Package: mixfishtools (via r-universe)

September 1, 2024

Type Package

Title WGMIXFISH tools for reproducibility

Version 0.4.0

Date 2024-08-23

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 (>= 4.0), ggplot2, htmlwidgets, networkD3, htmltools, dplyr

Suggests png, knitr, kableExtra, rmarkdown

License GPL-3 + file LICENSE

Encoding UTF-8

LazyData false

RoxygenNote 7.3.1

VignetteBuilder knitr

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

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

RemoteRef HEAD

RemoteSha ad759fdaa5715c06f81e936b0d63902cf98324e6

Contents

gm_mean *Geometric Mean*

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
```

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"
)
```

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` 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`.

`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")

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, 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, hjust=1)) +
  facet_wrap(divider,  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
selectors <- c(”country”, ”metier”)
divider <- c(”area”)
p <- plot_catchComp(data,refTable,filters=NULL,selectors, divider)
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)

# 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)
```

`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_catch_change` *Plot fleet landings taken up relative to recent landings / quota*

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_effortFltStk *Plot fleet effort to uptake stock quotas*

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 = "KW days ('000)",
  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 (‘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 = "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. 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$fleet)
df2 <- vector("list", length(fls))
names(df2) <- fls
for(i in seq(fls)){
 tmp <- subset(df, fleet == fls[i])
 tmp$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_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_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 [t]",
  fillLegendTitle = "Stock"
)
```

Arguments

`data` data.frame Contains information on the landings (or catch) by stock and metiers/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. pals::brewer.paired())
- 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 *Pie chart of landings by stock*

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. pals::brewer.paired())
- 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_MetMetFleet *Metier to Metier to Fleet Sankey plot*

Description

function to plot metier to mixedfish metier and fleet flow to provide a description and visualization of how metiers are constructed

Usage

```
plot_MetMetFleet(MetMetData, MetFleetData = NULL, Col_2_Link = NA)
```

Arguments

MetMetData	data.frame containing the original metier from the accession file and the output metier and a Link value (default assumption is Landings)

`MetFleetData` data.frame containing the the output metier and the fleet to be used in the model and a Link value (default assumption is Landings)

`Col_2_Link` column name (character) for the linking "value" variable. Default assumption is NA and the function defaults to Landings column

Details

Users will need to provide a data frame with three columns, two for metiers and one for the value used to link them.if a second dataframe is provided to link through to fleets you will need a metier column matching the output metier of the first, a fleet column and a value to link them. The data must be surmised to the metier columns using a group_by statement or similar. Where a metier goes to itself for example SDN_DEF to SDN_DEF you will experiences a doughnut

Value

a sanky plot, see the example for how to save a static sankey plot.

Examples

```
mtcars$Name <- rownames(mtcars)
dat <- mtcars %>% select(Name,gear,hp)
dat$gear <- as.character(dat$gear )
names(dat) <- c("Original_Metier","Metier","hp")

P <- plot_MetMetFleet(dat,Col_2_Link = "hp")

# Sankey plots are interactive by nature and are saved as an html, to get a static image they
# are captured using webshot from the htmlwidgets

P <- htmlwidgets::prependContent(P, htmltools::tags$h1("Title"))
P <- htmlwidgets::appendContent(P, htmltools::tags$p("Caption"))

# save plot
# saveNetwork(P, file =file.path("Plot_path" ,paste("A_Name","_sn.html",sep="")))
# save as png
# webshot::webshot(
# url = file.path("Plot_path",
#   paste("A_Name","_sn.html",sep="")),
# file.path("Metier_Sankey", paste(i,"_sn.png",sep="")),
# vwidth = 640,
# vheight=840)
```

`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

data	data.frame Contains information on relative effort (to status quo effort, ‘var‘) required to uptake quotas by fleet (‘fleet‘) and stock (‘scenario‘).
limits	vector Two value vector with lower and upper limits for fill colors (Default: ‘limits = c(-100,100)‘)
xlab	character X-axis label (Default: ‘xlab = "Stock"‘)

`ylab` character Y-axis label (Default: ‘ylab = "Fleet"‘)

`fillLegendTitle`
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()
```

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 *Data.frame containing short-term forecast summary of catch-related variables per stock and fleet combination*

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(stfFltSum)
head(stfFltSum)
```

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

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)
```

Index