This document describes computer code used in the manuscript "A Method for Estimating Abundance of Mobile Populations Using Telemetry and Counts of Unmarked Animals" by Clement et al. It is recommended to read the manuscript prior to this document.

Contents

1: Getting Started

First, install Program R. http://www.r-project.org/.

Next, load the functions contained in the file "coincidence_functions.R" into R. To do this, save "coincidence_functions.R" to your computer. Then open R and select File->Source R code, and select the file "coincidence_functions.R".

2: R functions

If you understand the functions in "coincidence_functions.R" you can perform several useful analyses with just a few lines of computer code. The functions are sufficient to perform the analyses in the manuscript, but they are not written or tested to cover all possible analyses. Illogical requests or unsupported analyses may encounter errors, crashes, or bugs.

2.1: `sim.data.foo`

Description: simulates the locations of social animals through time in a population of interest.

Usage:
```
sim.data.foo(days=20, colony=100, trees=20, attract=800, base=20,
cliques=1, repel=0, stay=0, BI=0, method="log")
```

Arguments:

`days`	number of time periods to be simulated
`colony`	number of animals in the population
`trees`	number of potential locations for the population
`attract`	strength of social attraction within a social sub-group
`base`	an arbitrary base score for the social attraction
`cliques`	the number of social sub-groups in the population
`repel`	the social attraction or repulsion between different social sub-groups
`stay`	the attraction to the previously occupied location
`BI`	burn-in. If `stay`>0, the simulation will begin in disequilibrium. A `BI`>0 (maybe 10 or 20) gives the simulation time to achieve equilibrium, but increases run time
`method`	affects a formula used to calculate social attraction. Can be set to "`log`" or "`linear`"

Value: Generally, it is not necessary to review the output of this function. It is a list that includes large arrays that describe the location of animals in the population, as well as the original arguments to the function.

2.2: `summ.data.foo`

Description: summarizes the output of `sim.data.foo`.

Usage: `summ.data.foo(data)`

Arguments:

`data`	output of `sim.data.foo`

Value: Returns a list that includes the mean and range for the number of groups formed by the simulated population and the mean and range of the size of those groups.

2.3: `est.N.foo`

Description: estimates the size of the population using the coincidence rate method

Usage: `est.N.foo(data, num.bats=4)`

Arguments:

`data`	output of `sim.data.foo`
`num.bats`	number of radio-tagged animals

Value: Returns a list that includes the estimated abundance, the number of radio-tagged animals, and the identity of the radio-tagged animals.

2.4: `boot.data.foo`

Description: uses a bootstrap procedure to estimate a confidence interval on abundance

Usage: `boot.data.foo(sim.data, N.data, strap=1000)`

Arguments:

`sim.data`	output of `sim.data.foo`
`N.data`	output of est.`N.foo`

`strap` number of times to repeat bootstrap

Value: Returns a list that includes every bootstrapped estimate of abundance.

2.5: `fager.foo`

Description: if there are two social sub-groups, calculates how often animals associate with individuals in their own sub-group and the other sub-group. Only works with two sub-groups.

Usage: `fager.foo(sim.data)`

Arguments:

`sim.data` output of `sim.data.foo` function

Value: Returns a list that includes a (rather large) matrix of associations, how often individuals from each sub-group associate with their own sub-group (c11 and c22) or the other sub-group (c12 and c21) and the ratio of these numbers.

2.6: `move.foo`

Description: estimates the percent of a population that moves to a new location each time period.

Usage: `move.foo(sim.data)`

Arguments:

`sim.data` output of `sim.data.foo` function

Value: Returns a list that includes an estimate of the percent of the population that moved to a new location each time period.

2.7: `pow.analysis`

Description: conducts analysis on simulated data to estimate the power to detect a treatment effect given a certain survey effort.

Usage:
```
pow.analysis(sites=1, days=20, control=100, treat=40, trees=20, num.bats=4,
attract=800, reps=300, rand=300, base=20, cliques=1, repel=0, stay=0, BI=0,
method="log")
```

Arguments:

`sites` number of study sites per treatment
`days` number of time periods to be simulated
`control` average number of animals at each control site
`treat` treatment effect given as number of animals
`trees` number of potential locations for the population
`num.bats` number of radio tagged animals at each study site
`attract` strength of social attraction within a social sub-group
`reps` number of different studies to simulate
`rand` within each simulated study, number of randomizations used to test for significance of treatment
`base` an arbitrary base score for the social attraction
`cliques` the number of social sub-groups in the population

`repel`	the social attraction or repulsion between different social sub-groups
`stay`	the attraction to the previously occupied location
`BI`	burn-in. If `stay`>0, the simulation will begin in disequilibrium. A `BI`>0 (maybe 10 or 20) gives the simulation time to achieve equilibrium, but increases run time
`method`	affects a formula used to calculate social attraction. Can be set to "`log`" or "`linear`"

Value: Returns a list that includes one- and two-tailed estimates of power as well as the arguments used in the analysis.

Notes: Assumes one type of treatment and one type of control. `sites` is assumed to be equal for treatment and control, so that 's' sites will analyze a study with 's' control sites and 's' treatment sites. `num.bats` is assumed equal for all study sites, so that a study with 'nb' animals and 's' sites will have a total of nb*2*s radio tagged animals. The default settings [`pow.analysis()`] take 4.5 minutes to run on a Dell E6540 with an Intel Core i7-4800MQ processor.

2.8: `field.N.foo`

Description: estimates the size of the population using the coincidence rate method. See section 4 for an example.

Usage: `field.N.foo(bat.loc, grp.size)`

Arguments:

`bat.loc`	Three dimensional array that summarizes location of radio-tagged animals through time. Length of first dimension equals the number of surveys; length of the second dimension equals the number of radio-tagged animals; length of the third dimension equals the number of known animal locations. Value in array is 1 if animal is present, 0 if absent
`grp.size`	matrix that gives daily group size for the radio-tagged animals. Length of first dimension equals the number of surveys; length of the second dimension equals the number of known animal locations. Value in matrix is group size, or NA if unknown

Value: Returns estimated abundance.

2.9: `field.boot.foo`

Description: uses a bootstrap procedure to estimate a confidence interval on abundance

Usage: `field.boot.foo(bat.loc, grp.size, strap=1000)`

Arguments:

`bat.loc`	Three dimensional array that summarizes location of radio-tagged animals through time. Length of first dimension equals the number of surveys; length of the second dimension equals the number of radio-tagged animals; length of the third dimension equals the number of known animal locations. Value in array is 1 if animal is present, 0 if absent
`grp.size`	matrix that gives daily group size for the radio-tagged animals. Length of first dimension equals the number of surveys; length of the second dimension equals the number of known animal locations. Value in matrix is group size, or NA if unknown
`strap`	number of times to repeat bootstrap

Value: Returns a list that includes every bootstrapped estimate of abundance.

3: Example analysis of simulated data

Below, we give pared-down examples of the analyses in the manuscript. We recommend studying the function descriptions in section 2, prior to reading this section. The functions are moderately flexible, so that analysis can quickly be extended along the parameters we considered, such as population size or number of ratio-tagged animals. However, the functions are not written to cover all possible analyses. Illogical requests or unsupported analyses may encounter errors, crashes, or bugs.

3.1: Number and size of social groups

We simulate the behavior of individual animals, and social groups are an emergent property of individual behavior. Therefore, it can be difficult to predict the number and size and social groups from the function arguments. Here, we estimate the number and size of social groups, as in Table 2 in the manuscript.

For this analysis, simulate data with sim.data.foo and then summarize it with sum.data.foo. Set days to a large number so we have a large sample of days for generating the summary. Approximate run time = 0.4 minutes.

```
sim.data      <- sim.data.foo(days=3000, colony=100, trees=20, attract=800, base=20,
cliques=1, repel=0, stay=0, BI=0, method="log")
example       <- summ.data.foo(sim.data)
example
```

3.2: Accuracy of abundance estimates

We would like to know if abundance estimates are accurate, using simulated data. Here, we simulate data many times (`sims=3000`), estimate abundance, and compare this to known abundance (`colony=100` in this example). This type of analysis (with additional values for `days` and `num.bats`) was used to construct Figure 1. Similar analyses can be used for different scenarios (`colony=200, attract=1600`, etc). Approximate run time = 7.6 minutes.

```
sims          <- 3000
test.N        <- 1:sims
for (i in 1:sims) {
       test          <- sim.data.foo(days=20, colony=100, trees=20, attract=800,
              base=20, cliques=1, repel=0, stay=0, BI=0, method="log")
       test.N[i]     <- est.N.foo(test, num.bats=4)$N
}
(median.example     <- median(test.N))
(ci.example         <- quantile(test.N, probs=c(0.025,0.975)))
(mean.example       <- mean(test.N))    # watch out for infinity
```

3.3: Coverage

If we estimate a confidence interval on abundance, is this likely to include true abundance? We use a bootstrap procedure to estimate the confidence interval and compare to true abundance. Approximate run time = 37 minutes.

```
sims          <- 3000             # how many times we repeat the simulation
num.boots     <- 1000             # how many bootstrap operations within each sim
sim.N         <- 1:sims                         # vector to hold output
boot.N        <- matrix(0, num.boots, sims)     # matrix to hold output
for (i in 1:sims) {
       test          <- sim.data.foo(days=20, colony=100, trees=20, attract=800,
              base=20, cliques=1, repel=0, stay=0, BI=0, method="log")
       test.N        <- est.N.foo(test, num.bats=4)
       sim.N[i]      <- test.N$N
```

```
      boot.N[,i]    <- boot.data.foo(test, test.N, num.boots)$N.strap
}

ci.example2         <- apply(boot.N, 2, quantile, probs=c(0.025,0.975))
coverage.example2   <- mean(ci.example2[1,]<100 & ci.example2[2,]>100)
coverage.example2
```

3.4: Association ratios

If there are two social sub-units, we may want to know how often an animal associates with members of its own sub-unit, relative to the other one. Because this is an emergent property of individual behavior, it takes a little analysis to estimate. We calculate how often an animal associates with each sub-unit and then report the ratio. Approximate run time = 0.8 minutes.

```
test          <- sim.data.foo(days=3000, colony=100, trees=20, attract=800, base=20,
      cliques=2, repel=-21, stay=0, BI=0, method="log")
test.fager    <- fager.foo(test)
test.fager$ratio
```

3.5: Probability of moving

If `stay=0`, the probability of movement will equal (t-1)/t where t is `trees`, the number of locations available for the animals. If `stay`>0, the probability of movement will be reduced. This code estimates that probability. Approximate run time = 0.5 minutes.

```
test   <- sim.data.foo(days=3000, colony=100, trees=20, attract=3200, base=20,
      cliques=1, repel=0, stay=26500, BI=0, method="log")
test.move     <- move.foo(test)
test.move
```

3.6: Power analysis

The following code can be used as part of a power analysis. Arguments describe various assumptions about the biology, while `reps` and `rand` determine how many simulations to run. It outputs one- and two-tailed power estimates. Approximate run time = 18 minutes.

```
test.PA       <- pow.analysis(sites=4, days=20, control=100, treat=40, trees=20,
      num.bats=4, attract=800, reps=300, rand=300, base=20, cliques=1, repel=0,
      stay=0, BI=0, method="log")
test.PA$power
test.PA$power.2tail
```

4: Example analysis of field data

This section begins with some hypothetical field data and demonstrates an analysis

4.1: Format data

Imagine we have telemetry data and count data in this format:

	Animal 1		Animal 2		Animal 3	
Date	**Location**	**Count**	**Location**	**Count**	**Location**	**Count**
Day 1	L1	40	L1	40	L1	40
Day 2	L1	45	L1	45	L4	20
Day 3	L2	75	L4	20	L2	75

Day 4	L2	70	L4	25	L2	70
Day 5	L3	15	L4	55	NA	NA
Day 6	L3	15	L4	41	L2	22

Note that four unique locations were found and that Animal 3 was not located on Day 5. We need to organize the count data into a Date by Location table, shown below.

Date	L1	L2	L3	L4
Day 1	40	NA	NA	NA
Day 2	45	NA	NA	NA
Day 3	NA	75	NA	20
Day 4	NA	70	NA	25
Day 5	NA	NA	15	55
Day 6	NA	22	15	41

Note that the count data has NA any time a location was not counted. This can be entered into R by typing in the values going down each column, starting with the left column, and then telling R how many rows and columns there are:

```
cnt.data          <- matrix(c(40,45,NA,NA,NA,NA,NA,75,70,65,NA,NA,NA,
NA,NA,15,15,NA,NA,20,25,55,50), nrow=6, ncol=4)
```

We also need similar Date by Location tables for each animal, in this case three matrices. These matrices have a 1 if the animal was present, a 0 if it was absent, and an NA if the location was unknown.

ANIMAL 1	L1	L2	L3	L4
Day 1	1	0	0	0
Day 2	1	0	0	0
Day 3	0	1	0	0
Day 4	0	1	0	0
Day 5	0	0	1	0
Day 6	0	0	1	0

ANIMAL 2	L1	L2	L3	L4
Day 1	1	0	0	0
Day 2	1	0	0	0
Day 3	0	0	0	1
Day 4	0	0	0	1
Day 5	0	0	0	1
Day 6	0	0	0	1

ANIMAL 3	L1	L2	L3	L4
Day 1	1	0	0	0
Day 2	0	0	0	1
Day 3	0	1	0	0
Day 4	0	1	0	0
Day 5	NA	NA	NA	NA
Day 6	0	1	0	0

This location information can be entered into R in the same way:

```
ind1        <- matrix(c(1,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0),
nrow=6, ncol=4)

ind2        <- matrix(c(1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1,1),
nrow=6, ncol=4)

ind3        <- matrix(c(1,0,0,0,NA,0,0,0,1,1,NA,1,0,0,0,0,0,NA,0,0,0,0,0,NA,0),
nrow=6, ncol=4)
```

The location data can then be combined into a single array by row binding the data for each animal, and then telling R the number of survey days (6), the number of radio-tagged animals (3), and the number of known locations (4):

```
loc.data    <- array(rbind(ind1,ind2,ind3), dim=c(6,3,4))
```

4.2: Analyze data

We can then estimate abundance by entering the formatted data into the field.N.foo function.

```
(abund      <- field.N.foo(loc.data, cnt.data))
```

We can also use field.boot.foo to use bootstrapping to characterize the precision of the estimate and generate a confidence interval:

```
out <- field.boot.foo(loc.data, cnt.data, 1000)

hist(out)
quantile(out, c(0.025,0.975), na.rm=T)
mean(out, na.rm=T)
mean(out[-which(out==Inf)], na.rm=T)
median(out, na.rm=T)
```