV"] <- AVGPVAR
  COLLECTIVE_THREE[t, "AVGCOMB"] <- AVGCVAR
  COLLECTIVE_THREE[t, "THEORC"] <- TheorC
  COLLECTIVE_THREE[t, "THEORP"] <- TheorP
  COLLECTIVE_THREE[t, "V.Sq"] <- V.SQ
}

#Specifications for plotting the segments (Green arrows)
#X_AXIS specifies where on the x-axis to expect a green arrow
#For example, when V.Sq = 0.0484, plot a green vertical arrow.


#Left arrow
X_Axis_one <- COLLECTIVE_THREE[COLLECTIVE_THREE$V.Sq == 0.0564, "AVGPRIV"]

#Center arrow
X_Axis_two <- COLLECTIVE_THREE[COLLECTIVE_THREE$V.Sq == 0.1476, "AVGPRIV"]

#right arrow
X_Axis_three <- COLLECTIVE_THREE[COLLECTIVE_THREE$V.Sq == 0.25, "AVGPRIV"]


#left arrow
Y_end_one <- COLLECTIVE_THREE[COLLECTIVE_THREE$V.Sq == 0.0564, "AVGCOMB"]

#center arrow
Y_end_two <- COLLECTIVE_THREE[COLLECTIVE_THREE$V.Sq == 0.1476, "AVGCOMB"]

#right arrow
Y_end_three <- COLLECTIVE_THREE[COLLECTIVE_THREE$V.Sq == 0.25, "AVGCOMB"]


#Dataframe for the 3 total arrows within one wedge plot
GRAPH_SEGMENTS2 <- data.frame(XONE = 0.0564, XTWO = 0.1476, XTHREE = 0.2500,
                              YONE = X_Axis_one, YTWO = X_Axis_two,
                              YTHREE = X_Axis_three,
                             YEND_ONE = Y_end_one,
                             YEND_TWO = Y_end_two, YEND_THREE = Y_end_three)

#-----FIGURE 2 SPECIFICATIONS-----#

#Accuracy plot for the bernoulli environment
#Plots personal and combined mean squared deviations and their expected values
```

```
#which are extracted from COLLECTIVE_THREE

#X-axis = values of STDV^2 = V.Sq and all other values are layered on top of it
#using the aes (y = AVGPRIV) argument and geom_lines()

#geom_ribbon() shades in color between personal (TOP) and combined (BOTTOM)
#simulated and expected values

ACC_PLOT2 <- ggplot(COLLECTIVE_THREE, aes(x = V.Sq, y = AVGPRIV)) +
  geom_line(size = 1.5, color = "red", linetype = "dashed") +
  geom_line(aes(x = V.Sq, y = AVGCOMB), size = 1.5,
            color = "blue", linetype = "dashed") +
  geom_ribbon(aes(V.Sq, ymin = AVGCOMB, ymax = AVGPRIV),
              fill = "lightblue") + theme_classic() +
  labs(x = "Environmental variance"~sigma^2,
       y = "Assessment accuracy", title = "B) Discrete") +
  geom_line(aes(x = V.Sq, y = THEORP), size = 0.7,
            linetype = "solid", color = "black") +
  geom_line(aes(x = V.Sq, y = THEORC), size = 0.7,
            linetype = "solid", color = "black")

#SEGMENTS ARE ADDED ONTO THE PLOT
ACC_PLOT2 <- ACC_PLOT2 +
  geom_segment(data = GRAPH_SEGMENTS2, aes(x = XONE, xend = XONE,
                                           y = YONE, yend = YEND_ONE),
               size = 0.5, color = "darkgreen",
               arrow = arrow(length = unit(0.2, "centimeters"))) +
  geom_segment(data = GRAPH_SEGMENTS2, aes(x = XTWO, xend = XTWO,
                                           y = YTWO, yend = YEND_TWO),
               size = 0.5, color = "darkgreen",
               arrow = arrow(length = unit(0.2, "centimeters"))) +
  geom_segment(data = GRAPH_SEGMENTS2, aes(x = XTHREE, xend = XTHREE,
                                           y = YTHREE, yend = YEND_THREE),
               size = 0.5, color = "darkgreen",
               arrow = arrow(length = unit(0.2, "centimeters")))



#Combines normally distributed plot with bernoulli plot (CONTINUOUS vs DISCRETE)
(ACC_PLOT1 | ACC_PLOT2)
```



For two environmental distributions (continuous and dichotomous), we visualize the differences in for group changes in accuracy for two different group sizes, N = 10 vs N = 100. **Figure 5**

```
#Figure 5: CONTINUOUS
#The following parameters control the simulation.

#Personal sample size where K is greater than
#or equal to 1
K <- 4

#Social sample size where Ks is greater than
#or equal to one
Ks <- 2

#Social weight controls how much weight is
#placed on the social information relative
#to personal information (1 -w)
w <- 0.5


#Mean environmental state for a normally
#distributed environment
mu <- 0

#Standard deviation of the environmental state
STDV <- 1
```

```r
#Group Size
N <- 10

#Number of replicates performed per
#value in INCREMENT. For example,
#the model is performed 10,000 times
#for each value in INCREMENT (50).
REPS <- 10000



#We introduce this resample function because, according to R documentation,
#the core function sample() can give 'undesired behaviour' when applied to
#sequences with only 1 element
resample <- function(x, Ks,...) x[sample.int(length(x), size=Ks,...)]




#Eqn. 10 specifies value of W that maximizes the value of social information
#under different K and Ks parameters
W_HAT <- Ks/(K + Ks)

#Eqn. 9- expected change in accuracy
EXP_DA <- ((((STDV^2)*w)/K)*(2 - (w/W_HAT)))


#Dataframe that stores data for each replicate (10,000 total) for the group avg.
#change in accuracy (DACC)
COLLECTIVE <- data.frame(DACC = rep(0, REPS))



for(i in 1:REPS){



  #Matrix of values that represent all individuals personal samples
  #Column 1 = Individual ID
  #Columns 2-K represent the personal samples
  PRIVATE_SAMPLE_LIST <- list()

  #vector of personl assessments
  P.ASSESSMENT <- c()

  for(j in 1:N){


    #Sample environment with mean mu, STDV, and variance= STDV^2 K times
    PRIVATE_SAMPLE <- rnorm(K, mu, STDV)

    #Store individual i's K personal samples into a list
    PRIVATE_SAMPLE_LIST[[j]] <- PRIVATE_SAMPLE
    #average K personal samples for personal assessment
    PRIV_ASSESS <- mean(PRIVATE_SAMPLE)

    #store individual i's personal assessment into vector
    P.ASSESSMENT[j] <- PRIV_ASSESS
  }
  #Bind each individual's personal samples into a matrix
  PRIVATE_SAMPLE_LIST <- do.call(rbind,PRIVATE_SAMPLE_LIST)





  #-----------SOCIAL ASSESSMENT-----------#

  #Stores social information acquired from matrix
  SOCIAL_MATRIX2 <- c()

  #individual indicies (1 - N)
  MEMBERS <- seq(N)


  for (j in 1:N) {

    #Select Ks number of social partners from which to sample from
    PARTNERS <- resample(MEMBERS[-j], Ks)

    #Ensure that the value in the social info storage is empty
    SOCIAL_MATRIX2[j] = 0


    for (k in PARTNERS) {

      #For the total value Ks, acquire 1 personal sample from social partner
      SOCIAL_MATRIX2[j] <- SOCIAL_MATRIX2[j] +
        resample(PRIVATE_SAMPLE_LIST[k, ], 1)
    }

    #Mean value of social samples = social assessment, store it.
    SOCIAL_MATRIX2[j] <- (SOCIAL_MATRIX2[j]/Ks)
  }

  #any NAN numbers get converted into 0
  SOCIAL_MATRIX2[is.nan(SOCIAL_MATRIX2)] <- 0


  #Combined Assessment
  COMBINED <- (w*SOCIAL_MATRIX2 + (1-w)*P.ASSESSMENT)





  #Model Analysis
```

```r
  #mean squared deviations for all N individuals for personal and combined
  #assessment accuracies
  PRIVATE_ACCURACY <- mean((P.ASSESSMENT - mu)^2)
  COMBINED_ACCURACY <- mean((COMBINED - mu)^2)


  #Group change in accuracy
  DACC <- (PRIVATE_ACCURACY - COMBINED_ACCURACY)

  #Standardize group chanve in accuracy
  DACC <- DACC / (STDV^2)



  #Store group change in accuracy into COLLECTIVE Data.frame
  COLLECTIVE[i, "DACC"] <- DACC
}

#variance of Delta Accuracy from REPS
DACC_VARIATION <- var(COLLECTIVE$DACC)

Ten_DACC_Mean <- mean(COLLECTIVE$DACC)




#new group size
N <- 100

#Dataframe that stores data for each replicate (10,000 total) for the group avg.
#change in accuracy (DACC)
COLLECTIVE2 <- data.frame(DACC = rep(0, REPS))

for(i in 1:REPS){

  #Matrix of values that represent all individuals personal samples
  #Column 1 = Individual ID
  #Columns 2-K represent the personal samples
  PRIVATE_SAMPLE_LIST <- list()

  #vector of personl assessments
  P.ASSESSMENT <- c()

  for(j in 1:N){


    #Sample environment with mean mu, STDV, and variance= STDV^2 K times
    PRIVATE_SAMPLE <- rnorm(K, mu, STDV)

    #Store individual i's K personal samples into a list
    PRIVATE_SAMPLE_LIST[[j]] <- PRIVATE_SAMPLE

    #average K personal samples for personal assessment
  PRIV_ASSESS <- mean(PRIVATE_SAMPLE)

  #store individual i's personal assessment into vector
  P.ASSESSMENT[j] <- PRIV_ASSESS
}
#Bind each individual's personal samples into a matrix
PRIVATE_SAMPLE_LIST <- do.call(rbind,PRIVATE_SAMPLE_LIST)



#-----------SOCIAL ASSESSMENT-----------#

#Stores social information acquired from matrix
SOCIAL_MATRIX2 <- c()

#individual indicies (1 - N)
MEMBERS <- seq(N)


for (j in 1:N) {

  #Select Ks number of social partners from which to sample from
  PARTNERS <- resample(MEMBERS[-j], Ks)

  #Ensure that the value in the social info storage is empty
  SOCIAL_MATRIX2[j] = 0


  for (k in PARTNERS) {

    #For the total value Ks, acquire 1 personal sample from social partner
    SOCIAL_MATRIX2[j] <- SOCIAL_MATRIX2[j] +
      resample(PRIVATE_SAMPLE_LIST[k, ], 1)
  }

  #Mean value of social samples = social assessment, store it.
  SOCIAL_MATRIX2[j] <- (SOCIAL_MATRIX2[j]/Ks)
}

#any NAN numbers get converted into 0
SOCIAL_MATRIX2[is.nan(SOCIAL_MATRIX2)] <- 0


#Combined Assessment
COMBINED <- (w*SOCIAL_MATRIX2 + (1-w)*P.ASSESSMENT)




#Model Analysis
#mean squared deviations for all N individuals for personal and combined
#assessment accuracies
PRIVATE_ACCURACY <- mean((P.ASSESSMENT - mu)^2)
```

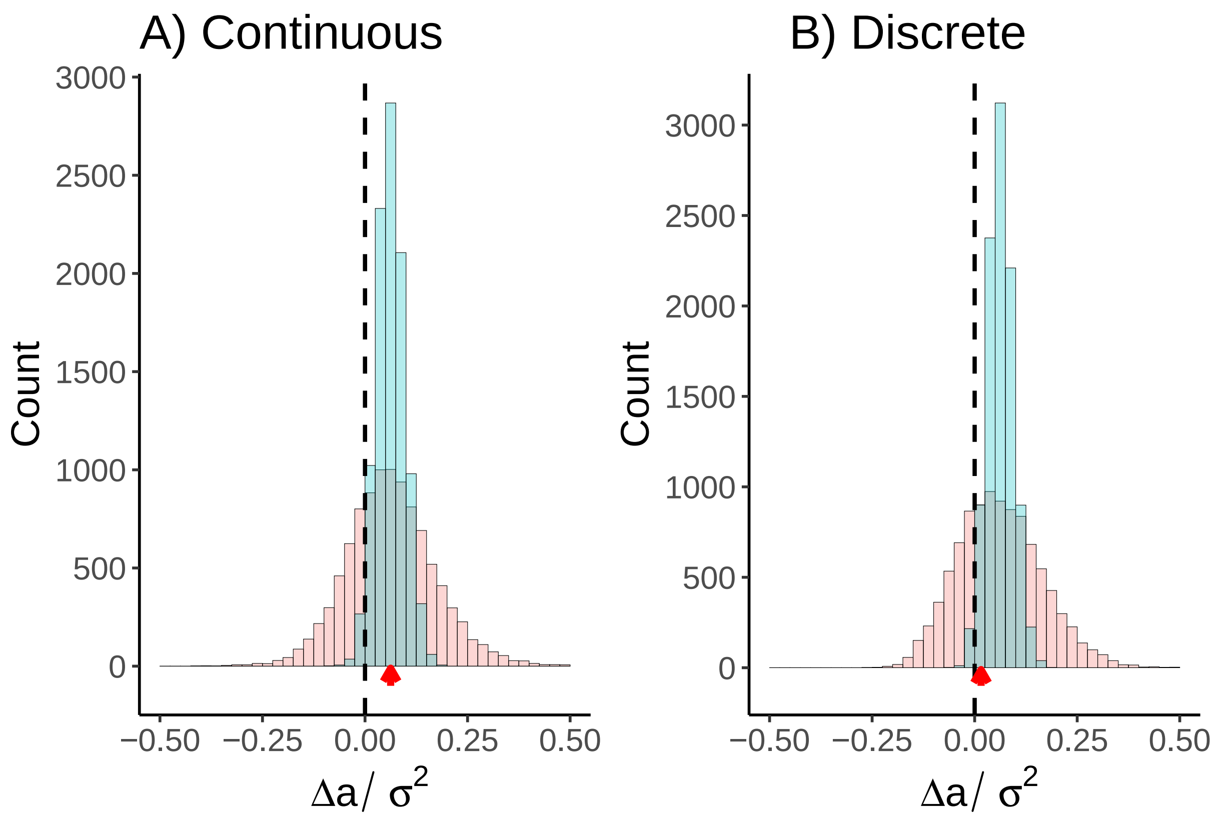
```r
  COMBINED_ACCURACY <- mean((COMBINED - mu)^2)


  #Group change in accuracy
  DACC <- (PRIVATE_ACCURACY - COMBINED_ACCURACY)

  #Standardize group chanve in accuracy
  DACC <- DACC / (STDV^2)

  #Store group change in accuracy into COLLECTIVE Data.frame
  COLLECTIVE2[i, "DACC"] <- DACC
}

#variance of Delta Accuracy from REPS
DACC_VARIATION2 <- var(COLLECTIVE2$DACC)
Hundred_DACC_Mean <- mean(COLLECTIVE2$DACC)


#PLOTTING SPECIFICATIONS

#Boundaries of X-AXIS
X_LIM_MAX <- 0.5
X_LIM_MIN <- -0.5

#Binwidth
BIN_WIDTH <- 0.005

#Dashed lines for Expectation
DASH_SIZE <- 0.75


#GGPLOT purposes to identify between different data sets
NTEN <- rep(10, 10000)
NHUNDRED <- rep(100, 10000)

#Create dataframes for plotting purposes
NTENDF <- data.frame(N = NTEN, DACC = COLLECTIVE$DACC)
NHUNDREDDF <- data.frame(N = NHUNDRED, DACC = COLLECTIVE2$DACC)
NDATAFRAME <- rbind(NTENDF, NHUNDREDDF)
NDATAFRAME$N <- as.factor(NDATAFRAME$N)


#Y positions for arrow denoting expected value
GRAPH_SEGMENTS2 <- data.frame(XONE = EXP_DA, YONE = -100, YEND_ONE = -10)



#Plots from NDATAFRAME (merge of N = 10 and N = 100 data)
#teal represents N = 100 Data
#Pink represents N = 10 data
OVERLAY_HIST <- ggplot(NDATAFRAME, aes(x = DACC, fill = factor(N))) +
  geom_histogram(position = "identity", alpha = 0.3, breaks = seq(-0.5, 0.5, 0.025), color = "black",
                 size = 0.1, show.legend = FALSE) +
  geom_vline(xintercept = 0, color = "black", linetype = "dashed", size = DASH_SIZE) +
  labs(x = expression(~ Delta*"a"/~sigma^2), y = "Count", title = "A) Continuous") +
  theme_classic() +
  geom_segment(data = GRAPH_SEGMENTS2, aes(x = XONE, xend = XONE, y = YONE, yend = YEND_ONE),
               size = 1.25, color = "red", arrow = arrow(length = unit(0.2,"centimeters"))) +
  theme(text= element_text(size = 15)) + scale_y_continuous(breaks = seq(0, 3000, 500))


#--------------------------------------------------------------------------------#

#==================DISCRETE========================#
#The following parameters control the simulation.

#Personal sample size where K is greater than
#or equal to 1
K <- 4

#Social sample size where Ks is greater than
#or equal to one
Ks <- 2

#Social weight controls how much weight is
#placed on the social information relative
#to personal information (1 -w)
w <- 0.5


#Mean environmental state for a normally
#distributed environment
Epsilon <- 0.5

#Standard deviation of the environmental state
STDV <- 1

#Group Size
N <- 10

#Number of replicates performed per
#value in INCREMENT. For example,
#the model is performed 10,000 times
#for each value in INCREMENT (50).
REPS <- 10000



#We introduce this resample function because, according to R documentation,
#the core function sample() can give 'undesired behaviour' when applied to
#sequences with only 1 element
resample <- function(x, Ks,...) x[sample.int(length(x), size=Ks,...)]
```

```r
#Eqn. 10 specifies value of W that maximizes the value of social information
#under different K and Ks parameters
W_HAT <- Ks/(K + Ks)

#Environmental variance
V.SQ <- Epsilon*(1 - Epsilon)


#Eqn. 9- expected change in accuracy
EXP_DA <- ((((V.SQ)*w)/K)*(2 - (w/W_HAT)))


#Dataframe that stores data for each replicate (10,000 total) for the group avg.
#change in accuracy (DACC)
COLLECTIVE3 <- data.frame(DACC = rep(0, REPS))



for(i in 1:REPS){



  #Matrix of values that represent all individuals personal samples
  #Column 1 = Individual ID
  #Columns 2-K represent the personal samples
  PRIVATE_SAMPLE_LIST <- list()

  #vector of personl assessments
  P.ASSESSMENT <- c()

  for(j in 1:N){


    #Sample environment with mean mu, STDV, and variance= STDV^2 K times
    PRIVATE_SAMPLE <- rbinom(K, 1, Epsilon)

    #Store individual i's K personal samples into a list
    PRIVATE_SAMPLE_LIST[[j]] <- PRIVATE_SAMPLE

    #average K personal samples for personal assessment
    PRIV_ASSESS <- mean(PRIVATE_SAMPLE)

    #store individual i's personal assessment into vector
    P.ASSESSMENT[j] <- PRIV_ASSESS
  }
  #Bind each individual's personal samples into a matrix
  PRIVATE_SAMPLE_LIST <- do.call(rbind,PRIVATE_SAMPLE_LIST)






  #-----------SOCIAL ASSESSMENT-----------#
```

```r
#Stores social information acquired from matrix
SOCIAL_MATRIX2 <- c()

#individual indicies (1 - N)
MEMBERS <- seq(N)


for (j in 1:N) {

  #Select Ks number of social partners from which to sample from
  PARTNERS <- resample(MEMBERS[-j], Ks)

  #Ensure that the value in the social info storage is empty
  SOCIAL_MATRIX2[j] = 0


  for (k in PARTNERS) {

    #For the total value Ks, acquire 1 personal sample from social partner
    SOCIAL_MATRIX2[j] <- SOCIAL_MATRIX2[j] +
      resample(PRIVATE_SAMPLE_LIST[k, ], 1)
  }

  #Mean value of social samples = social assessment, store it.
  SOCIAL_MATRIX2[j] <- (SOCIAL_MATRIX2[j]/Ks)
}

#any NAN numbers get converted into 0
SOCIAL_MATRIX2[is.nan(SOCIAL_MATRIX2)] <- 0


#Combined Assessment
COMBINED <- (w*SOCIAL_MATRIX2 + (1-w)*P.ASSESSMENT)




#Model Analysis
#mean squared deviations for all N individuals for personal and combined
#assessment accuracies
PRIVATE_ACCURACY <- mean((P.ASSESSMENT - Epsilon)^2)
COMBINED_ACCURACY <- mean((COMBINED - Epsilon)^2)


#Group change in accuracy
DACC <- (PRIVATE_ACCURACY - COMBINED_ACCURACY)

#Standardize group chanve in accuracy
DACC <- DACC / V.SQ



#Store group change in accuracy into COLLECTIVE Data.frame
```

```r
  COLLECTIVE3[i, "DACC"] <- DACC
}

#variance of Delta Accuracy from REPS
DACC_VARIATION3 <- var(COLLECTIVE3$DACC)
Ten_DACC_Mean2 <- mean(COLLECTIVE3$DACC)




#new group size
N <- 100

#Dataframe that stores data for each replicate (10,000 total) for the group avg.
#change in accuracy (DACC)
COLLECTIVE4 <- data.frame(DACC = rep(0, REPS))

for(i in 1:REPS){

  #Matrix of values that represent all individuals personal samples
  #Column 1 = Individual ID
  #Columns 2-K represent the personal samples
  PRIVATE_SAMPLE_LIST <- list()

  #vector of personl assessments
  P.ASSESSMENT <- c()

  for(j in 1:N){


    #Sample environment with mean mu, STDV, and variance= STDV^2 K times
    PRIVATE_SAMPLE <- rbinom(K, 1, Epsilon)

    #Store individual i's K personal samples into a list
    PRIVATE_SAMPLE_LIST[[j]] <- PRIVATE_SAMPLE

    #average K personal samples for personal assessment
    PRIV_ASSESS <- mean(PRIVATE_SAMPLE)

    #store individual i's personal assessment into vector
    P.ASSESSMENT[j] <- PRIV_ASSESS
  }
  #Bind each individual's personal samples into a matrix
  PRIVATE_SAMPLE_LIST <- do.call(rbind,PRIVATE_SAMPLE_LIST)



  #-----------SOCIAL ASSESSMENT-----------#

  #Stores social information acquired from matrix
  SOCIAL_MATRIX2 <- c()

  #individual indicies (1 - N)
  MEMBERS <- seq(N)


  for (j in 1:N) {

    #Select Ks number of social partners from which to sample from
    PARTNERS <- resample(MEMBERS[-j], Ks)

    #Ensure that the value in the social info storage is empty
    SOCIAL_MATRIX2[j] = 0


    for (k in PARTNERS) {

      #For the total value Ks, acquire 1 personal sample from social partner
      SOCIAL_MATRIX2[j] <- SOCIAL_MATRIX2[j] +
        resample(PRIVATE_SAMPLE_LIST[k, ], 1)
    }

    #Mean value of social samples = social assessment, store it.
    SOCIAL_MATRIX2[j] <- (SOCIAL_MATRIX2[j]/Ks)
  }

  #any NAN numbers get converted into 0
  SOCIAL_MATRIX2[is.nan(SOCIAL_MATRIX2)] <- 0


  #Combined Assessment
  COMBINED <- (w*SOCIAL_MATRIX2 + (1-w)*P.ASSESSMENT)




  #Model Analysis
  #mean squared deviations for all N individuals for personal and combined
  #assessment accuracies
  PRIVATE_ACCURACY <- mean((P.ASSESSMENT - Epsilon)^2)
  COMBINED_ACCURACY <- mean((COMBINED - Epsilon)^2)


  #Group change in accuracy
  DACC <- (PRIVATE_ACCURACY - COMBINED_ACCURACY)

  #Standardize group chanve in accuracy
  DACC <- DACC / V.SQ

  #Store group change in accuracy into COLLECTIVE Data.frame
  COLLECTIVE4[i, "DACC"] <- DACC
}

#variance of Delta Accuracy from REPS
DACC_VARIATION4 <- var(COLLECTIVE4$DACC)
Hundred_DACC_Mean2 <- mean(COLLECTIVE4$DACC)
```

```r
#PLOTTING SPECIFICATIONS

#Boundaries of X-AXIS
X_LIM_MAX <- 0.15
X_LIM_MIN <- -0.15

#Binwidth
BIN_WIDTH <- 0.0000005

#Dashed lines for Expectation
DASH_SIZE <- 0.75

#GGPLOT purposes to identify between different data sets
NTEN <- rep(10, 10000)
NHUNDRED <- rep(100, 10000)

#Create dataframes for plotting purposes
NTENDF2 <- data.frame(N = NTEN, DACC = COLLECTIVE3$DACC)
NHUNDREDDF2 <- data.frame(N = NHUNDRED, DACC = COLLECTIVE4$DACC)
NDATAFRAME2 <- rbind(NTENDF2, NHUNDREDDF2)
NDATAFRAME2$N <- as.factor(NDATAFRAME2$N)


#Y positions for arrow denoting expected value
GRAPH_SEGMENTS2 <- data.frame(XONE = EXP_DA, YONE = -100, YEND_ONE = -10)



#Plots from NDATAFRAME (merge of N = 10 and N = 100 data)
#teal represents N = 100 Data
#Pink represents N = 10 data
OVERLAY_HIST2 <- ggplot(NDATAFRAME2, aes(x = DACC, fill = factor(N))) +
  geom_histogram(position = "identity", alpha = 0.3, breaks = seq(-0.5, 0.5, 0.025), color = "black",
                 size = 0.1, show.legend = FALSE) +
  geom_vline(xintercept = 0, color = "black", linetype = "dashed", size = DASH_SIZE) +
  labs(x = expression(~ Delta*"a"/~sigma^2), y = "Count", title  = "   B) Discrete") +
  theme_classic() +
  geom_segment(data = GRAPH_SEGMENTS2, aes(x = XONE, xend = XONE, y = YONE, yend = YEND_ONE),
               size = 1.25, color = "red", arrow = arrow(length = unit(0.2,"centimeters"))) +
  theme(text= element_text(size = 15)) + scale_y_continuous(breaks = seq(0, 3000, 500))



(OVERLAY_HIST | OVERLAY_HIST2)
```



Displays how different combinations of personal and social information sample sizes impacts the proportion of those that experience improvements, reductions, or no changes in accuracy for dichotomous and continuous environments **Figure 6**.

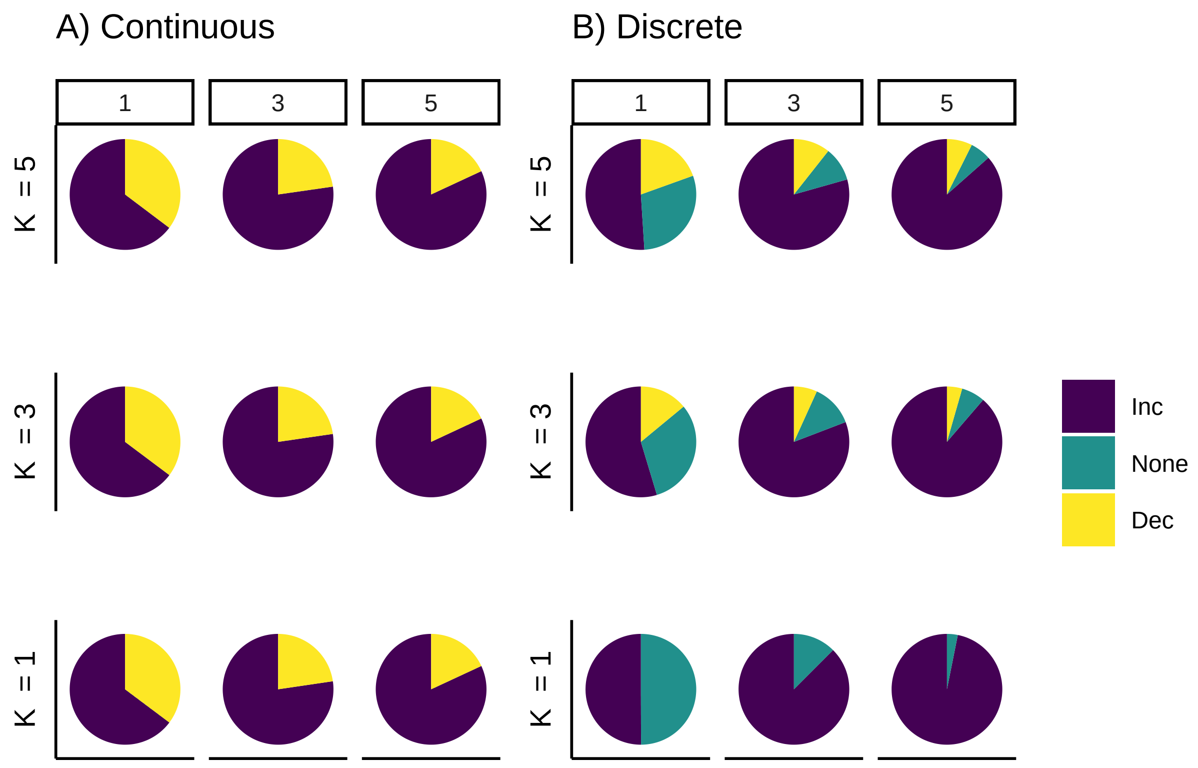
```
#Figure 6
#Group Size
N <- 100

#Total replicates performed fro number of sims (4)
reps <- 10000

#Personal sample size
K <- -1

#Social sample size
Ks <- -1

#social weight
w <- 0.5

#mean environmental state mu = Epsilon
Epsilon <- 0.5

#Increment values for personal and social sample size
Ks.Sims <- 3
KSims <- 3
```

```r
#the value to increment by
Ks.inc <- 2
K.inc <- 2


#Inner loop-- stores values for those that increase, decrease and experience
#no change in accruacy per group (mean group proportions)
proportions <- data.frame(Inc = rep(0, reps), Dec = rep(0, reps),
                          None = rep(0, reps))

#Outerloop - stores values of grand mean proportions (average across replicates)
prop.as <- data.frame(K = rep(0, Ks.Sims),
                                 Ks = rep(0, Ks.Sims),
                                 Inc = rep(0, Ks.Sims),Dec = rep(0, Ks.Sims),
                                 None = rep(0, Ks.Sims))


#For data frame merging
p.tot1 <- list()
stacked_prop <- list()

#We introduce this resample function because, according to R documentation,
#the core function sample() can give 'undesired behaviour' when applied to
#sequences with only 1 element
resample <- function(x, Ks,...) x[sample.int(length(x), size=Ks,...)]


for (t in 1:KSims) {

  #Increment the value of K by 1 each time
  K <- K + K.inc

  for (k in 1:Ks.Sims){

    #increment the value of Ks by 1 each time
    Ks <- Ks + Ks.inc

    for (i in 1:reps){


      #Personal sample size vector
      Private <- c()

      #social sample size vector
      Others <- c()



      #-----PERSONAL ASSESSMENT-----#
      for(j in 1:N) {

        #Each individual acquires K personal samples from environment with mean
        #state equivilanet to Mu = Epsilon
        PA <- rbinom (K, 1, Epsilon)

        #Mean of those personal samples is individual i's personal assessment
        AvgPA <- mean(PA)

        #Store personal assessment into vector
        Private[j] <- AvgPA
      }

      #-----------SOCIAL ASSESSMENT-------#
      for(j in 1:N) {

        #Sample from vector of personal assessments
        y <- resample(Private[-j], Ks)

        #Average of soial samples is social assessment
        AvgOthers <- mean(y)

        #store information into vector
        Others[j] <- AvgOthers
      }

      #NAN values converted into 0
      Others[is.nan(Others)] <- 0


      #Combined Assessment
      Combined <- (((1 - w) * Private) + (w * Others))


      #-----------ACCURACY CALCULATIONS-----------------#
      #vector of epsilon values
      Quality <- rep(Epsilon, N)

      #squared deviations
      P.acc <- (Quality - Private)^2
      C.acc <- (Quality - Combined)^2

      #individual level changes in accuracy
      Dacc <- (P.acc - C.acc)

      #-------PROPORTIONS OF INC, DEC, and UNEFFECTED-------#

      #increasing in accuracy
      inc.prop <- ((length(Dacc[Dacc > 0])) / N)

      #decreasing in accuracy
      dec.prop <- ((length(Dacc[Dacc < 0])) / N)

      #no change in accuracy
      none.prop <- 1 - (inc.prop + dec.prop)
```

```
      #Store values into inner loop
      proportions[i, "Inc"] <- inc.prop
      proportions[i, "Dec"] <- dec.prop
      proportions[i, "None"] <- none.prop
    }


    #---Grand Means---#
    #total proportions across simulation
    inc.across <- round(mean(proportions$Inc), 5)
    dec.across <- round(mean(proportions$Dec), 5)
    none.across <- round(mean(proportions$None), 5)


    #Store into outer loop data frame
    #-----------------PROPORTIONS------------------------#
    prop.as[k, "K"] <- K
    prop.as[k, "Ks"] <- Ks
    prop.as[k, "Inc"] <- inc.across
    prop.as[k, "Dec"] <- dec.across
    prop.as[k, "None"] <- none.across

  }

  #merge each increment data frame into list
  p.tot1[[t]] <- prop.as

  #Reset the value of social sample each time K is incremented
  Ks <- -1
}

#once the simulation is complete, merge all dataframes together
stacked_prop <- bind_rows(p.tot1)


#melt data frame according to stacked barplot requirements
stacked_prop1 <- melt(stacked_prop, id.vars = c("K", "Ks"), measure.vars = c("Inc", "None", "Dec"))



K_ONE_DF <- stacked_prop1[stacked_prop1$K == 1, ]

K_ONE_PLOT <- ggplot(K_ONE_DF, aes(x = K, y = value, fill = variable)) +
  geom_bar(stat = "identity", position = "stack") + coord_polar("y") + facet_wrap(~Ks) + theme_classic(
  theme(axis.text = element_blank(), axis.ticks = element_blank(), panel.grid = element_blank(),
        strip.background = element_blank(), strip.text = element_blank(), legend.position = "none") +
  scale_fill_viridis_d(name= "", option = "D") + labs( x = "K  = 1", y = "")


K_THREE_DF <- stacked_prop1[stacked_prop1$K == 3, ]

K_THREE_PLOT <- ggplot(K_THREE_DF, aes(x = K, y = value, fill = variable)) +
  geom_bar(stat = "identity", position = "stack") + coord_polar("y") + facet_wrap(~Ks) + theme_classic(
  theme(axis.text = element_blank(), axis.ticks = element_blank(), panel.grid = element_blank(),
        strip.background = element_blank(), strip.text = element_blank(), panel.border = element_blank(
  scale_fill_viridis_d(name= "", option = "D")  + labs( x = "K  = 3", y = "")


K_FIVE_DF <- stacked_prop1[stacked_prop1$K == 5, ]

K_FIVE_PLOT <- ggplot(K_FIVE_DF, aes(x = K, y = value, fill = variable)) +
  geom_bar(stat = "identity", position = "stack") + coord_polar("y") + facet_wrap(~Ks) + theme_classic(
  theme(axis.text = element_blank(), axis.ticks = element_blank(), panel.grid = element_blank(), axis.l
  scale_fill_viridis_d(name= "", option = "D") + labs( x = "K  = 5", y = "", title = "B) Discrete")


DISCRETE_PLOTS <- (K_FIVE_PLOT / K_THREE_PLOT / K_ONE_PLOT)








#-------------------------------------------------CONTINUOUS-------------------#







#Group Size
N <- 100

#Total replicates performed fro number of sims (4)
reps <- 10000

#Personal sample size
K <- -1

#Social sample size
Ks <- -1

#social weight
w <- 0.5

#mean environmental state mu = Epsilon
mu <- 0

#Standard deviation
STDV <- 1

#Increment values for personal and social sample size
Ks.Sims <- 3
KSims <- 3
```

```r
#the value to increment by
Ks.inc <- 2
K.inc <- 2


#Inner loop-- stores values for those that increase, decrease and experience
#no change in accruacy per group (mean group proportions)
proportions2 <- data.frame(Inc = rep(0, reps), Dec = rep(0, reps),
                           None = rep(0, reps))

#Outerloop - stores values of grand mean proportions (average across replicates)
prop.as2 <- data.frame(K = rep(0, Ks.Sims),
                                  Ks = rep(0, Ks.Sims),
                                  Inc = rep(0, Ks.Sims),Dec = rep(0, Ks.Sims),
                                  None = rep(0, Ks.Sims))


#For data frame merging
p.tot12 <- list()
stacked_prop2 <- list()

#We introduce this resample function because, according to R documentation,
#the core function sample() can give 'undesired behaviour' when applied to
#sequences with only 1 element
resample <- function(x, Ks,...) x[sample.int(length(x), size=Ks,...)]


for (t in 1:KSims) {

  #Increment the value of K by 1 each time
  K <- K + K.inc

  for (k in 1:Ks.Sims){

    #increment the value of Ks by 1 each time
    Ks <- Ks + Ks.inc

    for (i in 1:reps){


      #Personal sample size vector
      Private <- c()

      #social sample size vector
      Others <- c()



      #-----PERSONAL ASSESSMENT-----#
      for(j in 1:N) {

        #Each individual acquires K personal samples from environment with mean
        #state equivilanet to Mu
        PA <- rnorm(K, mu, STDV)

        #Mean of those personal samples is individual i's personal assessment
        AvgPA <- mean(PA)

        #Store personal assessment into vector
        Private[j] <- AvgPA
      }

      #-----------SOCIAL ASSESSMENT-------#
      for(j in 1:N) {

        #Sample from vector of personal assessments
        y <- resample(Private[-j], Ks)

        #Average of social samples is social assessment
        AvgOthers <- mean(y)

        #store information into vector
        Others[j] <- AvgOthers
      }

      #NAN values converted into 0
      Others[is.nan(Others)] <- 0


      #Combined Assessment
      Combined <- (((1 - w) * Private) + (w * Others))


      #-----------ACCURACY CALCULATIONS-----------------#
      #vector of epsilon values
      Quality <- rep(mu, N)

      #squared deviations
      P.acc <- (Quality - Private)^2
      C.acc <- (Quality - Combined)^2

      #individual level changes in accuracy
      Dacc <- (P.acc - C.acc)

      #-------PROPORTIONS OF INC, DEC, and UNEFFECTED-------#

      #increasing in accuracy
      inc.prop <- ((length(Dacc[Dacc > 0])) / N)

      #decreasing in accuracy
      dec.prop <- ((length(Dacc[Dacc < 0])) / N)

      #no change in accuracy
      none.prop <- 1 - (inc.prop + dec.prop)
```

```r
      #Store values into inner loop
      proportions2[i, "Inc"] <- inc.prop
      proportions2[i, "Dec"] <- dec.prop
      proportions2[i, "None"] <- none.prop
    }


    #---Grand Means---#
    #total proportions across simulation
    inc.across <- round(mean(proportions2$Inc), 5)
    dec.across <- round(mean(proportions2$Dec), 5)
    none.across <- round(mean(proportions2$None), 5)


    #Store into outer loop data frame
    #-----------------PROPORTIONS------------------------#
    prop.as2[k, "K"] <- K
    prop.as2[k, "Ks"] <- Ks
    prop.as2[k, "Inc"] <- inc.across
    prop.as2[k, "Dec"] <- dec.across
    prop.as2[k, "None"] <- none.across

  }

  #merge each increment data frame into list
  p.tot12[[t]] <- prop.as2

  #Reset the value of social sample each time K is incremented
  Ks <- -1
}

#once the simulation is complete, merge all dataframes together
stacked_prop2 <- bind_rows(p.tot12)


#melt data frame according to stacked barplot requirements
stacked_prop12 <- melt(stacked_prop2, id.vars = c("K", "Ks"), measure.vars = c("Inc", "None", "Dec"))




K_ONE_DF2 <- stacked_prop12[stacked_prop12$K == 1, ]

K_ONE_PLOT2 <- ggplot(K_ONE_DF2, aes(x = K, y = value, fill = variable)) +
  geom_bar(stat = "identity", position = "stack") + coord_polar("y") + facet_wrap(~Ks) + theme_classic(
  theme(axis.text = element_blank(), axis.ticks = element_blank(), panel.grid = element_blank(),
        strip.background = element_blank(), strip.text = element_blank(), legend.position = "none") +
  scale_fill_viridis_d(name= "", option = "D") + labs( x = "K  = 1", y = "")


K_THREE_DF2 <- stacked_prop12[stacked_prop12$K == 3, ]

K_THREE_PLOT2 <- ggplot(K_THREE_DF2, aes(x = K, y = value, fill = variable)) +
  geom_bar(stat = "identity", position = "stack") + coord_polar("y") + facet_wrap(~Ks) + theme_classic(
  theme(axis.text = element_blank(), axis.ticks = element_blank(), panel.grid = element_blank(),
        strip.background = element_blank(), strip.text = element_blank(), panel.border = element_blank(
  scale_fill_viridis_d(name= "", option = "D")  + labs( x = "K  = 3", y = "")


K_FIVE_DF2 <- stacked_prop12[stacked_prop12$K == 5, ]

K_FIVE_PLOT2 <- ggplot(K_FIVE_DF2, aes(x = K, y = value, fill = variable)) +
  geom_bar(stat = "identity", position = "stack") + coord_polar("y") + facet_wrap(~Ks) + theme_classic(
  theme(axis.text = element_blank(), axis.ticks = element_blank(), panel.grid = element_blank(), axis.li
  scale_fill_viridis_d(name= "", option = "D") + labs( x = "K  = 5", y = "", title = "A) Continuous")


CONTINUOUS_PLOTS <- (K_FIVE_PLOT2 / K_THREE_PLOT2 / K_ONE_PLOT2)



(CONTINUOUS_PLOTS | DISCRETE_PLOTS )
```


A) Continuous
B) Discrete
K = 5
K = 3
K = 1
Inc
None
Dec