Weak trait matching is all you need

Michael Roswell

1/5/2022

Preamble

I just saw a trait matching dataset where ~50% of species anti-matched... that is they were outside the inner 95% quantile for difference in trait values. I am wondering what numerical features of the data govern the frequency of these occurrences, and how much to interpret the biology of this. The goal of the code below is to simulate some datasets that would vaguely match the original and then, maybe, play with the simulation to adjust the frequency of anti-matchers.

Simulate data roughly like ours

Trait matching is obviously not only about species associations, but also the basic distribution of traits in each class. Before we think about species associations, let's simulate some trait values that are, roughly speaking, in line with our data.

```
# assume abundances are negative binomially distributed

# first, match them pretty well.
bee_rich <-75
plant_rich <- 100
bee_abund<-rnbinom(n = bee_rich
              , size = 1
              , mu = 6)



# we had a bimodal-ish bee tongue distribution
bee_var<- 1.5

bee_traits = c(rnorm(floor(2*bee_rich/3), mean = 1, sd = bee_var)
               , rnorm(ceiling(bee_rich/1), mean = 8, sd = bee_var))

# and some kind of long-tailed plant trait distribution
plant_var <-1
plant_traits = rlnorm(plant_rich, meanlog = log(3), sdlog = plant_var)
```

Species associations

The next thing is thinking about how species are, or are not, associated, based on traits. Probably it's helpful to think about a totally neutral community first, just to set some guidelines. In this simulation,

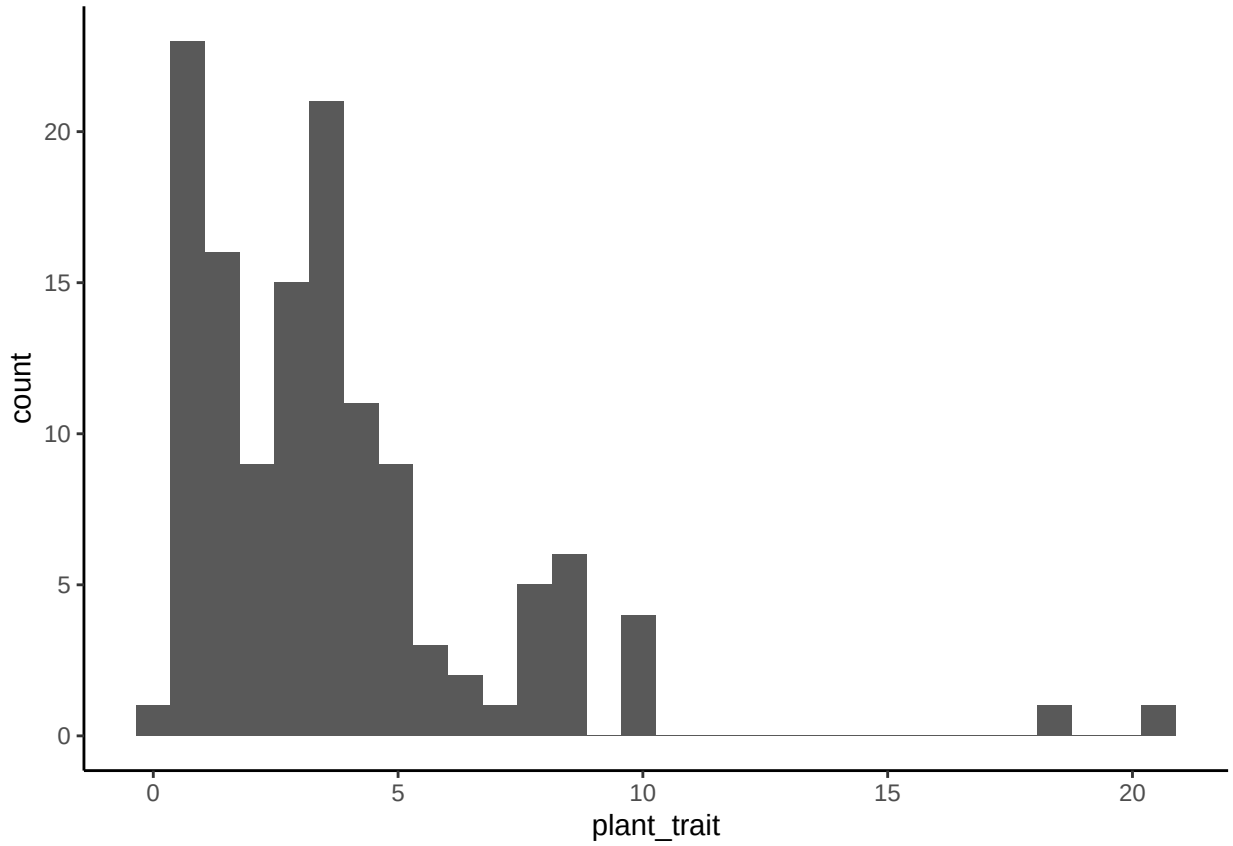
there is no biological association between bees and the plants they visit.

```
neutral<-map_dfr(1:bee_rich, function(bee_sp){
  if(bee_abund[bee_sp]>0){
   plant_sp = sample(1:plant_rich, bee_abund[bee_sp], replace =T)
   plant_trait = plant_traits[plant_sp]
  }
  else{plant_sp = 0
  plant_trait = NA}
   return(data.frame(bee_sp
                     , plant_sp
                     , bee_trait = bee_traits[bee_sp]
                     , plant_trait))
})
```

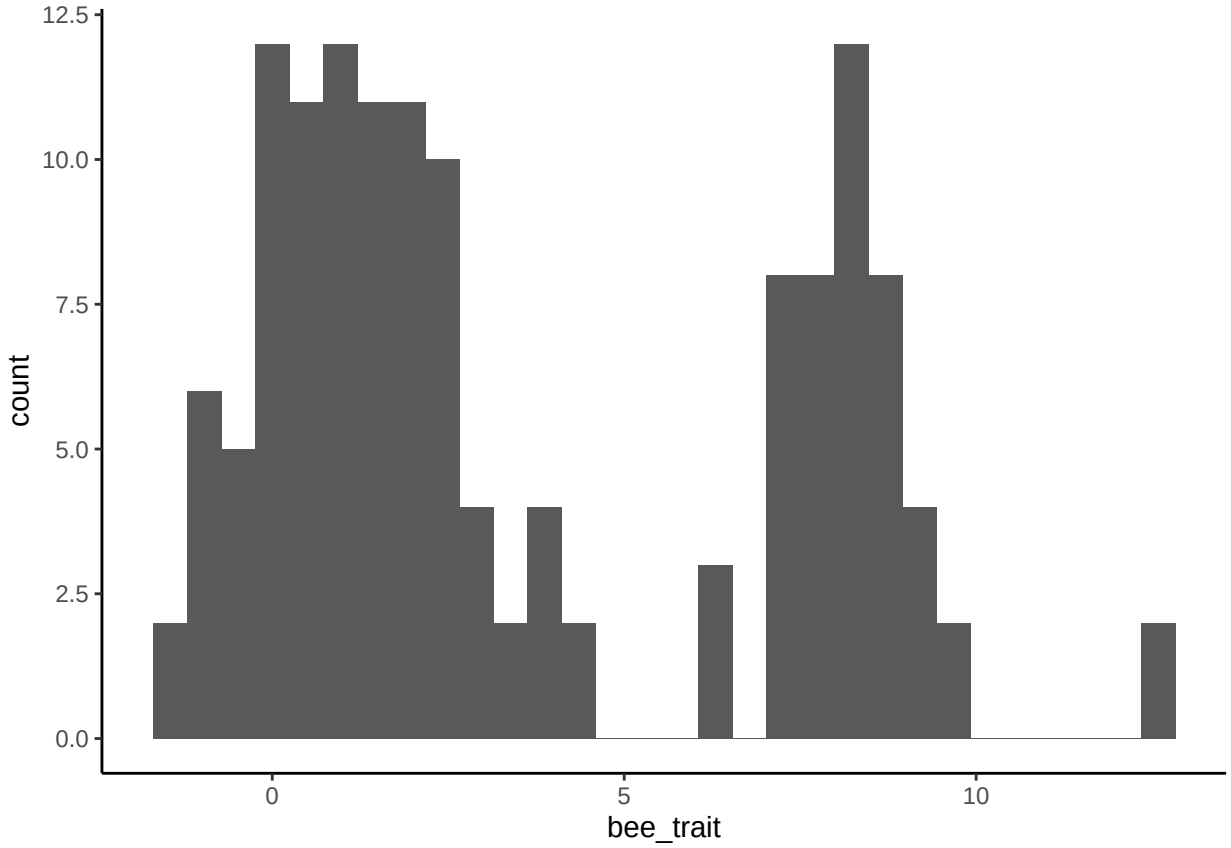
We can also think about a more realistic community. Here, bee species have associations with specific plant species. This is "biological" but it's not based on the trait values we're looking at. Rather, it's something we can't predict.

```
specfactor <- 5 # this controls the strenght of specialization
random_specialization<-map_dfr(1:bee_rich, function(bee_sp){
  N = bee_abund[bee_sp]
  if(N > 0){
    p<-list()
    p[1] = sample(1:plant_rich, 1)
    for(i in 2:N){ # this just makes it more likely that a plant species
      # chosen by one individual will be chosen by other individuals of
      # the same bee sp.
      i<-2
      p[i] = sample(1:plant_rich
                    , 1
                    , prob = c(rep(1, unlist(p[i-1])-1) #not chosen plants
                               , 5 # upweight selected plant
                               , rep(1, plant_rich - unlist(p[i-1]))))
      # not chosen plants
    }
   plant_sp = unlist(p)
   plant_trait = plant_traits[plant_sp]
  }
  else{plant_sp = 0
  plant_trait = NA}
   return(data.frame(bee_sp
                     , plant_sp
                     , bee_trait = bee_traits[bee_sp]
                     , plant_trait))
})

# quickly visualize the distribution of traits:
random_specialization %>% ggplot(aes(plant_trait)) +
  geom_histogram() +
  theme_classic()
```




```
random_specialization %>% ggplot(aes(bee_trait)) +
  geom_histogram() +
  theme_classic()
```




```
# plant abundances are a result of the bee choices
plant_abund<-random_specialization %>%
  group_by(plant_sp) %>%
  summarize(plant_abund = n())

# so not all plants are visited, don't forget them
explant<-c(1:plant_rich)[!1:plant_rich %in% random_specialization$plant_sp]

plant_abund<-plant_abund %>%
  bind_rows(data.frame(plant_sp = explant, plant_abund = 0)) %>%
  arrange(plant_sp) %>%
  filter(plant_sp>0)
```

That simulation seems to capture some biological realism (bees and plants don't actually interact *randomly* with respect to species ID, even if the interactions aren't governed by the traits we measure). Now, let's generate something like the "null model" we've been using from this simulation, which breaks species associations.

It's probably possible to fix abundances better using the null models from the R package `bipartite` or something. The code below will mean plant abundance will not be exactly the same but I think it gets the point across anyways.I'm also taking a short cut and just assuming that doing it 99 times in one network (rather than doing a new network each time) will give the same mean values.

```
break_specialization<-map_dfr(1:99, function(iter){
    map_dfr(1:bee_rich, function(bee_sp){
    if(bee_abund[bee_sp]>0){
     plant_sp = sample(1:plant_rich, bee_abund[bee_sp], replace =T
```

```
                        , prob = plant_abund$plant_abund)
     plant_trait = plant_traits[plant_sp]
    }
    else{plant_sp = 0
    plant_trait = NA}
     return(data.frame(bee_sp
                        , plant_sp
                        , bee_trait = bee_traits[bee_sp]
                        , plant_trait))
  })
})
```

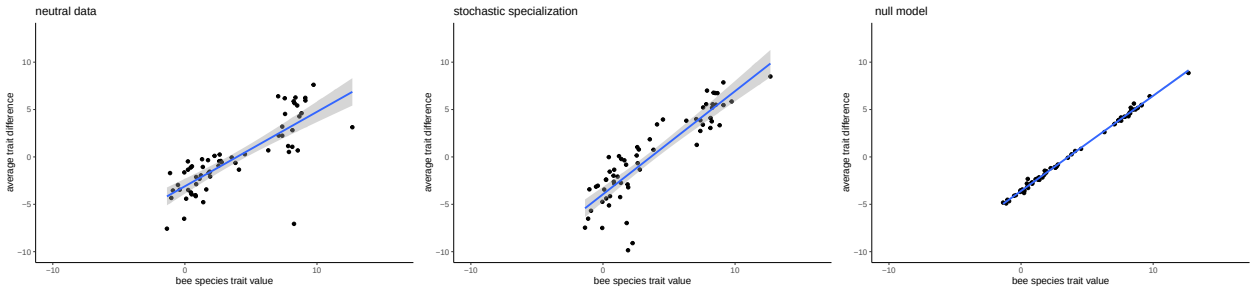
My first fear not realized

So in the totally neutral data you see what you probably expected... bees with "short tongues" (i.e. low trait values) tended to have lower trait values than the plants they visited had, the ones with high trait values had higher trait values than the plants they visited. Because of the trait distributions we started with, once we included specialization, this pattern fuzzier– but not weaker. That is becuase, for example, some bee species with low trait values just happened to specialize on plants that also had unusually low trait values, Even as others did just the opposite.

I was somehow worried that making this fuzzier would also flatten the line.. in other words that it would match our data. But no, to get the line flatter you need biological specialization that is actually based on trait values.. like we think we have in our data.

```
neut<-plot_matching(neutral) + ggtitle("neutral data")
spec<-plot_matching(random_specialization) + ggtitle("stochastic specialization")
broke<-plot_matching(break_specialization)  + ggtitle(" null model")

neut
spec
broke
```



An experiment

What if we make the associations based on trait values? Let's do weak trait matching across the board.

```
weak_matching <- map_dfr(1:bee_rich, function(bee_sp){
  N = bee_abund[bee_sp]
  if(N > 0){
```

```
   plant_sp = sample(x=1:plant_rich, size = N, replace =T
                     # this is where the trait matching occurs. the rnorm adds
                     # noise to the trait matching so it's not soo strong. Also,
                     # rnorm will be different for each bee spp, so it's not
                     # just assigning new trait values, it's adding other
                     # dimensions each time
                , prob = 1/(rnorm(plant_rich
                                  , mean = bee_traits[bee_sp]-plant_traits)^2))
   plant_trait = plant_traits[plant_sp]
  }
  else{plant_sp = 0
  plant_trait = NA}
   return(data.frame(bee_sp
                     , plant_sp
                     , bee_trait = bee_traits[bee_sp]
                     , plant_trait))
})

# make a fresh null model since plant frequencies likely changed

plant_abund<-weak_matching %>%
  group_by(plant_sp) %>%
  summarize(plant_abund = n())

# so not all plants are visited, don't forget them
explant<-c(1:plant_rich)[!1:plant_rich %in% weak_matching$plant_sp]

plant_abund<-plant_abund %>%
  bind_rows(data.frame(plant_sp = explant, plant_abund = 0)) %>%
  arrange(plant_sp) %>%
  filter(plant_sp>0)

break_match<-map_dfr(1:99, function(iter){
    map_dfr(1:bee_rich, function(bee_sp){
    if(bee_abund[bee_sp]>0){
     plant_sp = sample(1:plant_rich, bee_abund[bee_sp], replace =T
                       , prob = plant_abund$plant_abund)
     plant_trait = plant_traits[plant_sp]
    }
    else{plant_sp = 0
    plant_trait = NA}
     return(data.frame(bee_sp
                       , plant_sp
                       , bee_trait = bee_traits[bee_sp]
                       , plant_trait))
  })
})
```

Now we should see the observed relationship flatter than the null one. This would lead to bees with low-trait values to have a significantly-non-null high mean trait difference (i.e. closer to 0 than expected) and the high-trait ones to have a significantly-non-null low mean trait difference (i.e. still closer to 0 than expected). But it might not say something about an overall strategy (that is, there was not different rules for short vs. long tongued bees, just weak trait matching overall).

```
spec<-plot_matching(weak_matching)
broke<-plot_matching(break_match)

spec + ggtitle("with matching")+
  broke + ggtitle("null model")
```