imarpe: un paquete para la automatización de gráficos, tablas y reportes usando R

Criscely Luján-Paredes

26 de Julio de 2017

Contents

Resumen

Este documento es una introducción al uso del paquete `imarpe`, el cuál proporciona herramientas para la elaboración de gráficas, tablas y reportes que se realizan de manera rutinaria en las investigaciones producidas por el Instituto del Mar del Perú (IMARPE). El objetivo principal de `imarpe` es que los usuarios trabajen en R sin requerir conocimientos avanzados de programación, ahorrando tiempo en el procesamiento de información.

Palabras clave: R, imarpe, automatización, reportes, figuras, tablas.

Introducción

`imarpe` es un paquete implementado en R cuya principal función es automatizar el procesamiento de información así como automatizar la elaboración de gráficas, tablas, y reportes que son necesarios en el trabajo rutinario para el personal del Instituto del Mar del Perú (IMARPE). El diseño de este paquete es flexible, permitiendo al usuario realizar cambios sobre los parámetros de los resultados ya que tendrán a su disposición el código en R que genera los productos; adicionalmente, el diseño de este paquete permite trabajar con funciones genéricas (e.g. plot, summary, print) sobre cada clase de datos, facilitando de esta manera su utilización incluso con conocimientos mínimos de R.

Las principales ventajas que proporciona `imarpe` son:

- No requiere que el usuario posea conocimientos avanzados de programación y/o R,
- Permite ahorrar tiempo en el procesamiento de la información,
- Brinda al usuario un procesamiento automatizado de la información,
- Permite que el usuario realice cambios sobre los parámetros de las salidas (tablas, figuras y reportes),
- Proporciona gráficas de buena calidad así como la capacidad de realizar análisis reproducibles.

Programación orientada a objetos

R tiene tres sistemas orientados a objetos (S3, S4 y R5), y todos trabajan con los conceptos de "clase" y "método". Una clase se define como un tipo de objeto, describiendo que propiedades posee, como funciona y como este objeto se puede relacionar con otros tipos de objetos; de esta manera, cada objeto debe poseer una clase. Por otro lado, un método se define como aquella función que está asociada a un tipo particular de objetos.

S3 implementa un estilo de programación orientada a objetos llamada "función genérica OO" (o en inglés, generic-function OO). En el funcionamiento de este sistema se realizan cálculos a través de métodos y a través de un tipo especial de función llamada "función genérica", la cual decide qué método utilizar.

El uso principal de S3 en R es a través de los métodos print, summary and plot. Estos métodos permiten tener una función genérica por ejemplo de print, que mostraría el objeto de una manera especial.

Instalación de `imarpe` Para usar `imarpe` se debe tener instalado R, y se recomienda usar la interface gráfica de R Studio. Para la instalación de todo paquete de R se require contar con internet y luego de verificar esto, desde R Studio se debe intalar y cargar el paquete devtools, de la siguiente manera:

```
install.packages("devtools")
library("devtools")
```

La instalación de `imarpe` se realizará directamente de la cuenta github del Instituto del Mar del Peru (IMARPE) https://github.com/imarpe, lugar donde se trabaja en las mejoras y actualizaciones de los paquetes en R desarrollados por el personal del IMARPE.

En esta web también se encuentra `imarpe`, y su instalación se realizará usando las siguientes líneas de código. La primera línea de código instala el paquete y debe ser corrida por única vez, sin embargo, cada vez que se haga una actualización del paquete, se deberá correr nuevamente para cargar de manera automática la nueva versión de `imarpe`. La segunda línea de código se encargará de cargar el paquete ya instalado, esta debe ser ejecutada cada vez que se quiera hacer uso de `imarpe`.

```
install_github("imarpe/imarpe")
library(imarpe)
```

Estructura de 'imarpe'

`imarpe` cuenta con el desarrollo del módulo de pesquería. Sin embargo, el objetivo es extender el desarrollo del paquete a tres módulos más, los cuales estén conformados por los modulos de biología, oceanografía y cruceros hidroacústicos.

Módulo Pesquería

Esta sección trabaja sobre información pesquera, orientada al análisis de las variables: captura y desembarque, esfuerzo y captura por unidad de esfuerzo (cpue), reproduciendo tablas, gráficas y reportes de manera automatizada. Así mismo, el Programa de Observadores a Bordo (Programa de Bitácoras de Pesca - PBP) del IMARPE también cuenta un subsección de trabajo dentro del módulo de pesquería, el cual permite obtener las principales gráficas y tablas que son requeridas en los reportes periódicos que el PBP emite.

Las principales funciones con las que cuenta `imarpe` son 4 y las explicaremos al detalle en las siguientes sub-secciones.

```
?getFishingData
?downloadDailyFishing
?getDailyReport
?getBitacoraData
?getMainResults.bitacora
```

1. Análisis de información pesquera

1.1 Uso de `getFishingData`

Esta función es usada para dos tipos de clases de datos: `fishery` y `cpue`. La primera clase de datos contiene variables pesqueras que en este caso incluye a los desembarques (landing) y al esfuerzo (effort). La segunda clase de datos incluye a la captura por unidad de esfuerzo, de ahi el nombre de la clase.

`imarpe` incluye una base de datos interna (`fisheryData`) que será usada para ejemplificar el uso de esta función. Sin embargo, toda base de datos debe poseer información sobre el tiempo (año, mes y dia), sobre la especie en estudio, sobre el tipo de flota y puertos, la información de la captura (o desembarque) así como del esfuerzo pesquero, la información de posición (latitud y longitud) es opcional.

El nombre de las columnas debe ser el mismo al de la base de datos de ejemplo, sin embargo el uso de mayúsculas y minúsculas será resuelto internamente.

En la siguiente demostración del uso de `getFishingData`, la función analiza los desembarques de la base de datos contenida en el archivo .csv (archivo delimitado por comas) para obtener como producto i) un archivo de clase data.frame con la base de datos analizada por puerto en resolución temporal diaria, ii) una lista con las principales características de la base de datos analizada, y iii) un resumen de la información por tipo de flota en resolución temporal mensual.

Adicionalmente, la ayuda de esta función puede ser visualizada para una mejor comprensión de los parámetros de la función.

```
# Cargar la base de datos
fisheryData = system.file("extdata", "fisheryData.csv", package = "imarpe")

# Usar getFishingData con la base de datos ejemplo (fisheryData) de caballa
landing  = getFishingData(file = fisheryData, type = "fisheryinfo", varType = "landing",
                          sp = "caballa")
class(landing)
```

```
## [1] "fishery"
```

```
dataBase = landing$data
head(dataBase)
```

```
##   year month day ATICO CALLAO CANCAS CHIMBOTE HUACHO ILO LA.PLANCHADA
## 1 2008   Jan   1     0      0      0        0      0   0            0
## 2 2008   Jan   2     0      0      0        0      0   0            0
```

```
## 3 2008   Jan   3     0      0      0        0      0   0           0
## 4 2008   Jan   4     0      0      0        0      0   0           0
## 5 2008   Jan   5     0      0      0        0      0   0           0
## 6 2008   Jan   6     0      0      0        0      0   0           0
##   MATARANI MORRO.SAMA PAITA PUCUSANA QUILCA SALAVERRY SAMANCO
## 1        0          0     0        0      0         0       0
## 2        0          0     0        0      0         0       0
## 3        0          0     0        0      0         0       0
## 4        0          0     0        0      0         0       0
## 5        0          0     0        0      0         0       0
## 6        0          0     0        0      0         0       0
```

```
info = landing$info
info
```

```
## $file
## [1] "C:/R/library/imarpe/extdata/fisheryData.csv"
##
## $records
## [1] 731
##
## $months
## [1] 24
##
## $years
## [1] 2
##
## $ports
## [1] 14
##
## $sp
## [1] "caballa"
##
## $varType
## [1] "landing"
##
## $efforType
## NULL
```

```
fleet = landing$fleeTable
head(fleet)
```

```
##          Artesanal Industrial Menor_escala
## Dic 2008     0.000   15244.77            0
## Ene 2009 30577.187   40024.28            0
## Feb 2009 15365.772       0.00            0
## Mar 2009  4057.597       0.00            0
## Abr 2009   455.251       0.00            0
## May 2009   182.500       0.00            0
```

```
# Para analizar un periodo de tiempo de la base de datos
landing = getFishingData(file = fisheryData, type = "fisheryinfo", varType = "landing",
                         sp = "caballa", start = "2009-04-10", end = "2009-08-30")

# Para analizar la información de un puerto específico
landing = getFishingData(file = fisheryData, type = "fisheryinfo", varType = "landing",
```

```
                          sp = "caballa", start = "2009-04-10", end = "2009-08-30",
                          port = "PAITA")
```

De manera similar al análisis de los desembarques se analizará la información del esfuerzo pesquero continuando con el uso de la función `getFishingData` así como de la base de datos interna (`fisheryData`).

Para el análisis de esta variable (esfuerzo) el tipo de información (parámetro `type`) sigue siendo `fisheryInfo` pero el tipo de variable (parámetro `varType`) será `effort`; y se debe precisar que tipo de esfuerzo pesquero se analizará en el parámetro `efforType`. En el siguiente ejemplo se usó capacidad de bodega, pero podría ser número de viajes, número de anzuelos así como número de embarcaciones (ver ayuda de la función `getFishingData` para más información).

```
# Para analizar la información de esfuerzo con la base de datos ejemplo (fisheryData)
# de caballa
effort = getFishingData(file = fisheryData, type = "fisheryinfo", varType = "effort",
                        sp = "caballa", efforType = "capacidad_bodega")
class(effort)
```

```
## [1] "fishery"
```

```
dataBase = effort$data
head(dataBase)
```

```
##   year month day ATICO CALLAO CANCAS CHIMBOTE HUACHO ILO LA.PLANCHADA
## 1 2008   Jan   1     0      0      0        0      0   0            0
## 2 2008   Jan   2     0      0      0        0      0   0            0
## 3 2008   Jan   3     0      0      0        0      0   0            0
## 4 2008   Jan   4     0      0      0        0      0   0            0
## 5 2008   Jan   5     0      0      0        0      0   0            0
## 6 2008   Jan   6     0      0      0        0      0   0            0
##   MATARANI MORRO.SAMA PAITA PUCUSANA QUILCA SALAVERRY SAMANCO
## 1        0          0     0        0      0         0       0
## 2        0          0     0        0      0         0       0
## 3        0          0     0        0      0         0       0
## 4        0          0     0        0      0         0       0
## 5        0          0     0        0      0         0       0
## 6        0          0     0        0      0         0       0
```

```
info = effort$info
info
```

```
## $file
## [1] "C:/R/library/imarpe/extdata/fisheryData.csv"
##
## $records
## [1] 731
##
## $months
## [1] 24
##
## $years
## [1] 2
##
## $ports
## [1] 14
##
## $sp
```

```
## [1] "caballa"
##
## $varType
## [1] "effort"
##
## $efforType
## [1] "capacidad_bodega"
```

```
fleet = effort$fleeTable
head(fleet)
```

```
##          Artesanal Industrial Menor_escala
## Dic 2008       0.0       1623            0
## Ene 2009    4409.0       3982            0
## Feb 2009    2945.5          0            0
## Mar 2009     993.0          0            0
## Abr 2009     256.0          0            0
## May 2009     195.0          0            0
```

Finalmente, el análisis de la captura por unidad de esfuerzo (cpue) se analizará en la siguiente sección. Aquí los parámetros `type` y `varType` deben ser iguales a `cpue`, y se debe precisar que tipo de esfuerzo se usará (en `efforType`) para calcular la cpue.

```
# Para analizar la información de cpue con la base de datos ejemplo (fisheryData)
# de caballa
cpue = getFishingData(file = fisheryData, type = "cpue", varType = "cpue",
                      sp = "caballa", efforType = "capacidad_bodega")
class(cpue)
```

```
## [1] "cpue"
```

```
dataBase = cpue$data
head(dataBase)
```

```
##   year month day cpue
## 1 2008   Jan   1    0
## 2 2008   Jan   2    0
## 3 2008   Jan   3    0
## 4 2008   Jan   4    0
## 5 2008   Jan   5    0
## 6 2008   Jan   6    0
```

```
info = cpue$info
info
```

```
## $file
## [1] "C:/R/library/imarpe/extdata/fisheryData.csv"
##
## $records
## [1] 731
##
## $months
## [1] 24
##
## $years
## [1] 2
##
## $ports
```

```
## [1] 14
##
## $sp
## [1] "caballa"
##
## $varType
## [1] "cpue"
##
## $efforType
## [1] "capacidad_bodega"
```

```
fleet = cpue$fleeTable
head(fleet)
```

```
##          Artesanal Industrial Menor_escala
## Dic 2008 0.0000000   8.851901            0
## Ene 2009 6.7330139   9.938333            0
## Feb 2009 5.2105938   0.000000            0
## Mar 2009 4.5742930   0.000000            0
## Abr 2009 2.7501865   0.000000            0
## May 2009 0.7603815   0.000000            0
```

Para los dos últimos ejemplos, de esfuerzo y cpue, también se pueden usar los parámetros `start` y `end` para hacer un análisis sobre un periodo de tiempo específico, así como el parámetro `port`, para analizar la información de un determinado puerto.

Respecto a los métodos, se construyeron cinco métodos para cada una de las clases de datos `fishery` y `cpue`. Estos son: `print`, `summary`, `print.summary`, `plot` y `report`; y las ayudas para cada método debe ser llamada por el método seguido por un punto (".") y luego la clase de datos correspondiente.

Métodos de la clase `fishery`

```
# Revisar la ayuda de los métodos de la clase fishery
?print.fishery
?summary.fishery
?print.summary.fishery
?plot.fishery
?report.fishery
```

La descripción de cada método se dará a continuación:

- print.fishery: muestra un resumen del contenido de información de la base de datos de clase fishery. Esta información incluye: (1) el nombre de la base de datos utilizada; (2) el número de registros que poseen los datos; el periodo de tiempo analizado ((3) meses y (4) años); (5) el número de puertos que poseen los datos; (6) la especie analizada en la base de datos y (7) el tipo de variable, en este caso sería `landing` o `effort`.
- summary.fishery: proporciona una lista con: (1) el tipo de variable que fue analizada; (2) una base de datos diaria por puerto de la variable analizada; (3) una base de datos total en escala temporal diaria; una base de datos por puerto; (4) una base de datos en escala temporal mensual; (5) y una base de datos en escala temporal anual.
- print.summary.fishery: este método permite visualizar los productos del método `summary.fishery`.
- plot.fishery: este método permite realizar siete tipos de gráficos con datos de la clase `fishery`.
- report.fishery: este método exporta un reporte en formato pdf de la base de datos analizada. Internamente este método distingue que tipo de variable ha sido analizada y reproduce un reporte con un

formato para los desembarques y otro para el esfuerzo.

Una breve ejemplificación del uso de los métodos de la clase `fishery` se muestra en la siguiente sección:

```
# Cargar la base de datos
fisheryData = system.file("extdata", "fisheryData.csv", package = "imarpe")

# Crear un objeto de la clase fishery usando la base de datos ejemplo (fisheryData) de #caballa
landing  = getFishingData(file = fisheryData, type = "fisheryinfo", varType = "landing",
                          sp = "caballa")

# Algunos ejemplos del uso de los métodos
print(landing)
```

```
## Datos de : 'C:/R/library/imarpe/extdata/fisheryData.csv'
## Numero de registros: 731
## Numero de meses de la data: 24
## Numero de ahnos de la data: 2
## Numero de puertos: 14
## Especie analizada: caballa
## Variable analizada: landing
```

```
sumLanding = summary(landing)
sumLanding$var
```

```
## [1] "landing"
```

```
# Visualizar sólo las primeras seis filas de cada base de datos contenida en sumLanding
head(sumLanding$portDay)
```

```
##   anho mes dia ATICO CALLAO CANCAS CHIMBOTE HUACHO ILO LA.PLANCHADA
## 1 2008 Ene   1     0      0      0        0      0   0            0
## 2 2008 Ene   2     0      0      0        0      0   0            0
## 3 2008 Ene   3     0      0      0        0      0   0            0
## 4 2008 Ene   4     0      0      0        0      0   0            0
## 5 2008 Ene   5     0      0      0        0      0   0            0
## 6 2008 Ene   6     0      0      0        0      0   0            0
##   MATARANI MORRO.SAMA PAITA PUCUSANA QUILCA SALAVERRY SAMANCO
## 1        0          0     0        0      0         0       0
## 2        0          0     0        0      0         0       0
## 3        0          0     0        0      0         0       0
## 4        0          0     0        0      0         0       0
## 5        0          0     0        0      0         0       0
## 6        0          0     0        0      0         0       0
```

```
head(sumLanding$day)
```

```
##   anho mes dia desembarque
## 1 2008 Ene   1           0
## 2 2008 Ene   2           0
## 3 2008 Ene   3           0
## 4 2008 Ene   4           0
## 5 2008 Ene   5           0
## 6 2008 Ene   6           0
```

```
head(sumLanding$port)
```

```
##           Desembarque
```

```
## Cancas         222.00
## Paita       125490.97
## Salaverry    10610.77
## Chimbote     52969.91
## Samanco        428.50
## Huacho         181.10
```

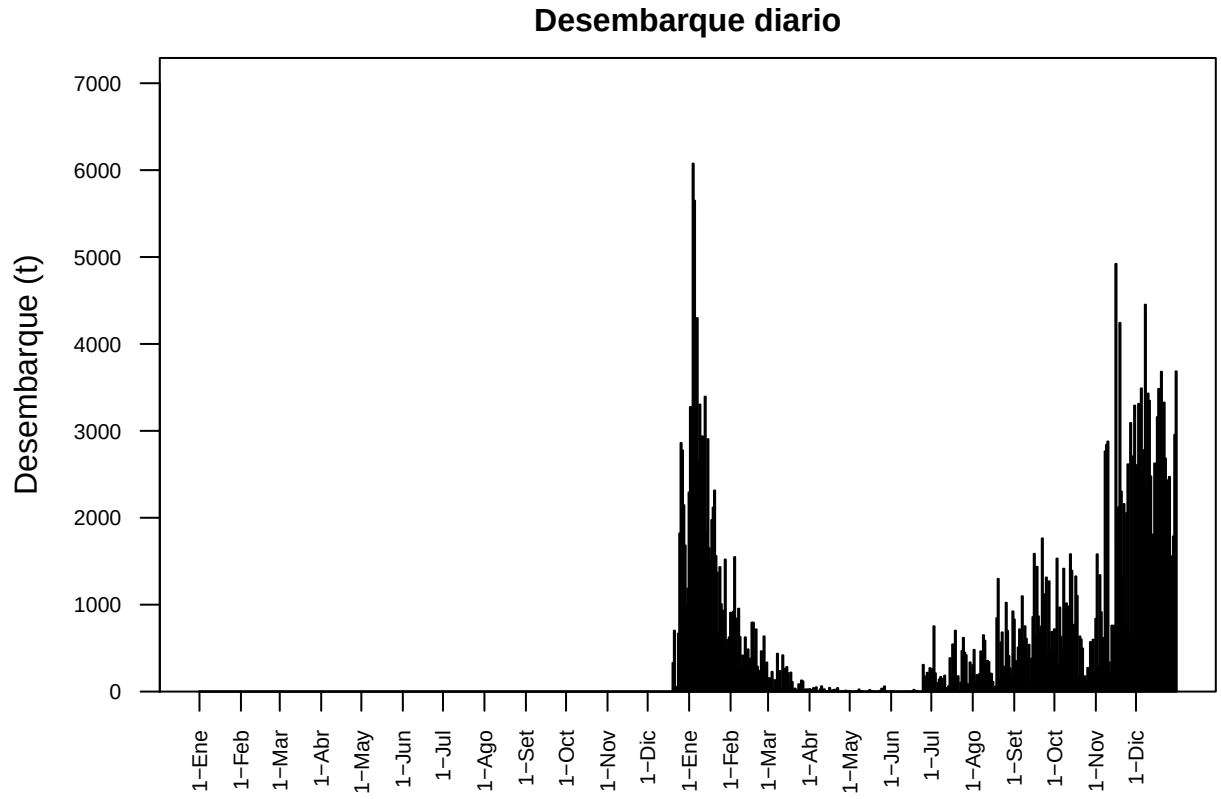
```r
head(sumLanding$months)
```

```
##     2008      2009
## Ene    0 70601.470
## Feb    0 15365.772
## Mar    0  4057.597
## Abr    0   455.251
## May    0   182.500
## Jun    0  1225.150
```

```r
head(sumLanding$years)
```

```
##      Desembarque
## 2008    15244.77
## 2009   284848.54
```

```r
# Ejemplo de gráfica diaria, mostrando en el eje x sólo los días "1"
# revisar la ayuda de plot.fishery para más información (?plot.fishery)
plot(landing, daysToPlot = "1")
```



```r
# Ver en el Anexo el reporte de desembarque
report(landing)
```

Métodos de la clase cpue

```r
# Revisar la ayuda de los métodos de la clase cpue
?print.cpue
?summary.cpue
?print.summary.cpue
?plot.cpue
?report.cpue
```

La descripción de cada método se dará a continuación:

- print.cpue: muestra un resumen del contenido de información de la base de datos de clase cpue. La información que proporciona este método es similar a la que proporciona `print.fishery`. Revisar la ayuda de `print.cpue` para mayor información.
- summary.cpue: proporciona una lista con: (1) el tipo de esfuerzo que fue usado para estimar la cpue; (2) una base de datos diaria por puerto; (3) una base de datos total en escala temporal diaria; una base de datos por puerto; (4) una base de datos en escala temporal mensual; (5) y una base de datos en escala temporal anual.
- print.summary.cpue: este método permite visualizar los productos del método `summary.cpue`.
- plot.cpue: este método permite realizar seis tipos de gráficos con datos de la clase `cpue`.
- report.cpue: este método exporta un reporte en formato pdf de la base de datos analizada.

Una breve ejemplificación del uso de los métodos de la clase `cpue` se muestra en la siguiente sección:

```r
# Cargar la base de datos
fisheryData = system.file("extdata", "fisheryData.csv", package = "imarpe")

# Crear un objeto de la clase cpue usando la base de datos ejemplo (fisheryData) de
#caballa
cpue  = getFishingData(file = fisheryData, type = "cpue", varType = "cpue",
                       sp = "caballa", efforType = "capacidad_bodega")

#Algunos ejemplos del uso de los métodos
print(cpue)
```

```
## Datos de : 'C:/R/library/imarpe/extdata/fisheryData.csv'
## Numero de registros: 731
## Numero de meses de la data: 24
## Numero de ahnos de la data: 2
## Numero de puertos: 14
## Especie analizada: caballa
## Tipo de esfuerzo: capacidad_bodega
```

```r
sumCpue = summary(cpue)
sumCpue$effort
```

```
## [1] "capacidad_bodega"
```

```r
head(sumCpue$portDay)
```

```
##   anho mes dia Atico Callao Cancas Chimbote Huacho Ilo La.planchada
## 1 2008 Ene   1     0      0      0        0      0   0            0
```

```
## 2 2008 Ene   2     0      0      0        0      0   0           0
## 3 2008 Ene   3     0      0      0        0      0   0           0
## 4 2008 Ene   4     0      0      0        0      0   0           0
## 5 2008 Ene   5     0      0      0        0      0   0           0
## 6 2008 Ene   6     0      0      0        0      0   0           0
##   Matarani Morro.sama Paita Pucusana Quilca Salaverry Samanco
## 1        0          0     0        0      0         0       0
## 2        0          0     0        0      0         0       0
## 3        0          0     0        0      0         0       0
## 4        0          0     0        0      0         0       0
## 5        0          0     0        0      0         0       0
## 6        0          0     0        0      0         0       0
```

```
head(sumCpue$day)
```

```
##   anho mes dia cpue
## 1 2008 Ene   1    0
## 2 2008 Ene   2    0
## 3 2008 Ene   3    0
## 4 2008 Ene   4    0
## 5 2008 Ene   5    0
## 6 2008 Ene   6    0
```

```
head(sumCpue$port)
```

```
##                Cpue
## Cancas    0.02607045
## Paita     2.62220346
## Salaverry 1.07908202
## Chimbote  1.89101639
## Samanco   0.02710898
## Huacho    0.02987590
```

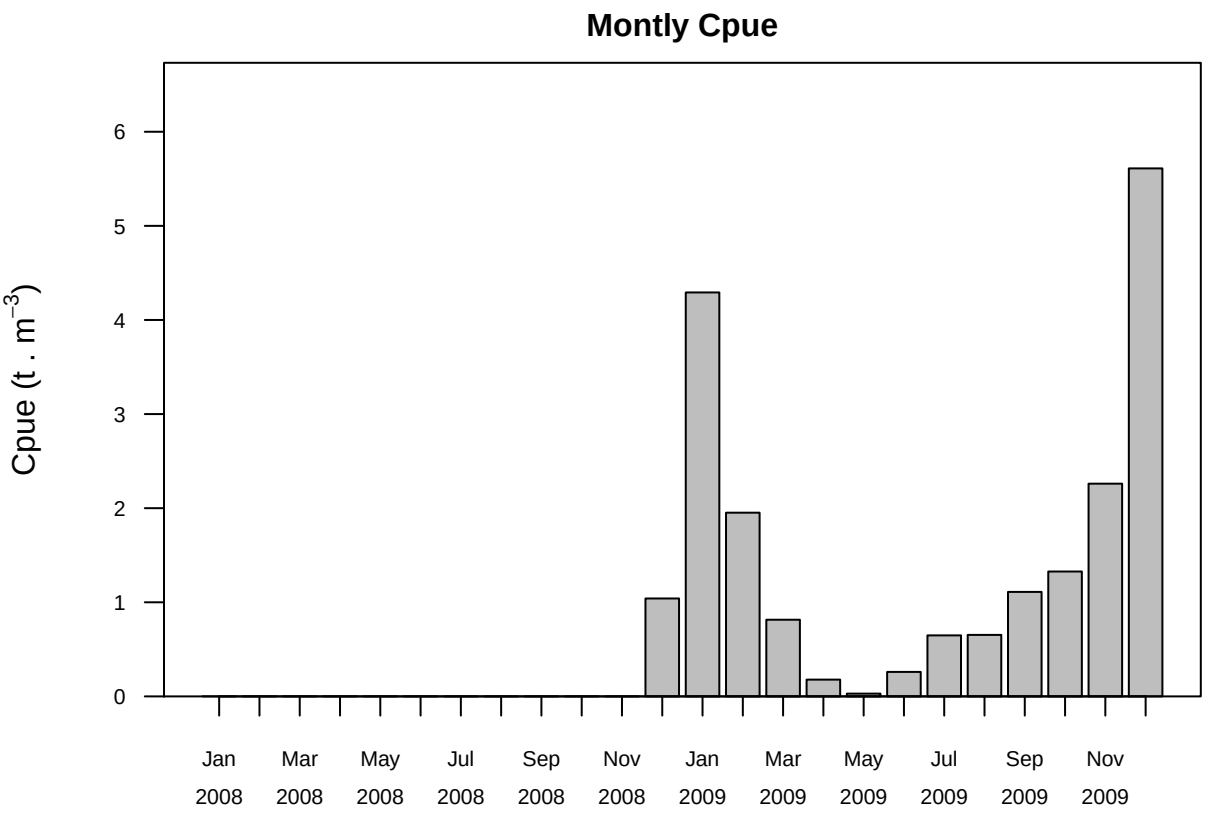
```
head(sumCpue$months)
```

```
##     2008       2009
## Ene    0 4.29262642
## Feb    0 1.95225048
## Mar    0 0.81471237
## Abr    0 0.17815640
## May    0 0.02933906
## Jun    0 0.26009025
```

```
head(sumCpue$years)
```

```
##           Cpue
## 2008 0.08812666
## 2009 1.59891293
```

```
# Ejemplo de gráfica mensual
# revisar la ayuda de plot.fishery para más información (?plot.fishery)
plot(cpue, ploType = "plotMonthly", language= "english", colBar = "gray")
```




```
# # Ver en el Anexo el reporte de cpue
report(cpue)
```

Uso de `combineFisheryVar`

`combineFisheryVar` permite combinar en un sólo objeto de clase `fishery` la información del desembarque y esfuerzo. El método `report.fishery` está disponible para ser usado en este tipo de objetos, permitiendo de esta manera la producción de un reporte unificado con la información de ambas variables pesqueras.

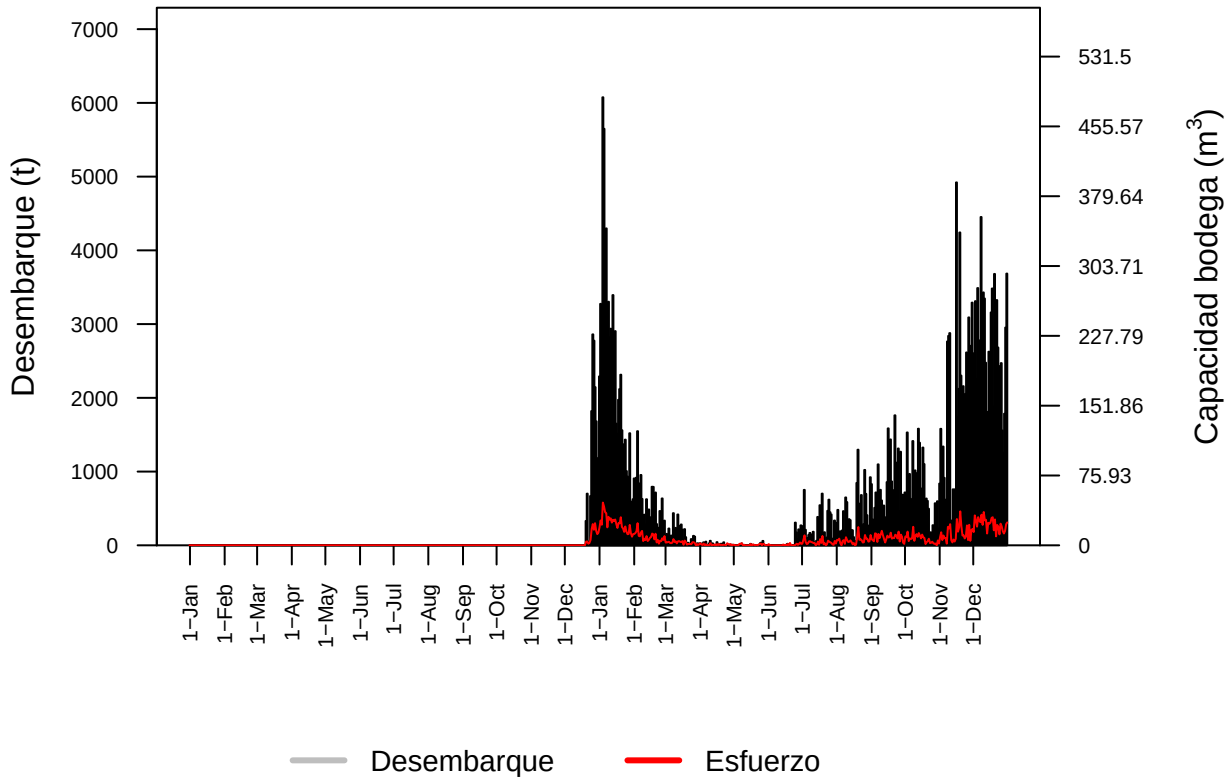
Adicionalmente, este tipo de objetos puede ser usado por el método `plot.fishery` usando `ploType = "joined"`. Esto produce una gráfica en escala temporal diaria con la información de ambas variables (desembarque y esfuerzo), el desembarque en el eje vertical izquierdo y el esfuerzo en el eje vertical derecho.

```
# Cargar la base de datos
fisheryData = system.file("extdata", "fisheryData.csv", package = "imarpe")

# Objeto de clase fishery para la variable tipo landing
landingOject = getFishingData(file = fisheryData, type = "fisheryinfo",
                              varType = "landing", sp = "caballa")

# Objeto de clase fishery para la variable tipo effort
effortObject = getFishingData(file = fisheryData, type = "fisheryinfo",
                              varType = "effort", sp = "caballa",
                              efforType = "capacidad_bodega")

# Uso de la función combineFisheryVar
fisheryVar = combineFisheryVar(landing = landingOject, effort = effortObject)
```

```r
class(fisheryVar)
```

```
## [1] "fishery"
```

```r
# Gráfica
plot(fisheryVar, ploType = "plotJoined", daysToPlot = "1")
```




```r
# # Ver en el Anexo el reporte
report(fisheryVar, type = "joined")
report(fisheryVar, type = "joined", daysToPlot = "1")
```

1.2 Uso de `downloadDailyFishing`

La función `downloadDailyFishing` fue creada con la finalidad de descargar de manera automática la información del seguimiento pesquero de anchoveta. Esta información se encuentra disponible en la página web del Instituto del Mar del Perú (IMARPE) e incluye dos variables pesqueras importantes: i) desembarque y ii) esfuerzo (por número de embarcaciones), ambas en escala temporal diaria.

`downloadDailyFishing` retorna una base de datos de clase `data.frame` con las variables (en las columnas) de: años, meses, dias, especie, tipo de flota, puerto, desembarque y embarcaciones; el cual representa al esfuerzo pesquero. Esta base de datos puede ser guardada en el directorio en el que se trabaja usando el parámetro `saveFile = TRUE`.

```r
# Revisar la ayuda de downloadDailyFishing para más información
?downloadDailyFishing
```

```r
# Obtener información del seguimiento pesquero
fishingInfo = downloadDailyFishing(startDate = "2017-4-20", endDate = "2017-6-19")
```

```r
head(fishingInfo)
```

La importancia de usar `downloadDailyFishing` es que la base de datos que se genera y se guarda en el directorio de trabajo (con extensión .csv) puede ser usado para crear un objeto de clase `fishery` con ayuda de la función `getFishingData`. Esto permitirá observar y analizar el seguimiento pesquero de la anchoveta de una manera más sencilla.

```r
fishingInfo = downloadDailyFishing(startDate = "2017-4-20", endDate = "2017-6-19")

# Utilizando el archivo de extensión .csv que se generó al crear fishingInfo se
# analizará el desembarque:
dailyLanding = getFishingData(file = "dailyFishing_2017420_2017619.csv",
                              type = "fisheryinfo", varType = "landing", sp = "anchoveta",
                              toTons = FALSE)
class(dailyLanding)
print(dailyLanding)
report(dailyLanding, daysToPlot = "1")

# Utilizando el archivo de extensión .csv que se generó al crear fishingInfo se
# analizará el esfuerzo (embarcaciones):
dailyEffort = getFishingData(file = "dataDailyReport_2017420_2017619.csv",
                             type = "fisheryinfo", varType = "effort", sp = "anchoveta",
                             efforType = "embarcaciones")
print(dailyEffort)
report(dailyEffort, daysToPlot = "1")
```

1.3 Uso de `getDailyReport`

`getDailyReport` es usada para realizar un análisis de los desembarques pesqueros de anchoveta respecto a la biomasa estimada en las cuotas de captura del recurso. Esta función tiene como finalidad descargar de manera automática los desembarques diarios de anchoveta que estan disponibles en la página web del IMARPE y en función a sus parámetros principales: fechas de análisis, información biológica de la especie, resultados del crucero para la estimación de la biomasa, la cuota oficial de biomasa; se realiza un análisis cuyos resultados serán guardados en un objeto de clase `fishingMonitoring`. Este objecto se guardará en el formato `RData` en el directorio donde se esté realizando el trabajo.

```r
# Revisar la ayuda de getDailyReport para más información
?getDailyReport
```

Método de la clase `fishingMonitoring`

La finalidad principal de `getDailyReport` es la construcción del reporte diario de la pesquería de anchoveta, por lo que el método construido para los objetos de la clase `fishingMonitoring` es `report.fishingMonitoring`.

```r
# Revisar la ayuda del método de la clase fishingMonitoring.
?report.fishingMonitoring
```

A continuación se incluirá un ejemplo del uso de `getDailyReport`. Este código servirá para comprender el formato que deben tener los datos para poder obtener un objeto de clase `fishingMonitoring`. El siguiente código debe ser ejecutado sólo si se cuenta con los datos de entrada necesarios.

```r
# Correr el siguiente código solo si se cuenta con la información necesaria.

# Lista de fechas
datesList = list(startDate = "2017-4-20",
```

```
                  endDate = "2017-6-19",
                  startExploringDate = "2017-4-22",
                  endExploringDate = "2017-4-25",
                  startSeasonDate = "2017-4-22",
                  endSeasonDate = "2017-7-31")

# Frecuencia simple por talla
simpleFreqSizes = "DataTallas_2017-I.csv"

# Archivo de crucero en formato RData
dataCruise = "cr_170304_det.RData"

#Valor de biomasa oficial (en toneladas)
officialBiomass = 7778463

# Parámetros a y b
a = 0.0034
b = 3.273

x = getDailyReport(datesList = datesList, simpleFreqSizes = simpleFreqSizes,
                   dataCruise = dataCruise, officialBiomass = officialBiomass,
                   a = a, b = b)
class(x)

report(x)
```

1.4 Uso de `getBitacoraData`

`getBitacoraData` tiene como finalidad leer la información recogida por el programa de Bitácoras de Pesca (PBP) del IMARPE para realizar diversos análisis de manera automatizada y guardarlos en un objeto de clase `bitacora`.

La información necesaria para usar `getBitacoraData` es un archivo de extensión .csv (delimitado por comas) con el formato original en la que esta base de datos se descarga del IMARSIS (Sistema de Información del IMARPE). Esta base de datos será analizada y se obtendrá un objeto (de clase `bitacora`) con dos elementos: i) la base de datos procesada y ii) las principales características de dicha base de datos.

```
# Cargar la base de datos
bitacoraData = system.file("extdata", "bitacoraData.csv", package = "imarpe")

# Usar getBitacoraData con la base de datos ejemplo (bitacoraData)
bitacoraObject = getBitacoraData(file = bitacoraData)

# Revisar la clase de bitacoraObject
class(bitacoraObject)
```

```
## [1] "bitacora"
```

```
# Obtener la base de datos filtrada y revisar las principales características de la
# base de datos
dataBase = bitacoraObject$data

info = bitacoraObject$info
info
```

```
## $file
## [1] "C:/R/library/imarpe/extdata/bitacoraData.csv"
##
## $trips
## [1] 700
##
## $ports
## [1] 22
##
## $years
## [1] 2014
##
## $months
## [1] 575
##
## $fleets
## [1] "Menor escala"      "Industrial"        "Artesanal"
## [4] "Industrial madera"
##
## $colTrip
## [1] "CODIGO_VIAJE"
##
## $colDateOut
## [1] "DIA_ARRIBO"
##
## $colDateStart
## [1] "DIA_SALIDA"
##
## $colSearchTime
## [1] "DURACION_BUSQUEDA"
##
## $colStorageCapacity
## [1] "CAPACIDAD_BODEGA_REGISTRADA"
##
## $colLat
## [1] "LATITUD_INICIAL"
##
## $colLon
## [1] "LONGITUD_INICIAL"
##
## $colHaul
## [1] "NUMERO_CALA"
##
## $colHaulTotal
## [1] "TOTAL_CALAS"
##
## $colCatchHaul
## [1] "CAPTURA_CALA"
##
## $capAnch
## [1] "CAPTURA_ANCHOVETA"
##
## $capSar
## [1] "CAPTURA_SARDINA"
```

```
##
## $capJur
## [1] "CAPTURA_JUREL"
##
## $capCab
## [1] "CAPTURA_CABALLA"
##
## $capBon
## [1] "CAPTURA_BONITO"
```

Métodos de la clase `bitacora`

```
# Revisar la ayuda de los métodos de la clase bitacora
?print.bitacora
?summary.bitacora
?print.summary.bitacora
?plotFishingPoints.bitacora
?plotFishingPresence.bitacora
?plotSpeciesComposition.bitacora
?report.bitacora
```

Los métodos construidos para la clase de datos `bitacora` son 7 y se describirán a continuación:

- print.bitacora: muestra la principal información del objeto de clase `bitacora`. Esta información incluye: (1) el nombre de la base de datos utilizada; (2) el número de viajes de pesca que posee la base de datos, (3) el número de puertos que se encuentra en la base de datos, (4) el número de meses (5) y años analizados (6), así como los tipos de flota que figuran en la base de datos, estos pueden ser: artesanal, menor escala, industrial de madera e industrial.
- summary.bitacora: método que proporciona una lista de clase `summary.bitacora` conteniendo: (1) una base de datos de viajes observados por puerto, y (2) una base de datos de lances pesqueros por latitud.
- print.summary.bitacora: este método permite visualizar los productos del método `summary.bitacora`.
- plotFishingPoints.bitacora: este método toma un objeto de clase `bitacora` y grafica en un mapa los puntos de pesca de la especie precisada en `dataType`.
- plotFishingPresence.bitacora: este método toma un objeto de clase `bitacora` y grafica en un mapa la incidencia de especies en la captura, ya sea por grupo taxonómico o de manera general.
- plotSpeciesComposition.bitacora: este método toma un objeto de clase `bitacora` y grafica en un diagrama de pie la composición de las capturas.
- report.bitacora: este método exporta un reporte en formato pdf un conjunto de análisis realizados con la base cargada del PBP.

Una breve ejemplificación del uso de los métodos de la clase `bitacora` se muestra en la siguiente sección.

```
# Cargar la base de datos
bitacoraData = system.file("extdata", "bitacoraData.csv", package = "imarpe")

# Crear un objeto de la clase bitacora usando la base de datos de ejemplo (bitacoraData)
bitacoraObject = getBitacoraData(file = bitacoraData)

# Algunos ejemplos del uso de los métodos
print(bitacoraObject)
```

```
## Datos de : 'C:/R/library/imarpe/extdata/bitacoraData.csv'
## Numero de viajes: 700
```

```
## Numero de puertos: 22
## Numero de meses de la data: 575
## Numero de ahnos de la data: 2014
## Tipos de flota:  Menor escala  Industrial  Artesanal  Industrial madera
```

```
sumBitacora = summary(bitacoraObject)
class(sumBitacora)
```

```
## [1] "summary.bitacora"
```

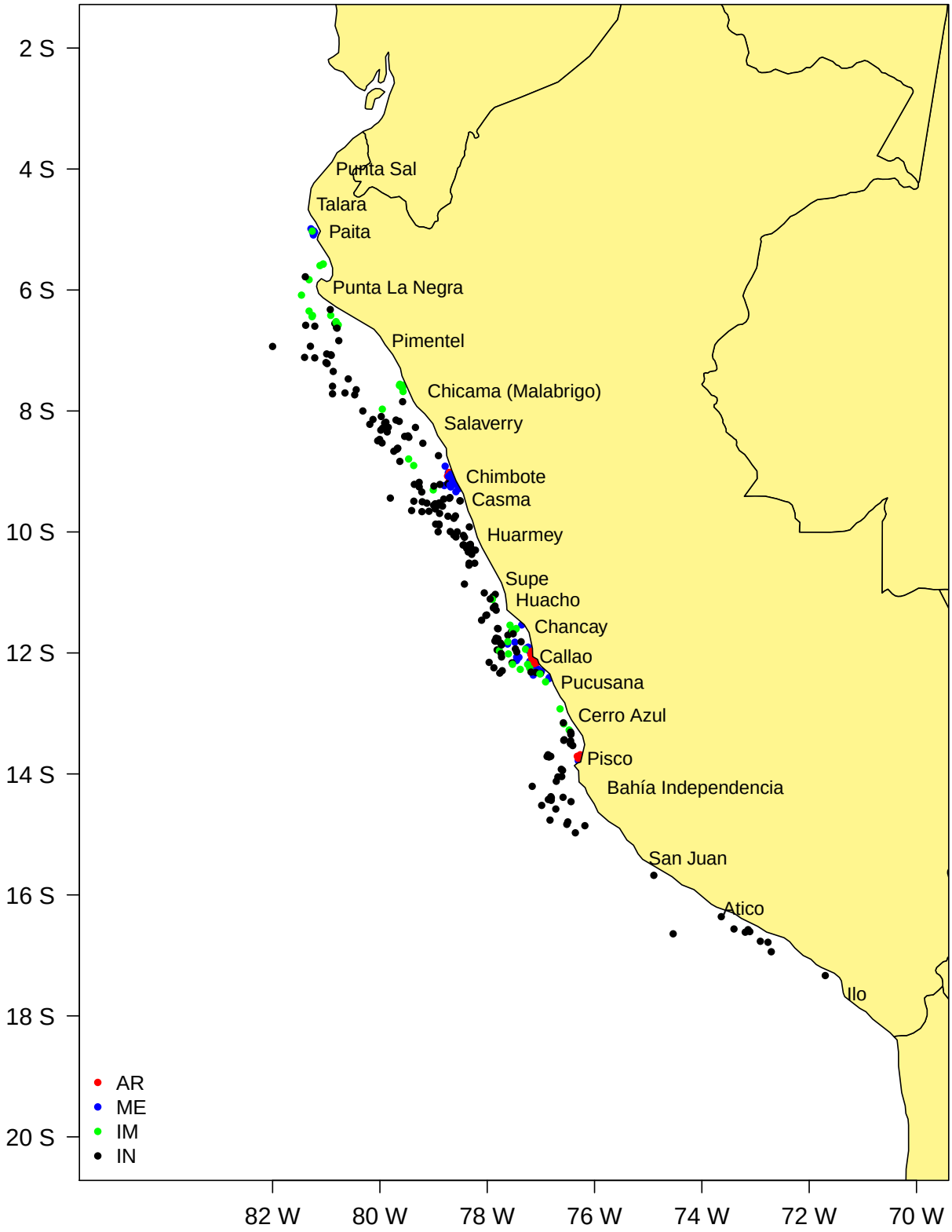
```
head(sumBitacora$observedTrip)
```

```
##                Puerto Artesanal Menor escala Industrial madera Industrial
## 1               Paita        12           17                 8         20
## 2        Las Delicias         0            0                 0          2
## 3          Parachique         1            2                 0          2
## 4             Bayóvar         3            5                 1          3
## 5         Puerto Rico         6            2                 0          4
## 6 Chicama (Malabrigo)        20           21                13         38
##   Total
## 1    57
## 2     2
## 3     5
## 4    12
## 5    12
## 6    92
```

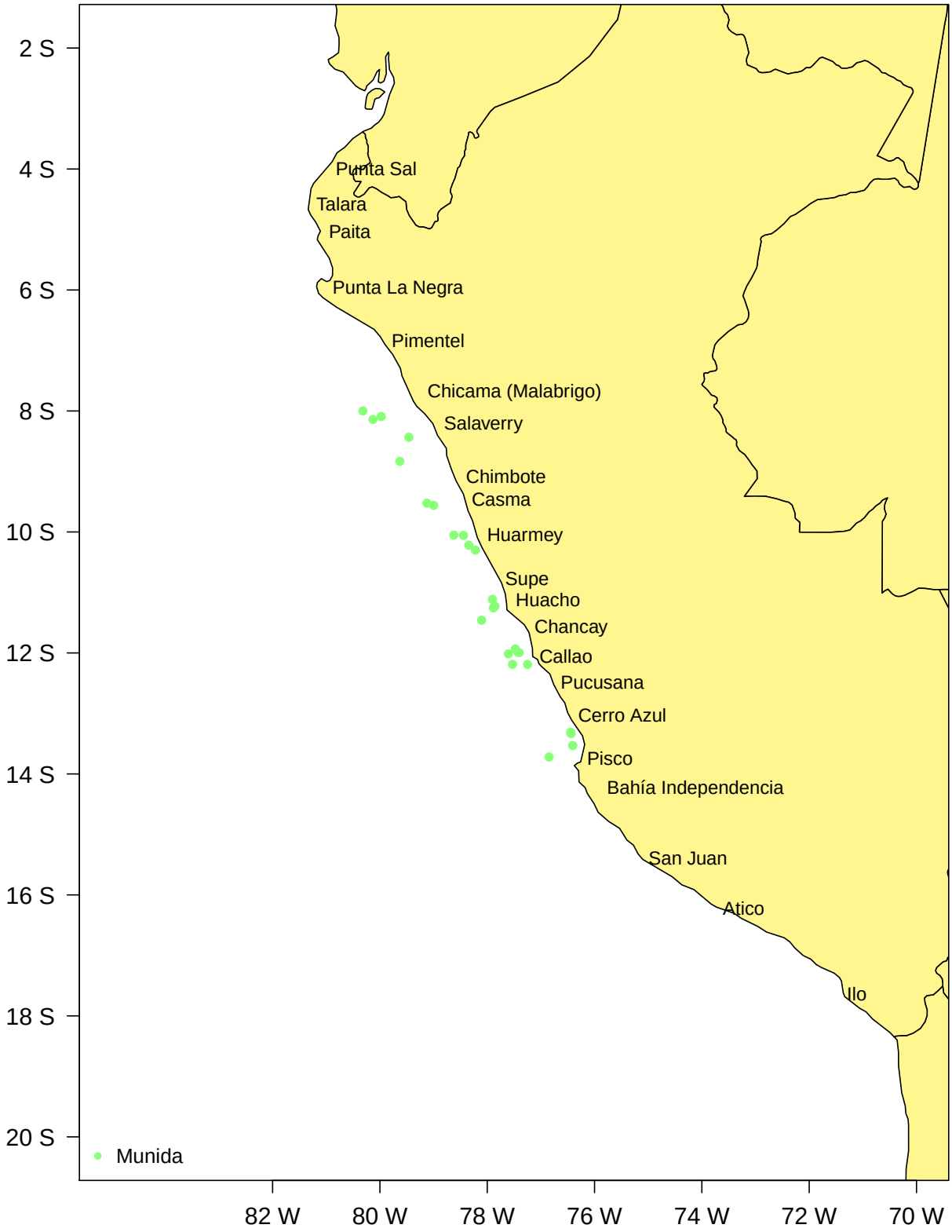
```
head(sumBitacora$fishingHaul)
```

```
##   Latitud Artesanal Menor escala Industrial madera Industrial Total
## 1      3S         0            0                 0          0     0
## 2      4S         0           41                 1          1    43
## 3      5S         2           35                11          2    50
## 4      6S         0            0                 9         11    20
## 5      7S         0            0                10         24    34
## 6      8S         2            4                 3         43    52
```

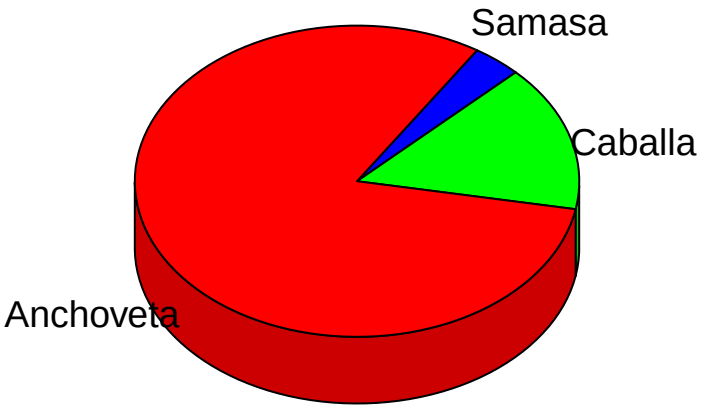
```
plotFishingPoints(x = bitacoraObject, language = "spanish", dataType = "dataAnch")
```



```
plotFishingPresence(x = bitacoraObject, byGroup = TRUE, group = "crustaceo")
```



```
plotSpeciesComposition(x = bitacoraObject, minPercentage = 0.7)
```




```
# # Ver en el Anexo el reporte de bitacora
report(bitacoraObject)
```

Uso de `getMainResults`

Los resultados que se generan usando la base de datos del PBP es muy amplia, incluyendo una serie de cuadros de datos de diversos tipos. `getMainResults` brinda una herramienta sencilla de obtener sólo los resultados que le interesan al usuario de una manera muy sencilla. Para esto, es necesario tener un objeto de clase `bitacora`, indicar el idioma en el que se desean visualizar los resultados (español es el idioma por defecto), precisar la especie en estudio, y seleccionar los resultados que se querrar obtener poniendo `TRUE` en cada uno de ellos.

A continuación se muestra un breve ejemplo del uso de `getMainResults`:

```
# Cargar la base de datos
bitacoraData = system.file("extdata", "bitacoraData.csv", package = "imarpe")

# Crear un objeto de la clase bitacora usando la base de datos de ejemplo (bitacoraData)
bitacoraObject = getBitacoraData(file = bitacoraData)

# Uso de getMainResults.bitacora
mainBitacoraData = getMainResults.bitacora(object = bitacoraObject, language = "spanish",
                                           specie = "anchoveta",
                                           observedTrip = TRUE)
# Viajes observados
observedTrip = mainBitacoraData$observedTrip

head(observedTrip)
```

```
##                Puerto Artesanal Menor escala Industrial madera Industrial
## 1               Paita        12           17                 8         20
## 2        Las Delicias         0            0                 0          2
## 3          Parachique         1            2                 0          2
## 4             Bayóvar         3            5                 1          3
## 5         Puerto Rico         6            2                 0          4
## 6 Chicama (Malabrigo)        20           21                13         38
##   Total
## 1    57
## 2     2
## 3     5
## 4    12
## 5    12
## 6    92
```

Uso de `effortSpeciesData`

El esfuerzo pesquero es una variable importante en el seguimiento de la pesquería de un recurso, y ésta también puede ser analizada con la información que el PBP recoge de manera rutinaria. Por tanto, el uso de `effortSpeciesData` tiene la finalidad recibir un objeto de clase `bitacora` y realizar un filtro de los datos para una especie, clasificando en función a ella las temporadas de pesca para cada región (norte-centro, sur, o total). El producto generado de dicha función es una data.frame con información que será util para poder estimar el esfuerzo pesquero.

Luego de obtener la base de datos filtrada se puede hacer uso de `getEffort`. Esta función permite estimar el esfuerzo en función a cuatro tipos de variables: (1) duración del viaje ("travelTime"), (2) al número de calas ("haulTotal"), (3) capacidad de bodega ("storageCapacity""), y (4) tiempo de búsqueda ("searchTime"); sobre el tiempo (años, meses, dias o temporadas) o sobre los puertos. Adicionalmente, el esfuerzo puede ser estimado para toda la base de datos o haciendo un filtro de algún tipo de flota en especial.

El resultado de `getEffort` puede ser utilizado para producir una gráfica típica de los informes de PBP, la cual relaciona los series de esfuerzo pesquero en los ejes Y (izquiero y derecho) por cada puerto en el eje X.

```
# Cargar la base de datos
bitacoraData = system.file("extdata", "bitacoraData.csv", package = "imarpe")

# Crear un objeto de la clase bitacora usando la base de datos de ejemplo (bitacoraData)
bitacoraObject = getBitacoraData(file = bitacoraData)

# Uso de getMainResults.bitacora
mainBitacoraData = getMainResults.bitacora(object = bitacoraObject, language = "spanish",
                                           specie = "anchoveta", effortData = TRUE)

# Data de esfuerzo
effortData = mainBitacoraData$effortData

# Filtro de datos: datos de anchoveta para la región norte-centro
effortNC = effortSpeciesData.bitacora(data = effortData, species = "anchoveta", region = "norte-centro")

# Obtener el esfuerzo por duración  de viaje
effortNC_tt = getEffort(data = effortNC, efforType = "travelTime", effortBy = "port")
effortNC_tt
```

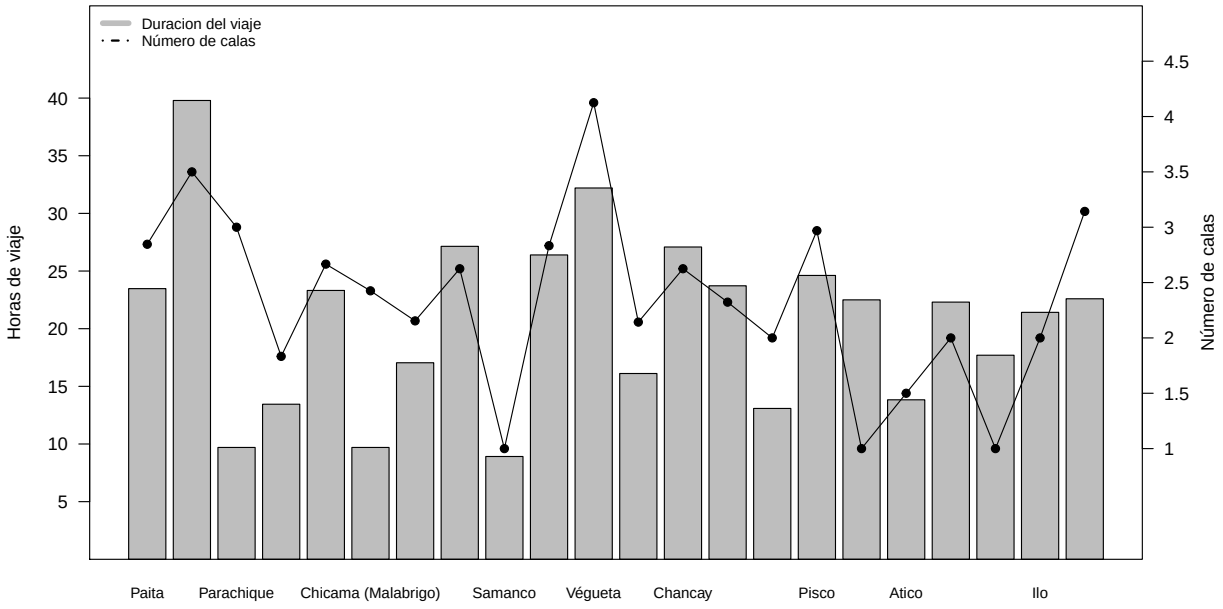
```
##               Paita        Las Delicias          Parachique
##           23.475000           39.791667            9.700000
##             Bayóvar         Puerto Rico Chicama (Malabrigo)
##           13.450000           23.316667            9.700355
##            Coishco            Chimbote             Samanco
##           17.046154           27.141927            8.916667
##               Supe              Végueta              Huacho
##           26.395833           32.202083           16.114286
##            Chancay              Callao       Tambo de Mora
##           27.085417           23.720513           13.083333
##               Pisco            San Juan                Atico
##           24.618750           22.500000           13.833333
##        La Planchada Matarani (Mollendo)                 Ilo
##           22.305556           17.700000           21.416667
##          Morro Sama
##           22.595238
```

```
# Obtener el esfuerzo por número de calas
effortNC_hn = getEffort(data = effortNC, efforType = "haulTotal", effortBy = "port")
effortNC_hn
```

```
##               Paita        Las Delicias          Parachique
##            2.846154            3.500000            3.000000
##             Bayóvar         Puerto Rico Chicama (Malabrigo)
##            1.833333            2.666667            2.425532
##            Coishco            Chimbote             Samanco
##            2.153846            2.625000            1.000000
##               Supe              Végueta              Huacho
##            2.833333            4.125000            2.142857
##            Chancay              Callao       Tambo de Mora
##            2.625000            2.323077            2.000000
##               Pisco            San Juan                Atico
##            2.968750            1.000000            1.500000
```

```
##        La Planchada Matarani (Mollendo)                 Ilo
##                2.000000         1.000000            2.000000
##              Morro Sama
##                3.142857
```

```
# Gráfico de esfuerzo
plotEffort(effort1 = effortNC_tt, effort2 = effortNC_hn, labAxis2 = "Horas de viaje",
           labAxis4 = "N\u{00FA}mero de calas", colBar="gray", colLine = "black",
           legend = c("Duracion del viaje", "N\u{00FA}mero de calas"))
```



Adicionalmente, `imarpe` cuenta con una función llamada `getCpue_relative` la cual permitirá obtener la captura por unidad de esfuerzo (cpue) removiendo el efecto de la capacidad de bodega. Para obtener la cpue relativa se debe contar con la base de dato generada por `effortSpeciesData.bitacora`. Ver el siguiente ejemplo:

```
effortNC = effortSpeciesData.bitacora(data = effortData, species = "anchoveta", region = "norte-centro")

# Obtener la cpue relativa
getCpue_relative(data = effortNC, efforType = "travelTime", cpueBy = "port")
```

```
##              Paita        Las Delicias          Parachique
##         0.023421093         0.003622917         0.025067190
##             Bayóvar        Puerto Rico Chicama (Malabrigo)
##         0.023703140         0.011533388         0.019375687
##             Coishco            Chimbote             Samanco
##         0.027304126         0.012852410         0.089405607
##               Supe             Végueta              Huacho
##         0.014946511         0.006633507         0.017947560
##             Chancay              Callao       Tambo de Mora
##         0.015744638         0.019220422         0.013292717
##              Pisco            San Juan               Atico
##         0.011008988         0.002057232         0.019188203
##        La Planchada Matarani (Mollendo)                 Ilo
##         0.010127146         0.013635682         0.021223405
```

```
##          Morro Sama
##         0.012046036
```

```
getCpue_relative(data = effortNC, efforType = "travelTime", cpueBy = "time", timeBy = "months")
```

```
##               1          2          3          4          5           6
## 2014 0.01179402 0.01130165 0.02365051 0.03320912 0.03086383 0.005509881
##               7          8          9         10         11          12
## 2014 0.01220855 0.02244904 0.07763895 0.03741497 0.0447221 0.05140036
```