

R: A Software System for Oceanographic Work at the Interface

Clark Richards¹ Dan Kelley²

RBR



¹RBR Ltd.
Ottawa, ON
`clarkrichards.org`

²Dalhousie University
Halifax, NS

2016-02-24

Data analysis tools are as important to our work as the instruments that collect the data

Increasingly scientists need to consider:

- code sharing (i.e. “collaboration”)
- public access
- long-term access and reproducibility

Data analysis tools are as important to our work as the instruments that collect the data

Increasingly scientists need to consider:

- code sharing (i.e. “collaboration”)
- public access
- long-term access and reproducibility

What is R?

www.r-project.org

www.r-project.org

R is a free software environment for statistical computing and graphics. R is available as Free Software under the terms of the Free Software Foundation's GNU General Public License in source code form. It compiles and runs on a wide variety of UNIX platforms and similar systems (including FreeBSD and Linux), Windows and MacOS.



Why R?

Community:

- Free (“as in beer” and “as in speech”)
- Large (and growing), diverse user base (including non-geoscience)
- Organized (R Core Team, R Foundation, R Consortium)

Technical:

- packages and the CRAN package repository
- object orientation
- reproducible research tools

Why R?

Community:

- Free (“as in beer” and “as in speech”)
- Large (and growing), diverse user base (including non-geoscience)
- Organized (R Core Team, R Foundation, R Consortium)

Technical:

- packages and the CRAN package repository
- object orientation
- reproducible research tools

Package system



What is a “package”?

A rigorous and precise system for collecting functions and data that ensures:

- working code
- accurate documentation
- correct results (tests)
- cross-platform

Comprehensive R Archive Network (CRAN)



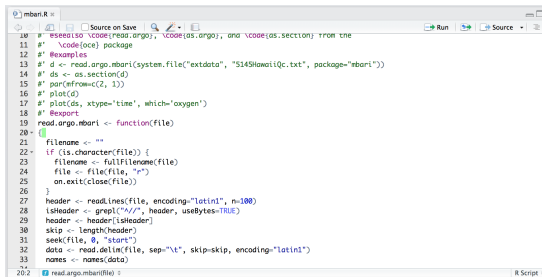
CRAN

A **curated** repository for packages

- ensures dependency installation
- archived versions
- frequently updated and maintained
- unambiguous copyright and license requirements (i.e. FOSS)
- currently over 6000 packages available

What's in a package?

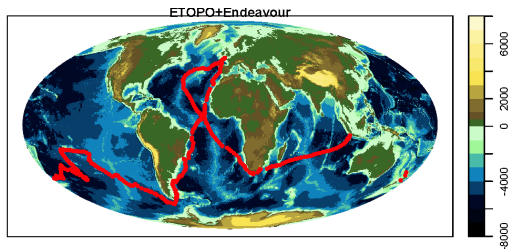
- code (functions)
- data
- documentation
- tests/examples/vignettes
- dependency information (and other metadata)



```
10 #' @section{code} package
11 #' \code{read.argo}, \code{as.argo}, and \code{as.section} from the
12 #' \code{mbari} package
13 #' @examples
14 #' d <- read.argo.mbari(system.file("extdata", "5145HawaiiQc.txt", package="mbari"))
15 #' ds <- as.section(d)
16 #' par(mfrow=c(2, 1))
17 #' plot(d)
18 #' plot(ds, xtype='time', which='oxygen')
19 #' @export
20 read.argo.mbari <- function(file)
21 {
22   filename <- ""
23   if (is.character(file)) {
24     filename <- fullFilename(file)
25     file <- file(file, "r")
26     on.exit(close(file))
27   }
28   header <- readLines(file, encoding="latin1", n=100)
29   isHeader <- grepl("^//", header, useBytes=TRUE)
30   header <- header[isHeader]
31   skip <- length(header)
32   seek(file, 0, "start")
33   data <- read.delim(file, sep="\t", skip=skip, encoding="latin1")
34   names <- names(data)
35 }
```

What's in a package?

- code (functions)
- data
- documentation
- tests/examples/vignettes
- dependency information (and other metadata)



What's in a package?

- code (functions)
- data
- documentation
- tests/examples/vignettes
- dependency information (and other metadata)

Read an MBARI BioArgo file

Description

Reads a "Bio-Argo" formatted file into an `oce` `argo` object.

Usage

```
read.argo.mbari(file)
```

Arguments

`file` filename or connection from which to read data

Details

What's in a package?

- code (functions)
- data
- documentation
- tests/examples/vignettes
- dependency information (and other metadata)

Clark Richards, Dan Kelley, Cara Wilson

See Also

`read.argo`, `as.argo`, and `as.section` from the `oce` package

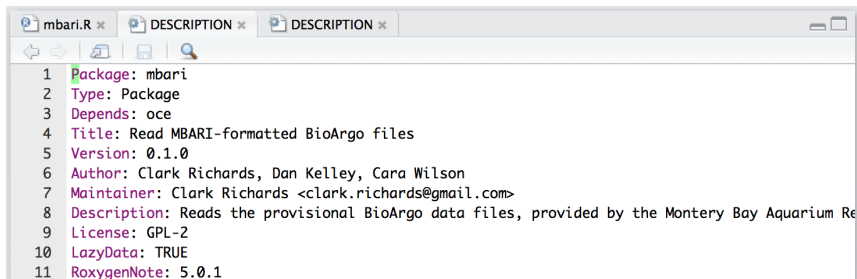
Examples

```
d <- read.argo.mbari(system.file("extdata", "5145HawaiiQc.txt", package="mbari"))
ds <- as.section(d)
par(mfrow=c(2, 1))
plot(d)
plot(ds, xtype='time', which='oxygen')
```

[Package *mbari* version 0.1.0 [Index](#)]

What's in a package?

- code (functions)
- data
- documentation
- tests/examples/vignettes
- dependency information (and other metadata)

A screenshot of an R IDE window showing the 'DESCRIPTION' file for a package named 'mbari'. The window has three tabs: 'mbari.R', 'DESCRIPTION', and another 'DESCRIPTION'. The 'DESCRIPTION' tab is active, showing the package metadata. The code is as follows:

```
1 Package: mbari
2 Type: Package
3 Depends: oce
4 Title: Read MBARI-formatted BioArgo files
5 Version: 0.1.0
6 Author: Clark Richards, Dan Kelley, Cara Wilson
7 Maintainer: Clark Richards <clark.richards@gmail.com>
8 Description: Reads the provisional BioArgo data files, provided by the Monterey Bay Aquarium Research Institute
9 License: GPL-2
10 LazyData: TRUE
11 RoxygenNote: 5.0.1
```

The **oce** package

What is **oce**?

Oce is a package that helps Oceanographers by providing functions to read Oceanographic data files, to process the data in instrument-specific ways, and to represent the results with plots that follow Oceanographic convention.

dankelley.github.io/oce



[Overview](#)

[Installation](#)

[Examples](#)

[Documentation](#)

[Datasets](#)

[Bugs](#)

Overview

Oce is a package for the [R statistical language](#) that helps Oceanographers do their work by providing functions to read Oceanographic data files, to process the data in instrument-specific ways, and to represent the results with plots that follow Oceanographic convention.

Developed in university and research settings, Oce is simple enough for novices but powerful enough for experts.

Oce makes heavy use of the R notion of *generic functions*, so that a single function call works across a wide range of data types. For example,

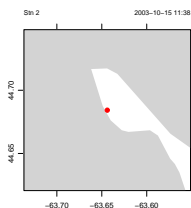
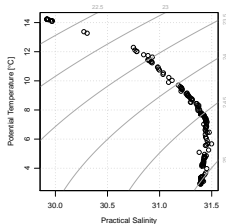
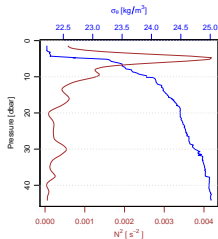
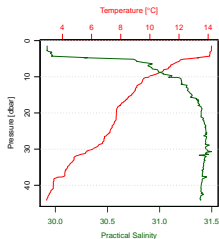
```
library(oce)
d <- read.oce("file")
plot(d)
```

will produce a useful plot that is tailored to the type of data stored in the file named `file`. For example, if `file`

oce example

```
## install.packages('oce')  
library(oce)  
data(ctd)  
plot(ctd)
```

```
# install from CRAN  
# load the package  
# an example dataset  
# default ctd plot
```



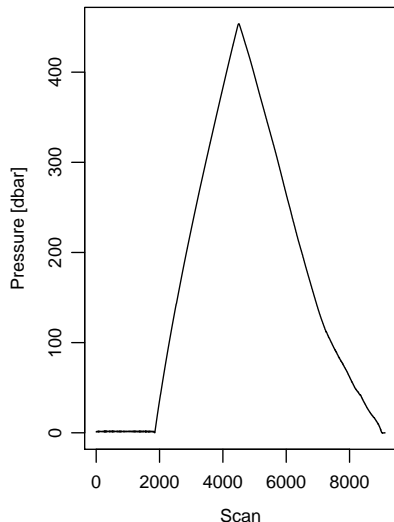
Working with CTD data

Data collected in Northwest Greenland by OceanResearchProject.org

```
library(oce)
rsk <- read.oce('data/ORP.rsk')
ctd <- as.ctd(rsk)
plotScan(ctd)
```

```
## trim the cast
ctd <- ctdTrim(ctd,
  parameters=list(pmin=3))
plotScan(ctd)
```

```
## Plot some profiles
plotProfile(ctd)
```



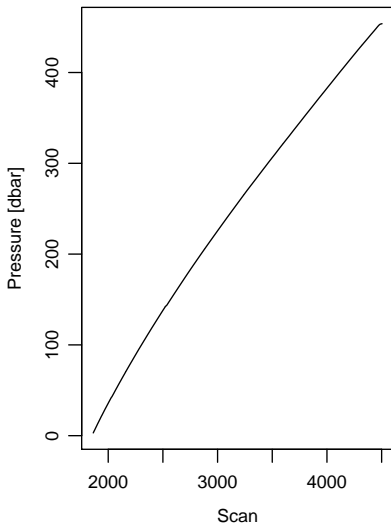
Working with CTD data

Data collected in Northwest Greenland by OceanResearchProject.org

```
library(oce)
rsk <- read.oce('data/ORP.rsk')
ctd <- as.ctd(rsk)
plotScan(ctd)
```

```
## trim the cast
ctd <- ctdTrim(ctd,
  parameters=list(pmin=3))
plotScan(ctd)
```

```
## Plot some profiles
plotProfile(ctd)
```



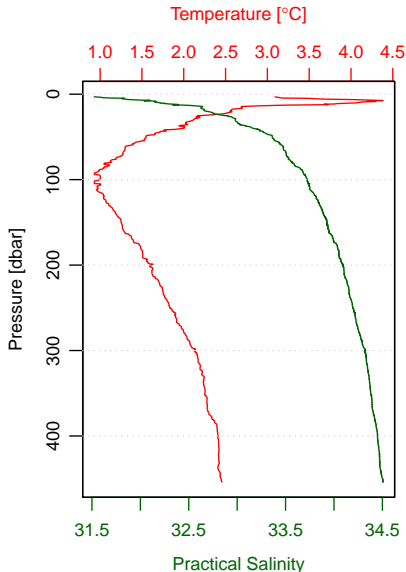
Working with CTD data

Data collected in Northwest Greenland by OceanResearchProject.org

```
library(oce)
rsk <- read.oce('data/ORP.rsk')
ctd <- as.ctd(rsk)
plotScan(ctd)
```

```
## trim the cast
ctd <- ctdTrim(ctd,
  parameters=list(pmin=3))
plotScan(ctd)
```

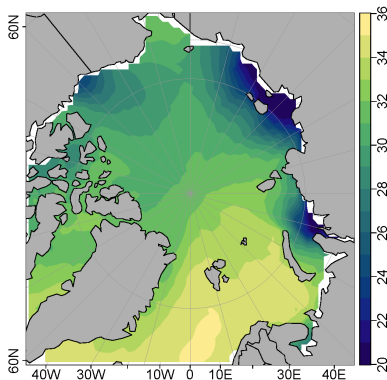
```
## Plot some profiles
plotProfile(ctd)
```



```

data(levitus)
SSScmap <- colormap(levitus$SSS, breaks=seq(20, 36, 1),
                    col=oceColorsSalinity, missingColor = NA)
drawPalette(colormap=SSScmap)
mapPlot(coastlineWorld, projection='+proj=stere +lat_0=90',
        longitudelim = c(-180, 180), latitudelim = c(70, 90))
mapImage(levitus$longitude, levitus$latitude,
        levitus$SSS, colormap=SSScmap, filledContour = TRUE)
mapPolygon(coastlineWorld, col='grey'); mapGrid()

```



R and oce are user driven

- Development on Github
 - ▶ Issues/bugs/requests
 - ▶ Pull Requests

<https://github.com/dankelley/oce>

The screenshot shows the GitHub interface for the repository `dankelley / oce`. The top navigation bar includes links for Pull requests, Issues, and Gist. The repository name is displayed as `dankelley / oce`. Below the repository name, there are tabs for Code, Issues (12), Pull requests (0), Wiki, Pulse, and Graphs. The Issues tab is selected, showing a list of 12 open issues. The issues are filtered by 'is:issue is:open'. The list of issues includes:

- Bio-Argo (again)** (argo, graphics, Needs Recheck, not an oce issue, section) - #881 opened 4 days ago by CaraWilson
- landsat terralook has an unnatural hue** (landsat) - #874 opened 9 days ago by dankelley
- landsat pan-sharpening desired** (landsat, low priority) - #868 opened 15 days ago by dankelley
- Read all relevant RSK metadata** (VO, Next Release, rsk) - #850 opened 21 days ago by richardsc

The interface also shows options to Unwatch (8), Unstar (35), and Fork (12) the repository. A 'New Issue' button is visible on the right side of the issues list.

Other useful ocean packages

- **ocedata** (extra data sets for oce)
- **ncdf4** (read/write netcdf files)
- **gsw** (TEOS-10 routines)
- **rerddap** (Connect to the NOAA ERDDAP server)
- **OceanView** (3D data/model visualization)
- **marmap** (bathymetry data and mapping)
- **seacarb** (seawater carbonate chemistry)
- **signal** (time series tools, e.g. filters)
- **fields** (spatial gridding/interpolation)
- **deSolve** (differential equations)
- **rpy2** (connect R and python)
- ...

More resources

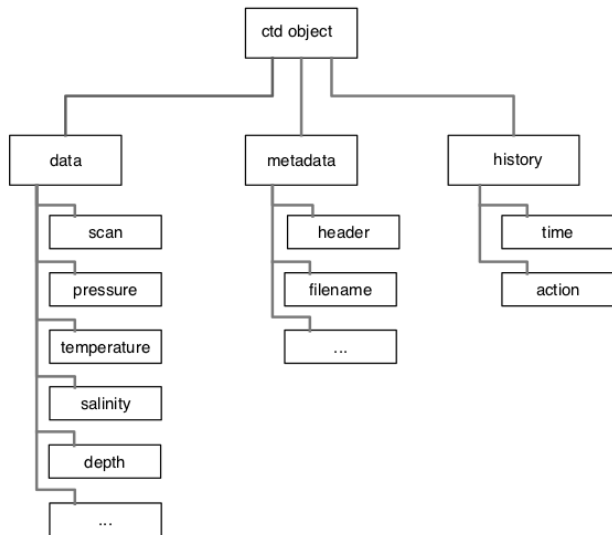
- **R in Nature:** http://www.nature.com/news/programming-tools-adventures-with-r-1.16609?WT.ec_id=NATURE-20141225
- **Hadley Wickam**
 - ▶ **Advanced R:** <http://adv-r.had.co.nz/>
 - ▶ **R packages:** <http://r-pkgs.had.co.nz/>
- **RStudio:** <https://www.rstudio.com/>

Thanks!

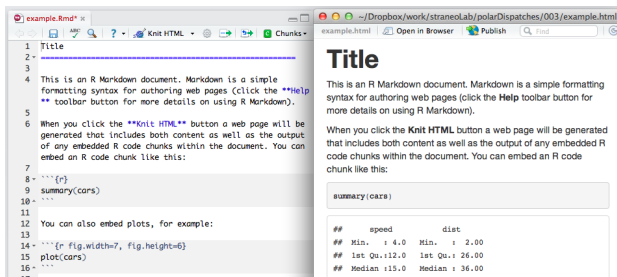
Object orientation

Example ctd object

- object classes
- generic methods
- `plot()`,
`summary()`,
`subset()`



Reproducible Research



Dynamic Documents

- Seamless integration of text, code, and figures
- Markdown or \LaTeX source (also Jupyter notebooks)
- Output to various different formats (PDF, HTML, DOCX, ...)
- This document is an example