Sociality Simulation Outline & To-Do

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.2.1 --
```

```
## v ggplot2 3.0.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.6
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Colony parameters:

1. Birth Ratio -> tendency to create reproductive or workers
    - Seasonal
    - Population, number of reproductives and workers
    -
2. When Queen stops foraging
    - Number of workers
3.

Evolving paremeters

1. Worker body size (energy required to create)

Environmental parameters:

1. Forager mortality rate
2. Forager success (resource availability)

Seasonal changes

Interesting questions:

Future additions:

1. Add brood care or growth time

```
#Whether to birth a reproductive or a worker
#TRUE for reproductive, FALSE for worker
birthReprod <- function(nReprod,nWorker,Day,nDayCycle)
{
  temp <- 1-(Day/nDayCycle);
  temp2 <- runif(1,0,1) > .5
  return(temp>temp2);
```

```
}

#Set foraging mortality rate given day in cycle
fMortRate <- function(Day,nDayCycle)
{
  temp <- .65+(Day/nDayCycle-.5)^2;
  return(temp);
}

#Set parameters for when reproductives stop foraging
isReprodForage <- function(nWorker)
{
  return(ifelse(nWorker<2,TRUE,FALSE))
}
```

```
#Set number of Days in a Cycle
Day <- 0; nDayCycle <- 20;
#Starting number of reproductives, workers and energy stores
nReprod <- 1; nWorker <- 0; kStore <- 0;
#Number of trips that workers and reproductives can make each day
nTripReprod <- 2; nTripWorker <- 2.5;
#Set amount of energy per trip that workers and reproductives make
kTripWorker <- 2; kTripReprod <- 2;
#Set amount of energy needed to create a worker and reproductive
kCreateWorker <- 3; kCreateReprod <- 3.5;

#Create dataframe to store data.
data <- data.frame(timestep=0,nReprod=nReprod,nWorker=nWorker,kStore=kStore)

for (i in 1:nDayCycle)
{
  #Forage as much as you can/need with workers
  #!!!!! When does Queen/reproductives stop foraging?
  #!!!!! How to incorporate mortality rate?

  nWorker <- (nWorker*fMortRate(Day,nDayCycle));
  kStore <- kStore + nWorker * nTripWorker;

  if(isReprodForage(nWorker=nWorker))
  {
    nReprod <- (nReprod*fMortRate(Day,nDayCycle));
    kStore <- kStore + nReprod * nTripReprod;
  }

  while (kStore > 0)
  {
    if(kStore > kCreateReprod)
    {
      if(birthReprod(nReprod,nWorker,Day,nDayCycle))
        {kStore <- kStore - kCreateReprod;  nReprod <- nReprod+1;}
        else
        {kStore <- kStore - kCreateWorker;  nWorker <- nWorker+1;}
    }
    else{
```

```
      if(kStore > kCreateWorker)
      {kStore <- kStore - kCreateWorker;  nWorker <- nWorker + 1;}
      else{break}
    }
  }

  if(i%%5==0){print(paste("Day: ",i,"  kStore: ",kStore,"  nReprod: ",nReprod,"  nWorker: ",nWorker));}
  data <- rbind(data,list(i,nReprod,nWorker,kStore))
}
```

```
## [1] "Day:  5   kStore:  0.312700000000001   nReprod:  1.6561   nWorker:  4.439"
## [1] "Day:  10   kStore:  2.34878502500001   nReprod:  11.6561   nWorker:  12.32628511"
## [1] "Day:  15   kStore:  0.29911787141225   nReprod:  44.6561   nWorker:  41.0840480946039"
## [1] "Day:  20   kStore:  2.96525991389024   nReprod:  161.6561   nWorker:  122.432219559383"
```

```
ggplot(data,aes(x=timestep)) + geom_line(aes(y=kStore,color="kStore")) + geom_line(aes(y=nReprod,color=
```