

Package ‘RstoxData’

July 17, 2021

Version 1.2.1

Date 2021-07-13

Title Tools to Read and Manipulate Fisheries Data

Depends R (>= 3.6)

Description Set of tools to read and manipulate various data formats for fisheries. Mainly catered towards scientific trawl survey sampling ('biotic') data, acoustic trawl data, and commercial fishing catch ('landings') data. Among the supported data formats are the data products from the Norwegian Institute Marine Research ('IMR') and the International Council for the Exploration of the Sea (ICES).

URL <https://github.com/StoXProject/RstoxData>

BugReports <https://github.com/StoXProject/RstoxData/issues>

License LGPL-3

LazyData true

Encoding UTF-8

Imports Rcpp (>= 1.0.0), data.table (>= 1.12.6), xml2 (>= 1.2.2), xslt (>= 1.4), units (>= 0.7), stringi (>= 1.4.3)

Suggests testthat

LinkingTo Rcpp

RoxygenNote 7.1.1

NeedsCompilation yes

Author Ibrahim Umar [cre, aut],
Sindre Vatnehol [aut],
Arne Johannes Holmin [aut],
Edvin Fuglebakk [aut],
Espen Johnsen [aut],
Norwegian Institute of Marine Research [cph, fnd]

Maintainer Ibrahim Umar <ibrahim.umar@hi.no>

Repository CRAN

Date/Publication 2021-07-17 10:10:05 UTC

R topics documented:

| | |
|------------------------------------|----|
| AcousticData | 3 |
| AddToStoxBiotic | 4 |
| backwardCompatibility | 4 |
| BioticData | 5 |
| ConvertAcoustic | 5 |
| ConvertBiotic | 6 |
| ConvertStoxAcoustic | 7 |
| ConvertStoxBiotic | 8 |
| DataTypes | 9 |
| DefineTranslation | 9 |
| FilterAcoustic | 10 |
| FilterBiotic | 11 |
| filterData | 11 |
| FilterLanding | 12 |
| FilterStoxAcoustic | 13 |
| FilterStoxBiotic | 13 |
| FilterStoxLanding | 14 |
| generalSamplingHierarhcy | 14 |
| general_arguments | 15 |
| getNumberOfCores | 15 |
| getRstoxDataDefinitions | 16 |
| getStoxKeys | 17 |
| ICESAcoustic | 17 |
| ICESAcousticData | 18 |
| ICESBiotic | 18 |
| ICESBioticData | 19 |
| ICESDatras | 19 |
| ICESDatrasData | 20 |
| is.LandingData | 20 |
| is.StoxLandingData | 21 |
| LandingData | 21 |
| lapplyOnCores | 22 |
| mapplyOnCores | 22 |
| mergeByIntersect | 23 |
| mergeByStoxKeys | 23 |
| mergeDataTables | 24 |
| MergeStoxAcoustic | 24 |
| MergeStoxAcousticData | 25 |
| MergeStoxBiotic | 25 |
| MergeStoxBioticData | 26 |
| ModelData | 26 |
| parseInterCatch | 27 |
| ProcessData | 28 |
| processPropertyFormats | 28 |
| ReadAcoustic | 28 |
| ReadBiotic | 29 |

| | |
|----------------------------------|-----------|
| readErsFile | 30 |
| ReadLanding | 30 |
| readLssFile | 31 |
| readXmlFile | 32 |
| RedefineStoxBiotic | 33 |
| RstoxData | 33 |
| setorderv_numeric | 34 |
| setRstoxPrecisionLevel | 35 |
| StoxAcoustic | 35 |
| StoxAcousticData | 36 |
| StoxBiotic | 36 |
| StoxBioticData | 37 |
| StoxBioticFormat | 37 |
| stoxBioticObject | 39 |
| stoxFunctionAttributes | 39 |
| StoxLanding | 40 |
| StoxLandingData | 41 |
| TranslateAcoustic | 42 |
| TranslateBiotic | 42 |
| TranslateICESAcoustic | 43 |
| TranslateICESBiotic | 43 |
| TranslateLanding | 44 |
| TranslateStoxAcoustic | 44 |
| TranslateStoxBiotic | 45 |
| TranslateStoxLanding | 45 |
| Translation | 46 |
| WriteICESAcoustic | 46 |
| WriteICESAcousticData | 47 |
| WriteICESBiotic | 47 |
| WriteICESBioticData | 48 |
| WriteICESDatras | 48 |
| WriteICESDatrasData | 49 |
| xsdObjects | 49 |
| Index | 50 |

AcousticData

StoX data type AcousticData

Description

Biotic data read from biotic xml files.

Details

This StoX data type is produced by [ReadAcoustic](#), and contains one list per input acoustic file holding the tables read from each file, added a table named "metadata" holding the input file path and format. Currently supported are NMDEchosounder1 (<https://www.imr.no/formats/nmdechounder/v1/>), and ICESAcoustic (<https://ices.dk/data/data-portals/Pages/acoustic.aspx>, click on "Acoustic data format" to download the format description).

See Also

[DataTypes](#) for a list of all StoX data types produced by [RstoxData](#)

| | |
|-----------------|--|
| AddToStoxBiotic | <i>Add variables to StoxBioticData from BioticData</i> |
|-----------------|--|

Description

Add variables to StoxBioticData from BioticData

Usage

```
AddToStoxBiotic(StoxBioticData, BioticData, VariableNames = character())
```

Arguments

StoxBioticData [StoxBioticData](#).

BioticData [BioticData](#).

VariableNames A character vector with names of the variables to add from the BioticData.

Value

An object of StoX data type [StoxBioticData](#).

| | |
|-----------------------|--|
| backwardCompatibility | <i>Backward compabitibility actions:</i> |
|-----------------------|--|

Description

Backward compabitibility actions:

Usage

```
backwardCompatibility
```

Format

An object of class list of length 5.

| | |
|------------|----------------------------------|
| BioticData | <i>StoX data type BioticData</i> |
|------------|----------------------------------|

Description

Biotic data read from biotic xml files.

Details

This StoX data type is produced by [ReadBiotic](#), and contains one list per input biotic file holding the tables read from each file, added a table named "metadata" holding the input file path and format. Currently supported are NMDBiotic1.4 (<https://www.imr.no/formats/nmdbiotic/v1.4/>), NMDBiotic3.0 (<https://www.imr.no/formats/nmdbiotic/v3/>), and ICESBiotic (<https://ices.dk/data/data-portals/Pages/acoustic.aspx>, click on "Acoustic data format" to download the format description).

See Also

[DataTypes](#) for a list of all StoX data types produced by [RstoxData](#)

| | |
|-----------------|-----------------------------|
| ConvertAcoustic | <i>Convert AcousticData</i> |
|-----------------|-----------------------------|

Description

This function converts one or more columns of [AcousticData](#) by the function given by `ConversionFunction`.

Usage

```
ConvertAcoustic(
  AcousticData,
  TargetVariable = character(),
  ConversionFunction = c("Constant", "Addition", "Scaling", "AdditionAndScaling"),
  GruopingVariables = character(),
  Conversion = data.table::data.table()
)
```

Arguments

AcousticData An input of [ModelData](#) object
 The parameters of the `ConversionFunction` are "Constant" for `ConversionFunction` "Constant", "Addition" for `ConversionFunction` "Addition", "Scaling" for `ConversionFunction` "Scaling", and "Addition" and "Scaling" for `ConversionFunction` "AdditionAndScaling".

TargetVariable The variable to modify.

ConversionFunction

Character: The function to convert by, one of "Constant", for replacing the specified columns by a constant value; "Addition", for adding to the columns; "Scaling", for multiplying by a factor; and "AdditionAndScaling", for both adding and multiplying.

GroupingVariables

A vector of variables to specify in the Conversion. The parameters specified in the table are valid for the combination of the GroupingVariables in the data.

Conversion

A table of the GroupingVariables and the columns "TargetVariable", "SourceVariable" and the parameters of the ConversionFunction (see details).

The parameters of the ConversionFunction are "Constant" for ConversionFunction "Constant", "Addition" for ConversionFunction "Addition", "Scaling" for ConversionFunction "Scaling", and "Addition" and "Scaling" for ConversionFunction "AdditionAndScaling".

Value

A [AcousticData](#) object.

 ConvertBiotic

Convert BioticData

Description

This function converts one or more columns of [BioticData](#) by the function given by ConversionFunction.

Usage

```
ConvertBiotic(
  BioticData,
  TargetVariable = character(),
  ConversionFunction = c("Constant", "Addition", "Scaling", "AdditionAndScaling"),
  GroupingVariables = character(),
  Conversion = data.table::data.table()
)
```

Arguments

BioticData An input of [ModelData](#) object
 The parameters of the ConversionFunction are "Constant" for ConversionFunction "Constant", "Addition" for ConversionFunction "Addition", "Scaling" for ConversionFunction "Scaling", and "Addition" and "Scaling" for ConversionFunction "AdditionAndScaling".

TargetVariable The variable to modify.

| | |
|--------------------|---|
| ConversionFunction | Character: The function to convert by, one of "Constant", for replacing the specified columns by a constant value; "Addition", for adding to the columns; "Scaling", for multiplying by a factor; and "AdditionAndScaling", for both adding and multiplying. |
| GroupingVariables | A vector of variables to specify in the Conversion. The parameters specified in the table are valid for the combination of the GroupingVariables in the data. |
| Conversion | A table of the GroupingVariables and the columns "TargetVariable", "SourceVariable" and the parameters of the ConversionFunction (see details). The parameters of the ConversionFunction are "Constant" for ConversionFunction "Constant", "Addition" for ConversionFunction "Addition", "Scaling" for ConversionFunction "Scaling", and "Addition" and "Scaling" for ConversionFunction "AdditionAndScaling". |

Value

A [BioticData](#) object.

ConvertStoxAcoustic *Convert StoxAcousticData*

Description

This function converts one or more columns of [StoxAcousticData](#) by the function given by ConversionFunction.

Usage

```
ConvertStoxAcoustic(
  StoxAcousticData,
  TargetVariable = character(),
  ConversionFunction = c("Constant", "Addition", "Scaling", "AdditionAndScaling"),
  GroupingVariables = character(),
  Conversion = data.table::data.table()
)
```

Arguments

StoxAcousticData An input of [ModelData](#) object
The parameters of the ConversionFunction are "Constant" for ConversionFunction "Constant", "Addition" for ConversionFunction "Addition", "Scaling" for ConversionFunction "Scaling", and "Addition" and "Scaling" for ConversionFunction "AdditionAndScaling".

TargetVariable The variable to modify.

ConversionFunction

Character: The function to convert by, one of "Constant", for replacing the specified columns by a constant value; "Addition", for adding to the columns; "Scaling", for multiplying by a factor; and "AdditionAndScaling", for both adding and multiplying.

GroupingVariables

A vector of variables to specify in the Conversion. The parameters specified in the table are valid for the combination of the GroupingVariables in the data.

Conversion

A table of the GroupingVariables and the columns "TargetVariable", "SourceVariable" and the parameters of the ConversionFunction (see details).

The parameters of the ConversionFunction are "Constant" for ConversionFunction "Constant", "Addition" for ConversionFunction "Addition", "Scaling" for ConversionFunction "Scaling", and "Addition" and "Scaling" for ConversionFunction "AdditionAndScaling".

Value

A [StoxAcousticData](#) object.

| | |
|-------------------|-------------------------------|
| ConvertStoxBiotic | <i>Convert StoxBioticData</i> |
|-------------------|-------------------------------|

Description

This function converts one or more columns of [StoxBioticData](#) by the function given by ConversionFunction.

Usage

```
ConvertStoxBiotic(
  StoxBioticData,
  TargetVariable = character(),
  ConversionFunction = c("Constant", "Addition", "Scaling", "AdditionAndScaling"),
  GroupingVariables = character(),
  Conversion = data.table::data.table()
)
```

Arguments

StoxBioticData An input of [ModelData](#) object

TargetVariable The variable to modify.

ConversionFunction

Character: The function to convert by, one of "Constant", for replacing the specified columns by a constant value; "Addition", for adding to the columns; "Scaling", for multiplying by a factor; and "AdditionAndScaling", for both adding and multiplying.

GroupingVariables

A vector of variables to specify in the Conversion. The parameters specified in the table are valid for the combination of the GroupingVariables in the data.

Conversion

A table of the GroupingVariables and the columns "TargetVariable", "SourceVariable" and the parameters of the ConversionFunction (see details).

The parameters of the ConversionFunction are "Constant" for ConversionFunction "Constant", "Addition" for ConversionFunction "Addition", "Scaling" for ConversionFunction "Scaling", and "Addition" and "Scaling" for ConversionFunction "AdditionAndScaling".

Value

A [StoxBioticData](#) object.

 DataTypes

StoX data types of the RstoxData package

Description

StoX data types are the data types used to transfer data and information between processes in a StoX estimation model. The data types are divided into two types, the [ModelData](#) and [ProcessData](#).

 DefineTranslation

Define translation

Description

This function defines the translation table used as input to [TranslateStoxBiotic](#) and similar functions to translate values of one or more columns to new values given by a table or read from a CSV file.

Usage

```
DefineTranslation(
  processData,
  UseProcessData = FALSE,
  DefinitionMethod = c("ResourceFile", "TranslationTable"),
  TranslationTable = data.table::data.table(),
  Conditional = FALSE,
  FileName = character()
)
```

Arguments

| | |
|------------------|---|
| processData | The current data produced by a previous instance of the function. |
| UseProcessData | Logical: If TRUE use the existing function output in the process. |
| DefinitionMethod | Character: A string naming the method to use, one of "TranslationTable" for defining the TranslationTable, and "ResourceFile" for reading the table from the file given by FileName. |
| TranslationTable | A table of the columns VariableName, representing the variable to translate; Value, giving the values to translate; and NewValue, giving the values to translate to. Use NA in the Value column to translate missing values (shown as "-" in View output in the StoX GUI, and usually as empty cell in excel). In the current version NAs cannot be mixed with non-NAs in the Value column. Please use a separate DefineTranslation & Translate process to translate NAs. |
| Conditional | Logical: If TRUE the columns ConditionalVariableName and ConditionalValue are expected in the TranslationTable. These define a variable interacting with the VariableName and Value, so that VariableName is changed from Value to NewValue only when ConditionalVariableName has the value given by ConditionalValue. Note that ConditionalVariableName must exist in the same table as VariableName. |
| FileName | The csv file holding a table with the three variables listed for TranslationTable. |

Value

A [Translation](#) object.

| | |
|----------------|-----------------------------------|
| FilterAcoustic | <i>Filter (raw) Acoustic data</i> |
|----------------|-----------------------------------|

Description

Filters [AcousticData](#).

Usage

```
FilterAcoustic(AcousticData, FilterExpression, FilterUpwards = FALSE)
```

Arguments

| | |
|------------------|--|
| AcousticData | Input AcousticData data. |
| FilterExpression | Filter expression in list of strings. The name of the list and structures should be identical to the names of the input data list. |
| FilterUpwards | Whether the filter action will propagate in the upwards direction. Default to FALSE. |

Value

An object of filtered data in the same format as the input data.

| | |
|--------------|---------------------------------|
| FilterBiotic | <i>Filter (raw) Biotic data</i> |
|--------------|---------------------------------|

Description

Filters [BioticData](#).

Usage

```
FilterBiotic(BioticData, FilterExpression, FilterUpwards = FALSE)
```

Arguments

BioticData Input [BioticData](#) data.

FilterExpression

Filter expression given as a list of strings. The name of the list and structures should be identical to the names of the input data list. To extract or exclude missing values (NAs) use the operator `%in%` or the special operator `%notin%`, which is defined in [RstoxData](#).

FilterUpwards Whether the filter action will propagate in the upwards direction. Default to FALSE. Use this option with caution, particularly for swept-area survey estimates, where setting `FilterUpwards` to TRUE could affect the estimated mean density.

Value

An object of filtered data in the same format as the input data.

| | |
|------------|---|
| filterData | <i>Run filter on any StoX related data source</i> |
|------------|---|

Description

Run filter on any StoX related data source

Usage

```
filterData(
  inputData,
  filterExpression,
  propagateDownwards = TRUE,
  propagateUpwards = FALSE
)
```

Arguments

| | |
|--------------------|---|
| inputData | An input data. Can be a list of biotic data (StoX data type BioticData), list of acoustic data, StoxBiotic data, or StoxAcoustic data. |
| filterExpression | Filter expression in list of strings. The name of the list and structures should be identical to the names of the input data list. |
| propagateDownwards | Whether the filter action will propagate in the downwards direction. Default to TRUE. |
| propagateUpwards | Whether the filter action will propagate in the upwards direction. Default to FALSE. |

Value

An object of filtered data in the same format as the input data.

| | |
|---------------|---------------------------|
| FilterLanding | <i>Filter LandingData</i> |
|---------------|---------------------------|

Description

Filters [LandingData](#).

Usage

```
FilterLanding(LandingData, FilterExpression, FilterUpwards = FALSE)
```

Arguments

| | |
|------------------|--|
| LandingData | Input LandingData data. |
| FilterExpression | Filter expression in list of strings. The name of the list and structures should be identical to the names of the input data list. |
| FilterUpwards | Whether the filter action will propagate in the upwards direction. Default to FALSE. |

Value

An object of filtered data in the same format as the input data.

FilterStoxAcoustic *Filter StoxAcoustic data*

Description

Filters [StoxAcousticData](#).

Usage

```
FilterStoxAcoustic(StoxAcousticData, FilterExpression, FilterUpwards = FALSE)
```

Arguments

StoxAcousticData Input [StoxAcousticData](#) data.

FilterExpression Filter expression in list of strings. The name of the list and structures should be identical to the names of the input data list.

FilterUpwards Whether the filter action will propagate in the upwards direction. Default to FALSE.

Value

An object of filtered data in the same format as the input data.

FilterStoxBiotic *Filter StoxBiotic data*

Description

Filters [StoxBioticData](#).

Usage

```
FilterStoxBiotic(StoxBioticData, FilterExpression, FilterUpwards = FALSE)
```

Arguments

StoxBioticData Input [StoxBioticData](#) data.

FilterExpression Filter expression in list of strings. The name of the list and structures should be identical to the names of the input data list.

FilterUpwards Whether the filter action will propagate in the upwards direction. Default to FALSE.

Value

An object of filtered data in the same format as the input data.

FilterStoxLanding *Filter StoxLanding data*

Description

Filters [StoxLandingData](#).

Usage

```
FilterStoxLanding(StoxLandingData, FilterExpression)
```

Arguments

StoxLandingData

Input [StoxLandingData](#) data.

FilterExpression

Filter expression in list of strings. The name of the list and structures should be identical to the names of the input data list.

Value

An object of filtered data in the same format as the input data.

generalSamplingHierarchy

General sampling hierarchy of StoX

Description

The general sampling hierarchy of StoX defines a common hierarchy of sampling levels for the StoxBiotic and StoxAcoustic data formats.

Details

The general sampling hierarchy of StoX is defined by 6 levels (tables) as shown alongside the levels of the StoxBiotic and StoxAcoustic format in the following table:

| General level | StoxBiotic level | StoxAcoustic level |
|---------------|------------------|--------------------|
| Cruise | Cruise | Cruise |
| Station | Station | Log |
| Equipment | Haul | Beam |
| Species | SpeciesCategory | AcousticCategory |
| Sample | Sample | ChannelReference |
| Individual | Individual | NASC |

The levels can be interpreted as follows:

- (1) The Cruise level is the entire trip or mission conducted by a platform, such as a research vessel.
- (2) The Station level is a geographical position at a specific point in time where sampling is conducted.
- (3) The Equipment level specifies the equipment used to sample, possibly several equipments at the same station, such as two different trawls or different acoustic instruments or acoustic frequencies.
- (4) The Species level is the biological species or acoustic category (normally reflecting one or more biological species) sampled by the equipment.
- (5) The Sample level is the specific sample of the Species, such as herring or cod for StoxBiotic. For StoxAcoustic the Sample level denotes different coordinate systems in which the acoustic data are defined, with possible values "P" for pelagic channels defined by origin at the surface and z axis pointing vertically downwards, and "B" for bottom referenced channels with origin on the seabed and z axis pointing vertically upwards.
- (6) The Individual level contains for the StoxBiotic format the individuals selected for specific measurements of individual properties such as length, weight and gender, whereas for StoxAcoustic the individual samples along an acoustic beam.

| | |
|-------------------|---|
| general_arguments | <i>General parameters of RstoxData.</i> |
|-------------------|---|

Description

All functions referring to a project, a model, a process or an output table use the same parameters, listed here.

Arguments

| | |
|----------------|--|
| processData | The current data produced by a previous instance of the function. |
| UseProcessData | Logical: If TRUE use the existing function output in the process. |
| NumberOfCores | The number of cores to use (defaulted to 1), truncated to the number of available cores. |

| | |
|------------------|--|
| getNumberOfCores | <i>Pick a suitable number of cores</i> |
|------------------|--|

Description

Pick a suitable number of cores

Usage

```
getNumberOfCores(NumberOfCores = NULL, n = NULL)
```

Arguments

| | |
|---------------|--|
| NumberOfCores | The number of cores to use (defaulted to 1), truncated to the number of available cores. |
| n | Optional length of the data to apply parallel processing to. |

Value

The number of cores to apply.

`getRstoxDataDefinitions`

Get RstoxData definitions

Description

This function gets vital definitions from the RstoxData environment.

Usage

```
getRstoxDataDefinitions(name = NULL, ...)
```

Arguments

| | |
|------|--|
| name | An optional string vector denoting which definitions to extract. |
| ... | values overriding the values of definitions. |

Value

A list of definitions.

Examples

```
getRstoxDataDefinitions()
```

| | |
|-------------|--------------------------------------|
| getStoxKeys | <i>Get the keys of a StoX format</i> |
|-------------|--------------------------------------|

Description

Get the keys of a StoX format

Usage

```
getStoxKeys(  
  StoXDataType = c("StoxBiotic", "StoxAcoustic"),  
  level = NULL,  
  keys.out = c("all", "only.present", "all.but.present")  
)
```

Arguments

| | |
|--------------|--|
| StoxDataType | The name of the StoX format (only StoxBiotic implemented yet). |
| level | The name of the level/table to get keys for. |
| keys.out | Specification of what to return. One of "all", to return all keys of the level; "only.present", to return only the key of the level; and "all.but.present", to return all keys except the present key. |

| | |
|--------------|--|
| ICESAcoustic | <i>Convert AcousticData to ICESAcousticData.</i> |
|--------------|--|

Description

Note that this function only supports [AcousticData](#) object created from reading an ICES acoustic XML files.

Usage

```
ICESAcoustic(AcousticData)
```

Arguments

| | |
|--------------|--|
| AcousticData | A AcousticData object from an ICES acoustic XML format file. |
|--------------|--|

Value

An [ICESAcousticData](#) object.

| | |
|------------------|--|
| ICESAcousticData | <i>StoX data type ICESAcousticData</i> |
|------------------|--|

Description

Acoustic data stored in the ICESAcoustic (CSV) format.

Details

This StoX data type is produced by [ICESAcoustic](#), and contains one list per input biotic file read to produce the input to [ICESAcoustic](#), each holding the tables Instrument, Calibration, DataAcquisition, DataProcessing, Cruise and Data (here Data is a table merged from Log, Sample and Data of the ICESAcoustic xml format). Each file read to produce the input to [ICESAcoustic](#)

See Also

[DataTypes](#) for a list of all StoX data types produced by [RstoxData](#)

| | |
|------------|--|
| ICESBiotic | <i>Convert BioticData to ICESBiotic format</i> |
|------------|--|

Description

Given an [BioticData](#) object, this function converts to ICESBiotic format. Note that this function only supports BioticData generated from NMDBiotic version > 3 XML files.

Usage

```
ICESBiotic(
  BioticData,
  SurveyName = character(),
  Country = character(),
  Organisation = integer(),
  AllowRemoveSpecies = TRUE
)
```

Arguments

| | |
|------------|---|
| BioticData | a BioticData object from an XML file with NMD biotic version 3 format. |
| SurveyName | A string naming the survey. Must be one of the names listed on https://vocab.ices.dk/?ref=1453 or NONE (the default). |
| Country | The ISO_3166 code of the country running the cruise. See http://vocab.ices.dk/?ref=337 . |

- Organisation** An integer code representing the organization running the cruise. See <https://vocab.ices.dk/?ref=1398> for a list of possible codes (e.g., Institute of Marine Research: 612).
- AllowRemoveSpecies** ICES submission will not allow the resulting CSV file to be uploaded if the file contains species not listed in <https://acoustic.ices.dk/Services/Schema/XML/SpecWoRMS.xml>. Setting this parameter to TRUE will remove the unlisted species records.

Value

An [ICESBioticData](#) object.

| | |
|----------------|--------------------------------------|
| ICESBioticData | <i>StoX data type ICESBioticData</i> |
|----------------|--------------------------------------|

Description

Biotic data stored in the ICESBiotic (CSV) format.

Details

This StoX data type is produced by [ICESBiotic](#), and contains one list per input biotic file read to produce the input to [ICESBiotic](#), each holding the tables Cruise, Haul, Catch and Biology, in that hierarchical order.

See Also

[DataTypes](#) for a list of all StoX data types produced by [RstoxData](#)

| | |
|------------|--|
| ICESDatras | <i>Convert BioticData to ICESDatras format</i> |
|------------|--|

Description

Given an [BioticData](#) object, this function converts to ICESDatras format. Note that this function only supports [BioticData](#) NMDBiotic version > 3 XML files.

Usage

```
ICESDatras(BioticData)
```

Arguments

BioticData a [BioticData](#) object from an XML file with NMD biotic version 3 format.

Value

An [ICESDatrasData](#) object.

| | |
|----------------|--------------------------------------|
| ICESDatrasData | <i>StoX data type ICESDatrasData</i> |
|----------------|--------------------------------------|

Description

Biotic data stored in the ICESDatras (CSV) format.

Details

This StoX data type is produced by [ICESDatras](#), and contains one list per input biotic file read to produce the input to [ICESDatras](#), each holding the tables HH, HL and CA, in that hierarchical order.

See Also

[DataTypes](#) for a list of all StoX data types produced by [RstoxData](#)

| | |
|----------------|---|
| is.LandingData | <i>Check if argument is LandingData</i> |
|----------------|---|

Description

Checks if argument conforms to specification for [LandingData](#)

Usage

```
is.LandingData(LandingData)
```

Arguments

LandingData argument to be checked for data conformity

Value

logical, TRUE if argument conformed to specification for [LandingData](#)

| | |
|--------------------|---|
| is.StoxLandingData | <i>Check if argument is StoxLandingData</i> |
|--------------------|---|

Description

Checks if argument conforms to specification for [StoxLandingData](#)

Usage

```
is.StoxLandingData(StoxLandingData)
```

Arguments

StoxLandingData
argument to be checked for data conformity

Value

logical, TRUE if argument conformed to specification for [StoxLandingData](#)

| | |
|-------------|--------------------|
| LandingData | <i>LandingData</i> |
|-------------|--------------------|

Description

LandingData

Data

One entry 'Seddellinje' is one line of a sales-note or landing-note. These are issued as fish is landed, and a complete set of these for a period can be considered a census of all first hand sale of fish sold from Norwegian vessels.

Format

list with one member for each sales-note set. Each member is a list of [data.table](#) representing the different complexTypes in namespace <http://www.imr.no/formats/landinger/v2> For ease of merging: all top level attributes are repeated for all tables. And all line-identifying variables are included as top-level attributes.

See Also

[DataTypes](#) for a list of all StoX data types produced by [RstoxData](#)

| | |
|---------------|---|
| lapplyOnCores | <i>Run a function on all elements of x on one or more cores</i> |
|---------------|---|

Description

Run a function on all elements of x on one or more cores

Usage

```
lapplyOnCores(x, FUN, NumberOfCores = 1L, ...)
```

Arguments

| | |
|---------------|--|
| x | An object to apply FUN to. |
| FUN | The function to apply. |
| NumberOfCores | The number of cores to use (defaulted to 1), truncated to the number of available cores. |
| ... | Additional arguments to FUN. |

Value

A list of outputs from FUN.

| | |
|---------------|---|
| mapplyOnCores | <i>Run a function on all elements of x on one or more cores</i> |
|---------------|---|

Description

Run a function on all elements of x on one or more cores

Usage

```
mapplyOnCores(FUN, NumberOfCores = 1L, ..., MoreArgs = NULL, SIMPLIFY = FALSE)
```

Arguments

| | |
|-------------------------|--|
| FUN | The function to apply. |
| NumberOfCores | The number of cores to use (defaulted to 1), truncated to the number of available cores. |
| ..., MoreArgs, SIMPLIFY | See mapply . |

Value

A list of outputs from FUN.

| | |
|------------------|--|
| mergeByIntersect | <i>Merge two data tables by the intersect of the names</i> |
|------------------|--|

Description

Merge two data tables by the intersect of the names

Usage

```
mergeByIntersect(x, y, ..., msg = FALSE)
```

Arguments

| | |
|------|---|
| x, y | Data tables of class data.table . |
| ... | Various overrides. |
| msg | Verbose message switch, default to FALSE. |

Value

A merged data table.

| | |
|-----------------|---|
| mergeByStoxKeys | <i>Merge two data tables by StoX keys</i> |
|-----------------|---|

Description

Merge two data tables by StoX keys

Usage

```
mergeByStoxKeys(x, y, StoxDataType, toMergeFromY = NULL, replace = FALSE, ...)
```

Arguments

| | |
|--------------|--|
| x, y | Data tables of class data.table . |
| StoxDataType | Input data type. Text string of StoxBiotic or StoxAcoustic . |
| toMergeFromY | Specify key columns from y. NULL means all similarly named columns from x and y will be merged. Default to NULL. |
| replace | Whether to replace the variables in the target. Default to FALSE. |
| ... | Extra parameters that will be passed into merge . |

Value

A merged data table.

| | |
|-----------------|--|
| mergeDataTables | <i>Merge list of data tables recursively</i> |
|-----------------|--|

Description

Merge list of data tables recursively

Usage

```
mergeDataTables(data, tableNames = NULL, output.only.last = FALSE, ...)
```

Arguments

| | |
|------------------|---|
| data | A list of data tables. |
| tableNames | A character vector holding the names of the tables to merge. |
| output.only.last | Only returns last merged table. |
| ... | Extra parameters that will be passed into merge . |

Value

A merged data table.

| | |
|-------------------|-------------------------------|
| MergeStoxAcoustic | <i>Merge StoxAcousticData</i> |
|-------------------|-------------------------------|

Description

Merge StoxAcousticData

Usage

```
MergeStoxAcoustic(StoxAcousticData, TargetTable = "NASC")
```

Arguments

| | |
|------------------|---|
| StoxAcousticData | A list of StoX acoustic data (StoX data type StoxAcousticData). |
| TargetTable | The name of the table up until which to merge (the default "NASC" implies merging all tables) |

Value

An object of StoX data type [MergeStoxAcousticData](#).

MergeStoxAcousticData *StoX data type MergeStoxAcousticData*

Description

Merged [StoxAcousticData](#).

Details

This StoX data type is produced by [MergeStoxAcoustic](#), and contains one merged table of [StoxAcousticData](#).

See Also

[DataTypes](#) for a list of all StoX data types produced by [RstoxData](#)

MergeStoxBiotic *Merge StoxBioticData*

Description

Merge StoxBioticData

Usage

```
MergeStoxBiotic(StoxBioticData, TargetTable = "Individual")
```

Arguments

StoxBioticData A list of StoX biotic data (StoX data type [StoxBioticData](#)).

TargetTable The name of the table up until which to merge (the default "Individual" implies merging all tables)

Value

An object of StoX data type [MergeStoxBioticData](#).

MergeStoxBioticData *StoX data type MergeStoxBioticData*

Description

Merged [StoxBioticData](#).

Details

This StoX data type is produced by [MergeStoxBiotic](#), and contains one merged table of [StoxBioticData](#).

See Also

[DataTypes](#) for a list of all StoX data types produced by [RstoxData](#)

ModelData *StoX data types of the RstoxData package*

Description

StoX data types are the data types used to transfer data and information between processes in a StoX estimation model.

Arguments

BioticData [BioticData](#).
StoxBioticData [StoxBioticData](#).
AcousticData [AcousticData](#).
StoxAcousticData
 [StoxAcousticData](#).

Details

This RstoxData package produces the following StoX data types:

- [BioticData](#)
- [StoxBioticData](#)
- [MergeStoxBioticData](#)
- [AcousticData](#)
- [StoxAcousticData](#)
- [MergeStoxAcousticData](#)
- [LandingData](#)
- [StoxLandingData](#)

- [ICESAcousticData](#)
- [ICESBioticData](#)
- [ICESDatrasData](#)
- [WriteICESAcousticData](#)
- [WriteICESBioticData](#)
- [WriteICESDatrasData](#)

See Also

[RstoxBase](#) and [RstoxFDA](#) for a list of all StoX data types produced by the other official StoX function packages.

| | |
|-----------------|--------------------------|
| parseInterCatch | <i>Parses InterCatch</i> |
|-----------------|--------------------------|

Description

Parses the InterCatch exchange format v 1.0 for Commercial Catch and Sample Data.

Usage

```
parseInterCatch(file, encoding = "UTF-8")
```

Arguments

| | |
|----------|---|
| file | path to file containing intercatch formatted data |
| encoding | encoding of 'file' |

Details

The InterCatch exchange format is a jagged comma-separated format, where the number of fields on a line is determined by a record-type identifier in position 1. Three record types are defined, "HI" (header information), "SI" (species information), and "SD" (species data). The format is specified on <https://ices.dk/data/Documents/Intercatch/IC-ExchangeFormat1-0.pdf>.

Value

named list with three members:

HI [data.table](#) with HI records

SI [data.table](#) with SI records

SD [data.table](#) with SD records

| | |
|-------------|---|
| ProcessData | <i>Process data used in estimation models in StoX</i> |
|-------------|---|

Description

The process data of the RstoxData package.

Details

- [Translation](#)

See Also

[ModelData](#) for model data types and [DataTypes](#) for all data types produced by [RstoxData](#).

| | |
|------------------------|---|
| processPropertyFormats | <i>Define the process property formats:</i> |
|------------------------|---|

Description

Define the process property formats:

Usage

```
processPropertyFormats
```

Format

An object of class list of length 14.

| | |
|--------------|--------------------------------|
| ReadAcoustic | <i>Read acoustic XML files</i> |
|--------------|--------------------------------|

Description

This function reads multiple acoustic file to a list with a list of tables for each file.

Usage

```
ReadAcoustic(FileNames)
```

Arguments

| | |
|-----------|----------------------------------|
| FileNames | The paths of the acoustic files. |
|-----------|----------------------------------|

Details

This function is awesome and does excellent stuff.

Value

An object of StoX data type AcousticData: A list of a list of data.tables of the different levels of the input acoustic files.

See Also

[readXmlFile](#).

Examples

```
exampleFile <- system.file(
  "testresources", "libas_ListUserFile20__L40.0-2259.9_small.xml", package="RstoxData")
bioticData <- ReadBiotic(exampleFile)
```

| | |
|------------|------------------------------|
| ReadBiotic | <i>Read biotic XML files</i> |
|------------|------------------------------|

Description

This function reads multiple biotic file to a list with a list of tables for each file.

Usage

```
ReadBiotic(FileNames)
```

Arguments

FileNames The paths of the biotic files.

Details

This function is awesome and does excellent stuff.

Value

An object of StoX data type BioticData: A list of a list of data.tables of the different levels of the input biotic files.

See Also

[readXmlFile](#).

Examples

```
exampleFile <- system.file("testresources","biotic3.1_example.xml", package="RstoxData")
bioticData <- ReadBiotic(exampleFile)
```

| | |
|-------------|------------------------------|
| readErsFile | <i>Parses logbooks (ERS)</i> |
|-------------|------------------------------|

Description

Parses electronic logbooks (ERS) from tabular format delivered by Directorate of Fisheries (FDIR)

Usage

```
readErsFile(file, encoding = "Latin-1")
```

Arguments

| | |
|----------|--|
| file | path to file |
| encoding | encoding for 'file', must be accepted by fread |

Details

The format is a pipe-separated format encoding aggregated ERS records (logbooks). It is provided to IMR on a regular basis from FDIR. Column headers are in Norwegian.

Value

data.table() with logbooks

| | |
|-------------|-------------------------------|
| ReadLanding | <i>Read landing XML files</i> |
|-------------|-------------------------------|

Description

This function reads multiple landing files (sales-notes) to a list with a list of tables for each file.

Usage

```
ReadLanding(FileNames)
```

Arguments

| | |
|-----------|---------------------------------|
| FileNames | The paths of the landing files. |
|-----------|---------------------------------|

Details

This sales notes are expected to be XML-formatted with elements defined by the namespace: <http://www.imr.no/formats/landinger/v2>

Value

An object of StoX data type [LandingData](#)).

See Also

[readXmlFile](#).

Examples

```
exampleFile <- system.file(
  "testresources", "landing.xml", package="RstoxData")
landingData <- ReadLanding(exampleFile)
```

| | |
|-------------|--------------------------------------|
| readLssFile | <i>Parses landings (sales notes)</i> |
|-------------|--------------------------------------|

Description

Parses sales notes data from the Norwegian Directorate of Fisheries (FDIR) on the LSS format

Usage

```
readLssFile(file, encoding = "Latin-1", guessMax = 1e+05, strict = T)
```

Arguments

| | |
|----------|--|
| file | path to file with LSS landings |
| encoding | encoding for 'file', must be accepted by fread |
| guessMax | deprecated parameter, has no effect. |
| strict | enforce strict adherence to data format. |

Details

The LSS format is a pipe-separated format encoding landings (sales-notes). It is provided to IMR on a regular basis from FDIR. Column headers are in Norwegian.

Historically, columns in the landings provided from FDIR has been adapted for each data delivery. Lately data deliveries has become standardized, but in order to support variants adherence to the standardization is not enforced by this function, unless option 'strict' is selected. If column names does not match specification, but data is otherwise parse-able, a warning will be issued.

If the parameter 'strict' is not TRUE, data types may be inferred from data.

Value

data.table with LSS landings

| | |
|-------------|--|
| readXmlFile | <i>Read fisheries XML data format file</i> |
|-------------|--|

Description

Read fisheries XML data format file. Currently supports IMR Biotic version 1 until 3, IMR Echosounder version 1, and IMR Landing version 2 formats at the moment. Streaming XML pull parser can be used to avoid loading the whole XML into memory and it supports ZIP file reading. Please note that the XML file inside the zip file should be using the same name as the zip file itself (e.g. test.xml inside test.zip).

Usage

```
readXmlFile(
  xmlFilePath,
  stream = TRUE,
  useXsd = NULL,
  usePrefix = NULL,
  verbose = FALSE
)
```

Arguments

| | |
|-------------|---|
| xmlFilePath | full path to the XML file to be read. |
| stream | a streaming XML pull parser is used if this is set to TRUE. An XML DOM parser is used if this is set to FALSE. Default to TRUE. |
| useXsd | Specify an xsd object to use. Default to NULL. |
| usePrefix | Manually specify a namespace prefix. Default to NULL. |
| verbose | Show verbose output. Default to FALSE. |

Value

List of data.table objects containing the "flattened" XML data.

Examples

```
## Not run:
# Reading test.xml using XML pull parser
one <- readXmlFile("./test.xml")
# Reading test.xml using XML DOM parser
two <- readXmlFile("./test.xml", stream = FALSE)
# Reading test.xml inside test.zip file
three <- readXmlFile("./test.zip")
```



```
## End(Not run)
```

RedefineStoxBiotic *Redefine StoxBioticData variables by data from BioticData*

Description

This function redefines one or more columns of `StoxBioticData` by columns of `BioticData`.

Usage

```
RedefineStoxBiotic(
  StoxBioticData,
  BioticData,
  Redefinition = data.table::data.table()
)
```

Arguments

`StoxBioticData` An input of `ModelData` object
`BioticData` An input of `ModelData` object
`Redefinition` A table of the columns "VariableName", representing the variable to redefine; and "ReplaceBy", representing the variable from `BioticData` to replace by.

Value

A `StoxBioticData` object.

RstoxData *Tools to Read and Manipulate Fisheries Data*

Description

Set of tools to read and manipulate various data formats for fisheries. Mainly catered towards scientific trawl survey sampling ('biotic') data, acoustic trawl data, and commercial fishing catch ('landings') data. Among the supported data formats are the data products from the Norwegian Institute Marine Research ('IMR') and the International Council for the Exploration of the Sea (ICES).

Details

The RstoxData package contains functions for reading, filtering and writing biotic, acoustic and landing data as XML files. Filtering can be done by R syntax such as `longitude > 10`, or by pre defined functions such as `inside()`. On computers that return errors when trying to run the Rtools through RStudio (most institutional Windows machines), install the binary directly from <https://github.com/StoXProject/RstoxData/releases>. Download the newest RstoxData zip file, click the "Packages" tab -> "Install" -> "Install from:" "Package Archive File" -> "Install". If the installer does not complain, the package is installed correctly.

Author(s)

Maintainer: Ibrahim Umar <ibrahim.umar@hi.no>

Authors:

- Sindre Vatnehol
- Arne Johannes Holmin
- Edvin Fuglebakk
- Espen Johnsen

Other contributors:

- Norwegian Institute of Marine Research [copyright holder, funder]

See Also

Useful links:

- <https://github.com/StoXProject/RstoxData>
- Report bugs at <https://github.com/StoXProject/RstoxData/issues>

| | |
|-------------------|--|
| setorderv_numeric | <i>Order a data.table (by reference) by interpreting characters as numeric if possible</i> |
|-------------------|--|

Description

Order a data.table (by reference) by interpreting characters as numeric if possible

Usage

```
setorderv_numeric(dataOne, by = NULL, key = NULL, ...)
```

Arguments

| | |
|---------|---|
| dataOne | A data.table. |
| by | Order by the given columns. |
| key | If given and by is empty, orderr by the columns with names ending with key. |
| ... | Passed on to setorderv |

`setRstoxPrecisionLevel`*Round off to number of digits*

Description

Round off to number of digits

Usage

```
setRstoxPrecisionLevel(x)
```

Arguments

x A list of data . tables or a single data . table object.

Value

A transformed object.

`StoxAcoustic`*Convert AcousticData to StoxAcousticData*

Description

Convert AcousticData to StoxAcousticData

Usage

```
StoxAcoustic(AcousticData)
```

Arguments

AcousticData [AcousticData](#).

Value

An object of StoX data type [StoxAcousticData](#).

StoxAcousticData *StoX data type StoxAcousticData*

Description

Acoustic data stored in the StoxAcoustic format, which contains the variables needed for most estimation models used by StoX.

Details

This StoX data type is produced by [StoxAcoustic](#), and contains the tables Cruise, Log, Beam, AcousticCategory, ChannelReference and NASC in that hierarchical order.

See Also

[DataTypes](#) for a list of all StoX data types produced by [RstoxData](#)

StoxBiotic *Convert BioticData to StoxBioticData*

Description

Convert BioticData to StoxBioticData

Usage

```
StoxBiotic(BioticData)
```

Arguments

BioticData [BioticData](#).

Value

An object of StoX data type [StoxBioticData](#).

See Also

The definition of the [StoxBiotic format](#) and [generalSamplingHierarhcy](#).

| | |
|----------------|--------------------------------------|
| StoxBioticData | <i>StoX data type StoxBioticData</i> |
|----------------|--------------------------------------|

Description

Biotic data stored in the [StoxBiotic format](#), which contains the variables needed for most estimation models used by StoX.

Details

This StoX data type is produced by [StoxBiotic](#), and contains the tables Cruise, Station, Haul, SpeciesCategory, Sample and Individual in that hierarchical order.

See Also

[DataTypes](#) for a list of all StoX data types produced by [RstoxData](#)

| | |
|------------------|--------------------------------|
| StoxBioticFormat | <i>StoxBiotic data format.</i> |
|------------------|--------------------------------|

Description

The StoxBiotic data format is defined by StoX as a common format to which data from different biotic sampling formats are converted, guaranteeing consistent interpretation and documentation of all its variables.

Details

The StoxBiotic format is defined according to the [general sampling hierarchy of StoX](#) which is used as a basis for both the StoxcBiotic and StoxAcoustic format. The variables of the StoxBiotic format are given by the tables below:

Cruise level:**Variable Description**

| | |
|-----------|---|
| CruiseKey | Key of the Cruise table |
| Cruise | Unique Cruise identifier ("/" separated concatenation of cruise, missiontype, startyear, platform and missionnum) |
| Platform | Data collection platform identifier |

Station level:**Variable Description**

| | | |
|---------------|--|-------------|
| StationKey | Key of the Station level | Unit |
| Station | Unique Station identifier | None |
| CatchPlatform | Platform performing the actual sampling (can be different from the data collection platform) | None |

| | | |
|-------------|--------------------------------------|--------------|
| DateTime | UTC time at start of the station | ISO 8601 for |
| Longitude | Longitude at start of the station | Decimal degr |
| Latitude | Latitude at start of the station | Decimal degr |
| BottomDepth | Bottom depth at start of the station | m |

Haul level:

| Variable | Description | Unit | Data type | Example |
|----------------------|--|----------------|-----------|-------------|
| HaulKey | Key of the Haul level | None | Character | 2 |
| Haul | Unique Haul identifier | None | Character | 2021105-1-2 |
| Gear | Identifier of the gear | None | Character | 3270 |
| TowDistance | Distance between start and end of the haul | Nautical miles | Numeric | 1.5 |
| EffectiveTowDistance | Effective tow distance | Nautical miles | Numeric | 1.5 |
| MinHaulDepth | Minimum depth of the haul (trawl headline) | m | Numeric | 65 |
| MaxHaulDepth | Maximum depth of the haul (trawl headline) | m | Numeric | 35 |
| VerticalNetOpening | Vertical span of the net | m | Numeric | 23 |
| HorizontalNetOpening | Vertical span of the net | m | Numeric | 105 |
| TrawlDoorSpread | Distance between the trawl doors | m | Numeric | 125 |

SpeciesCategory level:

| Variable | Description | Unit | Data type | Example |
|--------------------|----------------------------------|------|-----------|---------|
| SpeciesCategoryKey | Key of the SpeciesCategory level | None | Character | 126417 |
| SpeciesCategory | The species category | None | Character | Herring |

Sample level:

| Variable | Description |
|---------------------|---|
| SampleKey | Key of the Sample level |
| Sample | Unique Sample identifier |
| CatchFractionWeight | Total weight of the catch SpeciesCategory and sub category (fractions such as juveniles and adults) |
| CatchFractionCount | Total number of individuals of the catch SpeciesCategory and sub category (fractions such as juveniles) |
| SampleWeight | Total weight of the sample for individual measurements |
| SampleCount | Size of the sample for individual measurements |

Individual level:

| Variable | Description | Unit | Data type | Example |
|-----------------------|---|------|-----------|---------|
| IndividualKey | Key of the Individual level | None | Character | |
| Individual | Unique Individual identifier | None | Character | |
| IndividualRoundWeight | Round weight (the whole fish) fo the individual | g | Numeric | 123 |
| IndividualTotalLength | Total length (from snoute to end of fin) | cm | Numeric | 14.5 |
| LengthResolution | Resolution of IndividualTotalLength | cm | Numeric | 0.5 |

| | | | | |
|-------------------|----------------------|------------------------|-----------|---|
| WeightMeasurement | | None | Character | |
| IndividualAge | Age of an individual | year | Numeric | 3 |
| IndividualSex | sex of an individual | F is female, M is male | Character | F |

stoxBioticObject *stoxBioticObject*

Description

Pre-processed objects for raw XML data to StoXBiotic format

Usage

stoxBioticObject

Format

An object of class list of length 9.

stoxFunctionAttributes
Function specification for inclusion in StoX projects

Description

Function specification for inclusion in StoX projects

Usage

stoxFunctionAttributes

Format

An object of class list of length 31.

StoxLanding

Convert landing data

Description

Convert landing data to the aggregated format [StoxLandingData](#)

Usage

```
StoxLanding(LandingData)
```

Arguments

LandingData Sales-notes data. See [LandingData](#)

Details

All columns that are not the ones aggregated (weight), will be used as aggregation variables. Correspondences indicate which field a value is derived from, not necessarily verbatim copied. Correspondence to LandingData (<http://www.imr.no/formats/landinger/v2>):

Species Art_kode

Year Fangstår

CatchDate SisteFangstdato

Gear Redskap_kode

Area Hovedområde_kode

Location Lokasjon_kode

Coastal KystHav_kode

N62Code NordSørFor62GraderNord

VesselLength StørsteLengde

CountryVessel Fartøynasjonalitet_kode

LandingSite Mottaksstasjon

CountryLanding Landingsnasjon_kode

Usage HovedgruppeAnvendelse_kode

RoundWeight Rundvekt

Value

[StoxLandingData](#), aggregated landings data.

StoxLandingData *StoxLandingData*

Description

Contains a list with one element 'Landing', described below.

Details

'Landing' is a [data.table](#) with aggregated weight of landings from landing records. Columns are specified in the section Column definitions Landing

Column definitions Landing

Species character() code for species category (species identified by market or regulation standards. Several codes may code the same species or stock, and some catch may be recorded only at higher taxonomic classifications)

Year integer() Year of catch

CatchDate POSIXct() Date of catch (last catch on trip) in UTC

Gear character() Code for gear used for catch (dominant gear for trip)

Area character() Area code for the position where the catch was caught (dominant area for trip)

SubArea character() Subdivision of area code for the position where the catch was caught (dominant area for trip)

Coastal character() Code indicating whether catch was taken within coastal delimitation line (dominant side for trip)

N62Code character() Code indicating whether catch was taken north or south of 62 deg. Lat. (dominant side for trip)

VesselLength character() Length of vessel in m

CountryVessel character() Country of the vessel that caught the catch

LandingSite character() Code identifying landing site (buyer of catch)

CountryLanding character() Country where catch was landed

Usage character() Code for market usage of catch.

RoundWeight numeric() Weight of round catch in kg.

See Also

[DataTypes](#) for a list of all StoX data types produced by [RstoxData](#)

| | |
|-------------------|-------------------------------|
| TranslateAcoustic | <i>Translate AcousticData</i> |
|-------------------|-------------------------------|

Description

This function translates one or more columns of [AcousticData](#) to new values given by the input Translation.

Usage

```
TranslateAcoustic(AcousticData, Translation)
```

Arguments

| | |
|--------------|---|
| AcousticData | An input of ModelData object |
| Translation | The process from which to get the Translation definition. |

Value

A [AcousticData](#) object.

| | |
|-----------------|-----------------------------|
| TranslateBiotic | <i>Translate BioticData</i> |
|-----------------|-----------------------------|

Description

This function translates one or more columns of [BioticData](#) to new values given by the input Translation.

Usage

```
TranslateBiotic(BioticData, Translation)
```

Arguments

| | |
|-------------|---|
| BioticData | An input of ModelData object |
| Translation | The process from which to get the Translation definition. |

Value

A [BioticData](#) object.

TranslateICESAcoustic *Translate ICESAcousticData*

Description

This function translates one or more columns of [ICESAcousticData](#) to new values given by the input Translation.

Usage

```
TranslateICESAcoustic(ICESAcousticData, Translation)
```

Arguments

ICESAcousticData

An input of [ModelData](#) object

Translation

The process from which to get the [Translation](#) definition.

Value

A [ICESAcousticData](#) object.

TranslateICESBiotic *Translate ICESBioticData*

Description

This function translates one or more columns of [ICESBioticData](#) to new values given by the input Translation.

Usage

```
TranslateICESBiotic(ICESBioticData, Translation)
```

Arguments

ICESBioticData An input of [ModelData](#) object

Translation

The process from which to get the [Translation](#) definition.

Value

A [ICESBioticData](#) object.

TranslateLanding *Translate LandingData*

Description

This function translates one or more columns of [LandingData](#) to new values given by the input Translation.

Usage

TranslateLanding(LandingData, Translation)

Arguments

LandingData An input of [ModelData](#) object
 Translation The process from which to get the [Translation](#) definition.

Value

A [LandingData](#) object.

TranslateStoxAcoustic *Translate StoxAcousticData*

Description

This function translates one or more columns of [StoxAcousticData](#) to new values given by the input Translation.

Usage

TranslateStoxAcoustic(StoxAcousticData, Translation)

Arguments

StoxAcousticData
 An input of [ModelData](#) object
 Translation The process from which to get the [Translation](#) definition.

Value

A [StoxAcousticData](#) object.

TranslateStoxBiotic *Translate StoxBioticData*

Description

This function translates one or more columns of [StoxBioticData](#) to new values given by the input Translation.

Usage

```
TranslateStoxBiotic(StoxBioticData, Translation)
```

Arguments

StoxBioticData An input of [ModelData](#) object

Translation The process from which to get the [Translation](#) definition.

Value

A [StoxBioticData](#) object.

TranslateStoxLanding *Translate StoxLandingData*

Description

This function translates one or more columns of [StoxLandingData](#) to new values given by the input Translation.

Usage

```
TranslateStoxLanding(StoxLandingData, Translation)
```

Arguments

StoxLandingData

 An input of [ModelData](#) object

Translation The process from which to get the [Translation](#) definition.

Value

A [StoxLandingData](#) object.

| | |
|-------------|--|
| Translation | <i>Translation definition (from file or from table).</i> |
|-------------|--|

Description

Translation definition (from file or from table).

Details

This StoX data type is produced by [DefineTranslation](#), and contains the columns VariableName, Value and NewValue.

See Also

[DataTypes](#) for a list of all StoX data types produced by [RstoxData](#)

| | |
|-------------------|---|
| WriteICESAcoustic | <i>Writes ICESAcousticData to a csv file for each input acoustic file used to create the ICESAcousticData</i> |
|-------------------|---|

Description

Writes [ICESAcousticData](#) to a csv file for each input acoustic file used to create the [ICESAcousticData](#)

Usage

```
WriteICESAcoustic(ICESAcousticData)
```

Arguments

ICESAcousticData
A [ICESAcousticData](#) object obtained from an ICES acoustic XML format file.

Value

List of string matrices in the ICES acoustic CSV format.

WriteICESAcousticData *Rbind ICESAcousticData to a string matrix.*

Description

The output of this function is suited for submission to <https://acoustic.ices.dk/>.

Details

The ICESAcoustic CSV format is one string matrix containing all tables of [ICESAcousticData](#), where column names are included as header rows.

See Also

[DataTypes](#) for a list of all StoX data types produced by [RstoxData](#)

WriteICESBiotic *Writes ICESBioticData to a csv file for each input acoustic file used to create the ICESBioticData*

Description

Writes [ICESBioticData](#) to a csv file for each input acoustic file used to create the [ICESBioticData](#)

Usage

```
WriteICESBiotic(ICESBioticData)
```

Arguments

ICESBioticData A [ICESBioticData](#) object obtained from an ICES acoustic XML format file.

Value

List of string matrices in the ICES acoustic CSV format.

WriteICESBioticData *Rbind ICESBioticData to a string matrix.*

Description

The output of this function is suited for submission to <https://acoustic.ices.dk/>.

Details

The ICESBiotic CSV format is one string matrix containing all tables of [ICESBioticData](#), where column names are included as header rows.

See Also

[DataTypes](#) for a list of all StoX data types produced by [RstoxData](#)

WriteICESDatras *Writes ICESDatrasData to a csv file for each input acoustic file used to create the ICESDatras*

Description

Writes [ICESDatrasData](#) to a csv file for each input acoustic file used to create the [ICESDatras](#)

Usage

```
WriteICESDatras(ICESDatrasData)
```

Arguments

ICESDatrasData A [ICESDatrasData](#) object returned from [ICESDatras](#).

Value

List of string matrices in the ICES Datras CSV format.

WriteICESDatrasData *Rbind ICESDatrasData to a string matrix.*

Description

The output of this function is suited for submission to <https://www.ices.dk/data/data-portals/Pages/DATRAS.aspx>.

Details

The ICESDatras CSV format is one string matrix containing all tables of `ICESDatrasData`, where column names are included as header rows.

See Also

[DataTypes](#) for a list of all StoX data types produced by [RstoxData](#)

xsdObjects *xsdObjects*

Description

Pre-processed XSD file objects

Usage

xsdObjects

Format

A list with 4 elements

landingv2.xsd List Landing Format v2

nmdbioticv1.xsd List NMD Biotic Format v1

nmdbioticv1.1.xsd List NMD Biotic Format v1.1

nmdbioticv1.2.xsd List NMD Biotic Format v1.2

nmdbioticv1.3.xsd List NMD Biotic Format v1.3

nmdbioticv1.4.xsd List NMD Biotic Format v1.4

nmdbioticv3.xsd List NMD Biotic Format v3

nmdbioticv3.1.xsd List NMD Biotic Format v3.1

nmdechounder1.xsd List NMD Echosounder Format v1

Source

<https://www.imr.no/formats>

Index

* datasets

- backwardCompatibility, 4
 - processPropertyFormats, 28
 - stoxBioticObject, 39
 - stoxFunctionAttributes, 39
 - xsdObjects, 49
- AcousticData, 3, 5, 6, 10, 17, 26, 35, 42
- AddToStoxBiotic, 4
- backwardCompatibility, 4
- BioticData, 4, 5, 6, 7, 11, 12, 18, 19, 26, 33, 36, 42
- ConvertAcoustic, 5
- ConvertBiotic, 6
- ConvertStoxAcoustic, 7
- ConvertStoxBiotic, 8
- data.table, 21, 23, 27, 41
- DataTypes, 4, 5, 9, 18–21, 25, 26, 28, 36, 37, 41, 46–49
- DefineTranslation, 9, 46
- FilterAcoustic, 10
- FilterBiotic, 11
- filterData, 11
- FilterLanding, 12
- FilterStoxAcoustic, 13
- FilterStoxBiotic, 13
- FilterStoxLanding, 14
- fread, 30, 31
- general sampling hierarchy of StoX, 37
- general_arguments, 15
- generalSamplingHierarhcy, 14, 36
- getNumberOfCores, 15
- getRstoxDataDefinitions, 16
- getStoxKeys, 17
- ICESAcoustic, 17, 18
- ICESAcousticData, 17, 18, 27, 43, 46, 47
- ICESBiotic, 18, 19
- ICESBioticData, 19, 19, 27, 43, 47, 48
- ICESDatras, 19, 20, 48
- ICESDatrasData, 19, 20, 27, 48, 49
- is.LandingData, 20
- is.StoxLandingData, 21
- LandingData, 12, 20, 21, 26, 31, 40, 44
- lapplyOnCores, 22
- mapply, 22
- mapplyOnCores, 22
- merge, 23, 24
- mergeByIntersect, 23
- mergeByStoxKeys, 23
- mergeDataTables, 24
- MergeStoxAcoustic, 24, 25
- MergeStoxAcousticData, 24, 25, 26
- MergeStoxBiotic, 25, 26
- MergeStoxBioticData, 25, 26, 26
- ModelData, 5–9, 26, 28, 33, 42–45
- parseInterCatch, 27
- ProcessData, 9, 28
- processPropertyFormats, 28
- ReadAcoustic, 4, 28
- ReadBiotic, 5, 29
- readErsFile, 30
- ReadLanding, 30
- readLssFile, 31
- readXmlFile, 29, 31, 32
- RedefineStoxBiotic, 33
- RstoxData, 4, 5, 18–21, 25, 26, 28, 33, 36, 37, 41, 46–49
- RstoxData-package (RstoxData), 33
- setorderv, 34
- setorderv_numeric, 34
- setRstoxPrecisionLevel, 35

StoxAcoustic, [35](#), [36](#)
StoxAcousticData, [7](#), [8](#), [13](#), [24–26](#), [35](#), [36](#), [44](#)
StoxBiotic, [36](#), [37](#)
StoxBiotic format, [36](#), [37](#)
StoxBioticData, [4](#), [8](#), [9](#), [13](#), [25](#), [26](#), [33](#), [36](#),
[37](#), [45](#)
StoxBioticFormat, [37](#)
stoxBioticObject, [39](#)
stoxFunctionAttributes, [39](#)
StoxLanding, [40](#)
StoxLandingData, [14](#), [21](#), [26](#), [40](#), [41](#), [45](#)

TranslateAcoustic, [42](#)
TranslateBiotic, [42](#)
TranslateICESAcoustic, [43](#)
TranslateICESBiotic, [43](#)
TranslateLanding, [44](#)
TranslateStoxAcoustic, [44](#)
TranslateStoxBiotic, [9](#), [45](#)
TranslateStoxLanding, [45](#)
Translation, [10](#), [28](#), [42–45](#), [46](#)

WriteICESAcoustic, [46](#)
WriteICESAcousticData, [27](#), [47](#)
WriteICESBiotic, [47](#)
WriteICESBioticData, [27](#), [48](#)
WriteICESDatras, [48](#)
WriteICESDatrasData, [27](#), [49](#)

xsdObjects, [49](#)