ICES
International Council for the Exploration of the Sea
CIEM
Conseil International pour l'Exploration de la Mer

ICES Data Centre – DATRAS
1.1 DATRAS R – 2015

DATRAS Guidelines Document

R for ICES: Linking R with DATRAS Web Services

How to access DATRAS exchange format data with R through DATRAS webservices

By Scott Large, June 24, 2015

Contents

Introduction

Most of the advice that ICES provides relies heavily upon DATRAS and other ICES Data Centre data products. The user interfaces of these products are well developed and very user friendly for point-and-click data exploration and accessing a few years or surveys. Users interested in accessing the data in a more transparent, reproducible, and streamlined way can use the following R code to download batches of surveys, years, and quarters.

The following function interacts directly with the ICES DATRAS WebServices and is able to extract station data (HH), age data (CA), and length data (HL) and returns each record as an R object.

Keep in mind that the speed of this function is limited by bandwidth availability and parsing XML can be a labor intensive task, especially for long time-series. The code is written to parse the XML using parallel processing, which can significantly reduce run-time. The full time series of North Sea IBTS length based data has ~ 3 million rows and can take 30+ minutes to download and parse in parallel. Note: users of these products are subject to the ICES Data Policy and the following code is provided without warranty and may be under further development. Please contribute to the development on GitHub or contact the author directly scott.large@ices.dk.

Basic Usage

First, load the function into your local R environment.

```
getDATRAS <- function(record, survey, startyear, endyear, quarters,
                      parallel = FALSE, cores = NULL, keepTime = FALSE) {
  #   library(XML)
  #   library(doParallel)
  #   library(parallel)
  #   library(foreach)
  #   library(data.table)
  if(keepTime == TRUE) strt <- Sys.time()
  #
  seqYear <- startyear:endyear
  #
  if(!record %in% c("HL", "HH", "CA")) stop("Please specify record type:
                                              HH (haul meta-data),
                                              HL (Species length-based data),
                                              CA (species age-based data)")
  getURL <- apply(expand.grid(record, survey, seqYear, quarters),
                    1,
                    function(x) paste0("http://datras.ices.dk/WebServices/",
                                       "DATRASWebService.asmx/get", x[1],
                                       "data?survey=", x[2],
                                       "&year=", x[3],
                                       "&quarter=", x[4]))
  #
  if(parallel == TRUE) {
    if(missing("cores")) stop("Please specify the number of cores.")
    #
    unregister <- function() {
      env <- foreach:::.foreachGlobals
      rm(list=ls(name=env), pos=env)
    } # close unregister
    #
    cl <- makeCluster(cores)
    registerDoParallel(cores = cl)
    #
    getDATA <- foreach(temp = getURL,
                     .combine = function(...) rbindlist(list(...), fill = TRUE),
                     .multicombine = T,
                     .inorder = F,
                     .maxcombine = 1000,
                     .packages = c("XML", "data.table")) %dopar% {
                       data.table(t(xmlSApply(xmlRoot(xmlTreeParse(temp, isURL = T,
                                                          options = HUGE,
                                                          useInternalNodes =  T)),
                                             function(x) xmlSApply(x, xmlValue))))
                     } # close foreach %dopar%
    stopCluster(cl)
    unregister()
  } # close parallel == TRUE
  #
  if(parallel == FALSE) {
    getDATA <- foreach(temp = getURL,
                     .combine = function(...) rbindlist(list(...), fill = TRUE),
                     .multicombine=T,
                     .inorder=F,
                     .maxcombine=1000,
                     .packages = c("XML", "data.table")) %do% {
                       data.table(t(xmlSApply(xmlRoot(xmlTreeParse(temp, isURL = T,
                                                          options = HUGE,
                                                          useInternalNodes =  T)),
                                             function(x) xmlSApply(x, xmlValue))))
                     } # close foreach %do%
  } # close parallel == FALSE
  if(keepTime == TRUE) print(Sys.time()-strt)
  return(getDATA)
} # close function
```

Next, load the following packages.

```
library(XML, quietly = TRUE)
library(doParallel, quietly = TRUE)
library(parallel, quietly = TRUE)
library(foreach, quietly = TRUE)
library(data.table, quietly = TRUE)
#
```

Haul Data

To download the haul meta-data from the first and third quarters in 2010 and 2011 of the North Sea International Bottom Trawl Survey, use the following code:

```
haulData <- getDATRAS(record = "HH",
                      survey="NS-IBTS",
                      startyear = 2010,
                      endyear = 2011,
                      quarters = c(1,3),
                      parallel = TRUE,
                      cores = 4)
dim(haulData)
```

```
## [1] 1461   60
```

Length Data

To download the length-based information from the first quarter in 2000-2001 of the Baltic International Trawl Survey, use the following code:

```
lengthData <- getDATRAS(record = "HL",
                      survey="BITS",
                      startyear = 2000,
                      endyear = 2001,
                      quarters = 1,
                      parallel = TRUE,
                      cores = 4)
dim(lengthData)
```

```
## [1] 49139    27
```

Age Data

To download the age-based information from all quarters in 1999 of the French Southern Atlantic Bottom Trawl Survey, use the following code:

```
ageData <- getDATRAS(record = "CA",
                      survey="EVHOE",
                      startyear = 1999,
                      endyear = 1999,
                      quarters = c(1:4),
                      parallel = TRUE,
                      cores = 4)
dim(ageData)
```

```
## [1] 696  27
```

The function above returns *data.table* objects, which are able to efficiently handle very large amounts of data.