Reproducible ecology data with R

Scott Chamberlain

@recology_ & @ropensci

Shortcuts: M =  G = 

Interact with the talk

http://bit.ly/bestalk

Data used to be like this...

Image credits: boat and Darwin

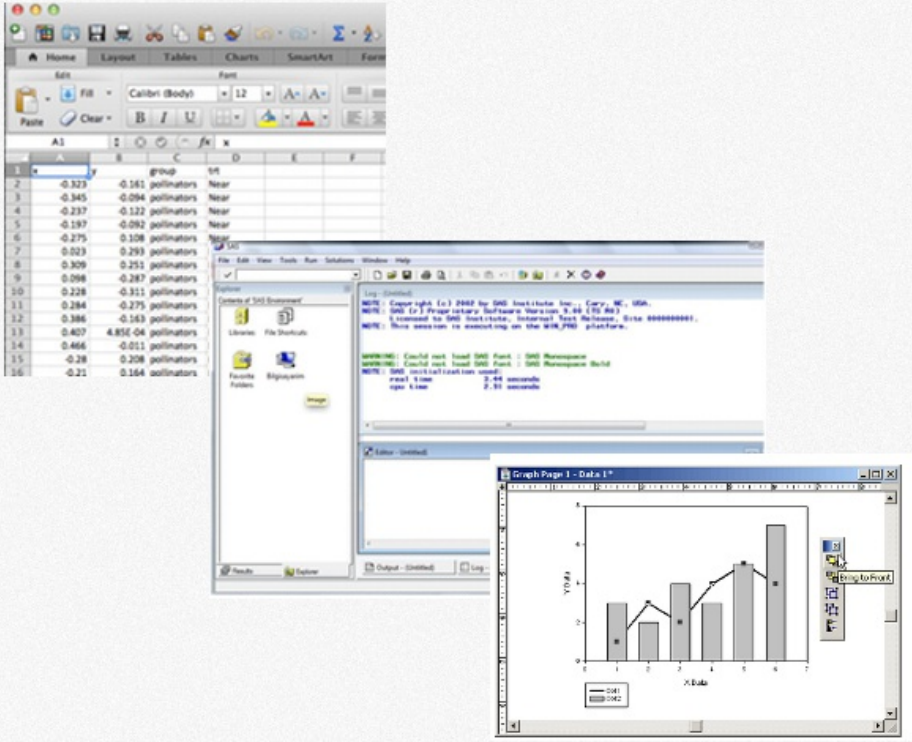
...but now lots of accessible data on the interwebs

But how should we get data?

- Programatically
- Within a familiar environment to biologists
- Within one environment to: manipulate, visualize, analyze

R + Data = smiles

The old way...



The new way...all in one...R in this case

Make an API call

```
library(RCurl); library(RJSONIO)
dat <- fromJSON(getURL("https://api.github.com/users/hadley/repos"))
```

Manipulate the data

```
library(plyr); library(reshape2)
dat_melt <- melt(ldply(dat, function(x) as.data.frame(x[names(x) %in% c("name", "watchers_count",
    "forks", "open_issues")])))
```

Run some statistical model

```
lm(value ~ variable, data = dat_melt)
```

Visualize results

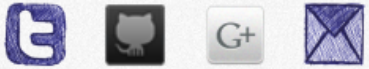
```
library(ggplot2)
ggplot(dat_melt, aes(name, value, colour = variable)) + geom_point() + coord_flip()
```

Reproducible,

repeatable,

analytic workflow


R
OpenSci



Development team

Carl Boettiger

Karthik Ram

Scott Chamberlain

Edmund Hart

Advisory team

Duncan Temple Lang
UC Davis

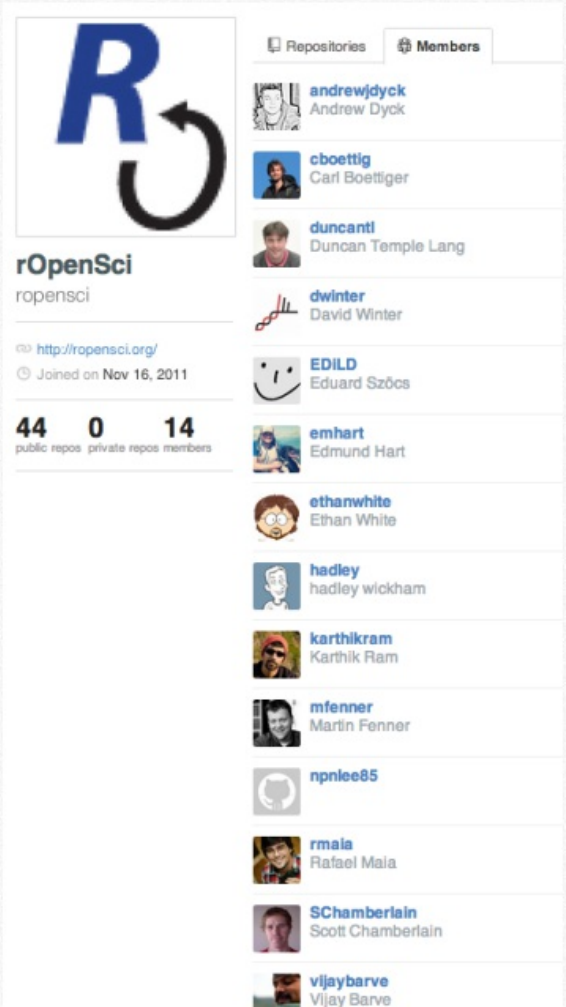
Hadley Wickham
RStudio

JJ Allaire
RStudio

Bertram Ludascher
UC Davis

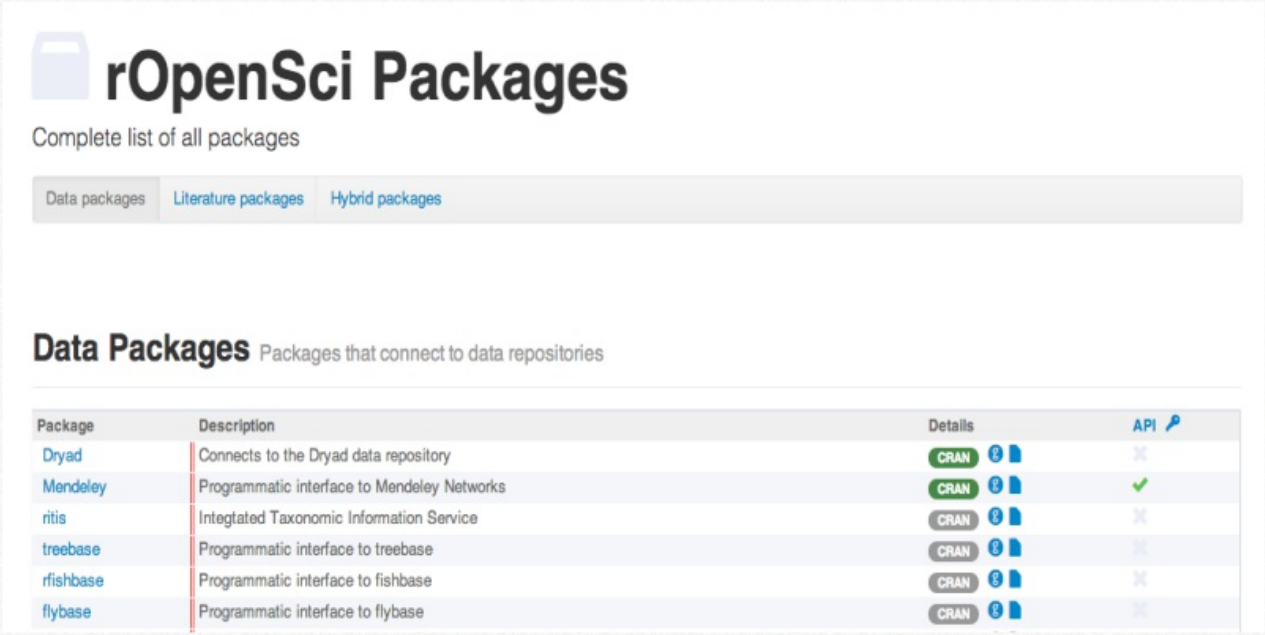
Matt Jones
DataONE/NCEAS

But many people contribute:



rOpenSci packages

http://ropensci.org/packages/index.html



rOpenSci Packages

Complete list of all packages

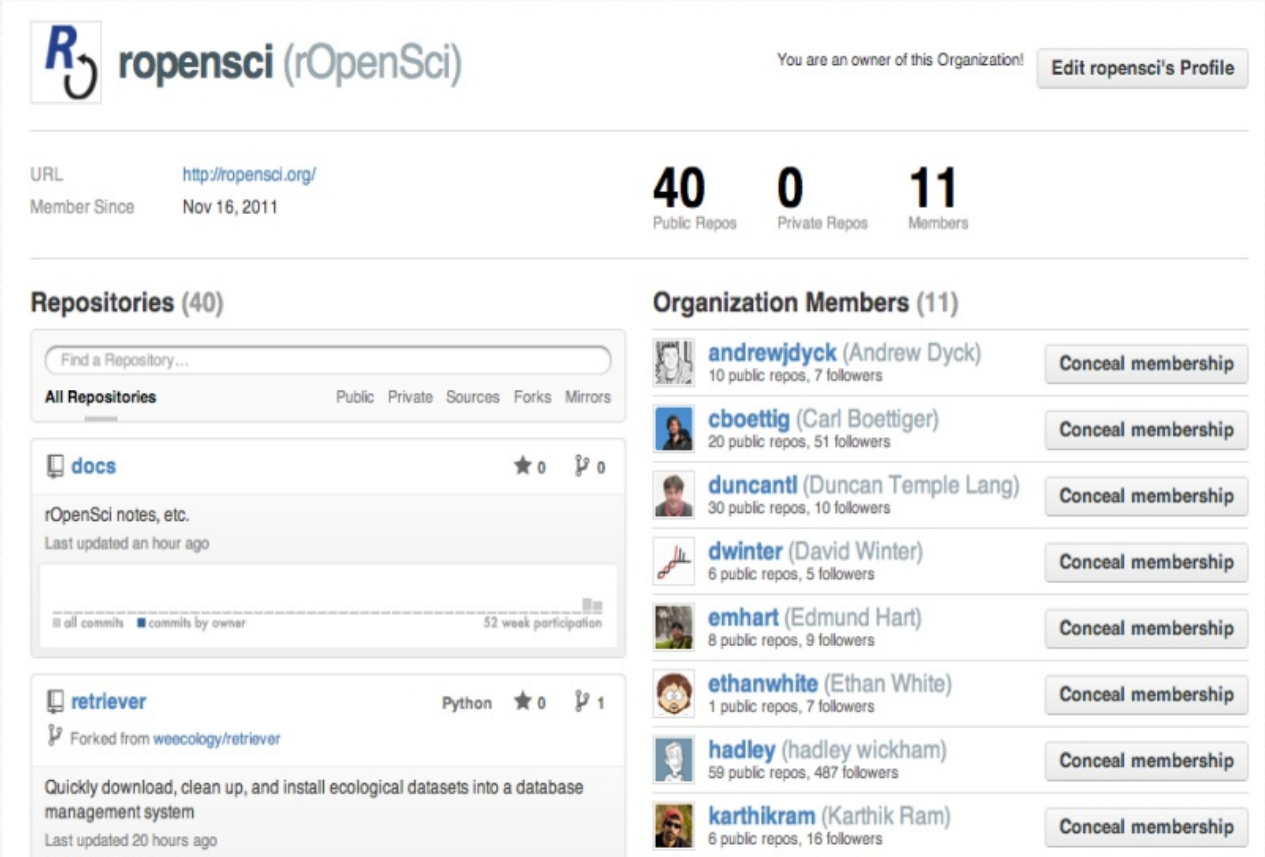
Data packages | Literature packages | Hybrid packages

Data Packages Packages that connect to data repositories

Package	Description	Details	API
Dryad	Connects to the Dryad data repository	CRAN	✗
Mendeley	Programmatic interface to Mendeley Networks	CRAN	✓
ritis	Integtated Taxonomic Information Service	CRAN	✗
treebase	Programmatic interface to treebase	CRAN	✗
rfishbase	Programmatic interface to fishbase	CRAN	✗
flybase	Programmatic interface to flybase	CRAN	✗

rOpenSci packages

https://github.com/ropensci



Others have R pkgs to get data

Others have R pkgs to get data

But rOpenSci is a focused effort to build bridges to data via R

Getting data from the web into R

- **Web scraping html, xml, etc.**
- **Reading json, csv, txt, etc.**
- **Hitting an Application Programming Interface (API)**
  - **This is the preferred, and most common method we use**
  - **Some require an API key, some do not**

Authentication: R and APIs - API keys

API keys can be stored in a users.Rprofile file

Click here for help on .Rprofile files

```
options(MendeleyKey = "uf5daib7wyil7ag5buc")
options(MendeleyPrivateKey = "faj2os5dyd7jop2fok6")
options(PlosApiKey = "ef3vip9yak7od3hud4g")
options(SpringerMetdataKey = "ri9hi7woc6jax4vaf8w")
```

Note: These keys aren't real.

A workflow

Taxonomic Correction:

Are our species names correct? Need higher taxonomy (e.g., family names)?

Getting data:

Occurrence record, Specimen records from museums, Citizen science data

Sharing secondary data:

Sharing data increases impact of your research

And allows others to reuse (possibly you)

Taxonomic correction

Check species names - pkg taxize w/ plantminer

```
library(taxize)

# Using plantminer
plants <- c("Myrcia lingua", "Myrcia bella", "Coffea arabica", "Bleh")
out <- plantminer(plants)
```

```
out[, !names(out) %in% c("author", "source", "source.id")]
```

```
       fam  genus       sp    status confidence suggestion       database
1 Myrtaceae Myrcia  lingua        NA         NA         NA       Tropicos
2 Myrtaceae Myrcia   bella Accepted          H         NA The Plant List
3 Rubiaceae Coffea arabica Accepted          H         NA The Plant List
4        NA   Bleh       NA        NA         NA       Baea           NA
```

Check species names - taxize with theplantlist.org

```
# Using theplantlist.org
splist <- c("Poa annua", "Platanus occidentalis", "Carex abrupta", "Arctostaphylos canescens",
   "Ocimum basilicum", "Vicia faba", "Quercus kelloggii", "Lactuca serriola")
out2 <- tpl_search(taxon = splist)
out2[, names(out2) %in% c("Genus", "Species", "New.Genus", "New.Species")]
```

```
          Genus      Species      New.Genus  New.Species
1           Poa       annua           Poa       annua
2      Platanus occidentalis      Platanus occidentalis
3         Carex      abrupta         Carex      abrupta
4 Arctostaphylos    canescens Arctostaphylos    canescens
5        Ocimum    basilicum        Ocimum    basilicum
6         Vicia        faba         Vicia         faba
7       Quercus    kelloggii       Quercus    kelloggii
8       Lactuca     serriola       Lactuca     serriola
```

Taxonomic problems? - taxize with ITIS

```
# Using ITIS
tsn <- get_tsn("Compositae")
data.frame(getacceptednamesfromtsn(tsn))
```

```
  submittedTsn acceptedName acceptedTsn
1        35421   Asteraceae       35420
```

Taxonomic problems? - taxize with ITIS

```
# Using ITIS
tsn <- get_tsn("Compositae")
data.frame(getacceptednamesfromtsn(tsn))
```

```
  submittedTsn acceptedName acceptedTsn
1        35421   Asteraceae       35420
```

```
# Get names downstream
itis_downstream(tsns = 846509, downto = "Genus")
```

```
    tsn parentName parentTsn   taxonName rankId rankName
1 11531      11530      Bangia    180    Genus
2 11540      11530    Porphyra    180    Genus
3 11577      11530 Porphyrella    180    Genus
4 11580      11530 Conchocelis    180    Genus
```

Get higher taxonomy names - taxize

```
library(taxize)
species <- c("Poa annua", "Abies procera", "Helianthus annuus", "Coffea arabica")
famnames <- sapply(species, tax_name, get = "family", db = "ncbi", USE.NAMES = F)
data.frame(species = species, family = famnames)
```

Get higher taxonomy names - taxize

```
library(taxize)
species <- c("Poa annua", "Abies procera", "Helianthus annuus", "Coffea arabica")
famnames <- sapply(species, tax_name, get = "family", db = "ncbi", USE.NAMES = F)
data.frame(species = species, family = famnames)
```

```
          species     family
1       Poa annua    Poaceae
2    Abies procera   Pinaceae
3 Helianthus annuus Asteraceae
4   Coffea arabica  Rubiaceae
```

Getting data

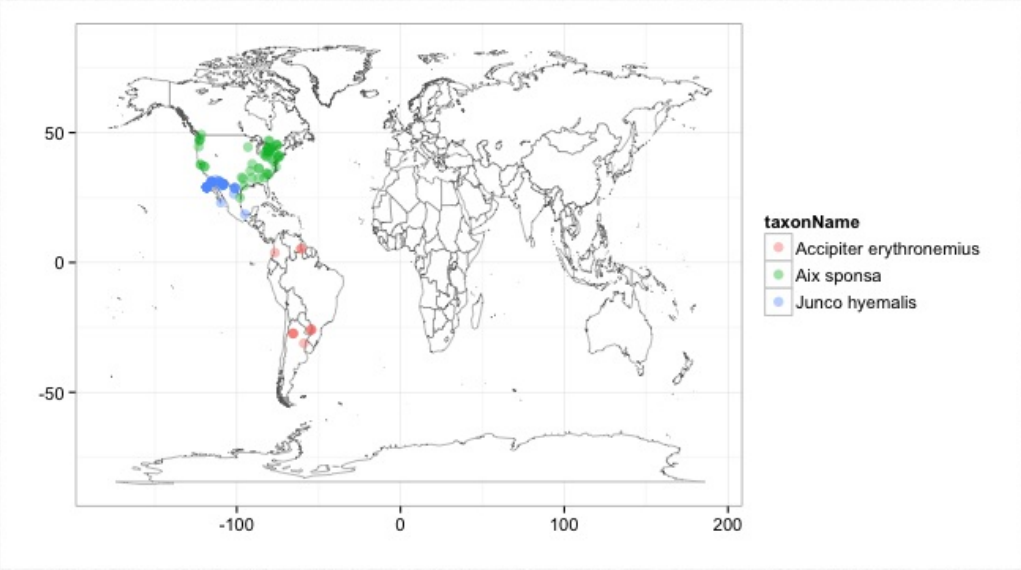
rgbif demo

Mapping biodiversity data - pkg rgbif

```
library(rgbif)
splist <- c("Accipiter erythronemius", "Junco hyemalis", "Aix sponsa")
out <- lapply(splist, function(x) occurrencelist(x, coordinatestatus = T, maxresults = 50))
gbifmap(out)
```

Mapping biodiversity data - pkg rgbif

```
library(rgbif)
splist <- c("Accipiter erythronemius", "Junco hyemalis", "Aix sponsa")
out <- lapply(splist, function(x) occurrencelist(x, coordinatestatus = T, maxresults = 50))
gbifmap(out)
```



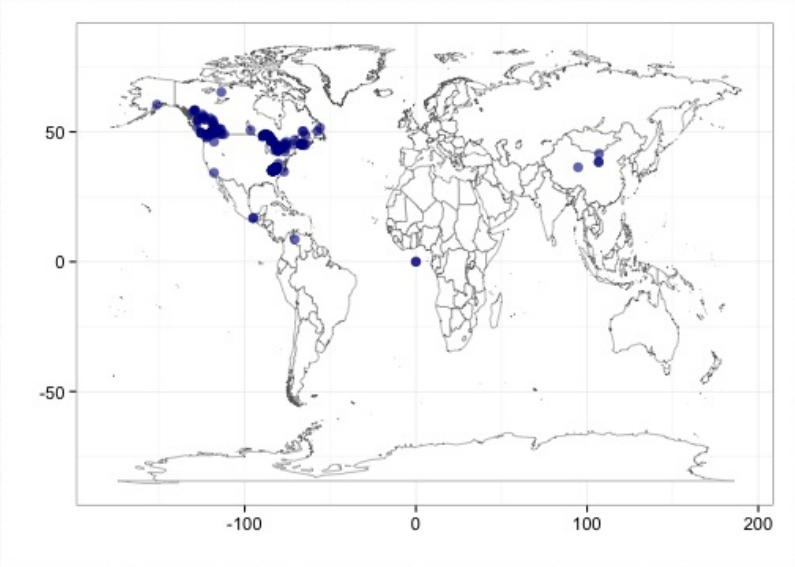
rvertnet demo

Vertebrate specimens - VertNet.org - rvertnet

```
library(rvertnet)
out <- vertoccurrence(t = "Oncorhynchus mykiss", grp = "fish", num = 300)
vertmap(input = out)
```

Vertebrate specimens - VertNet.org - rvertnet

```
library(rvertnet)
out <- vertoccurrence(t = "Oncorhynchus mykiss", grp = "fish", num = 300)
vertmap(input = out)
```

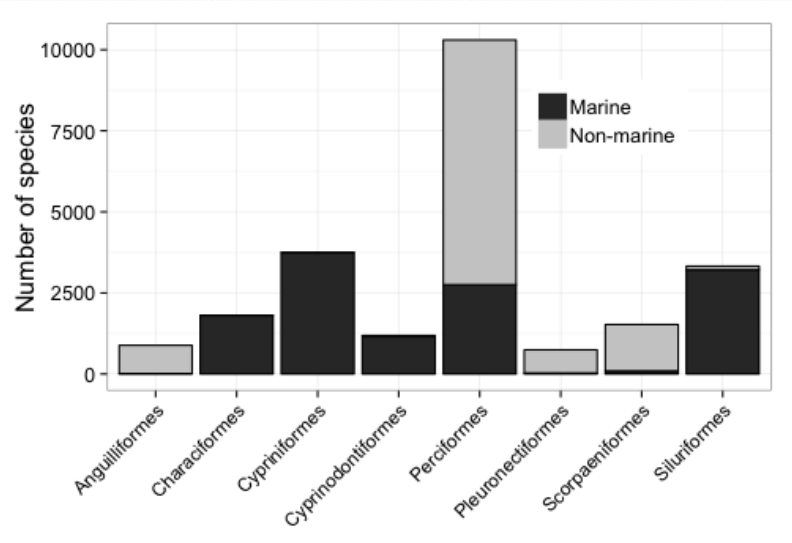


Fishbase.org - pkg rfishbase

```
library(rfishbase)
library(ggplot2)
loadCache()
reef <- which_fish("reef", "habitat", fish.data)

...code here

ggplot(primary_orders, aes(order, fill = marine)) + geom_bar() + # a few commands to customize appearance
geom_bar(colour = "black", show_guide = FALSE) + theme_bw(base_size = 16) +
   theme(axis.text.x = element_text(angle = 45, hjust = 1, size = 12), legend.title = element_blank(),
        legend.justification = c(1, 0), legend.position = c(0.9, 0.6)) +
   scale_fill_grey(labels = c("Marine",
   "Non-marine")) +
   labs(x="",y="Number of species")
```

Fishbase.org - pkg rfishbase

```
library(rfishbase)
library(ggplot2)
loadCache()
reef <- which_fish("reef", "habitat", fish.data)

...code here

ggplot(primary_orders, aes(order, fill = marine)) + geom_bar() + # a few commands to customize appearance
geom_bar(colour = "black", show_guide = FALSE) + theme_bw(base_size = 16) +
   theme(axis.text.x = element_text(angle = 45, hjust = 1, size = 12), legend.title = element_blank(),
        legend.justification = c(1, 0), legend.position = c(0.9, 0.6)) +
   scale_fill_grey(labels = c("Marine",
   "Non-marine")) +
   labs(x="",y="Number of species")
```

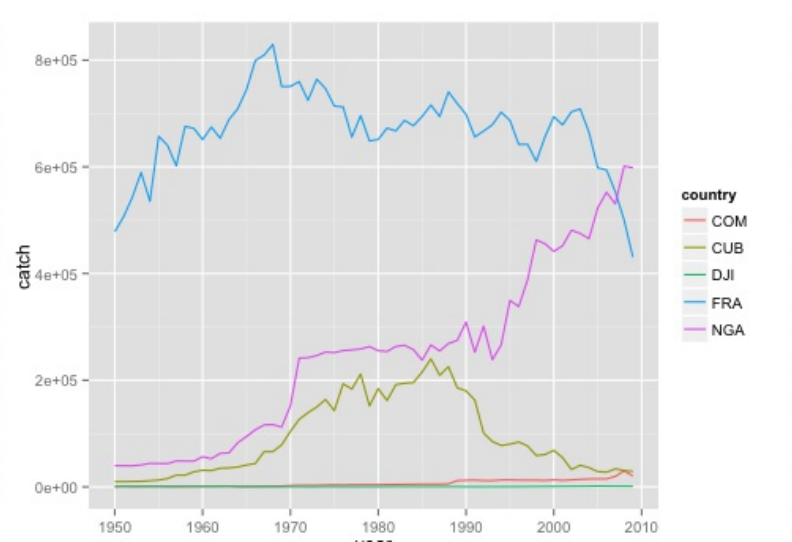


Openfisheries.org - pkg rfisheries

```
library(rfisheries); library(ggplot2)
countries <- country_codes()

# let's take a small subset, say 5 random countries
c_list <- countries[sample(nrow(countries), 5), ]$iso3c

# and grab landings data for these countries
results <- ldply(c_list, function(x) {
    df <- landings(country = x)
    df$country <- x
    df
}, .progress = "text")

ggplot(results, aes(year, catch, group = country, color = country)) + geom_line()
```

Openfisheries.org - pkg rfisheries

```
library(rfisheries); library(ggplot2)
countries <- country_codes()

# let's take a small subset, say 5 random countries
c_list <- countries[sample(nrow(countries), 5), ]$iso3c

# and grab landings data for these countries
results <- ldply(c_list, function(x) {
    df <- landings(country = x)
    df$country <- x
    df
}, .progress = "text")

ggplot(results, aes(year, catch, group = country, color = country)) + geom_line()
```



Get phylogenies Treebase.org - pkg treebase

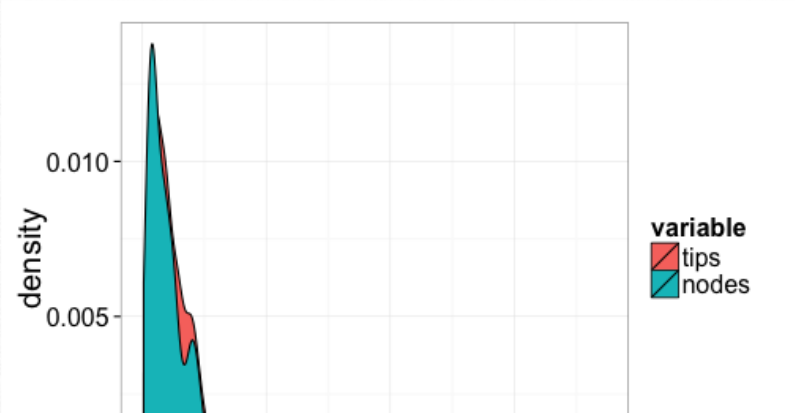
```
library(treebase); library(ggplot2); library(reshape2)

data(treebase)
have <- have_branchlength(treebase)
branchlengths <- treebase[have]
dat <- data.frame(tips = sapply(branchlengths, Ntip), nodes = sapply(branchlengths, Nnode))
dat_m <- melt(dat)
```

```
ggplot(dat_m[dat_m$value < 1000, ], aes(value, fill = variable)) + theme_bw(base_size = 20) +
    geom_density()
```

Get phylogenies Treebase.org - pkg treebase

```
library(treebase); library(ggplot2); library(reshape2)

data(treebase)
have <- have_branchlength(treebase)
branchlengths <- treebase[have]
dat <- data.frame(tips = sapply(branchlengths, Ntip), nodes = sapply(branchlengths, Nnode))
dat_m <- melt(dat)
```

```
ggplot(dat_m[dat_m$value < 1000, ], aes(value, fill = variable)) + theme_bw(base_size = 20) +
    geom_density()
```



Sharing data

Sharing and using data - pkg rfigshare

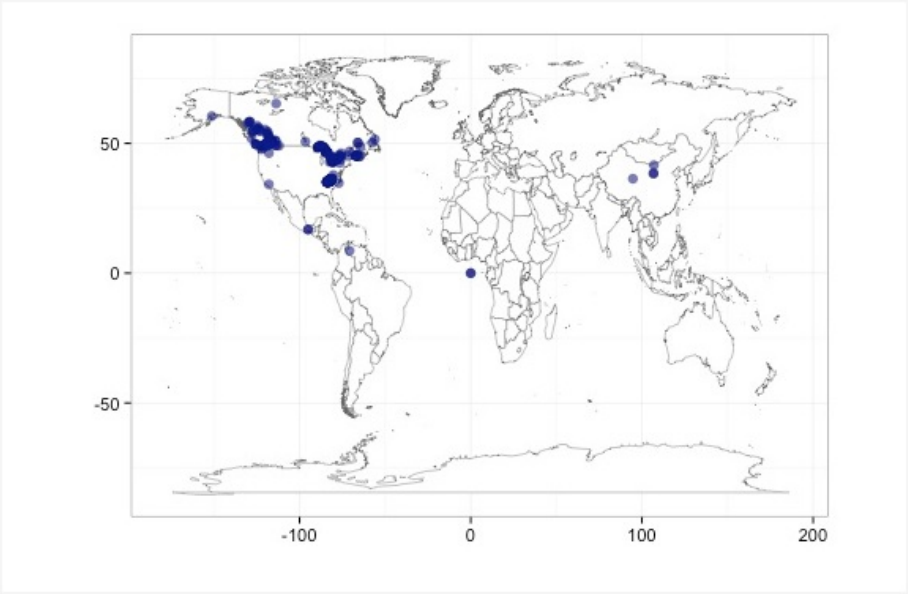
Using Figshare's new API, it is now possible to share figures, data, and any other object generated in R directly to one's figshare account. After authenticating:

```
library(rfigshare)

# Try creating your own content:
id <- fs_new_article(title = "A Test of rfigshare", description = "testing",
    type = "figure", authors = "Scott Chamberlain", tags = c("ecology", "openscience"),
    categories = "Ecology", links = "http://ropensci.org", files = "figure/rvertnet.png")
```

Sharing and using data - pkg rfigshare

A Test of rfigshare

Edit article



Preview Download

Share this: No sharing options for unpublished data

0 views 0 shares cites coming soon

Categories

- Ecology

Authors

Scott Chamberlain

Tags

ecology openscience

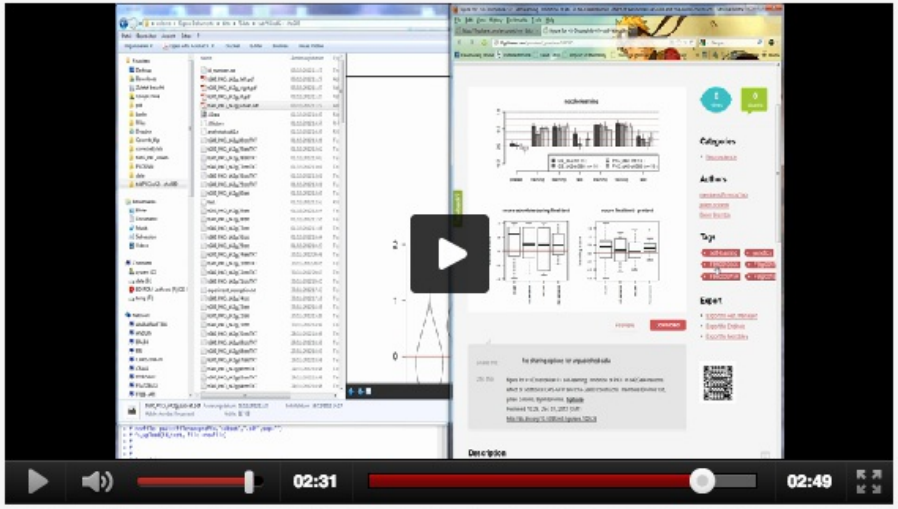
Export

- Export to Ref. Manager
- Export to Endnote
- Export to Mendeley

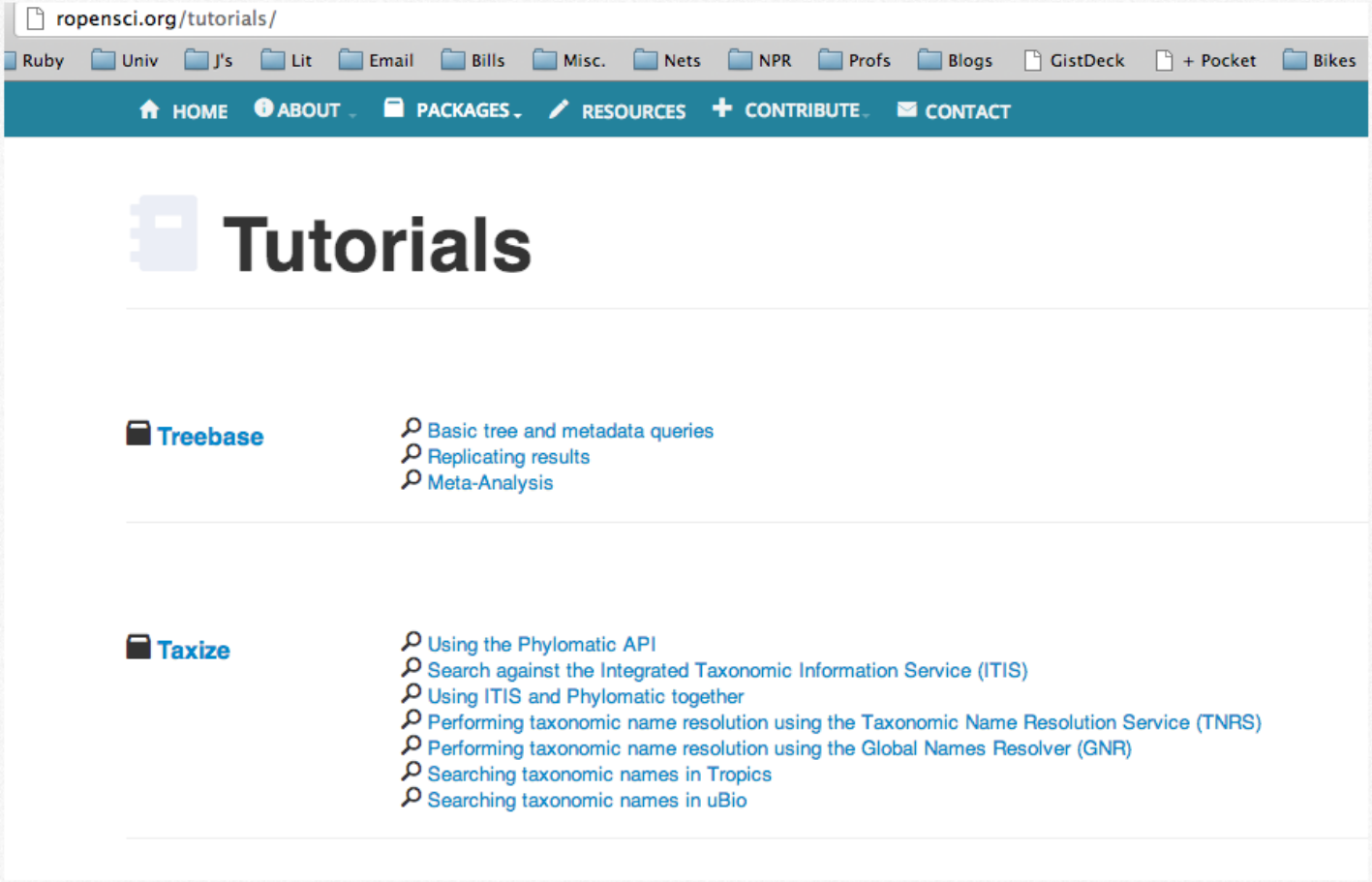
Sharing and using data - (rfigshare)

figshare My data Browse

automatic analysis and publishing of data



rOpenSci tutorials



ropensci.org

Please contact us if you have feedback or ideas for collaborations!

Code open-sourced on GitHub

Elsewhere @ropensci and G+

And I'm @recology_