

Supplemental online material

Contents:

1. Supplemental Figures

Figure S1. Measuring metabolic rates and behavior in seed harvester ant colonies.
Figure S2. Antennation-based social network construction and spatial trajectories for a colony of *P. californicus* with 239 workers.
Figure S3. Social network construction and spatial trajectories for a simulated colony of 239 randomly moving virtual individuals.
Figure S4. Proximity-based social network construction and spatial trajectories for a colony of *P. californicus* with 239 workers.
Figure S5. Effects of the experimental colony size reduction on network properties.
Figure S6. Scaling relationships in *P. californicus* interaction networks.
Figure S7. Scaling of network attributes identified by each of the two experimental states.
Figure S8. Comparing the scaling of social insect networks constructed using different methods for identifying interactions.

2. Summary data tables

Table S1. Colony demographic characteristics and composition
Table S2. Network structure metrics
Table S3. Network scaling coefficients

3. Source network data for *P. californicus* colonies

Edge lists identifying each interaction in each of the 10 empirical networks
All results from the simulation model (coordinates and interactions)
Available at: <https://github.com/waterslab/ant-networks>

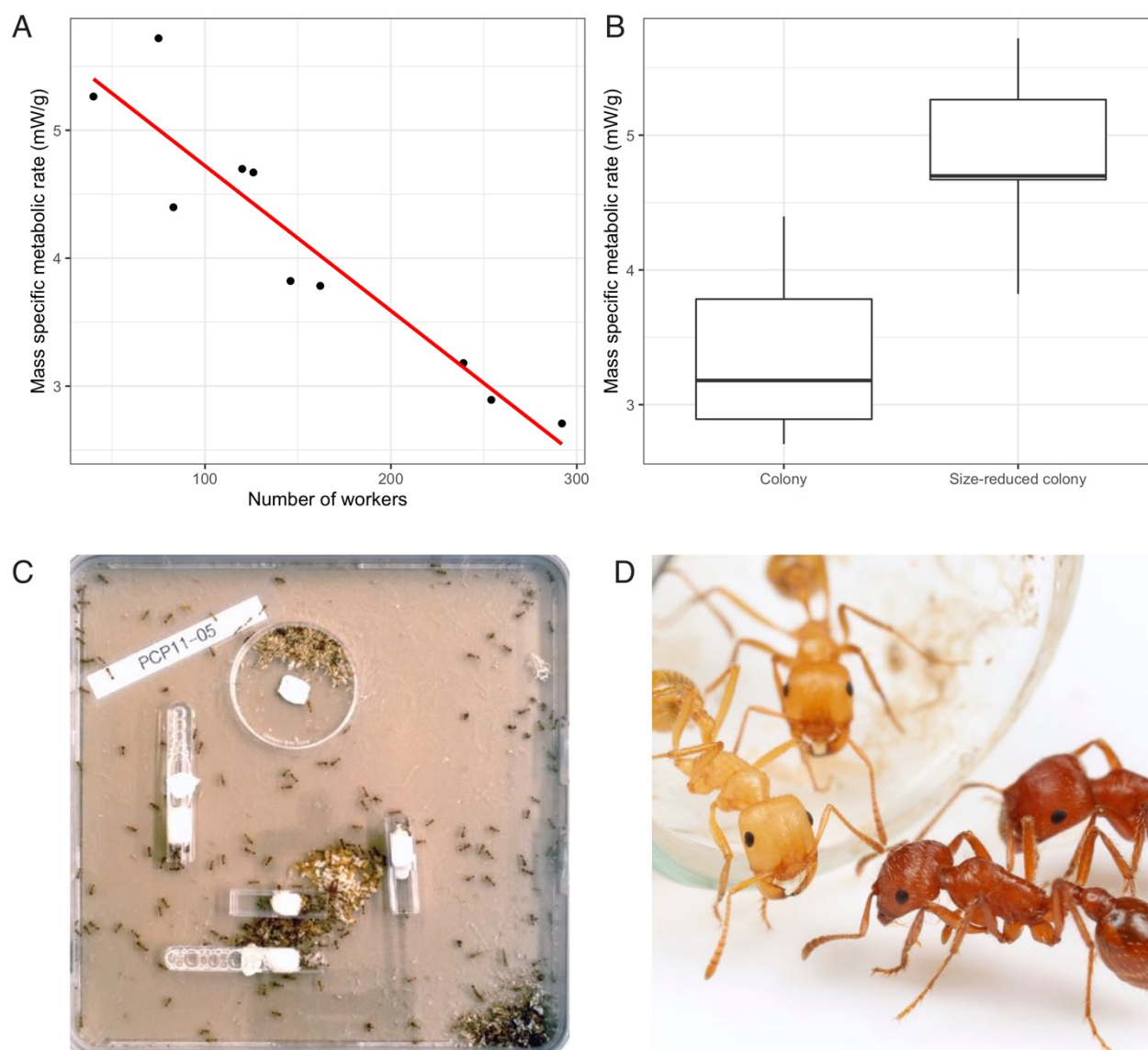
4. Network visualization animation

Movies illustrating the spatial trajectories and social network progression over time based on real-world observations and the simulated model, uploaded separately
Also available at: <https://github.com/waterslab/ant-networks>

5. R code

- a. AntSim-01-function_v2.R
This file includes the main function to simulate interactions
- b. AntSim-02-simulation-loop.R
This file includes the loop to simulate many networks
- c. AntSim-03a-analysis_network-graphs.R
Plotting networks
- d. AntSim-03b-analysis_pcal-networks.R
Analysis of our real-world network data
- e. AntSim-03c-analysis_simulated-networks.R
Analysis of the simulated networks
- f. AntSim-04-network-and-spatial-animation.R
Analysis of the simulated networks
- g. AntSim-05-trajectory-based-proximity-networks.R
Inferring interactions based on spatial proximity

47



48

49 **Figure S1.** California seed-harvester ants, *Pogonomyrmex californicus*, are a model system to
 50 investigate how behavior and metabolism scale with colony size. (A) As previously reported [30],
 51 these colonies exhibit a metabolic allometry in which mass-specific metabolic rates decrease with
 52 increasing colony size. (B) Following an experimental manipulation that reduced the size of colonies
 53 by 50%, the mass-specific metabolic rates of the remaining individuals increased, indicating support
 54 for colony size as a driver for the metabolic allometry. These graphs (A-B) summarize the effects of
 55 colony size on the mass-specific metabolic rate of colonies including only the data for the colonies
 56 whose social networks were analyzed in this study. (C) This image is a frame of the video recording
 57 of one of the focal colonies while it is inside its artificial nest enclosure and also simultaneously
 58 inside a respirometry chamber. Also visible in the enclosure are water tubes, food within a small petri
 59 dish, a central pile with the queens, larvae, and pupae, and a refuse pile in the corner. (D) This
 60 photograph shows four *P. californicus* sisters interacting with each other by direct antennal contact.

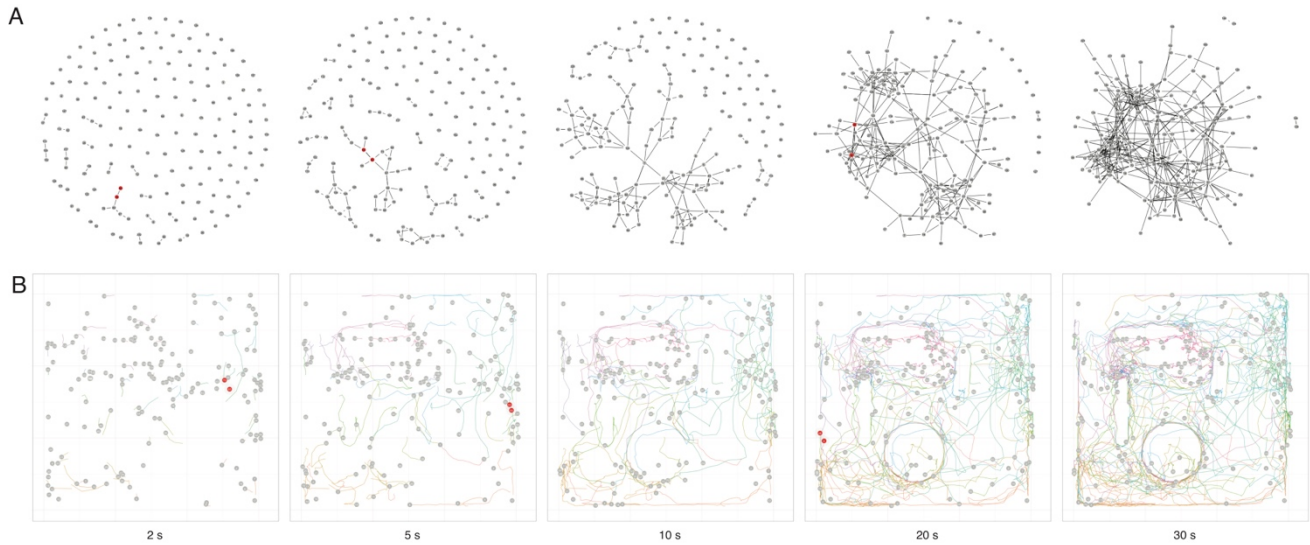


Figure S2. Antennation-based social network construction and spatial trajectories for a colony of *P. californicus* with 239 workers. (A) The social network is shown as it developed over time, with nodes representing individual ants and edges based on interactions manually identified by reviewing video of the colony for direct antennal contacts between workers. (B) These panels illustrate the trajectories of individual ants within this colony simultaneous with the corresponding social network graphs above. Individuals interacting at the specific time of each of these snapshots are highlighted in red.

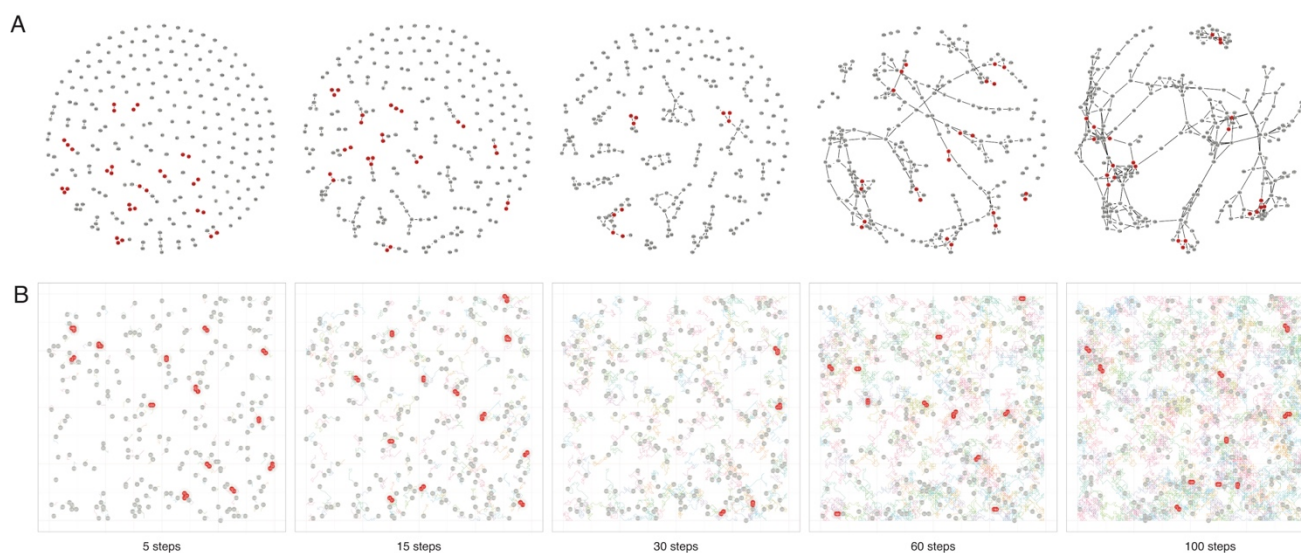


Figure S3. Social network construction and spatial trajectories for a simulated colony of 239 randomly moving virtual individuals. (A) The social network is shown as it developed over 100 timesteps of the model, with nodes representing individuals and edges based on proximity. (B) These panels illustrate the trajectories of individuals within this group’s simulation, simultaneous with the corresponding social network graphs above. Individuals interacting at the specific time of each of these snapshots are highlighted in red.

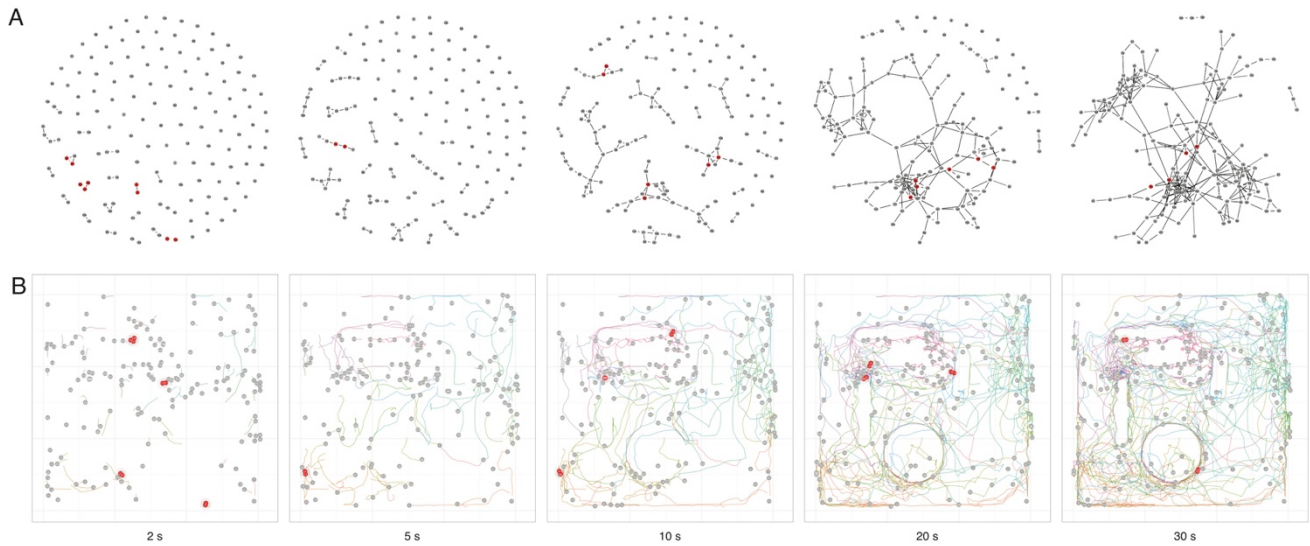


Figure S4. Proximity-based social network construction and spatial trajectories for a colony of *P. californicus* with 239 workers. (A) The social network is shown as it developed over time, with nodes representing individual ants and edges based on interactions inferred based on the proximity between individuals as calculated from their tracked positions. (B) These panels illustrate the spatial trajectories of individual ants (the same as shown in Figure 2B) within this colony simultaneous with the corresponding social network graphs above. Individuals inferred to be interacting at the specific time of each of these snapshots are highlighted in red.

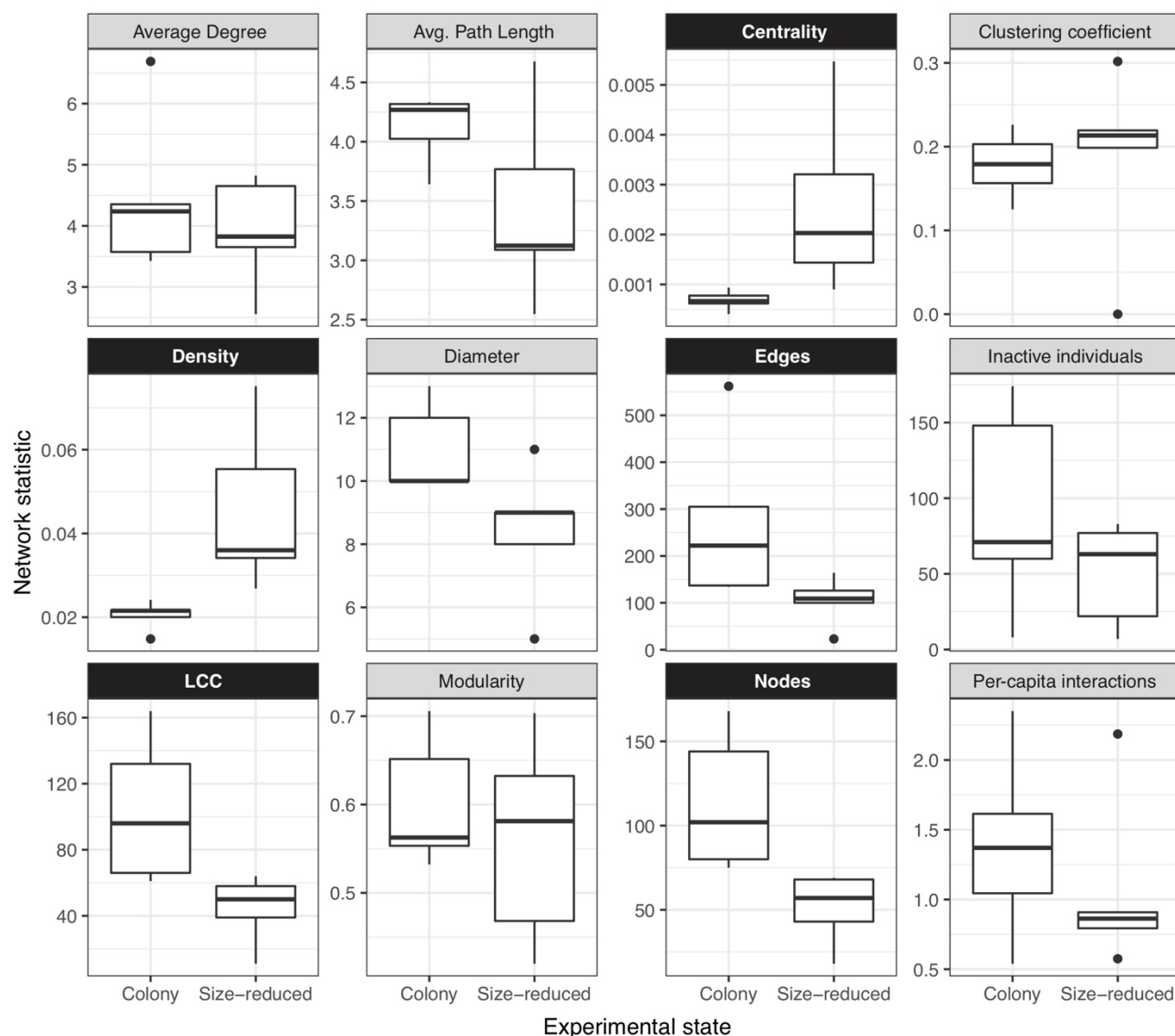


Figure S5. Effects of the experimental colony size reduction on network properties. Five of the properties were significantly affected by the manipulation, as indicated by solid black facet labels, including the number of nodes, the number of edges, closeness centrality, graph density, and the largest connected component.

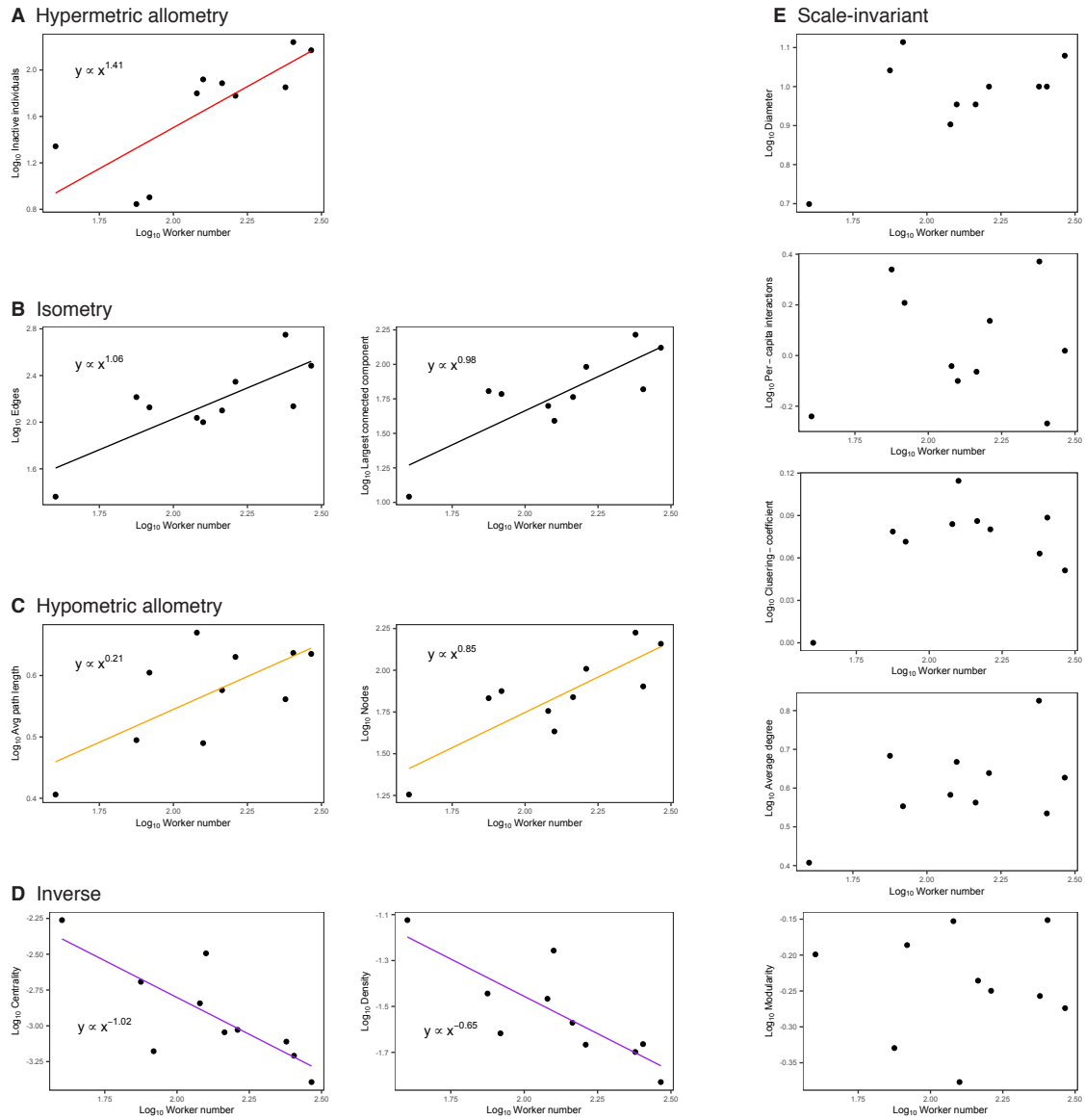


Figure S6. Scaling relationships in *P. californicus* interaction networks. (A) The number of inactive individuals (i.e. individuals not connected to the network) increases hypermetrically with colony size. (B) The number of edges and the size of the largest connected component scale proportional to colony size. (C) The average path length and the number of nodes in the network increase hypometrically with colony size. (D) Centrality and density both have an inverse relationship with colony size. (E) Many aspects of the colony network structure including diameter, per-capita interaction rate, clustering coefficient, average node degree, and modularity did not show any consistent changes with colony size.

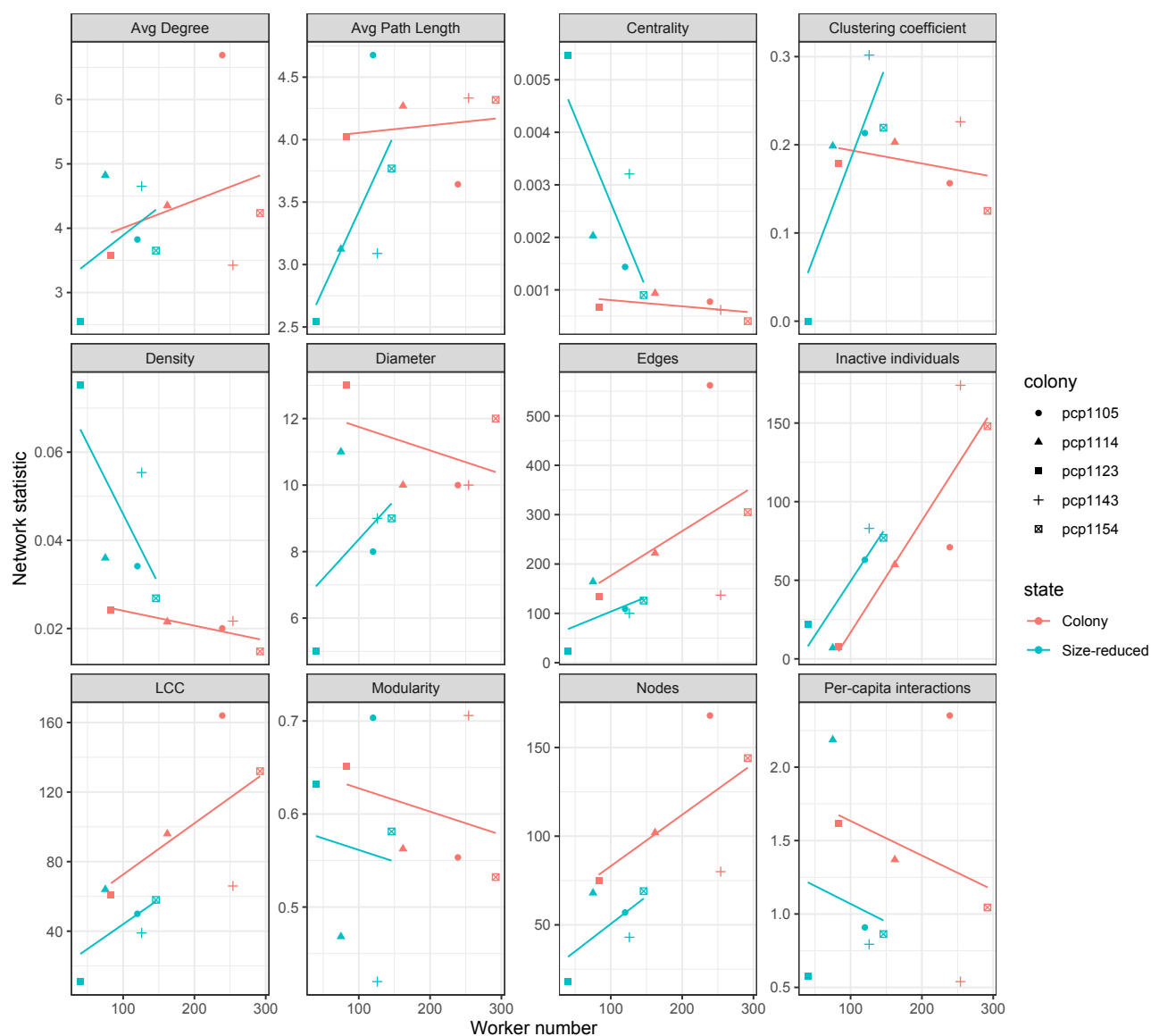


Figure S7. Scaling of network attributes identified by originating colony (point shapes) for each of the two experimental states (distinguished by color). Regression lines are plotted separately for the five whole colonies (red lines) and the same five source colonies, but reduced to half their size (green lines).

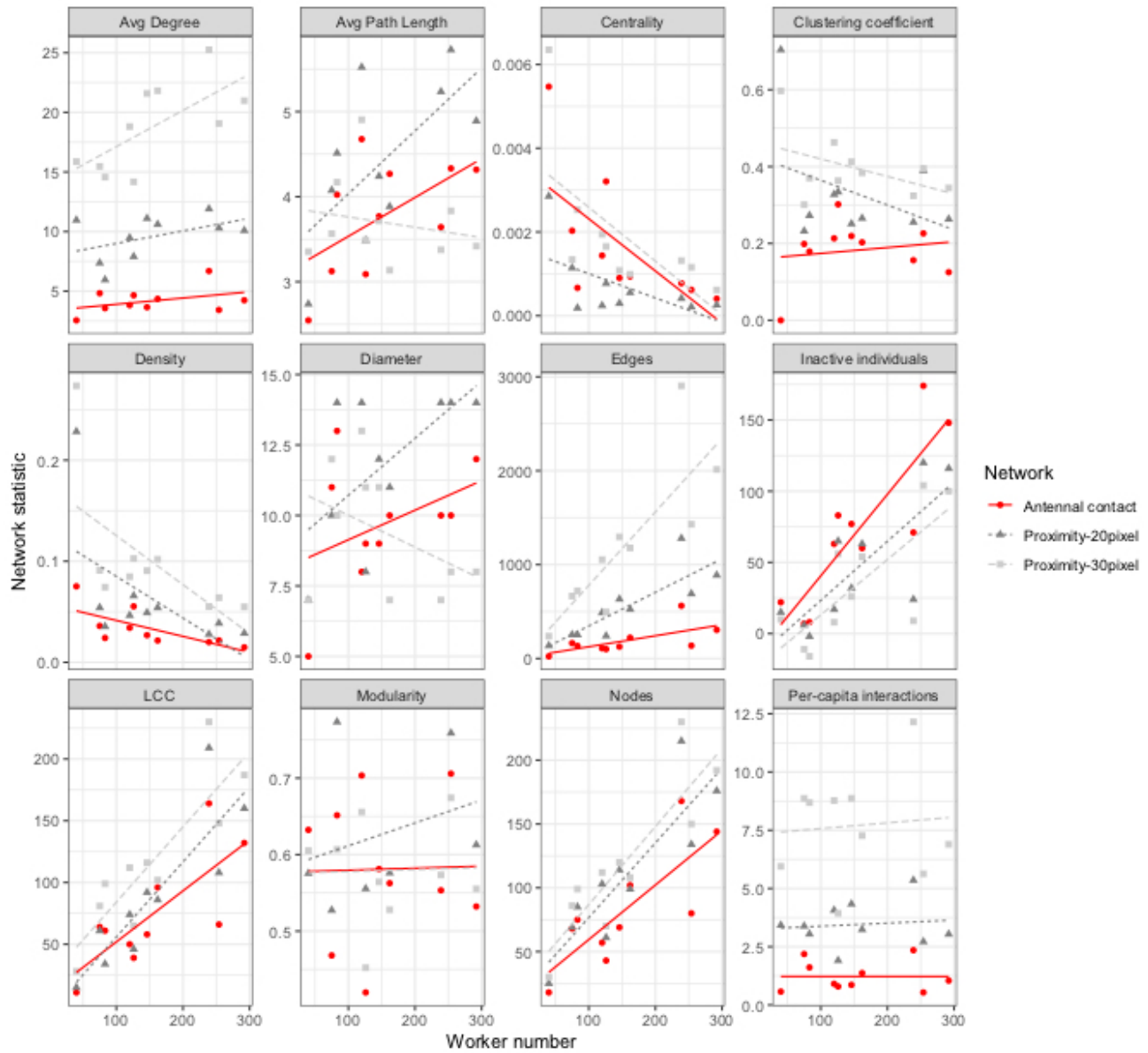


Figure S8. Comparing the scaling of social insect networks constructed using different methods for identifying interactions. We constructed networks based on observing direct antennal contacts (“antennation”; red points and lines) and by indirectly inferring contacts based on spatial proximity (“proximity”; grey points and dotted lines). Results for inferring networks using two different proximity thresholds are included, a 30-pixel distance between ants which was selected to match observed distances between ants making antennal contact, and a 20-pixel threshold selected to require a closer proximity. To evaluate trends in the data, all regression lines are plotted regardless of statistical significance, but complete summary statistics are provided in Table S3. The antennal contact network metrics in this figure are the same data plotted (also in red) in Figure 3 in the main manuscript text where they are compared with the results of the random interaction simulation. The proximity networks are sensitive to the threshold parameter and over-estimated the number of interactions, but the scaling patterns were not significantly different between the 20-pixel proximity networks and those based on observing direct antennal contacts.

Table S1. Colony demographic characteristics and composition. Note that data are from the previous study with these colonies, Waters et al. 2017.

Colony	State	Metabolic rate (mW)	Mass (g)	Queens	Larvae	Pupae	Workers
pcp1105	Whole colony	3.075	0.967	3	25	24	239
pcp1114	Whole colony	3.254	0.860	3	38	44	162
pcp1123	Whole colony	1.403	0.319	1	2	0	83
pcp1143	Whole colony	3.121	1.079	3	49	0	254
pcp1154	Whole colony	3.279	1.211	3	29	0	292
pcp1105	Size-reduced	2.349	0.500	3	12	12	120
pcp1114	Size-reduced	2.402	0.420	3	16	22	75
pcp1123	Size-reduced	0.858	0.163	1	2	0	40
pcp1143	Size-reduced	2.578	0.552	3	24	0	126
pcp1154	Size-reduced	2.400	0.628	3	15	0	146

133 **Table S2.** Network structure metrics.

Colony	State	Vertices	Edges	Diameter	Avg. Path Length	Modularity
pcp1105	Whole colony	168	562	10	3.642	0.553
pcp1114	Whole colony	102	222	10	4.268	0.562
pcp1123	Whole colony	75	134	13	4.024	0.651
pcp1143	Whole colony	80	137	10	4.332	0.705
pcp1154	Whole colony	144	305	12	4.317	0.532
pcp1105	Size-reduced	57	109	8	4.676	0.703
pcp1114	Size-reduced	68	164	11	3.123	0.468
pcp1123	Size-reduced	18	23	5	2.546	0.632
pcp1143	Size-reduced	43	100	9	3.088	0.419
pcp1154	Size-reduced	69	126	9	3.768	0.581

Colony	State	Density	Transitivity	Mean degree	Mean centrality	Inactive
pcp1105	Whole colony	0.020	0.156	6.690	0.0008	71
pcp1114	Whole colony	0.022	0.203	4.353	0.0009	60
pcp1123	Whole colony	0.024	0.179	3.573	0.0007	8
pcp1143	Whole colony	0.022	0.226	3.425	0.0006	174
pcp1154	Whole colony	0.015	0.125	4.236	0.0004	148
pcp1105	Size-reduced	0.033	0.211	3.793	0.0014	63
pcp1114	Size-reduced	0.036	0.199	4.824	0.0020	7
pcp1123	Size-reduced	0.075	0.000	2.556	0.0055	22
pcp1143	Size-reduced	0.055	0.302	4.651	0.0032	83
pcp1154	Size-reduced	0.027	0.219	3.652	0.0009	77

134

135

136

Table S3. Scaling coefficients. This table summarizes the scaling of network characteristics for the ten networks as determined by either watching for antennal contact or inferring interaction based on proximity. The exponents and their 95% confidence intervals reported below are the power-law scaling exponents for models using the colony size (number of workers) as the independent variable. Network characteristics indicated with superscript=1 indicates an aspect of the antennal contact networks that scaled with an exponent significantly different than 0, ones labeled with an asterisk (*) indicate significantly different scaling exponents for the antennal contact networks compared to the proximity networks.

	Antennal contact networks		Proximity networks (20 pixel)	
	Exponent	95% C.I.	Exponent	95% C.I.
Avg. degree	0.21 +/- 0.12 p=0.12, R ² =0.27	(-0.07, 0.50)	0.14 +/- 0.12 p=0.27, R ² =0.15	(-0.13, 0.40)
Avg. path length¹	0.21 +/- 0.08 p=0.03, R ² =0.48	(0.03, 0.40)	0.27 +/- 0.09 p=0.02, R ² =0.53	(0.06, 0.47)
Centrality	-1.02 +/- 0.28 p=0.007, R ² =0.62	(-1.69, -0.37)	-0.96 +/- 0.37 p=0.03, R ² =0.46	(-1.81, -0.10)
Clustering¹	0.21 +/- 0.08 p= 0.03, R ² =0.48	(0.03, 0.40)	-0.27 +/- 0.16 p=0.14, R ² =0.25	(-0.64, 0.10)
Density¹	-0.65 +/- 0.17 p=0.004, R ² =0.66	(-1.03, -0.27)	-0.76 +/- 0.21 p=0.006, R ² =0.63	(-1.25, -0.29)
Diameter	0.23 +/- 0.13 p=0.1, R ² =0.30	(-0.05, 0.53)	0.28 +/- 0.11 p=0.03, R ² =0.47	(0.04, 0.53)
Edges¹	1.05 +/- 0.29 p=0.007, R ² =0.62	(0.38, 1.73)	1.03 +/- 0.16 p=0.0002, R ² =0.84	(0.66, 1.40)
Inactive individuals¹	1.42 +/- 0.39 p=0.008, R ² =0.61	(0.49, 2.33)	1.16 +/- 0.42 p=0.03, R ² =0.52	(0.17, 2.14)
LCC¹	0.99 +/- 0.24 p=0.003, R ² =0.68	(0.43, 1.54)	1.14 +/- 0.17 p=0.0001, R ² =0.85	(0.75, 1.53)
Modularity	-0.004 +/- 0.1 p=0.97, R ² =0.0002	(-0.23, 0.22)	0.06 +/- 0.09 p=0.52, R ² =0.09	(-0.17, 0.30)
Nodes¹	0.84 +/- 0.20 p=0.003, R ² =0.67	(0.38, 1.31)	0.89 +/- 0.14 p=0.0003, R ² =0.83	(0.56, 1.22)
Per-capita interactions	0.06 +/- 0.29 p=0.84, R ² = 0.006	(-0.62, 0.73)	0.03 +/- 0.16 p=0.85, R ² =0.004	(-0.34, 0.40)

```

147 # AntSim-01-function.R
148 # Spatially explicit random interaction simulation paper
149 # JM Toth & JS Waters
150 # June 2021
151
152 # The below code is the actual algorithm to run the simulation
153 # See manuscript for more information
154
155 antsim <- function(size,arena,time,trackingout,graphout){
156   unlink(trackingout)
157   unlink(graphout)
158   tfile <- file(trackingout,"w")
159   dfile <- file(graphout,"w")
160
161   #Initialize ants
162   X = sample(1:arena,size,replace=T)
163   Y = sample(1:arena,size,replace=T)
164
165   lastint = rep(0,size) #last interaction to prevent duplication
166
167   minBound = 0;
168   maxBound = arena;
169
170   interaction = ""
171   coords = ""
172   for(t in 1:timesteps){
173     print(t)
174
175     #Calculate Motion
176     for(i in 1:size){
177       generating = TRUE
178       while(generating){
179         invalidX = TRUE
180         while(invalidX){
181           newX = X[i] + sample(-1:1,1,replace=T)
182           if(newX >= minBound && newX < maxBound){
183             invalidX = FALSE;
184           }
185         }
186         invalidY = TRUE
187         while(invalidY){
188           newY = Y[i] + sample(-1:1,1,replace=T)
189           if(newY >= minBound && newY < maxBound){
190             invalidY = FALSE;
191           }
192         }
193         #Ensure no collisions
194         generating = FALSE
195         for(j in 1:size){
196           if(X[j]==newX && Y[j]==newY && i!=j){

```

```

197         generating = TRUE;
198     }
199 }
200 X[i] = newX
201 Y[i] = newY
202 }
203 }
204 #Calculate Interactions
205 for(i in 1:size){
206     for(j in 1:size){
207         if( abs(X[i]-X[j])<=1 && abs(Y[i]-Y[j])<=1 && lastint[i]!=j && i!=j){
208             #Interaction
209             lastint[i] = j
210             interaction = paste(interaction,i,",",j,",",t,"\n",sep="")
211         }
212     }
213 }
214 #Print Coordinates
215 for(i in 1:size){
216     coords = paste(coords,i,",",X[i],",",Y[i],",",t,"\n",sep="")
217 }
218
219 if(t%%diskcache == 0){
220     cat(interaction,file=dfile,append=T)
221     cat(coords,file=tfile,append=T)
222     interaction = ""
223     coords = ""
224 }
225 }
226
227 close(tfile)
228 close(dfile)
229 }
230

```

```

231 # AntSim-02-simulation-loop.R
232 # Spatially explicit random interaction simulation paper
233 # JM Toth & JS Waters
234 # June 2021
235
236 # The below code is the loop to run the antsim() simulation
237 # based on the antsim() function defined in the AntSim-01-function.R file
238 # See manuscript for more information
239
240 # set working directory
241 setwd("~/Desktop")
242
243 # 1. Define the range of group sizes that should be simulated
244 # 2. Specify how large the (square) virtual arena should be
245 # 3. Indicate how long the simulation should run
246 # 4. Optionally, edit the file names of the saved output files
247
248 for (i in 0:510){
249   size <- 5+i
250   arena <- 100
251   timesteps <- 100
252   trackingout <- paste("coordinates_", size, ".txt", sep="")
253   graphout <- paste("interactions_", size, ".txt", sep="")
254
255   # This input decides how much the program will store before writing to disk.
256   # Tune this to whatever makes the program run fast.
257   diskcache = 10
258
259   # This section stores metadata
260   metadata <- paste("Colony Size: ",size,"\n",
261                     "Arena Size: ",arena,"\n",
262                     "Time Steps: ",timesteps,"\n"
263                     ,sep="")
264   metafile <- file("metadata.txt")
265   writeLines(metadata,metafile)
266   close(metafile)
267
268   antsim(size, arena, time, trackingout, graphout)
269 }
270
271

```

```

272 # AntSim-03a-analysis_network-graphs.R
273 # Spatially explicit random interaction simulation paper
274 # JM Toth & JS Waters
275 # June 2021
276
277 # This script plots the ant colony networks and a selection of the simulated
278 networks
279 # See manuscript for more information
280
281 # LOAD NETWORK PACKAGES
282 library(igraph)
283 library(ggplot2)
284 library(ggraph)
285 library(tidygraph)
286 library(patchwork)
287
288 # SET WORKING DIRECTORY
289 setwd("~/Desktop/pogo-data/")
290
291 # set files to be the contents of the working directory
292 files <- dir()
293 file.number <- length(files)
294
295 #Turn each file in the directory into an igraph directly
296 for(i in 1:length(files)){
297   nam <- files[i]
298   assign(files[i], graph.data.frame(read.table(nam, sep="\t", header=F)))
299 }
300
301 # newfile <- ls(pattern="interactions*")
302 newfile <- ls(pattern="pcp*")
303
304
305 pcal <- c("g01", "g02", "g03", "g04", "g05", "g06", "g07", "g08", "g09", "g10")
306
307 for(i in 1:length(newfile)){
308   g <- get(newfile[i])
309   gg <- as_tbl_graph(g)
310   assign(pcal[i], ggraph(gg, layout=layout_with_fr(g)) + geom_edge_link() +
311     geom_node_point(shape=21, fill="white") +
312     theme_graph(plot_margin=margin(0,0,0,0)) + labs(subtitle=newfile[i]))
313 }
314
315 (g05 + g07 + g03 + g09 + g01 + plot_layout(ncol=5)) / (g06 + g08 + g04 + g10 +
316   g02 + plot_layout(ncol=5))
317
318 g <- get(newfile[i])
319 gg <- as_tbl_graph(g)

```

```

320 assign(pcal[i], ggraph(gg, layout=layout_with_fr(g)) + geom_edge_link() +
321   geom_node_point(shape=21, fill="white") +
322   theme_graph(plot_margin=margin(0,0,0,0)) + labs(subtitle=newfile[i]))
323
324
325 m1 <- graph.data.frame(read.csv("interactions_24.txt"))
326 m1t <- as_tbl_graph(m1)
327 m1g <- ggraph(m1t, layout=layout_with_fr(m1t)) + geom_edge_link() +
328   geom_node_point(shape=23, fill="white") +
329   theme_graph(plot_margin=margin(0,0,0,0)) + labs(subtitle="simulation n=24")
330
331 m2 <- graph.data.frame(read.csv("interactions_75.txt"))
332 m2t <- as_tbl_graph(m2)
333 m2g <- ggraph(m2t, layout=layout_with_fr(m2t)) + geom_edge_link() +
334   geom_node_point(shape=23, fill="white") +
335   theme_graph(plot_margin=margin(0,0,0,0)) + labs(subtitle="simulation n=75")
336
337 m3 <- graph.data.frame(read.csv("interactions_150.txt"))
338 m3t <- as_tbl_graph(m3)
339 m3g <- ggraph(m3t, layout=layout_with_fr(m3t)) + geom_edge_link() +
340   geom_node_point(shape=23, fill="white") +
341   theme_graph(plot_margin=margin(0,0,0,0)) + labs(subtitle="simulation n=150")
342
343 m4 <- graph.data.frame(read.csv("interactions_225.txt"))
344 m4t <- as_tbl_graph(m4)
345 m4g <- ggraph(m4t, layout=layout_with_fr(m4t)) + geom_edge_link() +
346   geom_node_point(shape=23, fill="white") +
347   theme_graph(plot_margin=margin(0,0,0,0)) + labs(subtitle="simulation n=225")
348
349 m5 <- graph.data.frame(read.csv("interactions_300.txt"))
350 m5t <- as_tbl_graph(m5)
351 m5g <- ggraph(m5t, layout=layout_with_fr(m5t)) + geom_edge_link() +
352   geom_node_point(shape=23, fill="white") +
353   theme_graph(plot_margin=margin(0,0,0,0)) + labs(subtitle="simulation n=300")
354
355 (g05 + g07 + g03 + g09 + g01 + plot_layout(ncol=5)) / (g06 + g08 + g04 + g10 +
356   g02 + plot_layout(ncol=5)) / (m1g + m2g + m3g + m4g + m5g + plot_layout(ncol=5))
357
358

```

```

359 # AntSim-03b-analysis-pcal-networks.R
360 # Spatially explicit random interaction simulation paper
361 # JM Toth & JS Waters
362 # June 2021
363
364 # The below code calculates network statistics based
365 # on the observed Pogonomyrmex californicus colonies
366
367 # LOAD NETWORK PACKAGES
368 library(igraph)
369 library(ggplot2)
370 library(ggraph)
371 library(tidygraph)
372 library(patchwork)
373 library(tidyverse)
374 library(brainGraph)
375
376 # SET WORKING DIRECTORY
377 setwd("~/Desktop/pogo-data/Pcal-networks")
378
379 # set files to be the contents of the working directory
380 files <- dir()
381 file.number <- length(files)
382
383 #Turn each file in the directory into an igraph directly
384 for(i in 1:length(files)){
385   nam <- files[i]
386   assign(files[i], graph.data.frame(read.table(nam, sep="\t", header=F)))
387 }
388
389 newfile <- ls(pattern="pcp*")
390
391 # Creating empty vectors with same length as number of networks
392 Network.nodes <- rep(NA, length(newfile))
393 Network.edges <- rep(NA, length(newfile))
394 Diameters <- rep(NA, length(newfile))
395 Diameters.2 <- rep(NA, length(newfile))
396 Diameters.3 <- rep(NA, length(newfile))
397 Diameters.4 <- rep(NA, length(newfile))
398 Avg.degree <- rep(NA, length(newfile))
399 Network.density <- rep(NA, length(newfile))
400 Network.centrality <- rep(NA, length(newfile))
401 Clustering.coefficient <- rep(NA, length(newfile))
402 Average.path.length.un <- rep(NA, length(newfile))
403 Average.path.length.dir <- rep(NA, length(newfile))
404 StDev.degree <- rep(NA, length(newfile))
405 Variance.degree <- rep(NA, length(newfile))
406 Variance.degree <- rep(NA, length(newfile))
407 Efficiency <- rep(NA, length(newfile))
408 LCC <- rep(NA, length(newfile))

```

```

409 Modularity <- rep(NA, length(newfile))
410
411 # define this function
412 mean centrality <- function(x){mean(closeness(x, mode=c("all")))}
413
414 # Calculate the network metrics for each network
415 for(i in 1:length(newfile)){
416   Network.nodes[i] <- vcount(get(newfile[i]))
417   Network.edges[i] <- ecoun t(get(newfile[i]))
418   Diameters[i] <- diameter(get(newfile[i]))
419   Diameters.2[i] <- diameter(get(newfile[i]), unconnected=F)
420   Diameters.3[i] <- diameter(get(newfile[i]), unconnected=F, directed=F)
421   Diameters.4[i] <- diameter(get(newfile[i]), directed=F)
422   Avg.degree[i] <- mean(degree(get(newfile[i])))
423   Network.density[i] <- graph.density(get(newfile[i]))
424   Network.centrality[i] <- mean.centrality(get(newfile[i]))
425   Clustering.coefficient[i] <- transitivity(get(newfile[i]))
426   Average.path.length.un[i] <- average.path.length(get(newfile[i]), directed=F)
427   Average.path.length.dir[i] <- average.path.length(get(newfile[i]), directed=T)
428   StDev.degree[i] <- sd(degree(get(newfile[i])))
429   Variance.degree[i] <- sd(degree(get(newfile[i])))^2
430   Efficiency[i] <- efficiency(get(newfile[i]), type="global")
431   LCC[i] <- max(clusters(get(newfile[i]), mode="weak")$csize)
432   Modularity[i] <- modularity(cluster_leading_eigen(get(newfile[i])))
433 }
434
435 # Bring the vectors together as a data frame
436 net.df <- data.frame(Nodes=Network.nodes, Edges=Network.edges,
437   Diameter=Diameters, Avg.Degree=Avg.degree, Density=Network.density,
438   Centrality=Network.centrality, Diameter.2=Diameters.2, Diameter.3=Diameters.3,
439   Diameter.4=Diameters.4, Clustering.coefficient=Clustering.coefficient,
440   Avg.Path.Length.un=Average.path.length.un,
441   Avg.Path.Length.dir=Average.path.length.dir, Efficiency=Efficiency, LCC=LCC,
442   Modularity=Modularity)
443
444 # Add additional information, including colony sizes and IDs
445 net.df$id <- newfile
446 net.df$worker.number <- c(239, 120, 162, 75, 83, 40, 254, 126, 292, 146)
447 net.df$metrate <- c(3.075, 2.349, 3.254, 2.402, 1.403, 0.858, 3.121, 2.578,
448   3.279, 2.400)
449 net.df$mass <- c(.967, .5, .860, .420, .319, .163, 1.079, .552, 1.211, .628)
450 net.df$state <- rep(c("Colony", "Size-reduced"), 5)
451 net.df$metrate.per.g <- net.df$metrate/net.df$mass
452 net.df$connected.fraction <- net.df$Nodes/net.df$worker.number
453 net.df$per.capita.interactions <- net.df$Edges/net.df$worker.number
454 net.df$inactives <- net.df$worker.number - net.df$Nodes
455 net.df$inactive.fr <- net.df$inactives/net.df$worker.number
456
457
458 # Metabolic rate figure

```

```

459 metrate.plot.1 <- ggplot(net.df, aes(x=worker.number, y=metrate.per.g)) +
460   geom_point() + stat_smooth(method="lm", se=F, col="red") + theme_bw() +
461   ylab("Mass specific metabolic rate (mW/g)") + xlab("Number of workers")
462 metrate.plot.2 <- ggplot(net.df, aes(x=state, y=metrate.per.g)) + geom_boxplot()
463   + ylab("Mass specific metabolic rate (mW/g)") + xlab("") + theme_bw()
464 metrate.plot.1 + metrate.plot.2
465
466
467 # Network scaling figures
468 pcal.scaling <- pivot_longer(net.df, names_to="metric", values_to="value", -
469   c("Diameter.2", "Diameter.3", "Diameter.4", "Avg.Path.Length.dir", "id",
470     "worker.number", "metrate", "mass", "metrate.per.g", "state", "Efficiency",
471     "connected.fraction"))
472 ggplot(pcal.scaling, aes(x=worker.number, y=value)) + facet_wrap(~metric,
473   scales="free_y", ncol=5) + geom_point() + theme_bw()
474
475 ggplot(pcal.scaling, aes(x=worker.number, y=value)) + facet_wrap(~metric,
476   scales="free_y", ncol=4) + geom_point(aes(shape=state)) + theme_bw() +
477   stat_smooth(method="lm", se=F, lty=2, size=0.5, color="grey") + guides(shape=F)
478   + xlab("Worker number") + ylab("Network statistic")
479
480 ggplot(pcal.scaling, aes(x=state, y=value)) + facet_wrap(~metric,
481   scales="free_y", ncol=4) + geom_boxplot() + theme_bw() +
482   stat_smooth(method="lm", se=F, lty=2, size=0.5, color="grey") + guides(shape=F)
483   + ylab("Network statistic")
484
485
486 # T-tests
487 wilcox.test(Centrality ~ state, data=net.df, paired=F)
488 wilcox.test(Density ~ state, data=net.df, paired=F)
489 wilcox.test(Diameter ~ state, data=net.df, paired=F)
490 wilcox.test(Edges ~ state, data=net.df, paired=F)
491 wilcox.test(LCC ~ state, data=net.df, paired=F)
492 wilcox.test(Nodes ~ state, data=net.df, paired=F)
493
494 # Scaling analyses
495
496 # HYPERMETRY
497 summary(lm(log10(inactives) ~ log10(worker.number), data=net.df))
498 # p=0.007, R2=0.61, F1,8=12.61, exponent 1.41 +/- 0.39 SE
499 s1 <- ggplot(net.df, aes(x=log10(worker.number), y=log10(inactives))) +
500   geom_point() + theme_bw() + xlab(expression(Log[10]*~Worker~number)) +
501   ylab(expression(Log[10]*~Inactive~individuals)) + stat_smooth(method="lm", se=F,
502     col="red", size=0.5) + annotate("text", x=1.75, y=2,
503     label=expression(y%prop%x^1.41)) + theme(panel.grid.major = element_blank(),
504     panel.grid.minor = element_blank(), axis.title=element_text(size=8),
505     axis.text=element_text(size=6))
506
507 # ISOMETRY
508 summary(lm(log10(LCC) ~ log10(worker.number), data=net.df))

```

```

509 # p=0.003, R2=0.67, F1,8=16.75, exponent 0.98 +/- 0.24 SE
510 s2 <- ggplot(net.df, aes(x=log10(worker.number), y=log10(LCC))) + geom_point() +
511   theme_bw() + xlab(expression(Log[10]*~Worker~number)) +
512   ylab(expression(Log[10]*~Largest~connected~component)) +
513   stat_smooth(method="lm", se=F, col="black", size=0.5) + annotate("text", x=1.75,
514   y=2, label=expression(y%prop%x^0.98)) + theme(panel.grid.major =
515   element_blank(), panel.grid.minor = element_blank(),
516   axis.title=element_text(size=8), axis.text=element_text(size=6))
517
518 summary(lm(log10(Edges) ~ log10(worker.number), data=net.df))
519 # p=0.006, R2=0.62, F1,8=13.1, exponent 1.06 +/- 0.29 SE
520 s3 <- ggplot(net.df, aes(x=log10(worker.number), y=log10(Edges))) + geom_point()
521   + theme_bw() + xlab(expression(Log[10]*~Worker~number)) +
522   ylab(expression(Log[10]*~Edges)) + stat_smooth(method="lm", se=F, col="black",
523   size=0.5) + annotate("text", x=1.75, y=2.5, label=expression(y%prop%x^1.06)) +
524   theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
525   axis.title=element_text(size=8), axis.text=element_text(size=6))
526
527 # ALLOMETRY: HYPOMETRIC
528 summary(lm(log10(Nodes) ~ log10(worker.number), data=net.df))
529 # p=0.003, R2=0.69, F1,8=17.52, exponent 0.85 +/- 0.20 SE
530 s4 <- ggplot(net.df, aes(x=log10(worker.number), y=log10(Nodes))) + geom_point()
531   + theme_bw() + xlab(expression(Log[10]*~Worker~number)) +
532   ylab(expression(Log[10]*~Nodes)) + stat_smooth(method="lm", se=F, col="orange",
533   size=0.5) + annotate("text", x=1.75, y=2.1, label=expression(y%prop%x^0.85)) +
534   theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
535   axis.title=element_text(size=8), axis.text=element_text(size=6))
536
537 summary(lm(log10(Avg.Path.Length.un) ~ log10(worker.number), data=net.df))
538 # p=0.03, R2=0.48, F1,8=7.25, exponent 0.21 +/- 0.08 SE
539 s5 <- ggplot(net.df, aes(x=log10(worker.number), y=log10(Avg.Path.Length.un))) +
540   geom_point() + theme_bw() + xlab(expression(Log[10]*~Worker~number)) +
541   ylab(expression(Log[10]*~Avg~path~length)) + stat_smooth(method="lm", se=F,
542   col="orange", size=0.5) + annotate("text", x=1.75, y=.62,
543   label=expression(y%prop%x^0.21)) + theme(panel.grid.major = element_blank(),
544   panel.grid.minor = element_blank(), axis.title=element_text(size=8),
545   axis.text=element_text(size=6))
546
547 # INVERSE
548 summary(lm(log10(Density) ~ log10(worker.number), data=net.df))
549 # p=0.004, R2=0.65, F1,8=15.5, exponent -0.65 +/- 0.17 SE
550 s6 <- ggplot(net.df, aes(x=log10(worker.number), y=log10(Density))) +
551   geom_point() + theme_bw() + xlab(expression(Log[10]*~Worker~number)) +
552   ylab(expression(Log[10]*~Density)) + stat_smooth(method="lm", se=F,
553   col="purple", size=0.5) + annotate("text", x=1.75, y=-1.6,
554   label=expression(y%prop%x^-0.65)) + theme(panel.grid.major = element_blank(),
555   panel.grid.minor = element_blank(), axis.title=element_text(size=8),
556   axis.text=element_text(size=6))
557
558 summary(lm(log10(Centrality) ~ log10(worker.number), data=net.df))

```

```

559 # p=0.007, R2=0.62, F1,8=12.95, exponent -1.02 +/- 0.28 SE
560 s7 <- ggplot(net.df, aes(x=log10(worker.number), y=log10(Centrality))) +
561   geom_point() + theme_bw() + xlab(expression(Log[10]*~Worker~number)) +
562   ylab(expression(Log[10]*~Centrality)) + stat_smooth(method="lm", se=F,
563     col="purple", size=0.5) + annotate("text", x=1.75, y=-3,
564     label=expression(y~prop~x^-1.02)) + theme(panel.grid.major = element_blank(),
565     panel.grid.minor = element_blank(), axis.title=element_text(size=8),
566     axis.text=element_text(size=6))
567
568 # SCALE INVARIANT:
569 summary(lm(log10(per.capita.interactions) ~ log10(worker.number), data=net.df))
570 s8 <- ggplot(net.df, aes(x=log10(worker.number),
571   y=log10(per.capita.interactions))) + geom_point() + theme_bw() +
572   xlab(expression(Log[10]*~Worker~number)) +
573   ylab(expression(Log[10]*~per.capita.interactions)) + theme(panel.grid.major =
574     element_blank(), panel.grid.minor = element_blank(),
575     axis.title=element_text(size=8), axis.text=element_text(size=6))
576
577 summary(lm(log10(Modularity) ~ log10(worker.number), data=net.df))
578 s9 <- ggplot(net.df, aes(x=log10(worker.number), y=log10(Modularity))) +
579   geom_point() + theme_bw() + xlab(expression(Log[10]*~Worker~number)) +
580   ylab(expression(Log[10]*~Modularity)) + theme(panel.grid.major =
581     element_blank(), panel.grid.minor = element_blank(),
582     axis.title=element_text(size=8), axis.text=element_text(size=6))
583
584 summary(lm(log10(Diameter) ~ log10(worker.number), data=net.df))
585 s10 <- ggplot(net.df, aes(x=log10(worker.number), y=log10(Diameter))) +
586   geom_point() + theme_bw() + xlab(expression(Log[10]*~Worker~number)) +
587   ylab(expression(Log[10]*~Diameter)) + theme(panel.grid.major =
588     element_blank(), panel.grid.minor = element_blank(),
589     axis.title=element_text(size=8), axis.text=element_text(size=6))
590
591 summary(lm(log10(Clustering.coefficient+1) ~ log10(worker.number), data=net.df))
592 s11 <- ggplot(net.df, aes(x=log10(worker.number),
593   y=log10(Clustering.coefficient+1))) + geom_point() + theme_bw() +
594   xlab(expression(Log[10]*~Worker~number)) + ylab(expression(Log[10]*~Clusering-
595     coefficient)) + theme(panel.grid.major = element_blank(), panel.grid.minor =
596     element_blank(), axis.title=element_text(size=8),
597     axis.text=element_text(size=6))
598
599 summary(lm(log10(Avg.Degree) ~ log10(worker.number), data=net.df))
600 s12 <- ggplot(net.df, aes(x=log10(worker.number), y=log10(Avg.Degree))) +
601   geom_point() + theme_bw() + xlab(expression(Log[10]*~Worker~number)) +
602   ylab(expression(Log[10]*~Average~degree)) + theme(panel.grid.major =
603     element_blank(), panel.grid.minor = element_blank(),
604     axis.title=element_text(size=8), axis.text=element_text(size=6))
605
606 # Composite figure

```

```
607 (s1 + plot_spacer() + plot_spacer()) / (s2 + s3 + plot_spacer()) / (s4 + s5 +
608 plot_spacer()) / (s6 + s7 + plot_spacer()) / (s8 + s9 + s10) / (s11 + s12 +
609 plot_spacer())
610
611 # Summary statistic (Mean +/- SEM)
612 std <- function(x) {
613   print(mean(x))
614   print(sd(x)/sqrt(length(x)))
615   }
616
```

```

617 # AntSim-03c-analysis-simulated-networks.R
618 # Spatially explicit random interaction simulation paper
619 # JM Toth & JS Waters
620 # June 2021
621
622 # The below code calculates network statistics based
623 # on the random simulation results
624
625 # LOAD NETWORK PACKAGES
626 library(igraph)
627 library(ggplot2)
628 library(ggraph)
629 library(tidygraph)
630 library(patchwork)
631 library(tidyverse)
632 library(brainGraph)
633
634 # SET WORKING DIRECTORY
635 setwd("~/Desktop/simulation-data/interactions")
636
637 # set files to be the contents of the working directory
638 files <- dir()
639 file.number <- length(files)
640
641 #Turn each file in the directory into an igraph directly
642 for(i in 1:length(files)){
643   nam <- files[i]
644   assign(files[i], graph.data.frame(read.csv(nam, sep=",", header=F)))
645 }
646
647 newfile <- ls(pattern="interactions*")
648
649 # Creating empty vectors with same length as number of networks
650 Network.nodes <- rep(NA, length(newfile)) #create an empty vector with same
651 length as number of networks
652 Network.edges <- rep(NA, length(newfile)) #create an empty vector with same
653 length as number of networks
654 Diameters <- rep(NA, length(newfile))
655 Diameters.2 <- rep(NA, length(newfile))
656 Diameters.3 <- rep(NA, length(newfile))
657 Diameters.4 <- rep(NA, length(newfile))
658 Avg.degree <- rep(NA, length(newfile))
659 Network.density <- rep(NA, length(newfile))
660 Network.centrality <- rep(NA, length(newfile))
661 Clustering.coefficient <- rep(NA, length(newfile))
662 Average.path.length.un <- rep(NA, length(newfile))
663 Average.path.length.dir <- rep(NA, length(newfile))
664 StDev.degree <- rep(NA, length(newfile))
665 Variance.degree <- rep(NA, length(newfile))
666 Variance.degree <- rep(NA, length(newfile))

```

```

667 Efficiency <- rep(NA, length(newfile))
668 LCC <- rep(NA, length(newfile))
669 Modularity <- rep(NA, length(newfile))
670
671 # define this function
672 mean centrality <- function(x){mean(closeness(x, mode=c("all")))}
673
674 # Needed to make this function because some of the simulated
675 # networks were returning errors in the modularity function
676 james.modularity <- function(x){
677   var1 <- try(modularity(cluster_leading_eigen(x)), silent=T)
678   print(ifelse(is.numeric(var1), var1, NA))
679 }
680
681 # Calculate the network metrics for each network
682 for(i in 1:length(newfile)){
683   Network.nodes[i] <- vcount(get(newfile[i]))
684   Network.edges[i] <- ecount(get(newfile[i]))
685   Diameters[i] <- diameter(get(newfile[i]))
686   Diameters.2[i] <- diameter(get(newfile[i]), unconnected=F)
687   Diameters.3[i] <- diameter(get(newfile[i]), unconnected=F, directed=F)
688   Diameters.4[i] <- diameter(get(newfile[i]), directed=F)
689   Avg.degree[i] <- mean(degree(get(newfile[i])))
690   Network.density[i] <- graph.density(get(newfile[i]))
691   Network centrality[i] <- mean centrality(get(newfile[i]))
692   Clustering.coefficient[i] <- transitivity(get(newfile[i]))
693   Average.path.length.un[i] <- average.path.length(get(newfile[i]), directed=F)
694   Average.path.length.dir[i] <- average.path.length(get(newfile[i]), directed=T)
695   StDev.degree[i] <- sd(degree(get(newfile[i])))
696   Variance.degree[i] <- sd(degree(get(newfile[i])))^2
697   Efficiency[i] <- efficiency(get(newfile[i]), type="global")
698   LCC[i] <- max(clusters(get(newfile[i]), mode="weak")$csize)
699   Modularity[i] <- james.modularity(get(newfile[i]))
700 }
701
702 # Bring the vectors together as a data frame
703 sim.df <- data.frame(Nodes=Network.nodes, Edges=Network.edges,
704   Diameter=Diameters, Avg.Degree=Avg.degree, Density=Network.density,
705   Centrality=Network centrality, Diameter.2=Diameters.2, Diameter.3=Diameters.3,
706   Diameter.4=Diameters.4, Clustering.coefficient=Clustering.coefficient,
707   Avg.Path.Length.un=Average.path.length.un,
708   Avg.Path.Length.dir=Average.path.length.dir, Efficiency=Efficiency, LCC=LCC,
709   Modularity=Modularity)
710
711 # Add additional information, including group sizes and IDs
712 sim.df$id <- newfile
713 sim.df$number <- as.numeric(gsub(".*?([0-9]+).*", "\\1", sim.df$id))
714 sim.df$connected.fraction <- sim.df$LCC/sim.df$number
715 sim.df$per.capita.interactions <- sim.df$Edges/sim.df$number
716 sim.df$inactives <- sim.df$number - sim.df$Nodes

```

```

717
718
719 # Network scaling figure
720 sim.scaling <- pivot_longer(sim.df, names_to="metric", values_to="value", -
721   c("Diameter.2", "Diameter.3", "Diameter.4", "Avg.Path.Length.dir", "id",
722     "number", "Efficiency", "connected.fraction"))
723
724 ggplot(sim.scaling, aes(x=number, y=value)) + facet_wrap(~metric,
725   scales="free_y", ncol=5) + geom_point() + theme_bw()
726 ggplot(sim.scaling, aes(x=number, y=value)) + facet_wrap(~metric,
727   scales="free_y", ncol=4) + geom_point(alpha=0.5, size=.5) + theme_bw() +
728   guides(shape=F) + xlab("Simulated colony size") + ylab("Network statistic")
729
730 # Merging simulation data with colony data
731 # to create the overaly in Figure 7
732 sim.scaling.2 <- sim.scaling[,c(7, 9, 10)]
733 sim.scaling.2$data <- "simulation"
734
735 # Note that this requires running code in the 03b.R file
736 pcal.scaling.2 <- pcal.scaling[,c(7, 13, 14)]
737 names(pcal.scaling.2)[1] <- "number"
738 pcal.scaling.2$data <- "colony"
739
740 # Creating the merged data frame with matching columns
741 mega <- rbind(sim.scaling.2, pcal.scaling.2)
742
743 # Plot for Figure 7
744 ggplot(mega, aes(x=number, y=value, col=data)) + facet_wrap(~metric,
745   scales="free_y", ncol=4) + geom_point(size=.5) + theme_bw() + guides(shape=F) +
746   xlab("Group size") + ylab("Network statistic") + stat_smooth(data=subset(mega,
747     data=="colony"), method = "loess", formula = y ~ x, size = 1, se=T, span=1) +
748   scale_color_manual("Data source: ", values=c("red", "black"))+
749   theme(legend.position="bottom")
750
751 # Modified Fig 7 plot, minimum size 25
752 # to avoid centrality/density sensitivity at small group sizes
753 ggplot(mega[mega$number > 25,], aes(x=number, y=value, col=data)) +
754   facet_wrap(~metric, scales="free_y", ncol=4) + geom_point(size=.5) + theme_bw() +
755   guides(shape=F) + xlab("Group size") + ylab("Network statistic") +
756   stat_smooth(data=subset(mega, data=="colony"), method = "loess", formula = y ~ x,
757     size = 1, se=T, span=1) + scale_color_manual("Data source: ", values=c("red",
758     "black"))+ theme(legend.position="bottom") + scale_x_continuous(breaks=c(0,100,
759     200, 300), limits=c(0,350))
760

```

```

761 # AntSim-04-network-and-spatial-animation.R
762 # Spatially explicit random interaction simulation paper
763 # JM Toth & JS Waters
764 # June 2021
765
766 # The below code generates frames for an animation
767 # showing the development of a social network at the
768 # same time as the spatial trajectories of the individuals
769 # This is the code for one of the simulated data sets,
770 # but it's the same for real colony data too.
771
772 # LOAD NETWORK PACKAGES
773 library(ggplot2)
774 library(igraph)
775 library(patchwork)
776 library(ggraph)
777 library(tidygraph)
778
779 setwd("~/Desktop/simulation-data")
780
781 # Import interaction data from the simulation
782 net <- read.table("interactions/interactions_239.txt", sep=",")
783 names(net) <- c("ant1", "ant2", "t")
784
785 # Import spatial data from the simulation
786 map <- read.table("coordinates/coordinates_239.txt", sep=",")
787 head(map)
788 names(map) <- c("ant", "x", "y", "t")
789 map$interaction <- 0
790
791 # Label the coordinates when two ants interact
792 for(i in 1:length(net$t)){
793   time <- net$t[i]
794   ant1 <- net$ant1[i]
795   ant2 <- net$ant2[i]
796   map$interaction[map$t == time & map$ant == ant1] <- 1
797   map$interaction[map$t == time & map$ant == ant2] <- 1
798 }
799
800 # Make graph
801 g <- graph.data.frame(net, directed=F)
802 E(g)$time <- net[,3]
803
804 gg_color_hue <- function(n) {
805   hues = seq(15, 375, length=n+1)
806   hcl(h=hues, l=65, c=100)[1:n]
807 }
808
809 # Make tidy graph and set initial layout
810 g1 <- as_tbl_graph(g)

```

```

811 old.fr.coordinates <- layout_randomly(g1)
812 # ggraph(g1, layout=layout_with_fr(g1)) + geom_edge_link() + geom_node_point() +
813   theme_graph()
814
815 ###
816 ###
817 ###
818
819 # Directory to save animation frames to
820 setwd("~/Desktop/Animations/vis04-n239")
821
822 png("SimulationModeln234-%03d.png", width=1600, height=900)
823
824 for(ti in seq(1,max(map$t),1)){
825
826   # Making the left panel (network)
827   E(g)$edge.weight <- ifelse(E(g)$time < ti+1,1,0)
828   E(g)$status <- ifelse(E(g)$time < ti+1,"a","b")
829   E(g)$color <- ifelse(E(g)$time < ti+1,"gray",rgb(0,0,0,0))
830   V(g)$color <- ifelse(graph.strength(g)==0,rgb(0,0,0,0),"grey")
831   interacting.now <- as.character(map$ant[map$t == ti & map$interaction == 1])
832   V(g)$color[V(g)$name %in% interacting.now] <- gg_color_hue(1)
833
834   g1 <- as_tbl_graph(g)
835   coordinates.fr <- layout_with_fr(g1, coords = old.fr.coordinates, start.temp=.5,
836     weights=E(g)$edge.weight)
837
838   p.1 <- ggraph(g1, layout=coordinates.fr) + geom_edge_link(aes(width=status,
839     start_cap=label_rect(node1.name), end_cap=label_rect(node2.name))) +
840     theme_graph() + scale_edge_width_manual(values=c(.5,0), breaks=c("a",
841     "b"),drop=F) + geom_node_point(aes(color=color), size=5) +
842     geom_node_text(aes(label=name), size=3) + scale_color_manual(values=c("grey",
843     "red"), breaks=c("grey", "#F8766D")) + guides(edge_width=F, color=F) +
844     ggtitle("Social network") + theme(plot.title = element_text(size = 10, face =
845     "bold", family="sans"))
846
847   old.fr.coordinates <- coordinates.fr
848
849   # Making the right panel (spatial trajectories)
850   map.t <- map[map$t == ti,]
851   df.1 <- map[map$t < ti+1,]
852   df.2 <- map[map$t == ti,]
853
854   p.2 <- ggplot(df.1, aes(x=x, y=y, group=ant, color=as.factor(ant))) +
855     geom_path(size=0.4, alpha=.5) + theme_bw() + scale_color_discrete(guide=F) +
856     labs(x=NULL, y=NULL) + scale_x_continuous(limits=c(0,100))+
857     scale_y_continuous(limits=c(0,100)) + annotate("point", x=df.2$x[df.2$ant %in%
858     interacting.now], y=df.2$y[df.2$ant %in% interacting.now], size=15, color="light
859     grey", alpha=0.2) + annotate("point", x=df.2$x, y=df.2$y, size=6, color="grey")
860     + annotate("point", x=df.2$x[df.2$ant %in% interacting.now], y=df.2$y[df.2$ant

```

```

861 %in% interacting.now], size=6, color="red") + annotate("text", x=df.2$x,
862 y=df.2$y, label=df.2$ant, size=3, color="white") +
863 theme(axis.line=element_blank(),
864 axis.text.x=element_blank(),axis.text.y=element_blank(),
865 axis.ticks=element_blank(),axis.title.x=element_blank(),axis.title.y=element_bla
866 nk()) + ggtitle("Spatial trajectories") + theme(plot.title = element_text(size =
867 10, face = "bold", family="sans"))
868
869 print((p.1 + p.2) + plot_annotation(title="Spatially explicit random interaction
870 social network simulation", subtitle="Colony size: N=239 virtual ants", caption=
871 paste("Simulation frame: ", ti, "/100", sep="")))
872 }
873 dev.off()
874
875 # NOTES
876 # The code above only saves a sequence of frames.
877 # Beware: For large networks each frame can take 1-5 minutes to render and save.
878 #
879 #To put the frames together and save as a movie,
880 # we used the "Open Image Sequence..." option in an old version of Quicktime
881 Player,
882 # but you could also use ImageJ to do the same thing.
883

```

```

884 # AntSim-05-trajectory-based-proximity-networks.R
885 # Extracting social network based on proximity
886 # June 2021
887
888 library(ggplot2)
889 library(igraph)
890
891 # Set working directory (to find data files)
892 setwd("~/Desktop/source-trajectories")
893
894 # Set files to be the contents of the directory
895 files <- dir()
896 file.number <- length(files)
897
898 for(i in 1:length(files)){
899   setwd("~/Desktop/source-trajectories")
900   nam <- files[i]
901   assign(files[i], read.csv(nam, header=T))
902   map <- get(files[i])
903   label <- nam
904
905   interaction_distance <- 30
906   diskcache = 10
907
908   names(map) <- c("ant", "frame", "x", "y")
909
910   # This file had some extra data, so limiting it for consistency
911   map <- map[map$frame<=90,]
912
913   # Plot of all trajectories
914   setwd("~/Desktop/figures")
915   pdf(file=paste(label, "-traj", ".pdf", sep=""), width=8, height=8)
916   p <- ggplot(map, aes(x=x, y=y, col=ant, group=ant)) + geom_path() +
917     guides(col="none") + labs(caption=label)
918   print(p)
919   dev.off()
920
921   setwd("~/Desktop/results-proximity-networks")
922
923   #Summary data about the recording
924   ant_list = unique(map$ant)
925   num_ants = max(ant_list) #This may not actually be the exact number of ants as
926     some are not counted on occasion
927   num_frames <- max(map$frame)
928
929   #Variables for calculating interactions
930   lastint = rep(0,num_ants) #last interaction to prevent duplication
931   distance_reset =
932     kronecker(matrix(1,num_ants,num_ants),TRUE)#rep(TRUE,[num_ants,num_ants]) #Must
933     exit and re-enter interaction distance

```

```

934
935 #Setting up write file
936 graphout <- paste("interactions_", "trajectories_", label, ".txt", sep="")
937 dfile <- file(graphout,"w")
938 interaction <- ""
939
940 #Run analysis
941 for(t in 1:num_frames){
942   #Select relevant time point
943   print(t)
944   mapt <- map[map$frame==t,]
945
946   #Process each pair of ants
947   for(ant1 in ant_list){
948
949     if( !(ant1 %in% mapt$ant)){
950       print(paste("ant number ",ant1," not tracked on timestamp ",t,sep=""))
951       next
952     }
953
954     for(ant2 in ant_list){
955
956       if( ! (ant2 %in% mapt$ant)){
957         next
958       }
959
960       if(ant1 != ant2){
961
962         #Finding indices of X and Y is necessary because some ants are skipped
963         ant1x <- mapt$x[which(mapt$ant==ant1)]
964         ant2x <- mapt$x[which(mapt$ant==ant2)]
965         ant1y <- mapt$y[which(mapt$ant==ant1)]
966         ant2y <- mapt$y[which(mapt$ant==ant2)]
967         #Need to check distance, last interaction, and whether the ants have
968         moved apart
969         distance <- sqrt( (ant1x - ant2x)**2 + (ant1y - ant2y)**2 )
970         if( (distance < interaction_distance) && (lastint[ant1] != ant2) &&
971         (distance_reset[ant1,ant2]>0) ){
972           #Interaction
973           lastint[ant1] <- ant2
974           interaction = paste(interaction,ant1,",",ant2,",",t,"\n",sep="")
975           distance_reset[ant1,ant2] <- 0
976         }
977         else if(distance >= interaction_distance){
978           distance_reset[ant1,ant2] <- 1
979         }
980         #Uncomment this to turn off the requirement for moving apart before
981         interacting
982         #distance_reset[ant1,ant2] = TRUE
983

```

```
984     }
985   }
986 }
987
988 #Write data
989 if(t%%diskcache == 0){
990   cat(interaction,file=dfile,append=T)
991   interaction <- ""
992 }
993 #ggplot(mapt, aes(x=x, y=y, col=ant)) + geom_point() + guides(col="none")
994 }
995 cat(interaction,file=dfile,append=T)
996 close(dfile)
997
998 }
```