Social Associations

Kyra Bankhead

2023-03-02

In this markdown I will:

1. Subset data to easily visualize analytical steps. Then, create an association matrix from the gambit of the group assumption using the simple-ratio index function.
2. Form repeated permutations from the true association matrix to create a null distribution of their coefficients of variations (CV). This will determine if the true CV ranges outside of what is expected by chance associations.
3. Calculate the average SRI index within different combinations of HI status pairings.

PART 1: *Social Association Matrix*

Association Matrices will have an annual temporal resolution.

To ensure enough information is available to analyze individual association preference, I have made sure that each association matrix only indludes individuals that were seen at least 10 times in a year.

Create gambit of the group index

```
# Group each individual by date and sighting
# Gambit of the group index
gbi <- readRDS("../data/gbi.RData")
```

Calculate simple-ratio index with group by individual matrix above.

```
# Create association matrix
nxn <- readRDS("../data/nxn.RData")
```

PART 2: *Permutations*

Create 1000 repeated matrices from true matrix to form null distribution

Permute association within samples to test long-term associations and control for gregariousness.

True association index coefficient of variation

CV = (SD/mean)*100

Form CV distribution

We can reject the null hypothesis that individuals associate at random and conclude that there is evidence that associations are different from what we would expect by chance. Since the CV(TAI) is lower than the other CV, the associations are lower than expected.

PART 3: *SRI Within HI Status Pairs*

Assign HI Status to each individual within the SRI matrix

```
# Read in different behavior's data frames
IDbehav_Beg <- readRDS("../data/IDbehav_Beg.RData")
IDbehav_Pat <- readRDS("../data/IDbehav_Pat.RData")
IDbehav_Dep <- readRDS("../data/IDbehav_Dep.RData")

# Get unique behavior assignments
status <- function(IDbehav, HI, NonHI){
  lapply(seq_along(IDbehav), function(i) {
    IDbehav[[i]]$Stat <- ifelse(IDbehav[[i]]$HI > 0, HI, NonHI)
    df <- IDbehav[[i]][, c('Code', 'Stat')]
    df
    })
}

## Match each individual with it's behavior
Beg <- status(IDbehav_Beg, "B", "NB")
Pat <- status(IDbehav_Pat, "P", "NP")
Dep <- status(IDbehav_Dep, "D", "ND")

# Replace individuals in the matrix with their assigned behavior
replace_ID_with_HI <- function(sri_matrix, ID_HI_df) {
  # Create vector that matches IDs to their stat
  id_to_stat <- setNames(ID_HI_df$Stat, ID_HI_df$Code)

  # Replace each ID with stat in row and column names
  row_names <- id_to_stat[rownames(sri_matrix)]
  col_names <- id_to_stat[colnames(sri_matrix)]

  # Create the replaced matrix
  replaced_matrix <- sri_matrix

  # Assign row and column names with behavioral states
  dimnames(replaced_matrix) <- list(row_names, col_names)
  return(replaced_matrix)
}
```

```r
# Make a replaced nxn for each behavior
Beg_nxn <- lapply(seq_along(nxn), function(i) {
  replace_ID_with_HI(nxn[[i]], Beg[[i]])
})

Pat_nxn <- lapply(seq_along(nxn), function(i) {
  replace_ID_with_HI(nxn[[i]], Pat[[i]])
})

Dep_nxn <- lapply(seq_along(nxn), function(i) {
  replace_ID_with_HI(nxn[[i]], Dep[[i]])
})
```

Get an average SRI for each category of pairing

Step 1: Create a matrix for each category of HI status

```r
# Lists of non-HI versus HI status matrices for each period
is_NB <- is_B <- list()
for (i in seq_along(Beg_nxn)) {
  is_NB[[i]] <- rownames(Beg_nxn[[i]]) == "NB"
  is_B[[i]] <- rownames(Beg_nxn[[i]]) == "B"
}

is_NP <- is_P <- list()
for (i in seq_along(Pat_nxn)) {
  is_NP[[i]] <- rownames(Pat_nxn[[i]]) == "NP"
  is_P[[i]] <- rownames(Pat_nxn[[i]]) == "P"
}

is_ND <- is_D <- list()
for (i in seq_along(Dep_nxn)) {
  is_ND[[i]] <- rownames(Dep_nxn[[i]]) == "ND"
  is_D[[i]] <- rownames(Dep_nxn[[i]]) == "D"
}
```

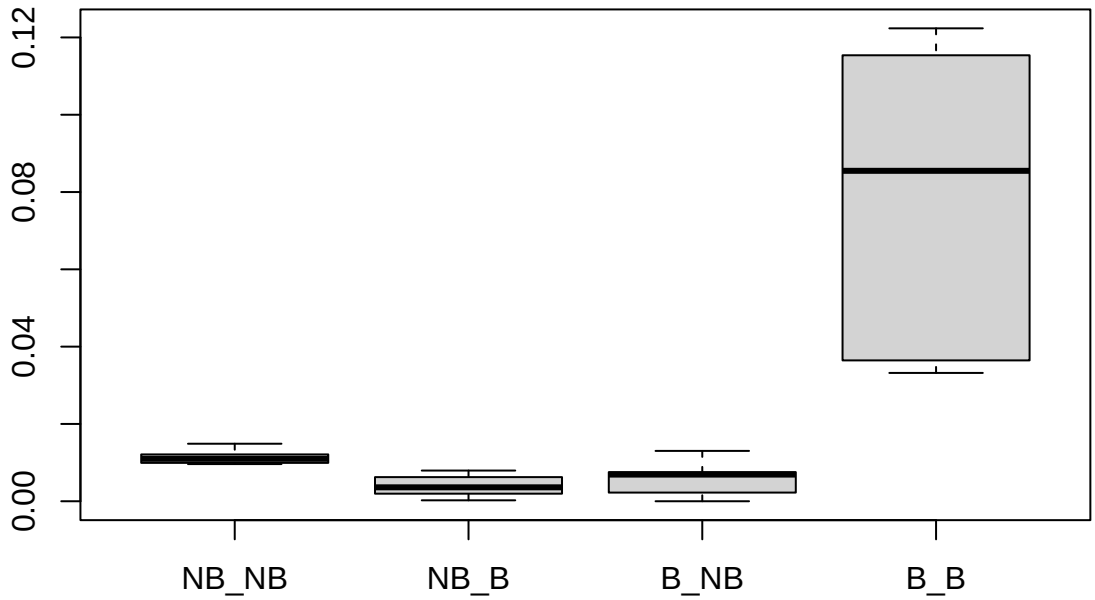
Step 2: Extract the HI status combinations

```r
# Function to extract combinations
extract_combs <- function(HI_nxn, is_row, is_col) {
  combs <- lapply(seq_along(HI_nxn), function(i) {
    HI_nxn[[i]][is_row[[i]], is_col[[i]]]
  })
  return(combs)
}

# Apply for each stat comb
NB_NB <- extract_combs(Beg_nxn, is_NB, is_NB)
NB_B <- extract_combs(Beg_nxn, is_NB, is_B)
B_NB <- extract_combs(Beg_nxn, is_B, is_NB)
```

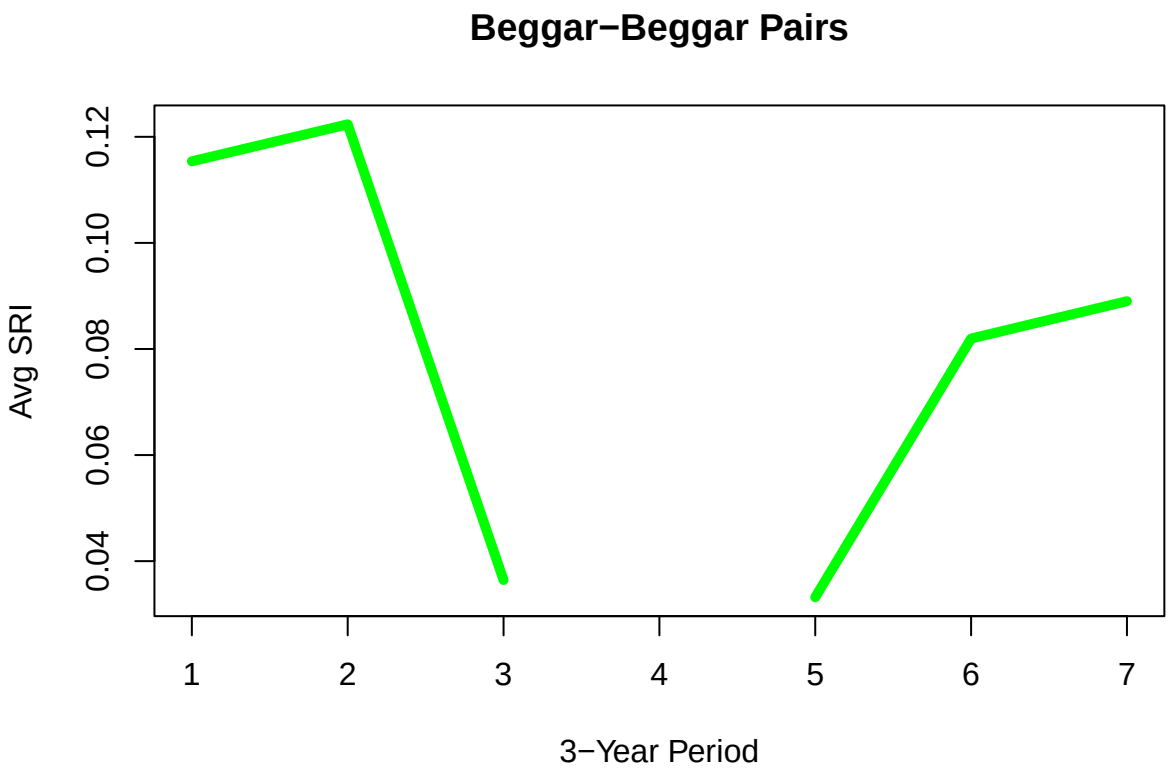
```r
B_B <- extract_combs(Beg_nxn, is_B, is_B)

NP_NP <- extract_combs(Pat_nxn, is_NP, is_NP)
NP_P <- extract_combs(Pat_nxn, is_NP, is_P)
P_NP <- extract_combs(Pat_nxn, is_P, is_NP)
P_P <- extract_combs(Pat_nxn, is_P, is_P)

ND_ND <- extract_combs(Dep_nxn, is_ND, is_ND)
ND_D <- extract_combs(Dep_nxn, is_ND, is_D)
D_ND <- extract_combs(Dep_nxn, is_D, is_ND)
D_D <- extract_combs(Dep_nxn, is_D, is_D)
```

Step 3: Calculate the average of non-diagonal elements in the pairing sub-matrices

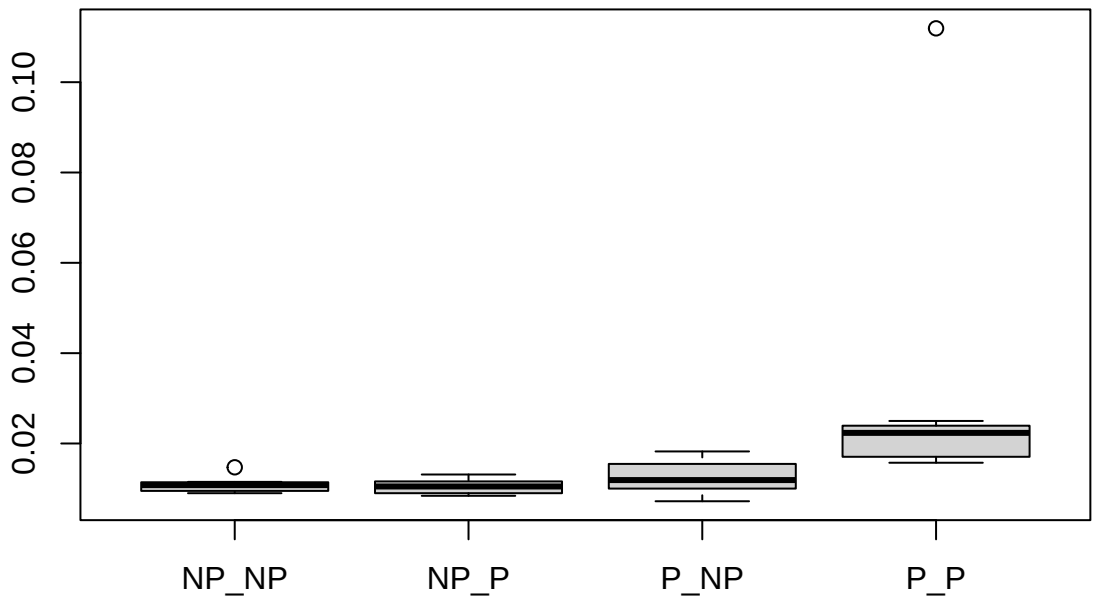
```r
### Function to calculate avg
avg_comb <- function(a, b, c, d) {
  avg_a <- lapply(seq_along(a), function(i) {
    mean(a[[i]][lower.tri(a[[i]])])
  })
  avg_b <- lapply(seq_along(b), function(i) {
    mean(b[[i]][lower.tri(b[[i]])])
  })
  avg_c <- lapply(seq_along(c), function(i) {
    mean(c[[i]][lower.tri(c[[i]])])
  })
  avg_d <- lapply(seq_along(d), function(i) {
    mean(d[[i]][lower.tri(d[[i]])])
  })
  avg_df <- data.frame(
    Avg_A = unlist(avg_a),
    Avg_B = unlist(avg_b),
    Avg_C = unlist(avg_c),
    Avg_D = unlist(avg_d)
  )
  return(avg_df)
}


avg_Beg <- avg_comb(NB_NB, NB_B, B_NB, B_B)
colnames(avg_Beg) <- c("NB_NB", "NB_B", "B_NB", "B_B") # Only one beggar in period 4 (2002-2004)
boxplot(avg_Beg)
```



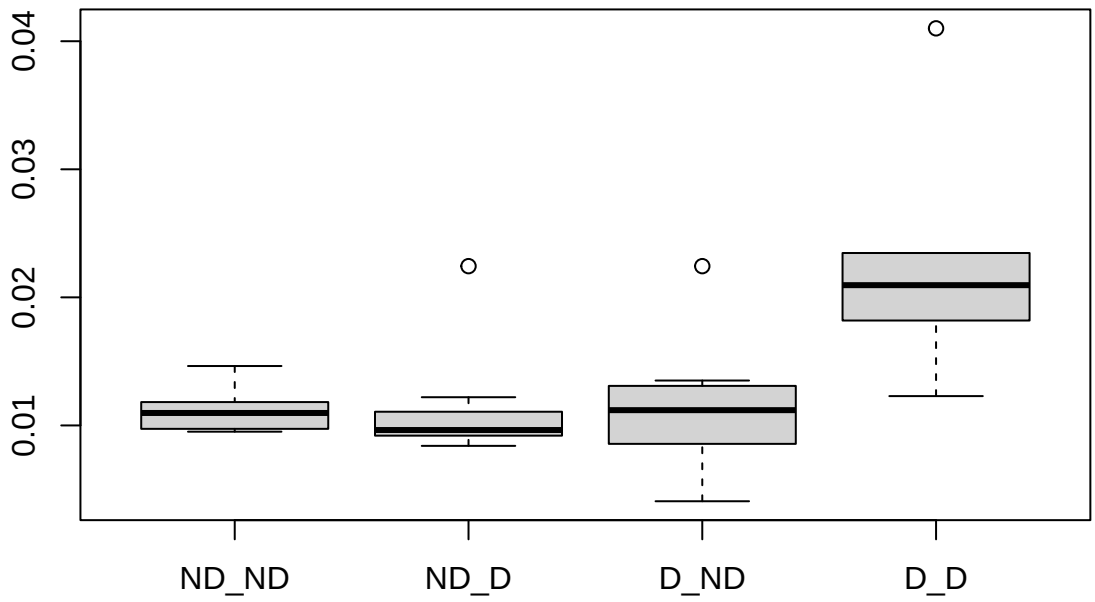

```
plot(avg_Beg[,'B_B'], type="l", col="green", lwd=5,
     xlab="3-Year Period", ylab="Avg SRI", main = "Beggar-Beggar Pairs")
```




```
avg_Pat <- avg_comb(NP_NP, NP_P, P_NP, P_P)
colnames(avg_Pat) <- c("NP_NP", "NP_P", "P_NP", "P_P")
boxplot(avg_Pat)
```




```
avg_Dep <- avg_comb(ND_ND, ND_D, D_ND, D_D)
colnames(avg_Dep) <- c("ND_ND", "ND_D", "D_ND", "D_D") # Only one depredation in period 4 (2002-2004)
boxplot(avg_Dep)
```


0.01
0.02
0.03
0.04
ND_ND
ND_D
D_ND
D_D