Reol

Barb Banbury, University of Tennessee, bbanbury@utk.edu

May 29, 2013

Reol is a package that interfaces the Encyclopedia of Life (EOL) with the R environment. It will download EOL pages via the API, and text is scraped for content and amassed into various datasets. *Reol* can be used to download and manipulate data about any taxonomic groups. In addition, data from provider pages can be downloaded and used for creating taxonomic trees or gathering taxonomic synonyms.

This document will provide a deeper explanation about the various functions than the help pages, and provide examples of typical application. I will be using the Great Apes as a working example, since it is a nice small group and the individual species have a lot of information.

1 Getting Started

This vignette assumes you have the current version of R. First, install and load the package. A stable release is available through CRAN (http://cran.r-project.org/web/packages/Reol/) or a working repository can also be used through R-Forge (https://r-forge.r-project.org/projects/reol/).

You can register as an EOL user on their website (http://eol.org/users/register) and generate and save an API key in your user profile. This key is a unique identifier that you can use when using the EOL API. Though it is not required, it is recommended to use a key, especially if you are going to be using the API heavily. All *Reol* functions that interact with the API have the option to include a key (`MyKey`).

To ensure that the API is up and running, you can use the `PingAPI` function. If there is an error, it will report the error message.

```
> library(Reol)
> PingAPI()

[1] "Success"
```

2 Downloading EOL Pages

The first step is to download EOL pages to your local machine. In later version, we might bypass the downloading functions and go straight into reading the xml files, but for now, you must download then gather data with the files. If you are running this example via sweave, these files will download to the R working directory you have at startup. If you are running the example by hand, you can set whichever working directory you wish to use using `setwd(your/path/)`. EOL pages will all download with an eol prefix, followed by the EOL ID, so they can easily be stored all in the same place. Verbosity will print downloaded file status to screen.

```
> GreatApes <- c("Pongo pygmaeus", "Pongo abelii", "Gorilla gorilla", "Gorilla b
> DownloadedApes <- DownloadSearchedTaxa(GreatApes, , verbose=F)
```

`DownloadSearchedTaxa` will return a table with the names of taxa originally submitted, the EOL taxon name (sometimes can be different than the submitted name), EOL ID numbers, and the associated file name. It is also possible to download taxa using the `DownloadEOLpages` function, which accepts the EOL ID number rather than a taxonomic name.

```
> DownloadedApes

       ListOfTaxa                          speciesNameForRef
1   Pongo pygmaeus                            Pongo pygmaeus
2     Pongo abelii                              Pongo abelii
3  Gorilla gorilla Gorilla gorilla (Savage and Wyman, 1847)
4 Gorilla beringei                          Gorilla beringei
5     Homo sapiens               Homo sapiens Linnaeus, 1758
6  Pan troglodytes       Pan troglodytes (Blumenbach, 1775)
7     Pan paniscus               Pan paniscus Schwartz, 1929
  eolPageNumbers      fileNames
1         326450  eol326450.xml
```

```
2        2925671 eol2925671.xml
3         326447  eol326447.xml
4        2923523 eol2923523.xml
5         327955  eol327955.xml
6         326449  eol326449.xml
7         326448  eol326448.xml
```

```
> MyEOLs <- DownloadedApes[,4]
```

3 Gathering Data from EOL pages

Any of EOL data can be gathered that is available via the API, but for now *Reol* is focused on numerical data (text mining is a future possibility). These gathering functions will all use the downloaded EOL pages. Remember though, that in order to find them, you have to be in the same working directory as the files are located. EOL file names should be as a vector, they will be used by all of the data gathering functions. The functions will collect data in various ways.

3.1 Richness

Richness score is an EOL metric that measures the amount of information a page contains. The value can be between 0 (no information) to 100 (all information) and is based on how much text a page has, how many multimedia or map files are available, how many different topics are covered, how many different sources contribute information, and whether information has been reviewed or not. You can read more about how it is calculated here: http://eol.org/pages/1/updates/statistics.

```
> GetRichnessScores(MyEOLs)
```

```
                                  Taxon   eolID
1                        Pongo pygmaeus  326450
2                          Pongo abelii 2925671
3 Gorilla gorilla (Savage and Wyman, 1847)  326447
4                      Gorilla beringei 2923523
5             Homo sapiens Linnaeus, 1758  327955
6       Pan troglodytes (Blumenbach, 1775)  326449
```

```
7              Pan paniscus Schwartz, 1929  326448
  Richness_Score
1        83.6257
2        81.3129
3        84.7522
4        72.6229
5        87.2866
6        87.2632
7        84.4066
```

3.2 Data Objects

Another type of data we can assemble is the kind and number of data objects that EOL pages house. These data objects can be images, videos, sound recordings, text, etc. The `CombineDataObjectInformation` function will return a very large data frame with information about each dataobject. This might be useful if you are looking for all the data objects from a particular provider or type (for example, all images submitted by fishbase). If there are a lot of data objects, it may hang your computer to try to print this to the screen. This function is probably best when used as an object and then sorted and subsetted. The `DataObjectOverview` gives an overview of the data object information by returning counts of each type of data. This function doesn't return any specific information, but you can determine if there is even distribution of objects across data types (for example, do birds and frogs have similar numbers of sound recordings). Verbosity refers to turning on or off print statements as it combines files for the analysis (may be helpful if you have a large number of files to combine, so you know that the program is running).

```
> DataObjectInfo <- CombineDataObjectInformation(MyEOLs, verbose=F)
> DataObjectInfo[1,]
```

```
           Taxon  eolID                     dataObjectID
1 Pongo pygmaeus 326450 36162c627fc99add86efa8bd66f75c13
  taxonConceptID                         dataType  mimeType
1         326450 http://purl.org/dc/dcmitype/Text text/html
   agent   title language
1 ARKive Biology       en
```

```
                                           license
1 http://creativecommons.org/licenses/by-nc-sa/3.0/
                        rights rightsHolder
1 Copyright Wildscreen 2003-2008   Wildscreen
        audience
1 General public
                                                  source
1 http://www.arkive.org/bornean-orang-utan/pongo-pygmaeus/
                                                  subject
1 http://rs.tdwg.org/ontology/voc/SPMInfoItems#TaxonBiology

1 Bornean orang-utans are predominantly solitary, occupying large overlapping home
  additionalInformation bibliographicCitation created
1              Trusted                  <NA>    <NA>
  modified reference mediaURL thumbnailURL location Point
1     <NA>      <NA>     <NA>         <NA>     <NA>  <NA>
```

```
> DataObjectOverview(MyEOLs, verbose=F)

             Taxon   eolID text.html image.jpeg text.plain
1   Pongo pygmaeus  326450        32         76          0
2     Pongo abelii 2925671        30         26          0
3  Gorilla gorilla  326447        33         76          0
4 Gorilla beringei 2923523        32         50         10
5     Homo sapiens  327955        52         76          5
6  Pan troglodytes  326449        35         76          0
7     Pan paniscus  326448        31         63          0
  image.png
1         0
2         0
3         0
4         0
5         1
6         0
7         0
```

3.3 Common Names

Common or vernacular names are also available on the EOL pages and their associated languages. If output is set to detail (or d), it will return a data frame with the taxon, EOL ID, common name, and language. In the following example, just the common names for humans are retrieved, but vectors of taxa are supported as well. If output=counts, then a data frame of language counts will be retuned without the common names.

```
> GetCommonNames(MyEOLs, output="c")
```

```
                                     Taxon   eolID en es fr
1                           Pongo pygmaeus  326450  5  2  2
2                             Pongo abelii 2925671  6  0  1
3 Gorilla gorilla (Savage and Wyman, 1847)  326447  4  2  1
4                         Gorilla beringei 2923523  6  0  0
5              Homo sapiens Linnaeus, 1758  327955  2  1  0
6        Pan troglodytes (Blumenbach, 1775)  326449  3  2  1
7              Pan paniscus Schwartz, 1929  326448  4  1  3
  de ru ca ur
1  0  0  0  0
2  1  0  0  0
3  0  1  0  0
4  1  0  0  0
5  0  2  1  3
6  1  1  0  0
7  0  0  0  0
```

3.4 References

This function gathers a collective bibliography from EOL pages. If output is set to detail, full bibliographic data will be returned as a data frame that contains the taxon, EOL ID, and the entire reference. This data is also available as counts, which will return a data frame with taxon, EOL ID, and the number of references each page contains.

```
> GetReferences(MyEOLs[1], output="d")[1,]
```

```
           Taxon  eolID
1 Pongo pygmaeus 326450

1 1. IUCN Red List  (September, 2007) <a href="http://www.iucnredlist.org">http://
```

```
> GetReferences(MyEOLs, output="c")

                                     Taxon   eolID
1                           Pongo pygmaeus  326450
2                             Pongo abelii 2925671
3 Gorilla gorilla (Savage and Wyman, 1847)  326447
4                         Gorilla beringei 2923523
5              Homo sapiens Linnaeus, 1758  327955
6      Pan troglodytes (Blumenbach, 1775)  326449
7              Pan paniscus Schwartz, 1929  326448
  Number.Of.References
1                   22
2                   26
3                   29
4                   24
5                  126
6                   46
7                   45
```

3.5 Providers

EOL has a number of content providers (see http://eol.org/info/222) that provide information about classifications and synonymy. This is data that falls under the names tab on the EOL website. *Reol* has a few functions for gathering provider information. `GatherProviderDataFrame` gathers the providers that are available for each taxon in the vector. It returns a data frame with boolean response, 1 if the provider has contributed information and 0 if they have not. Total number of providers for each taxon is the last column of the data frame. There is also the option to have it print an extended output, which will return information about provider IDs, taxonomic rank, database ID. The extended output format is used for other *Reol* functions. The `BestProvider` function calculates the provider that contributes the most information for the given set of taxa. If there is a tie, it returns only the first one on the list. This doesn't necessarily mean it is the best or most complete provider, so users beware. This function can be useful, however, for choosing provider pages to download.

```
> GatherProviderDataFrame(MyEOLs)

            Taxon   eolID
1   Pongo pygmaeus  326450
2     Pongo abelii 2925671
3  Gorilla gorilla  326447
4 Gorilla beringei 2923523
5     Homo sapiens  327955
6  Pan troglodytes  326449
7     Pan paniscus  326448
  IUCN Red List (Species Assessed for Global Conservation)
1                                                        1
2                                                        1
3                                                        1
4                                                        1
5                                                        1
6                                                        1
7                                                        1
  Integrated Taxonomic Information System (ITIS)
1                                              1
2                                              0
3                                              1
4                                              0
5                                              1
6                                              1
7                                              1
  Paleobiology Database NCBI Taxonomy
1                     1             1
2                     1             1
3                     1             1
4                     1             1
5                     1             1
6                     1             1
7                     1             1
```

```
  Species 2000 & ITIS Catalogue of Life: April 2013
1                                                 1
2                                                 0
3                                                 1
4                                                 0
5                                                 1
6                                                 1
7                                                 1
  number.sources
1              5
2              3
3              5
4              3
5              5
6              5
7              5
```

```
> BestProvider(MyEOLs)

[1] "IUCN Red List (Species Assessed for Global Conservation)"
```

4 Downloading Provider Pages

Just as EOL page content can be downloaded and scraped for content, so can the content off the provider pages. These pages will download to the working directory, and should be ok to stored together. Downloaded provider page names are prefixed with hier and followed by their provider ID, so they can be easily separated from EOL pages. Verbosity will print downloaded file status to screen

```
> NCBIfiles <- DownloadHierarchy(MyEOLs, database="NCBI Taxonomy", verbose=F)
```

5 Gathering Data from Provider Pages

There are essentially two pieces of information that can be gathered from the provider pages, the taxonomic hierarchy and a synonyms list. *Reol* utilizes both bits in several functions, which are described in detail below.

5.1 Taxonomic Synonyms

Each provider records their own set of taxonomic synonyms, so lists may be different from provider to provider. If output is set to detail, a data frame will be returned with the taxon name, the provider ID, and the synonym. If output is set to counts, then a data frame with taxon, provider ID, and the number of taxonomic synonyms is returned. These synonyms are scientific synonyms only, not misidentifications or vernacular names.

```
> GatherSynonyms(NCBIfiles, "d")

         Taxon   hierID               Synonym
1 Pongo abelii 51378546  Pongo pygmaeus abeli
2 Pongo abelii 51378546 Pongo pygmaeus abelii
```

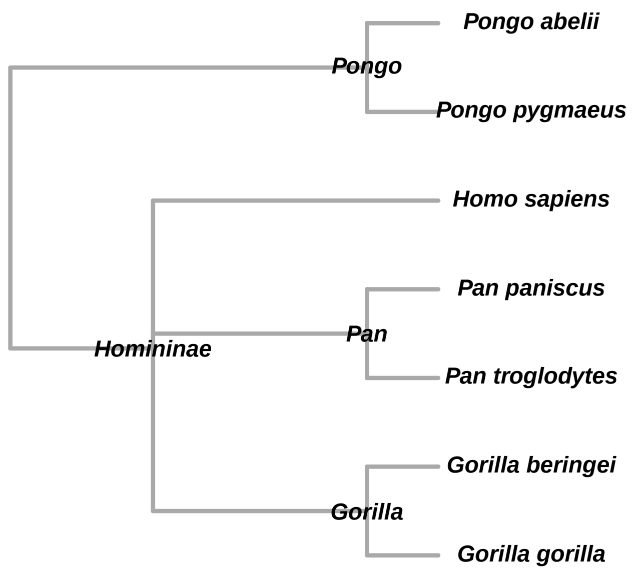
```
> GatherSynonyms(NCBIfiles, "c")

             Taxon   hierID NumberOfSynonyms
1   Pongo pygmaeus 51378544                0
2     Pongo abelii 51378546                2
3  Gorilla gorilla 51378523                0
4 Gorilla beringei 51378527                0
5     Homo sapiens 51378539                0
6  Pan troglodytes 51378532                0
7     Pan paniscus 51378531                0
```

5.2 Creating a Taxonomic Dendrogram

Each EOL provider contributes taxonomic hierarchy data. This data can be used to create a tree structure or dendrogram of taxonomic structure. These trees can be used in lieu of a phylogenetic tree if none exists and a re a good way to see patterns in the data. These trees can also be used to see taxonomic inconsistencies, either compared to a phylogenetic tree (ie paraphyletic taxa) or among providers. Note, that these trees only represent the taxonomic hierarchy, and are not a replacement for a phylogenetic analysis.

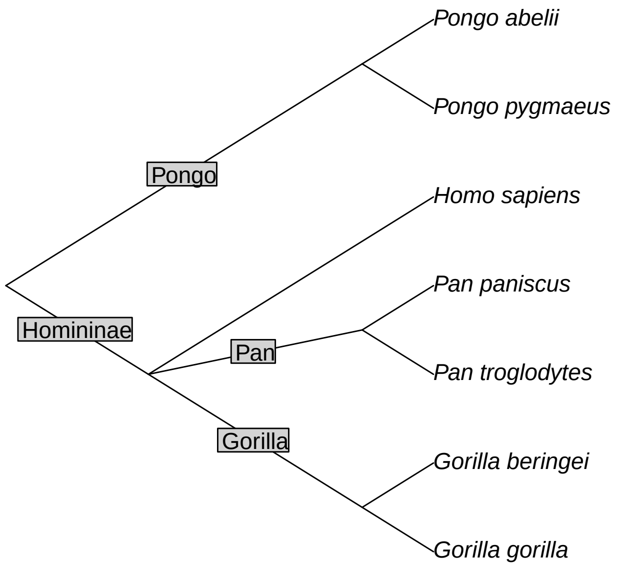
The tree structure follows the same formatting of the package ape (http://cran.r-project.org/web/packages/ape/), in the class phylo. The benefit is that you can use all of ape's plotting functions to make nice looking trees and mapping of traits.

```
> ApeTree <- MakeHierarchyTree(NCBIfiles, includeNodeLabels = TRUE)
> plot(ApeTree, "p", show.node.label=TRUE, adj=0.5, font=4, edge.width=3, edge.c
```



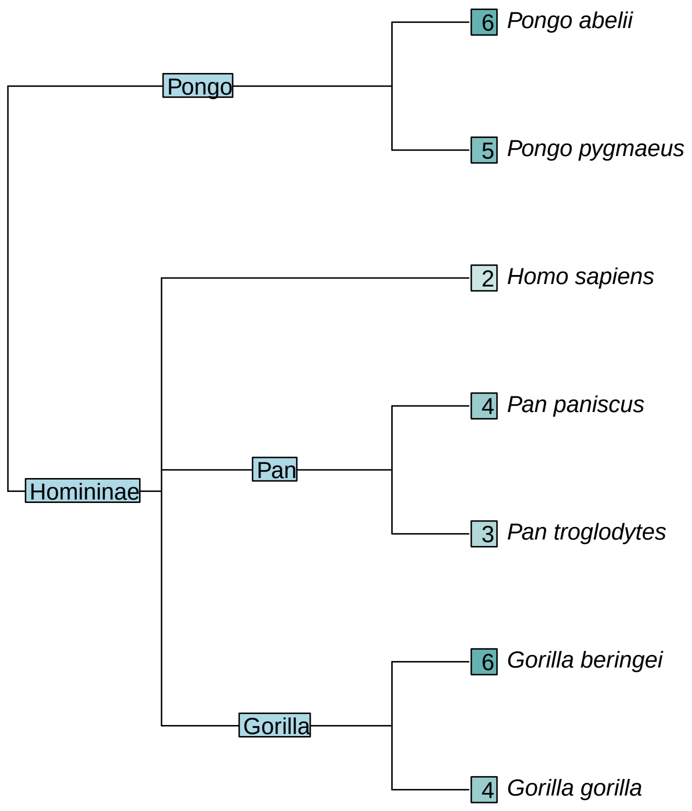
Reol also has a function to create edge labels with the taxonomic group names automatically. In some cases there are multiple taxonomic names for one edge, and at this point it will only print the last one. There is a bit more flexibility with visualization using the edge label functions rather than the node label functions.

```
> edges <- MakeEdgeLabels(NCBIfiles)
> plot(ApeTree, "c", show.node.label=FALSE)
> edgelabels(text=names(edges), edge=edges, bg="light gray")
```



These trees can be used to plot information about EOL data. For example, if we want to know patterns of the number of common names across our taxa, we can plot that information as a continuous trait along our new taxonomy tree.

```
> CNs <- GetCommonNames(MyEOLs, output="c")
> plot(ApeTree, label.offset=0.5, x.lim=10, no.margin=TRUE)
> edgelabels(text=names(edges), edge=edges, bg="light blue")
> trans <- CNs[,3]/10
> tiplabels(pch=22, bg=rgb(0,0.5,0.5,trans), cex=2.8, adj=0.7)
> tiplabels(CNs[,3], 1:7, frame="none", bg="clear",adj=-1)
```


6 Pongo abelii
Pongo
5 Pongo pygmaeus
2 Homo sapiens
4 Pan paniscus
Homininae
Pan
3 Pan troglodytes
6 Gorilla beringei
Gorilla
4 Gorilla gorilla