

Package: mixfishtools (via r-universe)

September 25, 2026

Type Package

Title WGMIXFISH tools for reproducibility

Version 0.5.1

Date 2025-09-29

Author Marc Taylor, Mikel Aristegui, Johnathan Ball, Harriet Cole,
Paul Dolder

Maintainer Marc Taylor <marc.taylor@thuenen.de>

Description Contains plot templates used in WGMIXFISH-ADVICE and
Fisheries Overviews.

Depends R (>= 4.1), ggplot2, ggrepel, ggalluvial, dplyr, RColorBrewer

Suggests tidyr, png, knitr, kableExtra, rmarkdown, FLCore, FLasher,
ggplotFL, FLfse, stockassessment, spict, mgcv

License GPL-3 + file LICENSE

Encoding UTF-8

LazyData false

VignetteBuilder knitr

Config/roxygen2/version 8.1.0

Config/pak/sysreqs libicu-dev

Repository <https://ices-tools-prod.r-universe.dev>

Date/Publication 2026-09-25 08:15:11 UTC

RemoteUrl <https://github.com/ices-tools-dev/mixfishtools>

RemoteRef HEAD

RemoteSha 402194a7e7cad7739bf85c0c3f4ce474fc3ead9b

Contents

gm_mean	2
haddockWts	3
NSeffort	4

NSlandings	5
plot_alluvial	6
plot_catch_change	8
plot_catchComp	10
plot_catchCompSpp	12
plot_catchCompStk	15
plot_catchScenStk	18
plot_effortFltStk	20
plot_FOgearType	22
plot_landByMetStock	23
plot_landByStock	25
plot_overUnderFltStk	27
plot_relEffortFltStk	29
plot_stkMetProp	31
refTable	32
stfFltStkSum	33
stfFltSum	34
stfMtStkSum	35
stock_props	36
wgnsskStocks	37
Index	38

gm_mean	<i>Geometric Mean</i>
---------	-----------------------

Description

Calculates the geometric mean of a vector

Usage

```
gm_mean(x, na.rm = TRUE, zero.propagate = FALSE)
```

Arguments

- x numeric vector of positive numbers.
- na.rm Remove NAs befor calculation (as in [mean](#))
- zero.propagate Logical. Should zeros be considered (resulting in output of zero)

References

From [stackoverflow answer](#) posted by [Paul McMurdie](#)

Examples

```
### simple usage
gm_mean(c(1:4))
gm_mean(c(-1:4)) # negative values not allowed
gm_mean(c(0:4)) # zeros do not propagate
gm_mean(c(0:4), zero.propagate=TRUE) #zeros allowed to propagate
gm_mean(c(1,2,3,4, NaN)) # na.rm=TRUE
gm_mean(c(1,2,3,4, NaN), na.rm=FALSE) # na.rm=FALSE

### example of proportional change
df <- data.frame(index1 = 5, index2 = 25) # two indices of differing magnitude
mult <- c(1.25, 1.5) # multiplier
df <- rbind(df, df*mult) # indices change by differing proportions
df # view dataframe
gm_mean(mult) # mean proportional increase
gm_mean(df[2,]) / gm_mean(df[1,]) # equal
gm_mean(df[2,] / df[1,]) # equal
```

haddockWts

*Short-term forecast individual weights for had.27.46a20***Description**

The haddockWts data contains weights at age for catch, landings, discards and stock as modeled by a cohort-based linear model approach (Jarowski, 2011). Values are provided by the stock assessor in text file to WGMIXFISH. Example data is from the assessment conducted in 2024.

- catch - individual catch weight
- landings - individual landings weight
- discards - individual discards weight
- stock - individual stock weight

Usage

```
data(haddockWts)
```

Format

```
list
```

Source

```
WGNSSK 2024
```

Examples

```
data(haddockWts)
haddockWts
```

NSeffort

Data.frame Contains information on the total effort by year and gear categories

Description

The NSeffort data.frame provides example data for use in 'plot_FOgearType'.

- ID
- Country
- Year
- Quarter
- Metier
- gear
- target
- mesh
- gearCat
- metCat
- Vessel_length
- FDF
- Area
- kw_days
- Days_at_sea
- No_vessels

Usage

```
data(NSeffort)
```

Format

```
data.frame
```

Source

Intercatch

Examples

```
data(NSlandings)
head(NSlandings)
```

NSlandings

Data.frame Contains information on the total landings by year and gear categories

Description

The NSlandings data.frame provides example data for use in ‘plot_FOgearType’ and ‘plot_catchCompSpp’.

- ID
- Country
- Year
- Quarter
- Metier
- gear
- target
- mesh
- gearCat
- metCat
- Vessel_length
- FDF
- Area
- Species
- MIXFISHspp
- Stock
- Landings
- Value

Usage

```
data(NSlandings)
```

Format

data.frame

Source

Intercatch

Examples

```
data(NSlandings)
head(NSlandings)
```

plot_alluvial	<i>Alluvial plot</i>
---------------	----------------------

Description

Alluvial plot

Usage

```
plot_alluvial(
  data,
  refTable,
  group_vars = c("fleet", "metier", "stock"),
  fill_var = c("stock"),
  group_labs = NULL,
  text_repel = FALSE,
  text_size = 2,
  mult_x = c(0.1, 0.1),
  nudge_x = 1/3,
  stratum_width = 1/3,
  stratum_col = "white",
  xlab = NULL,
  ylab = "Catch [t]",
  fillLegendTitle = "Stock",
  addLegend = TRUE,
  plotTitle = NULL
)
```

Arguments

data	data.frame Contains information on for making an alluvial-type plot (e.g. catch or landings). The data.frame should contain several variables for grouping (e.g. ‘fleet’, ‘metier’, ‘stock’) as well as a variable to plot values (‘value’).
refTable	data.frame Contains stock look-up information for consistent plotting of colors. The data.frame should contain a variable that matches the ‘fill_var’ definition (Default: ‘fill_var = "stock"’), and also found in the ‘data’ object. Contains a variable ‘col’ that defines the color used to fill bars in plot. ‘order’ defines the order of levels used in plot.

group_vars	vector. Contains variable names (order from left to right) that define the alluvial axes (Default: 'group_vars = c("fleet", "metier", "stock")')
fill_var	character. Data variable name used for coloring alluvial (Default: 'fill_var = "stock"').
group_labs	vector Contains the x-axis labels used for the group_vars. (Default: group_labs = group_vars).
text_repel	logical Should stratum labels be repelled to prevent overlapping (using 'ggrepel::geom_text_repel') (Default: 'text_repel = FALSE')
text_size	numeric Value for text label size (Default: 'text_size = 2')
mult_x	vector. Two values defining the multiplier to expand the x-axis (Default: 'mult_x = c(0.1, 0.1)'). Passes values to 'expansion(mult = mult_x)'.
nudge_x	numeric. Value to nudge direction of stratum labels when 'text_repel = TRUE'. (Default: 'nudge_x = 1/3')
stratum_width	numeric. Width of the stratum bar (Default: 'stratum_width = 1/3')
stratum_col	color definition. Color used to fill the non-stock strata (Default: 'stratum_col = "white"')
xlab	character X-axis label (Default: 'xlab = NULL' removes label)
ylab	character Y-axis label (Default: 'ylab = Catch [t]')
fillLegendTitle	character Fill legend title (Default: 'fillLegendTitle = "Stock"')
addLegend	logical Should legend of stock fill colors be added
plotTitle	character Plot title (Default: 'plotTitle = NULL' removes title)

Value

plot output of class ggplot

Examples

```
data("stfMtStkSum")
data("refTable")

## create catch data by fleet, metier, and stock
dataYr <- 2020
# add country for filter (if desired)
stfMtStkSum$country <- unlist(lapply(strsplit(stfMtStkSum$fleet, "_"),
  function(x){x[1]}))
# filter data
df <- subset(stfMtStkSum, year == 2020 & scenario == "min" &
  country == "BE")[,c("fleet", "metier", "stock", "landings")]
df$stock <- refTable$stock[match(df$stock, refTable$stock_short)]
names(df)[4] <- "value"

plot_alluvial(data = df, refTable = refTable,
  group_vars = c("fleet", "metier", "stock"),
  fill_var = c("stock"))
```

```
# doesn't make sense, but redefine group_var order
plot_alluvial(data = df, refTable = refTable,
  group_vars = c("fleet", "stock", "metier"),
  fill_var = c("stock"))

# a 2-level version
plot_alluvial(data = df, refTable = refTable,
  group_vars = c("fleet", "stock"),
  fill_var = c("stock"))

# repel labels
plot_alluvial(data = df, refTable = refTable,
  group_vars = c("fleet", "metier", "stock"),
  text_repel = TRUE, text_size = 2)

# repel labels and suppress legend
plot_alluvial(data = df, refTable = refTable,
  text_repel = TRUE, text_size = 2, addLegend = FALSE)
```

plot_catch_change	<i>Plot fleet landings taken up relative to recent landings / quota</i>
-------------------	---

Description

Plot of a fleets catch difference from the recent catches or the quota. By fleet. Most- and least-limiting stocks are also denoted. Testing in response to WKMIXFISH2.

Usage

```
plot_catch_change(
  data = NULL,
  basis = "recent_catch",
  dataYrs = NULL,
  advYr = NULL,
  sc = "min",
  fleets_excl = NULL,
  refTable = NULL,
  xlab = "Stock",
  ylab = "catch change (tonnes)",
  fillLegendTitle = "Stock",
  colLegendTitle = "Limiting stock"
)
```

Arguments

data	data.frame Contains information on catch by fleet and stock
------	---

basis	is a character vector with the basis on which to compare the scenario landings, either 'recent_catch' or 'Quota'. When 'recent_catch' is used, the average landings from the defined years (argument 'dataYrs') is used as the reference instead of the advice year quota ('Quota')
dataYrs	is a vector of years on which to base recent catches. Used when 'basis = 'recent_catch'.
advYr	is a vector of the year in which the scenario catches are generated.
sc	is a vector with the scenario to plot, e.g. "min"
fleets_excl	is a vector of fleet names not to plot, e.g. "OTH_OTH"
refTable	data.frame Contains stock look-up information for consistent plotting of stocks. 'Advice_name' defines the stock names corresponding to 'data' object. 'col' defines the color used to fill bars in plot. 'order' defines the order of stocks in the plot facets.
xlab	character X-axis label (Default: 'xlab = "Stock"')
ylab	character Y-axis label (Default: 'ylab = "KW days ('000)"')
fillLegendTitle	character Fill legend title (Default: 'fillLegendTitle = "Effort stock"')
colLegendTitle	character Color legend title (Default: 'colLegendTitle = "Limiting stock"')

Details

Users will need to provide the data and reference table objects to produce the plot.

Value

plot output of class ggplot

Examples

```
# make example data
data(refTable) # reference table with stock advice names, colors, order, etc.
data(stfFltStkSum) # summary of fleet/stock-related catch variables
advYr <- 2022 # advice year

# replace short stock names with ICES stock codes
stfFltStkSum$stock <- refTable$stock[match(stfFltStkSum$stock,
  refTable$stock_short)]

p <- plot_catch_change(data = stfFltStkSum,
  basis = "Quota",
  dataYrs = 2020:2022,
  advYr = advYr,
  sc = "min",
  fleets_excl = "OTH_OTH",
  refTable = refTable,
  xlab = "Stock",
  ylab = "landings change (tonnes)",
```

```

fillLegendTitle = "Stock",
colLegendTitle = "Limiting stock")

print(p)

# export plot
# png("plot_change.png", width = 8, height = 10, units = "in", res = 400)
# print(p); dev.off()

```

plot_catchComp

Plot landings or catch compositions

Description

Landings or catch compositions by stock for selected years, countries, fleets, metiers etc

Usage

```

plot_catchComp(
  data,
  refTable,
  filters = NULL,
  selectors = "metier",
  divider = NULL,
  yvar = "landings",
  tryNumericSelector = FALSE,
  flipAxes = FALSE,
  relative = TRUE
)

```

Arguments

- | | |
|----------|--|
| data | data.frame Contains information on fleet data to make catch (landings) compositions. Required variables are: 'year', 'area', 'country', 'fleet', 'metier', 'stock', 'landings', 'catch', and 'fleet_type' which indicates if the 'fleet' is a 'main' or 'residual' fleet. |
| refTable | <p>data.frame A look-up reference table for stocks and associated attributes. The refTable data.frame lists stock names and corresponding colours for consistency across plots. To be used as a look-up table in converting between variable stock names and printed ones.</p> <ul style="list-style-type: none"> • 1) stock - ICES stock codes used in advice • 2) order - stock order to be used in plots • 3) col - stock colors for plots (e.g. pals::brewer.paired()) • 4) stock_short - short stock name used in mixed fishery model |

filters	list of character strings listing the 'year', 'area', 'country', 'fleet' and/or 'metier' to filter from data. Default value of NULL will produce catch compositions using all data in data.
selectors	character string of one of 'year', 'area', 'country', 'fleet' or 'metier'. The chosen selector will be plotted on the x-axis. Multiple variables can be listed as selectors and these will be concatenated into a "label" for plotting. The default value is metier and will produce catch compositions by 'metier'.
divider	character string of one of 'year', 'area', 'country', 'fleet' or 'metier'. Only one variable can be listed as a 'divider'. The chosen divider will be used to divide the catch compositions into subplots - e.g. one per 'fleet'. The default value of NULL will plot just one catch composition (i.e. no subplots).
yvar	character string of variable to be plotted on the y-axis (Default: yvar = "landings")
tryNumericSelector	logical. Should the selector variable be converted to a numeric variable. Likely only makes sense when 'selector = "year"' (Default: 'tryNumericSelector = FALSE')
flipAxes	logical. Should 'coord_flip' be applied to flip x- and y-axes. (Default: 'flipAxes = FALSE')
relative	logical. Should composition be presented in relative terms (Default: 'relative = TRUE')

Details

Users will need to provide the data and refTable objects to produce the plot.

Value

plot output of class ggplot

Examples

```
# prepare example data
data(refTable)
data(stfMtStkSum)

# subset data to a single scenario (e.g. min)
data <- subset(stfMtStkSum, scenario == "min")

# add country and area identifiers (if desired)
tmp <- strsplit(data$metier, ".", fixed = TRUE)
data$area <- unlist(lapply(tmp, FUN = function(x){ifelse(length(x)==2, x[2], NA)}))
tmp <- strsplit(data$fleet, "_", fixed = TRUE)
data$country <- unlist(lapply(tmp, FUN = function(x){ifelse(length(x)==2, x[1], NA)}))

# replace stock with ICES stock code
data$stock <- refTable$stock[match(data$stock, refTable$stock_short)]
```

```

# Plot catch composition for each fleet over time
selectors <- c("year")
divider <- c("fleet")
p <- plot_catchComp(data, refTable, filters = NULL,
  selectors = selectors, divider = divider, yvar = "catch")
print(p)

# ggplot format adjustments
p2 <- p + theme(text = element_text(size = 8),
  axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  facet_wrap(divider, scales = "fixed") # remove free axes
print(p2)

# Remove relative scaling, and treat year axis (selector) as numeric
plot_catchComp(data, refTable, filters = NULL, selectors, divider,
  yvar = "catch", relative = FALSE, tryNumericSelector = TRUE) +
  theme(text = element_text(size = 8),
    axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  facet_wrap(divider, ncol = 6, scales = "free_y")

# export plot
# png("catchComp1.png", width = 7, height = 7, units = "in", res = 400)
# print(p2); dev.off()

# lot landings composition for each area by country-metier combinations
selectors <- c("country", "metier")
divider <- c("area")
p <- plot_catchComp(data, refTable, filters=NULL,
  selectors, divider, relative = TRUE, flipAxes = TRUE)
print(p)

# plot landings composition for each metier by country for 2022
filters <- list(year = 2022)
selectors <- c("metier")
divider <- c("country")
plot_catchComp(data, refTable, filters, selectors, divider, flipAxes = TRUE)

# plot landings compositions for each fleet by metier for Scottish fleets.
filters <- list(year=2022, country="SC")
selectors <- c("metier")
divider <- c("fleet")
plot_catchComp(data, refTable, filters, selectors, divider, flipAxes = TRUE)

```

Description

landings compositions of species landed by country and metier category

Usage

```
plot_catchCompSpp(
  data,
  refTable,
  land_threshold = 10,
  fillLegendTitle = "Species"
)
```

Arguments

data	data.frame Contains information on the total landings of species listed in the WGMIXFISH data call by country and metier category. Required variables are: 'country' (2 letter country code), 'metCat' (which indicates the metier categories), 'spp' (which indicates the 3 letter FAO species code as listed in the WGMIXFISH data call), and 'landings'.
refTable	data.frame Contains stock look-up information for consistent plotting of stocks. 'spp' defines the species names corresponding to 'data' object. 'col' defines the color used to fill bars in plot. 'order' defines the order of stocks in the plot facets. 'label' = label to use in the fill color legend
land_threshold	numeric A number, in tonnes, to set a minimum landings threshold. Metier-country combinations with total landings below this threshold will not be plotted. Default value is 10 tonnes.
fillLegendTitle	character Fill legend title (Default: 'fillLegendTitle = "Species"')

Details

Users will need to provide the data to produce the plot.

Value

plot output of class ggplot

Examples

```
# Landings - prepare example data
data(NSlandings)

dat <- NSlandings

# rename
names(dat)[which(names(dat)=="MIXFISHspp")] <- "spp"

# aggregate some metiers
```

```

dat <- dat %>% mutate(metCat = if_else(metCat %in%
  c("dredges", "beam_oth"), "MIS", metCat))
dat <- dat %>% mutate(
  metCat = if_else(metCat %in%
    c("Dseine_TR3", "Dseine_TR1", "Dseine_TR2", "Dseine_oth"),
    "Demersal seine", metCat))

# re-aggregate
datayear <- 2023
dat <- dat %>% filter(Year %in% datayear) %>%
  group_by(country=Country, metCat, spp) %>%
  summarise(landings=sum(Landings))

# Set landings threshold
land_th <- 10 # 10 tonnes.

# group some species together to make plotting easier
dat <- dat %>% mutate(spp = if_else(spp %in%
  c("ANK", "MON"), "ANF", spp))
dat <- dat %>% mutate(spp = if_else(spp %in%
  c("MEG", "LDB"), "LEZ", spp))
dat <- dat %>% mutate(spp = if_else(spp %in%
  c("GUG", "GUR"), "Gurnards", spp))

# remove less important species from data call to make plotting easier
dat <- dat %>% filter(!spp %in%
  c("OTH", "CAA", "COE", "HAL", "HER", "MAC", "WHB", "HOM", "SPR"))

# re-aggregate
dat <- dat %>% group_by(country, metCat, spp) %>% summarise(landings=sum(landings))

# make color lookup data.frame (refTable)
COLS <- c(brewer.pal(n = 9, name = "YlOrBr")[c(2,4)],
  brewer.pal(n = 9, name = "Oranges")[c(6)],
  brewer.pal(n = 9, name = "Reds")[c(8)],
  brewer.pal(n = 11, name = "PRGn")[c(4,3,2,1)],
  brewer.pal(n = 9, name = "PiYG")[c(4,2,1)],
  brewer.pal(n = 9, name = "Blues")[c(4,6,8,9)],
  brewer.pal(n = 11, name = "BrBG")[c(7:10)],
  brewer.pal(n = 9, name = "Greens")[c(3,6,8)],
  brewer.pal(n = 11, name = "BrBG")[c(1:4)],
  brewer.pal(n = 9, name = "Greys")[c(3,5,7)])

# plot
# possible colors
COLS <- c(brewer.pal(n = 9, name = "YlOrBr")[c(2,4)],
  brewer.pal(n = 9, name = "Oranges")[c(6)],
  brewer.pal(n = 9, name = "Reds")[c(8)],
  brewer.pal(n = 11, name = "PRGn")[c(4,3,2,1)],
  brewer.pal(n = 9, name = "PiYG")[c(4,2,1)],
  brewer.pal(n = 9, name = "Blues")[c(4,6,8,9)],

```

```

brewer.pal(n = 11, name = "BrBG")[c(7:10)],
brewer.pal(n = 9, name = "Greens")[c(3,6,8)],
brewer.pal(n = 11, name = "BrBG")[c(1:4)],
brewer.pal(n = 9, name = "Greys")[c(3,5,7)]

refTable <- as.data.frame(rbind(
  c("ANF", "Anglerfishes"),
  c("BLL", "Brill"),
  c("COD", "Cod"),
  c("DAB", "Dab"),
  c("FLE", "Flounder"),
  c("Gurnards", "Gurnards"),
  c("HAD", "Haddock"),
  c("LEM", "Lemon sole"),
  c("LIN", "Ling"),
  c("PLE", "Plaice"),
  c("SKA", "Skates and rays"),
  c("SOL", "Sole"),
  c("TUR", "Turbot"),
  c("WHG", "Whiting"),
  c("HKE", "Hake"),
  c("POK", "Saithe"),
  c("POL", "Pollack"),
  c("WIT", "Witch"),
  c("LEZ", "Megrim"),
  c("NEP", "Norway lobster"),
  c("SDV", "Smoothhounds"),
  c("NOP", "Norway pout")
))
names(refTable) <- c("spp", "label")
refTable$col <- COLS[seq(nrow(refTable))]
refTable$order <- seq(nrow(refTable))

plot_catchCompSpp( data = dat, refTable = refTable, land_threshold=land_th)

```

plot_catchCompStk *Plot landings or catch compositions*

Description

Landings or catch compositions by stock for selected years, countries, fleets, metiers etc

Usage

```

plot_catchCompStk(
  data,

```

```

    refTable,
    filters = NULL,
    categories = "fleet",
    split = "stock",
    facet = NULL,
    yvar = "landings"
)

```

Arguments

data	data.frame Contains information on fleet data to make catch (landings) compositions. Required variables are: 'year', 'fleet', 'stock', 'landings'. Additional variables, if required, include: 'area', 'country', 'catch', 'metier' and 'value'.
refTable	data.frame A look-up reference table for stocks and associated attributes. The refTable data.frame lists stock names and corresponding colours for consistency across plots. To be used as a look-up table in converting between variable stock names and printed ones. • 1) stock - ICES stock codes used in advice • 2) order - stock order to be used in plots • 3) col - stock colors for plots (e.g. pals::brewer.paired()) • 4) stock_short - short stock name used in mixed fishery model
filters	list of character strings listing the 'year', 'area', 'country', 'fleet' and/or 'metier' to filter from data. Default value of NULL will produce catch compositions using all data in data.
categories	character string of one of 'year', 'area', 'country', 'fleet', 'metier' or 'stock'. The chosen category will be plotted on the x-axis. Multiple variables can be listed as categories and these will be concatenated into a "label" for plotting. The default value is metier and will produce catch compositions by 'metier'.
split	character string of one of 'year', 'area', 'country', 'fleet', 'metier' or 'stock'. The chosen split will be used to divide the bar chart based on relative proportions. The default value is stock and will produce compositions by 'stock'.
facet	character string of one of 'year', 'area', 'country', 'fleet' or 'metier'. Only one variable can be listed as a 'facet'. The chosen facet will be used to divide the catch compositions into subplots - e.g. one per 'fleet'. The default value of NULL will plot just one catch composition (i.e. no subplots).
yvar	character string of variable (e.g. 'landings' or 'value') to be plotted on the y-axis (Default: yvar = "landings")

Details

Users will need to provide the data and refTable objects to produce the plot.

Value

plot output of class ggplot

Examples

```

# prepare example data
data(refTable)
data(stfMtStkSum)

# subset data to a single scenario (e.g. min)
data <- subset(stfMtStkSum, scenario == "min")

# add country and area identifiers (if desired)
tmp <- strsplit(data$metier, ".", fixed = TRUE)
data$area <- unlist(lapply(tmp, FUN = function(x){ifelse(length(x)==2, x[2], NA)}))
tmp <- strsplit(data$fleet, "_", fixed = TRUE)
data$country <- unlist(lapply(tmp, FUN = function(x){ifelse(length(x)==2, x[1], NA)}))

# replace stock with ICES stock code
data$stock <- refTable$stock[match(data$stock, refTable$stock_short)]

# Plot catch composition for each fleet over time
categories <- c("year")
split <- c("stock")
facet <- c("fleet")
p <- plot_catchCompStk(data, refTable, filters = NULL, categories, split, facet, yvar = "catch")
print(p)

# ggplot format adjustments
p2 <- p + theme(text = element_text(size = 8),
  axis.text.x = element_text(angle = 90, vjust = 0, hjust=1)) +
  facet_wrap(facet, scales = "fixed") # remove free axes
print(p2)

# export plot
# png("catchComp1.png", width = 7, height = 7, units = "in", res = 400)
# print(p2); dev.off()

# Plot landings composition for each area by country-metier combinations
categories <- c("country", "metier")
split <- c("stock")
facet <- c("area")
p <- plot_catchCompStk(data, refTable, filters=NULL, categories, split, facet)
print(p)

# Plot landings composition for each metier by country for 2022
filters <- list(year = 2022)
categories <- c("metier")
split <- c("stock")
facet <- c("country")
plot_catchCompStk(data, refTable, filters, categories, split, facet)

# Plot landings compositions for each fleet by metier for Scottish fleets.

```

```

filters <- list(year=2022, country="SC")
categories <- c("metier")
split <- c("stock")
facet <- c("fleet")
plot_catchCompStk(data,refTable,filters, categories, split, facet)

```

plot_catchScenStk *Headline advice plot*

Description

Plot summarizing over- and under-quota catches by stock and scenario. Dashed line displays quota by stock. Colored background further emphasizes over- and under-quota catches. Used as the headline plot in WGMIXFISH-ADVICE.

Usage

```

plot_catchScenStk(
  data,
  adv,
  ofwhich = FALSE,
  xlab = "Scenario",
  ylab = "Catch [t]"
)

```

Arguments

data	data.frame Contains catch ('catch') by scenario ('scenario') and stock ('stock').
adv	data.frame Contains advice ('advice') by stock ('stock'). Optional upper ('upper') and lower ('lower') advice limits can be included.
ofwhich	logical. If TRUE an of which limit will be plotted. Requires a 'catch_ofwhich' column in data and an 'advice_ofwhich' column in adv.
xlab	character X-axis label (Default: 'xlab = "Scenario"')
ylab	character Y-axis label (Default: 'ylab = "Catch [t]"')

Value

plot output of class ggplot

Examples

```

# make example data
data(stfFltStkSum)
head(stfFltStkSum)

# subset data to advice year and restrictive stocks
advYr <- 2022 # advice year
restr.stks <- c("COD-NS", "HAD", "PLE-EC", "PLE-NS", "POK", "SOL-EC",
  "SOL-NS", "TUR", "WHG-NS", "WIT")
stfFltStkSum <- subset(stfFltStkSum, year == advYr & stock %in% restr.stks)

# data for plotting (catch by scenario and stock)
catchScenStk <- aggregate(catch ~ scenario + stock, data = stfFltStkSum,
  FUN = sum)

# re-order scenarios (sq_E, max, min, ... )
catchScenStk$scenario <- factor(catchScenStk$scenario,
  levels = c("min", "max", "sq_E", "cod-ns"),
  labels = c("min", "max", "sq_E", "cod-ns"))
head(catchScenStk)

catchRange <- rbind(
  data.frame(stock = "COD-NS", advice = 14276, lower = 9701, upper = 14276),
  data.frame(stock = "HAD", advice = 128708, lower = 111702, upper = 128708),
  data.frame(stock = "PLE-EC", advice = 6365, lower = 4594, upper = 6365),
  data.frame(stock = "PLE-NS", advice = 142507, lower = 101854,
    upper = 195622),
  data.frame(stock = "POK", advice = 49614, lower = 30204, upper = 49614),
  data.frame(stock = "SOL-EC", advice = 1810, lower = 1068, upper = 2069),
  data.frame(stock = "SOL-NS", advice = 15330, lower = 9523, upper = 21805),
  data.frame(stock = "TUR", advice = 3609, lower = 2634, upper = 4564),
  data.frame(stock = "WHG-NS", advice = 88426, lower = 70169, upper = 91703),
  data.frame(stock = "WIT", advice = 1206, lower = 875, upper = 1206)
)

# use ICES stock codes
data(refTable)
head(refTable)
catchScenStk$stock <- refTable$stock[match(catchScenStk$stock,
  refTable$stock_short)]
catchRange$stock <- refTable$stock[match(catchRange$stock,
  refTable$stock_short)]

# plot without range
p <- plot_catchScenStk(data = catchScenStk, adv = catchRange[,1:2])
print(p)

# plot with range
p <- plot_catchScenStk(data = catchScenStk, adv = catchRange)
print(p)

```

```
# export plot
# png("catchScenStk1.png", width = 6, height = 5, units = "in", res = 400)
# print(p); dev.off()
```

plot_effortFltStk	<i>Plot fleet effort to uptake stock quotas</i>
-------------------	---

Description

Plot of effort required to uptake each stock's quota by fleet. Most- and least-limiting stocks are also denoted. Used in WGMIXFISH-ADVICE.

Usage

```
plot_effortFltStk(
  data,
  refTable,
  xlab = "Stock",
  ylab = "Effort ['000 KW days]",
  fillLegendTitle = "Stock",
  colLegendTitle = "Limiting stock",
  linewidthDefault = 0.5,
  linewidthLimitation = 1
)
```

Arguments

data	data.frame Contains information on effort required to uptake quotas by fleet and stock, plus designation of each stock's limitation status to the fleet's fishing effort. Stock variable names ('stock') should match those of refTable . Other required variables include: 'Limitation' - defines, by fleet, the most- ('most'), least- ('least'), and intermediate-limiting ('NA') stocks; 'quotaEffort' - the effort, by fleet, required to take up the quota share of each stock; 'sqEffort' - status quo effort corresponding to most recent data year before forecast.
refTable	data.frame Contains stock look-up information for consistent plotting of stocks. 'stock' defines the stock names corresponding to 'data' object. 'col' defines the color used to fill bars in plot. 'order' defines the order of stocks in the plot facets.
xlab	character X-axis label (Default: 'xlab = "Stock"')
ylab	character Y-axis label (Default: 'ylab = "Effort ['000 KW days]"')
fillLegendTitle	character Fill legend title (Default: 'fillLegendTitle = "Effort stock"')
colLegendTitle	character Color legend title (Default: 'colLegendTitle = "Limiting stock"')
linewidthDefault	numeric Value for default line width of intermediate- limiting stocks (Default: 'linewidthDefault = 0.5')

linewidthLimitation

numeric Value for default line width of limiting stocks (Default: 'linewidthLimitation = 1')

Details

Users will need to provide the data and reference table objects to produce the plot. In the best case, effort associated with complete quota uptake by fleet ('data\$quotaEffort') may be derived from scenarios restricting fleet catch one stock at a time. In the following example, however, effort levels are derived by linearly extrapolating the quota uptake levels by the effort of the "min" scenario. This is strictly linear when quotas are based on partial F, as in FCube. In FLBEIA, quotas are based on catch (or landings), which may deviate from a linear relationship when a stock is close full exploitation (should not result from an ICES harvest control rule).

Value

plot output of class ggplot

Examples

```
# example data for plot_effortFltStk -----

data(refTable) # reference table with stock advice names, colors, order, etc.
data(stfFltSum) # summary of fleet-related variables (e.g. effort)
data(stfFltStkSum) # summary of fleet/stock-related catch variables

## get data from advice year

# catches by fleet and stock
advYr <- 2022 # advice year
df <- subset(stfFltStkSum, scenario == "min" & year == advYr)

## effort by fleet and scenario
eff <- subset(
  stfFltSum, scenario == "min" & year == advYr)[,c("fleet", "effort")]
sqEff <- subset(
  stfFltSum, scenario == "sq_E" & year == advYr)[,c("fleet", "effort")]
names(sqEff)[2] <- "sqEffort"
eff <- merge(x = eff, y = sqEff, all.x = TRUE)
df <- merge(x = df, y = eff, all.x = TRUE)
df$quotaEffort <- df$effort / df$quotaUpt

## Determine most- and least-limiting stock by fleet
# restrictive stocks
restr.stks <- c("COD-NS", "HAD", "PLE-EC", "PLE-NS", "POK", "SOL-EC",
  "SOL-NS", "TUR", "WHG-NS", "WIT", "NEP6", "NEP7", "NEP8", "NEP9")
fls <- unique(df$fleet)
df2 <- vector("list", length(flfs))
names(df2) <- fls
for(i in seq(flfs)){
  tmp <- subset(df, fleet == fls[i])
  tmp$Limitation <- NA # initial NA setting for all stocks
```

```

# most-limiting (highest quota uptake in min scenario)
mostLimStk <- subset(tmp, stock %in% restr.stks)
mostLimStk <- mostLimStk$stock[which.max(mostLimStk$quotaUpt)]
tmp$Limitation[which(tmp$stock == mostLimStk)] <- "most"

# least-limiting (lowest quota uptake in max scenario)
leastLimStk <- subset(stfFltStkSum, scenario == "max" & year == advYr &
  fleet == fls[i] & stock %in% restr.stks)
leastLimStk <- leastLimStk$stock[which.min(leastLimStk$quotaUpt)]
tmp$Limitation[which(tmp$stock == leastLimStk)] <- "least"

# return result
df2[[i]] <- tmp
}
df2 <- do.call("rbind", df2)

# replace short stock names with ICES stock codes
df2$stock <- refTable$stock[match(df2$stock, refTable$stock_short)]

# plot
p <- plot_effortFltStk(data = df2, refTable = refTable)
# png("effortFltStk1.png", width = 8, height = 10, units = "in", res = 400)
# print(p); dev.off()

# adjust ggplot2 settings
p <- p + theme(text = element_text(size = 12))
# png("effortFltStk2.png", width = 8, height = 10, units = "in", res = 400)
# print(p); dev.off()

```

plot_F0gearType

Plot total landings or effort by broad gear categories

Description

total landings or effort by broad gear categories such as 'otter trawls', 'beam trawls'

Usage

```
plot_F0gearType(data)
```

Arguments

data	data.frame Contains information on the total landings or effort by year and gear categories. Required variables are: 'year', 'gearType' (which indicates the broad gear categories), 'value' (which indicates the total to be plotted), and 'dataType' which indicates the type of data to be plotted (either "landings", "days at sea" or "KW days").
------	--

Details

Users will need to provide the data to produce the plot.

Value

plot output of class ggplot

Examples

```
# Landings - prepare example data
data(NSlandings)

dat <- NSlandings
dat <- rename(dat, gearType=gearCat)
dat <- rename(dat, year=Year)

# aggregate landings data
dat <- dat %>% group_by(year, gearType) %>%
  summarise(value=sum(Landings, na.rm=TRUE))
dat$dataType <- "landings"

# plot
# windows()
plot_F0gearType(dat)

# Effort - prepare example data
data(NSeffort)

dat <- NSeffort
dat <- rename(dat, gearType=gearCat)
dat <- rename(dat, year=Year)
dat <- dat %>% group_by(year, gearType) %>%
  summarise(value=sum(Days_at_sea, na.rm=TRUE))
dat$dataType <- "days at sea"

# plot
# windows()
plot_F0gearType(dat)
```

plot_landByMetStock *Bar chart of landings by stock and metier*

Description

Bar chart of landings by stock and by metier/gear groupings. Used in WGMIXFISH-ADVICE

Usage

```
plot_landByMetStock(
  data,
  refTable,
  xlab = "",
  ylab = "Landings ['000 t]",
  fillLegendTitle = "Stock"
)
```

Arguments

data	data.frame Contains information on the landings (or catch) in tonnes by stock and metier/gear grouping from the fleet data used at WGMIXFISH-ADVICE. Stock variable names ('stock') should match those of refTable .
refTable	data.frame A look-up reference table for stocks and associated attributes. The refTable data.frame lists stock names and corresponding colours for consistency across plots. To be used as a look-up table in converting between variable stock names and printed ones. • 1) stock - ICES stock codes used in advice • 2) order - stock order to be used in plots • 3) col - stock colors for plots (e.g. <code>pals::brewer.paired()</code>) • 4) stock_short - short stock name used in mixed fishery model
xlab	character X-axis label (Default (blank): 'xlab = ""')
ylab	character Y-axis label (Default: 'ylab = "Landings [t]"')
fillLegendTitle	character Fill legend title Other required variables include: 'metier' which defines the metier code or gear grouping code; 'value' the value of landings (or catch) for each 'stock' and 'metier'

Details

Users will need to provide the data object to produce the plot.

Value

plot output of class ggplot

Examples

```
# make example data
data(stfMtStkSum)
head(stfMtStkSum)
data(refTable)
head(refTable)

data <- stfMtStkSum
```

```

# add metier_cat
tmp <- strsplit(data$metier, ".", fixed = TRUE)
data$metier_cat <- unlist(lapply(tmp, FUN = function(x){x[1]}))

# select final data year and a single scenario, and aggregated total landings
# by stock and metier
datYr <- 2020
data <- subset(data, year == datYr & scenario == "min")
agg <- aggregate(landings ~ metier_cat + stock, data, FUN = sum, na.rm = TRUE)

# In the North Sea model, all Nephrops FUs area aggregated together
agg$isNEP <- seq(nrow(agg)) %in% grep("NEP", agg$stock)
agg1 <- subset(agg, !isNEP)[,c(1:3)]
agg2 <- aggregate(landings ~ metier_cat, data = subset(agg, isNEP),
  FUN = sum, na.rm = TRUE)
agg2$stock <- "Nephrops"
agg <- merge(agg1, agg2, all = TRUE)
agg <- agg[,c("stock", "metier_cat", "landings")]

names(agg) <- c("stock", "metier", "value")
agg

# subset included metiers
metIncl <- c("TR1", "TR2", "BT1", "BT2", "GN1", "GT1", "LL1", "beam_oth",
  "pots", "OTH", "MIS")
agg <- subset(agg, metier %in% metIncl)

# replace stock with ICES stock code
agg$stock <- refTable$stock[match(agg$stock, refTable$stock_short)]

plot_landByMetStock(data = agg, refTable)

```

plot_landByStock	<i>Pie chart of landings by stock</i>
------------------	---------------------------------------

Description

Pie chart of landings by stock. Used in WGMIXFISH-ADVICE

Usage

```

plot_landByStock(
  data,
  refTable,
  ylab = "Landings [t]",
  fillLegendTitle = "Stock"
)

```

Arguments

data	data.frame Contains information on the stocks to include and their landings (or catch) to plot. Stock variable names ('stock') should match those of refTable . Other required variables include: 'value' the value of landings (or catch) for each stock; and 'col' which defines the fill colour as a hex colour code, by stock, to be used.
refTable	data.frame A look-up reference table for stocks and associated attributes. The refTable data.frame lists stock names and corresponding colours for consistency across plots. To be used as a look-up table in converting between variable stock names and printed ones. • 1) stock - ICES stock codes used in advice • 2) order - stock order to be used in plots • 3) col - stock colors for plots (e.g. <code>pals::brewer.paired()</code>) • 4) stock_short - short stock name used in mixed fishery model
ylab	character Y-axis label (Default: 'ylab = "Landings [t]"')
fillLegendTitle	character Fill legend title

Details

Users will need to provide the data object to produce the plot.

Value

plot output of class ggplot

Examples

```
# make example data
data(stfFltStkSum)
head(stfFltStkSum)

data(refTable)
head(refTable)

# select final data year and a single scenario, and aggregated total landings
datYr <- 2020
dat <- subset(stfFltStkSum, year == datYr & scenario == "min")
agg <- aggregate(landings ~ stock, dat, sum, na.rm = TRUE)

# In the North Sea model, all Nephrops FUs area aggregated together
agg$isNEP <- seq(nrow(agg)) %in% grep("NEP", agg$stock)

agg <- rbind(subset(agg, !isNEP)[,c(1:2)],
  data.frame(stock = "Nephrops", landings = sum(subset(agg, isNEP)$landings)))

# replace stock with ICES stock code
agg$stock <- refTable$stock[match(agg$stock, refTable$stock_short)]
```

```
names(agg) <- c("stock", "value")
agg

plot_landByStock(data = agg, refTable)
```

plot_overUnderFltStk *Plot over- and undershoot of stock quotas by fleet*

Description

Plot of over- and undershoot of each stock's quota by fleet. Most- and least-limiting stocks are also denoted.

Usage

```
plot_overUnderFltStk(
  data,
  refTable,
  yExt = 0.3,
  xlab = "Stock",
  ylab = "Predicted catch [t] with advice undershoot (negative extent)",
  borderSize = 0.5,
  fillLegendTitle = "Stock",
  colLegendTitle = "Limiting stock"
)
```

Arguments

data	data.frame Contains information on effort required to uptake quotas by fleet and stock, plus designation of each stock's limitation status to the fleet's fishing effort. Stock variable names ('Advice_name') should match those of refTable . Other required variables include: 'Limitation' - defines, by fleet, the most- ('most'), least- ('least'), and intermediate-limiting ('NA') stocks; 'quotaEffort' - the effort, by fleet, required to take up the quota share of each stock; 'sqEffort' - status quo effort corresponding to most recent data year before forecast.
refTable	data.frame Contains stock look-up information for consistent plotting of stocks. 'Advice_name' defines the stock names corresponding to 'data' object. 'col' defines the color used to fill bars in plot. 'order' defines the order of stocks in the plot facets.
yExt	Fraction of absolute range to extend y-axis for each fleet facet (Default: yExt = 0.3).
xlab	character X-axis label (Default: xlab = "Stock")
ylab	character Y-axis label (Default: ylab = "Predicted catch [t] with advice undershoot (negative extent)")
borderSize	line width of border around bars (Default: borderSize=0.5)

fillLegendTitle

character Fill legend title (Default: 'fillLegendTitle = "Stock"')

colLegendTitle character Color legend title (Default: 'colLegendTitle = "Limiting stock"')

Details

Users will need to provide the data and reference table objects to produce the plot. In the best case, effort associated with complete quota uptake by fleet ('data\$effortQuota') may be derived from scenarios restricting fleet catch one stock at a time. In the following example, however, effort levels are derived by linearly extrapolating the quota uptake levels by the effort of the "min" scenario. This is strictly linear when quotas are based on partial F, as in FCube. In FLBEIA, quotas are based on catch (or landings), which may deviate from a linear relationship when a stock is close full exploitation (should not result from an ICES harvest control rule).

Value

plot output of class ggplot

Examples

```
# example data for plot_effortFltStk -----

data(refTable) # reference table with stock advice names, colors, order, etc.
data(stfFltSum) # summary of fleet-related variables (e.g. effort)
data(stfFltStkSum) # summary of fleet/stock-related catch variables

## get data from advice year

# catches by fleet and stock
advYr <- 2022 # advice year
df <- subset(stfFltStkSum, scenario == "min" & year == advYr)

## effort by fleet and scenario
eff <- subset(
  stfFltSum, scenario == "min" & year == advYr)[,c("fleet", "effort")]
sqEff <- subset(
  stfFltSum, scenario == "sq_E" & year == advYr)[,c("fleet", "effort")]
names(sqEff)[2] <- "sqEffort"
eff <- merge(x = eff, y = sqEff, all.x = TRUE)
df <- merge(x = df, y = eff, all.x = TRUE)
df$quotaEffort <- df$effort / df$quotaUpt

## Determine most- and least-limiting stock by fleet
# restrictive stocks
restr.stks <- c("COD-NS", "HAD", "PLE-EC", "PLE-NS", "POK", "SOL-EC",
  "SOL-NS", "TUR", "WHG-NS", "WIT", "NEP6", "NEP7", "NEP8", "NEP9")
fls <- unique(df$fleet)
df2 <- vector("list", length(fls))
names(df2) <- fls
for(i in seq(fls)){
  tmp <- subset(df, fleet == fls[i])
```

```

tmp$Limitation <- NA # initial NA setting for all stocks

# most-limiting (highest quota uptake in min scenario)
mostLimStk <- subset(tmp, stock %in% restr.stks)
mostLimStk <- mostLimStk$stock[which.max(mostLimStk$quotaUpt)]
tmp$Limitation[which(tmp$stock == mostLimStk)] <- "most"

# least-limiting (lowest quota uptake in max scenario)
leastLimStk <- subset(stfFltStkSum, scenario == "max" & year == advYr &
  fleet == fls[i] & stock %in% restr.stks)
leastLimStk <- leastLimStk$stock[which.min(leastLimStk$quotaUpt)]
tmp$Limitation[which(tmp$stock == leastLimStk)] <- "least"

# return result
df2[[i]] <- tmp
}
df2 <- do.call("rbind", df2)

# replace short stock names with ICES stock codes
df2$stock <- refTable$stock[match(df2$stock, refTable$stock_short)]

# plot
p <- plot_overUnderFltStk(data = df2, refTable = refTable)
p
# png("overUnderFltStk1.png", width = 8, height = 10, units = "in", res = 400)
# print(p); dev.off()

# adjust ggplot2 settings
p <- p + theme(text = element_text(size = 12))
p
# png("overUnderFltStk2.png", width = 8, height = 10, units = "in", res = 400)
# print(p); dev.off()

```

plot_relEffortFltStk *Relative fleet effort to uptake stock quotas*

Description

Plot of relative effort required to uptake each stock's quota by fleet. To be used in fishery overviews.

Usage

```

plot_relEffortFltStk(
  data,
  limits = c(-100, 100),
  xlab = "Stock",
  ylab = "Fleet",

```

```
    fillLegendTitle = "Variation\n in effort"
  )
```

Arguments

<code>data</code>	data.frame Contains information on relative effort (to status quo effort, 'var') required to uptake quotas by fleet ('fleet') and stock ('scenario').
<code>limits</code>	vector Two value vector with lower and upper limits for fill colors (Default: 'limits = c(-100,100)')
<code>xlab</code>	character X-axis label (Default: 'xlab = "Stock"')
<code>ylab</code>	character Y-axis label (Default: 'ylab = "Fleet"')
<code>fillLegendTitle</code>	character Fill legend title (Default: 'fillLegendTitle = "Variation in effort"')

Details

Users will need to provide the data and reference table objects to produce the plot. In the best case, effort associated with complete quota uptake by fleet may be derived from scenarios restricting fleet catch one stock at a time. In the following example, however, effort levels are derived by linearly extrapolating the quota uptake levels by the effort of the "min" scenario. This is strictly linear when quotas are based on partial F, as in FCube. In FLBEIA, quotas are based on catch (or landings), which may deviate from a linear relationship when a stock is close full exploitation (should not result from an ICES harvest control rule).

Value

plot output of class ggplot

Examples

```
# make data
data(refTable) # reference table with stock advice names, colors, order, etc.
data(stfFltSum) # summary of fleet-related variables (e.g. effort)
data(stfFltStkSum) # summary of fleet/stock-related catch variables

## get data from advice year
advYr <- 2022 # advice year
df <- subset(stfFltStkSum, scenario == "min" & year == advYr)

eff <- subset(
  stfFltSum, scenario == "min" & year == advYr)[,c("fleet", "effort")]
sqEff <- subset(
  stfFltSum, scenario == "sq_E" & year == advYr)[,c("fleet", "effort")]
names(sqEff)[2] <- "sqEffort"
eff <- merge(x = eff, y = sqEff, all.x = TRUE)
df <- merge(x = df, y = eff, all.x = TRUE)
df$quotaEffort <- df$effort / df$quotaUpt
df$relEffort <- df$quotaEffort / df$sqEffort

# df$scenario <- df$stock
```

```

restr.stks <- c("COD-NS", "HAD", "PLE-EC", "PLE-NS", "POK", "SOL-EC",
               "SOL-NS", "TUR", "WHG-NS", "WIT", "NEP6", "NEP7", "NEP8", "NEP9")
df <- subset(df, stock %in% restr.stks)

# replace short stock names with ICES stock codes
df$stock <- refTable$stock[match(df$stock, refTable$stock_short)]

# adjust stock order for the plot
df$stock <- factor(df$stock, levels = refTable$stock)

# convert to percentage change
df$var <- 100*(df$relEffort-1)

# optional upper limit (e.g. 100)
df$var <- ifelse(df$var > 100, 100, df$var)

# plot
p <- plot_relEffortFltStk(data = df)
print(p)

# export plot
# png("relEffortFltStk1.png", width = 4, height = 6, units = "in", res = 400)
# print(p); dev.off()

```

plot_stkMetProp

Plot proportion of stock landings by metier

Description

landings proportions of each stock associated with different metiers

Usage

```
plot_stkMetProp(data, text_size = 9, face = "plain")
```

Arguments

data	data.frame Contains information on the stock, metiers and landings proportions. Required variables are: 'Stock' (numeric; numbers the stocks in alphabetical order), 'metCat' (which indicates the metier categories), 'prop_stock' (which indicates proportion of total stock landings associated with each metier category), and 'label' (which indicates the ICES stock code).
text_size	size of text
face	an aesthetic or parameter inside element_text() that controls the font style of text elements like titles, axis labels, and legends.

Details

Users will need to provide the data to produce the plot.

Value

plot output of class ggplot

Examples

```
# Landings - prepare example data
data(stock_props)
# plot
plot_stkMetProp(stock_props)
```

refTable

Look-up reference table for stocks and associated attributes

Description

The refTable data.frame lists stock names and corresponding colors for consistency across plots. To be used as a look-up table in converting between variable stock names and printed ones.

- 1) stock - ICES stock codes used in advice
- 2) order - stock order to be used in plots
- 3) col - stock colors for plots (e.g. pals::brewer.paired())
- 4) stock_short - short stock name used in mixed fishery model

Usage

```
data(refTable)
```

Format

```
bla bla
```

Source

WGMIXFISH-Advice 2021, North Sea case study. (https://github.com/ices-taf/2021_NrS_MixedFisheriesAdvice)

Examples

```
data(refTable)
refTable
```

stfFltStkSum	<i>Data.frame containing short-term forecast summary of catch-related variables per stock and fleet combination</i>
--------------	---

Description

The `stfFltStkSum` data.frame is an output of `'FLBEIA::fltStkSum()'`. Provides example data for use in `'plot_effortFltStk'`.

- scenario - advice scenario
- year - advice year
- fleet - fleet names
- stock - stock names used in mixed fishery model
- iter - iteration number
- catch -
- landings -
- discards -
- discRat -
- price -
- tacshare - fraction of the total stock quota for a given fleet
- quota - advised catch quota
- quotaUptake - effort required to take up quota
- choke - (logical) is stock the limiting one for the fleet

Usage

```
data(stfFltStkSum)
```

Format

```
data.frame
```

Source

WGMIXFISH-Advice 2021, North Sea case study (https://github.com/ices-taf/2021_NrS_MixedFisheriesAdvice)

Examples

```
data(stfFltStkSum)
head(stfFltStkSum)
```

`stfFltSum`*Data.frame containing short-term forecast summary of catch-related variables per fleet*

Description

The `stfFltSum` data.frame is an output of `'FLBEIA::fltSum()'`. Provides example data for use in `'plot_effortFltStk'`.

- scenario - scenario name
- year - year
- fleet - fleet names
- iter - iteration number
- catch -
- landings -
- discards -
- capacity -
- effort -
- fcosts -
- vcosts -
- costs -
- grossValue -
- nVessels -
- discRat -
- grossSurplus -
- price -
- salaries -
- gva -
- profitability -
- fep -
- netProfit -
- quotaUptake - effort required to take up quota

Usage

```
data(stfFltSum)
```

Format

```
data.frame
```

Source

WGMIXFISH-Advice 2021, North Sea case study. (https://github.com/ices-taf/2021_NrS_MixedFisheriesAdvice)

Examples

```
data(stfFItSum)
head(stfFItSum)
```

stfMtStkSum	<i>Data.frame containing short-term forecast summary of catch-related variables per stock, fleet, and metier combination</i>
-------------	--

Description

The stfMtStkSum data.frame is an output of 'FLBEIA::mtStkSum()'. Provides example data for use in 'plot_catchComp'.

- scenario - advice scenario
- year - advice year
- fleet - fleet names
- metier - metier names
- stock - stock names used in mixed fishery model
- iter - iteration number
- catch -
- landings -
- discards -
- discRat -
- price -

Usage

```
data(stfMtStkSum)
```

Format

```
data.frame
```

Source

WGMIXFISH-Advice 2021, North Sea case study (https://github.com/ices-taf/2021_NrS_MixedFisheriesAdvice)

Examples

```
data(stfMtStkSum)
head(stfMtStkSum)
```

stock_props	<i>Data.frame contains information on the stock, metiers and landings proportions</i>
-------------	---

Description

The stock_props data.frame provides example data for use in 'plot_stkMetProp'.

- Stock - scenario name
- metCat - year
- prop_stock - fleet names
- label

Usage

```
data(stock_props)
```

Format

```
data.frame
```

Source

Intercatch

Examples

```
data(stock_props)
head(stock_props)
```

wgnsskStocks	<i>FLStocks object with assessment estimated values</i>
--------------	---

Description

The wgnsskStocks data is an FLStocks object containing estimated values for 6 stocks assessed during WGNSSK 2024 ("HAD", "NEP6", "PLE-EC", "PLE-NS", "POK", "WIT").

Usage

data(wgnsskStocks)

Format

FLStocks

Source

WGNSSK 2024

Examples

```
library(FLCore)
data(wgnsskStocks)
wgnsskStocks[["HAD"]]
```

Index

* datasets

- haddockWts, [3](#)
- NSeffort, [4](#)
- NSlandings, [5](#)
- refTable, [32](#)
- stfFltStkSum, [33](#)
- stfFltSum, [34](#)
- stfMtStkSum, [35](#)
- stock_props, [36](#)
- wgnsskStocks, [37](#)

gm_mean, [2](#)

haddockWts, [3](#)

mean, [2](#)

NSeffort, [4](#)

NSlandings, [5](#)

plot_alluvial, [6](#)

plot_catch_change, [8](#)

plot_catchComp, [10](#)

plot_catchCompSpp, [12](#)

plot_catchCompStk, [15](#)

plot_catchScenStk, [18](#)

plot_effortFltStk, [20](#)

plot_F0gearType, [22](#)

plot_landByMetStock, [23](#)

plot_landByStock, [25](#)

plot_overUnderFltStk, [27](#)

plot_reIEffortFltStk, [29](#)

plot_stkMetProp, [31](#)

refTable, [20](#), [24](#), [26](#), [27](#), [32](#)

stfFltStkSum, [33](#)

stfFltSum, [34](#)

stfMtStkSum, [35](#)

stock_props, [36](#)

wgnsskStocks, [37](#)