**Formatting datasets for the core-transient project**

In this project, we evaluate the proportion of core and transient species at a given site using data collected across a wide range of taxa, regions of the world, and environmental systems. A challenge that we face is that ecological data are collected at highly variable spatial, temporal, and biological scales and it is necessary to consistently format the data in preparation for analysis. Here we provide instructions associated with formatting datasets. The goal is to format a dataset as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| datasetID | site | species | year | count |
| 01 | d01\_Treatment1PlotA | Acrocephalus\_schoenobaenus | 1928 | 1 |
| 01 | d01\_Treatment1PlotA | Buteo\_buteo | 1928 | 3 |
| 01 | d01\_Treatment1PlotB | Corvus\_corax | 1929 | 7 |
| 01 | d01\_Treatment2PlotA | Cuculus\_canorus | 1929 | 1 |
| 01 | d01\_Treatment2PlotB | Gallinula\_chloropus | 1928 | 2 |

**datasetID:**

DatasetID’s are available from the dataset\_summary\_table. This must be the first column of your formatted dataset. To do so, you simply use the rep command in the base package and tell it to repeat the value for the number of rows of the data frame. *Note: example\_df is the unformatted dataset being prepared for analysis.*

datasetID = rep(01, dim(example\_df)[1])

**site:**

Determining sites can sometimes be challenging. In the example above, the sites were broken down into different treatments and plots. To construct the sites field, we paste the datasetID as well as any site information provided. If the above example were constructed from a data frame with the fields “Treatment” and “Plot”, the site column would be made using the following code:

site = paste(‘d’, datasetID, example\_df$Treatment, example\_df$Plot, sep = ‘’)

***Removing site information:***

Some data sources include information in the site field that are problematic for analysis. For example, some sites include the time that a sample was collected as a part of the site field. Consider a column named “site” where year is included in the field, such that the first entry might be “Treatment1PlotA1928”. You can extract the plot information easily in one of two ways:

1) Substring the plot information by from the first to the last plot character. The following code extracts all characters from the first to the 15th within the site field. This is ONLY to be used is the plot information contains the same number of characters!

site1 = substr(example\_df$site, 1, 15)

*Note: If the field is not a character field, you can convert it on the fly using:*

site1 = substr(as.character(example\_df$site) , 1, 15)

2) Substring the plot information by removing the last characters in a field. This method is valid if the number characters that make up the true site field are not the same across sites but there is an equal number of characters that need to be removed. To do so, the easiest way is to use the str\_sub function in Hadley Wickham’s **stringr** package (though this can be easily accomplished by writing your own function in base).

require(stringr)

x = “hello world”

str\_sub(x, 1, -7)

[1] "hello"

***Separating a field to extract site information:***

It is also often necessary to separate the site field by some common character (such as, in the example below “\_”). This is done using the transform and colsplit functions. Colsplit is located in Hadley Wickham’s package **reshape2**. The output of this function is a multiple field dataset containing the original data (field 1) and a column for each split. In this case, the second column contains the site information, so using “[,2]” returns a vector with just the relevant site information.

require(reshape2)

x = 'Treatment1PlotB\_1927'

site1a = transform(x, site = colsplit(x, pattern = '\\\_', names = c('site','year')))[,2]

***Using latitude and longitude to define sites:***

It is sometimes necessary to define sites using latitude and longitude if this is the only site information provided. While making decisions for the appropriate scale to analyze the data requires an understanding of the taxa and method of collection, the process of creating the site information is relatively straitforward. Here we will use the “round\_any” function in Hadley Wickham’s **plyr** package to turn decimal latitude and longitude data sites composed of 2 degree lat-lon blocks.

x = 13.35679

y = 46.87

site1 = paste(round\_any(x, 2), round\_any(y, 2))