Feasability trait sampling in LiMMBo

Hannah Meyer

18/07/2018

1. Set-up

```
#### libraries ##
library(ggplot2)
```

```
#### functions ##
subsetSampling <- function(P, s, minCooccurrence, verbose=FALSE) {
    if (verbose) cat("Sampling for", P, "traits\n")
    counter <- 1
    counts <- matrix(0, nrow=P, ncol=P)

    while (min(counts) < minCooccurrence) {
        newindex <- sample(P, s, replace=FALSE)
        counts[newindex, newindex] <-  counts[newindex, newindex] + 1
        counter <- counter + 1
    }
    return(list(counter=counter, counts=counts))
}

ttcooccurence <- function(nrtraits, nrtraitssampled) {
    1/nrtraits * 1/(nrtraits-1) * nrtraitssampled * (nrtraitssampled-1)
}
```

2. Simulate sampling as implemented in LiMMBo 0.1.4

```
## parameters
set.seed(10)

# minimum number of co-occurence of trait pair
minCooccurrence <- 3
# number of traits
P <- seq(50,500,50)
# sub-sampling size
s <- c(10, 15)
```

```
simulatedSampling <- lapply(1:10, function(dummy) {
    tmp_P <- lapply(P, function(traits) {
        tmp_s <- lapply(s, function(subsample, traits) {
            subsetSampling(P=traits, s=subsample,
                           minCooccurrence=minCooccurrence,
                           verbose=FALSE)
        }, traits=traits)
        counter <- sapply(tmp_s, function(x) x$counter)
        names(counter) <- paste("Subsample", s, sep="")
        counts <- lapply(tmp_s, function(x) x$counts)
        names(counts) <- paste("Subsample", s, sep="")
        return(list(counter=counter, counts=counts))
    })

    counter <- sapply(tmp_P, function(x) x$counter)
    colnames(counter) <- paste("Traits", P, sep="")
    counter <- reshape2::melt(counter)
    counts <- lapply(tmp_P, function(x) x$counts)
    names(counts) <- paste("Traits", P, sep="")
    return(list(counter=counter, counts=counts))
})
names(simulatedSampling) <- paste("rep", 1:10, sep="")
```

3. Reformat simulation results and plot

```
counter <- do.call(rbind, lapply(simulatedSampling, function(x) x$counter))
colnames(counter) <- c("Subsample", "Traits", "Bootstraps")
traitCosampling_empirical <- reshape2::acast(counter, Subsample ~ Traits, mean)
```

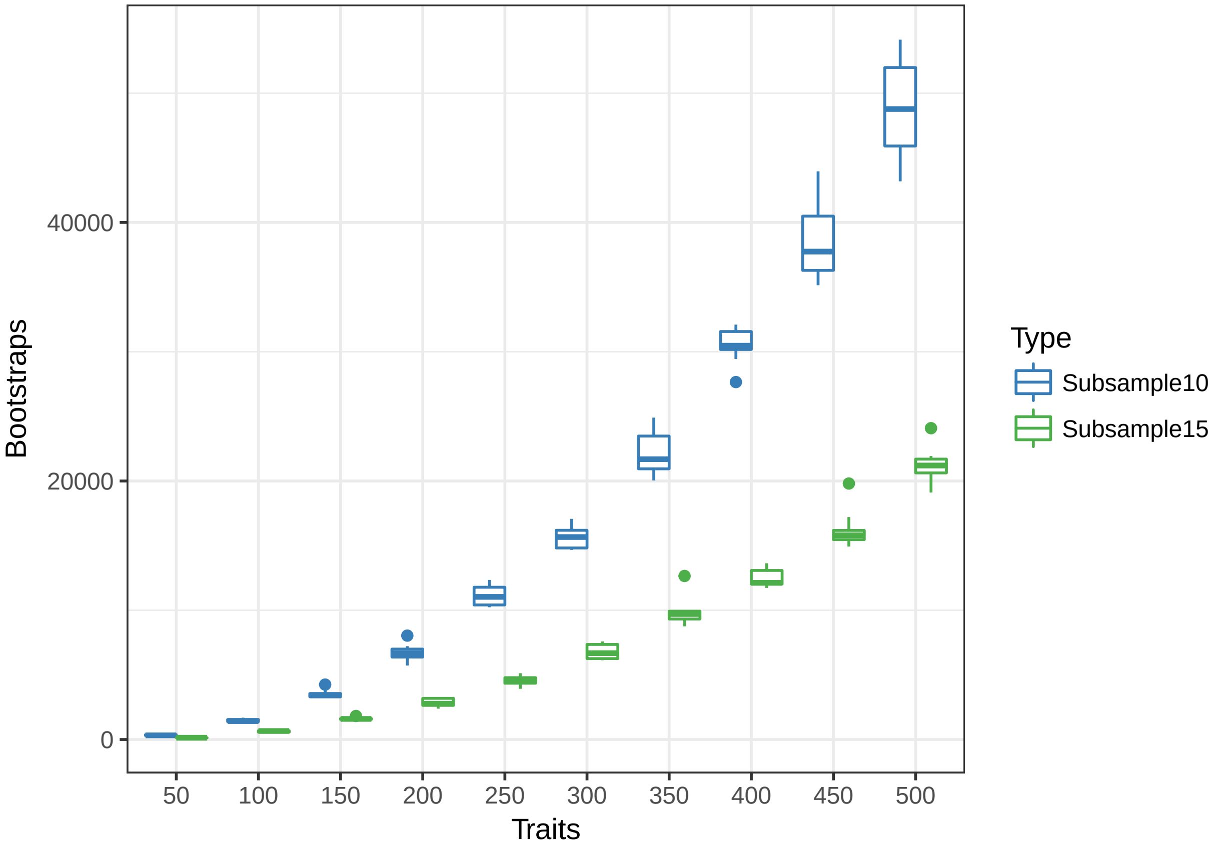
```
## Using Bootstraps as value column: use value.var to override.
```

```
traitCosampling_empirical_sd <- reshape2::acast(counter, Subsample ~ Traits, sd)
```

```
## Using Bootstraps as value column: use value.var to override.
```

```
counter$Traits <- as.factor(as.numeric(gsub("Traits", "", counter$Traits)))


p_counter <- ggplot()
p_counter + geom_boxplot(data=counter,
                                      aes(x=Traits, y=Bootstraps,
                                          color=as.factor(Subsample))) +
    scale_color_manual(values=c('#377eb8','#4daf4a'), name="Type") +
    theme_bw()
```


Bootstraps
40000
20000
0
50
100
150
200
250
300
350
400
450
500
Traits
Type
Subsample10
Subsample15