Load necessary packages

```
source('GraphBuilder.R')
library(ggplot2)
library(NetworkDistance)
library(sna)
library(xlsx)
library(RSiena)
library(data.table)
theme_set(theme_classic(base_size = 10))
```

Make directed student-level adjacency matrices

```
g <- boris.to.adjacency(file1 = 'C:/Users/Cole/Box Sync/Network analysis/Exported_Data/P1112_Fall2019/Sc
                        nvid1 = 3,
                        file2 = 'C:/Users/Cole/Box Sync/Network analysis/Exported_Data/P1112_Fall2019/Sc
                        nvid2 = 6, directed = TRUE,
                        filename = 'Directed/Scan-directed_9_11.csv')

g <- boris.to.adjacency(file1 = 'C:/Users/Cole/Box Sync/Network analysis/Exported_Data/P1112_Fall2019/Sc
                        nvid1 = 5,
                        file2 = 'C:/Users/Cole/Box Sync/Network analysis/Exported_Data/P1112_Fall2019/Sc
                        nvid2 = 7, directed = TRUE,
                        filename = 'Directed/Scan-directed_9_18.csv')

g <- boris.to.adjacency('C:/Users/Cole/Box Sync/Network analysis/Exported_Data/P1112_Fall2019/Scan_Meth
                        nvid1 = 1, directed = TRUE,
                        filename = 'Directed/Scan-directed_9_25.csv')

g <- boris.to.adjacency(file1 = 'C:/Users/Cole/Box Sync/Network analysis/Exported_Data/P1112_Fall2019/Sc
                        nvid1 = 6,
                        file2 = 'C:/Users/Cole/Box Sync/Network analysis/Exported_Data/P1112_Fall2019/Sc
                        nvid2 = 6, offset2 = 164, directed = TRUE,
                        filename = 'Directed/Scan-directed_10_9.csv')

g <- boris.to.adjacency('C:/Users/Cole/Box Sync/Network analysis/Exported_Data/P1112_Fall2019/Scan_Meth
                        nvid1 = 9, directed = TRUE,
                        filename = 'Directed/Scan-directed_10_23.csv')

g <- boris.to.adjacency(file1 = 'C:/Users/Cole/Box Sync/Network analysis/Exported_Data/P1112_Fall2019/Sc
                        nvid1 = 6,
                        file2 = 'C:/Users/Cole/Box Sync/Network analysis/Exported_Data/P1112_Fall2019/Sc
                        nvid2 = 5, offset2 = 116, directed = TRUE,
                        filename = 'Directed/Scan-directed_10_30.csv')

g <- boris.to.adjacency(file1 = 'C:/Users/Cole/Box Sync/Network analysis/Exported_Data/P1112_Fall2019/Sc
                        nvid1 = 6,
                        file2 = 'C:/Users/Cole/Box Sync/Network analysis/Exported_Data/P1112_Fall2019/Sc
                        nvid2 = 5, offset2 = 154, directed = TRUE,
                        filename = 'Directed/Scan-directed_11_13.csv')
```

```
g <- boris.to.adjacency(file1 = 'C:/Users/Cole/Box Sync/Network analysis/Exported_Data/P1112_Fall2019/S
                        nvid1 = 6,
                        file2 = 'C:/Users/Cole/Box Sync/Network analysis/Exported_Data/P1112_Fall2019/S
                        nvid2 = 5, offset2 = 94, directed = TRUE,
                        filename = 'Directed/Scan-directed_11_20.csv')
```

Create graphs from adjacencies

```
g1 <- graph.from.adjacency('C:/Users/Cole/Box Sync/Network analysis/Adjacency_Matrices/P1112_Fall2019/S
                           'scan-student', directed = TRUE, name = 'Time 1')

g2 <- graph.from.adjacency('C:/Users/Cole/Box Sync/Network analysis/Adjacency_Matrices/P1112_Fall2019/S
                           'scan-student', directed = TRUE, name = 'Time 2')

g3 <- graph.from.adjacency('C:/Users/Cole/Box Sync/Network analysis/Adjacency_Matrices/P1112_Fall2019/S
                           'scan-student', directed = TRUE, name = 'Time 3')

g4 <- graph.from.adjacency('C:/Users/Cole/Box Sync/Network analysis/Adjacency_Matrices/P1112_Fall2019/S
                           'scan-student', directed = TRUE, name = 'Time 4')

g5 <- graph.from.adjacency('C:/Users/Cole/Box Sync/Network analysis/Adjacency_Matrices/P1112_Fall2019/S
                           'scan-student', directed = TRUE, name = 'Time 5')

g6 <- graph.from.adjacency('C:/Users/Cole/Box Sync/Network analysis/Adjacency_Matrices/P1112_Fall2019/S
                           'scan-student', directed = TRUE, name = 'Time 6')

g7 <- graph.from.adjacency('C:/Users/Cole/Box Sync/Network analysis/Adjacency_Matrices/P1112_Fall2019/S
                           'scan-student', directed = TRUE, name = 'Time 7')

g8 <- graph.from.adjacency('C:/Users/Cole/Box Sync/Network analysis/Adjacency_Matrices/P1112_Fall2019/S
                           'scan-student', directed = TRUE, name = 'Time 8')
```

Assign student names to vertices

```
names.df <- read.xlsx2('C:/Users/Cole/Box Sync/Network analysis/Student_RostersCodes/P1112_Fall2019_Stu
                       sheetName = '408 - Scan', stringsAsFactors = FALSE) %>%
  select(-Photo) %>%
  rowwise() %>%
  mutate(Name = strsplit(Name, ',')[[1]][1]) %>%
  `colnames<-`(c('Name', 'Time.1', 'Time.2', 'Time.3', 'Time.4', 'Time.5',
                 'Time.6', 'Time.7', 'Time.8')) %>%
  filter(!is.na(Name))

students <- names.df[['Name']]

names.df[names.df == '(not present)'] <- NA_character_
```

```
set.student.names <- function(g, time){
  # convert vertex names for a given graph to student names
  student.ids <- names.df[[paste('Time', time, sep = '.')]]
  g <- induced_subgraph(g, V(g)$name %in% student.ids)
  v <- names.df[!is.na(student.ids),
                ][order(student.ids[!is.na(student.ids)]),]$Name
  v <- v[!is.na(v)]
  V(g)$name <- v

  absentees <- students[!(students %in% V(g)$name)]
  for(student in absentees){
    g <- add_vertices(g, 1, attr = list(name = student,
                                        group = as.numeric(max(V(g)$group)) + 1,
                                        centrality.total = 0,
                                        strength.total = 0,
                                        centrality.in = 0,
                                        strength.in = 0,
                                        centrality.out = 0,
                                        strength.out = 0))
  }
  g <- igraph::permute(g, match(V(g)$name, students))
  return(g)
}

g1 <- set.student.names(g1, 1)
g2 <- set.student.names(g2, 2)
g3 <- set.student.names(g3, 3)
g4 <- set.student.names(g4, 4)
g5 <- set.student.names(g5, 5)
g6 <- set.student.names(g6, 6)
g7 <- set.student.names(g7, 7)
g8 <- set.student.names(g8, 8)
```

Evaluate whole networks

Create SIENA object

```
time.networks <- array(c(as_adjacency_matrix(g1, sparse = FALSE),
                         as_adjacency_matrix(g2, sparse = FALSE),
                         as_adjacency_matrix(g3, sparse = FALSE),
                         as_adjacency_matrix(g4, sparse = FALSE),
                         as_adjacency_matrix(g5, sparse = FALSE),
                         as_adjacency_matrix(g6, sparse = FALSE),
                         as_adjacency_matrix(g7, sparse = FALSE),
                         as_adjacency_matrix(g8, sparse = FALSE)),
                       dim = c(length(students), length(students), 8))

interactions <- sienaDependent(time.networks)

names.df$Gender <-  c('W', 'M', 'M', 'M', 'M', 'W', 'M', 'M', 'M', 'M', 'M', 'W',
                      'M', 'W', 'W', 'W', 'W', 'W', 'M')
names.df$Gender <- ifelse(names.df$Gender == 'M', 0, 1)
gender <- coCovar(names.df$Gender)
data <- sienaDataCreate(interactions, gender)
data
```

```
## Dependent variables:  interactions
## Number of observations: 8
##
## Nodeset                  Actors
## Number of nodes              19
##
## Dependent variable interactions
## Type               oneMode
## Observations       8
## Nodeset            Actors
## Densities          0.12 0.28 0.19 0.15 0.2 0.22 0.1 0.14
##
## Constant covariates:  gender
```

Add effects and run algorithm

```
eff <- getEffects(data)
effectsDocumentation(eff)

eff <- includeEffects(eff, recip, inPop, outAct, transRecTrip)
```

```
##   effectName                   include fix   test  initialValue parm
## 1 reciprocity                  TRUE    FALSE FALSE          0   0
## 2 transitive recipr. triplets TRUE    FALSE FALSE          0   0
## 3 indegree - popularity       TRUE    FALSE FALSE          0   0
## 4 outdegree - activity        TRUE    FALSE FALSE          0   0
```

```
eff <- includeEffects(eff, simX, egoX, altX, interaction1 = "gender" )
```

```
##   effectName        include fix   test  initialValue parm
## 1 gender alter      TRUE    FALSE FALSE          0   0
## 2 gender ego        TRUE    FALSE FALSE          0   0
## 3 gender similarity TRUE    FALSE FALSE          0   0
```

```
eff
```

```
##    effectName                            include fix   test  initialValue
## 1  constant interactions rate (period 1) TRUE    FALSE FALSE    7.65539
## 2  constant interactions rate (period 2) TRUE    FALSE FALSE   13.19475
## 3  constant interactions rate (period 3) TRUE    FALSE FALSE    5.66122
## 4  constant interactions rate (period 4) TRUE    FALSE FALSE   10.42507
## 5  constant interactions rate (period 5) TRUE    FALSE FALSE    6.21516
## 6  constant interactions rate (period 6) TRUE    FALSE FALSE    9.09563
## 7  constant interactions rate (period 7) TRUE    FALSE FALSE    2.44840
## 8  outdegree (density)                   TRUE    FALSE FALSE   -0.73569
## 9  reciprocity                           TRUE    FALSE FALSE    0.00000
## 10 transitive recipr. triplets           TRUE    FALSE FALSE    0.00000
## 11 indegree - popularity                 TRUE    FALSE FALSE    0.00000
## 12 outdegree - activity                  TRUE    FALSE FALSE    0.00000
## 13 gender alter                          TRUE    FALSE FALSE    0.00000
## 14 gender ego                            TRUE    FALSE FALSE    0.00000
## 15 gender similarity                     TRUE    FALSE FALSE    0.00000
##    parm
## 1  0
## 2  0
## 3  0
## 4  0
## 5  0
## 6  0
## 7  0
## 8  0
## 9  0
## 10 0
## 11 0
## 12 0
## 13 0
## 14 0
## 15 0
```

```
alg <- sienaAlgorithmCreate(projname = 'F19-P1112-lab-interactions', seed = 11)
```

```
## siena07 will create an output file F19-P1112-lab-interactions.txt .
```

```
result <- siena07(alg, data = data, effects = eff, returnDeps = TRUE)
result
```

```
## Estimates, standard errors and convergence t-ratios
##
##                                         Estimate   Standard   Convergence
##                                                      Error      t-ratio
##
## Rate parameters:
##   0.1      Rate parameter period 1     16.9039  (  5.6993  )
##   0.2      Rate parameter period 2     27.7105  ( 10.0209  )
##   0.3      Rate parameter period 3      5.4642  (  1.1388  )
##   0.4      Rate parameter period 4     46.0053  ( 29.2337  )
##   0.5      Rate parameter period 5      6.1929  (  1.2555  )
##   0.6      Rate parameter period 6     14.9208  (  3.8710  )
##   0.7      Rate parameter period 7      2.0130  (  0.5484  )
##
## Other parameters:
##   1.  eval outdegree (density)         -1.7031  (  0.1478  )    0.0053
##   2.  eval reciprocity                  1.6281  (  0.1390  )   -0.0077
##   3.  eval transitive recipr. triplets  0.0826  (  0.0470  )   -0.0054
##   4.  eval indegree - popularity       -0.0870  (  0.0302  )    0.0004
##   5.  eval outdegree - activity         0.0758  (  0.0076  )   -0.0240
##   6.  eval gender alter                 0.0635  (  0.0970  )    0.0780
##   7.  eval gender ego                  -0.0804  (  0.0790  )    0.0014
##   8.  eval gender similarity            0.0058  (  0.0787  )    0.0062
##
## Overall maximum convergence ratio:    0.1422
##
##
## Total of 2366 iteration steps.
```

```
#sienaGOF(result, IndegreeDistribution, join = TRUE, varName = 'time.networks')
```

Evaluate between-groups network

Reduce whole networks to between-groups network

```
Reduce.graph <- function(g, edge.type){
  # remove within or between-group edges from graph
  g.reduce <- subgraph.edges(g, E(g)[E(g)$group == edge.type],
                              delete.vertices = FALSE)
  return(g.reduce)
}

g1.between  <- Reduce.graph(g1, edge.type = 'between')
g2.between  <- Reduce.graph(g2, edge.type = 'between')
g3.between  <- Reduce.graph(g3, edge.type = 'between')
g4.between  <- Reduce.graph(g4, edge.type = 'between')
g5.between  <- Reduce.graph(g5, edge.type = 'between')
g6.between  <- Reduce.graph(g6, edge.type = 'between')
g7.between  <- Reduce.graph(g7, edge.type = 'between')
g8.between  <- Reduce.graph(g8, edge.type = 'between')
```

Create SIENA object

```
time.between.networks <- array(c(as_adjacency_matrix(g1.between, sparse = FALSE),
                                 as_adjacency_matrix(g2.between, sparse = FALSE),
                                 as_adjacency_matrix(g3.between, sparse = FALSE),
                                 as_adjacency_matrix(g4.between, sparse = FALSE),
                                 as_adjacency_matrix(g5.between, sparse = FALSE),
                                 as_adjacency_matrix(g6.between, sparse = FALSE),
                                 as_adjacency_matrix(g7.between, sparse = FALSE),
                                 as_adjacency_matrix(g8.between, sparse = FALSE)),
                               dim = c(length(students), length(students), 8))

between.interactions <- sienaDependent(time.between.networks)
between.data <- sienaDataCreate(between.interactions, gender)
between.data
```

```
## Dependent variables:  between.interactions
## Number of observations: 8
##
## Nodeset                  Actors
## Number of nodes              19
##
## Dependent variable between.interactions
## Type               oneMode
## Observations       8
## Nodeset            Actors
## Densities          0.018 0.18 0.11 0.079 0.096 0.12 0.015 0.038
##
## Constant covariates:  gender
```

Add effects and run algorithm

```
eff <- getEffects(between.data)
effectsDocumentation(eff)

eff <- includeEffects(eff, recip)
eff

alg <- sienaAlgorithmCreate(projname = 'F19-P1112-lab-interactions', seed = 11)
result <- siena07(alg, data = between.data, effects = eff, returnDeps = TRUE)
result

#sienaGOF(result, IndegreeDistribution, join = TRUE, varName = 'time.networks')
```

Which groups do students visit in subsequent weeks?

Get obsered interactions between groups

Construct between-groups interactions data.frame for each student for each lab

```
create.edglist <- function(g, edge.type, suffix, vertices = FALSE){
  g.reduce <- Reduce.graph(g, edge.type = edge.type)
  df <- igraph::as_data_frame(g.reduce, what = 'edges')[, c('from', 'to')]
  df[[paste(edge.type, suffix, sep = '.')]] <- 1
  if(vertices){
    df.vertices <- igraph::as_data_frame(g.reduce,
                                         what = 'vertices')[, c('name',
                                                                'group')] %>%
      `colnames<-`(c('name', paste('group', suffix, sep = '.')))
    df <- left_join(df, df.vertices, by = c('to' = 'name'))
  }
  return(df)
}

within.1 <- create.edglist(g1, 'within', 1)
within.2 <- create.edglist(g2, 'within', 2)
```

```r
within.3 <- create.edglist(g3, 'within', 3)
within.4 <- create.edglist(g4, 'within', 4)
within.5 <- create.edglist(g5, 'within', 5)
within.6 <- create.edglist(g6, 'within', 6)
within.7 <- create.edglist(g7, 'within', 7)
within.8 <- create.edglist(g8, 'within', 8)

within.df <- Reduce(function(x,y) full_join(x = x, y = y, by = c('from', 'to')),
                    list(within.1, within.2, within.3, within.4, within.5,
                         within.6, within.7, within.8))

between.1 <- create.edglist(g1, 'between', 1, vertices = TRUE)
between.2 <- create.edglist(g2, 'between', 2, vertices = TRUE)
between.3 <- create.edglist(g3, 'between', 3, vertices = TRUE)
between.4 <- create.edglist(g4, 'between', 4, vertices = TRUE)
between.5 <- create.edglist(g5, 'between', 5, vertices = TRUE)
between.6 <- create.edglist(g6, 'between', 6, vertices = TRUE)
between.7 <- create.edglist(g7, 'between', 7, vertices = TRUE)
between.8 <- create.edglist(g8, 'between', 8, vertices = TRUE)

between.df <- Reduce(function(x,y) full_join(x = x, y = y, by = c('from', 'to')),
                     list(between.1, between.2, between.3, between.4, between.5,
                          between.6, between.7, between.8))

full.df <- left_join(between.df, within.df, by = c('from', 'to'))
```

Construct data.frame of obesrved interactions

```r
df.3 <- full.df %>%
  filter(!is.na(between.3)) %>%
  group_by(from, group.3) %>%
  summarize(first.lab = 1 * (sum(within.1, na.rm = TRUE) +
              sum(within.2, na.rm = TRUE) > 0),
            previous.lab = 1 * (sum(within.1, na.rm = TRUE) +
              sum(within.2, na.rm = TRUE) > 0),
            all.labs = 1 * (sum(within.1, na.rm = TRUE) +
              sum(within.2, na.rm = TRUE) > 0)) %>%
  select(-group.3) %>%
  mutate(lab = 3)

df.4 <- full.df %>%
  filter(!is.na(between.4)) %>%
  group_by(from, group.4) %>%
  summarize(first.lab = 1 * (sum(within.1, na.rm = TRUE) +
              sum(within.2, na.rm = TRUE) > 0),
            previous.lab = 1 * (sum(within.1, na.rm = TRUE) +
              sum(within.2, na.rm = TRUE) > 0),
            all.labs = 1 * (sum(within.1, na.rm = TRUE) +
              sum(within.2, na.rm = TRUE) > 0)) %>%
  select(-group.4) %>%
  mutate(lab = 4)
```

```
df.5 <- full.df %>%
  filter(!is.na(between.5)) %>%
  group_by(from, group.5) %>%
  summarize(first.lab = 1 * (sum(within.1, na.rm = TRUE) +
              sum(within.2, na.rm = TRUE) > 0),
            previous.lab = 1 * (sum(within.3, na.rm = TRUE) +
              sum(within.4, na.rm = TRUE) > 0),
            all.labs = 1 * (sum(within.1, na.rm = TRUE) +
              sum(within.2, na.rm = TRUE) + sum(within.3, na.rm = TRUE) +
                sum(within.4, na.rm = TRUE) > 0)) %>%
  select(-group.5) %>%
  mutate(lab = 5)

df.6 <- full.df %>%
  filter(!is.na(between.6)) %>%
  group_by(from, group.6) %>%
  summarize(first.lab = 1 * (sum(within.1, na.rm = TRUE) +
              sum(within.2, na.rm = TRUE) > 0),
            previous.lab = 1 * (sum(within.3, na.rm = TRUE) +
              sum(within.4, na.rm = TRUE) > 0),
            all.labs = 1 * (sum(within.1, na.rm = TRUE) +
              sum(within.2, na.rm = TRUE) + sum(within.3, na.rm = TRUE) +
                sum(within.4, na.rm = TRUE) > 0)) %>%
  select(-group.6) %>%
  mutate(lab = 6)

df.7 <- full.df %>%
  filter(!is.na(between.7)) %>%
  group_by(from, group.7) %>%
  summarize(first.lab = 1 * (sum(within.1, na.rm = TRUE) +
              sum(within.2, na.rm = TRUE) > 0),
            previous.lab = 1 * (sum(within.5, na.rm = TRUE) +
              sum(within.6, na.rm = TRUE) > 0),
            all.labs = 1 * (sum(within.1, na.rm = TRUE) +
              sum(within.2, na.rm = TRUE) + sum(within.3, na.rm = TRUE) +
                sum(within.4, na.rm = TRUE) + sum(within.5, na.rm = TRUE) +
                sum(within.6, na.rm = TRUE) > 0)) %>%
  select(-group.7) %>%
  mutate(lab = 7)

df.8 <- full.df %>%
  filter(!is.na(between.8)) %>%
  group_by(from, group.8) %>%
  summarize(first.lab = 1 * (sum(within.1, na.rm = TRUE) +
              sum(within.2, na.rm = TRUE) > 0),
            previous.lab = 1 * (sum(within.5, na.rm = TRUE) +
              sum(within.6, na.rm = TRUE) > 0),
            all.labs = 1 * (sum(within.1, na.rm = TRUE) +
              sum(within.2, na.rm = TRUE) + sum(within.3, na.rm = TRUE) +
                sum(within.4, na.rm = TRUE) + sum(within.5, na.rm = TRUE) +
                sum(within.6, na.rm = TRUE) > 0)) %>%
  select(-group.8) %>%
  mutate(lab = 8)
```

```
observed.df <- bind_rows(df.3, df.4, df.5, df.6, df.7, df.8)
```

Get probabilities of interactions occuring at random

Construct baseline interactions data.frame for each student for each lab

```
baseline.df <- expand.grid(V(g1.between)$name, V(g1.between)$name)
colnames(baseline.df) <- c('from', 'to')

Get.groups <- function(g, suffix){
  vertices <- igraph::as_data_frame(g, what = 'vertices')[, c('name', 'group')]

  df <- left_join(baseline.df, vertices, by = c('from' = 'name'))
  df <- left_join(df, vertices, by = c('to' = 'name'), suffix = c('.from', '.to'))

  colnames(df) <- c('from', 'to', paste('group.from', suffix, sep = '.'),
                    paste('group.to', suffix, sep = '.'))

  # absent students are in a group alone...change their group to NA
  dt <- data.table(df)[, N := .N, c('from', paste('group.to', suffix, sep = '.'))]
  dt[[paste('group.to', suffix,
          sep = '.')]] <- ifelse(dt$N > 1, dt[[paste('group.to', suffix,
                                                  sep = '.')]],
                                 NA_character_)
  return(dt)
}

baseline.df <- Reduce(function(x,y) left_join(x = x, y = y, by = c('from', 'to')),
                      list(Get.groups(g1, 1), Get.groups(g2, 2), Get.groups(g3, 3),
                           Get.groups(g4, 4), Get.groups(g5, 5), Get.groups(g6, 6),
                           Get.groups(g7, 7), Get.groups(g8, 8)))
```

```
## Warning: Column `from`/`name` joining factor and character vector, coercing
## into character vector

## Warning: Column `to`/`name` joining factor and character vector, coercing
## into character vector

## Warning: Column `from`/`name` joining factor and character vector, coercing
## into character vector

## Warning: Column `to`/`name` joining factor and character vector, coercing
## into character vector

## Warning: Column `from`/`name` joining factor and character vector, coercing
## into character vector

## Warning: Column `to`/`name` joining factor and character vector, coercing
## into character vector
## Warning: Column `from`/`name` joining factor and character vector, coercing
## into character vector

## Warning: Column `to`/`name` joining factor and character vector, coercing
## into character vector

## Warning: Column `from`/`name` joining factor and character vector, coercing
## into character vector

## Warning: Column `to`/`name` joining factor and character vector, coercing
## into character vector

## Warning: Column `from`/`name` joining factor and character vector, coercing
## into character vector

## Warning: Column `to`/`name` joining factor and character vector, coercing
## into character vector

## Warning: Column `from`/`name` joining factor and character vector, coercing
## into character vector

## Warning: Column `to`/`name` joining factor and character vector, coercing
## into character vector

## Warning: Column `from`/`name` joining factor and character vector, coercing
## into character vector

## Warning: Column `to`/`name` joining factor and character vector, coercing
## into character vector
```

Construct data.frame of probabilities of interactions with previous group members

```
df.3 <- baseline.df %>%
  filter(group.from.3 != group.to.3) %>%
  group_by(from, group.to.3) %>%
  summarize(first.lab = 1 * ((sum(group.from.1 == group.to.1, na.rm = TRUE) +
                               sum(group.from.2 == group.to.2, na.rm = TRUE)) >
                              0),
            previous.lab = 1 * ((sum(group.from.1 == group.to.1, na.rm = TRUE) +
                                  sum(group.from.2 == group.to.2,
                                      na.rm = TRUE)) > 0),
            all.labs = 1 * ((sum(group.from.1 == group.to.1, na.rm = TRUE) +
                              sum(group.from.2 == group.to.2, na.rm = TRUE)) >
                              0)) %>%
  select(-group.to.3) %>%
  mutate(lab = 3)

df.4 <- baseline.df %>%
  filter(group.from.4 != group.to.4) %>%
  group_by(from, group.to.4) %>%
  summarize(first.lab = 1 * ((sum(group.from.1 == group.to.1, na.rm = TRUE) +
                                sum(group.from.2 == group.to.2, na.rm = TRUE)) >
                               0),
            previous.lab = 1 * ((sum(group.from.1 == group.to.1, na.rm = TRUE) +
                                   sum(group.from.2 == group.to.2,
                                       na.rm = TRUE)) > 0),
            all.labs = 1 * ((sum(group.from.1 == group.to.1, na.rm = TRUE) +
                               sum(group.from.2 == group.to.2, na.rm = TRUE)) >
                               0)) %>%
  select(-group.to.4) %>%
  mutate(lab = 4)

df.5 <- baseline.df %>%
  filter(group.from.5 != group.to.5) %>%
  group_by(from, group.to.5) %>%
  summarize(first.lab = 1 * ((sum(group.from.1 == group.to.1, na.rm = TRUE) +
                                sum(group.from.2 == group.to.2, na.rm = TRUE)) >
                               0),
            previous.lab = 1 * ((sum(group.from.3 == group.to.3, na.rm = TRUE) +
                                   sum(group.from.4 == group.to.4,
                                       na.rm = TRUE)) > 0),
            all.labs = 1 * ((sum(group.from.1 == group.to.1, na.rm = TRUE) +
                               sum(group.from.2 == group.to.2, na.rm = TRUE) +
                               sum(group.from.3 == group.to.3, na.rm = TRUE) +
                               sum(group.from.4 == group.to.4, na.rm = TRUE)) >
                               0)) %>%
  select(-group.to.5) %>%
  mutate(lab = 5)

df.6 <- baseline.df %>%
  filter(group.from.6 != group.to.6) %>%
  group_by(from, group.to.6) %>%
  summarize(first.lab = 1 * ((sum(group.from.1 == group.to.1, na.rm = TRUE) +
                                sum(group.from.2 == group.to.2, na.rm = TRUE)) >
                               0),
            previous.lab = 1 * ((sum(group.from.3 == group.to.3, na.rm = TRUE) +
                                   sum(group.from.4 == group.to.4,
                                       na.rm = TRUE)) > 0),
            all.labs = 1 * ((sum(group.from.1 == group.to.1, na.rm = TRUE) +
                               sum(group.from.2 == group.to.2, na.rm = TRUE) +
                               sum(group.from.3 == group.to.3, na.rm = TRUE) +
                               sum(group.from.4 == group.to.4, na.rm = TRUE)) >
                               0)) %>%
  select(-group.to.6) %>%
  mutate(lab = 6)

df.7 <- baseline.df %>%
  filter(group.from.7 != group.to.7) %>%
  group_by(from, group.to.7) %>%
  summarize(first.lab = 1 * ((sum(group.from.1 == group.to.1, na.rm = TRUE) +
                                sum(group.from.2 == group.to.2, na.rm = TRUE)) >
                               0),
            previous.lab = 1 * ((sum(group.from.5 == group.to.5, na.rm = TRUE) +
                                   sum(group.from.6 == group.to.6,
                                       na.rm = TRUE)) > 0),
            all.labs = 1 * ((sum(group.from.1 == group.to.1, na.rm = TRUE) +
                               sum(group.from.2 == group.to.2, na.rm = TRUE) +
                               sum(group.from.3 == group.to.3, na.rm = TRUE) +
                               sum(group.from.4 == group.to.4, na.rm = TRUE) +
                               sum(group.from.5 == group.to.5, na.rm = TRUE) +
                               sum(group.from.6 == group.to.6, na.rm = TRUE)) >
                               0)) %>%
  select(-group.to.7) %>%
  mutate(lab = 7)

df.8 <- baseline.df %>%
  filter(group.from.8 != group.to.8) %>%
  group_by(from, group.to.8) %>%
  summarize(first.lab = 1 * ((sum(group.from.1 == group.to.1, na.rm = TRUE) +
                                sum(group.from.2 == group.to.2, na.rm = TRUE)) >
                               0),
            previous.lab = 1 * ((sum(group.from.5 == group.to.5, na.rm = TRUE) +
                                   sum(group.from.6 == group.to.6,
                                       na.rm = TRUE)) > 0),
            all.labs = 1 * ((sum(group.from.1 == group.to.1, na.rm = TRUE) +
                               sum(group.from.2 == group.to.2, na.rm = TRUE) +
                               sum(group.from.3 == group.to.3, na.rm = TRUE) +
                               sum(group.from.4 == group.to.4, na.rm = TRUE) +
                               sum(group.from.5 == group.to.5, na.rm = TRUE) +
                               sum(group.from.6 == group.to.6, na.rm = TRUE)) >
                               0)) %>%
  select(-group.to.8) %>%
  mutate(lab = 8)

probability.df <- bind_rows(df.3, df.4, df.5, df.6, df.7, df.8) %>%
  group_by(from, lab) %>%
  summarize_all(mean)
```

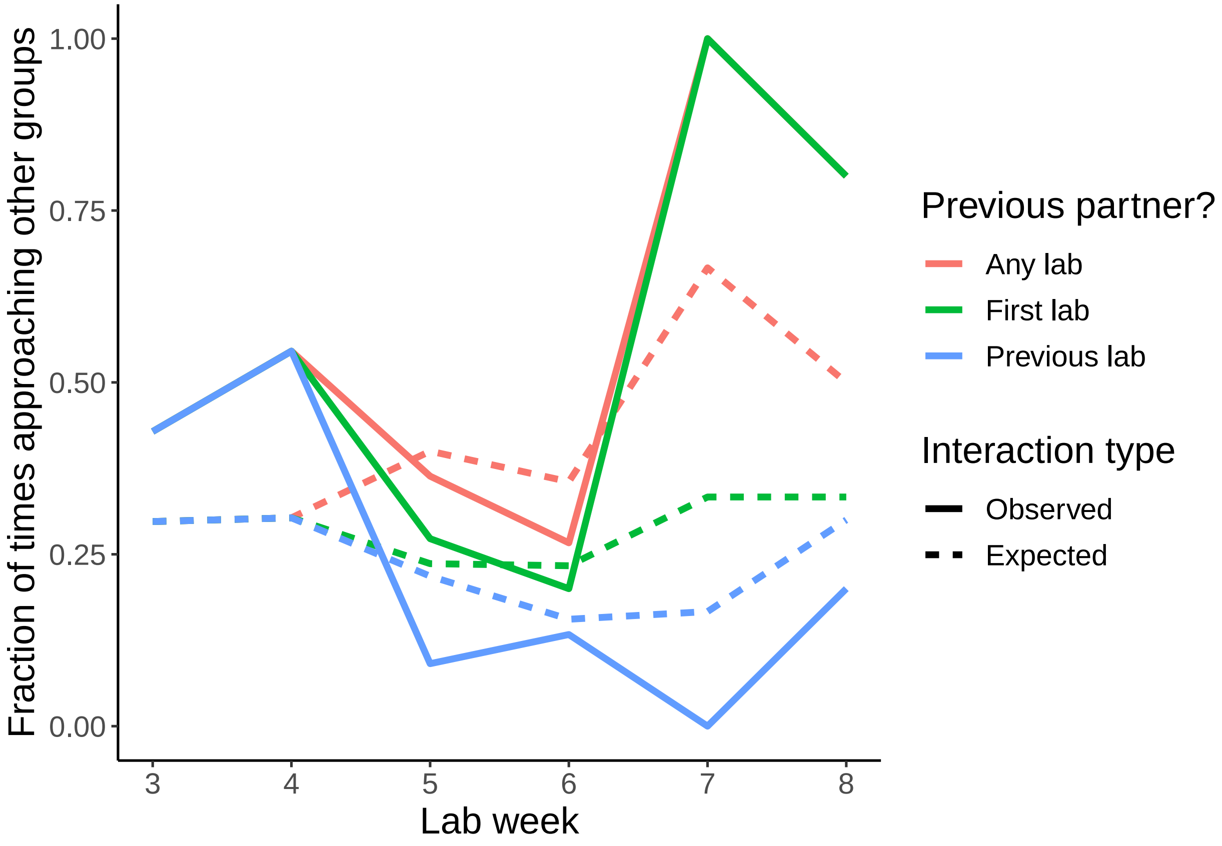
Combine observed and probability data.frames

```
observed.probability.df <- left_join(observed.df, probability.df,
                                     by = c('from', 'lab'),
                                     suffix = c('.observed', '.probability'))

melted.df <- observed.probability.df %>%
  ungroup() %>%
  select(-from) %>%
  group_by(lab) %>%
  summarize_all(mean) %>%
  melt(id.vars = 'lab', value.name = 'fraction.interactions') %>%
  mutate(variable = as.character(variable)) %>%
  rowwise() %>%
  mutate(type = strsplit(variable, '.', fixed = TRUE)[[1]][3],
         which = strsplit(variable, '.', fixed = TRUE)[[1]][1]) %>%
  select(-variable)
```

Plot results

```
ggplot(melted.df, aes(x = lab, y = fraction.interactions, color = which,
                      linetype = type)) +
  geom_line(size = 1.2) +
  theme(text = element_text(size = 14)) +
  labs(x = 'Lab week', y = 'Fraction of times approaching other groups',
       color = 'Previous partner?', linetype = 'Interaction type') +
  scale_color_discrete(labels = c('Any lab', 'First lab', 'Previous lab')) +
  scale_linetype_discrete(labels = c('Observed', 'Expected'))
```


Fraction of times approaching other groups
Lab week
Previous partner?
Any lab
First lab
Previous lab
Interaction type
Observed
Expected