Metrics for Recorder behaviour

Tom August

14 February 2017

Metrics

We are going to split metric into three broad groups: *Engagement profile*, *Spatial*, and *Taxanomic*

Temporal Metrics

These metrics measure the recording pattern across time such as the number of days that a recorder produces records. These have been termed engagement profiles by others., The metrics here are from Ponciano and Brasileiro 2014 who used the metrics on participant of zooniverse projects. The metrics were also used by Boakes *et al* 2016.

Summer period

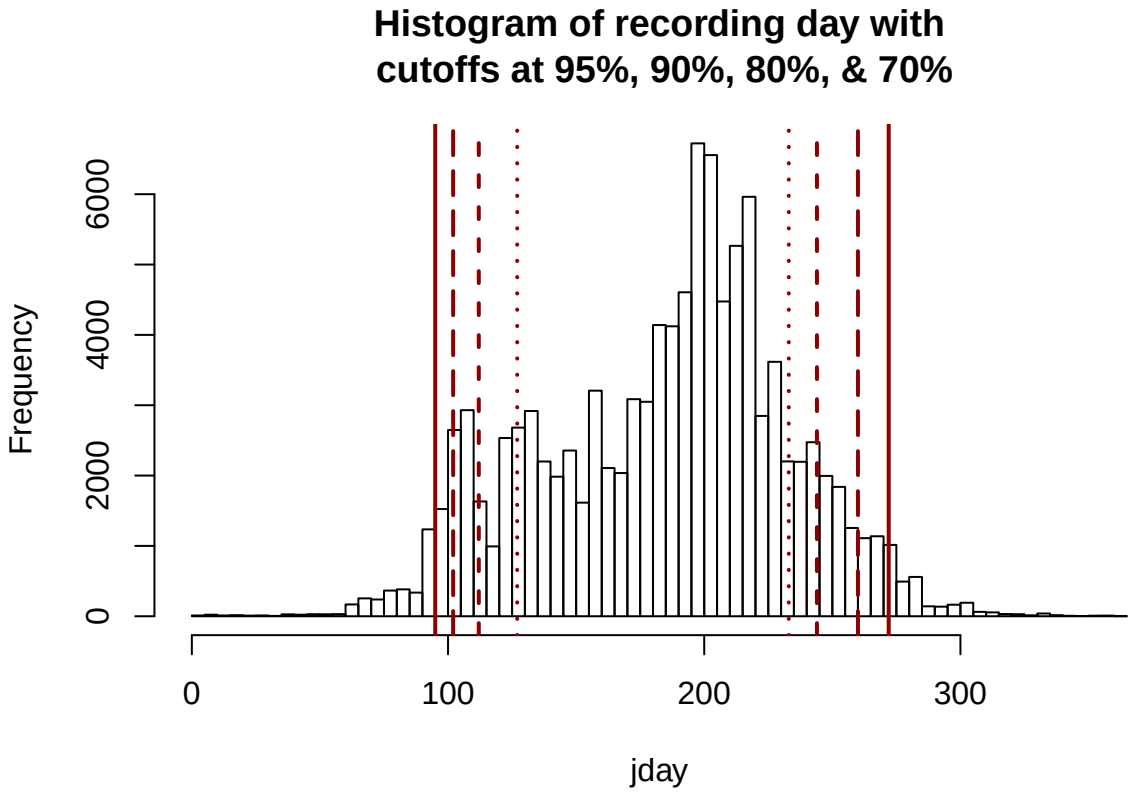
One issue that we have across these metrics and some others is that recording is not consistent across the year and so there can be issues with the numbers generated. To address this the data can be subset to only the summer period, when recorders are active. This period needs to be defined in such a way that the same method can be used across taxanomic groups and will be robust to changes in the start and end of teh summer period from year to year.

I suggest we use a percentage cut off, for example take the period of the year that contains 90% of the data. Lets have a look at how this might work

```
Sys.time()
```

```
## [1] "2016-12-07 11:36:19 GMT"
```

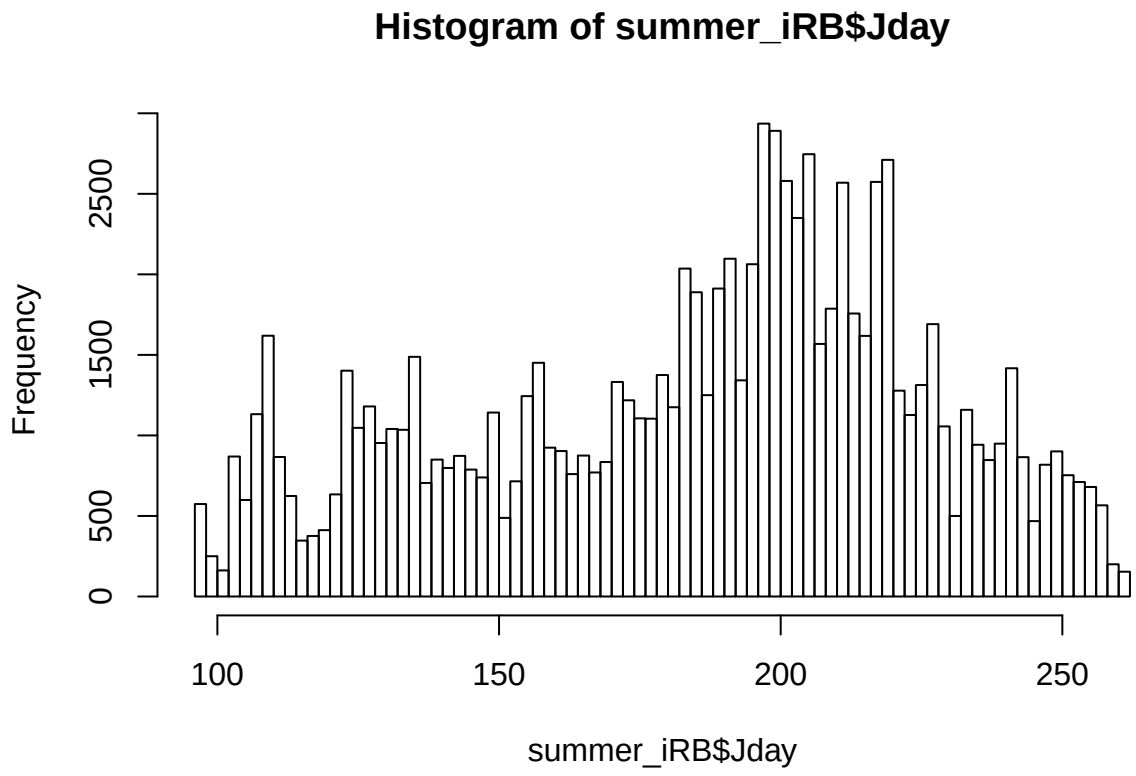
```
# First lets have a look at the distribution of
# records through the year
jday <- as.POSIXlt(iRB$date_start)$yday

# Where are the limits of different percentiles
p95 <- quantile(jday, c(0.025,0.975))
p90 <- quantile(jday, c(0.05,0.95))
p80 <- quantile(jday, c(0.1,0.9))
p70 <- quantile(jday, c(0.15,0.85))

hist(jday, 100, main = paste('Histogram of recording day with\n',
                             'cutoffs at 95%, 90%, 80%, & 70%'))
abline(v = p95, lty = 1, col = 'darkred', lwd = 2)
abline(v = p90, lty = 5, col = 'darkred', lwd = 2)
abline(v = p80, lty = 2, col = 'darkred', lwd = 2)
abline(v = p70, lty = 3, col = 'darkred', lwd = 2)
```



It looks like 90% might be a good value to go for in this case. We would then need a function that could create these values for each year of the data and throw out data that was outside the summer periods

```
summerData <- function(data, probs = c(0.05, 0.95),
                       date_col = 'date_start'){

  # check date column
  if(!inherits(data[, date_col], 'Date')){
    stop('Your date column is not a date')
  }

  # create J-day column
  data$Jday <- as.POSIXlt(data[,date_col])$yday

  # create year column
  data$year <- as.POSIXlt(data[,date_col])$year+1900

  # create summer column
  data$summer <- FALSE

  qsf <- qsl <- NULL

  # now for each year loop through and create an
  # index column
  for(i in sort(unique(data$year))){

    year_quantiles <- quantile(data$Jday[data$year == i], probs = probs)
```

```
    qsf <- c(qsf, year_quantiles[1])
    qsl <- c(qsl, year_quantiles[2])
    data$summer[data$Jday >= year_quantiles[1]
                & data$Jday <= year_quantiles[2]
                & data$year == i] <- TRUE
  }

  summer_data <- data[data$summer, ]
  attr(summer_data, 'cutoffs') <- data.frame(year = sort(unique(data$year)),
                                             quantile_first = qsf,
                                             quantile_last = qsl)
  return(summer_data)
}

summer_iRB <- summerData(data = iRB,
                         probs = c(0.05, 0.95),
                         date_col = 'date_start')

# Look at the 'cut' data
hist(summer_iRB$Jday, 100)
```




```
# Here are the cuts
attr(summer_iRB, 'cutoffs')
```

```
##   year quantile_first quantile_last
## 1 2014            104           260
```

```
## 2 2015             97           261
## 3 2016            107           258
```

Activity ratio

"The proportion of days on which the volunteer was active in relation to the total days he/she remained linked to the project" (Ponciano and Brasileiro 2014)

```
# Create a function to calculate activity ratio
activityRatio <- function(recorder_name,
                          data,
                          recorder_col = 'recorders',
                          date_col = 'date_start'){

  # check date column
  if(!inherits(data[, date_col], 'Date')){
    stop('Your date column is not a date')
  }

  # Get the recorders data
  data <- data[data[,recorder_col] == recorder_name, ]

  # Some people might have no data from the summer period
  if(nrow(data) < 1){

    return(data.frame(recorder = recorder_name,
                      activity_ratio = NA,
                      total_duration = NA,
                      active_days = NA))
  } else {

  # Get unique dates
  dates <- unique(data[,date_col])

  # Get the first and last date
  first_last <- range(dates)

  # Total duration of this recorder
  duration <- as.numeric(first_last[2] - first_last[1]) + 1

  # calculate ratio
  activity_ratio <- length(dates)/duration

  # return
  return(data.frame(recorder = recorder_name,
                    activity_ratio = activity_ratio,
                    total_duration = duration,
                    active_days = length(dates)))
  }
}

# Test on David and Tom
activityRatio(data = summer_iRB, recorder_name = 'Roy, David')
```

```
##     recorder activity_ratio total_duration active_days
## 1 Roy, David      0.1497696            868         130
```

```
activityRatio(data = summer_iRB, recorder_name = 'August, Tom')
```

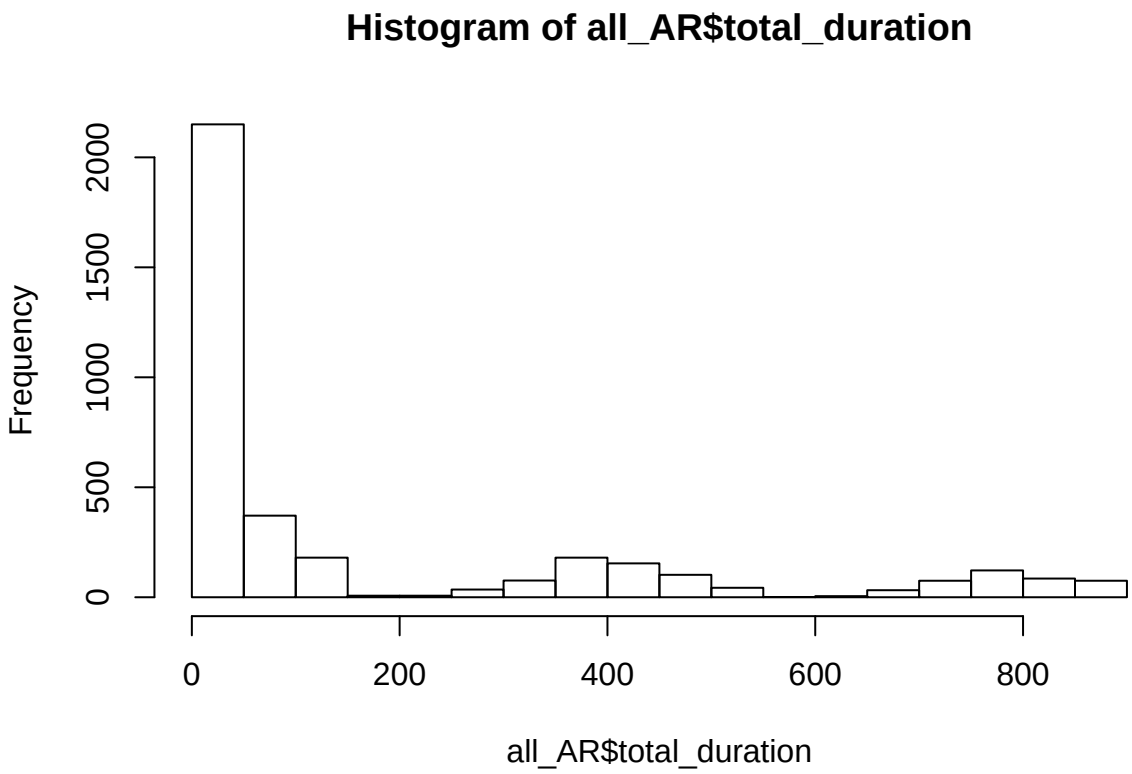
```
##      recorder activity_ratio total_duration active_days
## 1 August, Tom     0.01587302            441           7
```

```
## David is a more active recorder than Tom ##

# Run for everyone
all_AR <- do.call(rbind, parLapply(cl, X = unique(iRB$recorders),
                                   fun = activityRatio,
                                   data = summer_iRB))

# Lets have a look at some of these
head(all_AR, 20)
```

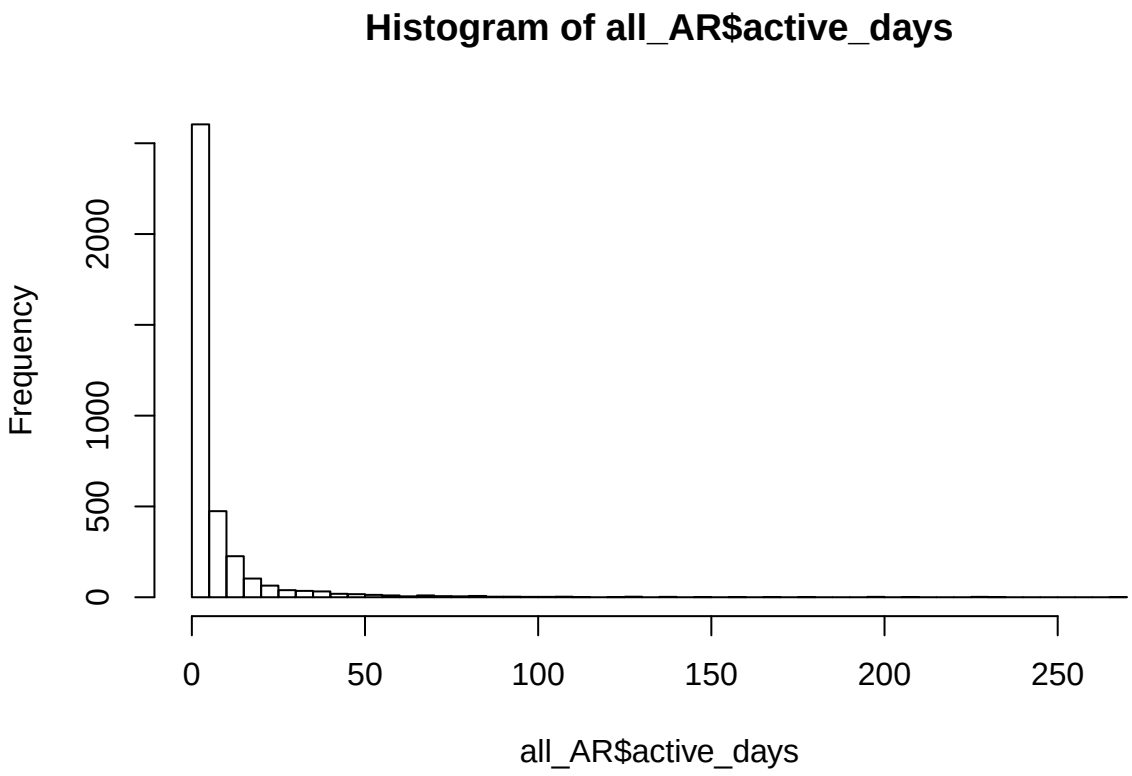
```
##                recorder activity_ratio total_duration active_days
## 1           Marsh, Nick     0.11450382            131          15
## 2             Limb, Ken     0.18595041            484          90
## 3            Ward, John     0.18181818             77          14
## 4         Hughes, Peter     0.22580645             31           7
## 5       Turner, Lindsey     0.11607143            336          39
## 6        Warren, Martin     0.20090806            881         177
## 7        Newbould, John     0.22727273            880         200
## 8            fenn, paul     0.16746411            836         140
## 9            Cox, Steve     0.19047619            882         168
## 10        Lewis, Steven     0.10432570            786          82
## 11         Anstie, John     0.17647059             34           6
## 12        Austin, David     0.13785311            885         122
## 13         Bowles, Nick     0.08113590            493          40
## 14         Binks, Rosie     0.19753086            486          96
## 15        Dean, Michael     0.07769784            695          54
## 16      Watkins, nicola     0.23913043             46          11
## 17         Kilbey, Dave     0.09239130            736          68
## 18        Dawson, John      0.25490196             51          13
## 19 rouncefield, marlene     0.05569007            413          23
## 20       Raymond, Colin     0.02816901            852          24
```

I think this metric tells a story in a combination of the ratio and the total number of days. I think the ratio means more when the recorder has been recording for a long duration

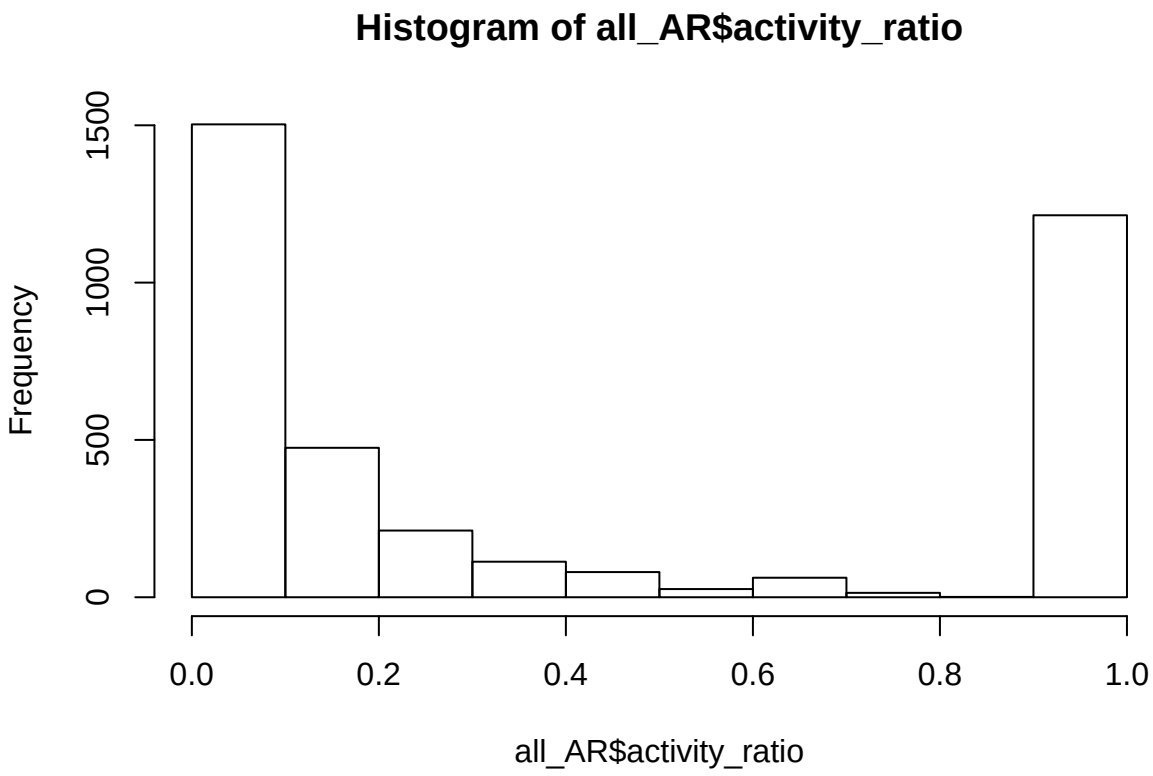
```
# Have a look at the distribution of these metrics
# There looks like there could be an effect of year
hist(all_AR$total_duration, breaks = 30)
```




```
hist(all_AR$active_days, breaks = 80)
```

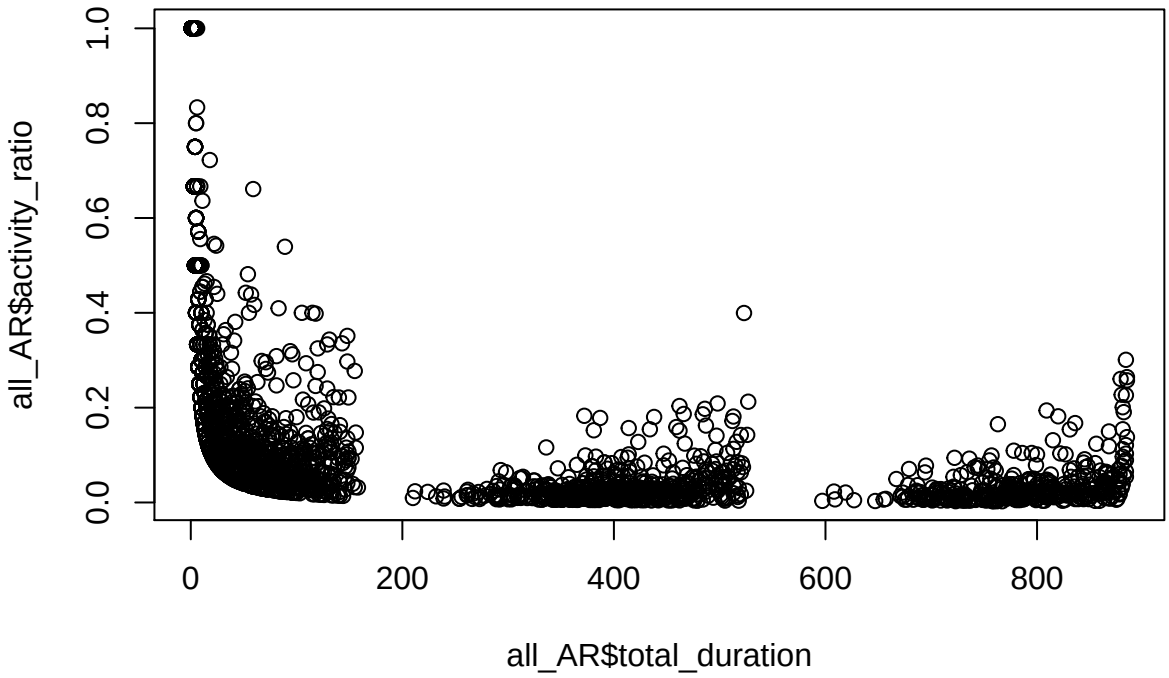



```
hist(all_AR$activity_ratio)
```

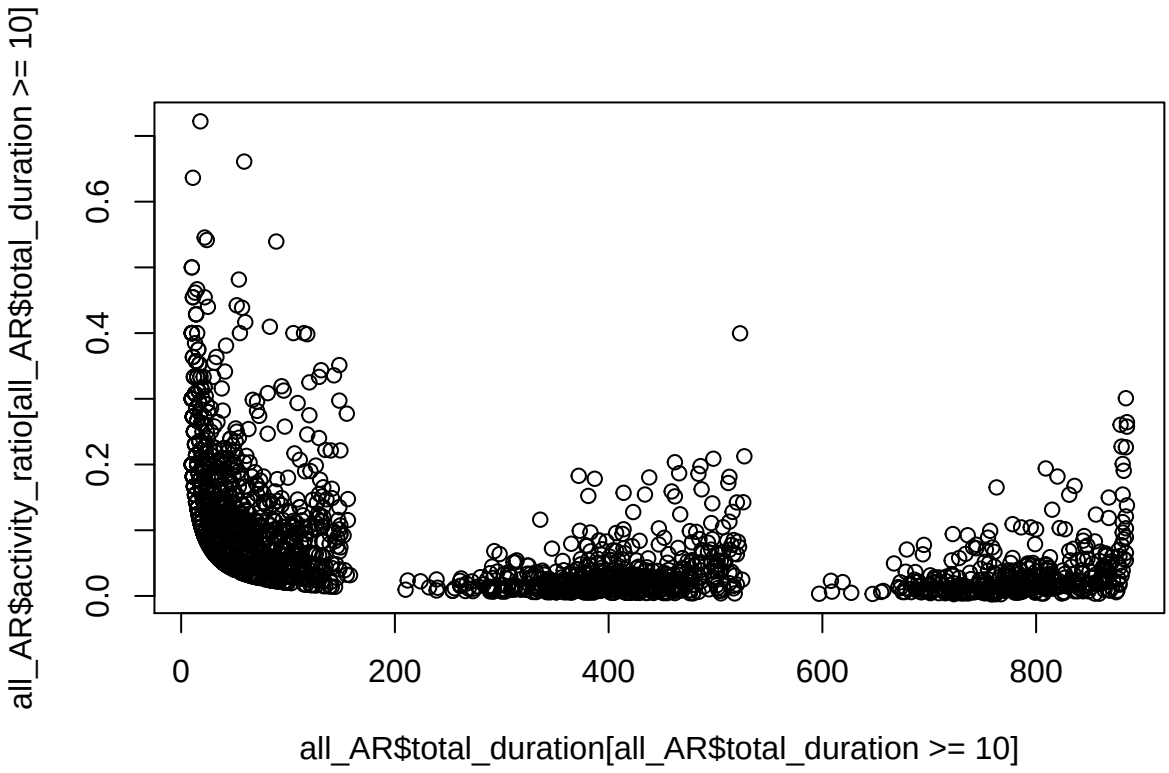


Both have nice distributions, though we can see the single record people in the ratio plot

```
# Plot activity_ratio against duration
# Here we probably want to subset to avoid bias
plot(all_AR$total_duration,
     all_AR$activity_ratio)
```




```
plot(all_AR$total_duration[all_AR$total_duration >= 10],
     all_AR$activity_ratio[all_AR$total_duration >= 10])
```



Weekly devoted days

This is an adaptation of the *Daily Devoted Time* in (Ponciano and Brasileiro 2014) which is clearly not applicable to biological recording. Though Boakes *et al* 2016 don't attempt to use this measure I think the idea can be adapted by using days in a week (summer only) rather than hours in a day.

```
# Create a function
weeklyDevotedDays <- function(recorder_name,
                              data,
                              recorder_col = 'recorders',
                              date_col = 'date_start'){

  # check date column
  if(!inherits(data[, date_col], 'Date')){
    stop('Your date column is not a date')
  }

  # Get the recorders data
  data <- data[data[,recorder_col] == recorder_name, ]

  # Get unique dates as dates
  dates <- unique(data[,date_col])

  # Get all week_year combinations
  week_year <- paste(strftime(as.POSIXlt(dates), format = '%W'),
```

```
                    format(dates, '%Y'), sep = '_')

  # here are the counts
  week_counts <- table(week_year)

  # As these are counts taking the median is probably best
  weekly_devoted_days <- median(week_counts)

  return(data.frame(recorder = recorder_name,
                    median_weekly_devoted_days = weekly_devoted_days,
                    n_weeks = length(week_counts),
                    n_recs = sum(week_counts), row.names = NULL))
}

# Test on David and Tom
weeklyDevotedDays(data = summer_iRB, recorder_name = 'Roy, David')
```

```
##     recorder median_weekly_devoted_days n_weeks n_recs
## 1 Roy, David                           2      60    130
```

```
weeklyDevotedDays(data = summer_iRB, recorder_name = 'August, Tom')
```

```
##      recorder median_weekly_devoted_days n_weeks n_recs
## 1 August, Tom                           1       6      7
```

```
## David contributes more of his time than Tom ##

# Run for everyone
all_WDD <- do.call(rbind, parLapply(cl, X = unique(iRB$recorders),
                                    fun = weeklyDevotedDays,
                                    data = summer_iRB))

# Lets have a look at some of these
head(all_WDD, 20)
```

```
##                recorder median_weekly_devoted_days n_weeks n_recs
## 1           Marsh, Nick                         1.0      11     15
## 2             Limb, Ken                         2.0      34     90
## 3            Ward, John                         1.0      11     14
## 4         Hughes, Peter                         1.0       5      7
## 5       Turner, Lindsey                         3.0      11     39
## 6        Warren, Martin                         3.0      60    177
## 7        Newbould, John                         3.0      67    200
## 8            fenn, paul                         3.0      54    140
## 9            Cox, Steve                         3.0      61    168
## 10        Lewis, Steven                         2.0      39     82
## 11         Anstie, John                         1.0       5      6
## 12        Austin, David                         2.0      57    122
## 13         Bowles, Nick                         2.0      22     40
## 14         Binks, Rosie                         2.5      32     96
## 15        Dean, Michael                         2.0      28     54
## 16      Watkins, nicola                         1.5       6     11
## 17         Kilbey, Dave                         2.0      31     68
## 18        Dawson, John                          2.0       6     13
## 19 rouncefield, marlene                         2.0      10     23
```

```
## 20       Raymond, Colin                          1.0      18      24
```

Clearly this metric is only really reliable when we have multiple weeks worth of data for an individual.

Relative activity duration

This is a metric from Ponciano and Brasileiro 2014 which is also used in Boakes *et al* 2016 but I don't think can be applied to biological records since there is no official end date for a project: *"The ratio of days during which a volunteer I remains linked to the project in relation to the total number of days elapsed since the volunteer joined the project until the project is over"*

Periodicity

There is a cluster of metrics that could be used to look at aspects of periodicity. The measure used in Ponciano and Brasileiro 2014 is 'variation in periodicity'; *"The standard deviation of the times elapsed between each pair of sequential active days"*. At the same time as calculating this I think there are another couple of metrics that might be of use. First, periodicity itself, i.e. *"The median time elapsed between each pair of sequential active days"*. Secondly, streak length, i.e. *"The average length of sequential active days"*

```
# Create a function to calculate the periodicity metrics
periodicity <- function(recorder_name,
                        data,
                        recorder_col = 'recorders',
                        date_col = 'date_start',
                        day_limit = 5){

  # check date column
  if(!inherits(data[, date_col], 'Date')){
    stop('Your date column is not a date')
  }

  # Get the recorders data
  data <- data[data[,recorder_col] == recorder_name, ]

  # Get unique dates as dates
  dates <- sort(unique(data[,date_col]))

  # we cannot calculate these metrics if people have very few
  # dates on which they record
  if(length(unique(dates)) < day_limit){

    # return
    return(data.frame(recorder = recorder_name,
                      periodicity = NA,
                      periodicity_variation = NA,
                      median_streak = NA,
                      sd_streak = NA,
                      max_streak = NA,
                      n_days = length(unique(dates))))

  } else {

    # Calculate the elapsed days between each date in sequence
    # this needs to be done within years
```

```
    elapses <- NULL

    for(year in unique(format(dates, '%Y'))){

      temp_dates <- dates[format(dates, '%Y') == year]

      # There must be at least 2 dates in a year
      if(length(temp_dates) > 1){
        temp_elapses <- sapply(1:(length(temp_dates)-1),
           FUN = function(x){
             return(as.numeric(temp_dates[x + 1] - temp_dates[x]))
           })

        elapses <- c(elapses, temp_elapses)

      }

    }

    # periodicity calculation
    periodicity <- median(elapses)

    # variation in periodicity
    periodicity_variation <- sd(elapses)

    # average streak length
    # Streaks are IDed by 1's
    non_streak <- length(elapses[elapses > 1])
    streaks <- rle(elapses)
    streaks_1 <- (streaks$lengths[streaks$value == 1]) + 1

    # Combine streaks and non-streaks
    streak_lengths <- c(rep(1, non_streak), streaks_1)

    # calculate ome metrics
    median_streak <- median(streak_lengths)
    sd_streak <- sd(streak_lengths)
    max_streak <- max(streak_lengths)

    # return
    return(data.frame(recorder = recorder_name,
                      periodicity = periodicity,
                      periodicity_variation = periodicity_variation,
                      median_streak = median_streak,
                      sd_streak = sd_streak,
                      max_streak = max_streak,
                      n_days = length(unique(dates))))

  }


}
```

```r
# Test on David and Tom
periodicity(data = summer_iRB, recorder_name = 'Roy, David')
```

```
##     recorder periodicity periodicity_variation median_streak sd_streak
## 1 Roy, David           2              3.206765             1 0.9400054
##   max_streak n_days
## 1          7    130
```

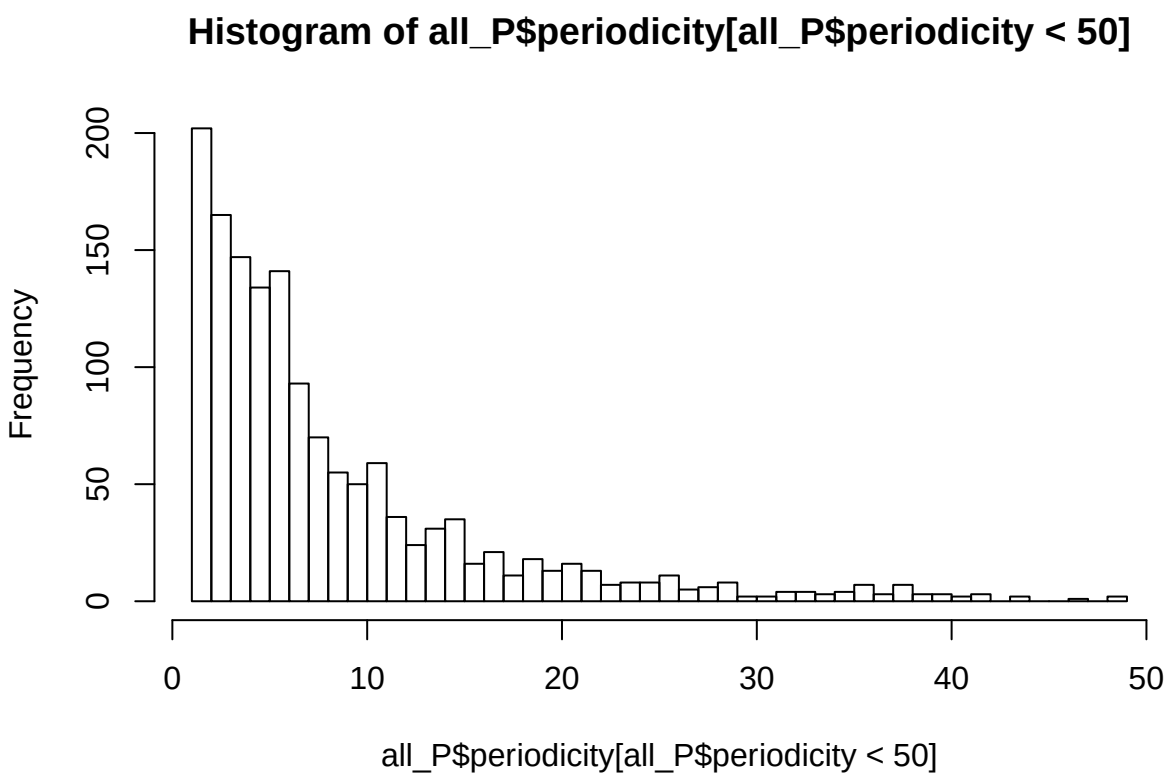
```r
periodicity(data = summer_iRB, recorder_name = 'August, Tom')
```

```
##      recorder periodicity periodicity_variation median_streak sd_streak
## 1 August, Tom          30              17.81853             1 0.4472136
##   max_streak n_days
## 1          2      7
```

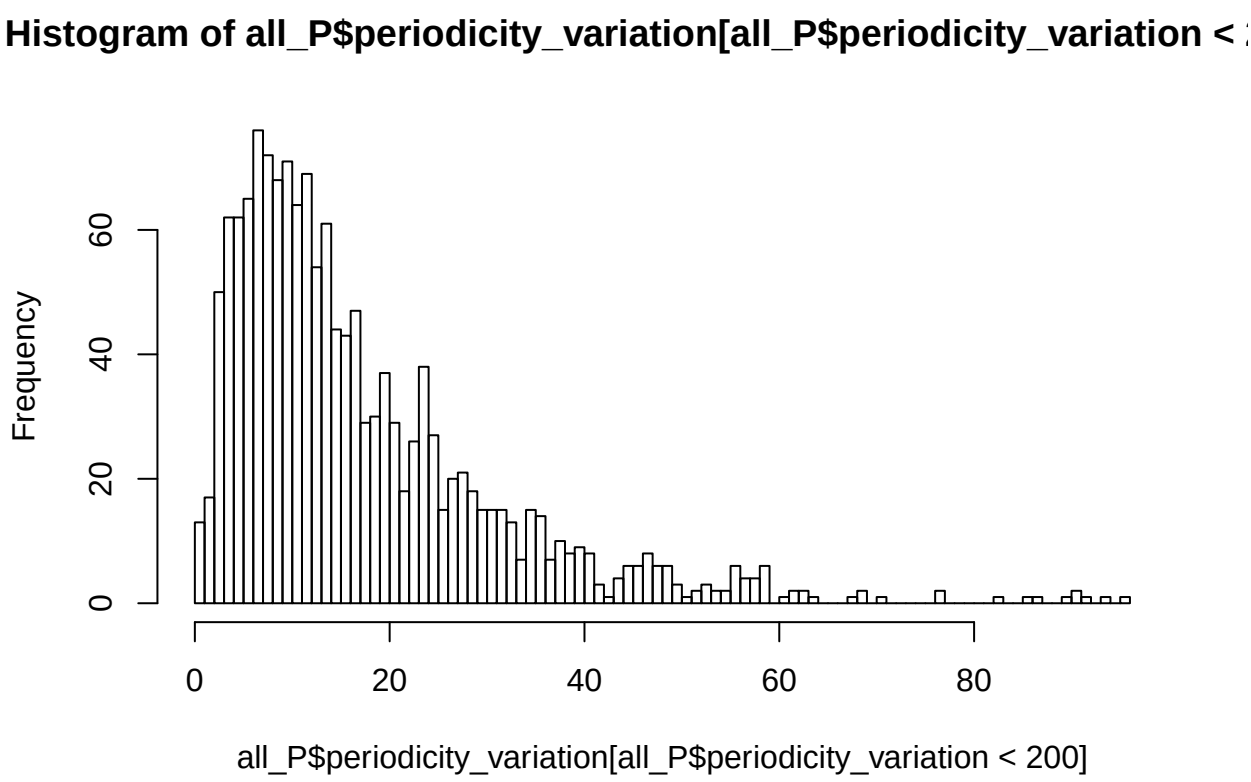
```r
# David is a much more regular recorder than Tom with less
# variation in periodicity and a longer max streak though
# Tom has less days of data to work with

# Run for everyone
all_P <- do.call(rbind, parLapply(cl, X = unique(iRB$recorders),
                                  fun = periodicity,
                                  data = iRB))

# Lets have a look at some of these
head(all_P, 20)[c(5,8,1),]
```

```
##          recorder periodicity periodicity_variation median_streak
## 5 Turner, Lindsey           1              2.506669             1
## 8      fenn, paul           2              5.538564             1
## 1     Marsh, Nick           7             10.973127             1
##   sd_streak max_streak n_days
## 5 2.6770631         14     44
## 8 1.2468806         10    190
## 1 0.5789342          3     16
```

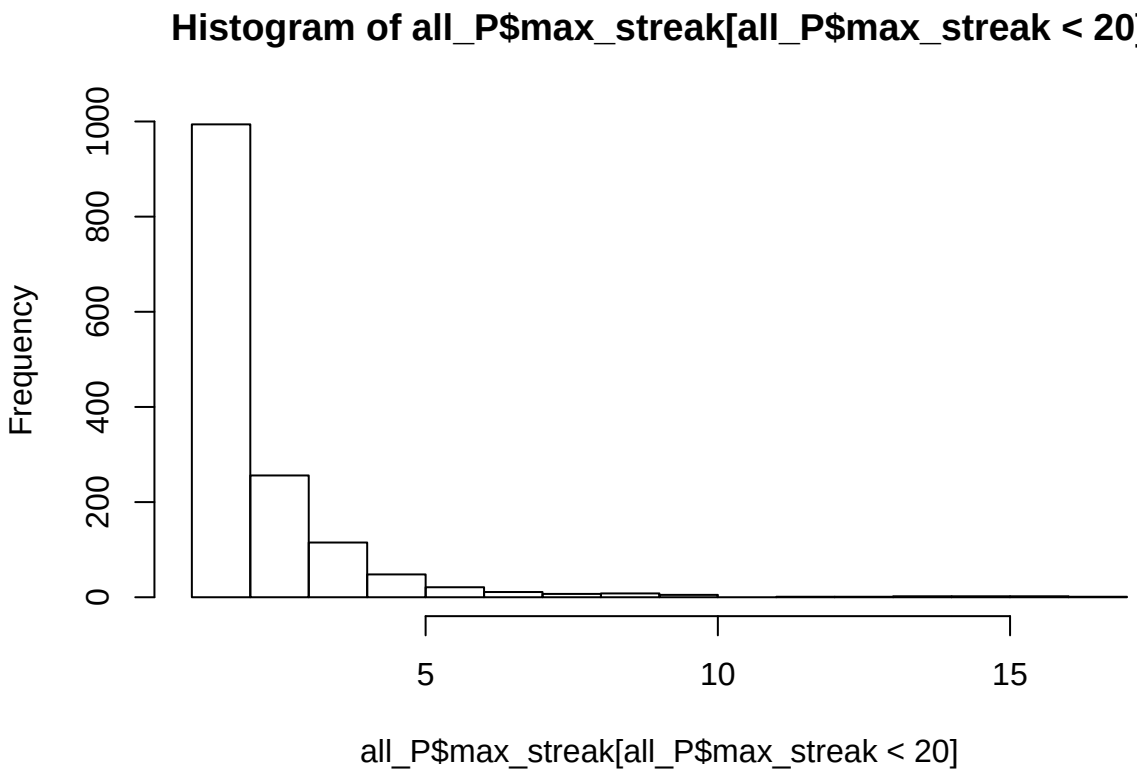
```r
# David a Tam are both very studious recorders with
# long max streaks and very low periodicity.
# Anne is less studious but still has a low periodicity

# Nice poission dist. for periodicity
hist(all_P$periodicity[all_P$periodicity < 50],
     breaks = 50)
```




```
# Nice poission dist. for periodicity_variation (long tail)
hist(all_P$periodicity_variation[all_P$periodicity_variation < 200],
     breaks = 100)
```




```
# Dist. of max_streak
hist(all_P$max_streak[all_P$max_streak < 20],
     breaks = 20)
```



By using the summer data only this analysis seems to be better than an earlier one that included all data. These metrics cannot be calculate for people who have only made one record. I have included a parameter `day_limit` to allow us to set a limit at which we calculate these metrics.

Spatial Meterics

These metrics deal with the spatial distribution of records

Area and heterogenity of recording

I think the first step for all of these metrics is to turn the points into a SpatialPoints object which will allow us to manipulate then more easily. Once we have done that we can calculate MCP (minimum convex polygons) around the points. We might want to change this method to a method that is less susceptible to outliers such as alpha hull (we can talk to Colin about this). Here I use 95% MCP as the total recording area (hopefully removing outliers), and use the ratio of 95%:50% as a measure of heterogeneity.

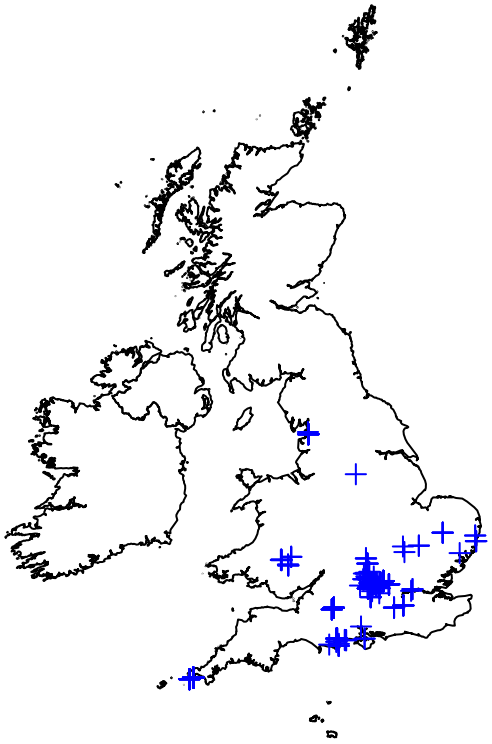
```
# Function takes data and username and returns spatial metrics
spatial_behaviour <- function(data, recorder_name,
                              latitude_col, longitude_col,
                              recorder_col = 'recorders',
                              upper_percentile = 95,
                              lower_percentile = 60,
                              h = 5000,
                              res = 1000){
```

```
if(is.factor(recorder_name)){
  recorder_name <- as.character(recorder_name)
}

n_row <- nrow(iRB[iRB[,recorder_col] == recorder_name, ])

if(n_row >= 5){

  # Convert to SpatialPoints
  spPoints_LL <- sp::SpatialPoints(iRB[iRB[,recorder_col] == recorder_name,
                                       c(longitude_col, latitude_col)])
  # Data is lat long
  proj4string(spPoints_LL) <- CRS("+init=epsg:4326")

  # Convert to Eastings Northings to get meters on X and Y
  spPoint_UK <- spTransform(spPoints_LL, "+init=epsg:27700")

  # set up grid
  # This allows us to ensure there is space for the
  # isoclines to be drawn and that the pixel res is the
  # same - here 1km
  minlong <- floor(bbox(spPoint_UK)['long','min'] - (10*h))
  minlat <- floor(bbox(spPoint_UK)['lat','min'] - (10*h))
  maxlong <- ceiling(bbox(spPoint_UK)['long','max'] + (10*h))
  maxlat <- ceiling(bbox(spPoint_UK)['lat','max'] + (10*h))

  grid_ras <- raster(ext = extent(minlong, maxlong, minlat, maxlat),
                     res = res,
                     crs = projection(spPoint_UK))

                     # matrix(NA,
                     # ncol = ceiling(nlat),
                     # nrow = ceiling(nlong)))

  grid_SP <- as(grid_ras, "SpatialPixels")

  # Try kernel density
  KD <- kernelUD(xy = spPoint_UK, h = h, grid = grid_SP)
  # image(KD)
  KA <- kernel.area(KD,
              percent = c(lower_percentile, upper_percentile),
              unin = "m",
              unout = "km2")
  area_upper <- KA[2]
  area_lower <- KA[1]
  poly_lower <- getverticeshr(KD, percent = lower_percentile)
  poly_upper <- getverticeshr(KD, percent = upper_percentile)
  rm(list = 'KD')
  npolys_upper <- length(poly_upper@polygons[[1]]@Polygons)
  npolys_lower <- length(poly_lower@polygons[[1]]@Polygons)

  # # Calculate the Local convex hull
  # LCH_poly <- LoCoH.k(xy = spPoint_UK,
```

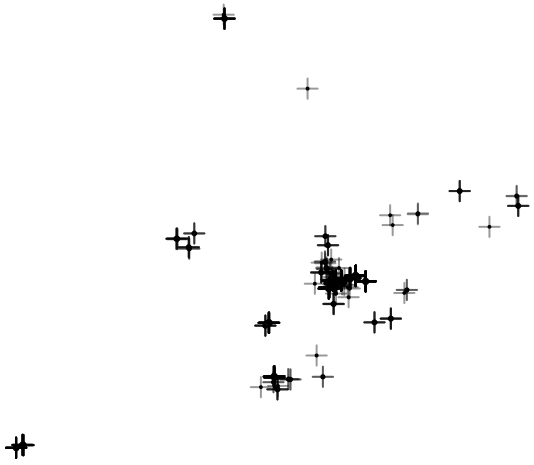
```
    #                       k =  10,
    #                       unin = 'm',
    #                       unout = 'km',
    #                       duplicates = 'remove')
    #
    # # Extract percentiles
    # LCH_MCHu <- spoldf2MCHu(LCH_poly)
    # poly_upper <- getverticeshr.MCHu(LCH_MCHu,
    #                                  percent = upper_percentile)
    # poly_lower <- getverticeshr.MCHu(LCH_MCHu,
    #                                  percent = lower_percentile)
    #
    # npolys_upper <- length(poly_upper@polygons[[1]]@Polygons)
    # npolys_lower <- length(poly_lower@polygons[[1]]@Polygons)
    #
    # area_upper <- MCHu2hrsize(x = LCH_MCHu, percent = upper_percentile,
    #                           plotit = FALSE)
    # area_lower <- MCHu2hrsize(x = LCH_MCHu, percent = lower_percentile,
    #                           plotit = FALSE)

    return(list(recorder = recorder_name,
                spPoint_UK = spPoint_UK,
                # poly = LCH_poly,
                poly_upper = poly_upper,
                poly_lower = poly_lower,
                upper_n_poly = npolys_upper,
                lower_n_poly = npolys_lower,
                upper_area = area_upper,
                lower_area = area_lower,
                ratio = area_lower/area_upper,
                n = n_row))
  } else {
    return(list(recorder = recorder_name,
                spPoint_UK = NA,
                # poly = NA,
                poly_upper = NA,
                poly_lower = NA,
                upper_n_poly = NA,
                lower_n_poly = NA,
                upper_area = NA,
                lower_area = NA,
                ratio = NA,
                n = n_row))
  }

}

# Function for plotting records
plot_ratio <- function(data){
  om <- par("mar")
  omf <- par('mfrow')
  par(mfrow = c(1,2), mar = c(1,1,1,1))
  data(UK)
```

```
  plot_GIS(UK, new.window = FALSE,
           main = 'Distribution of records',
           show.axis = FALSE, show.grid = FALSE)
  points(data$spPoint_UK, pch = 3, col = 'blue')

  # Plot heat map
  plot(data$poly_upper,
       main = paste('\n\n', data$recorder, '-', 'Ratio:',
                    round(data$ratio, 4), '\n',
                    'Upper/lower polygons:', data$upper_n_poly,
                    '/', data$lower_n_poly, '\n',
                    'Total Area:', data$upper_area),
       col = 'grey')
  plot(data$poly_lower, add = TRUE,
       col = 'red', border = 'red')
  points(data$spPoint_UK, col = rgb(0,0,0,0.4),
         pch = 3)
  par(mfrow = omf,
      mar = om)
}

for(h in c(1000, 5000, 10000)){
  DD <- spatial_behaviour(data = iRB, recorder_name = 'Roy, David',
                          latitude_col = 'lat', longitude_col = 'long',
                          h = h)
  plot_ratio(data = DD)

}
```

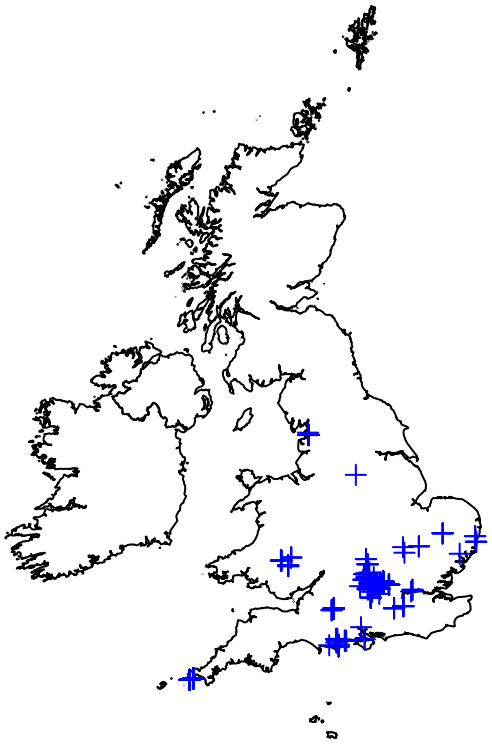
Distribution of records



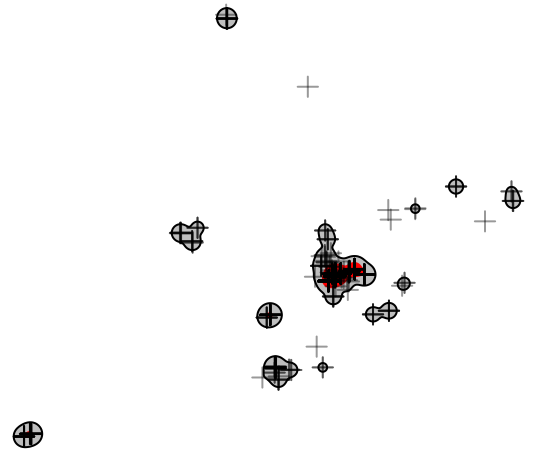
Roy, David − Ratio: 0.1118
Upper/lower polygons: 42 / 10
Total Area: 662

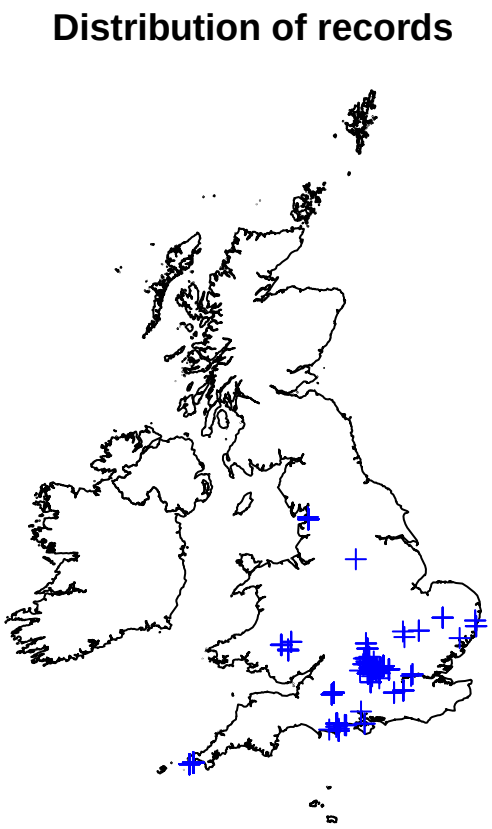


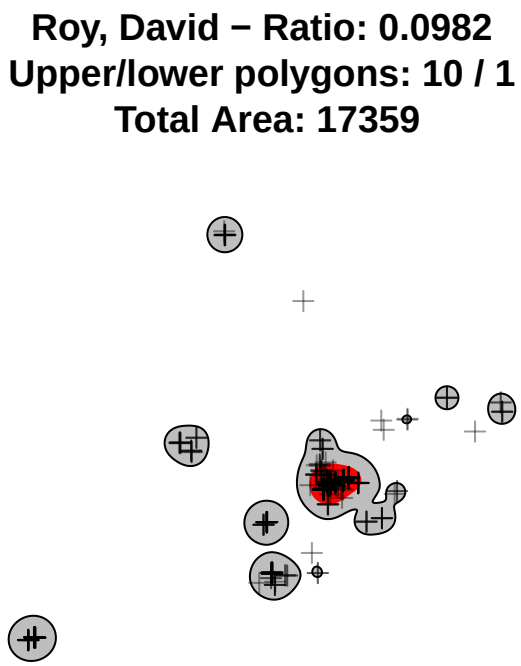
Distribution of records



Roy, David − Ratio: 0.1089
Upper/lower polygons: 12 / 3
Total Area: 6683








```
# for(recorder in c('Partridge, Francesca', 'Harley, Ross')){
#
#   RD <- spatial_behaviour(data = iRB, recorder_name = recorder,
#                           latitude_col = 'lat', longitude_col = 'long',
#                           h = 5000)
#   plot_ratio(data = RD)
#
# }

# Apply to all recorders
sum_spatial <- function(x){
  recorder_info <- spatial_behaviour(data = iRB, recorder_name = x,
                                     latitude_col = 'lat', longitude_col = 'long')
  return(data.frame(recorder = recorder_info$recorder,
                    upper_area = recorder_info$upper_area,
                    lower_area = recorder_info$lower_area,
                    upper_n_poly = recorder_info$upper_n_poly,
                    lower_n_poly = recorder_info$lower_n_poly,
                    ratio = recorder_info$ratio,
                    n = recorder_info$n))
}

clusterExport(cl, varlist = c('spatial_behaviour', 'iRB'))
all_spatial_list <- parLapply(cl, unique(iRB$recorders),
                          fun = sum_spatial)
```

```r
# combine results
all_spatial <- do.call(rbind, all_spatial_list)
temp <- all_spatial[all_spatial$n > 400, ]

# Lets have a look at some people who have recorded a lot
temp[order(temp$ratio, decreasing = TRUE),]
```

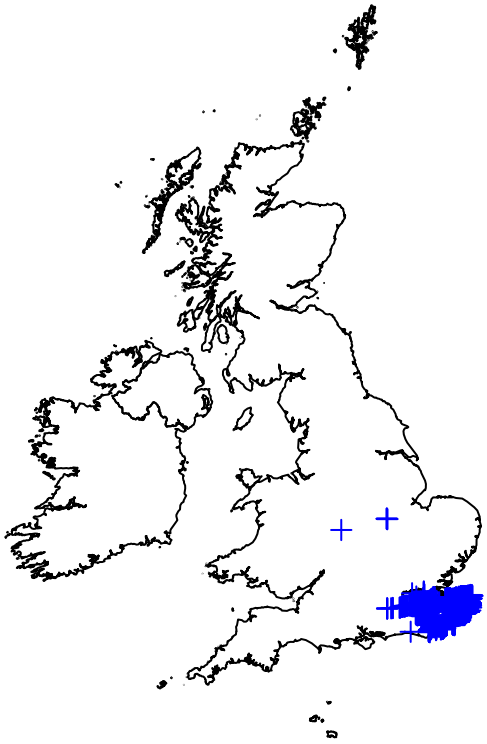
```
##                      recorder upper_area lower_area upper_n_poly
## 9541     Partridge, Francesca       5905       1905            2
## 9542         Cornish, Stephen        493        146            1
## 951                Limb, Ken       6531       1881            7
## 95123          Hunter, Amands       1653        464            1
## 9575              Jones, Dave        564        156            1
## 9555              Leaver, Kim       2169        561            3
## 95129          Saville, Simon       4290       1058            6
## 95187             Atkin, Paul       2480        592            3
## 9557            Lunnon, Marie        822        193            1
## 9583              Gillie, Tony       2635        613            3
## 958                 Cox, Steve       7884       1633           13
## 9533           Shanks, Scott         5888       1219           10
## 9562              Sell, Claire       1646        338            1
## 9516              Kilbey, Dave       6253       1276            9
## 95205             Ford, Rachel        843        163            2
## 9556           Steele, Andrew      16070       3002           28
## 9582         Checkley, Graham       1746        323            2
## 9554              Hill, Brian       4648        856           10
## 9540            Cowton, Keith       4424        790            7
## 9512              Bowles, Nick       1941        333            4
## 957                 fenn, paul       3192        539            6
## 9578              Sims, Clive       3392        514            6
## 956            Newbould, John       5372        811            8
## 955            Warren, Martin      11701       1747           20
## 9568              Stewart, Tam       9098       1286           16
## 95264           Dawson, Steve       1596        213            1
## 9564              Fox, Richard       7776       1012           14
## 95130 Lonsdale, Liz and Steve      11053       1429           25
## 9511             Austin, David       3607        434            5
## 95190               Roy, David       6683        728           12
## 9523           Shersby, Megan       4605        496            7
## 95169             Harley, Ross       2544        256            7
## 95234      Pennington, Robert       3798        335            8
## 9538              Allan, David       4360        242           13
## 95145          shilland, ewan       9871        242           26
##       lower_n_poly      ratio    n
## 9541             5 0.32260796 1438
## 9542             1 0.29614604  487
## 951              7 0.28801102  622
## 95123            1 0.28070175 1109
## 9575             1 0.27659574 2240
## 9555             1 0.25864454  542
## 95129            4 0.24662005  441
## 95187            2 0.23870968  615
## 9557             1 0.23479319  450
## 9583             2 0.23263757 1123
```

```
## 958              5 0.20712836 1004
## 9533             5 0.20703125  513
## 9562             1 0.20534629  555
## 9516             2 0.20406205  764
## 95205            1 0.19335706  433
## 9556            14 0.18680772  571
## 9582             1 0.18499427 1810
## 9554             2 0.18416523  858
## 9540             3 0.17857143  407
## 9512             1 0.17156105  609
## 957              2 0.16885965 2525
## 9578             2 0.15153302  873
## 956              2 0.15096798 1012
## 955              1 0.14930348 2483
## 9568             3 0.14134975 1799
## 95264            1 0.13345865  838
## 9564             4 0.13014403 1158
## 95130            6 0.12928617  565
## 9511             2 0.12032160  444
## 95190            3 0.10893311  590
## 9523             2 0.10770901  478
## 95169            1 0.10062893  656
## 95234            1 0.08820432  985
## 9538             1 0.05550459 3307
## 95145            1 0.02451626 1641
```

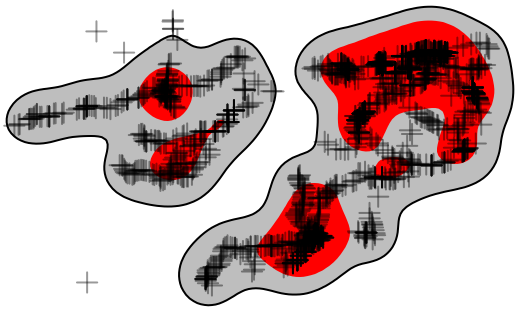
Lets have a look at two people with very different ratios

```
# Get the names of top and bottom
temp <- temp[order(temp$ratio, decreasing = TRUE),]
top <- as.character(head(temp$recorder, 1))
bottom <- as.character(tail(temp$recorder, 1))

# Plot the top and bottom ratio recorder
for(i in c(top, bottom)){
  top_d <- spatial_behaviour(data = iRB,
                             recorder_name = i,
                             latitude_col = 'lat',
                             longitude_col = 'long')
  plot_ratio(data = top_d)
}
```

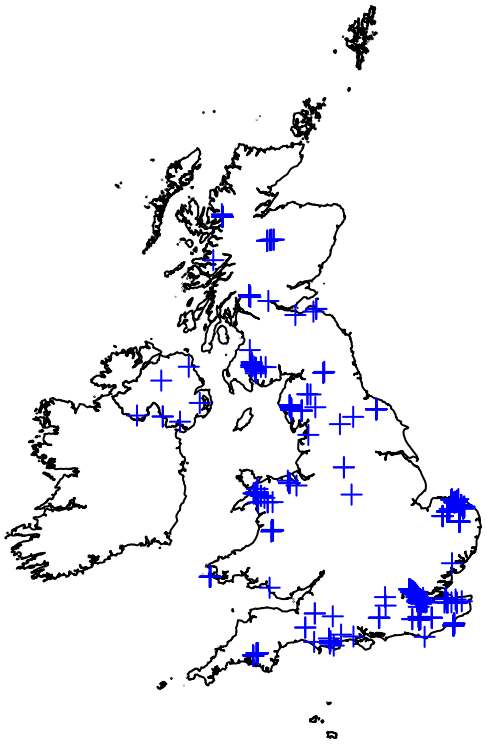
Distribution of records



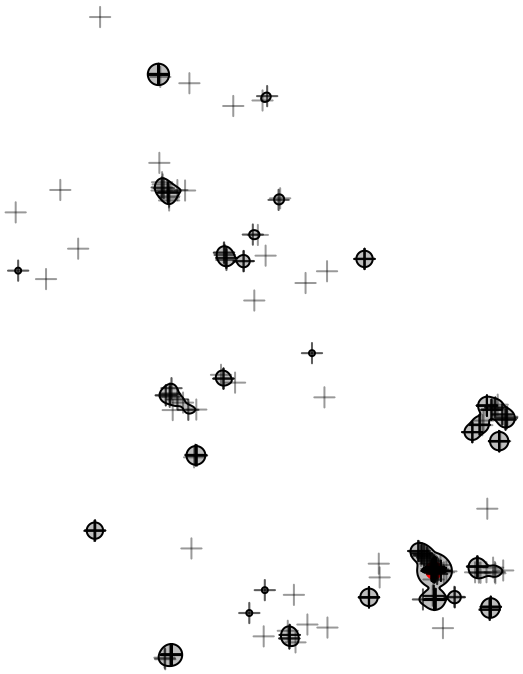
Partridge, Francesca − Ratio: 0.3226
Upper/lower polygons: 2 / 5
Total Area: 5905



Distribution of records



shilland, ewan − Ratio: 0.0245
Upper/lower polygons: 26 / 1
Total Area: 9871



Taxanomic Metrics

These metric relate the the species that people record

Taxanomic Breadth

This is simply a measure of the proportion of taxa a person has recorded. Note this is going to be correlated to the number of records.

```
taxa_breadth <- function(data, recorder_name,
                         sp_col = 'preferred_taxon',
                         recorder_col = 'recorders'){

  data_rec <- data[data[,recorder_col] == recorder_name, c(sp_col, recorder_col)]

  return(data.frame(recorder = recorder_name,
                    taxa_breadth = length(unique(data_rec[ ,sp_col])),
                    taxa_prop = length(unique(data_rec[ ,sp_col]))/length(unique(data[ ,sp_col])),
                    n = nrow(data_rec)))
}

taxa_breadth <- do.call(rbind, parLapply(cl, unique(iRB$recorders),
                                         fun = taxa_breadth,
                                         data = iRB))

temp <- taxa_breadth[taxa_breadth$n > 400, ]

# Lets have a look at some people who have recorded a lot
temp[order(temp$taxa_prop, decreasing = TRUE),]
```

```
##                    recorder taxa_breadth taxa_prop    n
## 6             Warren, Martin           52 0.7536232 2483
## 39              Allan, David           51 0.7391304 3307
## 247       Pennington, Robert           49 0.7101449  985
## 56                Hill, Brian           48 0.6956522  858
## 138           Saville, Simon           48 0.6956522  441
## 9                 Cox, Steve           47 0.6811594 1004
## 82               Sims, Clive           47 0.6811594  873
## 68              Fox, Richard           46 0.6666667 1158
## 58            Steele, Andrew           45 0.6521739  571
## 41             Cowton, Keith           43 0.6231884  407
## 179             Harley, Ross           42 0.6086957  656
## 197              Atkin, Paul           42 0.6086957  615
## 2                  Limb, Ken           41 0.5942029  622
## 8                 fenn, paul           41 0.5942029 2525
## 17              Kilbey, Dave           41 0.5942029  764
## 279            Dawson, Steve           40 0.5797101  838
## 88              Gillie, Tony           39 0.5652174 1123
## 24            Shersby, Megan           38 0.5507246  478
## 139 Lonsdale, Liz and Steve            37 0.5362319  565
## 200               Roy, David           37 0.5362319  590
## 13              Bowles, Nick           36 0.5217391  609
## 154           shilland, ewan           36 0.5217391 1641
## 7             Newbould, John           33 0.4782609 1012
```

```
## 42     Partridge, Francesca           32 0.4637681 1438
## 65             Sell, Claire           32 0.4637681  555
## 131          Hunter, Amands           31 0.4492754 1109
## 72             Stewart, Tam           29 0.4202899 1799
## 59             Lunnon, Marie          28 0.4057971  450
## 34            Shanks, Scott           26 0.3768116  513
## 57               Leaver, Kim           24 0.3478261  542
## 79               Jones, Dave           23 0.3333333 2240
## 12             Austin, David           22 0.3188406  444
## 87          Checkley, Graham           22 0.3188406 1810
## 44          Cornish, Stephen           19 0.2753623  487
## 217             Ford, Rachel           15 0.2173913  433
```

Species Rarity

We want to capture the rarity of the species that people record. For example are they just recording the common species or are they only recording the rare ones, or perhaps they are recording everything. Since we don't know the real frequency distribution we can only compare people to the global average in the dataset. We can look to see what the distribution of species rank for each recorder is and how this compares to all records. A recorder only interested in rare species will have a median rank higher than the average. A recorder only recording common species will have a value lower than the average. The number of taxa is going to vary between groups so to make these values comparable between groups we need to scale the ranks, here I scale to 100.

```
# Lets look at a recorder
species_rank <- function(data, recorder_name,
                         sp_col = 'preferred_taxon',
                         recorder_col = 'recorders'){

  data <- data[,c(sp_col, recorder_col)]
  rank_species <- rank(abs(table(data[,sp_col])-max(table(data[,sp_col]))))

  # scale ranks to 100
  rank_species <- (rank_species/max(rank_species)) * 100

  sp_counts <- table(data[,sp_col])

  rank_reps <- rep(rank_species, sp_counts)
  grand_median <- median(rank_reps)
  grand_sd <- sd(rank_reps)

  recorder_data <- data[data[,recorder_col] == recorder_name,]
  recorder_data$rank <- rank_species[recorder_data[ ,sp_col]]

  return(data.frame(recorder = as.character(recorder_name),
                    median_rarity = median(recorder_data$rank),
                    median_diff_rarity = median(recorder_data$rank) - grand_median,
                    stdev = sd(recorder_data$rank),
                    n = nrow(recorder_data)))
}

rarity_preference <- do.call(rbind,
                             parLapply(cl, unique(iRB$recorders),
                                       fun = species_rank,
```

```
                                        data = iRB))

temp <- cbind(rarity_preference[rarity_preference$n > 400, 'recorder'],
              round(rarity_preference[rarity_preference$n > 400, c('median_rarity', 'median_diff_rarity
colnames(temp)[1] <- 'recorder'

# Lets have a look at some people who have recorded a lot
temp[order(temp$median_diff, decreasing = TRUE),]
```

```
##                     recorder median_rarity median_diff_rarity stdev    n
## 138          Saville, Simon         18.84               7.25 17.56  441
## 41            Cowton, Keith         17.39               5.80 15.28  407
## 6            Warren, Martin         15.94               4.35 15.55 2483
## 82              Sims, Clive         15.94               4.35 14.66  873
## 13            Bowles, Nick          14.49               2.90 12.41  609
## 24           Shersby, Megan         14.49               2.90 12.53  478
## 197             Atkin, Paul         14.49               2.90 13.98  615
## 2                 Limb, Ken         13.04               1.45 13.30  622
## 7            Newbould, John         13.04               1.45 11.86 1012
## 8                fenn, paul         13.04               1.45 12.73 2525
## 34           Shanks, Scott          13.04               1.45 14.10  513
## 56              Hill, Brian         13.04               1.45 14.69  858
## 59            Lunnon, Marie         13.04               1.45 10.06  450
## 65             Sell, Claire         13.04               1.45 12.67  555
## 72             Stewart, Tam         13.04               1.45 15.78 1799
## 88             Gillie, Tony         13.04               1.45 12.46 1123
## 131          Hunter, Amands         13.04               1.45 10.41 1109
## 179            Harley, Ross         13.04               1.45 13.14  656
## 200              Roy, David         13.04               1.45 13.74  590
## 247      Pennington, Robert         13.04               1.45 13.13  985
## 17             Kilbey, Dave         11.59               0.00 13.40  764
## 57              Leaver, Kim         11.59               0.00  8.91  542
## 58           Steele, Andrew         11.59               0.00 13.12  571
## 68             Fox, Richard         11.59               0.00 14.05 1158
## 87         Checkley, Graham         11.59               0.00 10.18 1810
## 139 Lonsdale, Liz and Steve         11.59               0.00 12.42  565
## 154          shilland, ewan         11.59               0.00 11.99 1641
## 9                Cox, Steve         10.14              -1.45 13.05 1004
## 217            Ford, Rachel         10.14              -1.45  7.85  433
## 279           Dawson, Steve         10.14              -1.45 11.30  838
## 39              Allan, David          8.70              -2.90 12.43 3307
## 42     Partridge, Francesca          8.70              -2.90  9.89 1438
## 79               Jones, Dave         8.70              -2.90  7.13 2240
## 12             Austin, David         7.25              -4.35  7.93  444
## 44         Cornish, Stephen          7.25              -4.35  7.25  487
```

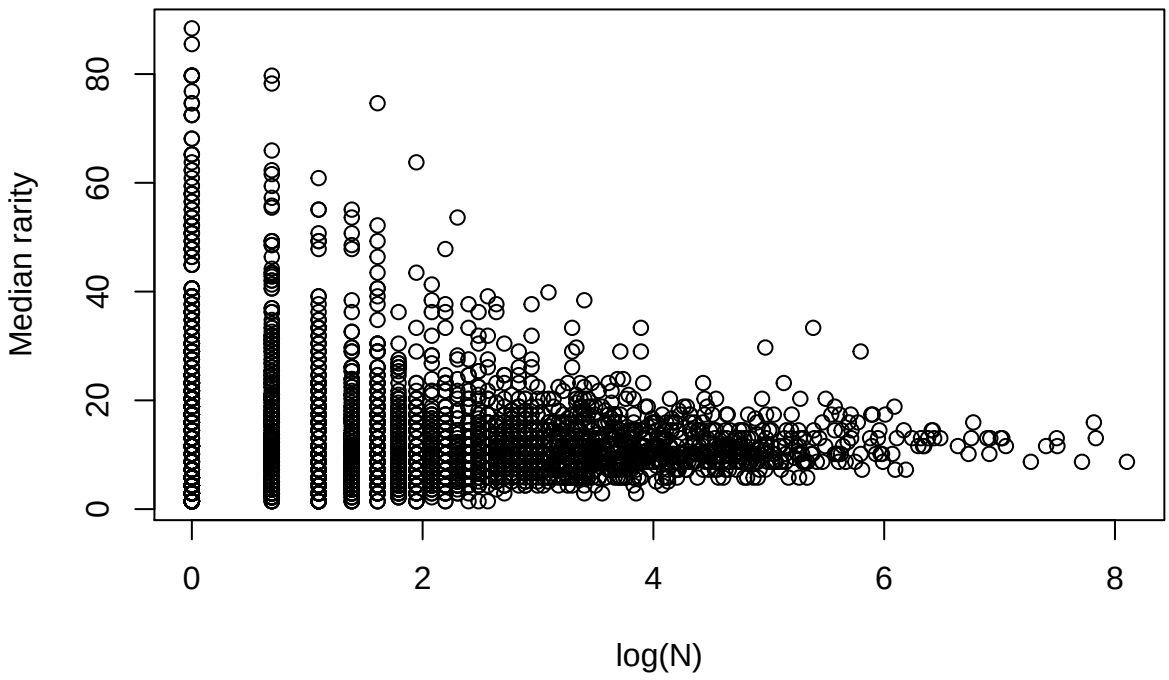
Here `median_diff` gives the difference between the grand median for all records and the recorders median. This suggests `Saville, Simon` prefers to record rare species and `Cornish, Stephen` prefers to record common species.

This could be correlated to the number of records.

```
mod <- glm(median_rarity ~ log(n), data = rarity_preference, family = 'quasipoisson')
summary(mod)
```

```
##
## Call:
## glm(formula = median_rarity ~ log(n), family = "quasipoisson",
##     data = rarity_preference)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.4677  -2.0144  -0.4888   0.9107  12.9938
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.694399   0.018357 146.780  < 2e-16 ***
## log(n)      -0.068026   0.008371  -8.127 5.83e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 7.470689)
##
##     Null deviance: 25197  on 3970  degrees of freedom
## Residual deviance: 24690  on 3969  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

```r
plot(log(rarity_preference$n),
     rarity_preference$median_rarity,
     xlab = 'log(N)',
     ylab = 'Median rarity')
```



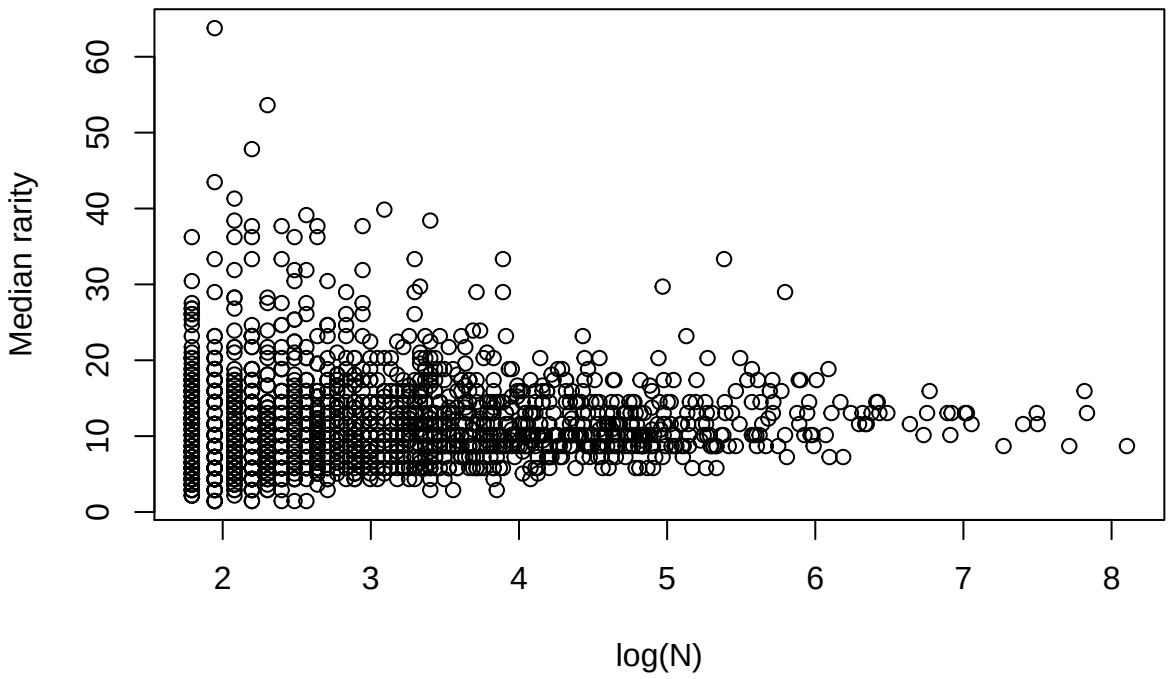
There is a significant negative relationship. The more records you make the lower your median value. This could be a result of the fact that people who make only a few records record rare stuff?

```
rarity_preference_above <- rarity_preference[rarity_preference$n > 5, ]
mod <- glm(median_rarity ~ log(n), data = rarity_preference_above, family = 'quasipoisson')
summary(mod)
```

```
##
## Call:
## glm(formula = median_rarity ~ log(n), family = "quasipoisson",
##     data = rarity_preference_above)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.8102  -1.1558  -0.2551   0.7583  10.5966
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.442240   0.034492  70.805   <2e-16 ***
## log(n)      0.008094   0.010545   0.768    0.443
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 3.046009)
##
##     Null deviance: 5079.1  on 1888  degrees of freedom
## Residual deviance: 5077.3  on 1887  degrees of freedom
```

```
## AIC: NA
##
## Number of Fisher Scoring iterations: 4
```

```
plot(log(rarity_preference_above$n),
     rarity_preference_above$median_rarity,
     xlab = 'log(N)',
     ylab = 'Median rarity')
```



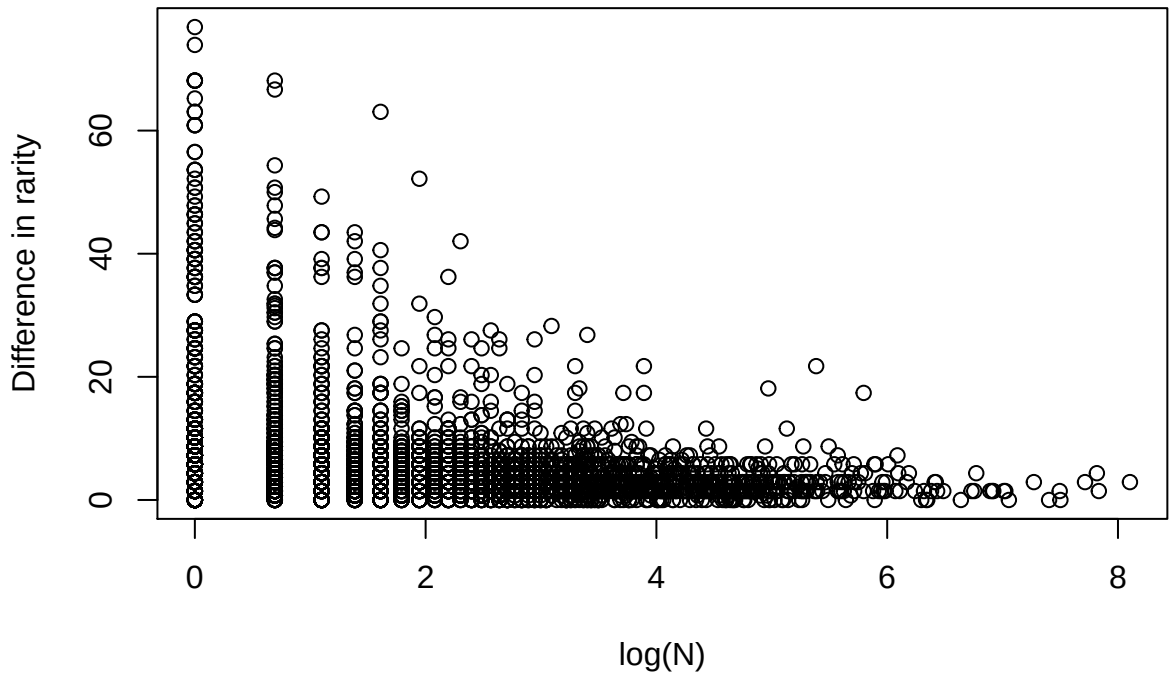
Okay, the relationship falls down once we get rid of the people who only record a few species. I suggest this metric not be estimates for people who contribute only a few records. The relationship might actually be between deviation from the median and `n`.

```
rarity_preference$median_diff_abs <- abs(rarity_preference$median_diff_rarity)
mod <- glm(median_diff_abs ~ log(n), data = rarity_preference, family = 'quasipoisson')
summary(mod)
```

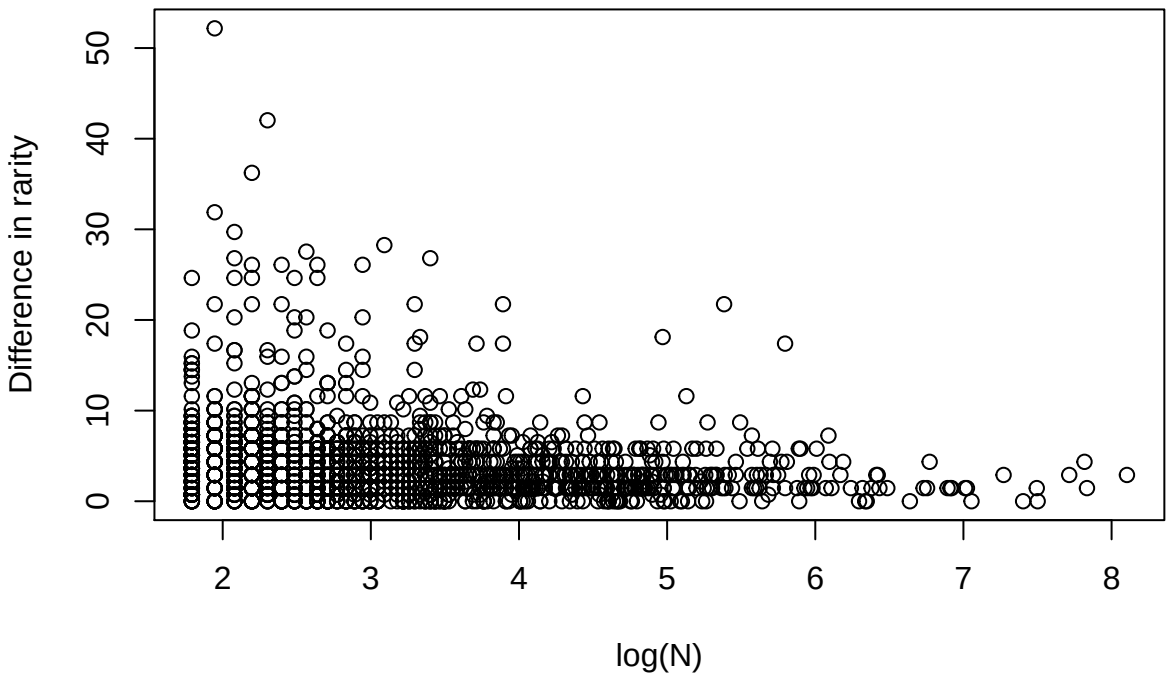
```
##
## Call:
## glm(formula = median_diff_abs ~ log(n), family = "quasipoisson",
##     data = rarity_preference)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -4.5660  -1.6186  -0.5512   0.4506  13.2661
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept)  2.34411    0.02392   98.01   <2e-16 ***
## log(n)      -0.30775    0.01367  -22.50   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 7.433661)
##
##     Null deviance: 26334  on 3970  degrees of freedom
## Residual deviance: 22046  on 3969  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

```
plot(log(rarity_preference$n),
     rarity_preference$median_diff_abs,
     xlab = 'log(N)',
     ylab = 'Difference in rarity')
```




```
plot(log(rarity_preference$n[rarity_preference$n >5]),
     rarity_preference$median_diff_abs[rarity_preference$n >5],
     xlab = 'log(N)',
     ylab = 'Difference in rarity')
```



The more records you record the less you deviate from the median. This is probably because you only get extreme values where the sample size is small.

Species accumulation curve

The rate at which a recorder accumulates species over time recorded could be seen as a measure of their effort and expertise. The time scale will be measured in days recorded (not days elapsed). This measure will need to be normalised within a taxonomic group as clearly one can accumulate species more rapidly for birds than amphibians and reptiles. Note that while we look only at active days we have no way of accounting for actual time (hrs) recorded as a 10 minute search on one days appears the same as a 3 hour search on another. This metric is therefore an approximation only.

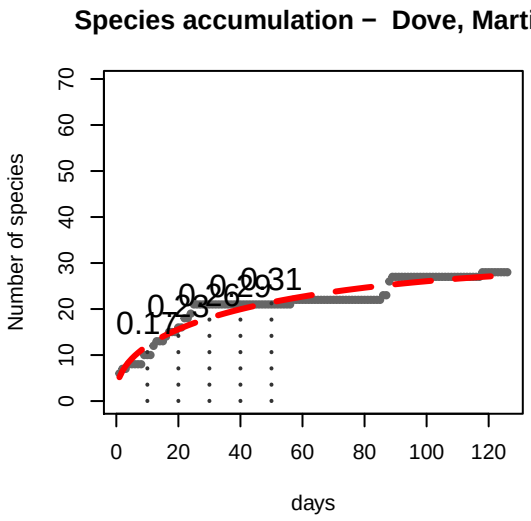
```
# This can be run with all data
species_accumulation <- function(data, recorder_name,
                                 n_taxa, prediction_days = c(10, 50),
                                 sp_col = 'preferred_taxon',
                                 date_col = 'date_start',
                                 recorder_col = 'recorders',
                                 plot = FALSE, ...){

  # Get the data for this recorder
  rec_data <- data[data[,recorder_col] == recorder_name,
                   c(date_col, sp_col)]

  # sort dates
  dates <- sort(unique(rec_data$date_start))
```

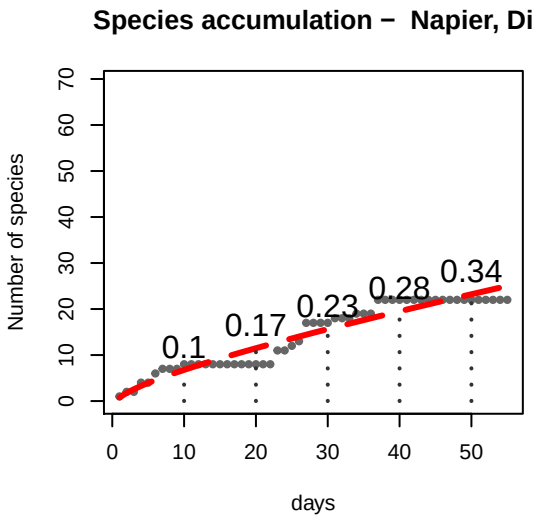
```
# Only carry out this test on people with more than the
# required number of days in their data
if(length(dates) >= max(prediction_days)){

  # accumulation function
  acc <- function(x){
    length(unique(rec_data[,sp_col][rec_data[,date_col] <= x]))
  }

  species_accumulation_data <- sapply(dates, FUN = acc)
  day <- seq_along(species_accumulation_data)

  # Fit a model
  m <- glm(formula = species_accumulation_data ~ day + sqrt(day))
  m_sum <- summary(m)
  # predict atleast up to day 100 (could be dangerous)
  # but just for visualiation purposes
  days_to_predict <- ifelse(test = length(dates) > max(prediction_days),
                            yes = length(dates),
                            no = prediction_days)
  predicted <- predict(m, newdata = data.frame(day = 1:days_to_predict))

  if(plot){
    plot(species_accumulation_data,
         main = paste('Species accumulation - ', recorder_name),
         xlab = 'days',
         ylab = 'Number of species',
         col = 'grey40',
         pch = 20,
         ... = ...)
    lines(predicted, col = 'red', lwd = 3, lty = 5)
    for(pred_day in prediction_days){
      lines(x = rep(pred_day, 2), y = c(0, predicted[pred_day]),
      lty = 3, col = 'grey20', lwd = 2)
      text(x = pred_day, y = predicted[pred_day] + 5, cex = 1.5,
         labels = round(predicted[pred_day]/n_taxa, 2))
    }
  }

  # create named vector for predictions
  x <- predicted[prediction_days]/n_taxa
  names(x) <- paste0('d', prediction_days)
  x <- data.frame(t(x))
  x$recorder <- as.character(recorder_name)

  return(list(species_accumulation_data = species_accumulation_data,
              predicted_data = predicted,
              day_pred = x,
              n_day = length(dates)))

} else {

  # empty df
```

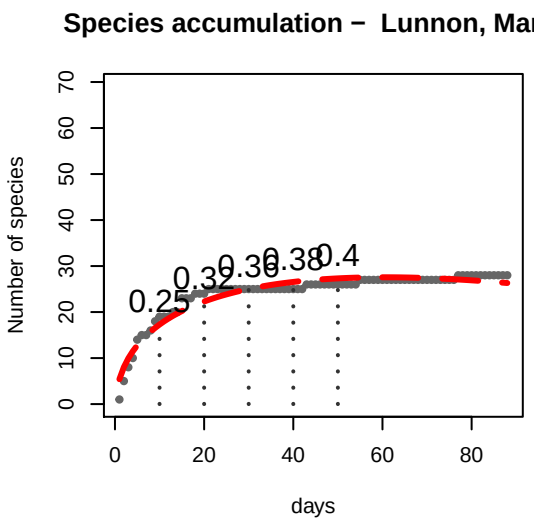
```
    eDF <- as.data.frame(cbind(matrix(data = rep(NA,
                                                 length(prediction_days)),
                                      nrow = 1,
                                      dimnames = list(recorder_name, paste0('d', prediction_days)))))

    eDF$recorder <- as.character(recorder_name)

    return(list(species_accumulation_data = NA,
                predicted_data = NA,
                day_pred = eDF,
                n_day = length(dates)))

  }
}
```

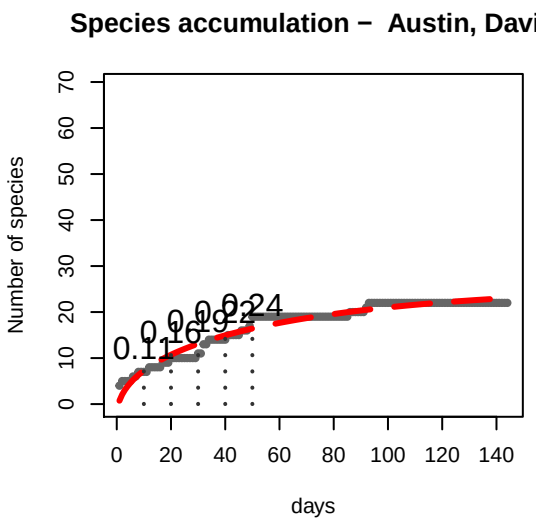
When we plot some of the species accumulation curves you can see variation between recorders. Note that the height of the plateaux varies as does the time taken to reach that point. Is this biased by location? Presumably a recorder in Scotland will always have a lower plateau than one in the South of England? Also in a less species rich area the rate of accumulation will be less? Another way to assess this would be to model d20 by recorder and mean latitude of that recorders observations. The coefficient of recorder would then be adjusted for latitude.

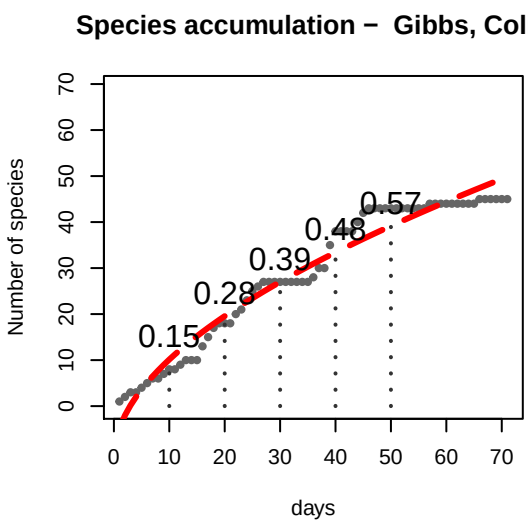
```
# Run for a few people
par(mfrow = c(3, 2))

# top recorders
user_dates <- tapply(iRB$date_start, iRB$recorders, FUN = function(x) length(unique(x)))
recorders <- sample(names(sort(user_dates[user_dates > 50], decreasing = TRUE)), size = 6)

for(recorder in recorders){
user_acc <- species_accumulation(data = iRB, n_taxa = length(unique(iRB$preferred_taxon)),
                                 recorder_name = recorder,
                                 plot = TRUE,
                                 prediction_days = seq(10, 50, 10),
                                 # xlim = c(0, 100),
                                 ylim = c(0, length(unique(iRB$preferred_taxon))))
}
```

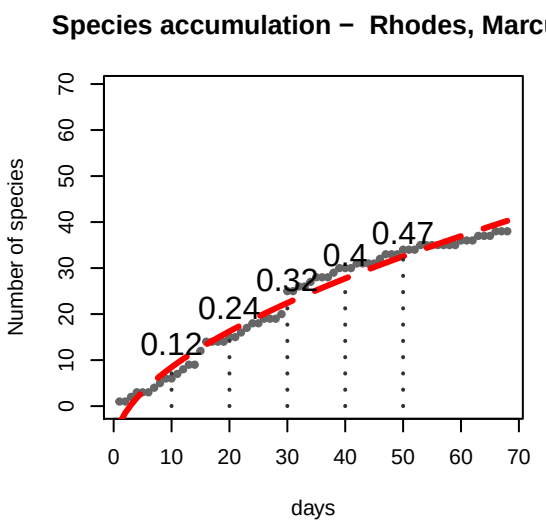

Species accumulation − Dove, Martin
Number of species
days
0.17
0.31


Species accumulation − Napier, Di
Number of species
days
0.1
0.17
0.23
0.28
0.34


Species accumulation − Lunnon, Marie
Number of species
days
0.25
0.32
0.38
0.4

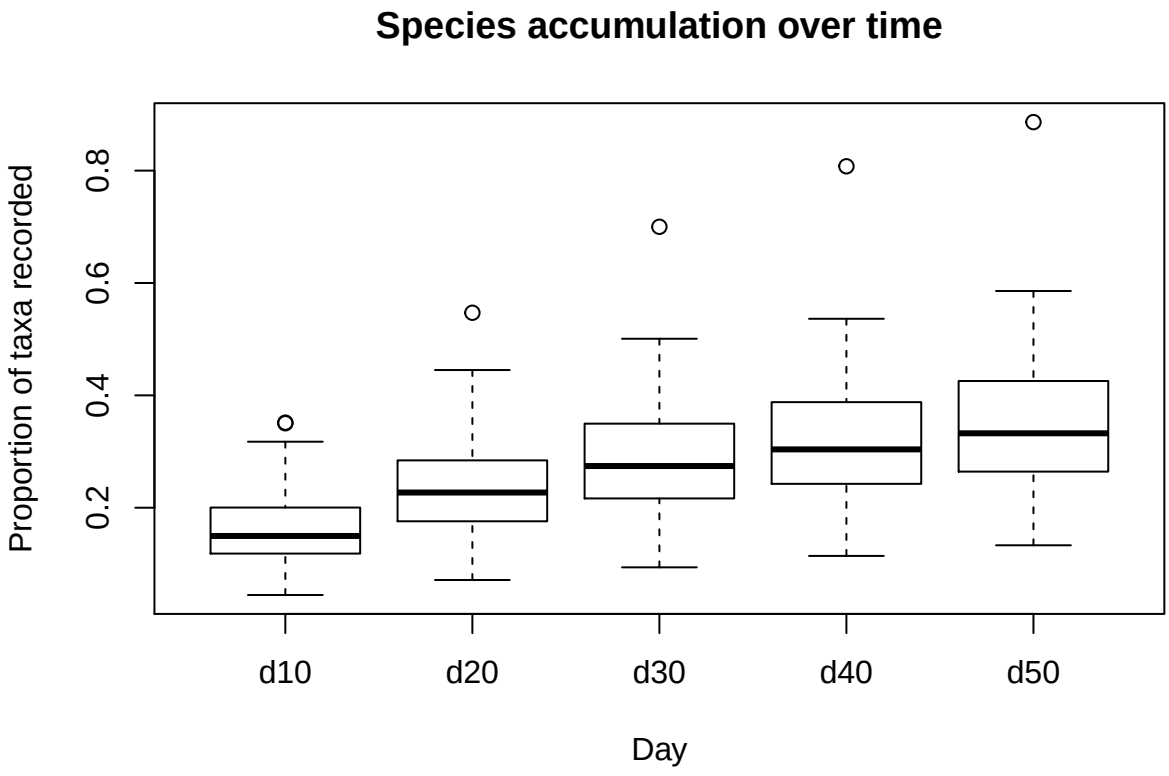

Species accumulation − Austin, David
Number of species
days
0.11
0.16
0.24


Species accumulation − Gibbs, Colin
Number of species
days
0.15
0.28
0.39
0.48
0.57

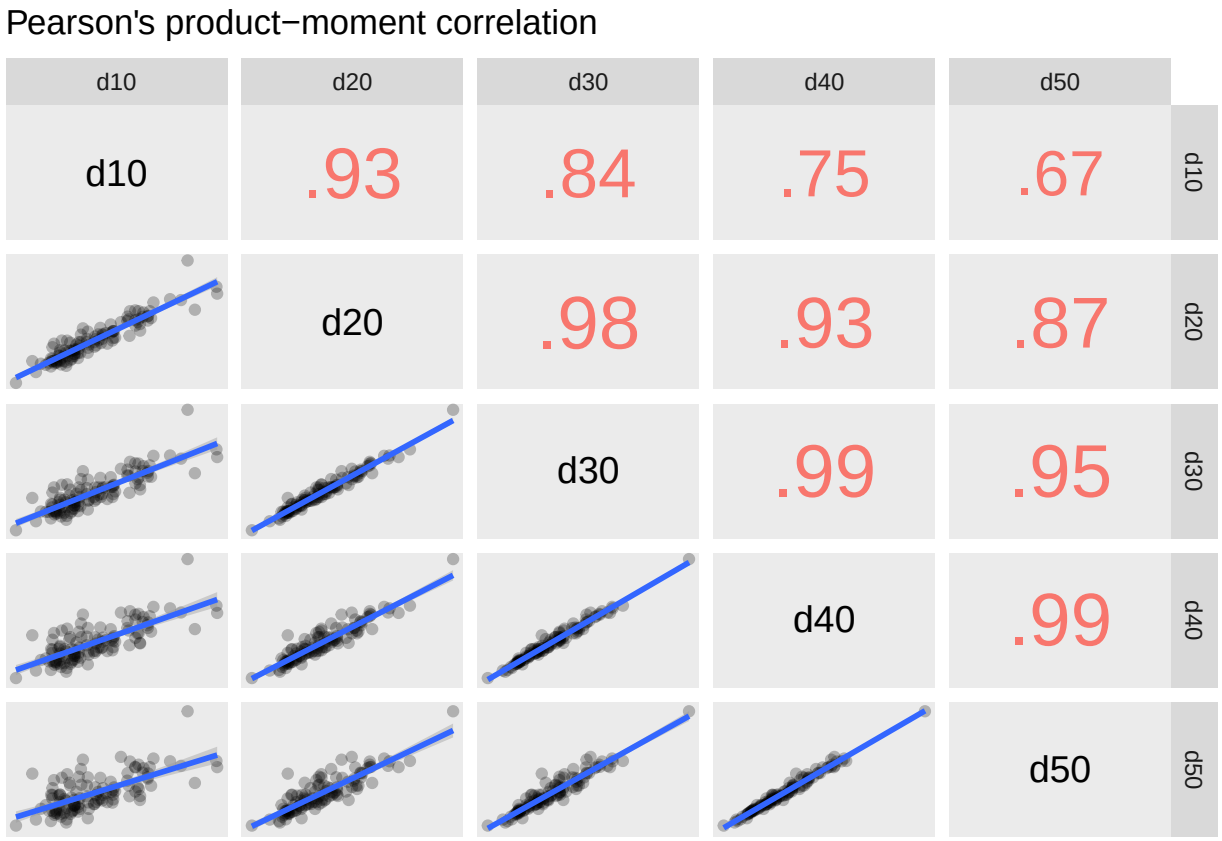

Species accumulation − Rhodes, Marcus
Number of species
days
0.12
0.24
0.32
0.4
0.47

We can explore the values in more detail when we calculate them for everyone with enought records. Note we only calculate these values for recorders with a total number of days equal or more to the highest number of days we want to predict

```
par(mfrow = c(1, 1))

clusterExport(cl, varlist = c('species_accumulation', 'iRB'))
# Calculate for everyone
all_acc_list <- parLapply(cl,
                          X = unique(iRB$recorders),
                          fun = function(x){
  day_pred <- species_accumulation(recorder_name = x,
                                   data = iRB,
                                   n_taxa = length(unique(iRB$preferred_taxon)),
                                   plot = FALSE,
                                   prediction_days = seq(10, 50, 10))$day_pred
})

all_acc <- do.call(rbind, all_acc_list)

# These values clearly rise over time
boxplot(na.omit(all_acc[,!colnames(all_acc) %in% 'recorder']),
        xlab = 'Day',
        ylab = 'Proportion of taxa recorded',
        main = 'Species accumulation over time')
```



```
# How do these correlate?
cor_data <- cor.matrix(na.omit(all_acc[,!colnames(all_acc) %in% 'recorder']))
ggcorplot(cor.mat = cor_data,
          data = na.omit(all_acc[,!colnames(all_acc) %in% 'recorder']),
          var_text_size = 5,
          cor_text_limits = c(5,10))
```



This correlation plot shows that - of hte cut offs I chose - 20 is the probably the best value. This is because it is the metric that is best correlated with all the other values chossen.

List-length: Species per 'visit'

This is the metric that Nick Isaac is most interested in. The interest is in knowing, or predicting, whether a person is recording complete lists or is recording ad hoc sightings. The difference is important as the former can be used to infer absence whereas the latter cannot.

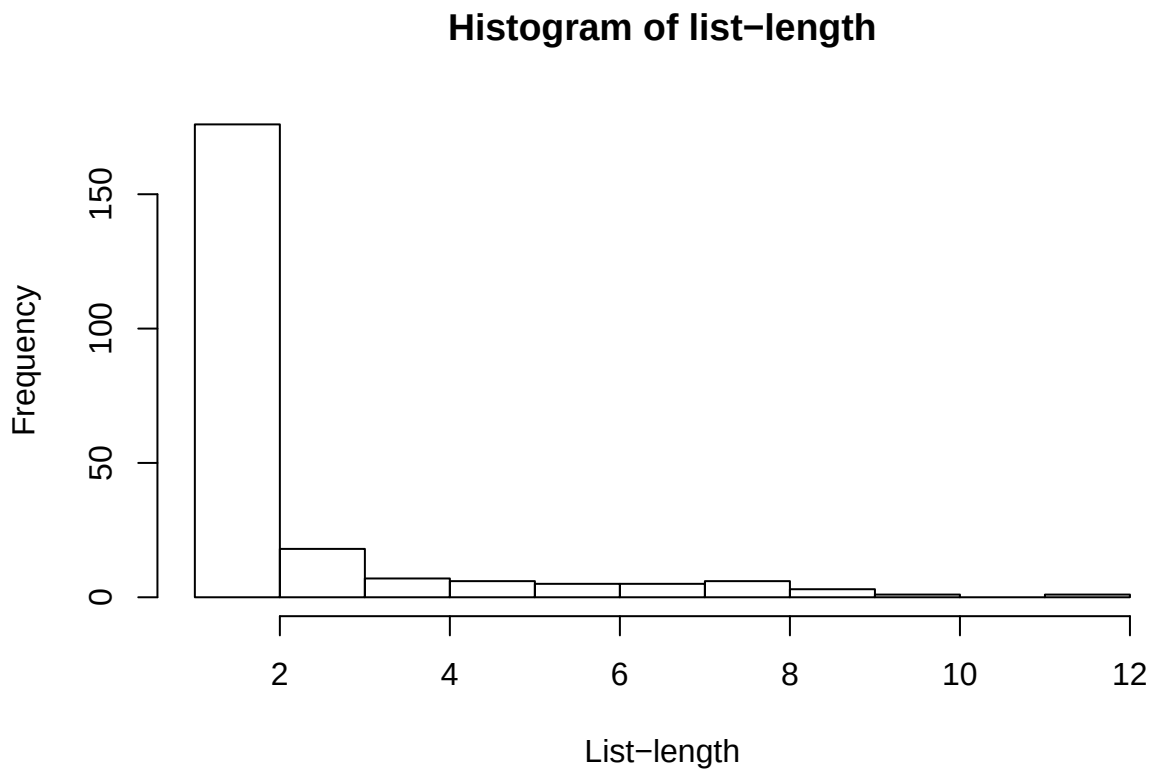
Here I create a few metrics to quantify this including mean list-length, median list-length and variance.

We define visit as a unique combination of date and location. Location is taken to be a 1km square in this example.

```
listLength <- function(data,
                       recorder_name,
                       threshold = 10,
                       plot = FALSE,
                       sp_col = 'preferred_taxon',
                       date_col = 'date_start',
```

```
                      recorder_col = 'recorders',
                      location_col = 'kmsq'){

  # get the data for this recorder
  rec_data <- data[data[ ,recorder_col] == recorder_name, ]

  # create a unique visit column
  rec_data$visit <- paste(rec_data[ , date_col], rec_data[ , location_col],
                      sep = '_')

  # calculate list-lengths
  rec_LL <- tapply(rec_data$preferred_taxon, rec_data$visit,
                   FUN = function(x) length(unique(x)))

  # Only calculate if the number of lists meets the threshold
  if(length(rec_LL) >= threshold){
    if(plot) hist(rec_LL,
                  breaks = max(rec_LL),
                  main = 'Histogram of list-length',
                  xlab = 'List-length')

    mean_LL <- mean(rec_LL)
    median_LL <- median(rec_LL)
    variance <- var(rec_LL)

    # proportion of 'visits' that are of length 1
    p1 <- sum(rec_LL == 1)/length(rec_LL)

    # proportion of 'visits' that are of 4 or more
    p4 <- sum(rec_LL >= 4)/length(rec_LL)


    df <- data.frame(recorder = recorder_name,
                     mean_LL = mean_LL,
                     median_LL = median_LL,
                     variance = variance,
                     p1 = p1,
                     p4 = p4,
                     n_lists = length(rec_LL))
    return(df)
  } else {

    df <- data.frame(recorder = recorder_name,
                     mean_LL = NA,
                     median_LL = NA,
                     variance = NA,
                     p1 = NA,
                     p4 = NA,
                     n_lists = length(rec_LL))
    return(df)
  }
}
```

```
# Add location column
raw_kmsq <- gps_latlon2gr(latitude = iRB$lat,
                          longitude = iRB$long,
                          return_type = 'gr')

iRB$kmsq <- reformat_gr(raw_kmsq, prec_out = 1000)

# Run for a recorder
listLength(data = iRB, recorder_name = 'Roy, David', plot = TRUE)
```



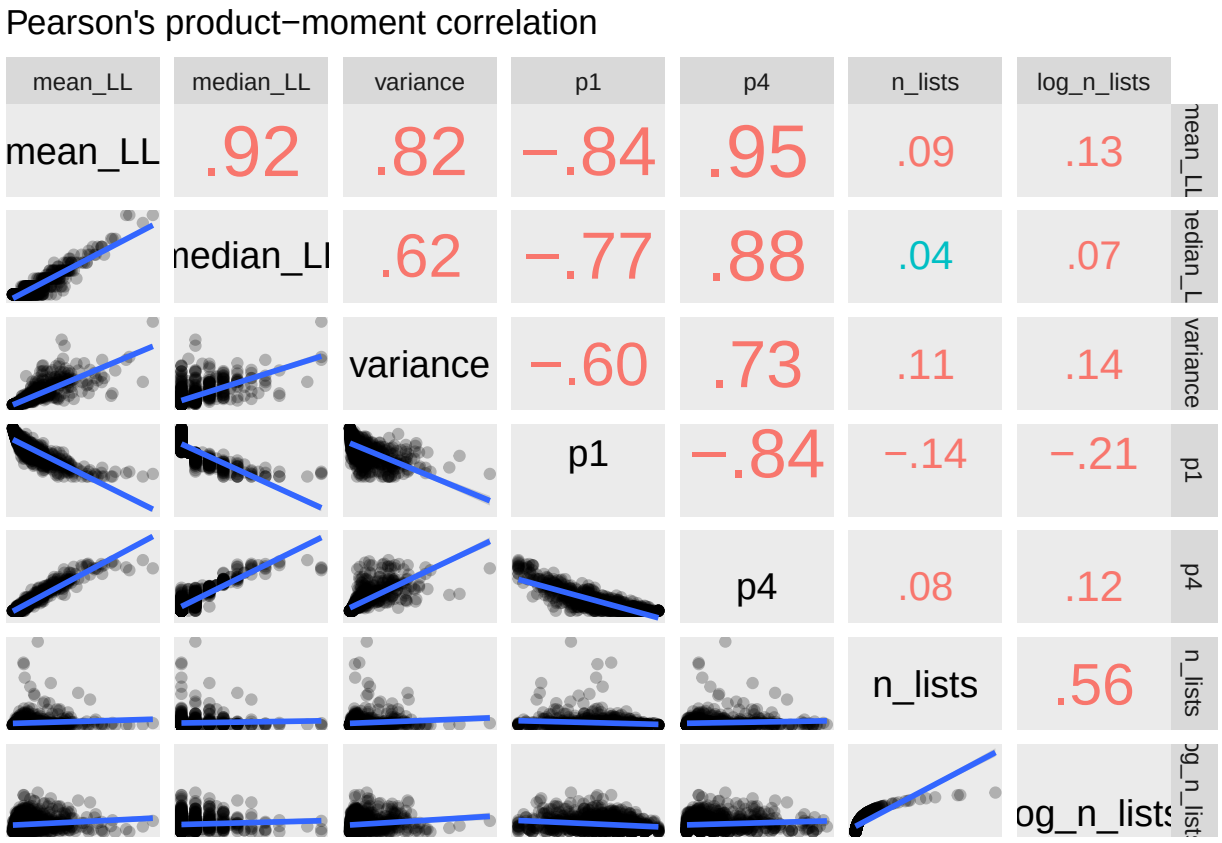

```
##     recorder  mean_LL median_LL variance        p1        p4 n_lists
## 1 Roy, David 2.149123         1 4.241982 0.5964912 0.1491228     228
```

```
# Run for all
all_LL <- parLapply(cl, unique(iRB$recorders), fun = listLength,
                    data = iRB)

LL_tab <- do.call(rbind, all_LL)
```

I wonder if there are correlations between the metrics and the number of lists?

```
# How do these correlate?
LL_tab$log_n_lists <- log(LL_tab$n_lists)
cor_data <- cor.matrix(variables = LL_tab[,!colnames(LL_tab) %in% 'recorder'])
ggcorplot(cor.mat = cor_data,
          data = na.omit(LL_tab),
          var_text_size = 5,
```

```
          cor_text_limits = c(5,10))
```




```
# Doesn't look like n_lists is strongly correlated
# ... but it is
m <- glm(n_lists ~ p1 + variance + median_LL + mean_LL, data = LL_tab, family = 'poisson')
summary(m)
```

```
##
## Call:
## glm(formula = n_lists ~ p1 + variance + median_LL + mean_LL,
##     family = "poisson", data = LL_tab)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -13.839   -4.580   -2.583    0.226   64.736
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  5.262513   0.049617 106.063   <2e-16 ***
## p1          -2.098008   0.048752 -43.034   <2e-16 ***
## variance     0.030008   0.002631  11.407   <2e-16 ***
## median_LL   -0.233371   0.013278 -17.576   <2e-16 ***
## mean_LL     -0.048047   0.021101  -2.277   0.0228 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 43901  on 849  degrees of freedom
## Residual deviance: 40137  on 845  degrees of freedom
##   (3121 observations deleted due to missingness)
## AIC: 44448
##
## Number of Fisher Scoring iterations: 6
```

It seems that the more lists you record the shorter they tend to be, this could be causally linked. People who record single species lists will accumulate lists more rapidly as it is less effort per list.

Combine metrics

Now we have allthe metrics we need we can combine them together into one table to be analysed

```
# Combine the metric data.frames
all_metrics <- merge(all_AR, all_WDD, by = 'recorder', all = T)
all_metrics <- merge(all_metrics, all_P, by = 'recorder', all = T)
all_metrics <- merge(all_metrics, all_spatial, by = 'recorder', all = T)
all_metrics <- merge(all_metrics, taxa_breadth, by = 'recorder', all = T)
all_metrics <- merge(all_metrics, rarity_preference, by = 'recorder', all = T)
all_metrics <- merge(all_metrics, all_acc, by = 'recorder', all = T)
all_metrics <- merge(all_metrics, LL_tab, by = 'recorder', all = T)

# Save this to file
save(all_metrics, file = 'output/all_metrics1.rdata')

# Let's see a summary of all this data
summary(all_metrics)
```

```
##             recorder    activity_ratio    total_duration
##  Marsh, Nick    :   1   Min.   :0.00261   Min.   :  1.0
##  Limb, Ken      :   1   1st Qu.:0.04167   1st Qu.:  1.0
##  Ward, John     :   1   Median :0.16667   Median : 26.0
##  Hughes, Peter  :   1   Mean   :0.41624   Mean   :167.6
##  Turner, Lindsey:   1   3rd Qu.:1.00000   3rd Qu.:323.2
##  Warren, Martin :   1   Max.   :1.00000   Max.   :885.0
##  (Other)        :3983   NA's   :289       NA's   :289
##   active_days      median_weekly_devoted_days    n_weeks
##  Min.   :  1.000   Min.   :1.000              Min.   : 0.000
##  1st Qu.:  1.000   1st Qu.:1.000              1st Qu.: 1.000
##  Median :  3.000   Median :1.000              Median : 2.000
##  Mean   :  7.725   Mean   :1.195              Mean   : 4.563
##  3rd Qu.:  7.000   3rd Qu.:1.000              3rd Qu.: 5.000
##  Max.   :266.000   Max.   :6.000              Max.   :68.000
##  NA's   :289       NA's   :289                NA's   :18
##      n_recs         periodicity      periodicity_variation median_streak
##  Min.   :  0.000   Min.   :  1.000   Min.   : 0.000        Min.   :1.000
##  1st Qu.:  1.000   1st Qu.:  3.500   1st Qu.: 7.319        1st Qu.:1.000
##  Median :  3.000   Median :  6.000   Median :12.918        Median :1.000
##  Mean   :  7.198   Mean   :  9.591   Mean   :17.045        Mean   :1.021
##  3rd Qu.:  6.000   3rd Qu.: 11.000   3rd Qu.:22.974        3rd Qu.:1.000
##  Max.   :266.000   Max.   :158.500   Max.   :95.692        Max.   :6.000
```

```
##  NA's   :18        NA's   :2513      NA's   :2513          NA's   :2513
##    sd_streak        max_streak         n_days          upper_area
##  Min.   :0.0000   Min.   : 1.000   Min.   :  1.000   Min.   :  470
##  1st Qu.:0.2310   1st Qu.: 2.000   1st Qu.:  1.000   1st Qu.:  475
##  Median :0.4472   Median : 2.000   Median :  3.000   Median :  832
##  Mean   :0.4689   Mean   : 2.473   Mean   :  8.561   Mean   : 1200
##  3rd Qu.:0.6045   3rd Qu.: 3.000   3rd Qu.:  7.000   3rd Qu.: 1405
##  Max.   :5.1134   Max.   :24.000   Max.   :388.000   Max.   :16070
##  NA's   :2515     NA's   :2513     NA's   :18        NA's   :1884
##    lower_area    upper_n_poly     lower_n_poly       ratio
##  Min.   : 143   Min.   : 1.000   Min.   : 1.00   Min.   :0.0245
##  1st Qu.: 145   1st Qu.: 1.000   1st Qu.: 1.00   1st Qu.:0.2386
##  Median : 203   Median : 1.000   Median : 1.00   Median :0.2939
##  Mean   : 296   Mean   : 2.258   Mean   : 1.66   Mean   :0.2704
##  3rd Qu.: 336   3rd Qu.: 3.000   3rd Qu.: 2.00   3rd Qu.:0.3059
##  Max.   :3002   Max.   :28.000   Max.   :15.00   Max.   :0.3779
##  NA's   :1884   NA's   :1884     NA's   :1884    NA's   :1884
##       n.x           taxa_breadth      taxa_prop            n.y
##  Min.   :   1.00   Min.   : 1.000   Min.   :0.01449   Min.   :   1.00
##  1st Qu.:   2.00   1st Qu.: 2.000   1st Qu.:0.02899   1st Qu.:   2.00
##  Median :   5.00   Median : 4.000   Median :0.05797   Median :   5.00
##  Mean   :  27.25   Mean   : 6.815   Mean   :0.09877   Mean   :  27.25
##  3rd Qu.:  15.00   3rd Qu.: 9.000   3rd Qu.:0.13043   3rd Qu.:  15.00
##  Max.   :3307.00   Max.   :56.000   Max.   :0.81159   Max.   :3307.00
##  NA's   :18        NA's   :18       NA's   :18        NA's   :18
##      median        median_diff         stdev              n
##  Min.   : 1.000   Min.   :-7.000   Min.   : 0.000   Min.   :   1.00
##  1st Qu.: 4.500   1st Qu.:-3.500   1st Qu.: 4.377   1st Qu.:   2.00
##  Median : 8.000   Median : 0.000   Median : 6.083   Median :   5.00
##  Mean   : 9.135   Mean   : 1.135   Mean   : 6.960   Mean   :  27.29
##  3rd Qu.:11.000   3rd Qu.: 3.000   3rd Qu.: 8.618   3rd Qu.:  15.00
##  Max.   :61.000   Max.   :53.000   Max.   :38.184   Max.   :3180.00
##  NA's   :44       NA's   :44       NA's   :861      NA's   :44
##  median_diff_abs       d10              d20              d30
##  Min.   : 0.000   Min.   :0.045   Min.   :0.071   Min.   :0.094
##  1st Qu.: 1.500   1st Qu.:0.117   1st Qu.:0.176   1st Qu.:0.217
##  Median : 3.000   Median :0.144   Median :0.227   Median :0.274
##  Mean   : 4.564   Mean   :0.163   Mean   :0.237   Mean   :0.286
##  3rd Qu.: 5.500   3rd Qu.:0.195   3rd Qu.:0.284   3rd Qu.:0.349
##  Max.   :53.000   Max.   :0.351   Max.   :0.547   Max.   :0.700
##  NA's   :44       NA's   :3874    NA's   :3877    NA's   :3877
##       d40              d50           mean_LL         median_LL
##  Min.   :0.114   Min.   :0.133   Min.   : 1.000   Min.   : 1.000
##  1st Qu.:0.243   1st Qu.:0.265   1st Qu.: 1.368   1st Qu.: 1.000
##  Median :0.304   Median :0.332   Median : 1.792   Median : 1.000
##  Mean   :0.321   Mean   :0.349   Mean   : 2.226   Mean   : 1.646
##  3rd Qu.:0.386   3rd Qu.:0.424   3rd Qu.: 2.556   3rd Qu.: 2.000
##  Max.   :0.808   Max.   :0.886   Max.   :10.973   Max.   :11.000
##  NA's   :3877    NA's   :3877    NA's   :3139     NA's   :3139
##     variance             p1            n_lists         log_n_lists
##  Min.   : 0.0000   Min.   :0.0000   Min.   :   1.00   Min.   :0.000
##  1st Qu.: 0.4909   1st Qu.:0.4583   1st Qu.:   1.00   1st Qu.:0.000
##  Median : 1.5158   Median :0.6111   Median :   3.00   Median :1.099
##  Mean   : 3.2860   Mean   :0.5893   Mean   :  10.96   Mean   :1.348
```

```
##  3rd Qu.: 4.2702   3rd Qu.:0.7619   3rd Qu.:   8.00   3rd Qu.:2.079
##  Max.   :38.2492   Max.   :1.0000   Max.   :1050.00   Max.   :6.957
##  NA's   :3139      NA's   :3139     NA's   :18        NA's   :18
```

Categorisation

We will first want to drop metrics that are clearly duplications of one another, i.e. do not provide additional information. Up until now we have simply been interested in gathering together as many metrics that provide information about the recorders as we can.

What to catagorise?

I think we will be better off by catagorising people in three different dimensions. Spatially, temporally and data content This makes the analyses an bit easier to handle as it breaks it down but also builds a modular system which might result in three types of spatial recording, 3 types of taxonomic recording and 3 types of temporal recording rather than 27 (`3*3*3`) recording types. Of course these 27 'types' could be easily reconstructed using the 'types' from each dimension. I think this will make the results easier to interpret and also probably reflects more the reality of the situation. I suspect a recorders spatial pattern of recording will be driven by different factors than their temporal recording pattern etc.

I have reviewed the 38 variables in the table above and reduced it to what I see as the most valuable metrics. Some of these metric do not make sense if very few records are availbale, as a result I suggest we only apply this to recorders with atleast 10 active summer days. Those that fail this test can be placed in their own group `Dabblers` (as in Boakes?). Variables marked with `[S]` use summer only data in there calculation.

Temporal

- *Activity ratio* [S]: this capture the proportion of a recorders time they dedicate to recording. "The proportion of days on which the volunteer was active in relation to the total days he/she remained linked to the project" (Ponciano and Brasileiro 2014)
- *Active days* [S]: this is a measure of the absolute effort given to recording. This is chosen over raw number of records which is likely to be taxa specific.
- *Median weekly devoted days* [S]: This is an adaptation of the Daily Devoted Time in (Ponciano and Brasileiro 2014) which can be adapted by using days in a week (summer only) rather than hours in a day. Unlike Activity Ratio this metric is not affected by periods of absence as it is only calculates across weeks when recording occured.
- *Periodicity* [S]: The median time elapsed between each pair of sequential active days. The captures regularity which isn't really captured by the other metrics.
- *Periodicity variation* [S]: "The standard deviation of the times elapsed between each pair of sequential active days" (Ponciano and Brasileiro 2014).

Spatial

- *Upper Area*: This is the area of the upper percentile (95%) polygon. This gives the extent of a recorders activites.
- *Upper Polygon Number*: This is the numbver of polygons that make up the upper percentile (95%) polygon. The greater the number the more fractured the records are.
- *Upper:Core Ratio*: This is the ration between the upper percentile (95%) and lower percentile (60%) polygons. This gives an indication of how aggregated the data points are.

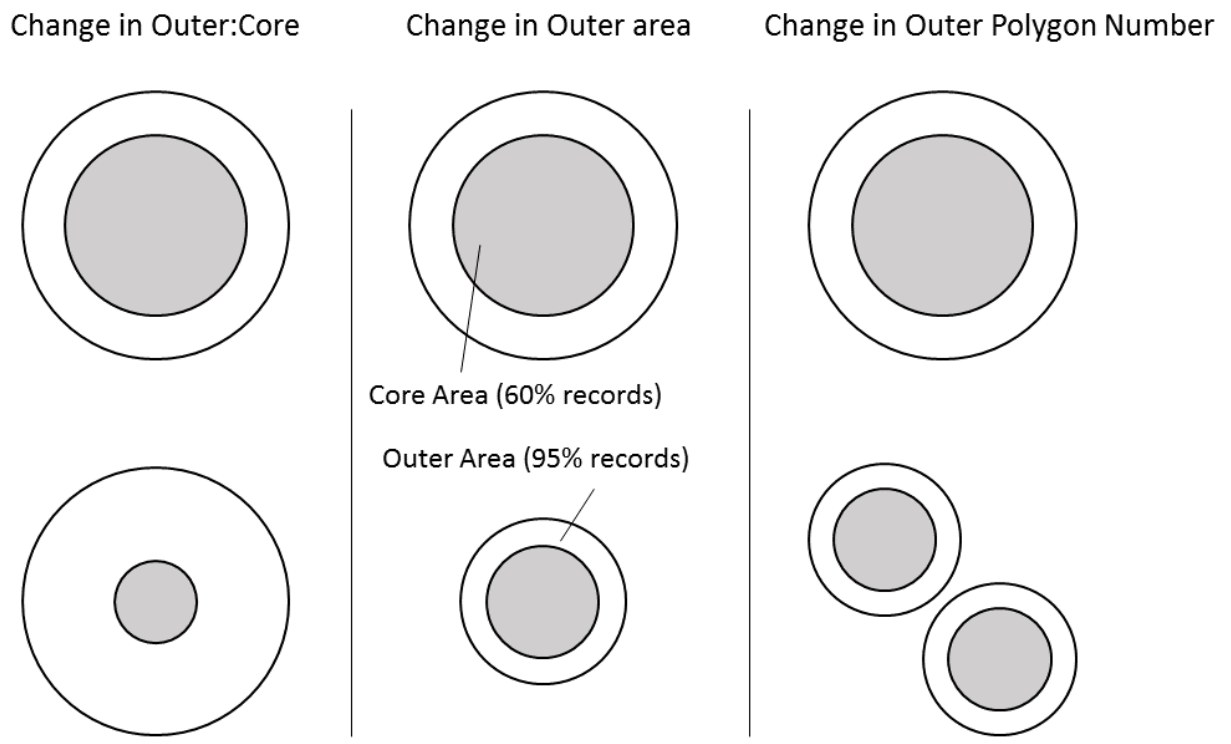


Figure 1: The three measures of recorder's spatial behaviour are all independent. Here each column shows a change in one variable while the others are held constant

Data content

- *Proportion of Taxa Recorded*: This is the number of unique taxa recorded by an individual as a proportion of the total number of taxa recorded by all recorders.
- *Median Rarity Difference*: Species are ranked according to the number of records in the entire dataset, and scaled to 100. This variable is the difference between the median rank overall and the median rank for the recorder. Negative numbers mean the recorder preferntially records common species while positive number means the recorder records rarer species.
- *p1*: The proportion of lists (day-location) that have a single record on them. This give the proportion of record that are 'ad-hoc'

```
key_variables <- all_metrics[,c('recorder', 'activity_ratio',
                                'active_days', 'median_weekly_devoted_days',
                                'periodicity', 'periodicity_variation',
                                'upper_area', 'upper_n_poly', 'ratio',
                                'taxa_prop', 'median_diff', 'p1')]
```

Dropping Dabblers

Drop the recorders with less that 10 active days

```
table(key_variables$active_days >= 10)
```

```
##
## FALSE  TRUE
##  3014   686
```

```
# The majority of recorders for this group are 'dabblers'

key_variables <- key_variables[key_variables$active_days >= 10, ]
```

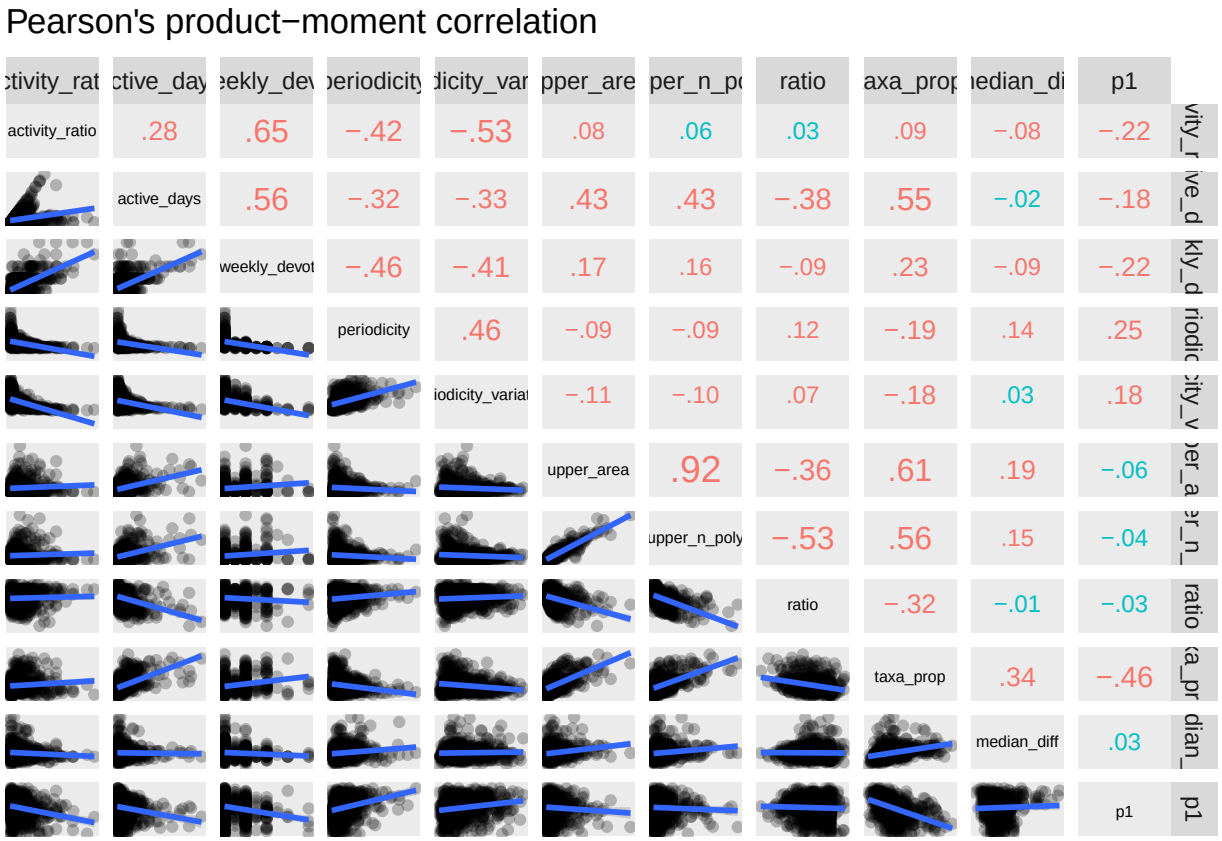
Testing Co-linearity

Let's have a look to see how correlated these selected measures are.

```
cor_data <- cor.matrix(variables = key_variables[,!colnames(key_variables) %in% 'recorder'])
ggcorplot(cor.mat = cor_data,
          data = key_variables,
          var_text_size = 2,
          cor_text_limits = c(3,5))
```

```
## Warning: Removed 15895 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 15895 rows containing missing values (geom_point).
```



Catagorisation methods

There are numerous methods for catagorisation out there and we need one which is going to give us the best results for our data. I think this method needs to:

- Work unsupervised. We do not want to have to catagorise recorders manually as we want the categorisation to emerge from the data.
- Work with `NA`s. We may have quite a few `NA`s in our data since some of the metrics can only be estimated for a subset of recorders, we want a method that can deal with this.
- Easy to reuse. We want a method that returns a model that can be applied to other data sets to see how our categories translate to other citizen science projects.

Principal components analysis

In the first instance it is probably a good idea to have a go at dimensionality reduction for each of the 3 'dimensions' and to assess the results.

```
summary(key_variables)
```

```
##             recorder   activity_ratio      active_days
##  Marsh, Nick    :  1   Min.   :0.01181   Min.   : 10.00
##  Limb, Ken      :  1   1st Qu.:0.02809   1st Qu.: 12.00
##  Ward, John     :  1   Median :0.05048   Median : 18.00
##  Turner, Lindsey:  1   Mean   :0.08942   Mean   : 29.14
##  Warren, Martin :  1   3rd Qu.:0.11364   3rd Qu.: 33.00
```

```
##  (Other)        :681   Max.   :0.72222   Max.   :266.00
##  NA's           :289   NA's   :289       NA's   :289
##  median_weekly_devoted_days  periodicity     periodicity_variation
##  Min.   :1.000              Min.   : 1.000   Min.   : 0.8348
##  1st Qu.:1.000              1st Qu.: 3.000   1st Qu.: 7.1186
##  Median :1.000              Median : 5.000   Median :11.6952
##  Mean   :1.398              Mean   : 5.861   Mean   :13.6743
##  3rd Qu.:2.000              3rd Qu.: 7.000   3rd Qu.:17.7538
##  Max.   :5.000              Max.   :36.000   Max.   :46.5451
##  NA's   :289                NA's   :289      NA's   :289
##    upper_area       upper_n_poly        ratio           taxa_prop
##  Min.   :  470.0   Min.   : 1.000   Min.   :0.02452   Min.   :0.04348
##  1st Qu.:  718.8   1st Qu.: 1.000   1st Qu.:0.19718   1st Qu.:0.17391
##  Median : 1360.5   Median : 2.000   Median :0.23850   Median :0.23188
##  Mean   : 1894.8   Mean   : 3.528   Mean   :0.23641   Mean   :0.26402
##  3rd Qu.: 2277.5   3rd Qu.: 4.000   3rd Qu.:0.29160   3rd Qu.:0.31884
##  Max.   :16070.0   Max.   :28.000   Max.   :0.36239   Max.   :0.81159
##  NA's   :289       NA's   :289      NA's   :289       NA's   :289
##   median_diff             p1
##  Min.   :-7.00000   Min.   :0.0000
##  1st Qu.:-2.00000   1st Qu.:0.4286
##  Median :-1.00000   Median :0.6000
##  Mean   : 0.08673   Mean   :0.5740
##  3rd Qu.: 1.50000   3rd Qu.:0.7412
##  Max.   :19.50000   Max.   :1.0000
##  NA's   :289        NA's   :289
```

```
library(vegan)

all.pca <- rda(na.omit(key_variables[,-1]), scale = TRUE)

summary(all.pca)
```

```
##
## Call:
## rda(X = na.omit(key_variables[, -1]), scale = TRUE)
##
## Partitioning of correlations:
##               Inertia Proportion
## Total              11          1
## Unconstrained      11          1
##
## Eigenvalues, and their contribution to the correlations
##
## Importance of components:
##                            PC1    PC2    PC3     PC4     PC5     PC6     PC7
## Eigenvalue            3.7090 2.2562 1.1726 0.94341 0.71758 0.68068 0.53006
## Proportion Explained  0.3372 0.2051 0.1066 0.08576 0.06523 0.06188 0.04819
## Cumulative Proportion 0.3372 0.5423 0.6489 0.73465 0.79989 0.86177 0.90995
##                            PC8     PC9    PC10    PC11
## Eigenvalue            0.49147 0.24736 0.19281 0.05887
## Proportion Explained  0.04468 0.02249 0.01753 0.00535
## Cumulative Proportion 0.95463 0.97712 0.99465 1.00000
##
## Scaling 2 for species and site scores
```

```
## * Species are scaled proportional to eigenvalues
## * Sites are unscaled: weighted dispersion equal on all dimensions
## * General scaling constant of scores:  9.316887
##
##
## Species scores
##
##                               PC1      PC2      PC3      PC4       PC5
## activity_ratio             -1.4127  1.78025 -0.03089  0.79532 -0.635507
## active_days                -2.1720  0.05379 -0.38026 -0.26807 -0.139913
## median_weekly_devoted_days -1.7978  1.42315 -0.26915  0.33104 -0.718127
## periodicity                 1.4660 -1.39817  0.18054  0.23653 -1.029275
## periodicity_variation       1.4565 -1.39896  0.06613 -0.75833 -0.820108
## upper_area                 -1.9690 -1.52312 -0.20226  0.22529 -0.774881
## upper_n_poly               -1.9821 -1.60296 -0.54436  0.07903 -0.453157
## ratio                       1.3324  1.08030  1.25366  0.50845 -1.299538
## taxa_prop                  -2.0682 -0.96522  1.20565 -0.34739  0.139149
## median_diff                -0.2857 -1.14592  1.50192  1.79680  0.691065
## p1                          1.0548 -0.62726 -1.83854  1.52810  0.003255
##                                PC6
## activity_ratio              0.1999
## active_days                -1.2284
## median_weekly_devoted_days -0.9535
## periodicity                -0.5800
## periodicity_variation      -0.8489
## upper_area                  0.8414
## upper_n_poly                0.7223
## ratio                       0.5063
## taxa_prop                  -0.2418
## median_diff                -0.5080
## p1                         -0.2320
##
##
## Site scores (weighted sums of species scores)
##
##            PC1       PC2        PC3        PC4        PC5       PC6
## 1     0.082971 -0.118179  0.3632484  0.6876254  0.0109890  0.150687
## 2    -0.781190 -0.026161  0.5501162 -0.0556917 -0.4848204  0.215209
## 3     0.084885  0.458480  0.5163445 -0.2197464  0.0085656  0.467299
## 5    -0.236246  0.673175  0.3975121 -0.2484695 -0.0281913 -0.194300
## 6    -1.731429 -0.699867  0.1349275 -0.1244318 -0.8226153 -0.100760
## 7    -1.150462  0.066285 -0.2444461 -0.0147549 -0.2208250 -1.093783
## 8    -0.944986  0.140749  0.3213529 -0.3766150  0.0104997 -0.938757
## 9    -1.307805 -0.215757 -0.0175232 -0.1868363 -0.7267845 -0.418178
## 10   -0.833337 -0.564083 -0.0582105  0.2932753  0.0154144 -0.018891
## 12   -0.591024  0.067102 -0.5332165 -0.3089348  0.2074966 -0.506775
## 13   -0.427851  0.005108  0.6158839 -0.4739400  0.4203740 -0.229478
## 14   -0.277378  0.653180  0.0950235  0.0527955 -0.2004239 -0.524998
## 15   -0.167959  0.307334  0.4188479 -0.0715614  0.1518541 -0.275960
## 16    0.179547  0.597862 -0.2642201  0.3421871 -0.2064913  0.399262
## 17   -0.720106 -0.301539  0.2214296 -0.2371665 -0.2017148  0.169716
## 18   -0.037478  0.579691  0.2682601  0.4621580 -0.0160510  0.136957
## 19   -0.151562  0.134195  0.0171531 -0.0344298  0.2150544  0.091703
## 20    0.231350  0.065174  0.3463110 -0.0905943  0.0513332 -0.043323
```

```
## 21   -0.002493  0.332477  0.0823098  0.0001461  0.0031544 -0.231069
## 22   -0.358473  0.503408 -0.5778603 -0.0080084  0.0345917 -0.494914
## 23    0.219589  0.253606  0.0632685 -0.1489945  0.1617368  0.382873
## 24   -0.812307 -0.073741 -0.0815558  0.0867365  0.1782078 -0.452953
## 25    0.187398  0.241742  0.2748973  0.1859566  0.0617863  0.232721
## 26    0.178277  0.088214 -0.0165994 -0.0007540  0.0654621 -0.129496
## 27   -1.011866 -1.101649  0.6113852 -0.1167641 -0.4336806  1.032263
## 28    0.084598  0.205043  0.2740466  0.0825442  0.2346528  0.270706
## 30   -0.261365 -0.505381  0.3485636  0.0749632  0.1652569  0.310053
## 34   -0.627090 -0.212731  0.1462386 -0.2604131 -0.1552644  0.259013
## 35   -0.211543 -0.074650  0.2987628 -0.3493765  0.6800675 -0.077352
## 36   -0.350965 -0.035400  0.8280610  1.1174206  0.1469799 -0.643074
## 37   -0.073171 -0.223366  0.2035545 -0.4653935 -0.4387092  0.533677
## 38   -0.256702  0.956321  0.0980965  0.5316366 -0.4216545 -0.085392
## 39   -1.729340  0.061744 -0.4959542 -0.4175550 -0.2321444 -1.835495
## 40   -0.634687  0.308679  0.9255199 -0.0446527 -0.1551384  0.202960
## 41   -0.522975 -0.459249  0.6307427 -0.2509141  0.3381060  0.051346
## 42   -0.664560  0.263579 -0.0270766  0.1237836 -0.7488845 -0.514246
## 44   -0.269642  0.617051  0.1242137 -0.2644555 -0.2171779 -0.431733
## 45   -0.248608 -0.177424 -0.4916374 -0.2018606  0.3912861  0.011493
## 46    0.387517 -0.068239  0.3083643 -0.3210060 -0.2735836 -0.266860
## 47    0.033742 -0.160542  0.1754987 -0.3223541  0.1601894 -0.034446
## 48    0.023312 -0.088784 -0.6233754 -0.1019689  0.2529349  0.321733
## 49   -0.060467 -0.487324  0.1406576  0.6319353  0.4668184 -0.217675
## 50    0.072816  0.286935 -0.4429720  0.1035732  0.2188914  0.472484
## 51   -0.065015  0.255409 -0.0175949 -0.1811181  0.1588057 -0.306753
## 53    0.238910  0.221717 -0.0572791 -0.1209011 -0.0855952  0.073973
## 54    0.054142 -0.087260  0.3385169 -0.5653007  0.1798008 -0.079513
## 55    0.019835  0.243119  0.2432861 -0.0066172  0.0830693 -0.161184
## 56   -0.910769 -0.236421  0.3565089 -0.3157649  0.0338110 -0.203479
## 57   -0.117709  0.044563  0.2112297 -0.0481863  0.0847003  0.108069
## 58   -1.485562 -1.297027 -0.0075918 -0.3432144 -1.3636780  1.594388
## 59   -0.144956  0.094707  0.4614005 -0.2801834  0.3676288 -0.281003
## 60    0.081958  0.112484 -0.2600662 -0.4048771  0.2340550 -0.030572
## 63   -0.663556  0.531963  0.4931863 -0.0565994 -0.1005696 -0.198416
## 64    0.010900 -0.007970  0.6268733 -0.6223215  0.0485362  0.234087
## 65   -0.375988  0.151098  0.4379337 -0.4087437  0.2466695 -0.657417
## 68   -1.448940 -0.309608 -0.3384445 -0.0739946 -0.4787605 -0.782732
## 69   -0.351095 -0.115243 -0.3355973 -0.2235914  0.6787280 -0.275819
## 70   -0.925581 -0.506877 -0.5970669 -0.1162790 -0.1765575  0.309562
## 71    0.070909  0.135361  1.1458894 -0.0855161  0.1808221  0.270639
## 72   -1.551447 -0.190629 -0.6945293  0.2360647 -0.7661136 -0.832331
## 73   -0.685462 -0.855506  0.2919082  0.1206549 -0.2389504  0.537105
## 76    0.390853 -0.102834 -0.2430427 -0.3578759 -0.1311684 -0.182639
## 77    0.255837 -0.428568 -0.1871518 -0.0911798 -0.0881353 -0.409047
## 79   -0.960494  1.021446  0.0286037 -0.2739624 -0.7784421 -2.134135
## 81    0.055827  0.536110 -0.2636550  0.1760210  0.0207812  0.130537
## 82   -0.804319 -0.152347  0.4182300 -0.2183444  0.4315716 -0.564315
## 83    0.235721 -0.119706  0.2762687 -0.1076319  0.1681159 -0.081255
## 85    0.035783  0.373912  0.2639525  0.3259099  0.1467386  0.274858
## 86   -0.560798 -0.138133 -0.7284765 -0.2750794  0.3334285 -0.163860
## 87   -1.074115  0.933187 -0.4568789  0.3650082 -0.5527821 -2.007516
## 88   -0.702211  0.308749  0.2369646  0.0685811 -0.1694739 -0.836637
## 92   -0.178876  0.307253  0.3184868 -0.2998227  0.3609253 -0.005104
```

```
## 93    0.071358  0.256696  0.0084486 -0.0270309 -0.1828951 -0.306422
## 94   -0.050859  0.477841  0.9504763  0.0558277 -0.0907734  0.564196
## 95    0.262631  0.043179  0.4099498 -0.3911774 -0.0393717 -0.062447
## 96    0.172608  0.001937  0.1356071 -0.7373370  0.2206550 -0.095873
## 99   -0.145059 -0.269753  0.0049136 -0.0808662  0.5862837  0.107602
## 100   0.266030  0.298052  0.1370864 -0.2791394 -0.1494982  0.301768
## 101   0.268743 -0.217420  0.1175200  0.6183649 -0.1805062 -0.029075
## 102  -0.602904 -0.118525 -0.8133825 -0.1462535  0.3463184 -0.296157
## 103   0.184143  0.360357 -0.1304437 -0.1080435 -0.6118742  0.137657
## 104   0.323766 -0.330208  0.3490162  0.1473868 -0.5297581 -0.203171
## 106  -0.148650 -0.311953 -0.2245867 -0.2096900  0.2852837  0.203831
## 107  -0.319671 -0.077646 -0.4429597 -0.1168989  0.3959867 -0.147748
## 108   0.206738  0.217669  0.0157142 -0.4317814 -0.1820391 -0.048065
## 111  -0.327781  0.634581 -0.0656465  0.0377417  0.0333613 -0.203708
## 112  -0.067039 -0.016306  0.5139433 -0.1943273  0.4489044 -0.077114
## 114   0.129787  0.164204 -0.2980201 -0.0611313  0.1641599  0.071484
## 115   0.228145  0.160736  0.1658412  0.0407943 -0.2421337 -0.293124
## 116  -0.084897  0.402678  0.3453458 -0.2769585  0.0022713  0.131966
## 117  -0.543092  0.592882 -0.2200734 -0.0215456 -0.2959423 -1.099386
## 119  -0.009871 -0.331832  0.3060190 -0.3461909  0.4808936 -0.307294
## 121   0.032875  0.550958  0.0250117  0.0219218 -0.1080735  0.058269
## 122   0.288005 -0.421385 -0.1365454  0.3748679 -0.1089591 -0.284991
## 126  -0.250129  0.578903  0.2231340 -0.4266909 -0.0724642  0.347929
## 127  -0.138484 -0.027430  0.3506475 -0.3068540 -0.0447423  0.527078
## 129   0.132097  0.194364  0.3234121 -0.2744654  0.0695065  0.061561
## 131  -0.495717  0.360996  0.4576419 -0.1535056 -0.0211394 -0.750649
## 133  -0.130758  0.005156  0.4443241 -0.1755691  0.4226458 -0.046913
## 135   0.151670 -0.148703  0.8282893 -0.2790229 -0.0275238 -0.296990
## 138  -0.450589 -0.375656  1.1033948 -0.2271431  0.1968614  0.326427
## 139  -1.335152 -0.838162 -0.5681102  0.0635735 -0.9427458  0.669748
## 140  -0.205266 -0.229971 -0.2825454 -0.0142139  0.6407550  0.070761
## 144   0.270830  0.256461 -0.0562449 -0.0069812  0.1469033  0.301354
## 145   0.006073  0.103773  0.2122984 -0.4455083  0.2198913 -0.144512
## 146  -0.455227 -0.143867 -0.2912221 -0.1918750  0.3587505 -0.033356
## 148  -0.042866 -0.253978  0.5194113 -0.1939411  0.2931773  0.006583
## 149   0.137205  0.129407  0.2569412  0.2295002  0.0538811  0.172231
## 150  -0.033689 -0.013802  0.0437603 -0.5410814  0.3478655 -0.207775
## 152   0.194842  0.132792  0.0420221 -0.1995229  0.0565173 -0.106536
## 153  -0.084512  0.479954 -0.2271097 -0.1240531 -0.1993352 -0.423975
## 154  -1.840606 -0.710471 -1.0822003  0.0190528 -0.5352471 -0.446508
## 155  -0.220126  0.019288  0.9589437  0.1121154  0.0254863 -0.301963
## 157   0.165441  0.346194  0.0008294 -0.0050124  0.1825561  0.267632
## 160  -0.556022 -0.697727  0.5016268  0.9679255  0.3034915 -0.010315
## 161  -0.239356  0.106898 -0.1868569  0.0831643  0.2020065 -0.042074
## 162   0.493074 -0.542809 -0.1814809  0.0139611 -0.8167846 -0.548892
## 166   0.029641  0.122887  0.3526052 -0.4517919  0.3655851  0.394576
## 167  -0.461580 -0.279555 -0.3135651  0.2043451 -0.1176153  0.799853
## 168   0.009834 -0.154011  0.5528828 -0.6594686  0.1406675 -0.143712
## 169   0.354186 -0.240418  0.4508868 -0.3272826 -0.5145206 -0.276308
## 170  -0.218656  0.966239  0.0690631  0.2964653 -0.3624126  0.194627
## 171   0.493857 -0.385730 -0.4721771  0.0100257 -0.8908538 -0.238416
## 172   0.191344  0.201947 -0.0816895 -0.1119991 -0.1971328 -0.254014
## 175  -0.301509  0.241229 -0.2195840  0.0173379  0.1245069 -0.236580
## 176  -0.025509 -0.192453  0.3939531 -0.4465095  0.3816336 -0.313170
```

```
## 178  -0.060397  0.335473  0.6281443 -0.2977095 -0.2489704  0.045390
## 179  -0.739962 -0.067675  0.1013228 -0.2810415  0.4084799 -0.543341
## 180   0.006402 -0.023854  0.1075987 -0.0602041 -0.3616318 -0.025158
## 181  -0.336585 -0.363949 -0.3421665 -0.1130344 -0.0516872  0.734883
## 182  -0.296841  0.080932 -0.0715889  0.0355563  0.6107421 -0.371004
## 183  -0.528281 -0.530225  0.0376541  0.0859797  0.3947462  0.325324
## 186  -0.386163  0.004467 -0.4427453 -0.0355703  0.4428998 -0.450399
## 187  -0.142946  0.208451  0.4930354 -0.3684377  0.0629223 -0.079282
## 188   0.099318 -0.364610  0.2179946 -0.1332119  0.2043908 -0.099344
## 193   0.270767 -0.374733 -0.3588382 -0.0029500 -0.8585730  0.261015
## 194  -0.117636  0.118934  0.0080025 -0.0627476  0.5209645  0.224852
## 195   0.396234  0.034790 -0.2816115  0.3015289 -0.2119453 -0.016393
## 196  -0.519453  0.042485 -0.1879780  0.0703896 -0.6492204  0.829999
## 197  -0.225968 -0.240645  0.8146012 -0.5087648  0.1983226 -0.095899
## 198  -0.330408 -0.045187 -0.5752493  0.3751799  0.1138955  0.006283
## 200  -1.000391 -0.420273 -0.3669502 -0.0027357  0.0604065 -0.245504
## 201  -0.088246 -0.396534  0.0201672 -0.2578914 -0.3683004  0.360848
## 203  -0.108403 -0.195833  0.1939358 -0.2749643 -0.2318961  0.482037
## 204   0.022377  0.250087 -0.3075292  0.1088385  0.0551528  0.206116
## 205   0.225756 -0.225466 -0.0452468 -0.3188041 -0.0429112 -0.093669
## 206  -0.273565 -0.207228 -0.1525436 -0.2470311  0.4719132  0.002950
## 207  -0.493855 -0.065585  0.2815873  0.0993728 -0.7228111  0.114060
## 208  -0.336590  0.137611 -0.3065671 -0.3185908  0.4259780 -0.423633
## 209   0.082512 -0.031220 -0.0404690 -0.0367930  0.2335079  0.193649
## 210   0.097823  0.043365 -0.1679362 -0.3626815  0.2009234  0.001580
## 212  -0.193240  0.464759 -0.0191268  0.3138925 -0.6274747  0.174893
## 215  -1.201516 -0.255537  0.0199366  0.8597243 -1.1595510  1.004513
## 217  -0.523494  0.609886 -0.3222127 -0.0350745  0.0160122 -1.105710
## 219  -0.519430  0.290968 -0.3945817  0.2811052  0.1039905  0.075160
## 224  -0.204664  0.161769  0.2960693 -0.1124465  0.3457854 -0.427127
## 225  -0.052441 -0.486319 -0.4147363  0.3249854  0.5693463 -0.049893
## 228  -0.328926  0.778752  0.0367768  0.3089865 -0.5240887  0.381432
## 234  -0.194954 -0.520315  0.0170871  0.3192420 -0.2949157  0.429809
## 236   0.135611  0.301593  0.2886585 -0.1641537  0.1189786  0.239315
## 238  -0.499018  0.380917 -0.0349082  0.3603642  0.0820541 -0.029054
## 239   0.286914 -0.366911  0.2489144  0.1743630 -0.1781052 -0.372784
## 240  -1.027402 -0.747314  0.2007717  0.2557325 -0.7639750  0.654251
## 242  -0.784853 -0.858632 -0.5815517  0.0244823 -0.8436465  1.102849
## 243  -0.121573  0.640677  0.2400721 -0.2902987 -0.0926685 -0.097441
## 244   0.192091  0.160225  0.4045448 -0.1251784  0.0876835  0.182401
## 246  -0.455810  0.099753  0.3704850 -0.0587904 -0.0220436 -0.213315
## 247  -0.974697 -0.240040  0.1548240 -0.4527214  0.4500011 -0.681964
## 250  -0.240079 -0.018482 -0.1843548 -0.1984339  0.3598622 -0.400881
## 251   0.080040  0.666753 -0.1757937  0.2202633 -0.2770280  0.285419
## 252   0.070143  0.344382  0.1465103  0.3112021 -0.0989509 -0.122518
## 253   0.298144 -0.317166 -0.0385804  0.3625124 -0.3689600 -0.001280
## 255   0.086543  0.116339  0.1586777 -0.5282940  0.1282690  0.370917
## 256  -0.502452 -0.103105 -0.3226691  0.0681593  0.4631725 -0.520387
## 257  -0.250310 -0.432343  0.2881964 -0.4242878 -0.3646706  0.561580
## 258  -0.051061  0.554975  0.2070696 -0.1441643 -0.2196132 -0.310825
## 259   0.125784 -0.129584 -0.3323326 -0.1188068  0.4959032  0.085676
## 261   0.101524 -0.067313 -0.2653263 -0.1646751  0.2317527 -0.136901
## 262   0.066010 -0.341057  0.5693768 -0.0617927 -0.0070064 -0.165071
## 263  -0.463291 -0.024768 -0.3227842 -0.0694081  0.4521849 -0.401722
```

```
## 264  -0.530877 -0.261793 -0.5707457 -0.1807047  0.2994248 -0.047401
## 265  -0.706541  0.026961  0.4065066 -0.1740407 -0.7240142  1.095229
## 267  -0.218521  0.355057  0.8661240 -0.2065179  0.1117011  0.532658
## 269   0.104919  0.011102  0.3901723 -0.5157853 -0.0400289  0.070527
## 270   0.476936  0.023020 -0.1843185 -0.2998950 -0.4375984 -0.098018
## 271  -0.216046  0.003417  0.0646732 -0.3414271  0.3227282  0.113732
## 272  -0.872483 -0.162920 -0.5968854  0.0606021  0.0037438  0.588448
## 273   0.012239  0.102383  0.5533194 -0.4437850  0.0505777  0.471580
## 274   0.364652 -0.081412  0.0566192  0.3109409 -0.5657336  0.181960
## 278   0.167687 -0.298104  0.5554155 -0.1323089  0.0342034 -0.370724
## 279  -0.610646  0.248715  0.2957730 -0.5278403  0.5476516 -0.615240
## 280   0.411673  0.120736 -0.1164319  0.0665397 -0.2293492  0.122426
## 282   0.172261 -0.057473 -0.3358567 -0.2270395  0.1551149  0.289051
## 283  -0.708687 -0.388289  0.2094584  0.1678150  0.1948536  0.009593
## 285   0.091234  0.073172  0.3682135 -0.0536005 -0.1577177  0.355276
## 286  -0.245832  0.720113 -0.2043370  0.4901027 -0.3106572  0.390051
## 287   0.033331 -0.031942  0.6840849 -0.2847277  0.0743356  0.168841
## 288  -0.017782  0.114636  0.1191304  0.1224931  0.6503025 -0.028851
## 289  -0.661098 -0.897044  0.3132950  0.1855407 -0.3332201  0.762138
## 290  -0.416199  0.505546 -0.1527885  0.1798017  0.0952681 -0.742702
## 291  -0.343882  0.742299  0.1264393 -0.0801210 -0.3286242 -0.609340
## 292   0.179726 -0.256458  0.3729921  0.0448622 -0.1241353 -0.074165
## 293   0.250899  0.193922  0.3225889 -0.3413023  0.0363365  0.168619
## 294  -0.398644 -0.094342 -0.0896385 -0.2060979 -0.1894745  0.084490
## 295  -0.035535 -0.269846 -0.4655375 -0.3278544  0.4206275  0.048463
## 297   0.027929  0.080962  0.0714584 -0.4685888  0.3541154 -0.012356
## 300   0.153551 -0.100859  0.6449157 -0.6555041  0.1259933 -0.116879
## 301  -0.007446  0.392965  0.0886972 -0.3634560 -0.1136356 -0.250389
## 302   0.072724 -0.271137 -0.1632608  0.3370282  0.1141734  0.365459
## 304   0.130391  0.079963 -0.0966950 -0.0231090  0.3670706  0.343666
## 307  -0.352282  0.038135 -0.4869839 -0.2871300  0.2580411 -0.323874
## 308  -0.389317  0.308462  0.2551719 -0.2554784  0.1928234 -0.119533
## 309   0.354442 -0.112209  0.4397577  0.4765847  0.0719388 -0.223143
## 310   0.286617 -0.019670 -0.0303384 -0.1681269  0.0548046 -0.019014
## 311   0.372069 -0.198304  0.2561450 -0.0475071 -0.4329605 -0.286391
## 312  -0.195439  0.596822 -0.3710122  0.3239593 -0.1800931  0.484818
## 313   0.081785  0.324795 -0.3093121  0.4612683  0.0343079 -0.018968
## 315  -0.402718  0.145959  0.1329261 -0.4793458  0.5557983 -0.028328
## 316  -0.048081  0.309478  0.3049037 -0.3199370  0.1356757  0.373833
## 317   0.079056 -0.045376  1.1816806 -0.1855256  0.1319273 -0.074643
## 318  -0.218529  0.404519  0.0671005  0.0002230  0.1757925 -0.038755
## 321  -0.648423 -0.761105 -0.2142390 -0.2052735 -0.6046745  1.186291
## 322   0.055415 -0.922707  0.1543564  0.5942575 -0.5019461 -0.225933
## 323   0.037662 -0.186672 -0.2448986  0.0258987  0.1776532  0.183544
## 326   0.074390  0.445615  0.5506657 -0.2285895  0.0118331  0.470820
## 328  -0.026078 -0.485133 -0.1002815 -0.0623251  0.4199511 -0.206329
## 333  -0.887073  0.300756  1.0253281  0.4796964 -0.8172012  0.291896
## 335  -0.317312  0.330792  0.6359082  0.0438384  0.2184552 -0.112353
## 336  -0.086239 -0.151330  0.1057329 -0.1401509  0.0919266  0.194707
## 338  -0.055489 -0.212560 -0.5221453 -0.0270254  0.2121627  0.424933
## 339   0.007310 -0.243169  0.0660527 -0.1178935 -0.2789966  0.092182
## 340   0.294612 -0.353481 -0.1244440  0.1868525  0.0038871 -0.368023
## 341   0.204185 -0.202533 -0.5724594 -0.9195051  0.1091936 -0.103753
## 342   0.075944  0.090626  0.5569340 -0.4707663 -0.0552008 -0.123939
```

```
## 343   0.021670  0.393640  0.5428143 -0.5518431  0.1268511  0.428636
## 344  -0.229430 -0.732529 -0.5144425  0.3055028 -0.4333337  0.591333
## 345   0.313529  0.119282  0.1933169 -0.2604705 -0.0727873  0.040970
## 347  -0.110586  0.006176 -0.6159279 -0.0957709  0.0698252 -0.298278
## 348   0.199291  0.034021  0.2820514  0.1461748  0.1647694  0.081316
## 349  -0.175107 -0.320607  0.6502145 -0.3624685  0.3338001 -0.163227
## 351   0.009527 -0.130130  0.1091897 -0.1785075  0.4927064  0.108891
## 352   0.240751 -0.442438  0.1802941 -0.8016611 -0.2995234 -0.427760
## 353   0.068778 -0.029594  0.3317497 -0.1839835  0.3573185  0.005433
## 354   0.075177  0.159529  0.1050942 -0.0794949  0.1766468 -0.197825
## 355  -0.149464 -0.529416  0.9684816  0.1893454  0.0124054 -0.347156
## 358   0.065962  0.072483 -0.0161442  0.3534574  0.1541841  0.320736
## 359   0.010175 -0.266478 -0.0406743 -0.2464917  0.0297281  0.035701
## 360  -0.035061 -0.291837  0.0119545 -0.0030853  0.3865655 -0.074738
## 361  -0.709526  1.518862  0.4479724  0.4123028 -1.0911388 -0.457075
## 365   0.199023 -0.025536 -0.3111132 -0.1072686  0.1078193  0.087901
## 367   0.131496  0.044682  0.0846310  0.0909420 -0.1485098 -0.288217
## 368   0.463064 -0.149515 -0.0256985 -0.4444244 -0.7768139 -0.077175
## 370   0.130177  0.293025  0.0814806 -0.3780918  0.0259826  0.055875
## 371   0.161909 -0.435168  0.3150357 -0.1923125  0.1700810 -0.499222
## 372   0.139903 -0.105848  0.0193168 -0.1206304  0.4658891 -0.122095
## 373   0.065533  0.169023  0.3024183  0.5223360  0.2183018  0.220537
## 374  -0.154980  0.356137  0.0008621 -0.0614035  0.1524895  0.014725
## 375   0.050018  0.004926  0.1087837 -0.2154188  0.2507688  0.120166
## 377   0.076404  0.339999  0.8090965 -0.5358846  0.1664677  0.556438
## 381   0.120810 -0.444991  0.1069053 -0.7755729 -0.0402482 -0.348953
## 383  -0.137103 -0.249049 -0.4239722  0.0897118  0.5068259  0.009594
## 385   0.408010 -0.011010 -0.2780770  0.2769017 -0.2776908 -0.029337
## 388  -0.353191  0.247199 -0.5292262  0.0826843  0.4428524 -0.513773
## 390   0.209979 -0.034049 -0.7439106 -0.0702602  0.1027047 -0.253235
## 391   0.109954 -0.221222 -0.1101559 -0.1847657 -0.1345509  0.126781
## 392  -0.087992  0.035948 -0.4878857  0.0372043  0.4180900 -0.135214
## 393   0.199984 -0.441618  0.9435592 -0.2380590 -0.6438829 -0.242451
## 397   0.043127  0.086761 -0.3251753  0.0789188  0.3380039 -0.137124
## 400  -0.193360 -0.200226 -0.2017898 -0.0299455  0.4611088  0.248175
## 401  -0.009320 -0.140206  0.2251772  0.3640884  0.6791504 -0.291123
## 409  -0.011193  0.358864  0.4795831  0.0235503  0.1843476 -0.037973
## 410  -0.243292 -0.223523 -0.2641035 -0.2861527  0.5856594  0.221800
## 412   0.138727  0.087466 -0.3370307 -0.2443535  0.2189212  0.099576
## 413   0.218361  0.124643  0.6461716 -0.6407501 -0.1089242  0.082868
## 418   0.291354 -0.038184 -0.0472358 -0.7412809 -0.0179639 -0.056892
## 423  -0.144036  0.011369 -0.1216625  0.0346220  0.3943692 -0.058163
## 425  -0.179466 -0.040250 -0.5652111 -0.1758315  0.4463661 -0.185408
## 426  -0.269453 -0.198404  0.1285808 -0.3475458 -0.1071273  0.744289
## 427  -0.237365 -0.303274  0.0374090 -0.5698423  0.3782710 -0.325072
## 429   0.133696 -0.301732  0.1879005 -0.4306715  0.0009705 -0.266944
## 430  -0.124362  0.277865  0.3184391 -0.3349392 -0.0208249 -0.105792
## 432   0.025385  0.209463  0.6277684  0.0750790  0.1266308 -0.228779
## 433   0.221612 -0.007450 -0.2875667 -0.0920014 -0.0235407  0.072697
## 434   0.242572  0.183710  0.2000495 -0.3010650 -0.0816913  0.072062
## 436  -0.731812 -0.876386  0.2882985  0.9941432  0.0592060 -0.111336
## 438   0.213188 -0.638777  0.6423474  1.4531899  0.7258499 -0.716592
## 440   0.117355 -0.040052  0.0191887  0.4167076  0.5600402  0.048153
## 444  -0.103568  0.507640  0.2072936  0.2513797 -0.1171584  0.187024
```

```
## 445  -0.232630 -0.371739  0.0817851 -0.1133498  0.2945491  0.166441
## 447   0.412873 -0.048139  0.1650353 -0.0872649 -0.3707287 -0.121155
## 448  -0.064211  0.256520 -0.4521098 -0.1308395  0.0517398 -0.005550
## 449   0.233813  0.192663 -0.0670934  0.0628420  0.0607007  0.017647
## 450   0.248088  0.257095 -0.3688861  0.0804829  0.0469997  0.445714
## 451   0.107103 -0.165494 -0.2861866 -0.1940142  0.3227779 -0.122506
## 453   0.153702  0.081053 -0.0060951  0.3623193  0.5016232 -0.022801
## 454  -0.100560  0.694697 -0.0142960  0.2564900 -0.2588056  0.030080
## 455   0.089842  0.399623  0.0016958  0.2108773  0.0050946  0.172014
## 456  -0.333871  0.127268  0.3691213 -0.0287508  0.4688360 -0.500714
## 457   0.060254  0.094166 -0.2585422  0.0497832  0.0338201  0.058330
## 461   0.382778 -0.128545  0.0539291  0.1646691 -0.0413557 -0.231753
## 464   0.207619  0.033987  0.3525388 -0.0318830  0.1495946 -0.095142
## 465   0.184059 -0.037194 -0.2129914 -0.4992917 -0.0852275 -0.381849
## 466   0.112048 -0.357261 -0.2006705 -0.3747339  0.0674739 -0.184433
## 470  -0.030544  0.056081 -0.1070579 -0.1596112  0.3580813  0.298567
## 473   0.180516 -0.009806 -0.4128899 -0.1595012  0.2907989 -0.055285
## 475  -0.029450  0.880484 -0.1468888 -0.0065912 -0.2947321  0.452710
## 476   0.130022 -0.396942 -0.2420504  0.1412433 -0.3139861  0.083026
## 478   0.121948  0.241152 -0.1491880 -0.1089405  0.2367133  0.208477
## 481  -0.138590  0.215180  0.0762171  0.0644114  0.2897667 -0.245355
## 482   0.255289 -0.141881  0.3127746 -0.3750473 -0.1507824 -0.092888
## 485   0.100617 -0.165736  0.4613903  0.2006846 -0.1161384  0.109674
## 486   0.134692 -0.063561 -0.0947994 -0.2796939  0.1892399 -0.114003
## 487  -0.005038 -0.041567  0.1260428  0.0545366  0.4751684 -0.236354
## 488  -0.293753  0.787569  0.0414978  0.0841476 -0.2517174 -0.105506
## 489   0.111818  0.307795  0.0866078 -0.0916827  0.1546098  0.312380
## 490   0.336189 -0.156687 -0.2831629  0.2548322  0.2480552 -0.197837
## 492  -0.991537 -0.636476 -0.1406359  0.1139785 -0.3596993  0.588861
## 495  -0.134707 -0.008010 -0.3453602 -0.1879820  0.1435133 -0.449024
## 499   0.028828  0.252960  0.0592282  0.0352703  0.0421952 -0.335453
## 502   0.180038 -0.074555  0.1313820  0.1112608  0.1714296 -0.137732
## 503  -0.105173 -0.347421  0.1176921 -0.1460822 -0.2845316  0.358010
## 507   0.229683  0.157318  0.5671123 -0.7706371 -0.1731942  0.037857
## 509  -0.629840  0.005145 -0.2812785  0.0732401  0.1837002  0.312709
## 512   0.109601  0.390323 -0.5181320  0.1886412  0.0135421  0.307569
## 513   0.253741 -0.152031  0.8932783 -0.6051764 -0.5056031 -0.095954
## 514  -0.259184 -0.180309  0.0909086  0.3110344 -0.0039018  0.246523
## 515   0.251079 -0.377357  0.3675481 -0.0902118 -0.0001773 -0.427565
## 516   0.112305 -0.179092 -0.2812722 -0.1821669 -0.1616603  0.417013
## 518   0.091296 -0.007866 -0.1473409  0.0120569  0.3660339  0.086358
## 519  -0.265093 -0.196210 -0.5264184 -0.1967551 -0.0413765  0.866572
## 521   0.217435  0.117727  0.5937381 -0.1907421  0.1695543  0.137321
## 523  -0.053178 -0.371131  0.3364911 -0.0043052  0.1489102  0.160534
## 524   0.134803 -0.006579 -0.3636435  0.2269629  0.3683203  0.035837
## 525  -0.139551  0.156838  0.0949053  0.0267438  0.5221247  0.205557
## 528   0.194529  0.308994  0.1411020 -0.0546342 -0.0376879  0.295794
## 529  -0.388880 -0.060145 -0.4474720 -0.2772284  0.2819242 -0.217324
## 530  -0.149442 -0.179186 -0.3170546 -0.2892941  0.3015886  0.005178
## 532   0.366067 -0.107340  0.1775183  0.7453448  0.1692240 -0.145613
## 537   0.216754  0.166941  0.0358446  0.1360517 -0.3190127  0.364228
## 539  -0.513232 -0.755371  0.4403953  0.5609576  0.1349646  0.404264
## 542   0.076855  0.191512 -0.1833456 -0.6567523  0.2967001  0.023436
## 543   0.109558 -0.192032 -0.2665198  0.0253766  0.1888897  0.001730
```

```
## 544   0.170903  0.243347  0.4004591 -0.3231376  0.0566373  0.194858
## 546   0.045682  0.508126  0.0632741  0.1913608 -0.1619778  0.143305
## 547   0.092358 -0.152431 -0.4244893  0.4916735  0.4351741  0.005984
## 549   0.303251  0.014926  0.1416410 -0.0567606 -0.2434508 -0.005838
## 551  -0.025473 -0.090377 -0.4792276 -0.3210473 -0.0852900  0.481671
## 553  -0.099320 -0.126162 -0.2628890 -0.2701132  0.2303256  0.251768
## 557  -0.481706 -0.170515  0.1683461  0.1162126  0.0284935  0.020240
## 558  -0.419043  1.490520 -0.0207494  0.7614320 -1.0420499  0.031029
## 563   0.274171 -0.206854 -0.5066724 -0.4764604 -0.2394135 -0.427023
## 564   0.099279 -0.153537 -0.1003924 -0.4067151  0.2810658 -0.035292
## 568   0.413800  0.186756 -0.3183504 -0.1982388 -0.0929714  0.165137
## 569   0.558871 -0.506761  0.6244777  0.2621312 -0.8616481 -0.741761
## 573   0.277255  0.207371 -0.0401515  0.4738878  0.0050763 -0.215231
## 574  -0.111932  0.243520  0.1814735 -0.2492561  0.3109985 -0.039246
## 575   0.518559 -0.647956  0.0496736 -0.0563876 -1.1169473 -0.632956
## 578   0.116136  0.556207  0.1571408 -0.1364750 -0.0131747  0.570038
## 583   0.056247  0.037849 -0.1490327  0.0239531  0.3497863  0.076281
## 584   0.117476  0.054014 -0.4623133  0.2901078  0.3835580  0.089543
## 586   0.068751 -0.151832 -0.0360416 -0.1595958  0.2627285  0.040391
## 589  -0.195230  0.259354  0.3810502 -0.1492455  0.2353846 -0.223392
## 595  -0.173169 -0.311231 -0.5102040 -0.4645467  0.4229834  0.109777
## 596   0.290892 -0.164570 -0.1181125  0.4092625  0.2775371 -0.077928
## 599  -0.154804  0.186862  0.2096437  0.6378846 -0.0372788 -0.062262
## 600  -0.057899  0.123171 -0.3947594 -0.2092677  0.0921373 -0.282435
## 603   0.017641  0.249933 -0.3527381  0.2574175  0.3193548  0.170654
## 604  -0.096916 -0.184048  0.3357872 -0.4841372  0.1302138  0.269303
## 610   0.079021 -0.090434 -0.4945496 -0.0477167  0.1939362  0.339140
## 613  -0.668217 -0.654262  0.2381824  0.0239291 -0.6072387  1.208025
## 618   0.028274 -0.148305 -0.2468741  0.0990199  0.3678482  0.270441
## 621   0.018588  0.117269  0.3240035 -0.1480606  0.5150507  0.101307
## 622   0.214627  0.191893  0.5116684 -0.1963902  0.1010389  0.188806
## 623  -0.192072  0.148586 -0.0547278 -0.1034001  0.5284897 -0.185710
## 624  -0.189547  0.088266 -0.1234835  0.4303201  0.5112492 -0.192115
## 625  -0.023233 -0.448892 -0.2447506  0.4203372  0.2799352  0.072113
## 628   0.045438 -0.086157  0.3368160 -0.6086148 -0.1483062 -0.032995
## 630   0.115066  0.305379  0.0692269  0.1417268  0.0682124  0.270359
## 632   0.358824  0.146778  0.1173973 -0.1535729 -0.2891288  0.101383
## 633   0.138595 -0.467499  0.4389547 -0.3387704 -0.2503449 -0.326142
## 635   0.418816 -0.084192  0.1231732  0.4133219 -0.1694466 -0.109793
## 644   0.070270  0.221017 -0.0287275 -0.2854282  0.2635322  0.228160
## 645  -0.199460 -0.514764  0.2905973 -0.0051203 -0.2221429  0.331994
## 649   0.069956  0.512524  0.1072356  0.7293223  0.1222159  0.326860
## 650  -0.081940 -0.168851  0.1700165 -0.1249850  0.4842440 -0.085257
## 651  -0.124786  0.076890  0.6260083  0.0338820  0.2168697  0.075936
## 652  -0.252291 -0.193379  0.6842186 -0.1204314  0.4844768  0.012869
## 653  -0.013528 -0.007896  0.1366934 -0.4525960  0.1092652  0.149792
## 654  -0.075329 -0.359967 -0.1614698 -0.3135093 -0.0070273  0.192299
## 655   0.002830 -0.014210  0.3306001 -0.3715350  0.1650694  0.327101
## 656   0.507712  0.124758 -0.4737870 -0.1511679 -0.4068069  0.101715
## 659  -0.840539 -0.126582 -0.8073413  0.1726395 -0.3205271  0.087942
## 663   0.230109 -0.114256 -0.2641941 -0.0110969 -0.2490503  0.235515
## 665   0.142705 -0.140345 -0.4216895  0.1772274  0.3291639  0.092759
## 668  -0.605674  1.343843 -0.0045003  0.4070900 -0.8988541 -0.650461
## 673   0.064762 -0.194427  0.0406780 -0.2227884  0.0250137  0.261405
```

```
## 674   0.371507 -0.553387 -0.0673359  0.1571154 -0.5419178 -0.397168
## 678  -0.009698  0.096821 -0.2701114  0.5268443  0.6647995 -0.037449
## 680   0.093989 -0.171804  0.3902060 -0.8342993 -0.0323521 -0.042244
## 687  -0.125395 -0.133804  0.0571732  0.2446806  0.4008024  0.241349
## 692   0.060276 -0.197425 -0.0270222  0.1002662  0.0351601  0.146963
## 694  -0.067563  0.010009 -0.4891605 -0.2948559  0.5539208  0.321321
## 698   0.225689  0.244550  0.1088664 -0.0385639 -0.0261930  0.164626
## 699   0.101762 -0.012468  0.2397378 -0.0700242  0.4333643 -0.026236
## 700  -0.122253  0.045498 -0.0917100 -0.3201605 -0.3476797  0.389609
## 701   0.106923 -0.096388 -0.5038293  0.1575595  0.4433544  0.267535
## 703  -0.156883 -0.586400 -0.1449614 -0.3142571 -0.0332883  0.134343
## 705   0.162957  0.000152  0.2469173  0.0336999  0.0118562 -0.093105
## 706   0.307404 -0.659237  0.3556193  0.3095782 -0.2221737 -0.647720
## 709  -0.341270 -0.452114 -0.3301868  0.0621277 -0.0961485  0.686741
## 710   0.381786  0.035200  0.4573254 -1.0141535 -0.5293221 -0.107651
## 712  -0.188740  0.416207  0.1805736 -0.1075906 -0.1768800  0.416256
## 713   0.204012  0.099626 -0.4136265 -0.0182300  0.2877522  0.091895
## 715   0.242777 -0.359417  0.0157276 -0.3986054 -0.2355053 -0.331474
## 716   0.388628 -0.236235  0.0726913 -0.1641091 -0.5963065 -0.307017
## 718  -0.566048  0.036493 -0.4199774 -0.2308734 -0.0130491 -0.226889
## 720   0.251260  0.229371 -0.1934597 -0.4201281  0.2353486  0.377071
## 721  -0.340254  0.618043  0.2145580  0.1680256 -0.5701689  0.151813
## 722  -0.077212 -0.309387 -0.4595779 -0.2576573  0.2140620  0.104762
## 723   0.657145 -0.276315 -0.2429784 -0.2002798 -0.8863738 -0.568975
## 725   0.200336 -0.128085  0.5983965  0.0708032 -0.4516396  0.109418
## 729   0.052219  0.230365  0.2730557 -0.0112148  0.0473017  0.091988
## 733  -0.157005  0.342671 -0.5081825  0.2605934  0.2836745 -0.597629
## 735   0.109772  0.251297 -0.3117726  0.2357042 -0.0444141 -0.122968
## 738   0.208274 -0.008989 -0.2803178 -0.1302161  0.1510264 -0.092943
## 739   0.392912 -0.016349  0.2989903 -0.4340546 -0.3964222 -0.100127
## 740   0.237252  0.232742 -0.1816034 -0.0889493 -0.2937133  0.252507
## 741  -0.043018  0.101610  0.4599287 -0.7134366  0.2469049  0.217324
## 748  -0.004918  0.873856 -0.0675877  0.0788480 -0.4687263  0.554111
## 750  -0.047844 -0.322454 -0.0247865  0.4700368  0.0847431 -0.576006
## 751  -0.012110  0.662801  0.1167331  0.4586900 -0.0248542  0.269684
## 752   0.209934 -0.083080 -0.9232851 -0.3153992  0.2505139  0.064992
## 753   0.350237 -0.084355  0.1135429 -0.4912000 -0.2583778 -0.274677
## 754   0.108569 -0.109718  0.3704983  0.3975632  0.1752617  0.050706
## 757  -0.931960 -0.510704 -0.4425847 -0.0668478  0.2226587  0.164386
## 760   0.271758 -0.196110 -0.3895494 -0.1798857  0.0310068 -0.149702
## 761  -0.042397  0.182751  0.0741692 -0.4455274  0.4690628  0.297770
## 764   0.258819  0.028630 -0.8671055  0.0569156  0.2762725  0.263085
## 766   0.216605  0.097104 -0.1260043 -0.2180139  0.0571342  0.415415
## 771  -0.361058  0.879858 -0.3261870  0.5238453 -0.2784098 -0.066883
## 774  -0.110625 -0.127785 -0.3911493 -0.2652602  0.2976375  0.110582
## 784   0.423133 -0.097816  0.0992889  0.0228228 -0.2699175 -0.249853
## 786   0.135700 -0.040781 -0.2675020 -0.2848511  0.0972272  0.165871
## 789   0.002114 -0.282374 -0.0486175  0.6026207 -0.1144382  0.426644
## 794   0.134709 -0.242813 -0.2805990  0.1582501 -0.0597067  0.058233
## 795   0.495306 -0.114169  0.1836943 -0.2895366 -0.4687337 -0.288855
## 798  -0.018743 -0.364107  0.7545423  1.0472496  0.9677736 -0.211516
## 801   0.359158 -0.081395 -0.2691777  0.0273466 -0.1772961 -0.250044
## 803   0.401232  0.063834 -0.0371369 -0.2523872 -0.2875405 -0.136862
## 805   0.198769  0.010293 -0.1676933  0.3769336  0.2526066  0.071602
```

```
## 808   0.032792  0.355550 -0.0257213 -0.1098536  0.2438741  0.388223
## 812   0.212981  0.083275 -0.4837980  0.3999754  0.3278047  0.148752
## 815   0.218465  0.437683 -0.0745495  0.0097541 -0.0538215  0.482923
## 816   0.625612 -0.145928 -0.4161757 -0.2629286 -0.7206939 -0.365921
## 817   0.320030  0.001827 -0.0536511  0.2770482 -0.0901246 -0.044856
## 821   0.134963 -0.204079 -0.1779352  0.1320300  0.1166286 -0.078137
## 822  -0.193743  0.345936  0.3344764 -0.4066664  0.2293371 -0.034319
## 824   0.027161  0.425662 -0.1223562  0.0658523 -0.1772568 -0.290891
## 826  -0.071595  0.064707  0.0100115 -0.1776485  0.0803916 -0.202748
## 827   0.089348 -0.329546  0.2627190 -0.0129229  0.1093330 -0.242818
## 828   0.041261  0.053784  0.3859907 -0.5990243  0.3594214  0.175110
## 829  -0.017586 -0.063976 -0.2155727 -0.1375178  0.4552425  0.076601
## 831   0.228721  0.010426 -0.5164803 -0.1113881 -0.1660666  0.072682
## 838  -0.248556  0.855670 -0.0460543  0.1030179 -0.3600192 -0.512081
## 840  -0.037542  0.287428 -0.2912465 -0.0020434  0.2309537 -0.140911
## 844   0.385796 -0.256460 -0.6289974 -0.1442322 -0.3284538 -0.251082
## 850   0.268781  0.134924 -0.0759881  0.2265155  0.0516672  0.097056
## 853  -0.016593 -0.339864  0.1523842 -0.3450713 -0.2877273  0.391048
## 854   0.324947 -0.468314 -0.0162470 -0.4463228 -0.3230632 -0.440347
## 858   0.313843 -0.104445 -0.4699022  0.0197945 -0.1931629  0.190925
## 859   0.258297 -0.169766 -0.1322718 -0.1447635  0.1042523 -0.183204
## 860  -0.333714  0.180938  0.5304855 -0.0614186 -0.3879179  0.213970
## 869   0.240077  0.215604 -0.4035046  0.1370394 -0.5834659  0.094452
## 873   0.239670  0.134506  0.3708037 -0.3744079 -0.1700197  0.095160
## 876   0.202305  0.047727 -0.1184751 -0.3026277  0.2760794  0.082104
## 880   0.121876 -0.721858  0.4492101  0.9950133 -0.2412974 -0.268397
## 882  -0.028490 -0.195004  0.1370133 -0.0604913  0.3553999 -0.013350
## 883   0.132852 -0.886051  0.4723670  0.6418817 -0.0947067 -0.463446
## 885  -0.054974 -0.213157  0.5494637 -0.0557064  0.3767402  0.101174
## 888   0.407772 -0.653342  0.0200698 -0.4657480 -0.7605030 -0.647299
## 889   0.273207  0.276207 -0.0570999 -0.4042460 -0.4096486 -0.167571
## 892   0.173093  0.053886  0.5212633  0.5260954 -0.0733994  0.226450
## 894   0.085202  0.484022  0.2694467 -0.1568758 -0.1917277  0.108597
## 903  -0.062341  0.013230  0.1843324  0.2967375 -0.0181023  0.404846
## 905  -0.093358 -0.754138 -0.0263655  0.4139347 -0.4300735  0.287227
## 906   0.089923  0.381115 -0.3825047 -0.0883571 -0.0475423 -0.155211
## 916  -0.204299  1.030244 -0.5032511  0.9144195 -0.7630474 -0.239266
## 918   0.048050  0.125010 -0.3823620  0.0288694  0.3584921  0.058178
## 926  -0.031909  0.002222  0.4079069 -0.2674017  0.4962657 -0.001233
## 927   0.053355 -0.045360 -0.2697182 -0.1242279 -0.1358973  0.590075
## 930   0.328034  0.158231 -0.0627402 -0.4299394 -0.1177453  0.098631
## 936   0.397677 -0.483166 -0.1717855 -0.1749962 -0.7246195 -0.228823
## 937   0.077065 -0.003180 -0.2856698 -0.0769416  0.0998130  0.210982
## 947   0.251802  0.035853  0.0618208 -0.0130242  0.2222171  0.087760
## 955   0.354705  0.167164 -0.1666173 -0.0108032 -0.1848347  0.137935
## 956  -0.285079 -0.070001 -0.2981857 -0.0023400  0.1564021 -0.394183
## 958   0.183963  0.087475  0.2569070  0.1601955  0.0662636  0.087380
## 959   0.386320  0.025882 -0.2772798 -0.1346737 -0.2145402  0.072121
## 960   0.234657  0.332182 -0.1312325  0.2209763 -0.1301282  0.095794
## 964   0.188726  0.086100  0.0381634  0.3603180  0.4981376  0.209449
## 971   0.262545 -0.377726 -0.1885892 -0.1382267  0.1766874 -0.336122
## 972   0.173544  0.040470  0.0209437  0.0682502 -0.0632301  0.239613
## 974   0.344209  0.012815 -0.1300887  0.3263053 -0.4557190  0.216643
## 975   0.197271  0.018608 -0.5218686 -0.2363070  0.2010019  0.075356
```

```
## 977   0.117289  0.275273 -0.0518200 -0.1695885  0.0322529 -0.149498
## 984   0.319428  0.126368 -0.0734056 -0.0562144  0.1256723  0.180716
## 986   0.294927 -0.039774  0.5304142 -0.4336091 -0.3304997 -0.131918
## 989   0.283449  0.073419  0.0369474 -0.2048789 -0.1098474  0.202363
## 993  -0.145572 -0.014014 -0.6964654 -0.2104003  0.4819763 -0.242452
## 996   0.218733  0.057925 -0.3076096 -0.1946443  0.0264794 -0.129182
## 1002 -0.132894  0.902997  0.1797333  0.3289163 -0.3064498  0.412795
## 1004 -0.074834  0.493759  0.4176516  0.4860215 -0.2091478  0.232737
## 1007 -0.085029 -0.090216 -0.0299896  0.3286059 -0.0017840  0.094358
## 1008  0.337828  0.254576 -0.1696274 -0.0216959 -0.2888074  0.240860
## 1013  0.178525 -0.509294 -0.5815319 -0.0502340  0.1424502 -0.298011
## 1017  0.100242  0.529430  0.0338586 -0.2794453 -0.0905211  0.354091
## 1024  0.203163 -0.182466  0.0158562  1.0247161  0.5175692 -0.190095
## 1025  0.112458 -0.237367  0.2091607  0.4604418  0.3995686 -0.048721
## 1027  0.369279 -0.344978  0.0464527  0.6796172  0.2493591 -0.493815
## 1031  0.504723 -0.233663  0.0685526  0.1637301 -0.6420216 -0.446888
## 1037  0.195325 -0.544395 -0.3130195  0.4700494  0.0333247 -0.324925
## 1038  0.267127 -0.634355 -0.0278299 -0.4969359 -0.2940148 -0.611481
## 1050  0.441882  0.030932 -0.0768687 -0.0836643 -0.2695992 -0.024519
## 1058 -0.199321  0.129752 -0.1891915 -0.0889591  0.3731769  0.061311
## 1060  0.221139  0.068856  0.2767367 -0.2515286 -0.0959780 -0.093650
## 1065 -0.103862  0.845067 -0.1658648  0.2191942 -0.4886361 -0.002940
## 1066  0.652761 -0.490799 -0.1493343  0.1067846 -0.9283627 -0.727134
## 1072  0.397106 -0.238666 -0.2751160 -0.1337632 -0.5078146 -0.111642
## 1073  0.568002 -0.629698 -0.1950032 -0.1753235 -1.2212959 -0.594350
## 1075  0.134270 -0.271323 -0.3604452 -0.3136616  0.2835591 -0.128999
## 1080 -0.108232  0.973758 -0.1039287  0.5823014 -0.4809760  0.284349
## 1081  0.033348 -0.057446  0.0173393 -0.1671544  0.4148849 -0.249412
## 1083 -0.050237  0.241940 -0.4324914  0.0082169  0.0250168 -0.264392
## 1086  0.472406 -0.010288 -0.0678707 -0.1079818 -0.2851725 -0.145639
## 1090  0.582784 -0.156372 -0.2240158 -0.1891865 -0.7601782 -0.310815
## 1092 -0.354761 -0.659156  0.0383286  0.5827720 -0.2885261  0.655779
## 1095  0.170437  0.045959 -0.0197938  0.7379766  0.4344077 -0.011235
## 1097  0.053027 -0.224309 -0.5188881 -0.0102382 -0.2802963  0.601222
## 1100  0.263376 -0.264627 -0.0698611 -0.0838405 -0.3619183 -0.050752
## 1101  0.253171  0.009659  0.0144641 -0.0530475  0.1664315  0.054258
## 1113  0.181916  0.052897 -0.2683707 -0.1668641 -0.2885070  0.411474
## 1119  0.343891 -0.035934  0.2717892 -0.2969829 -0.1530497 -0.178285
## 1124  0.085124  0.025263 -0.2433957 -0.0697835  0.3470949  0.124317
## 1125  0.067985  0.689002 -0.3753574  0.6560506 -0.1727607  0.181037
## 1126  0.221832 -0.202584  0.0219326  0.1377358 -0.4373442  0.213418
## 1132  0.125185 -0.306790  0.2440023  0.7592204  0.3757360  0.037405
## 1133  0.347033  0.293832 -0.2682609 -0.0403226 -0.2092794  0.302232
## 1135 -0.039865 -0.392257  0.7338461  0.0882952 -0.1917543  0.292483
## 1138  0.412550 -0.082717 -0.0937972  0.2754335  0.0056975 -0.200875
## 1139  0.107488 -0.062099  0.2667273  0.4318263  0.5181074 -0.070124
## 1140  0.271188  0.058516  0.2820619  0.1899526 -0.0437335  0.140822
## 1142  0.323367  0.195871 -0.2036505  0.2488266 -0.1725773  0.211832
## 1151  0.164616 -0.097584 -0.1363134 -0.0311377 -0.0328495  0.307183
## 1153 -0.308837  0.521965 -0.7884033  0.1502198 -0.0486336 -0.353175
## 1157  0.206796 -0.174659 -0.1963433 -0.4122880 -0.4712746 -0.135525
## 1160  0.237840  0.080105 -0.4317261 -0.2878088  0.0488951 -0.081232
## 1163  0.196912  0.174199 -0.2687572  0.2424245 -0.1541539 -0.211450
## 1172  0.159947 -0.088296  0.0750434 -0.0313323  0.4543524 -0.066193
```

```
## 1174 -0.273266  1.151636  0.3672069  0.1870259 -0.5244883  0.413042
## 1177 -0.147241  0.651534 -0.4351628  0.4283612 -0.4335580  0.505239
## 1185  0.326307 -0.357176  0.0887768  0.3388332  0.0048480 -0.404540
## 1191  0.235833  0.256870  0.2058809 -0.2212796  0.1096234  0.365855
## 1195 -0.148716  0.081136 -0.3976430 -0.0381472  0.1876285  0.036066
## 1200  0.215231  0.021291  0.1060407 -0.1460021  0.0604411  0.030228
## 1201  0.370678 -0.804452  0.4886843  1.1161335 -0.0559731 -0.999600
## 1210  0.527798 -0.194392  0.1276827  0.2510905 -0.4677677 -0.356883
## 1221  0.406208 -0.004670  0.1346615 -0.2936313 -0.3918977 -0.074603
## 1237 -0.014835 -0.108333  0.0061541  0.5388048  0.7022336 -0.054146
## 1242 -0.056038 -0.086952  0.1016772 -0.4538496 -0.0771279 -0.090119
## 1250  0.038722 -0.071504 -0.5302943 -0.2720277 -0.0350450 -0.057389
## 1253 -0.269072  1.095648 -0.1135655  0.5896591 -0.5509467  0.162766
## 1258  0.427325 -0.056971  0.0188878  0.3284610 -0.1254414 -0.161227
## 1261  0.258847 -0.094269 -0.3828262 -0.2707530 -0.1510292  0.240163
## 1263  0.413454 -0.168711 -0.3512119 -0.3612400 -0.3070408 -0.285356
## 1267  0.273910  0.052055  0.1481257  0.0452364  0.0066622  0.153509
## 1277  0.160388 -0.018844 -0.1972292 -0.3128862  0.3277906  0.082228
## 1282 -0.097425 -0.033181  0.0275849 -0.6252182  0.6750514  0.342550
## 1290 -0.171401  0.230771  0.6606947 -0.2041007  0.2254402  0.243118
## 1291  0.318556  0.238036 -0.4846787  0.0724503 -0.1900655  0.315892
## 1297  0.191276  0.107493 -0.1255037  0.2829491  0.2977076  0.140908
## 1320  0.387691 -0.024673  0.3020552 -0.1624651 -0.2648196 -0.095788
## 1321  0.278705  0.375249  0.1418137 -0.4021530  0.0100308  0.514701
## 1323  0.151933 -0.305867  0.4712137 -0.3041607 -0.1396993 -0.179298
## 1327  0.150314  0.017109 -0.0073559 -0.6242649  0.0789920  0.185903
## 1333  0.248025  0.103898  0.1714203  0.2811174  0.2170261  0.168261
## 1336  0.233286 -0.030390  0.0634639 -0.3494105  0.0999594 -0.005007
## 1341  0.321593  0.083881 -0.0957878 -0.0741423 -0.0705628 -0.011028
## 1345  0.330744  0.106822  0.0995756  0.0899876  0.0151703  0.124188
## 1346  0.178031 -0.218707 -0.0727281  0.0623587  0.0558164  0.047075
## 1349 -0.056116 -0.270876  0.1030307  0.4341528  0.2982095  0.400488
## 1351  0.133266  0.590391 -0.1013453  0.6039459  0.0095202  0.530938
## 1368  0.533530 -0.040307 -0.4030389 -0.0510616 -0.4748981 -0.153584
## 1374  0.326371  0.102159 -0.3430876  0.2620977  0.0267558  0.112377
## 1388  0.189577  0.127098  0.4252693 -0.4445795 -0.2623655 -0.368481
## 1391  0.103793  0.285554 -0.4173001  0.5011854  0.3137590  0.362726
## 1399  0.187343  0.197910 -0.3840073  0.3745040  0.3353551  0.219632
## 1400 -0.008618  0.726338  0.0766309  0.1905963 -0.2041547  0.324021
## 1403  0.001131  0.230475  0.3900587  0.3369735 -0.0924821  0.529996
## 1405  0.153882 -0.241916 -0.4711169  0.1289677  0.3189164 -0.055254
## 1430  0.222621  0.049135  0.4745605  1.3607211  0.5508080 -0.053468
## 1439  0.177441 -0.141323 -0.2062718  0.3030658 -0.2362158  0.475123
## 1442  0.328662  0.037442 -0.2533845  0.1374455  0.0888534  0.067028
## 1452 -0.036203  0.821784  0.0733350  0.3694927 -0.2196272  0.375293
## 1460  0.024733 -0.225792  0.3474818 -0.2215974 -0.1778019  0.331934
## 1463  0.135853  0.047833 -0.2731219 -0.1361644  0.5075137  0.233748
## 1465  0.295435  0.081876  0.0021696  0.0176570 -0.0008402  0.018514
## 1477  0.022033 -0.113231  0.3929166  0.6040843  0.4547813  0.098287
## 1483  0.314281  0.232686 -0.1413207  0.2831979 -0.1619100 -0.031900
## 1484  0.303877  0.174289 -0.1539880  0.1694931  0.2266302  0.303299
## 1492 -0.124619 -0.811745  0.6302154  1.8738879  0.1401793  0.215325
## 1509  0.121301  0.314114  0.2503283  0.5158615 -0.1489202 -0.083157
## 1512  0.055268 -0.199066 -0.3028027  0.0149378  0.3842484  0.129331
```

```
## 1517  0.235306 -0.102651  0.2639689  0.7493187  0.3995941 -0.070440
## 1526 -0.091587 -0.048272 -0.3737797  0.7416844  0.3386943  0.135084
## 1527 -0.023122 -0.042908  0.0500254  0.3341076 -0.1568359  0.055919
## 1550  0.342127 -0.262490 -0.2307287 -0.0644177 -0.1951438 -0.241050
## 1559  0.169618 -0.004667 -0.3065826  0.0477335  0.3039289 -0.073526
## 1560  0.591339 -0.152174 -0.3421550  0.2702209 -0.6211644 -0.278921
## 1561  0.238625 -0.335211 -0.6904499 -0.4040643 -0.0618973 -0.105885
## 1563  0.070351  0.033431 -0.1276323 -0.0630756  0.4050475  0.076724
## 1566  0.184172  0.344064  0.0934709 -0.4154836  0.2491796  0.494473
## 1567 -0.206438  0.347816  0.0547747  0.3402431  0.3127401 -0.167906
## 1580  0.260817  0.206930  0.2197078  0.1492718  0.3354167  0.264636
## 1582  0.115115  0.037578  0.6901827  1.1369644  0.4792249 -0.131447
## 1584 -0.027744  0.537839 -0.2559288 -0.0792212  0.0771066 -0.032367
## 1589 -0.116507  0.434100 -0.1413245  0.1401604  0.1818210  0.113280
## 1590  0.314717 -0.328640  0.0766986  0.0348828  0.0486057 -0.495748
## 1626  0.584889 -0.987424 -0.2864327  0.4158009 -1.8248613 -0.511529
## 1631  0.265740 -0.242655  0.3548753  0.4586933  0.3817673 -0.331564
## 1634  0.164309 -0.077706 -0.2784484  0.2180475  0.5458750  0.086297
## 1635  0.428775  0.058030 -0.3419144 -0.0324846 -0.1195416  0.011812
## 1642  0.316119  0.101426 -0.0227025 -0.1293633 -0.1669766 -0.061705
## 1644  0.082370 -0.031448 -0.2183675 -0.1410790  0.3587964 -0.072904
## 1678 -0.061665  0.017234 -0.4385687  0.0595310  0.4346522 -0.220220
## 1686  0.178489 -0.117537 -0.3710620 -0.2072689  0.1310655  0.161446
## 1689  0.224708 -0.082242 -0.3383272  0.0741607  0.1734928 -0.356758
## 1695  0.118120 -0.329412 -0.1768393  0.3413229 -0.3669427  0.520562
## 1701  0.304711  0.054470 -0.3199559  0.0987536  0.3038167  0.079079
## 1715 -0.016685 -0.328391 -0.0691289  0.4207723 -0.1572148  0.430752
## 1719  0.268741  0.046081 -0.0896636  0.4281481  0.2038202 -0.002785
## 1723  0.168746 -0.195549 -0.6247592 -0.1268928 -0.2844551  0.322347
## 1731 -0.045554 -0.218569  0.5501773  0.2104752  0.1528613  0.143810
## 1742  0.537519 -0.211206 -0.2019647 -0.5427848 -0.6411757 -0.465637
## 1743 -0.006262 -0.160824 -0.6151883 -0.0297231  0.4979049 -0.107954
## 1744  0.430628 -0.772974  0.8192686  1.4980485 -0.2069549 -0.797738
## 1771  0.241263 -0.032358 -0.1248781  0.7742890  0.6704739  0.105403
## 1795  0.282507 -0.191454 -0.6522299 -0.3648758  0.0589746 -0.271203
## 1806  0.673192 -0.391059  0.0369017  0.2417621 -0.9693591 -0.630542
## 1816 -0.034622 -0.310890 -0.4126827  0.1720166  0.1276977  0.285830
## 1850 -0.002386 -0.159225 -0.3769887  0.2023701  0.6605621 -0.040425
## 1851  0.253135 -0.240239  0.0175928 -0.0137849 -0.0060275 -0.106223
## 1873 -0.002183 -0.311959 -0.1287469  0.2550169  0.7359671  0.017183
## 1912  0.335152 -0.265894  0.1094084  0.3740995  0.0686945 -0.276908
## 1920  0.207371  0.273999 -0.4383453 -0.0030416 -0.1029510  0.560214
## 1930  0.438819 -0.710696  0.2270500  0.4449927 -0.4765051 -0.735176
## 1939  0.194618  0.294670  0.1188666  0.1771708 -0.2370548 -0.140727
## 1944  0.131210  0.002979 -0.0611916  1.0837068  0.5021458  0.058172
## 1950  0.450092  0.045948 -0.3311729  0.0314547 -0.3761800 -0.033126
## 1987  0.301907 -0.416762 -0.6204083  0.0125928 -0.1978603 -0.145930
## 2003  0.213183 -0.330453 -0.1405689  0.1846711  0.2631821 -0.401765
## 2004  0.074337 -0.079947 -0.3083672 -0.0388565  0.2540008  0.143081
## 2025  0.094494  0.154465 -0.3919079  0.3818852 -0.1269029  0.020269
## 2026  0.176698  0.185867 -0.1278858  0.5283546 -0.1279741  0.497542
## 2028  0.186176  0.429913  0.0165668 -0.2420384 -0.4234028  0.026476
## 2029  0.189786  0.080747 -0.2635422 -0.0674373  0.1555353  0.375494
## 2036  0.472846 -0.009581 -0.0617627 -0.0443027 -0.2679431 -0.097265
```

```
## 2047  0.209174 -0.091308  0.7508845  1.0060689  0.1791394 -0.314547
## 2050  0.336678  0.131928 -0.0741001  0.3236639 -0.1438450  0.120979
## 2115 -0.081070  0.241686  0.0041968  0.2737385 -0.0836435  0.244177
## 2121  0.253937  0.300725 -0.6868898  0.0931995  0.0820685  0.343215
## 2130  0.166887 -0.377443 -0.0210183 -0.0412034 -0.0336546 -0.120613
## 2132  0.309121  0.017247 -0.2270231 -0.0827585 -0.3702673 -0.061505
## 2160  0.309858 -0.432259 -0.6324059 -0.0804530 -0.1048272 -0.368639
## 2191  0.112180 -0.464602 -0.2721324  0.2851285  0.4492244 -0.308511
## 2205  0.339084 -0.316167  0.0755174  0.6932079  0.1223451 -0.394661
## 2206  0.220773  0.348668  0.0250854 -0.3349507 -0.0591109  0.248986
## 2238  0.469831  0.097916 -0.2967277 -0.0636977 -0.2880767  0.103143
## 2258  0.458796 -0.291379 -0.2860198 -0.1936819 -0.2752511 -0.372606
## 2264  0.183621  0.431661 -0.8427348 -0.0198100  0.1132793  0.317269
## 2275  0.360467  0.280409 -0.4016211  0.0969332  0.0678394  0.390669
## 2281  0.228882 -0.110858 -0.1742814  0.0006315  0.1886018  0.071074
## 2293  0.561261 -0.188962 -0.3718455  0.1375824 -0.8472414 -0.131375
## 2321  0.089043  0.071618  0.1367936  0.6608918 -0.0138808  0.376815
## 2339  0.236659 -0.327747  0.3236836  1.4539645  0.4472179 -0.367070
## 2465  0.190316  0.021019 -0.3213221  0.2280504 -0.3339231  0.525773
## 2764  0.140662  0.538937 -0.3102525  0.5035527 -0.4152540  0.272708
```

```
plot(all.pca)
```