Introduction to the 'vegsoup' package (version 0.2-7)

Roland Kaiser

March 7, 2019

1 Introduction

Almost all data sets in vegetation ecology do not easily fit into a single data frame. The problem is at least three-fold. Additional to the (i) species component, there is often (ii) header or sites data, quite regular also (iii) a table encoding a taxonomic concept. Importantly, explicit spatial information is further wished to be readily available for data sets, this demands another additional component. Although manageable, manipulating such inherently multiple table relations can result in long and difficult to read workflows. The main burden arises by tightly linking several objects in terms of order and content (this is not always straight forward). The vegsoup package aids in this situation and provides a well defined class that is capable of storing such data sets. It is obvious to use S4 classes in this situation and the vegsoup package exploits all its's benefits.

The vegsoup package offers base classes for all the core components outlined above. Spatial classes are imported from package sp. Vegsoup objects do not only have facilities (slots) to store data components but may have additional options specified. For example, if the input data provides more information than presence or absence of a species in a sampling unit, e.g. ordinal abundance classes, a decryption of the original measurement scale is desirable in order to translate coverscale labels to numbers. In this respect the package defines a class Coverscale. Basal analysis parameters, such as the definition of an appropriate dissimilarity measure for the data at hand, are often wished to be held constant during analysis. All Vegsoup* objects carry such an attribute. Regarding the spatial content a coordinate reference system is mandatory for the plethora of spatial methods provided by other packages, therefore, Vegsoup classes contain a slot proj4string.

The best way to introduce the functionalities of the package is a session with example code. We load the library as usual into our R environment.

```
> library(vegsoup)
```

```
> Vegsoup()
```

After installation you can invoke this PDF with

```
> vignette("vegsoup")
```

2 Classes

The package uses S4 classes that relate to each other by direct inheritance. First, the classes, their relation to each and accessor methods for the respective class slots are presented. In the following section generic methods for the classes are discussed.

2.1 Vegsoup

Objects of class `Vegsoup` contain a complete set of information typical for a data set in vegetation ecology (phytosociology). There is a bunch of methods to access relevant information or to manipulate objects in various ways. In this respect, the package methods try to be as comprehensive as possible.

```
> showClass("Vegsoup")
```

```
Class "Vegsoup" [package "vegsoup"]

Slots:

Name:                   species                     sites
Class:                  Species               data.frame

Name:                  taxonomy               coverscale
Class:                 Taxonomy               Coverscale

Name:                 decostand                     dist
Class:                decostand                character

Name:                    layers                    group
Class:                character                  integer

Name:                 sp.points              sp.polygons
Class:   SpatialPointsDataFrame SpatialPolygonsDataFrame

Known Subclasses:
Class "VegsoupPartition", directly
Class "VegsoupOptimstride", directly
Class "VegsoupPartitionFidelity", by class "VegsoupPartition", distance 2
```

2.2 VegsoupPartition

Objects of class `VegsoupPartition` extend class `Vegsoup` (that means all methods defined form this class apply to the extended class) and provide additional slots *k*, *part* and *method*. The number of partitions (clusters) an object is partitioned into is given by slot *k* and `getK` is the generic to access this slot. The assignment of plots to partitions is given by slot *part* and `Partitioning` returns the respective vector of group memberships. Finally the *method* slots, records the method which was applied when `VegsoupPartition` objects are created.

```
> showClass("VegsoupPartition")

Class "VegsoupPartition" [package "vegsoup"]

Slots:

Name:                      part      partitioning.method
Class:                  numeric                character

Name:                         k                  species
Class:                  numeric                  Species

Name:                     sites                 taxonomy
Class:                data.frame                 Taxonomy

Name:                 coverscale                decostand
Class:                Coverscale                decostand

Name:                      dist                    layers
Class:                character                character

Name:                     group                sp.points
Class:                  integer   SpatialPointsDataFrame

Name:                sp.polygons
Class: SpatialPolygonsDataFrame

Extends: "Vegsoup"

Known Subclasses: "VegsoupPartitionFidelity"
```

2.3 VegsoupOptimstride

Objects of class `VegsoupOptimstride` extend `Vegsoup`.

```
> showClass("VegsoupOptimstride")
```

```
Class "VegsoupOptimstride" [package "vegsoup"]

Slots:

Name:               optimstride                   species
Class:                      list                   Species

Name:                     sites                  taxonomy
Class:                data.frame                  Taxonomy

Name:                coverscale                 decostand
Class:                Coverscale                 decostand

Name:                      dist                    layers
Class:                 character                 character

Name:                     group                 sp.points
Class:                  integer   SpatialPointsDataFrame

Name:               sp.polygons
Class: SpatialPolygonsDataFrame

Extends: "Vegsoup"
```

2.4 VegsoupPartitionFidelity

Objects of class `VegsoupPartitionFidelity` extend `VegsoupPartition` and provide slots *stat*, *fisher.test*, *lowerCI*, *upperCI* and *nboot*. The before mentioned slots carry the results as obtained by the particular fidelity-method applied when creating an object of the class. Some slots of them might be NULL (lowerCI, upperCI, nboot).

```
> showClass("VegsoupPartitionFidelity")

Class "VegsoupPartitionFidelity" [package "vegsoup"]

Slots:

Name:                     stat              fisher.test
Class:                  matrix                   matrix

Name:                   lowerCI                  upperCI
Class:                  matrix                   matrix

Name:                     nboot           fidelity.method
Class:                  integer                 character
```

```
Name:                       part      partitioning.method
Class:                   numeric                character

Name:                          k                  species
Class:                   numeric                  Species

Name:                      sites                 taxonomy
Class:                 data.frame                 Taxonomy

Name:                  coverscale                decostand
Class:                 Coverscale                decostand

Name:                       dist                    layers
Class:                  character                 character

Name:                      group                 sp.points
Class:                   integer   SpatialPointsDataFrame

Name:                 sp.polygons
Class: SpatialPolygonsDataFrame

Extends:
Class "VegsoupPartition", directly
Class "Vegsoup", by class "VegsoupPartition", distance 2
```

3 Methods

```
> showMethods(class = "Vegsoup", where = "package:vegsoup")
> showMethods(class = "VegsoupPartition", where = "package:vegsoup")
> showMethods(class = "VegsoupPartitionFidelity", where = "package:vegsoup")
```

4 References

Chambers, J.M., 2009. Software for Data Analysis: Programming with R. Springer. 498p.