NPSForVeg Cheat Sheet

NPSForVeg Object

An S4 object that holds forest vegetation data from a park. Data is held in “slots” in the object.

These slots are text
- ParkCode – 4 letter park code
- ShortName – park’s short name
- LongName – park’s formal name
- Network – 4 letter network code

The slots are vectors with 2 numbers, e.g. c(12,1). They are the number of subplots that each kind of plant is measured in and how big each subplot is in m^2.
- TPlotSize – trees
- SapPlotSize - saplings
- SeedPlotSize - seedlings
- ShrubPlotSize - shrubs
- ShSeedPlotSize – shrub seedlings
- VPlotSize - vines
- HPlotSize – herbs

These slots are data.frames.
- Plots – Metadata on each plot
- Events – Metadata for each sampling event
- Trees
- Saplings
- Seedlings
- Shrubs
- ShSeedlings
- Vines
- Herbs

These are data.frames with the plant data

- Commons – links Latin names, common names and TSN (taxonomic serial number) Also a good place to put species metadata.

Don’t have all these kinds of data? Just leave some slots empty.

Source Data

Required columns for imported data.frames:

Plots: Unit_Code, Subunit_Code, Plot_Name, Location Status, Event_Count, Latitude, Longitude
Events: Unit_Code, Subunit_Code, Plot_Name, Event_Year, Cycle
Trees/Saplings/Seedlings/Shrubs/ShSeedlings/Vines/Herbs: Plot_Name, Status, Latin_Name, Cycle, Sample_Year, Crown_Description, Equiv_Live_DBH_cm, Height, Percent_Cover, SumLiveBasalArea_cm2, Host_Latin_Name, Condidtion
Commons: NCRN_Common, TSN

Status:	Plant alive, dead etc.
Unit_Code:	4 letter park code
Subunit_Code:	Code for subunit in park
Plot_Name:	Name of plot
Location_Status	Plot actively sampled or retired
Event_Count	Number of times plot has been monitored
Event_Year	Year monitoring event took place
Cycle:	Cycle of monitoring
Sample_Year	Year monitoring event took place
Crown_Description:	Crown class
Equiv_Live_DBH_cm:	Live DBH of stem in cm
Height:	Height of seedlings
Percent_Cover:	Percent cover of herbs
SumLiveBasalArea_cm2 :	Basal area in cm^2
Host_Latin_Name:	Latin name of tree that a vine is growing on
Condition:	Does a tree have a vine in its crown
NCRN_Common:	Common name of plant species
TSN:	Taxonomic Serial Number of plant species

Entering Data

importERMN / importMIDN / importNCRN /import NETN / importSHEN: Imports data for network/park from their specific .csv files.

make: Takes data from one or more existing NPSForVeg objects and makes an new one. Used for combing or splitting parks.

Getter functions

Accessor (“getter”) functions retrieve the data from the slots. They are intermediaries between the user and the object.

getNetwork (object**)**
getNames(object, name.class**)**
name.class = “code”, “short” or “long”
getArea(object, group, type**)** type= “single” (size of one subplot), “count” (number of subplots), “all” (total area sampled per plot).

getPlots(object,…**)**
getEvents(object,…**)**
getCommons(object,…**)**

Gets data.frames from these slots. See help for filtering options

getPlotNames(object, …) returns just the Plot_Names field from the Plots data.frame

getPlantNames(object, names, out.style, in.style **)**
Takes an object,a vector of names, output and input styles, translate names between common, Latin and TSN. Options for in.style / out.style = “common”, “Latin” or “TSN”

getPlants(object, group, status, species, cycles, years, plots, crown, size.min, size.man, BA.min, BA.max, host.tree, in.crown, common, output**)**
Retrieves the plant data from any of the data.frames. Many options for filtering. This function is called by many other functions in the package.

object can be a single NPSForVeg object or a whole list.
group is the type of plant desired. Can be “trees”, “saplings”, ”seedlings”, “shrubs”, “shseedlings”, “vines” or “herbs”

https://github.com/NCRN/NPSForVeg

February 9th 2018

Data Manipulation

SiteXSpec(object, group, values, Total, …**)**

This function creates a Site X Species matrix from the plant data. Data is first retrieved from the object using **getPlants()** , any argument to that function can be used **in SiteXSpec()** .

Site X Species matrices can take one of three forms indicated by the **values** argument. For "count" each element of the matrix is the abundance of a species (columns) found in a plot (rows). For herbs this is the number of subplots the species is found in. For "size" the elements represent either basal area, seedling height or cover depending on the **group** of plants. "presab" creates a presence-absence matrix.

The **Total** argument indicates if the final column of the matrix should be the total for each plot.

ChangeMatrix(object, groups, years1, years2, values, …**)**
This function creates a Site X Species matrix that shows change in the plots from one time period, **years1**, to the next**, years2**. Positive numbers indicate increases over time and negative numbers indicate losses.

This function works by using **SiteXSpec()** to create two Site X Species matrices, one corresponding to **years1** and another corresponding to **years2**. Any argument that is valid for either **SiteXSpec()** or **getPlants()** can be used in **ChangeMatrix()** . Once the two matrices are created, the **years1** matrix is subtracted from the **years2** matrix and the resulting matrix is returned.

Analysis

dens(object, group, values, density,…**)**

Calculates the mean and 95% confidence intervals for measures of abundance of plants at a park level. This function first creates a Site X Species matrix, so any argument to **SiteXSpec()** or **getPlants()** is valid for **dens()** as well.

For count data, a negative binomial distribution is assumed. For size data, including % cover, no distribution is assumed and confidence intervals are based on bootstrap estimates. For presence / absence data, a binomial distribution is used.

density indicates if the values from count data should be reported on a per-hectare basis (TRUE) or as raw counts (FALSE)

IV(object, group, …**)**

Calculates forestry importance values for trees, saplings and seedlings. Importance value of a species is the sum of its relative abundance, relative distribution and relative basal area. For seedlings, height is used in place of basal area in the calculation. The function calls **SiteXSpec()** to make Site X Species matrices to calculate each component of IV, so any argument to **SiteXSpec()** or **getPlants()** is valid for **IV()** as well.

Mapping Data

mapPlants(object, plots, values, maptype, colorgroups, radius, opacity, colortype, colors**)**:

Maps data onto the NPS ParkTiles maps using the leaflet package. Requires a vector of plot names (**plots**) and a vector of data (**values**), as well as an **object** from which to get lat / long data. **maptype** indicates which ParkTiles map to use. Other argument specify the color and size of the markers, see help file for details.

Visualizing

densplot(object, Total, top, densargs, compare, list, labels, …**)**

Makes a graph based on the output of **dens()**. Any valid argument to **dens()**, **SiteXSpec()** or **getPlants()** can be passed using **densargs** or **compare**. Any argument to **xyplot()** is also valid.

densargs is a list of arguments passed to **dens()**:
densargs=list(group="trees", years=2011:2014)

compare is similar, except that it is a list of lists, and each list indicating a different data set to compare to:
compare=list (list(NCRN "trees", years=2010:2013) , list(NCRN, "trees", years=2014:2017))

Total: should total be graphed or just individual species,
top: graph the top most commons species
labels: labels for datasets used with compare

IVplot(object, top, IVargs, parts, compare, labels, colors, … **)**

Makes a graph based on the output of **IV()**. Any valid argument to **IV()**, **SiteXSpec()** or **getPlants()** can be passed using **IVargs** or **compare**. Any argument to **barchart()** is also valid.

IVargs: list of arguments passed to **IV()**:
IVargs=list(group="trees", years=2011:2014)
compare, top, labels: identical in usage to arguments from **densplot()**
parts: should all three parts of the IV be graphed(T) or just the total (F)
colors: colors for graphing, If **parts**=T, then three colors are used, otherwise only one color is needed.