Filtering datasets

Renata Diaz

2021-05-09

We filter datasets in two stages. First, we filter **prior to trying to sample from the feasible set**, in order to remove communities that have a combined S and N too high for us to sample the feasible set, and to reduce the Forest Inventory and Analysis dataset to a manageable size. Second,we filter **after sampling the feasible but before we aggregate results across communities**, to remove cases where mathematical constraints result in uninformative results. Below we elaborate on the logic and methods for these filtering processes, including code to allow others to see precisely what was implemented.

## Pre-feasible set sampling

While our algorithm for sampling the feasible set improved our ability to assess the shape of the feasible set for larger communities, there were still computational limits on what we could do. We filtered out all communities with more than 40720 individuals, because this is the largest community we were able to sample given the available resources. This upper limit results in the removal of a total of 4 communities, all of them from the Miscellaneous Abundance Database.

For computational reasons, we also created a sub-sample of the FIA dataset, because sampling the feasible set for all 103,343 FIA communities overwhelms our computational pipeline. Because the FIA has so many extremely small communities (92,988 with fewer than 10 species), we decided to randomly select a sample of the small communities for analysis. Thus our FIA dataset consisted of ~20 ,000 communities comprising all communities with more than 10 species (10,355) plus 10,000 randomly selected communities with 3-10 species.

Code for downloading the data (and all other analyses) used for this project can be found at www.github.com/diazrenata/scadsanalysis. The data download functions are at <https://github.com/diazrenata/scadsanalysis/blob/master/R/download_data.R>.

### Overview

The download\_data function downloads raw data files from <https://github.com/weecology/sad-comparison/> (for BBS, Gentry, Mammal Community Database, and FIA; data from Baldridge (2016) and also used in White, Thibault, and Xiao (2012)) and figshare <http://figshare.com/files/3097079> (for the Miscellaneous Abundance Database; Baldridge (2015)). These raw files are stored in working-data\abund\_data and are not edited.

The datasets downloaded from GitHub exactly match the corresponding data files in the Zenodo archive for Baldridge (2016), available here: (<https://zenodo.org/record/166725>). If in the future it is preferred or necessary to use the archival versions, they can be downloaded and stored in working-data\abund\_data and the rest of the analysis should run just as if they had been downloaded from GitHub.

The Miscellaneous Abundance and FIA databases undergo additional filtering steps, implemented in the filter\_misc\_abund and filter\_fia\_short and filter\_fia\_small functions. These functions run automatically when the data are downloaded and save new files to working-data\abund\_data.

The load\_dataset function loads either the filtered datasets (if specified, for Misc. Abund and FIA) or the raw data (for all others, or if specified for Misc. Abund and FIA), and adds additional columns and column types to prepare the data for the computational pipeline. The datasets as returned from load\_dataset are exactly what goes into the analysis.

In the following sections, we load the raw data files (as they were downloaded), perform the filtering process manually (if applicable), and compare the resulting data to the data that are used in the analysis, i.e. the data that we get if we use load\_dataset.

### BBS (no filtering)

For BBS, and all datasets that are not filtered, we can simply load the data from the file downloaded by download\_data, add the columns and column formats that are added by load\_data, and compare the result to the result of load\_data:

bbs\_raw <- read.csv(here::here("working-data", "abund\_data", "bbs\_spab.csv"), stringsAsFactors = F, header = F, skip = 2)  
  
colnames(bbs\_raw) <- c("site", "year", "species", "abund")  
  
bbs\_raw <- bbs\_raw %>%  
 mutate(site = as.character(site),  
 dat = "bbs",  
 singletons = F,  
 sim = -99,  
 source = "observed") %>%  
 group\_by(site) %>%  
 arrange(abund) %>%  
 mutate(rank = row\_number()) %>%  
 ungroup()  
  
bbs\_loaded <- load\_dataset("bbs")

Here we confirm that all values for abund and site match between the version we get from load\_dataset and the version we just created from the raw data file:

any(!(bbs\_loaded$abund == bbs\_raw$abund))

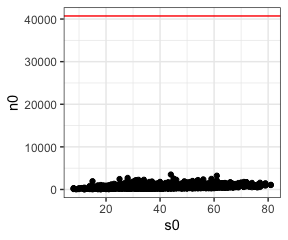
## [1] FALSE

any(!(bbs\_loaded$site == bbs\_raw$site))

## [1] FALSE

Here we check to confirm that no communities have a total abundance exceeding our upper limit (40720, marked by the horizontal red line):

bbs\_statevars <- get\_statevars(bbs\_raw)  
  
ggplot(bbs\_statevars, aes(s0, n0)) +  
 geom\_point() +  
 theme\_bw() +  
 geom\_hline(yintercept = 40720, color = "red")



No communities in BBS have more than 40720 individuals, so all are initially included in the analysis.

### Gentry (no filtering)

For Gentry, as with BBS, we can simply load the data from the file downloaded by download\_data, add the columns and column formats that are added by load\_data, and compare the result to the result of load\_data:

gentry\_raw <- read.csv(here::here("working-data", "abund\_data", "gentry\_spab.csv"), stringsAsFactors = F, header = F, skip = 2)  
  
colnames(gentry\_raw) <- c("site", "year", "species", "abund")  
  
gentry\_raw <- gentry\_raw %>%  
 mutate(site = as.character(site),  
 dat = "gentry",  
 singletons = F,  
 sim = -99,  
 source = "observed") %>%  
 group\_by(site) %>%  
 arrange(abund) %>%  
 mutate(rank = row\_number()) %>%  
 ungroup()  
  
gentry\_loaded <- load\_dataset("gentry")

Here we confirm that all values for abund and site match between the version we get from load\_dataset and the version we just created from the raw data file:

any(!(gentry\_loaded$abund == gentry\_raw$abund))

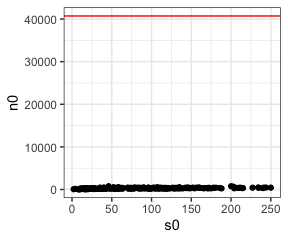
## [1] FALSE

any(!(gentry\_loaded$site == gentry\_raw$site))

## [1] FALSE

Here we check to confirm that no communities have a total abundance exceeding our upper limit (40720, marked by the horizontal red line):

gentry\_statevars <- get\_statevars(gentry\_raw)  
  
ggplot(gentry\_statevars, aes(s0, n0)) +  
 geom\_point() +  
 theme\_bw() +  
 geom\_hline(yintercept = 40720, color = "red")



No communities in Gentry have more than 40720 individuals, so all are initially included in the analysis.

### Mammal Community Database (not filtered)

For MCDB, as with BBS and Gentry, we can simply load the data from the file downloaded by download\_data, add the columns and column formats that are added by load\_data, and compare the result to the result of load\_data:

mcdb\_raw <- read.csv(here::here("working-data", "abund\_data", "mcdb\_spab.csv"), stringsAsFactors = F, header = F, skip = 2)  
  
colnames(mcdb\_raw) <- c("site", "year", "species", "abund")  
  
mcdb\_raw <- mcdb\_raw %>%  
 mutate(site = as.character(site),  
 dat = "mcdb",  
 singletons = F,  
 sim = -99,  
 source = "observed") %>%  
 group\_by(site) %>%  
 arrange(abund) %>%  
 mutate(rank = row\_number()) %>%  
 ungroup()  
  
mcdb\_loaded <- load\_dataset("mcdb")

Here we confirm that all values for abund and site match between the version we get from load\_dataset and the version we just created from the raw data file:

any(!(mcdb\_loaded$abund == mcdb\_raw$abund))

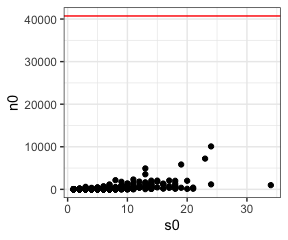
## [1] FALSE

any(!(mcdb\_loaded$site == mcdb\_raw$site))

## [1] FALSE

Here we check to confirm that no communities have a total abundance exceeding our upper limit (40720, marked by the horizontal red line):

mcdb\_statevars <- get\_statevars(mcdb\_raw)  
  
ggplot(mcdb\_statevars, aes(s0, n0)) +  
 geom\_point() +  
 theme\_bw() +  
 geom\_hline(yintercept = 40720, color = "red")



No communities in the MCDB have more than 40720 individuals, so all are initially included in the analysis.

### Miscellaneous Abundance Database (filtered)

For the Miscellaneous Abundance Database, we can proceed similarly to with the non-filtered datasets. We load the data from the file downloaded by download\_data, add the columns and column formats that are added by load\_data, and compare the result to the result of load\_data.

However, Misc. Abund includes datasets reported as relative abundance in addition to count data. Relative abundance data is not appropriate for this analysis, so we filter out records where abund = 0 to restrict our analysis to communities with count data. Also, because we will eventually filter out some highly abundant communities, at this stage we make sure to load the **unfiltered** version using load\_dataset.

misc\_abund\_raw <- read.csv(here::here("working-data", "abund\_data", "misc\_abund\_spab.csv"))  
  
misc\_abund\_raw <- misc\_abund\_raw %>%  
 dplyr::rename(site = Site\_ID,  
 abund = Abundance)  
  
misc\_abund\_raw <- misc\_abund\_raw %>%  
 mutate(site = as.character(site),  
 dat = "misc\_abund",  
 singletons = F,  
 sim = -99,  
 source = "observed") %>%  
 filter(abund > 0) %>%  
 group\_by(site) %>%  
 arrange(abund) %>%  
 mutate(rank = row\_number()) %>%  
 ungroup()  
  
misc\_abund\_loaded <- load\_dataset("misc\_abund")

Here we confirm that, if we load the unfiltered dataset using load\_dataset, the records match the version we have created manually from the raw file:

any(!(misc\_abund\_loaded$abund == misc\_abund\_raw$abund))

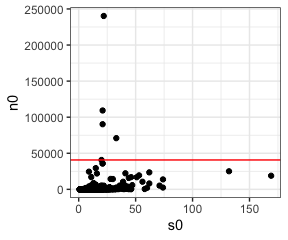
## [1] FALSE

any(!(misc\_abund\_loaded$site == misc\_abund\_raw$site))

## [1] FALSE

Here we check whether any communities have more than 40720 individuals (horizontal red line):

misc\_abund\_statevars <- get\_statevars(misc\_abund\_raw)  
  
ggplot(misc\_abund\_statevars, aes(s0, n0)) +  
 geom\_point() +  
 theme\_bw() +  
 geom\_hline(yintercept = 40720, color = "red")



Misc abund has 4 communities with more than 40720 individuals, and these communities are removed:

misc\_abund\_sv\_filtered <- misc\_abund\_statevars %>%  
 filter(n0 <= 40720)  
  
misc\_abund\_filtered <- filter(misc\_abund\_raw, site %in% misc\_abund\_sv\_filtered$site)

We can access the **filtered** version of the dataset, as used in the analysis, using load\_dataset("misc\_abund\_short"). We then check to confirm that the filtered version of the dataset that we get from load\_dataset matches the version that we filtered manually:

misc\_abund\_short\_loaded <- load\_dataset("misc\_abund\_short")  
  
any(!(misc\_abund\_short\_loaded$abund == misc\_abund\_filtered$abund))

## [1] FALSE

any(!(misc\_abund\_short\_loaded$site == misc\_abund\_filtered$site))

## [1] FALSE

### FIA (filtered)

Again, we can begin similarly to with the non-filtered datasets. We load the data from the file downloaded by download\_data, add the columns and column formats that are added by load\_data, and compare the result to the result of load\_data. Because we will remove some communities from FIA, we make sure to load the **unfiltered** version using load\_data.

fia\_raw <- read.csv(here::here("working-data", "abund\_data", "fia\_spab.csv"), stringsAsFactors = F, header = F, skip = 2)  
  
colnames(fia\_raw) <- c("site", "year", "species", "abund")  
  
fia\_raw <- fia\_raw %>%  
 mutate(site = as.character(site),  
 dat = "fia",  
 singletons = F,  
 sim = -99,  
 source = "observed") %>%  
 filter(abund > 0) %>%  
 group\_by(site) %>%  
 arrange(abund) %>%  
 mutate(rank = row\_number()) %>%  
 ungroup()  
  
fia\_loaded <- load\_dataset("fia")

Here we confirm that, if we load the unfiltered dataset using load\_dataset, the records match the version we have created manually from the raw file:

any(!(fia\_loaded$abund == fia\_raw$abund))

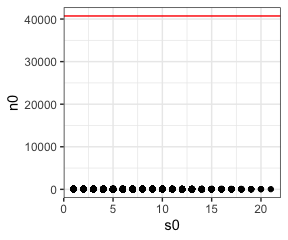
## [1] FALSE

any(!(fia\_loaded$site == fia\_raw$site))

## [1] FALSE

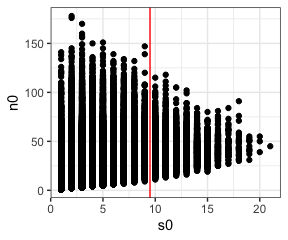
Here we check whether any communities have more than 40720 individuals (horizontal red line):

fia\_statevars <- get\_statevars(fia\_raw)  
ggplot(fia\_statevars, aes(s0, n0)) +  
 geom\_point() +  
 theme\_bw() +  
 geom\_hline(yintercept = 40720, color = "red")



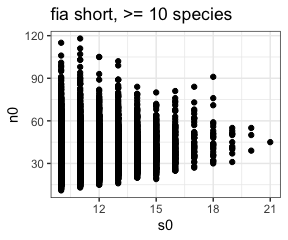
FIA has no communities with more than 40720 individuals. However, it presents additional issues. It it has 103343 communities, of which 92988 have fewer than 10 species (vertical red line in the figure below)

ggplot(fia\_statevars, aes(s0, n0)) +  
 geom\_point() +  
 theme\_bw() +  
 geom\_vline(xintercept = c(9.5), color = "red")

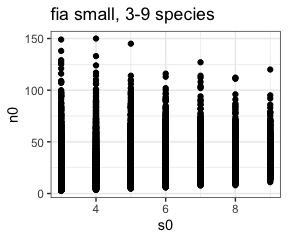


This many communities overwhelms our computational pipeline. We therefore sample all 10355 communities with 10 or more species, and a random subsample of 10,000 communities with 3-9 species. We then run these through the pipeline as two separate databases. fia\_short is the communities with 10 or more species, and fia\_small is the 10,000 communities with 3-9 species. We re-combine them as “FIA” for aggregate analyses. This results in a toal of 20,355 FIA communities in the analysis.

fia\_sv\_short <- fia\_statevars %>%  
 dplyr::filter(s0 >= 10)  
  
fia\_short <- fia\_raw %>%  
 dplyr::filter(site %in% fia\_sv\_short$site) %>%  
 dplyr::mutate(dat = "fia\_short")  
  
fia\_short\_statevars <- get\_statevars(fia\_short)  
  
ggplot(fia\_short\_statevars, aes(s0, n0)) +  
 geom\_point() +  
 ggtitle("fia short, >= 10 species") +  
 theme\_bw()



fia\_sv\_small <- fia\_statevars %>%  
 dplyr::filter(s0 >= 3) %>%  
 dplyr::filter(s0 <= 9)  
  
 set.seed(1977)  
 fia\_sv\_small <- fia\_sv\_small[ sample.int(nrow(fia\_sv\_small), size = 10000, replace = F), ]  
  
  
  
fia\_small <- fia\_raw %>%  
 dplyr::filter(site %in% fia\_sv\_small$site) %>%  
 dplyr::mutate(dat = "fia\_small")  
  
fia\_small\_statevars <- get\_statevars(fia\_small)  
  
ggplot(fia\_small\_statevars, aes(s0, n0)) +  
 geom\_point() +  
 ggtitle("fia small, 3-9 species")



The “short” and “small” datasets are saved as .csvs and can be loaded using load\_dataset as follows. These are the versions that are used in the analysis.

fia\_small\_loaded <- load\_dataset("fia\_small")  
  
fia\_short\_loaded <- load\_dataset("fia\_short")

Here we confirm that versions of “fia\_short” and “fia\_small” that we created manually match the ones we obtain from load\_dataset:

any(!(fia\_small\_loaded$abund == fia\_small$abund))

## [1] FALSE

any(!(fia\_small\_loaded$site == fia\_small$site))

## [1] FALSE

any(!(fia\_short\_loaded$abund == fia\_short$abund))

## [1] FALSE

any(!(fia\_short\_loaded$site == fia\_short$site))

## [1] FALSE

## Post-sampling

Some communities fall into mathematically special spaces where basic constraints prevent informative results. We removed all communities that fell into any of the below cases:

1. When communities have only 1 mathematically possible SAD (S = 1, N = S, or N = S + 1).
2. When there are fewer than 20 (for the dissimilarity metric) or 40 (for all other metrics) possible SADs. In these cases, it is not possible to fall in the 95th or 97.5/2.5th percentiles.
3. When a community consists of only 2 species, we excluded it from analyses using skewness, because e1071::skewness() always evaluates to 0 if S = 2.

Below is the procedure for performing this filtering (after sampling the feasible set). First, we load our results from all\_di.csv. This file contains the results of comparing the skewness and evenness for **observed** SADs to the distributions of skewness and evenness obtained from their sampled feasible sets; “all\_di” stands for “all diversity indices”. Every community that was included for sampling is included in all\_di. (Additionally, every community has a “singletons” counterpart, which is the same analysis run adjusted for rarefaction. That analysis is discussed elsewhere, and we ignore the rarefaction-adjusted versions here).

all\_statevars <- bind\_rows(bbs\_statevars, fia\_short\_statevars, fia\_small\_statevars, gentry\_statevars, mcdb\_statevars, misc\_abund\_sv\_filtered)  
  
  
all\_di <- read.csv(here::here("analysis", "reports", "submission2", "all\_di.csv"), stringsAsFactors = F)

all\_di <- all\_di %>%  
 filter(!singletons) %>%  
 mutate(dat = ifelse(grepl(dat, pattern = "fia"), "fia", dat),  
 dat = ifelse(dat == "misc\_abund\_short", "misc\_abund", dat)) %>%  
 mutate(Dataset = dat,  
 Dataset = ifelse(Dataset == "fia", "Forest Inventory and Analysis", Dataset),  
 Dataset = ifelse(Dataset == "bbs", "Breeding Bird Survey", Dataset),  
 Dataset = ifelse(Dataset == "mcdb", "Mammal Community DB", Dataset),  
 Dataset = ifelse(Dataset == "gentry", "Gentry", Dataset),  
 Dataset = ifelse(Dataset == "misc\_abund", "Miscellaneous Abundance DB", Dataset))  
  
#head(all\_di)

Here we confirm that we have the same number of communities in all\_di as we have in our set of communities included for sampling.

nrow(all\_di) == nrow(all\_statevars)

## [1] TRUE

Here we remove communities for the first case, only one possible SAD (N = S, N = S + 1, or S = 1).

all\_di %>%  
 group\_by\_all() %>%  
 mutate(only\_one\_sad = sum(s0 == n0, s0 == 1, n0 == (s0 + 1)) > 0) %>%  
 ungroup() %>%  
 group\_by(dat) %>%  
 summarize(total\_only\_one\_sad = sum(only\_one\_sad),  
 total\_sites = dplyr::n()) %>%  
 ungroup() %>%  
 mutate(all\_sites\_one\_sad = sum(total\_only\_one\_sad),  
 all\_sites = sum(total\_sites))

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| dat | total\_only\_one\_sad | total\_sites | all\_sites\_one\_sad | all\_sites |
| bbs | 0 | 2773 | 258 | 24647 |
| fia | 176 | 20355 | 258 | 24647 |
| gentry | 0 | 224 | 258 | 24647 |
| mcdb | 56 | 730 | 258 | 24647 |
| misc\_abund | 26 | 565 | 258 | 24647 |

all\_di\_filtered <- all\_di %>%  
 filter(s0 != n0,  
 s0 != 1,  
 n0 != (s0 + 1))  
  
nrow(all\_di\_filtered)

## [1] 24389

Removing those with only one SAD results in the removal of 258 sites total. 176 from FIA, 56 from MCDB, and 26 from Misc. Abund.

Finally, we filter for cases 2 and 3. That is, for aggregate analyses, we will restrict our analyses to sites whose feasible sets have more than 20 unique values for the dissimilarity metric, and 40 for all others. For analyses with skewness, we will also remove sites with only 2 species. This results in these final total numbers of communities included for dissimilarity, skewness, and all other metrics:

For dissimilarity:

all\_di\_filtered %>%  
 filter(nparts > 20) %>%  
 group\_by(dat) %>%  
 summarize(sites\_for\_diss = dplyr::n()) %>%  
 mutate(total\_sites\_for\_diss = sum(sites\_for\_diss))

|  |  |  |
| --- | --- | --- |
| dat | sites\_for\_diss | total\_sites\_for\_diss |
| bbs | 2773 | 22490 |
| fia | 18447 | 22490 |
| gentry | 224 | 22490 |
| mcdb | 552 | 22490 |
| misc\_abund | 494 | 22490 |

For skewness:

all\_di\_filtered %>%  
 filter(nparts > 40, s0 > 2) %>%  
 group\_by(dat) %>%  
 summarize(sites\_for\_skewness = dplyr::n()) %>%  
 mutate(total\_sites\_for\_skewness = sum(sites\_for\_skewness))

|  |  |  |
| --- | --- | --- |
| dat | sites\_for\_skewness | total\_sites\_for\_skewness |
| bbs | 2773 | 21395 |
| fia | 17410 | 21395 |
| gentry | 223 | 21395 |
| mcdb | 505 | 21395 |
| misc\_abund | 484 | 21395 |

For all other metrics:

all\_di\_filtered %>%  
 filter(nparts > 40) %>%  
 group\_by(dat) %>%  
 summarize(sites\_for\_others = dplyr::n()) %>%  
 ungroup() %>%  
 mutate(total\_sites\_for\_others = sum(sites\_for\_others))

|  |  |  |
| --- | --- | --- |
| dat | sites\_for\_others | total\_sites\_for\_others |
| bbs | 2773 | 21403 |
| fia | 17410 | 21403 |
| gentry | 223 | 21403 |
| mcdb | 511 | 21403 |
| misc\_abund | 486 | 21403 |

## References

Baldridge, E. (2015). Miscellaneous Abundance Database. figshare. Available at: MiscAbundanceDB\_main. <https://doi.org/10.6084/m9.figshare.95843.v4>

Baldridge, E., Harris, D.J., Xiao, X. & White, E.P. (2016). Data from An extensive comparison of species-abundance distribution models. Zenodo. Available at: <https://zenodo.org/record/166725>. Accessed from <https://github.com/weecology/sad-comparison>.

Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A. & Leisch, F. (2019). e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien.

White, E.P., Thibault, K.M. & Xiao, X. (2012). Characterizing species abundance distributions across taxa and ecosystems using a simple maximum entropy model. Ecology, 93, 1772–1778.