Package 'StrathE2E2'

January 21, 2021

Version 3.3.0

Title End-to-End Marine Food Web Model

Author Michael Heath [aut],
Ian Thurlbeck [ctb]

Maintainer Michael Heath <m.heath@strath.ac.uk>

Depends R (>= 3.6.0),

Imports deSolve, methods, NetIndices

Description A dynamic model of
the big-picture, whole ecosystem effects of hydrodynamics,
temperature, nutrients, and fishing on continental shelf marine
food webs. The package is described in: Heath, M.R.,
Speirs, D.C., Thurlbeck, I. and Wilson, R.J. (2020)
<doi.org/10.1111/2041-210X.13510> StrathE2E2: An R package
for modelling the dynamics of marine food webs and
fisheries. 8pp.

License GPL (>= 2)

LazyData yes

NeedsCompilation yes

Date 2021-01-21

URL https://gitlab.com/MarineResourceModelling/StrathE2E/StrathE2E2,
https://marineresourcemodelling.gitlab.io/

Suggests knitr, StrathE2E2examples, testthat (>= 2.1.0)

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.1.1

Additional_repositories https://marineresourcemodelling.gitlab.io/sran

Archs i386, x64

R topics documented:

e2e_calculate_hrscale 2
e2e_compare_obs 3
e2e_compare_runs_bar 5

e2e_compare_runs_box . . . 8
e2e_copy . . . 10
e2e_extract_hr . . . 11
e2e_extract_start . . . 12
e2e_get_parmdoc . . . 13
e2e_get_senscrit . . . 14
e2e_ls . . . 15
e2e_merge_sens_mc . . . 16
e2e_optimize_act . . . 19
e2e_optimize_eco . . . 24
e2e_optimize_hr . . . 27
e2e_plot_biomass . . . 30
e2e_plot_catch . . . 32
e2e_plot_eco . . . 33
e2e_plot_edrivers . . . 35
e2e_plot_fdrivers . . . 37
e2e_plot_migration . . . 39
e2e_plot_opt_diagnostics . . . 41
e2e_plot_sens_mc . . . 45
e2e_plot_trophic . . . 48
e2e_plot_ts . . . 50
e2e_plot_ycurve . . . 51
e2e_process_sens_mc . . . 53
e2e_read . . . 55
e2e_run . . . 58
e2e_run_mc . . . 59
e2e_run_sens . . . 64
e2e_run_ycurve . . . 69
StrathE2E2 . . . 71

Index **74**

`e2e_calculate_hrscale` *Calculate initial values of the scaling parameters for the fishing fleet model which link effort to harvest ratios.*

Description

The scaling parameters convert the effort applied by each gear to each living resource guild in the model, into a harvest ratio (proportion of mass captured per day). Effort is the product of activity and the relative power of each gear with respect to each guild.

Usage

```
e2e_calculate_hrscale(model)
```

Arguments

`model`	R-list object generated by the e2e_read() function which defined the model configuration

Details

In order to estimate these scaling parameters, data are needed for a 'calibration' time period when activity, catch and harvest ratio are all known.

The function assumes that the relevant model configuration has already been loaded using the e2e_read() function. It is expected that this has loaded the array of power parameters (catch per unit activity by gear and guild during the calibration period) given in the file /Param/fishing_power_*.csv, the corresponding activity in the file /Param/fishing_activity_*.csv, and the known harvest ratios of each guild in the file /Target/region_harvest_r_*.csv

The function returns rough estimates of the scaling parameters. These are only rough because they assume that the effort and resource biomass are distributed uniformly over the model domain. Hence they should be used as initial estimates to be refined by optimization.

The scaling parameters are dispayed on the screen and saved in a dataframe. The values then need to be manually edited into rows 12-21 (excluding the header row) of the file /Param/fishing_fleet_*.csv

Value

Dataframe of the effort-harvest ratio scaling parameter values for each guild

See Also

e2e_ls , e2e_read

Examples

```
# Load the 2003-2013 version of the North Sea model supplied with the package and calculate
# scaling parameter values:
    model <- e2e_read("North_Sea", "2003-2013", quiet=FALSE)
    scale_values <- e2e_calculate_hrscale(model)
    scale_values
```

---

e2e_compare_obs *Box and whisker plots comparing annual or monthly observational data with corresponding model outputs.*

Description

Generate a multi-panel set of box and whisker diagrams comparing annual or monthly averaged or integrated observational data on the state of the ecosystem with corresponding model outputs. Each panel displays a different category of observational and model data.

Usage

```
e2e_compare_obs(
  selection = "ANNUAL",
  model,
  ci.data = FALSE,
  use.saved = FALSE,
  use.example = FALSE,
  results = NULL
)
```

Arguments

selection	Text string from a list to select comparison with annual or monthly observations. Select from: "ANNUAL", "MONTHLY". Remember to include the phrase within "" quotes.
model	R-list object defining the baseline model configuration used to generate the data and compiled by the e2e_read() function.
ci.data	Logical. If TRUE plot credible intervals around model results based on Monte Carlo simulation with the e2e_run_mc() function (default=FALSE).
use.saved	Logical. If TRUE use data from a prior user-defined run held as csv files data in the current results folder (default=FALSE).
use.example	Logical. If TRUE use pre-computed example data from the internal North Sea model rather than user-generated data (default=FALSE).
results	R-list object of model output generated by the e2e_run() function (default=NULL).

Details

The observational data are read from the file 'annual_observed_*.csv' or 'monthly_observed_*.csv'in the /Target folder of the model setup defined by a e2e_read() function call. Column 3 of annual_observed_* (header "Use_1.yes_0.no") is a flag to set whether any given row is used in calculating the likelihood of the observed data given the model setup by functions such as e2e_optimize_eco(). Un-used rows of data are omitted from the box and whisker plotting panels.

Arguments determine the source of model data for comparison with the observations. These can be outputs from a single model run with data held in memory as a list object or in a saved csv file, or from a Monte Carlo simulation (using the function e2e_run_mc()) to estimate credible intervals of model outputs. Generation of credible interval data is a long computing task, so example data for the North Sea model provided with the package for illustration.

In each plot panel, the distribution of observation data is shown by a black box plot (box spans 50 derived from model outputs are always shown in red - either a red tick mark for data from a single model run, or a red boxplot for data from a Monte Carlo analysis.

Value

graphical display in a new graphics window

See Also

e2e_read, e2e_run, e2e_run_mc

Examples

```
# Load the 1970-1999 version of the North Sea model supplied with the package, run,
# and generate a plot:
    model <- e2e_read("North_Sea", "1970-1999")
    results <- e2e_run(model, nyears=2,csv.output=FALSE)
# Plot data annual data
    e2e_compare_obs(selection="ANNUAL", model, results=results)
# Note that these are the observational data that were used as the target for optimizing
# the model parameters

# Direct the graphics output to a file ... in this example the graphics file
# is sent to a temporary folder rather than the current working directory:
# or jpeg("plot.jpg"), png("plot.png")
```

```
    pdf(file.path(tempdir(), "plot.pdf"),width=8,height=6)
    e2e_compare_obs(selection="ANNUAL", model, results=results)
    dev.off()

# Plot monthly data
    dev.new()
    e2e_compare_obs(model, selection="MONTHLY",results=results)
# Note that these observational data were NOT used for optimizing the
# model parameters


# To create the same plots from the csv files saved by the e2e_run() function, use:
    model <- e2e_read("North_Sea", "1970-1999")
    results <- e2e_run(model, nyears=2,csv.output=TRUE)
# Here the csv outputs are saved to a temporary folder since results.path is not
# set in e2e_read()
    e2e_compare_obs(selection="ANNUAL", model, use.saved=TRUE)
    dev.new()
    e2e_compare_obs(selection="MONTHLY", model, use.saved=TRUE)


# Load the 1970-1999 version of the North Sea model supplied with the package and plot
# example data with credible intervals generated by a baseline mode Monte Carlo analysis.
# This example requires the Strathe2E2examples supplementary data package.
if(require(StrathE2E2examples)){
    model <- e2e_read("North_Sea", "1970-1999")
    e2e_compare_obs(model, selection="ANNUAL",ci.data=TRUE,use.example=TRUE)
    dev.new()
    e2e_compare_obs(selection="MONTHLY", model, ci.data=TRUE,use.example=TRUE)
}
```

`e2e_compare_runs_bar` *Create tornado bar-plots of the differences between two model runs.*

Description

Create a tornado bar-plot diagram of the differences in either annual average masses of ecology model varibles, or annual fishery catches, between two different runs of the StrathE2E model, referred to as baseline and sceanrio runs.

Usage

```
e2e_compare_runs_bar(
  selection = "AAM",
  model1 = NA,
  use.saved1 = FALSE,
  results1,
  model2 = NA,
  use.saved2 = FALSE,
  results2,
  log.pc = "PC",
  zone = "W",
```

```
  bpmin = (-50),
  bpmax = (+50),
  maintitle = ""
)
```

Arguments

selection	Text string from a list identifying which type of data are to be plotted. Select from: "AAM", "CATCH", corresponding to annual average mass data, and catch data respectively (default = "AAM"). Remember to include the phrase within "" quotes.
model1	R-list object defining the baseline model configuration compiled by the e2e_read() function.
use.saved1	Logical. If TRUE then use baseline data from a prior user-defined run held as csv files data in the current results folder (default=FALSE).
results1	R-list object of baseline model output generated by the e2e_run() (default=NULL).
model2	R-list object defining the scenario model configuration compiled by the e2e_read() function.
use.saved2	Logical. If TRUE then use scenario data from a prior user-defined run held as csv files data in the current results folder (default=FALSE).
results2	R-list object of scenario model output generated by the e2e_run() (default=NULL).
log.pc	Value="LG" for data to be plotted on a log10 scale, value = "PC" for data to be plotted on a percentage difference scale (default = "PC").
zone	Value = "O" for offshore, "I" for inshore, or "W" for whole model domain (all upper case) (default = "W").
bpmin	Axis minimum for plot - i.e. the maximum NEGATIVE value of (scenario-baseline). Default = -50, given log.pc="PC" (percentage differences). Needs to be reset to e.g. -0.3 if log.pc="LG" (log scale).
bpmax	Axis maximum for plot - i.e. the maximum POSITIVE value of (scenario-baseline). Default = +50, given log.pc="PC" (percentage differences). Needs to be reset to e.g. +0.3 if log.pc="LG" (log scale).
maintitle	A optional descriptive text field (in quotes) to be added above the plot. Keep to 45 characters including spaces (default="").

Details

A tornado plot is a horizontal barplot which shows the difference between baseline and sceanrio runs. Bars to the right indicate scenario values greater than the baseline, bars to the left indicate scenario values less than baseline. In this function the difference between scenario and baseline data is calculated as either ((scenario - baseline)/baseline) and plotted on a linear (percentage) scale, or (scenario/baseline) and plotted on a log10 scale. In both cases, zero represents scenario = baseline.

The choice of whether to plot data on the masses of ecosystem components, or fishery catches, is determined by an argument setting in the function call. In either case two panels of tornado diagrams are plotted. In the case of mass data, the panels show data on water column components of the ecoststem (upper panel) and seabed components of the system (lower panel). In the case of catch data, the upper panel shows landings, the lower shows discards. For mass data and landings, positive values (scenario > baseline) are indicated by green bars to the right, negative values (scenario < baseline) by red bars to the left. For discards, positive values are in black, negative in grey. For any bars extending outsided the selected plotting range, the numeric value is displayed inside the relevant bar.

The function accepts data from either saved csv files created by the e2e_run() function, or directly from e2e_run() results objects. The relevant csv files are: /results/Modelname/Variantname/ZONE_model_anav_biomass *.csv where ZONE is INSHORE, OFFSHORE or WHOLEDOMAIN and * represents the model run identifier (model.ident) embedded in the R-list object created by the e2e_read() function.

In addition to generating graphics output the function returns a list object of the data presented in the plot. The object comprises two dataframes (changewater and changeseabed where annual average mass data are selected; changeland and changedisc where catch data are selected). The first column in each dataframe is the proportional difference expressed on a log10 scale, the second column as a percentage.

Value

List object comprising two dataframes of the displayed data, graphical display in a new graphics window.

See Also

e2e_read, e2e_run

Examples

```
# Load the 1970-1999 version of the internal North Sea model and run for 1 year
    m1 <- e2e_read("North_Sea", "1970-1999",model.ident="70_99")
    r1 <-e2e_run(m1,nyears=1)

# Load the 2003-2013 version of the internal North Sea model and run for 1 year
    m2 <- e2e_read("North_Sea", "2003-2013",model.ident="03_13")
    r2 <-e2e_run(m2,nyears=1)

# Compare the annual average mass in 1970-1999 (as the baseline case) with 2002-2013
# (as the scenario case):
    mdiff_results1 <- e2e_compare_runs_bar(selection="AAM",
                                      model1=NA, use.saved1=FALSE, results1=r1,
         model2=NA, use.saved2=FALSE, results2=r2,
                         log.pc="PC", zone="W",
                           bpmin=(-50),bpmax=(+50),
                        maintitle="2003-2013 compared with 1970-1999")
    mdiff_results1

# Compare the annual catch in 1970-1999 (as the baseline case) with 2002-2013
# (as the scenario case):
    mdiff_results2 <- e2e_compare_runs_bar(selection="CATCH",
                                      model1=NA, use.saved1=FALSE, results1=r1,
         model2=NA, use.saved2=FALSE, results2=r2,
                          log.pc="LG", zone="W",
                                bpmin=(-0.9),bpmax=(+0.4),
                        maintitle="2003-2013 compared with 1970-1999")
    mdiff_results2


# Create a new scenario run from 2003-2013 case.
# Copy the 2003-2013 configuration into a new model object:
    scen1_model   <- m2
    scen1_model$setup$model.ident <- "scenario1"
# Gear 4 (Beam_Trawl_BT1+BT2) activity rate rescaled to 0.5*baseline:
    scen1_model$data$fleet.model$gear_mult[4] <- 0.5
```

```
    scen1_results <- e2e_run(scen1_model,nyears=20)

# Compare the annual average mass from the the 2003-2013 baseline with scenario1 data
    mdiff_results3 <- e2e_compare_runs_bar(selection="AAM",
                                model1=NA, use.saved1=FALSE, results1=r2,
                                model2=NA, use.saved2=FALSE, results2=scen1_results,
                                log.pc="PC", zone="W",
                                bpmin=(-30),bpmax=(+30),
                                maintitle="Beam Trawl activity reduced by half")
    mdiff_results3



# Create a second sceanario from the 1970-1999 case, this time saving the results to a file;
# csv output to temporary folder since results.path was not set in e2e_read() when creating m1.
# Copy the baseline configuration into a new model object:
    scen2_model   <- m1
    scen2_model$setup$model.ident <- "scenario2"
# Gear 1 (Pelagic_Trawl+Seine) activity rate rescaled to 0.5*baseline:
    scen2_model$data$fleet.model$gear_mult[1] <- 0.5
    scen2_results <- e2e_run(scen2_model,nyears=20, csv.output=TRUE)

# Compare the annual catches in the 1970-1999 base line with the Pelagic Trawl/seine scenario
    mdiff_results4 <- e2e_compare_runs_bar(selection="CATCH",
                                model1=NA, use.saved1=FALSE, results1=r1,
                           model2=scen2_model,use.saved2=TRUE, results2=NA,
                           log.pc="LG", zone="W",
                           bpmin=(-0.4),bpmax=(+0.6),
                           maintitle="Pelagic Trawl/Seine activity reduced by half")
    mdiff_results4
```

`e2e_compare_runs_box` *Box and whisker plots comparing annual or monthly outputs from single or Monte Carlo baseline and scenario model runs.*

Description

Generate a multi-panel set of box and whisker diagrams comparing annual or monthly averaged or integrated model outputs from single or Monte Carlo baseline (black) and scenario (red) model runs. Each panel of annual data displays a different category of model data; each panel of monthly data shows a different nutrient or plankton guild.

Usage

```
e2e_compare_runs_box(
  selection = "ANNUAL",
  model1,
  ci.data1 = FALSE,
  use.saved1 = FALSE,
  use.example1 = FALSE,
  results1 = NULL,
  model2,
```

```
  ci.data2 = FALSE,
  use.saved2 = FALSE,
  use.example2 = FALSE,
  results2 = NULL
)
```

Arguments

selection	Text string from a list to select comparison with annual or monthly observations. Select from: "ANNUAL", "MONTHLY". Remember to include the phrase within "" quotes.
model1	R-list object defining the baseline model configuration compiled by the e2e_read() function.
ci.data1	Logical. If TRUE plot credible intervals around baseline model results based on Monte Carlo simulation with the e2e_run_mc() function (default=FALSE).
use.saved1	Logical. If TRUE use baseline data from a prior user-defined run held as csv files data in the current results folder (default=FALSE).
use.example1	Logical. If TRUE use pre-computed example data from the internal North Sea model as the baseline rather than user-generated data (default=FALSE).
results1	R-list object of baseline model output generated by the e2e_run() function. Only needed if ci.data1=FALSE, use.saved1=FALSE and use.example1=FALSE. (Default=NULL).
model2	R-list object defining the scenario model configuration compiled by the e2e_read() function.
ci.data2	Logical. If TRUE plot credible intervals around scenario model results based on Monte Carlo simulation with the e2e_run_mc() function (default=FALSE).
use.saved2	Logical. If TRUE use scenario data from a prior user-defined run held as csv files data in the current results folder (default=FALSE).
use.example2	Logical. If TRUE use pre-computed example data from the internal North Sea model as the scenario rather than user-generated data (default=FALSE).
results2	R-list object of baseline model output generated by the e2e_run() function. Only needed if ci.data2=FALSE, use.saved2=FALSE and use.example2=FALSE. (Default=NULL).

Details

Arguments determine the source of model data for comparison. These can be outputs from a Monte Carlo simulations (using the function e2e_run_mc()) to estimate credible intervals of model outputs, or from single model runs using e2e_run(). Generation of credible interval data is a long computing task, so example data for the North Sea model provided with the package are available as an illustration.

In each plot panel, where credible intervals of model outputs (from Monte Carlo analysis) have been selected the box spans 50 always shown in black, scenario results in red.

Value

Graphical display in a new graphics window.

See Also

e2e_read, e2e_run, e2e_run_mc, e2e_compare_runs_bar, e2e_compare_obs

Examples

```
# Load the 1970-1999 version of the internal North Sea model and run for 1 year
    m1 <- e2e_read("North_Sea", "1970-1999")
    r1 <-e2e_run(m1,nyears=1)

# Load the 2003-2013 version of the internal North Sea model and run for 1 year
    m2 <- e2e_read("North_Sea", "2003-2013")
    r2 <-e2e_run(m2,nyears=1)

# Compare annual results from 1970-1999 as baseline with 2003-2013 as scenario:
    e2e_compare_runs_box(selection="ANNUAL", model1=m1, ci.data1=FALSE, results1=r1,
                                              model2=m2, ci.data2=FALSE, results2=r2)
# Compare monthly results from 1970-1999 as baseline with 2003-2013 as scenario:
    dev.new()
    e2e_compare_runs_box(selection="MONTHLY", model1=m1, ci.data1=FALSE, results1=r1,
                                           model2=m2, ci.data2=FALSE, results2=r2)

# This example requires the StrathE2E2examples supplementrary data package.
# Compare 1970-1999 as baseline (from single model run), with 2003-2013
# as scenario (from example data with credible interval results):
if(require(StrathE2E2examples)){
    e2e_compare_runs_box(selection="ANNUAL", model1=m1, ci.data1=FALSE, results1=r1,
                                            model2=m2, ci.data2=TRUE, use.example2=TRUE)
    dev.new()
    e2e_compare_runs_box(selection="MONTHLY", model1=m1, ci.data1=FALSE, results1=r1,
                                            model2=m2, ci.data2=TRUE, use.example2=TRUE)
}

# This example requires the StrathE2E2examples supplementrary data package.
# Compare 1970-1999 as baseline (from example data with cred.int.), with 2003-2013
# as scenario (from example data with credible interval results):
if(require(StrathE2E2examples)){
   e2e_compare_runs_box(selection="ANNUAL", model1=m1, ci.data1=TRUE, use.example1=TRUE,
                                            model2=m2, ci.data2=TRUE, use.example2=TRUE)
    dev.new()
   e2e_compare_runs_box(selection="MONTHLY", model1=m1, ci.data1=TRUE, use.example1=TRUE,
                                            model2=m2, ci.data2=TRUE, use.example2=TRUE)
}
```

`e2e_copy` *Make a copy of a named model/variant and documentation in a user defined workspace*

Description

Make a copy of a named model/variant and documentation in a user defined workspace

Usage

```
e2e_copy(model.name, model.variant, source.path = NULL, dest.path = NULL)
```

Arguments

model.name	Name of model to copy.
model.variant	Name of model variant to copy.
source.path	Relative path from the current working directory to the source model to be copied. Setting source.path="" is valid. Default source.path=NULL, meaning read a North Sea model setup from the package folder extdata/Models.
dest.path	Relative path from the current working directory to a destination address in which to create a 'Models' folder if necessary, and then copy the model files. Setting dest.path="" is valid. Default dest.path=NULL in which case the Models folder will be created in a temporary directory.

Value

A copy of an entire model/variant and the associated documentation folder in the designated workspace.

See Also

e2e_ls, e2e_read

Examples

```
# Copy the 2003-2013 version of the North Sea model supplied with the package into a
# temporary folder:
    e2e_copy("North_Sea", "2003-2013")

# Dummy example illustrating copy the 2003-2013 version of the North Sea model
# supplied with the package into a user-defined folder (edit "Folder/Models to
# your own relative path):
#   e2e_copy("North_Sea", "2003-2013",dest.path="Folder/Models")

# Dummy example illustrating copying a user model into a user workspace:
# Replace "Folder1/Models" and "Folder2/Models" with your own source.path and dest.path
# remembering that these are relative to the current working directory.
# e.g.... e2e_copy("Modelname", "Modelvariant",
#         source.path="Folder1/Models",
#         dest.path="Folder2/Models")
```

---

e2e_extract_hr *Extract the values of harvest ratios generated by the fleet model.*

Description

The function extracts the inshore, offshore and whole-domain values of harvest ratio (proportion of biomass captured per day) for each guild, from the results object generated by the e2e_run() function.

Usage

```
e2e_extract_hr(model, results, csv.output = FALSE)
```

Arguments

model	R-list object defining the model configuration compiled by the e2e_read() function.
results	R-list object generated by the e2e_run() function.
csv.output	Logical. If TRUE then enable writing of CSV output files (default=FALSE).

Value

Dataframe of harvest ratio values, rows = guilds, columns = zone

See Also

e2e_read, e2e_run

Examples

```
model <- e2e_read("North_Sea", "1970-1999")
results<-e2e_run(model, nyears=2,csv.output=FALSE)
harvest_ratio_data <- e2e_extract_hr(model,results,csv.output=FALSE)
```

---

e2e_extract_start — *Extract the values of all the state variables at the end of a model run and format for use as new initial conditions.*

Description

The function saves the state of the model at the end of a run (using the e2e_run() function) for use as initial conditions in future runs. This enables, for example, the model to be run for a long time to attain a stationary state, and then restarted in that state.

Usage

```
e2e_extract_start(model, results, csv.output = TRUE)
```

Arguments

model	R-list object defining the model configuration compiled by the e2e_read() function.
results	R-list object generated by the e2e_run() function.
csv.output	Logical. If TRUE then enable writing of csv output files (default=FALSE).

Details

Initial conditions for a model run are held in the /Param folder of the Model/Variant path specified in the e2e_read() function call used to define a run. By default, the function attampts to write the model end-state file back to this /Param folder. However, the package folders are read-only so if e2e_read() has been specified to load an internally provided Model/Variant then the output will revert to the currently specified results folder instead. To fix this, copy the required package model to a user workspace using the e2e_copy() function and re-run.

The new initial conditions file will have a name initial_values-*.csv, where * is the model.ident text identifier specified in e2e_read() To source the new initial conditions in a subsequent model run, edit the MODEL_SETUP.csv file in the required /Models/Variant folder

Value

Table of state variable values from the end of a run formatted as required for input to a new model run as initial conditions. By selecting csv.output=TRUE these data are also returned as a csv file to the Param folder of the current model, unless this is one of the North Sea model versions within the package.

See Also

e2e_read, e2e_run

Examples

```
# Example which generates an initial values object from an internal North Sea model but
# does not attempt to save it back to the Parameters folder of the model setup. Just
# run for 1 year in this example:
    model <- e2e_read("North_Sea", "2003-2013")
    results <- e2e_run(model, nyears=1)
    new_initial <- e2e_extract_start(model,results,csv.output=FALSE)
    new_initial

# Dummy example to illustrate the process of saving initial values
# data back into the model Param folder:
# Assumes that the model setup is held in models.path="Folder/Models". Replace this
# with your own relative path to your user library of models:
#   model <- e2e_read("Modelname","Variantname",models.path="Folder/Models",model.ident="new")
#    results <- e2e_run(model, nyears=30)
#    new_initial <- e2e_extract_start(model,results,csv.output=TRUE)
# The new initial values file will be written back into Folder/Models/Param with
# the identitifer "new"
```

---

e2e_get_parmdoc *Get documentation on the parameters in the model.*

Description

Provides access to documentation on the parameters in the model in the form of a dataframe.

Usage

```
e2e_get_parmdoc(id = 9)
```

Arguments

id	Integer value denoting the class of parameters details to be downloaded. Choose from: 0 = fitted ecology, 1 = fixed ecology, 2 = fishing fleet, 3 = harvest ratios, 4 = environmental drivers, 5 = physical configuration, 9 = all (default = 9).

Details

The full range of inputs to the model includes both numeric values which are constant over the during a run, and values which vary over time during a run according to an externally provided schedule. We refer to the latter as 'drivers', but they are all parameters nontheless. The constant parameters are divided into categories according to whether they are associated with the ecology model, the fishing fleet model, and whether they are available for optimization or not.

This function downloads documentation on the full range of parameters in the form of a dataframe which can be used to form labels in user-generated code. The parameters are identified by a numeric value linked to 6 different classes of parameter.

Note that the function does not supply the numeric values of parameters for any given model setup. These are available in the model definition object generated by the function e2e_read().

Value

dataframe of parameter documentation.

See Also

e2e_read

Examples

```
parm_list <- e2e_get_parmdoc(0)   # Get documentation on the fitted ecology parameters
 parm_list <- e2e_get_parmdoc(9)    # Get documentation all parameters
 parm_list <- e2e_get_parmdoc()     # Get documentation all parameters
```

---

e2e_get_senscrit *Get documentation on the model outputs available as the basis for sensitivity analysis.*

Description

Provides access to documentation on outputs from the model which may be deployed as the basis for a sensitivity analysis using e2e_run_sens().

Usage

```
e2e_get_senscrit(id = "A")
```

Arguments

id	Single character (in "") or integer value denoting the class or id of output criterion to be downloaded. Choose single value from 0:247 = unique criterion, A = All (default), L = Likelihood, M = Annual average mass, F = Annual integrated fluxes.

Details

The function e2e_run_sens() performs a Morris Method sensitivity analysis on the model. The default criterion for assessing the model sensitivity is the likelihood of the observed target data set on the state of the ecosystem given each set of model drivers and parameters. However, a function argument allows other criteria to be chosen as the basis for the analysis from the list of annually averaged or integrated variables saved in the output objects:

- results$final.year.output$mass_results_wholedomain (whole-domain annual averages of stage variables over the final year of a model run), and
- results$final.year.output$annual_flux_results_wholedomain (whole-domian annual integrals of fluxes between state variables over the final year of a model run).

The criterion is chosen by setting a value for the argument outID. The default outID=0 selects the likelihood of the observed target data. Other values in the range 1 to 247 select annualy averaged mass or annually integrated flux outputs. The function presented here provides a list of all the available outputs that may be used and their outID values.

Value

screen display and dataframe of parameter documentation.

See Also

e2e_run_sens

Examples

```
crit_list <- e2e_get_senscrit("M")   # Get a list of annual average mass criteria
crit_list <- e2e_get_senscrit("F")   # Get a list of annual integrated flux criteria
crit_list <- e2e_get_senscrit(24)    # Get details of criteria matching id=24
crit_list <- e2e_get_senscrit()      # Get a list of all available criteria
```

---

e2e_ls *List the available models in a designated workspace.*

Description

List the available models in a designated workspace.

Usage

```
e2e_ls(models.path = NULL)
```

Arguments

models.path	Relative path from the current working directory to a folder containing a library of model configurations (typically "Folder/Models"). Setting models.path="" is valid. Default models.path=NULL, meaning read a North Sea model setup from the package folder extdata/Models.

Value

An on-screen list of available models in the designated folder.

See Also

e2e_copy, e2e_read

Examples

```
# List the models/variants supplied with the package:
    e2e_ls()

# Dummy example to illustrate listing the models/variants in a user defined
# workspace. REPLACE "Folder1/Models" with your own models.path
# remembering that this are relative to the current working directory.
#    e2e_ls("Folder/Models")
```

---

e2e_merge_sens_mc *Combine two or more sets of raw output data from parallel sensitivity or Monte Carlo analysis runs performed on separate machines/processors.*

Description

The functions e2e_run_sens() and e2e_run_mc() are extremely time consuming so it makes sense to share the load across multiple processors in parallel and combine the results afterwards. This function merges the raw outputs from multiple separate runs of either function into a single file. This is not as simple as merely concatenating the files as it is necessary to keep track of the unique identities of the sets of trajectories.

Usage

```
e2e_merge_sens_mc(
  model,
  selection = "",
  ident.list,
  postprocess = TRUE,
  csv.output = FALSE
)
```

Arguments

model	Model object for the raw data to be combined generated by the e2e_read() function.
selection	Text string from a list identifying source of data to be merged. Select from: "SENS", "MC", referring to sensitivity analysis or Monte Carlo analysis. Remember to include the phrase within "" quotes.
ident.list	A vector of text variables corresponding to the "model.ident" identifiers for each of the files to merged (list must be length 2 or greater).

postprocess	Logical. if TRUE then process the results through to final data; if FALSE just produce the combined raw results (default=TRUE). The reason for NOT processing would be if there are further run results stil to be combined with the set produced by this function.
csv.output	Logical. If TRUE then enable writing of CSV output files (default=FALSE).

Details

The files to be combined must be transferred into the same folder, and this is where the new combined files will be placed. The path to locate the files is set in a e2e_read() function call. If not specified it is assumed that the files are located in the current temporary folder.

An identifying text string for the new combined files is set by the 'model.ident' argument in a e2e_read() function call.

The list of files to be combined (any number > 1) is defined by a vector of their individual "model.ident" identifiers ("ident.list" argument).

When combining the files, the function creates a seamless sequence of trajectory identifiers through the combined data, beginning from 1 for the first baseline (maximum likelihood) trajectory of the first set.

If for any reason there is a need to combine separate batches of multiple run results, then post-processing can be delayed with the 'postprocess' argument until the last merge when all the data have been gathered together. Stand-alone postprocess can be performed using the function e2e_process_sens_mc().

Details relating to merging sensitivity analysis files:

e2e_run_sens() generates two output files per run - OAT_results-*.csv, and OAT_parameter_values-*.csv, where * is a model.ident text string set as an argument of the e2e_read() function. Thus function merges both of these types of files.

When combining sensitivity analysis data the first-named model.ident in the ident.list vector MUST correspond to a run of the e2e_run_sens() function with the argument coldstart=TRUE, and all others with coldstart=FALSE. This forces the first trajectory of sensitivity test to be performed on the baseline (e.g. maximum-likelihood) parameter set loaded with the initial e2e_read() function call. Thus is important for the post-processing stage of the analysis which needs to be performed on the combined results.

Details relating to merging Monte Carlo analysis files:

e2e_run_mc() generates 11 different output files of accumulated outputs from the iterations (total 3.2 Mb per iteration) covering the range of model outputs. This merging function combines batches of each of these types of files. Check the memory capacity of you machine before starting a long run or before merging runs. The processed output consists of 13 files total ~4 Mb regardless of the number of iterations.

Value

csv files of merged data if csv.output=TRUE. If the argument postprocess=TRUE then also a dataframe of processed output (for sensitivity analysis) or a list object of dataframes (for Monte Carlo analysis).

See Also

e2e_read , e2e_run_sens , e2e_run_mc , e2e_process_sens_mc , e2e_plot_sens_mc

Examples

```
## Not run:
# The examples provided here are illustration of how to merge results from parallel
# sets of sensitivity or Monte Carlo analysis runs. Even though they are stripped-down
# minimalist examples they each still take a very long time to run.

# ------------------------------------------------------------------------

# Example of parallelizing the sensitivity analysis process.
# Here the model is only run for a 1 year each time and for 3 trajectories
# on each processor in order to illustrate the approach. A meaningful simulation
# would require many more years per run and more trajectories. Even so this
# example will be very time consuming.
# In the illustration here csv output is directed to a temporary folder since
# results.path is unspecified. To explore further, set up your own results folder and
# define results.path as relative to the current working directory.
# Launch two (or more) runs separately on different processors...
# Launch batch 1 (on processor 1):
    model1 <- e2e_read("North_Sea", "1970-1999", model.ident="BATCH1")
    sens_results <- e2e_run_sens(model1, nyears=1, n_traj=3, coldstart=TRUE,
                                 postprocess=FALSE, csv.output=TRUE)
# Note that coldstart=TRUE for the first batch only.
# Launch batch 2 (on processor 2):
    model2 <- e2e_read("North_Sea", "1970-1999", model.ident="BATCH2")
    sens_results <- e2e_run_sens(model2, nyears=1, n_traj=3, coldstart=FALSE,
                                 postprocess=FALSE, csv.output=TRUE)
# Note that these two runs return only raw data since postprocess=FALSE
#
# Then, afterwards, merge the two raw results files with text-tags BATCH1 and BATCH2,
# and post process the combined file:
    model3 <- e2e_read("North_Sea", "1970-1999", model.ident="COMBINED")
    processed_data <- e2e_merge_sens_mc(model3, selection="SENS",
          ident.list<-c("BATCH1","BATCH2"), postprocess=TRUE, csv.output=TRUE)
# or...
    combined_data <- e2e_merge_sens_mc(model3, selection="SENS",
          ident.list<-c("BATCH1","BATCH2"), postprocess=FALSE, csv.output=TRUE)
    processed_data <- e2e_process_sens_mc(model3, selection="SENS",
                                          use.example=FALSE,csv.output=TRUE)

## End(Not run)

# ------------------------------------------------------------------------


# Example of parallelizing the Monte Carlo process:
# Here the model is only run for a 2 year each time and for a 5 (or 6) trajectories
# on each processor in order to illustrate the approach. A meaningful simulation
# would require many more years per run and more trajectories. Even so this
# example will be time consuming.
# In the illustration here csv output is directed to a temporary folder since
# results.path is unspecified. To explore further, set up your own results folder and
# define results.path as relative to the current working directory.
# Launch two (or more) runs separately on different processors...
# Launch batch 1 (on processor 1)
    model1 <- e2e_read("North_Sea", "1970-1999", model.ident="BATCH1")
    results1 <- e2e_run_mc(model1,nyears=2,baseline.mode=TRUE,
```

```
                                n_iter=5,csv.output=TRUE,postprocess=FALSE)
# Launch batch 2 (on processor 2):
    model2 <- e2e_read("North_Sea", "1970-1999", model.ident="BATCH2")
    results2 <- e2e_run_mc(model2,nyears=2,baseline.mode=TRUE,
                           n_iter=6,csv.output=TRUE,postprocess=FALSE)
# Note that these two runs return only raw data since postprocess=FALSE
# Note 6 iterations in batch 2 - the first iteration wil be stripped off at merging so the
# combined data should include only 10 iterations.

# Then, afterwards, merge the two raw results files with text-tags BATCH1 and BATCH2,
# and post process the combined file:
    model3 <- e2e_read("North_Sea", "1970-1999", model.ident="COMBINED")
    processed_data <- e2e_merge_sens_mc(model3, selection="MC",
                ident.list<-c("BATCH1","BATCH2"), postprocess=TRUE, csv.output=TRUE)
# or...
    combined_data <- e2e_merge_sens_mc(model3, selection="MC",
           ident.list<-c("BATCH1","BATCH2"), postprocess=FALSE, csv.output=TRUE)
   processed_data<-e2e_process_sens_mc(model3, selection="MC",use.example=FALSE,csv.output=TRUE)

# ------------------------------------------------------------------------
```

`e2e_optimize_act` *Optimize StrathE2E fishing gear activity multipliers to maximumize the likelihood of either observed ecosystem target data or know harvest ratios.*

Description

Launches a simulated annealing process to find the set of fishing fleet model gear activity multipliers producing the maximum likelihood of either a) observed target data on the state of the ecosystem given specified environmental driving data and ecology model parameters, and effort-harvest ratio scaling parameters, or b) known harvest ratios for each guild in each spatial zone. The former version (optimizing to the state of the ecosystem uses the full StrathE2E system, ie the combined fishing fleet and ecology models. However, the latter version, i.e. optimizing to known harvest ratios, uses only the fishing fleet model. The version of optimization procedure is selected by an argument in the function call.

Usage

```
e2e_optimize_act(
  model,
  selection = "HR",
  n_iter = 3000,
  start_temperature = 0.5,
  cooling = 0.985,
  csv.output = FALSE,
  runtime.plot = TRUE,
  n_traj = 100,
  deltaHi = 0.2,
  attenuationstep = 500,
  deltaG = 0.25,
```

```
  nyears = 40,
  quiet = TRUE
)
```

Arguments

model	R-list object generated by the e2e_read() function which defined the model configuration.
selection	Text string from a list to select the method for optimization. Select from: "HR", "ECO", Remember to include the phrase within "" quotes (default = "HR"). "HR" selects a scheme for optimizing the activity multipler to known harvest ratios and uses just the fishing fleet model. "ECO" selects a scheme for optimizing to the observational data in the state of the ecosystem and so uses both the fishing fleet and ecology models.
n_iter	Number of iterations of the model (per trajectory in the case of selection = "HR") (default=3000). Suggest a value of 500 if selection = "ECO".
start_temperature	Initial value of the simulated annealing temperature parameter (default=1). Suggested a value of 0.5 if selection = "ECO". Values in the range 0.0005 - 5 are reasonable. Higher values increase the probability of rejecting parameter combinations producing an improvement in likelihood.
cooling	Rate at which the simulated annealing temperature declines with iterations (default=0.985). Suggested a value of 0.975 if selection = "ECO".
csv.output	Logical. If TRUE then enable writing of csv output files (default=FALSE).
runtime.plot	Logical. If FALSE then disable runtime plotting of the progress of the run - useful for testing (default=TRUE)
n_traj	Specific to selection="HR" : Number of repeats (trajectories) of the simulated annealing process (default=100).
deltaHi	Specific to selection="HR" : Initial coefficient of variation for jiggling the activity multiplier values (default=0.2). This is set in the parameter file "optimize_fishing.csv" in the case where selection="ECO" and can be manually varied during a run.
attenuationstep	Specific to selection="HR" : Number of itereations between down-steps of the jiggling factor applied to the multiplier values. The jiggling rate is attenuated by a factor of 2 every xx iterations (default=500).
deltaG	Specific to selection="HR" : Coefficient of variation for jiggling the gear linkage values (default=0.25).
nyears	Specific to selection="ECO" : Number of years to run the ecology model in each iteration (default=40).
quiet	Logical. Specific to selection="ECO" : If TRUE then suppress informational messages at the start of each iteration (default=TRUE).

Details

Generic details for both optimization procedures

Simulated annealing is is a probabilistic technique for approximating the global optimum of a given function. As implemented here the process searches the parameter space of a model to locate the combination which maximises the likelihood of a set of observed data corresponding to a suite of

derived outputs. Parameter combinations which result in an improved likelihood may be rejected according to a probability ('temperature') which decreases as the iterations progress. This is to avoid becoming stuck at local likelihood-maxima. The rate at which the 'temperature' decreases is set by a 'cooling' parameter (fraction of previous temperature at each iteration, 0<value<1)

Model configuration and initial values of the ecology model parameters need to be assembled by a prior call of the e2e_read() function.

NOTE that the models.path argument in the e2e_read() function call needs to point to a user workspace folder, not the default North Sea model provided with the package. This is because the optimization function needs write-access to the model /Param folder, but the /extdata/Models folder in the package installation is read-only. To use the annealing function on the North Sea model, use the e2e_copy() function to make a copy of the North Sea model in the user workspace.

The function produces a real-time graphical summary of the progress of the fitting procedure, displaying the likelihoods of the proposed and accepted parameter sets at each iteration.

At the end of the procedure (provided that the argument csv.output=TRUE) a new version of the gear activity multipliers file is exported to the folder /Param of the model version, with a user defined identifier specified by the model.ident argument in the e2e_read() function. The histories of parameter combinations and their likelihood during the course of the optimization procedure are saved as CSV files in the results folder (provided that the argument csv.output=TRUE), and returned as a list object.

The returned or saved data on the progress of the procedures can be be further analysed using the function e2e_plot_opt_diagnostics() to assess the performance of the optimization process.

To preserve the new activity multipliers and incorporate them into the fishing fleet model parameterisation the multiplier values need to be applied to the activity rates specified in the data input file /Param/fishing_activity_*.csv. Manually update the values in fishing_activity_*.csv, by multiplying the existing values by the new multipliers emerging from the annealing process.

If the edited file fishing_activity_*.csv is saved with a new identifier (*) then in order to use it in a subsequent run of the StrathE2E model (using the e2e_run() function) it will be necessary to edit the MODEL_SETUP.csv file in the relevant /Models/variant folder to point to the new file.

Specific details for optimization to observed data in the state of the ecosytem

The observational data to which the activity multipliers are optimized are loaded from the folder Modelname/Variantname/Target/annual_observed_*.csv as part of a e2e_read() function call and are built into the R-list object generated by e2e_read(). Column 3 of annual_observed_* (header: "Use_1.yes_0.no") is a flag to set whether any given row is used in calculating the likelihood of the observed data given the model setup and parameters. Un-used rows of data are omitted from calculations.

The coefficient of variation for jiggling the activity multipliers can be varied in real-time during the run by editing the file "optimize_fishing.csv" in the folder /Param/control/ of the model version. Suggested vaues for the SD are in the range 0.1 to 0.01

The y-axis (likelihood of the target data) range of the real time plot can be varied during the run by editing the setup file "optimize_fishing.csv".

At the end of the procedure the proposed and accepted activity multiplier values and corresponding likleihoods from each iteration of the procedure are saved as csv files in the results folder (provided that the argument csv.output=TRUE), and in a list object which is returned by the function. The two csv files generated by the procedure have names: annealing_ACTmult_proposalhistory-*, annealing_ACTmult_acceptedhistory-*, where * denotes the value of model.ident defined in the preceding e2e_read() function call. The returned list object contains three dataframes: parameter_proposal_history, parameter_accepted_history, new_parameter_data. The proposal and accepted histories can be further analysed with the function e2e_plot_opt_diagnostics() to assess the performance of the optimization process.

Specific details for optimization to known harvest ratios

The task here is a bit harder than normal because of the potentially large overlap in the selectivity patterns of the fishing gears with respect to the living guilds in the ecosystem. So there are some non-standard features to try and avoid local likelihood maxima.

The zonal harvest ratios to which the activity multipliers are optimized are loaded from the folder Modelname/Variantname/Target/zonal_harvest_r_*.csv as part of a e2e_read() function call and are built into the R-list object generated by e2e_read(). Column 4 of taget data file (header: "Use_1.yes_0.no") is a flag to set whether any given row is used in calculating the likelihood of the observed data given the model setup and parameters. Un-used rows of data are omitted from calculations.

To cope with the overlap in selectivity of the fishing gears, the function uses a set of linkage parameters specified in the the file /Param/fishing_gear_linkages.csv These parameters force selected gears activities to vary in concert +/- some variation, rather than independently. The parameters for the gear linkages are located in the file ../Modelname/Variantname/Param/fishing_gear_linkages.csv. The table of linkages specifies which gear activity rates are forced to vary in concert during the fitting process, as opposed to varying independently. The value of the linkage coefficient defines the scaling of changes in the activity rate of a dependent gear relative to its linked independent gear. For example, if gear 8 is permitted to vary independently (value in column "Gear to which linked" = NA and "Linkage coefficient" = NA). If gear 9 is dependent on gear 8 then the activity rate of gear 9 this would be specified by e.g. "Gear to which linked" = 8 and "Linkage coefficient" = 0.645. This would force the activity of gear 9 to be set at gear8_activity * (0.645 +/- a random variation defined by the argument deltaG).

The initial coefficient of variation for searching the parameter space is set by a function argument, and cannot be user-varied dring the run. The CV decreases in steps with increasing iterations. The function repeats the iteration process multiple times. The repeats are referred to as 'trajectories'. Each trajectory starts from and follows a different pathway through the parameter space. At the end of the process, the best-fit set of activity multipliers is selected from across all the trajectories.

At the end of the procedure the activity multiplier values and harvest ratios from the end of each trajectory are saved as csv files in the results folder (provided that the argument csv.output=TRUE), and to a list object which is returned by the function. These data are presented as both raw values, and as relative to the initial values (in the case of the activity multipliers) or the target values (in the case of the harvest ratios). The relativity is expressed as "(optimized - target)/target" for each trajectory (in the case of harvest ratios). The four csv files generated by the procedure have names activity_optim_gearmult_history-*, activity_optim_gearmult_relinitial_history-*, activity_optim_harvestratio_history-*, activity_optim_harvestratio_reltarget_history-*, where * denotes the value of model.ident defined inthe preceding e2e_read() function call. The returned list object contains four dataframes: gear_mult_results, gear_mult_rel_initial, harvest_ratio_results, harvest_ratio_rel_target, new_parameter_data. The relative multiplier and target data can be further analysed with the function e2e_plot_opt_diagnostics() to assess the performance of the optimization process.

Value

A list object containing the data from each trajectory and the final accepted parameter values. Optionally (by default), csv files of the histories and the final accepted parameter values. The latter are returned to the model parameter folder in a format to be read back into the model.

See Also

e2e_ls , e2e_read , e2e_plot_opt_diagnostics , e2e_optimize_eco , e2e_optimize_hr

Examples

```
# Load the 1970-1999 version of the North Sea model supplied with the package and generate a
# quick test data object with only 8 iterations and running the model for only 3 years
# optimising activity rates to the database of observed indices of the state of the ecosystem.
# In this example, the final parameter values are not saved back to the model Param folder.
# More realistic would be at least 500 iterations and running for 50 years.
# Even so this example will take a few minutes to run:
    model<-e2e_read(model.name="North_Sea",
                    model.variant="1970-1999",
                    model.ident="test")
# This model is already optimized to the observed ecosystem data supplied with the package.
# Here, so as to illustrate the optimization process in action, here we perturb the temperature
# driving data to knock the model away from its maximum likelihood state relative to the
# target data:
# add 3 degC to upper layer offshore temperatures:
    model$data$physics.drivers$so_temp <- model$data$physics.drivers$so_temp+3
# add 3 degC to inshore temperatures:
    model$data$physics.drivers$si_temp <- model$data$physics.drivers$si_temp+3
# add 3 degC to lower layer offshore temperatures:
    model$data$physics.drivers$d_temp  <- model$data$physics.drivers$d_temp+3
   test_run  <- e2e_optimize_act(model, selection="ECO", n_iter=8, start_temperature=0.4,
                 cooling=0.975, csv.output=FALSE, nyears=3 )
# View the structure of the returned list:
    str(test_run,max.level=1)
# View the structure of the returned list element containing parameter objects:
    str(test_run$new_parameter_data,max.level=1)
# View the new, final accepted parameter data:
    test_run$new_parameter_data


# ------------------------------------------------------------------------

# This is a dummy example to illustrate a realistic run in which optimised
# parameters are written back to the model Param folder. To try it out substitute
# your own relative folder path in place of \Folder in the e2e_copy() function...
# WARNING - this will take about 26 hours to run...
# Copy the 1970-1999 version of the North Sea model supplied with the package into a
# user workspace relative to the current working directory (../Folder):
#    e2e_copy("North_Sea", "1970-1999",
#                 dest.path="Folder")
# Load the copied version of the North Sea/1970-1999 model from the user workspace
# and assign a path for results data:
# (REPLACE "Folder/Models" and "Folder/results" with your own paths before running)
#    model<-e2e_read(model.name="North_Sea",
#                     model.variant="1970-1999",
#                     models.path="Folder/Models",
#                     results.path="Folder/results",
#                     model.ident="fittingrun")
# Launch the fitting process
#    fitting_data<-e2e_optimize_act(model, selection="ECO", n_iter=500,
#                   start_temperature=0.5, cooling=0.975, csv.output=TRUE, nyears=50)

# ------------------------------------------------------------------------
```

```
# Examples of using the function for optimizing to given zonal harvest ratios
    model<-e2e_read(model.name="North_Sea",
                    model.variant="1970-1999",
                    model.ident="test")
# Activity rates in this model are already optimized to the target harvest ratios supplied with
# the package but we woud not expect to recover these values in this very short demonstration run
   test_run  <- e2e_optimize_act(model, selection="HR", n_iter=30, start_temperature=1.0,
                                 cooling=0.985, csv.output=FALSE, n_traj=5)
# View the structure of the returned list:
    str(test_run,max.level=1)
# View the structure of the returned list element containing parameter objects:
    str(test_run$new_parameter_data,max.level=1)
# View the new, final accepted parameter data:
    test_run$new_parameter_data


# ------------------------------------------------------------------------

# This is a dummy example to illustrate a realistic run in which optimised
# parameters are written back to the model Param folder. To try it out substitute
# your own relative folder path in place of \Folder in the e2e_copy() function...
# WARNING - this will take about 14 hours to run...
# Copy the 1970-1999 version of the North Sea model supplied with the package into a
# user workspace relative to the current working directory (../Folder):
#    e2e_copy("North_Sea", "1970-1999",
#               dest.path="Folder")
# Load the copied version of the North Sea/1970-1999 model from the user workspace
# and assign a path for results data:
# (REPLACE "Folder/Models" and "Folder/results" with your own paths before running)
#    model<-e2e_read(model.name="North_Sea",
#                    model.variant="1970-1999",
#                    models.path="Folder/Models",
#                    results.path="Folder/results",
#                    model.ident="fittingrun")
# Launch the fitting process
#   fitting_data<-e2e_optimize_act(model, selection="HR", n_iter=3000, start_temperature=1.0,
#                                  cooling=0.985, csv.output=TRUE, n_traj=100 )

# ------------------------------------------------------------------------
```

`e2e_optimize_eco` *Optimize StrathE2E ecology model parameters to maximise the likelihood of observed ecosystem target data.*

Description

Launches a StrathE2E simulated annealing process to find the set of ecology model parameters producing the maximum likelihood of observed target data on the state of the ecosystem, given specified environmental driving data and fishing fleet parameters.

Usage

```
e2e_optimize_eco(
```

```
  model,
  nyears = 40,
  n_iter = 500,
  start_temperature = 1,
  cooling = 0.975,
  toppredlock = TRUE,
  quiet = TRUE,
  csv.output = FALSE,
  runtime.plot = TRUE
)
```

Arguments

model	R-list object generated by the e2e_read() function which defined the model configuration.
nyears	Number of years to run the model in each iteration (default=40).
n_iter	Number of iterations of the model (default=500).
start_temperature	Initial value of the simulated annealing temperature parameter (default=1). Suggested values in the range 0.0005 - 5. Higher values increase the probability of rejecting parameter combinations producing an improvement in likelihood.
cooling	Rate at which the simulated annealing temperature declines with iterations (default=0.975). Suggested values in the range 0.9 - 0.985
toppredlock	Logical. If TRUE then locks-down the uptake parameters of the birds pinnipeds and cetaceans as these are hard to fit alongside the other parameters (default=TRUE).
quiet	Logical. If TRUE then suppress informational messages at the start of each iteration (default=TRUE).
csv.output	Logical. If TRUE then enable writing of csv output files (default=FALSE).
runtime.plot	Logical. If FALSE then disable runtime plotting of the progress of the run - useful for testing (default=TRUE)

Details

Simulated annealing is is a probabilistic technique for approximating the global optimum of a given function. As implemented here the process searches the parameter space of a model to locate the combination which maximises the likelihood of a set of observed data corresponding to a suite of derived outputs. Parameter combinations which result in an improved likelihood may be rejected according to a probability ('temperature') which decreases as the iterations progress. This is to avoid becoming stuck at local likelihood-maxima. The rate at which the 'temperature' decreases is set by a 'cooling' parameter (fraction of previous temperature at each iteration, 0<value<1).

Model configuration and initial values of the ecology model parameters need to be assembled by a prior call of the e2e_read() function.

NOTE that the models.path argument in the e2e_read() function call needs to point to a user workspace folder, not the default North Sea model provided with the package. This is because the annealing function needs write-access to the model /Param folder, but the /extdata/Models folder in the package installation is read-only. To use the annealing function on the North Sea model, use the e2e_copy() function to make a copy of the North Sea model in the user workspace.

The observational data to which the ecology parameters are optimized are loaded from the folder Modelname/Variantname/Target/annual_observed_*.csv as part of a e2e_read() function call and

are built into the R-list object generated by e2e_read(). Column 3 of annual_observed_* (header: "Use1_0") is a flag to set whether any given row is used in calculating the likelihood of the observed data given the model setup and parameters. Un-used rows of data are omitted from calculations.

The coefficients of variation for jiggling the ecology parameter can be varied in real-time during the run by editing the file "optimize_ecology.csv" in the folder /Param/control/ of the model version.

The function produces a real-time graphical summary of the progress of the fitting procedure, displaying the likelihoods of the proposed and accepted parameter sets at each iteration. y-axis (likelihood of the target data) range of the real time plot can be varied during the run by editing the setup file "optimize_ecology.csv"

At the end of the procedure, provided that csv.output=TRUE, new versions of the three ecology model 'fitted_parameters..' files are exported to the folder /Param of the model version, with a user defined identifier specified by the model.ident argument in the e2e_read() function. These data are also saved in the list object returned by the function.

In order to use the new fitted parameter values in a subsequent run of the StrathE2E model (using the e2e_run() function) it will be necessary to edit the MODEL_SETUP.csv file in the relevant /Models/variant folder to point to the new files.

Also at the end of the procedure the histories of proposed and accepted ecology model parameter values and corresponding likleihoods from each iteration of the procedure are saved as CSV files in the results folder (provided that the argument csv.output=TRUE), and in a list object which is returned by the function. The two csv files generated by the procedure have names: annealing_par_proposalhistory-*, annealing_par_acceptedhistory-*, where * denotes the value of model.ident defined in the preceding e2e_read() function call. The returned list object contains three dataframes: parameter_proposal_history, parameter_accepted_history, new_parameter_data (a list of three). The proposal and accepted histories can be further analysed with the function e2e_plot_opt_diagnostics() to assess the performance of the optimization process.

Value

A list object containing the histories of proposed and accepted parameters and the final accepted parameter values. Optionally (by default), csv files of the histories and the final accepted parameter values. The latter are returned to the model parameter folder in a format to be read back into the model.

See Also

e2e_ls , e2e_read , e2e_plot_opt_diagnostics , e2e_optimize_hr , e2e_optimize_act

Examples

```
# Load the 1970-1999 version of the North Sea model supplied with the package and generate a
# quick test data object with only 8 itereations and running the model for only 3 years.
# Also, the final parameter values are not saved back to the model Param folder.
# More realistic would be at least 500 iterations and running for 50 years.
# Even so this example will take a few minutes to run:
    model<-e2e_read(model.name="North_Sea",
                    model.variant="1970-1999",
                    model.ident="test")
# This model is already optimized to the observed ecosystem data supplied with the package
# so to illustrate the performance of the process we perturb the temperature driving to knock
# the model away from its maximum likelihood state relative to the target data:
# add 3 degC to upper layer offshore temperatures:
    model$data$physics.drivers$so_temp <- model$data$physics.drivers$so_temp+3
```

```
# add 3 degC to inshore temperatures:
    model$data$physics.drivers$si_temp <- model$data$physics.drivers$si_temp+3
# add 3 degC to lower layer offshore temperatures:
    model$data$physics.drivers$d_temp  <- model$data$physics.drivers$d_temp+3
    test_run  <- e2e_optimize_eco(model, nyears=3, n_iter=8, start_temperature=0.4,
                                  csv.output=FALSE)
# View the structure of the returned list:
    str(test_run,max.level=1)
# View the structure of the returned list element containing parameter objects:
    str(test_run$new_parameter_data,max.level=1)
# View the new preference matrix:
    test_run$new_parameter_data$new_preference_matrix


# ------------------------------------------------------------------------

# This is a dummy example to illustrate a realistic run in which optimised
# parameters are written back to the model Param folder. To try it out substitute
# your own relative folder path in place of \Folder in the e2e_copy() function...
# WARNING - this will take about 26 hours to run...
# Copy the 1970-1999 version of the North Sea model supplied with the package into a
# user workspace relative to the current working directory (../Folder):
#    e2e_copy("North_Sea", "1970-1999",
#               dest.path="Folder")
# Load the copied version of the North Sea/1970-1999 model from the user workspace
# and assign a path for results data:
# (REPLACE "Folder/Models" and "Folder/results" with your own paths before running)
#    model<-e2e_read(model.name="North_Sea",
#                    model.variant="1970-1999",
#                    models.path="Folder/Models",
#                    results.path="Folder/results",
#                    model.ident="fittingrun")
# Launch the fitting process
#    fitting_data <- e2e_optimize_eco(model, nyears=50, n_iter=500, start_temperature=1,
#                                     csv.output=TRUE)

# ------------------------------------------------------------------------
```

`e2e_optimize_hr` *Optimize StrathE2E harvest ratio multipliers to maximum the likelihood of observed ecosystem target data.*

Description

Launches a StrathE2E simulated annealing process to find the set of fishing fleet model harvest ratio multipliers producing the maximum likelihood of observed target data on the state of the ecosystem, given specified environmental driving data and ecology model parameters.

Usage

```
e2e_optimize_hr(
  model,
  nyears = 40,
```

```
  n_iter = 500,
  start_temperature = 1,
  cooling = 0.975,
  quiet = TRUE,
  csv.output = FALSE,
  runtime.plot = TRUE
)
```

Arguments

model	R-list object generated by the e2e_read() function which defined the model configuration.
nyears	Number of years to run the model in each iteration (default=40).
n_iter	Number of iterations of the model (default=500).
start_temperature	Initial value of the simulated annealing temperature parameter (default=1). Suggested values in the range 0.0005 - 5. Higher values increase the probability of rejecting parameter combinations producing an improvement in likelihood.
cooling	Rate at which the simulated annealing temperature declines with iterations (default=0.975). Suggested values in the range 0.9 - 0.985
quiet	Logical. If TRUE then suppress informational messages at the start of each iteration (default=TRUE).
csv.output	Logical. If TRUE then enable writing of csv output files (default=FALSE).
runtime.plot	Logical. If FALSE then disable runtime plotting of the progress of the run - useful for testing (default=TRUE).

Details

Simulated annealing is is a probabilistic technique for approximating the global optimum of a given function. As implemented here the process searches the parameter space of a model to locate the combination which maximises the likelihood of a set of observed data corresponding to a suite of derived outputs. Parameter combinations which result in an improved likelihood may be rejected according to a probability ('temperature') which decreases as the iterations progress. This is to avoid becoming stuck at local likelihood-maxima. The rate at which the 'temperature' decreases is set by a 'cooling' parameter (fraction of previous temperature at each iteration, 0<value<1).

Model configuration and initial values of the ecology model parameters need to be assembled by a prior call of the e2e_read() function.

NOTE that the models.path argument in the e2e_read() function call needs to point to a user workspace folder, not the default North Sea model provided with the package. This is because the annealing function needs write-access to the model /Param folder, but the /extdata/Models folder in the package installation is read-only. To use the annealing function on the North Sea model, use the e2e_copy() function to make a copy of the North Sea model in the user workspace.

The coefficient of variation for jiggling the harvest ratio multipliers can be varied in real-time during the run by editing the file "optimize_fishing.csv" in the folder /Param/control/ of the model version. Suggested vaues for the CV are in the range 0.1 to 0.01

The observational data to which the harvest ratio multiplier parameters are optimized are loaded from the folder Modelname/Variantname/Target/annual_observed_*.csv as part of a e2e_read() function call and are built into the R-list object generated by e2e_read(). Column 3 of annual_observed_* (header: "Use1_0") is a flag to set whether any given row is used in calculating the likelihood of

the observed data given the model setup and parameters. Un-used rows of data are omitted from calculations.

The function produces a real-time graphical summary of the progress of the fitting procedure, displaying the likelihoods of the proposed and accepted parameter sets at each iteration. y-axis (likelihood of the target data) range of the real time plot can be varied during the run by editing the setup file "optimize_fishing.csv"

At the end of the procedure (provided that the argument csv.output=TRUE) a new version of the harvest ratio multipliers file is exported to the folder /Param of the model version, with a user defined identifier specified by the model.ident argument in the e2e_read() function. These new harvest ratio multipliers are also saved in the list object returned by the function.

To preserve the new harvest ratio multipliers and incorporate them into the fishing fleet model parameterisation the multiplier values need to be applied to the scaling parameters which link the integrated effort by each gear to the harvest ratio value which gets piped into the ecology model. Manually update the values in rows 12-21 (excluding the header row) of the file /Param/fishing_fleet_*.csv, by multiplying the existing values by the new multipliers emerging from the annealing process.

If the edited file 'fishing_fleet_*.csv' is saved with a new identifier (*) then in order to use it in a subsequent run of the StrathE2E model (using the e2e_run() function) it will be necessary to edit the MODEL_SETUP.csv file in the relevant /Models/variant folder to point to the new file.

Also at the end of the procedure the histories of proposed and accepted harvest ratio multiplier values and corresponding likleihoods from each iteration of the procedure are saved as CSV files in the results folder (provided that the argument csv.output=TRUE), and in the list object which is returned by the function. The two csv files generated by the procedure have names: annealing_HRmult_proposalhistory-*, annealing_HRmult_acceptedhistory-*, where * denotes the value of model.ident defined in the preceding e2e_read() function call. The returned list object contains three dataframes: parameter_proposal_history, parameter_accepted_history, new_parameter_data (a list of three). The proposal and accepted histories can be further analysed with the function e2e_plot_opt_diagnostics() to assess the performance of the optimization process.

Value

A list object containing the histories of proposed and accepted parameter values and the final accepted parameter values. Optionally (by default), csv files of the histories and the final accepted parameter values. The latter are returned to the model parameter folder in a format to be read back into the model.

See Also

e2e_ls , e2e_read , e2e_plot_opt_diagnostics , e2e_optimize_eco , e2e_optimize_act

Examples

```
# Load the 1970-1999 version of the North Sea model supplied with the package and generate a
# quick test data object with only 8 itereations and running the model for only 3 years.
# Also, the final parameter values are not saved back to the model Param folder.
# More realistic would be at least 500 iterations and running for 50 years.
# Even so this example will take a few minutes to run:
    model<-e2e_read(model.name="North_Sea",
                    model.variant="1970-1999",
                    model.ident="test")
# This model is already optimized to the observed ecosystem data supplied with the package
# so we perturb the temperature driving to knock the model away from its maximum likelihood
# state relative to the target data and illustrate the performance of the process:
```

```
# add 3 degC to upper layer offshore temperatures:
    model$data$physics.drivers$so_temp <- model$data$physics.drivers$so_temp+3
# add 3 degC to inshore temperatures:
    model$data$physics.drivers$si_temp <- model$data$physics.drivers$si_temp+3
# add 3 degC to lower layer offshore temperatures:
    model$data$physics.drivers$d_temp  <- model$data$physics.drivers$d_temp+3
    test_run  <- e2e_optimize_hr(model, nyears=3, n_iter=8, start_temperature=0.4,
                                 csv.output=FALSE)
# View the structure of the returned list:
    str(test_run,max.level=1)
# View the structure of the returned list element containing parameter objects:
    str(test_run$new_parameter_data,max.level=1)
# View the new, final accepted parameter data:
    test_run$new_parameter_data


# ------------------------------------------------------------------------

# This is a dummy example to illustrate a realistic run in which optimised
# parameters are written back to the model Param folder. To try it out substitute
# your own relative folder path in place of \Folder in the e2e_copy() function...
# WARNING - this will take about 26 hours to run...
# Copy the 1970-1999 version of the North Sea model supplied with the package into a
# user workspace relative to the current working directory (../Folder):
#    e2e_copy("North_Sea", "1970-1999",
#             dest.path="Folder")
# Load the copied version of the North Sea/1970-1999 model from the user workspace
# and assign a path for results data:
# (REPLACE "Folder/Models" and "Folder/results" with your own paths before running)
#    model<-e2e_read(model.name="North_Sea",
#                    model.variant="1970-1999",
#                    models.path="Folder/Models",
#                    results.path="Folder/results",
#                    model.ident="fittingrun")
# Launch the fitting process
#    fitting_data <- e2e_optimize_hr(model, nyears=50, n_iter=500, start_temperature=1,
#                                    csv.output=TRUE)

# ------------------------------------------------------------------------
```

`e2e_plot_biomass` *Plot showing the annual annual average biomass densities of each guild in the inshore and offshore zones, optionally with credible intervals.*

Description

Generate plots showing the annual average biomass densities of each guild in the ecology model in the inshore and offshore zones during the final year of a run. The default is to plot data from a single model run but if available, credible intervals of model output from a Monte Carlo analysis can also be plotted.

Usage

```
e2e_plot_biomass(
  model,
  ci.data = FALSE,
  use.saved = FALSE,
  use.example = FALSE,
  results = NULL
)
```

Arguments

model	R-list object defining the model configuration used to generate the data and compiled by the e2e_read() function.
ci.data	Logical. If TRUE plot credible intervals around model results based on Monte Carlo simulation with the e2e_run_mc() function (default=FALSE).
use.saved	Logical. If TRUE use data from a prior user-defined run held as csv files data in the current results folder (default=FALSE).
use.example	Logical. If TRUE use pre-computed example data from the internal North Sea model rather than user-generated data (default=FALSE).
results	R-list object of baseline model output generated by the e2e_run() function. Only needed if ci.data=FALSE, use.saved=FALSE and use.example=FALSE. (Default=NULL).

Details

Note that in this plot the biomass are expressed as mMN/m2, meaning that the mass in each zone has been scaled to the zonal sea surface area. So the data in this plot are area densities and not mass.

Arguments determine the source of model data to be plotted. These can be outputs from a single model run with data held in memory as a list object or in a saved csv file, or from a Monte Carlo simulation (using the function e2e_run_mc()) to estimate credible intervals of model outputs. Generation of credible interval data is a long computing task, so example data for the North Sea model provided with the package are available as example data sets.

If credible intervals are plotted these are displayed as box-and-whiskers. The box spans 50 by a tick mark.

Value

graphical display in a new graphics window.

See Also

e2e_read, e2e_run, e2e_run_mc, e2e_plot_migration, e2e_plot_catch, e2e_plot_trophic, e2e_plot_eco

Examples

```
# Load the 1970-1999 version of the North Sea model supplied with the package,
# run, and generate a plot:
    model <- e2e_read("North_Sea", "1970-1999")
    results <- e2e_run(model, nyears=2)
    e2e_plot_biomass(model, results=results)
```

```
# Direct the graphics output to a pdf file ...
# or jpeg("plot.jpg"), png("plot.png")
    pdf(file.path(tempdir(), "plot.pdf"),width=4,height=6)
    e2e_plot_biomass(model, results=results)
    dev.off()


# Alternatively, plot the same data from a csv file saved by the e2e_run() function. Here the
# csv output is directed to a temporary folder since results.path is not set in the e2e_read()
# function call:
    results <- e2e_run(model, nyears=2,csv.output=TRUE)
    dev.new()
    e2e_plot_biomass(model, use.saved=TRUE)


# For the same model, plot the example data with credible intervals:
# This example requires the Strathe2E2examples supplementary data package.
if(require(StrathE2E2examples)){
    e2e_plot_biomass(model, ci.data=TRUE, use.example=TRUE)
}
```

`e2e_plot_catch` *Plot the distribution and composition of annual catches across guild, zones and gears in the final year of a model run.*

Description

Create stacked barplots of the distribution of offshore and inshore landings and discards across gears by guild, or across guilds by gears, in the final year of a model run.

Usage

```
e2e_plot_catch(model, results, selection = "BY_GEAR")
```

Arguments

`model`	R-list object defining the baseline model configuration used to generate the data and compiled by the e2e_read() function.
`results`	List object of single-run model output generated by running the function e2e_run() function.
`selection`	Text string from a list identifying the group of model output variables to be plotted. Select from: "BY_GUILD", "BY_GEAR". With the former, each panel represents a different guild; with the latter, each panel represnts a different gear. Remember to include the phrase within "" quotes.

Details

Data in zonal landings and discards by guild and gear in the final year of a model run are generated as a standard output from the model, and saved both as csv files and in the results object returned by the e2e_run() function. This function organises these data in two different ways to display as barplots.

The first display is a multi-panel plot in which each panel represents a different guild, and the bars show the zonal landings by each gear. The alternative display has each panel as a different gear, and the bars show the landings and discards of each guild.

The unit of the displayed data are mMN/y from the model domain as a whole, which is taken as being 1m2.

Value

graphical display in a new graphics window.

See Also

e2e_read, e2e_run, e2e_run_mc, e2e_plot_migration, e2e_plot_eco, e2e_plot_trophic, e2e_plot_biomass

Examples

```
# Load the 1970-1999 version of the North Sea model supplied with the package, run, and
# generate a plot:
    model <- e2e_read("North_Sea", "1970-1999")
    results <- e2e_run(model, nyears=2,csv.output=FALSE)
    e2e_plot_catch(model, results, selection="BY_GEAR")
    dev.new()
    e2e_plot_catch(model, results, selection="BY_GUILD")
```

---

e2e_plot_eco

Plot daily data on ecological outputs from the model over the final year of a run, optionally with credible intervals.

Description

Generate a multi-panel set of one-year time series plots of selected outputs on the concentrations of state variables in the ecology model. The default is to plot data from a single model run but if available, credible intervals of model output from a Monte Carlo analysis can be plotted instead.

Usage

```
e2e_plot_eco(
  model,
  selection = "NUT_PHYT",
  ci.data = FALSE,
  use.saved = FALSE,
  use.example = FALSE,
  results = NULL
)
```

Arguments

model	R-list object defining the baseline model configuration used to generate the data and compiled by the e2e_read() function.

selection	Text string from a list identifying the group of model output variables to be plotted. Select from: "NUT_PHYT", "SEDIMENT", "ZOOPLANKTON", "FISH", "BENTHOS", "PREDATORS", "CORP_DISC", "MACROPHYTE", Remember to include the phrase within "" quotes.
ci.data	Logical. If TRUE plot credible intervals around model results based on Monte Carlo simulation with the e2e_run_mc() function (default=FALSE).
use.saved	Logical. If TRUE use data from a prior user-defined run held as csv files data in the current results folder (default=FALSE).
use.example	Logical. If TRUE use pre-computed example data from the internal North Sea model rather than user-generated data (default=FALSE).
results	R-list object of model output generated by the e2e_run() function. Only needed if ci.data=FALSE, use.saved=FALSE and use.example=FALSE. (Default=NULL).

Details

Arguments determine the source of model data to be plotted. These can be outputs from a single model run with data held in memory as a list object or in a saved csv file, or from a Monte Carlo simulation (using the function e2e_run_mc()) to estimate credible intervals of model outputs. Generation of credible interval data is a long computing task, so example data for the North Sea model provided with the package are available as an illustration.

If plotting of credible intervals is selected, results from the maximum likelihood model are shown by a red line. The median of the credible values distribution is shown my a solid black line. A grey-shaded area indicates the 50 of simulated values). Black dashed lines span the 99

Variables to be plotted (and units) depending on values of "selection":

- NUT_PHYT : Water column nitrate, ammonia, detritus and phytoplankton (mMN/m3)
- SEDIMENT : Sediment porewater nitrate and ammonia, detritus and corpses (mMN/m3 or gN/gDW)
- ZOOPLANKTON : Omnivorous and carnivorous zooplankton (mMN/m2)
- FISH : Planktivorous and demersal fish and fish larvae (mMN/m2)
- BENTHOS : Susp/dep and carn/scav benthos and benthos larvae (mMN/m2)
- PREDATORS : Birds, pinnipeds, cetaceans and migratory fish (mMN/m2)
- CORP_DISC : Corpses and discards (mMN/m2)
- MACROPHYTE : Macrphytes and macrophyte debris (mMN/m2)

To direct the graph output to a file rather than the screen, wrap the e2e_plot_eco() function call in a graphical device call: Since the plot pages contain different numbers of panels the suggested width:height ratios are as follows:

- NUT_PHYT .... 1.5 : 1
- SEDIMENT ... 0.67 : 1
- ZOOPLANKTON ... 1 : 1
- FISH ..... 2 : 1
- BENTHOS ..... 2 : 1
- PREDATORS ..... 2 : 1
- CORP_DISC ..... 1 : 1
- MACROPHYTE .... 2 : 1

Value

Graphical display in a new graphics window.

See Also

e2e_read, e2e_run, e2e_run_mc, e2e_plot_migration, e2e_plot_catch, e2e_plot_trophic, e2e_plot_biomass

Examples

```
# Load the 1970-1999 version of the North Sea model supplied with the package, run, and
# generate a plot:
    model <- e2e_read("North_Sea", "1970-1999")
    results <- e2e_run(model, nyears=2,csv.output=FALSE)
    e2e_plot_eco(model, selection="NUT_PHYT",results=results)
    dev.new()
    e2e_plot_eco(model, selection="ZOOPLANKTON",results=results)

# Direct the graphics output to a pdf file ...
# or jpeg("plot.jpg"), png("plot.png")
    pdf(file.path(tempdir(), "plot.pdf"),width=8,height=4)
    e2e_plot_eco(model, selection="FISH",results=results)
    dev.off()

# Load the 1970-1999 version of the North Sea model supplied with the package and
# plot example credible interval data:
# This example requires the Strathe2E2examples supplementary data package.
if(require(StrathE2E2examples)){
    e2e_plot_eco(model, selection="BENTHOS",ci.data=TRUE,use.example=TRUE)
}
```

---

e2e_plot_edrivers *Plot climatological year of environmental driving data.*

Description

Multi-panel time series plots of climatological annual cycles of driving data as provided in the input csv files.

Usage

```
e2e_plot_edrivers(model)
```

Arguments

model	R-list object defining the model configuration compiled by the e2e_read() function.

Details

The function plots a multi-panel page of time series plots of monthly values of the environmental driving data for the model.

Units for the plotted variables are as follows:

- Sea surface irradiance: uE/m2/d
- Suspended particulate matter: g/m3
- Temperature: deg-C
- Vertical diffusivity gradient: m/d (derived from the vertical diffusivity (m2/s) and mixing length scale (m))
- External inflows: m3 per m2 sea surface of model domain (derived from proportion input per layer volume, layer thicknesses and areas)
- River discharge: m3 per m2 sea surface of model domain (derived from proportion input to inshore volume, and inshore layer thickness and area)
- Inshore significant wave height: m
- Proportion of seabed disturbed: /d (aggregated over the three sediment classes in each zone)
- External boundary nitrate concentration: mMN/m3
- External boundary ammonia concentration: mMN/m3
- External boundary phytoplankton concentration: mMN/m3
- External boundary detritus concentration: mMN/m3
- River nitrate concentration: mMN/m3
- River ammonia concentration: mMN/m3
- Atmospheric nitrate deposition flux: mMN/m2/d
- Atmospheric ammonia deposition flux: mMN/m2/d

Value

Graphical display in a new graphics window. Does not return any data object since the data plotted are all available as input csv files.

See Also

e2e_read, e2e_run, e2e_plot_fdrivers

Examples

```
# Load the 2003-2013 version of the North Sea model supplied with the package:
    model <- e2e_read("North_Sea", "2003-2013")
# Plot the annual cyles of driving data
    e2e_plot_edrivers(model)

# Direct the graphics output to a pdf file ...
# or jpeg("plot.jpg"), png("plot.png")
    pdf(file.path(tempdir(), "plot.pdf"),width=8,height=6)
    e2e_plot_edrivers(model)
    dev.off()
```

`e2e_plot_fdrivers` *Plot shaded-grid-cell diagrams of the distributions of fishing-related drivers across gears, guilds and seabed habitats.*

Description

Select from a list of fishing-related drivers of the model (gear activity rates, seabed abrasion rates, harvest ratios, discard rates and offal generation rates), and generate a shaded-grid-cell diagram showing the distribution across gears, guilds and seabed habitats.

Usage

```
e2e_plot_fdrivers(model, selection)
```

Arguments

`model`	R-list object defining the model configuration corresponding to the data to be plotted, compiled by the e2e_read() function.
`selection`	Text string from a list identifying the class of fishing-related driving data to be plotted. Select from: "ACTIVITY", "ABRASION", "HARVESTR", "DISCARDS", "OFFAL". Remember to include the phrase within "" quotes.

Details

Details relating to fishing activity distributions:

The function plots a matrix of seabed habitats vs fishing gears with each cell shaded to indicate log(e) activity density (white = 0, purple = high).

The spatial distribution of fishing gear activity in the model is defined by two input data sets:

- Vector of whole domain activity density of each fishing gear (s/d per m2 of whole model domain)
- Matrix of the proportional distribution of whole domain activity density of each gear (rows) across seabed habitats (columns)

The activity density of each gear in each habitat is then obtained by multiplying the whole domain activity density into the proportional distribution matrix, and dividing by the area-proportions of habitats in the domain. The units of habitat-specific activity density are then (s/d/m2).

The vector of area-proportions of each habitat in the model domain is part of the moldel configuration parameter set.

The calculation take account of any activity multipliers set in the csv inputs to be applied to the activity density.

When interpreting patterns of activity density, bear in mind that the impact (harvest ratio, seabed abrasion) per unit activity can vary considerably between gears.

Details relating to seabed abrasion rate distributions:

The function plots a matrix of seabed habitats vs fishing gears with each cell shaded to indicate abrasion area ratio (log(e) scaled proportion of habitat area abraded per day; white = 0, purple = high).

The spatial distribution of seabed abrasion in the model is defined by three input data sets:

- Vector of whole domain activity density of each fishing gear (s/d per m2 of whole model domain)
- Matrix of the proportional distribution of whole domain activity density of each gear (rows) across seabed habitats (columns)
- Vector of seabed abrasion rate of each fishing gear (m2/s)

The activity density of each gear in each habitat is then obtained by multiplying the whole domain activity density into the proportional distribution matrix, and dividing by the area-proportions of habitats in the domain. The units of habitat-specific activity density are then (s/d/m2).

The product of activity density per gear and habitat and gear abrasion rate is then the abrasion area ratio, or abrasion rate (m2/m2/d)

The vector of area-proportions of each habitat in the model domain is part of the moldel configuration parameter set.

The calculation take account of any activity multipliers set in the csv inputs to be applied to the activity density.

Details relating to harvest ratio distributions:

The function plots a matrix of the values of harvest ratio (proportion of exploitable biomass of each guild capatured per day) for each guild in the ecology model (columns) arising from the activity of each gear (rows). Hence each row represents the selectivity pattern of each gear with respect to guilds. Cells in the matrix are shaded to indicate log(e) transformed harvest ratio (colour gradient: white = 0, purple = high).

Harvest ratio is calculated by the fishing fleet model and piped into the ecology model. The fleet model takes inputs of activity density by each gear, and a matrix of fishing power, to calculate effort (effort = activity * power). Harvest ratio is assumed to be lineally related to effort by a scaling coefficient which is input as a parameter to the fleet model.

Note that the visualization of harvest ratio generated by this function is based on the csv input data to the fleet model (including any multipliers to be applied to either fishing activity or harvest ratio scaling parameters). However, it does not reflect any effects on harvest ratio arising from discarding scenarios since these are dynamic and generated at run-time within the ecology model.

Details relating to discard rate distributions:

The function plots a matrix of the values of discard rate (proportion by weight of catch discarded at sea prior to any at-sea processing (evisceration)) for each guild in the ecology model (columns) arising from the activity of each gear (rows). Cells in the matrix are shaded to indicate discard rate (range 0 - max; colour gradient: white = 0, purple = high) on a linear scale.

It is possible (and allowable) for the discard rate of a gear-guild combination to be set as a positive number in the 'fishing_discards' csv input file, but nevertheless the the catching power settings in the corresponding 'fishing_power' file to be set to zero - in other words the given gear does not actually catch the given guild. In such cases this function resets the discard rate to NA for plotting purposes.

Note that the visualization of discard rate generated by this function is based on the csv input data to the fleet model. Hence, it does not reflect any effects on discard rate arising from discarding scenarios configured in the fleet model input since these are dynamic and generated during run-time within the ecology model.

Details relating to offal generation rate distributions:

The function plots a matrix of the values of offal generation rate (i.e. the proportion by weight of total catch which is returned to the sea as viscera after processing) for each guild (columns) by each gear (rows). Cells in the matrix are shaded to indicate offal generation rate (range 0 - max; colour gradient: white = 0, purple = high) on a linear scale.

The offal generation rate is a quantity derived from three input variables to the fishing fleet model:

- Matrix of the discard rate (proportion of catch weight discarded without being processed) for each guild and gear; (D); (input file '/Param/fishing_discards*.csv')
- Matrix of the processing-at-sea rate (proportion of retained catch which is processed) for each guild and gear; (P); (input file '/Param/fishing_processing*.csv')
- Proportion by weight of viscera in the retained catch (assumed constant across all guilds); (V); (value in input file '/Param/fishing_fleet_*.csv')

The offal generation rate is then given by (1 - D).P.V

It is possible (and allowable) for the discard rate and processing-at-sea rate of a gear-guild combination to be set as positive numbers in the relevant input files, but nevertheless the the catching power settings in the corresponding 'fishing_power' file to be set to zero - in other words the given gear does not actually catch the given guild. In such cases this function resets the discard rate and processing-at-sea rate to NA for plotting purposes.

Note that the visualization of offal rate generated by this function is based on the csv input data to the fleet model. Hence, it does not reflect any effects on discard rate arising from discarding scenarios configured in the fleet model input since these are dynamic and generated during run-time within the ecology model.

Value

Graphical display in a new graphics window and a list object comprising an array of the data in the plot and the axis labels.

See Also

e2e_read, e2e_run, e2e_plot_edrivers

Examples

```
# Load the 1970-1999 version of the North Sea model supplied with the package, run, and
# generate a plot:
    model <- e2e_read("North_Sea", "2003-2013")
    plotted_data<-e2e_plot_fdrivers(model, selection="ACTIVITY")
    str(plotted_data,max.level=1)
```

---

e2e_plot_migration *Plot daily data on migration fluxes by actively mobile guilds during the final year of a run, optionally with credible intervals.*

Description

Generate a multi-panel set of one-year time series plots of the mass fluxes between inshore and offshore zones due to migration by actively mobile guilds in the ecology model: all three fish guilds, birds, pinnipeds and cetaceans. The default is to plot data from a single model run but if available, credible intervals of model output from a Monte Carlo analysis can be plotted instead.

Usage

```
e2e_plot_migration(
  model,
  ci.data = FALSE,
  use.saved = FALSE,
  use.example = FALSE,
  results = NULL
)
```

Arguments

model	R-list object defining the baseline model configuration used to generate the data and compiled by the e2e_read() function.
ci.data	Logical. If TRUE plot credible intervals around model results based on Monte Carlo simulation with the e2e_run_mc() function (default=FALSE).
use.saved	Logical. If TRUE use data from a prior user-defined run held as csv files data in the current results folder (default=FALSE).
use.example	Logical. If TRUE use pre-computed example data from the internal North Sea model rather than user-generated data (default=FALSE).
results	R-list object of model output generated by the e2e_run() function. Only needed if ci.data=FALSE, use.saved=FALSE and use.example=FALSE. (Default=NULL).

Details

Daily interval post-processed data from the Monte Carlo e2e_run_mc() function are stored in the file ../Modelname/Variantname/CredInt/CredInt_processed_daily_migrations-*.csv, where * represents the model run identifier (model.ident) text embedded in the R-list object created by the e2e_read() function. The path to this file is relative to the value of results.path as set by a prior e2e_read() function call.

Arguments determine the source of model data to be plotted. These can be outputs from a single model run with data held in memory as a list object or in a saved csv file, or from a Monte Carlo simulation (using the function e2e_run_mc()) to estimate credible intervals of model outputs. Generation of credible interval data is a long computing task, so data for the North Sea model provided with the package are available as example data sets.

Each panel of the plot shows a time-series of the net flux densities (mMN/d in the model domain as a whole, assumed to be 1 m2 sea surface area) of one of the migratory guilds in the model (all three guilds of fish, birds, pinnipeds and cetaceans) between the inshore and offshore zones of the model, over the final year of a run. These migration fluxes are the dynamic product of gradients in the ratio of food concentration to predator concentration across the inshore-offshore boundary. Positive values of the net migration flux indicate net movement from the offshore to inshore zone. Negative values indicate net movement from inshore to offshore.

If plotting of credible intervals is selected, results from the maximum likelihood model are shown by a red line. The median of the credible values distribution is shown my a solid black line. A grey-shaded area indicates the 50 of simulated values). Black dashed lines span the 99

For details of how the distribution of credible output values from StrathE2E are calculated see the help information for the e2e_run_mc() function.

Value

Graphical display in a new graphics window.

See Also

e2e_read, e2e_run, e2e_run_mc, e2e_plot_eco, e2e_plot_catch, e2e_plot_trophic, e2e_plot_biomass

Examples

```
# Load the 1970-1999 version of the North Sea model supplied with the package,
# run, and generate a plot:
    model <- e2e_read("North_Sea", "1970-1999")
    results <- e2e_run(model, nyears=2,csv.output=FALSE)
    e2e_plot_migration(model, results=results)

# Direct the graphics output to a pdf file ...
# or jpeg("plot.jpg"), png("plot.png")
    pdf(file.path(tempdir(), "plot.pdf"),width=8,height=6)
    e2e_plot_migration(model, results=results)
    dev.off()

# For the same model, plot the example data with credible intervals:
# This example requires the Strathe2E2examples supplementary data package.
if(require(StrathE2E2examples)){
    e2e_plot_migration(model, ci.data=TRUE, use.example=TRUE)
}
```

---

e2e_plot_opt_diagnostics

Plot diagnostic data on the performance of parameter optimization functions.

Description

Creates diagnostic plots from outputs of the functions for optimizing ecology, harvest ratio, and fishing activity parameters: e2e_optimize_eco(), e2e_optimize_hr(), and e2e_optimize_act() respectively.

Usage

```
e2e_plot_opt_diagnostics(
  model,
  selection = "",
  fitted.to = "",
  use.saved = FALSE,
  use.example = FALSE,
  results = NULL
)
```

Arguments

model	R-list object defining the baseline model configuration used to generate the data and compiled by the e2e_read() function.

selection	Text string from a list identifying which type of optimization procedure generated the data to be plotted. Select from: "ECO", "HR", "ACT", corresponding to the functions e2e_optimize_eco(), e2e_optimize_hr(), and e2e_optimize_act(). Remember to include the phrase within "" quotes.
fitted.to	Specific to the case where selection="ACT"; text string from a list identifying which version of activity optimization procedure generated the data to be plotted. Select from: "ECO", "HR", corresponding to maximising the likelihod of ecosystem state data, or zonal harvest ratios respectively. Default="". Remember to include the phrase within "" quotes.
use.saved	Logical. If TRUE use data from a prior user-defined run held as csv files data in the current results folder (default=FALSE).
use.example	Logical. If TRUE use pre-computed example data from the internal North Sea model rather than user-generated data (default=FALSE).
results	R-list object of output from an optimization process generated by the e2e_optimize_eco(), e2e_optimize_hr(), or e2e_optimize_act() function. Only needed if use.saved1=FALSE and use.example1=FALSE. (Default=NULL).

Details

The function takes the history of parameter values proposed during each simulated annealing process, expresses these relative to the initial value of each parameter, generates qunatiles (0.005, 0.25, 0.5, 0.75 and 0.995) of the distribution for each parameter, and plots the results as box-and-whisker diagrams. The relativity to initial values is expressed as (proposed - initial)/initial so the initial is represented by a relative value of zero. For each parameter, the final accepted value is over-plotted onto the box-and-whisker as a red bar. The resulting diagram, with a separate box-and-whisker for each parameter shows the extent to which each parameter has migrated from its initial value en-route to the final accepted state.

The format of the parameter diagram is the same for each type of optimization scheme: ecology, harvest ratio, and fishing activity parameters. However, in the case where activity parameters are optimized to zonal harvest ratios rather than ecosystem state data, the diagram has an additional panel showing the distribution of proposal harvest ratios relative to the target, and (in red) the harvest ratios corresponding to the final accepted activity parmeters (relative to the targets).

Arguments for this function permit the input data to be drawn from an existing data object generated by the various optimization functions, previously genearted csv files, or example data provided with the package for versions of the internal North Sea models.

Documentation in a dataframe format on each of the classes of parameters in the model can be obtained with the function e2e_get_parmdoc(). This provides a key to the abbreviated parameter names especially in the diagnostic plots for e2e_optimize_eco().

As well as drawing the plot the function returns a list object containing a) an array of the quantiles of the proposal distributikn for each parameter, and b) an array of one row, of the final accepted parameter set from the procedure.

Value

List object of results from which the plot is created, graphical display in a new graphics window.

See Also

e2e_read, e2e_get_parmdoc, e2e_optimize_eco, e2e_optimize_hr, e2e_optimize_act

Examples

```
# The examples provided here are illustration of how to generate diagnostics plots
# from optimization runs of the model. Optimization runs are very time consuming so
# the examples only involve a bare minimum of model runs and are not realistic.
# Alternatively, data in the package StrathE2E2examples can be used to generate
# example plots.

# ------------------------------------------------------------------------


# Load the 1970-1999 version of the North Sea model supplied with the package and generate
# a quick test data object with only 8 itereations and running the model for 3 years.
# More realistic would be at least 500 iterations and running for 50 years.
# Even so this example will take a few minutes to run:
    model<-e2e_read(model.name="North_Sea",
                    model.variant="1970-1999",
                    model.ident="test")
# This model is already optimized to the observed ecosystem data supplied with the package
# Perturb the temperature driving to knock the model away from its maximum likelihood
# state relative to the target data to illustrate the performance:
# add 3 degC to upper layer offshore temperatures:
    model$data$physics.drivers$so_temp <- model$data$physics.drivers$so_temp+3
# add 3 degC to inshore temperatures:
    model$data$physics.drivers$si_temp <- model$data$physics.drivers$si_temp+3
# add 3 degC to lower layer offshore temperatures:
    model$data$physics.drivers$d_temp  <- model$data$physics.drivers$d_temp+3
    test_run  <- e2e_optimize_eco(model, nyears=3, n_iter=8, start_temperature=0.4,
                                  csv.output=FALSE)
    plot_data <- e2e_plot_opt_diagnostics(model,selection="ECO",results=test_run)
# Red bars shows the final accepted values of each parameters, boxes and whiskers
# show the dirtribution of paramater values explored, relative to the starting values
    str(plot_data,max.level=1)  # show the structure of the list object plot_data


# ------------------------------------------------------------------------

# Or... plot example date supplied with the package showing some data generated during
# the process of optimizing the North Sea model:
# This example requires the Strathe2E2examples supplementary data package.
if(require(StrathE2E2examples)){
    model     <- e2e_read(model.name="North_Sea", model.variant="1970-1999")
    plot_data <- e2e_plot_opt_diagnostics(model,selection="ECO",use.example=TRUE)
# Red bars shows the final accepted values of each parameters, boxes and whiskers
# show the dirtribution of paramater values explored, relative to the starting values
# Example data are only available for the 1970-1999 variant of the North Sea model
}

# ------------------------------------------------------------------------


# Same for harvest ratio optimization...
    model<-e2e_read(model.name="North_Sea",
                    model.variant="2003-2013",
                    model.ident="test")
# This model is already optimized to the observed ecosystem data supplied with the package
# Perturb the temperature driving to knock the model away from its maximum likelihood
```

```
# state relative to the target data to illustrate the performance o fthe process:
# add 3 degC to upper layer offshore temperatures:
    model$data$physics.drivers$so_temp <- model$data$physics.drivers$so_temp+3
# add 3 degC to inshore temperatures:
    model$data$physics.drivers$si_temp <- model$data$physics.drivers$si_temp+3
# add 3 degC to lower layer offshore temperatures:
    model$data$physics.drivers$d_temp  <- model$data$physics.drivers$d_temp+3
    test_run  <- e2e_optimize_hr(model, nyears=3, n_iter=8, start_temperature=0.4,
                                 csv.output=FALSE)
    plot_data <- e2e_plot_opt_diagnostics(model,selection="HR",results=test_run)
# Red bars shows the final accepted values of each parameters, boxes and whiskers
# show the dirtribution of paramater values explored, relative to the starting values
    str(plot_data,max.level=1)  # show the structure of the list object plot_data


# ------------------------------------------------------------------------

# Or... plot example date supplied with the package showing some data generated
# during the process of optimizing the North Sea model:
# This example requires the Strathe2E2examples supplementary data package.
if(require(StrathE2E2examples)){
    model     <- e2e_read(model.name="North_Sea", model.variant="2003-2013")
    plot_data <- e2e_plot_opt_diagnostics(model,selection="HR",use.example=TRUE)
# Red bars shows the final accepted values of each parameters, boxes and whiskers
# show the dirtribution of paramater values explored, relative to the starting values
# Example data are only available for the 2003-2013 variant of the North Sea model
}

# ------------------------------------------------------------------------


# For activity rate optimization relative to ecosystem data:
    model<-e2e_read(model.name="North_Sea",
                    model.variant="1970-1999",
                    model.ident="test")
# This model is already optimized to the observed ecosystem data supplied with the package,
# but not by optimizing gear activity rates
# The e2e_optimize_eco() function was used in this case.
# Perturb the temperature driving to knock the model away from its maximum likelihood
# state relative to the target data:
# add 3 degC to upper layer offshore temperatures:
    model$data$physics.drivers$so_temp <- model$data$physics.drivers$so_temp+3
# add 3 degC to inshore temperatures:
    model$data$physics.drivers$si_temp <- model$data$physics.drivers$si_temp+3
# add 3 degC to lower layer offshore temperatures:
    model$data$physics.drivers$d_temp  <- model$data$physics.drivers$d_temp+3
   test_run  <- e2e_optimize_act(model, selection="ECO", n_iter=8, start_temperature=0.4,
                                 cooling=0.975, csv.output=FALSE, nyears=3)
    plot_data <- e2e_plot_opt_diagnostics(model,selection="ACT",
                                          fitted.to="ECO",results=test_run)
# Red bars shows the final accepted values of each parameters, boxes and whiskers
# show the dirtribution of paramater values explored, relative to the starting values
    str(plot_data,max.level=1)  # show the structure of the list object plot_data

# There are no example data available in the package for this function
```

```
# ------------------------------------------------------------------------


# For activity rate optimization relative to zonal harvest ratios:
    model<-e2e_read(model.name="North_Sea",
                    model.variant="1970-1999",
                    model.ident="test")
# Activity rates in this model are already optimized to the target harvest ratios supplied with
# the package but we woud not expect to recover these values in this short demonstration run
   test_run  <- e2e_optimize_act(model, selection="HR", n_iter=30, start_temperature=1.0,
                                 cooling=0.985, csv.output=FALSE, n_traj=5 )
    plot_data <- e2e_plot_opt_diagnostics(model,selection="ACT",fitted.to="HR",
                                          results=test_run)
# Red bars shows the final accepted values of each parameters, boxes and whiskers
# show the dirtribution of paramater values explored, relative to the starting values
    str(plot_data,max.level=1)  # show the structure of the list object plot_data


# ------------------------------------------------------------------------

# Or... plot example date supplied with the package showing some data generated during
# the process of optimizing the North Sea model:
# This example requires the Strathe2E2examples supplementary data package.
if(require(StrathE2E2examples)){
    model     <- e2e_read(model.name="North_Sea", model.variant="1970-1999")
    plot_data <- e2e_plot_opt_diagnostics(model,selection="ACT",fitted.to="HR",
                                          use.example=TRUE)
# Red bars shows the final accepted values of each parameters, boxes and whiskers
# show the dirtribution of paramater values explored, relative to the starting values
# Example data are only available for the 1970-1999 variant of the North Sea model
}

# ------------------------------------------------------------------------
```

`e2e_plot_sens_mc` *Plot diagnostic data from sensitivity and Monte Carlo analyses.*

Description

The function generates diagnostic plots from either a parameter sensitivity analysis or a Monte Carlo simulation of credible intervals of model outputs.

Usage

```
e2e_plot_sens_mc(model, selection = "SENS", use.example = FALSE)
```

Arguments

`model`	R-list object defining the model configuration, compiled by the e2e_read() function
`selection`	Text string from a list identifying source of data to bed merged. Select from: "SENS", "MC", referring to sensitivity analysis or Monte Carlo analysis. Remember to include the phrase within "" quotes.

use.example Logical. If TRUE use pre-computed example data from the internal North Sea model rather than user-generated data (default=FALSE).

Details

In both cases, the inputs required are a model list object created by the e2e_read() function defining the model configuration and relevant model.ident argument to point to the required files, and a logical argument to determne the origin of data files (either saved as csv files in the user workspace, or example data for the North Sea model provided with the package.

Details relating to sensitivity analysis plot:

The function reads processed results generated by the function e2e_run_sens(), or the function e2e_process_sens_mc(...,selection="SENS"), and plots the Elementary Effect mean (x-axis; magnitude of sensitivity) against Elementary Effect standard deviation (y-axis; strength of interactions between parameters).

Processed results from the function e2e_run_sens(), or the function e2e_process_sens_mc(...,selection="SENS"), are stored in the csv file ../Modelname/Variantname/sorted_parameter_elementary_effects-*.csv where * represents the model run identifier (model.ident) text embedded in the R-list object created by the e2e_read() function. The path to this file is relative to the results folder specified in teh e2e_read() function call.

Each symbol in the plot represents a single parameter in the model. The parameters are colour-coded to indicate 6 different types - fitted and fixed parameters of the ecology model, fishing fleet model parameters, fishery harvest ratios, environmental drivers, and physical configuration parameters.

The plot also shows a wedge formed by the two dashed lines. These correspond to +/-2 standard errors of the mean, so for points falling outside of the wedge there is a significant expectation that the distribution of elementary effects is non-zero. For points falling within the wedge the distribution of elementary effects is not significantly different from zero.

For details of how the Elementary Effect values are derived for each parameter see help(e2e_run_sens)

Details relating to Monte Carlo analysis plot:

The function creates a plot showing the credible distributions of ecology model parameters based on the results from the e2e_run_mc() function. These distributions are formed from the input distributions to the Monte Carlo process, weighted by the likelihood of observed target data on the state of the ecosystem given each combination of parameter values.

Post-processed data from the e2e_run_mc() function are stored in the file ../Modelname/Variantname/CredInt/CredInt_pro
*.csv, where * represents the model run identifier (model.ident) text embedded in the R-list object created by the e2e_read() function. The path to this file is relative to the results folder specified in teh e2e_read() function call.

Each parameter in the plot is scaled to its baseline value as (value - baseline)/baseline. Ideally, the baseline is the maximum likelihood model developed by application of the optimization functions available in the package (e.g. e2e_optimize_eco()). Each parameter is then represented by a box and whisker plot which shows the distribution of credible parameter values around zero, i.e. around the baseline. The median of the credible values distribution for each parameter is shown my a black tick-mark. The box spans the 50 of simulated values). Whisker lines span the 99

The individual parameters are identified by numbers (rather than text names). These numbers correspond to the column numnbers in the file ../Modelname/Variantname/CredInt/CredInt_processed_parameters-*.csv . Details of the parameters associated with each identification number are available as a dataframe by using the function e2e_get_parmdoc().

The input distribution of parameter values to the Monte Carlo process is drawn from a random uniform distribution with limits specified in the monte_carlo control file for the model setup (located in a sub-folder of the /Param/control folder). This distribution is shown by a red box and whisker at

the bottom of the plot. Given the random uniform input we expect the quartiles (limits of the box) to be symetrical and located mid-way between zero and the upper and lower extremes. Vertical red lines show the expected limits of the quartiles boxes if model results were completely insensitive to individual parameter values.

The extent to which individual parameter distributions deviate from the random uniform input is an indication of their sensitivity in the model. Parameters whose distributions are tightly confined around zero (the baseline value) are highly sensitive.

For some parameters, in particular the preference parameters, their credible distributions may span a wider range than the inputs. This may seem unexpected, but arises because within each feeding guild the preference parameters are not independent of each other. The preferences within each guild must sum to 1. Hence, during the Monte Carlo process, after drawing new values of the preference values they are all rescaled to sum to 1, which may mean that some of them will have been varied by more than the original coefficient of variation of the input random uniform.

For details of how the distribution of credible output values from StrathE2E are calculated see the help information for the e2e_run_mc() function.

Value

Graphical display in a new graphics window.

References

For details on the sensitivity analysis method see: Morris, M.D. (1991). Factorial sampling plans for preliminary computational experiments. Technometrics, 33, 161-174.

For a review of sensitivity analysis methods including the Morris Method see: Wu, J. et al. (2013). Sensitivity analysis of infectious disease models: methods, advances and their application. J R Soc Interface 10: 20121018, 14pp.

See Also

e2e_read, e2e_run_mc, e2e_run_sens, e2e_get_parmdoc

Examples

```
# These examples requires the Strathe2E2examples supplementary data package.

# Load the 1970-1999 version of the North Sea model supplied with the package:
if(require(StrathE2E2examples)){
    model <- e2e_read("North_Sea", "1970-1999")
# Plotting illustrated here using the example data sets, but you could run the
# functions e2e_run_sens() or e2e_run_mc() to generate some results instead.

# Plot the results of a parameter sensitivity analysis. Each point represents one parameter.
# The x-axis represents the sensitivity of the overall likelihood of the observed data to
# The y-axis represents the extent to which each parameter interacts with others.
    e2e_plot_sens_mc(model, selection="SENS", use.example=TRUE)
}

#-------------------------------------------------------------------------

# Plot the credible distributions of ecology model parameters based on the results of a
# Monte Carlo analysis. These distributions are formed from the input distributions to
# the Monte Carlo process, weighted by the likelihood of observed target data
# on the state of the ecosystem given each combination of parameter values.
```

```
if(require(StrathE2E2examples)){
   e2e_plot_sens_mc(model, selection="MC", use.example=TRUE)   # for Monte Carlo results

# To direct the graph output to a file rather than the screen, wrap the
# plot_Monte_Carlo_parameter_distributions() function call in a graphical device call:
# or jpeg("plot.jpg"), png("plot.png")
    pdf(file.path(tempdir(), "plot.pdf"),width=9,height=4)
    e2e_plot_sens_mc(model, selection="SENS", use.example=TRUE)
    dev.off()
}

#-------------------------------------------------------------------------

# For a comprehensive workflow through the generation of sensitivity and
# Monte Carlo analysis data see:
    help(e2e_run_sens)
    help(e2e_run_mc)
```

`e2e_plot_trophic` *Plot showing the annual mean trophic level index, and the omnivory index of each guild during the final year of a model run, optionally with credible intervals.*

Description

Generate a two-panel plot showing: (upper panel) the mean trophic level of each guild in the ecology model, and (lower panel) the omnivory index of each guild. The data are generated by the NetIndices package from a flow matrix of nutrient fluxes through, into and out of the ecosystem during the final year of a run. The data are generated automatically as part of the output from every call of the e2e_run() function. The default is to plot data from a single model run but if available, credible intervals of model output from a Monte Carlo analysis can be plotted instead.

Usage

```
e2e_plot_trophic(
  model,
  ci.data = FALSE,
  use.saved = FALSE,
  use.example = FALSE,
  results = NULL
)
```

Arguments

`model`	R-list object defining the baseline model configuration used to generate the data and compiled by the e2e_read() function.
`ci.data`	Logical. If TRUE plot credible intervals around model results based on Monte Carlo simulation with the e2e_run_mc() function (default=FALSE).
`use.saved`	Logical. If TRUE use data from a prior user-defined run held as csv files data in the current results folder as set by an e2e_read() function call (default=FALSE).

use.example	Logical. If TRUE use pre-computed example data from the internal North Sea model rather than user-generated data (default=FALSE).
results	R-list object of model output generated by the e2e_run() function. Only needed if ci.data=FALSE, use.saved=FALSE and use.example=FALSE. (Default=NULL).

Details

If credible intervals are plotted these are displayed as box-and-whiskers. The box spans 50 by a black tick mark and the maximum likelihhod model by a red tick mark.

Arguments determine the source of model data to be plotted. These can be outputs from a single model run with data held in memory as a list object or in a saved csv file, or from a Monte Carlo simulation (using the function e2e_run_mc()) to estimate credible intervals of model outputs. Generation of credible interval data is a long computing task, so example data for the North Sea model provided with the package are available as illustration.

Value

Graphical display in a new graphics window.

See Also

e2e_read, e2e_run, e2e_run_mc, e2e_plot_migration, e2e_plot_catch, e2e_plot_eco, e2e_plot_biomass

Examples

```
# Load the 1970-1999 version of the North Sea model supplied with the package,
# run, and generate a plot:
   model <- e2e_read("North_Sea", "1970-1999")
   results <- e2e_run(model,nyears=1,csv.output=FALSE)
   e2e_plot_trophic(model, results=results)

# Direct the graphics output to a pdf file ...
# or jpeg("plot.jpg"), png("plot.png")
   pdf(file.path(tempdir(), "plot.pdf"),width=6,height=8)
   e2e_plot_trophic(model, results=results)
   dev.off()

# Alternatively, plot the same data from a csv file saved in a temporary
# folder by the e2e_run() function:
   model <- e2e_read("North_Sea", "1970-1999")
   results <- e2e_run(model, nyears=1, csv.output=TRUE)
   e2e_plot_trophic(model, use.saved=TRUE)

# For the same model, plot the example data with credible intervals:
# This example requires the Strathe2E2examples supplementary data package.
if(require(StrathE2E2examples)){
   e2e_plot_trophic(model, ci.data=TRUE, use.example=TRUE)
}
```

`e2e_plot_ts` *Plot ecology model state variables or fishery catches for the full duration of a model run.*

Description

Multi-panel time-series plots of either ecology state variable data, or fishery landings and discards, for the full duration of a model run generated by the e2e_run() function.

Usage

```
e2e_plot_ts(model, results, selection = "ECO")
```

Arguments

`model`	R-list object defining the model configuration compiled by the e2e_read() function.
`results`	R-list object containing model results generated by the e2e_run() function.
`selection`	Text string from a list identifying whether ecology state variable or fishery catches are to be plotted. Select from: "ECO", "CATCH", default = "ECO". Remember to include the phrase within "" quotes.

Details

The function plots a multi-panel page of time series plots of either a) daily ecology state variable values aggregated over the inshore and offshore zones of the domain and over sediment classes, or b) annual landings and discards by zone for each model guild, for the full duration of a model run. Currently the masses of macrophytes, corpses and discards are not included in the state varibale plots due to space constraints.

Be warned that if the run is more than about 10 years then the plot becomes extremely compressed and messy. It is not intended to be of publication quality. The intention is to provide a quick-look diagnostic of trends in the state variables. This is useful to assess whether the model is close to stationary state or not.

Units of the plotted state varaiables mMN i.e. mass in the model domain without any scaling to zone-area or layer thickness. Similarly, units of catches are mMN/y from the whole model domain without any scaling to zone-area. The assumed area of the whole model domain is 1 m2.

Selection of either ecology state variable or catches to plot is by a function argument.

A separate set of functions is provided for plotting more detailed visualizations of just the final year from a model run, e.g. e2e_plot_eco(), e2e_plot_catch().

Value

Graphical display in a new graphics window.

See Also

`e2e_read`, `e2e_run`, `e2e_plot_eco`, `e2e_plot_catch`

Examples

```
# Load the 2003-2013 version of the North Sea model supplied with the package:
    model <- e2e_read("North_Sea", "2003-2013")
#Run the model and generate the results object, with csv output suppressed
    results <- e2e_run(model,nyears=2, csv.output=FALSE)
# Plot the time series of ecology state variable outputs
    e2e_plot_ts(model, results, selection="ECO")
# Time series plot of catches in a new window leaving the existing window open
    dev.new()
    e2e_plot_ts(model, results, selection="CATCH")

#Direct the graphics output to a file (Windows OS)...
# or jpeg("plot.jpg"), png("plot.png")
    pdf(file.path(tempdir(), "plot.pdf"),width=8,height=6)
    e2e_plot_ts(model, results, selection="ECO")
    dev.off()


# Demonstrate transient behaviour in the time-series outputs (run for 10 years).
# Read the North Sea/1970-1999 model and set the identifier for output files to "baseline",
# run the model and plot the full length output
    model <- e2e_read("North_Sea", "1970-1999",model.ident="baseline")
    results <- e2e_run(model,nyears=10, csv.output=FALSE)
    e2e_plot_ts(model, results, selection="ECO")
    dev.new()
    e2e_plot_ts(model, results, selection="CATCH")
# Create a new scenario version of the North Sea/1970-1999 model by increasing the activity
# rate of gear 1 (pelagic trawls and seines) by a factor of 3
    scenario_model <- model
    scenario_model$data$fleet.model$gear_mult[1] <- 3
   scenario_model$setup$model.ident <- "gear1x3"   # Set a new identifier for the outputs
    scenario_results <- e2e_run(scenario_model,nyears=10, csv.output=FALSE)
    dev.new()
    e2e_plot_ts(scenario_model, scenario_results, selection="ECO")
    dev.new()
    e2e_plot_ts(scenario_model, scenario_results, selection="CATCH")
```

`e2e_plot_ycurve` *Plot fishery yield curve data for planktivorous or demersal fish.*

Description

Plot planktivorous or demersal fish yield curve data generated by the function e2e_run_ycurve().

Usage

```
e2e_plot_ycurve(
  model,
  selection = "",
  use.saved = FALSE,
  use.example = FALSE,
  results = NULL,
  title = ""
)
```

Arguments

model	R-list object defining the baseline model configuration used to generate the data and compiled by the e2e_read() function.
selection	Text string from a list identifying the fish guild for which a yield curve is to be generated. Select from: "PLANKTIV", "DEMERSAL". Remember to include the phrase within "" quotes.
use.saved	Logical. If TRUE use data from a prior user-defined run held as csv files data in the current results folder as set by an e2e_read() function call (default=FALSE).
use.example	Logical. If TRUE use pre-computed example data from the internal North Sea model rather than user-generated data (default=FALSE).
results	Dataframe generated by the function e2e_run_ycurve(). Only needed if use.saved2=FALSE and use.example2=FALSE. (Default=NULL).
title	Optional free text (enclosed in "") to be added as a header for the plot (default = "").

Details

In the function e2e_run_ycurve(), the baseline for the sequence of runs (harvest ratio multiplier = 1.0) is a model name and variant as loaded by the e2e_read() function. The function then generates a set of biomass, landings and discards data for multiples of the target fish (planktivorous or demersal) harvest ratios relative to this baseline. This is done for a given value of the alternative (demersal or planktivorous) harvest ratio (also a multiple of the the baseline).

This function plots two graphs - the annual average fish biomass in the whole model domain (mMN/m2) as a function of harvest ratio multiplier, and the yield curve, ie the annual catch (and discards) (mMN/m2/y) as functions of the multiplier.

The yield curve represents the catch that would be generated from the stationary state of the model attained with long-term repeating annual cycles of all driving data including fishing.

Arguments for this function permit the input data to be drawn from an existing data object generated by the function e2e_run_ycurve(), a previously generated csv file, or example data provided with the package for versions of the internal North Sea models.

Value

Dataframe of results from which the plot is created, graphical display in a new graphics window

See Also

e2e_read, e2e_run, e2e_run_ycurve

Examples

```
# Load the 1970-1999 version of the North Sea model supplied with the package and
# generate a yield data object:
# WARNING - this example will take several minutes to run even though the model is
# only run for 3 years per step in harvest ratio. A realistic run would require
# at least 50 years per step and take much longer.
    model <- e2e_read("North_Sea", "1970-1999")
    pfhr=c(0,0.5,0.75,1.0,1.25,1.5,2.0,2.5,3.0)
    pf_yield_data <- e2e_run_ycurve(model,selection="PLANKTIV",nyears=3, HRvector=pfhr,
                                    HRfixed=1,csv.output=FALSE)
```

```
    data <- e2e_plot_ycurve(model,selection="PLANKTIV", results=pf_yield_data,
                            title="Planktivorous yield with baseline demersal fishing")

# Users can then plot other biomass, landings and discards data in the results
# object by, for example:
    par(mfrow=c(2,1))
    par(mar=c(3.2,5,2,0.8))
    ym<-1.1*max(pf_yield_data$Cetaceanbiom)
    plot(pf_yield_data$PlankFishHRmult, pf_yield_data$Cetaceanbiom, ylim=c(0,ym),
         type="l",lwd=3,yaxt="n",xaxt="n",ann=FALSE)
       abline(v=1,lty="dashed")
       axis(side=1,las=1,cex.axis=0.9)
       axis(side=2,las=1,cex.axis=0.9)
       mtext("Planktiv. fish harvest ratio multiplier",cex=1,side=1,line=2)
       mtext("Cetacean biomass",cex=1,side=2,line=3.5)
       mtext(bquote("mMN.m"^-2),cex=0.7,side=3,line=-0.05,adj=-0.18)
    ym<-1.1*max(pf_yield_data$Cetaceandisc)
    plot(pf_yield_data$PlankFishHRmult, pf_yield_data$Cetaceandisc, ylim=c(0,ym),
         type="l",lwd=3,yaxt="n",xaxt="n",ann=FALSE)
       abline(v=1,lty="dashed")
       axis(side=1,las=1,cex.axis=0.9)
       axis(side=2,las=1,cex.axis=0.9)
       mtext("Planktiv. fish harvest ratio multiplier",cex=1,side=1,line=2)
       mtext("Cetacean by-catch",cex=1,side=2,line=3.5)
       mtext(bquote("mMN.m"^-2 ~ ".y"^-1),cex=0.7,side=3,line=-0.05,adj=-0.18)


# Using example data generated with selection="PLANKTIV" ...
# Plot example data for one of the North Sea model versions internal to the package
# This example requires the Strathe2E2examples supplementary data package.
if(require(StrathE2E2examples)){
     model <- e2e_read("North_Sea", "1970-1999")
     pf_yield_data<-e2e_plot_ycurve(model, selection="PLANKTIV", use.example=TRUE)
}

# Using example data generated with selection="DEMERSAL"...
# This example requires the Strathe2E2examples supplementary data package.
if(require(StrathE2E2examples)){
    model <- e2e_read("North_Sea", "1970-1999")
    df_yield_data<-e2e_plot_ycurve(model, selection="DEMERSAL", use.example=TRUE)
}
```

`e2e_process_sens_mc` *Post-process already-created data from sensitivity or Monte Carlo analysis runs.*

Description

The functions e2e_run_sens() and e2e_run_mc() are extremely time consuming so it makes sense to share the load across multiple processors in parallel and combine the results afterwards using the function e2e_merge_sens_mc(). Both the original run functions and the merging function have post-processing options in their workflow. Nevertheless, there may be occasions where it is necessary to simply post-process an already created raw data set. Hence the need for this function.

Usage

```
e2e_process_sens_mc(
  model,
  selection = "",
  use.example = FALSE,
  csv.output = FALSE
)
```

Arguments

model	R-list object defining the model configuration and compiled by the e2e_read() function
selection	Text string from a list identifying source of data to bed merged. Select from: "SENS", "MC", referring to sensitivity analysis or Monte Carlo analysis. Remember to include the phrase within "" quotes.
use.example	Logical. If TRUE use pre-computed example data from the internal North Sea model rather than user-generated data (default=FALSE).
csv.output	Logical. If TRUE then enable writing of csv output files (default=FALSE).

Details

The model.ident identifier and folder path for input files must be set in a e2e_read() function call. If enabled, output files will be directed to the same folder.

Details relating to sensitivity analysis processing:

The function reads a file of raw sensitivity analysis data (OAT_results-*.csv, where * refers to the identifier model.ident set in the e2e_read() function) from a /results folder and performs the post-processing that would ordinarily be done automatically within the e2e_run_sens() or e2e_merge_sens_mc() functions. The output of the processing is a dataframe (and optionaly a csv file) containing the mean Elementary Effect (EE_mean) and the standard deviation of EE (EE_sd) for each parameter, sorted by the absolute value of EE_mean. EE_mean is an index of the magnitude of the sensitivity of a parameter, and EE_sd is an index of the extent of interaction with other parameters.

csv output from the function (if the argument csv.output=TRUE) is a file named sorted_parameter_elementary_effects-*.csv in the results folder specified in an e2e_read() function call.

Optionally, the function can read example raw sensitivity data for one of the two North Sea model variants supplied with the package.

For details of how the Elementary Effect values are derived for each parameter see help(e2e_run_sens)

Details relating to Monte Carlo analysis processing:

The function e2e_run_mc() generates 11 different output files covering the range of data generated by StrathE2E. The post processing reads all of these and generates quantiles of the likelihood weighted distributions of all of the outputs and saves these as credible interval files.

If the argument csv.output=TRUE then the function writes output to csv files. The function always returns all of the processed data as a list object. The function takes a few minutes to run as it has a lot of work to do, especially in processing all of the daily output variables.

Although the e2e_run_mc() function generates large raw data files (11 files total 3.2 Mb per iteration), the processed data are much smaller (13 files total ~4 Mb regardless of the number of iterations.

Value

Dataframe (sensitivity analayis) or list object (Monte Carlo analysis) of the processed data. If csv.output=TRUE then CSV files of the processed data.

See Also

e2e_read, e2e_run_sens, e2e_run_mc, e2e_merge_sens_mc , e2e_plot_sens_mc

Examples

```
# The examples provided here are illustration of how to process data from sensitivity
# and Monte Carlo analyses. They are commented-out because they cannot be run in isolation;
# they need to have existing data objects or files on which to operate.

# ------------------------------------------------------------------------

# Sensitivity analysis processing:
# This example requires the Strathe2E2examples supplementary data package.
# Load details of the 1970-1999 version of the North Sea model supplied with the package:
    model <- e2e_read("North_Sea", "1970-1999")
# Process the example data for this model variant provided with the package
if(require(StrathE2E2examples)){
    sens_results <- e2e_process_sens_mc(model, selection="SENS", use.example=TRUE)
# View the first few rows of the results dataframe
    head(sens_results)
}

# ------------------------------------------------------------------------

# Monte Carlo analysis processing:
# This dummy example here assumes that raw results data have been previously
# generated by a of the e2e_run_mc() function, with the model.ident value
# assigned as "testdata", or that data from parallel runs have been gathered
# together in the same folder with model.ident="testdata".
# Load the model name and version for the data to be processed :
# (REPLACE "Folder/results" your own results.path before running)
# e.g. ... model <- e2e_read("North_Sea", "2003-2013", results.path="Folder/results",
#                            model.ident="testdata")
#          mc_results <- e2e_process_sens_mc(model, selection="MC", csv.output=TRUE)
#          str(mc_results,max.level=1)

# ------------------------------------------------------------------------
```

---

e2e_read *Read all the configuration, driving and parameter files for a designated model and compile the data into a list object.*

Description

This function is the key point of entry to the package. It reads all the csv files containing the configuration, driving and parameter values which define a model for a particular region and compiles them into a single R-list object. This data object forms the basis for almost all other functions in the package.

Usage

```
e2e_read(
  model.name,
  model.variant,
  models.path = NULL,
  results.path = NULL,
  results.subdir = "",
  model.ident = "base",
  quiet = TRUE,
  silent = FALSE
)
```

Arguments

model.name	Name of model to read (no default).
model.variant	Variant of the model to be read (no default).
models.path	Relative path from the current working directory to a folder containing a library of model configurations (typically "Folder/Models"). Setting models.path="" is valid. Default models.path=NULL, meaning read a North Sea model setup from the package folder extdata/Models.
results.path	Relative path from the current working directory to a folder for writing and reading model output files (e.g. "E2E_results"). Setting results.path="" is valid. Model-specific sub-folders will be assigned and if necessary auto-created according to the model name and variant. Default results.path=NULL, meaning write/read to/from a temporary directory.
results.subdir	Subdirectory of "working_directory/results.path/model_name/model_variant" to be created if required. (Default="", meaning no subdirectory will be created).
model.ident	Identifier text appended to output file names (e.g. OFFSHORE_model_annualresults-TEXT.csv instead of just OFFSHORE_model_annualresults.csv). (Default="base").
quiet	Logical. If FALSE then see on-screen information on indvidual parameter files as they are read (default=TRUE).
silent	Logical. If FALSE then see on-screen information on model and results paths (default=FALSE). If set TRUE then this over-rides any quiet=FALSE setting and forces quiet=TRUE.

Details

The input csv files are specified in the MODEL_SETUP.csv file located in the Model/variant folder specified by the argument models.path in a e2e_read() call, or from the extdata/Models folder in the package installation. The models supplied with the package are two variants (1970-1999 and 2003-2013) of a fully parameterised and documented model of the North Sea.

Starting from a baseline model configuration defined in the MODEL_SETUP.csv file, any of the terms in the R-list objcet created by e2e_read() can be modified by coding to produce an unlimited range of scenario configurations representing, for example, changes in the physical environment, changes in the composition or activity of the fishing fleets, or any combination of these. The power of this approach is that large numbers of scenarios can be coded in standard R and simulated without any need for manual editing of input files.

Value

R-list object containing all of the model configuration data.

See Also

e2e_ls, e2e_copy, e2e_run

Examples

```
# Load the 1970-1999 version of the North Sea model supplied with the package; outputs go
# to a temporary results folder; default model.ident setting.
# Default screen information settings - on-screen parameter file information suppressed but
# path information enabled:
    model <- e2e_read("North_Sea", "1970-1999")

# User-specified model.ident; on-screen parameter file and path information enabaled:
    model <- e2e_read("North_Sea", "1970-1999",model.ident="baseline",quiet=FALSE)

# All on-screen information suppressed even though quiet=FALSE:
   model <- e2e_read("North_Sea", "1970-1999",model.ident="baseline",quiet=FALSE,silent=TRUE)

# View details of the structure and contents of the model list-object:
    str(model,max.level=1)
    str(model,max.level=2)

# Dummy example to illustrate loading a user defined model from a user workspace
# ... substitute your own path details for e.g. "Folder/Models":
#     model<-e2e_read("Modelname", "Variantname",
#                     models.path="Folder/Models",
#                     model.ident="demo")

# Dummy example to illustrate loading a user defined model from a user workspace and
# sending the model outputs to a specified folder ... substitute your own path details
# for the dummy paths shown here
# for "Folder/Models" and "Folder/results":
#     model<-e2e_read("Modelname", "Variantname",
#                     models.path="Folder/Models",
#                     results.path="Folder/results",
#                     model.ident="demo")

# Create a new scenario version of the North Sea/1970-1999 model by increasing the
# activity rate of gear 1 (pelagic trawls and seines) by a factor of 2:
    model <- e2e_read("North_Sea", "1970-1999")
    scenario_model <- model
    scenario_model$data$fleet.model$gear_mult[1] <- 2
# Set a new identifier for any csv outputs:
    scenario_model$setup$model.ident <- "gear1x2"

# Run the baseline model for 2 years:
    results<-e2e_run(model,nyears=2)
# Visualise the time series of output from the baseline model run:
    e2e_plot_ts(model,results,selection="ECO")


# Run the scenario model for 20 years:
    scenario_results<-e2e_run(scenario_model,nyears=20)
# Visualise the time series of output from the scenario model run
# in a new graphics window:
    dev.new()
    e2e_plot_ts(scenario_model,scenario_results,selection="ECO")
```

`e2e_run` *Run the StrathE2E model with a given configuration.*

Description

Perform a single deterministic run of the StrathE2E model for a configuration defined by an R-list object compiled by a prior call of the e2e_read() function.

Usage

```
e2e_run(model, nyears = 20, quiet = TRUE, csv.output = FALSE)
```

Arguments

`model`	R-list object defining the model configuration and compiled by the e2e_read() function.
`nyears`	Number of years (integer) to run the model (default=20).
`quiet`	Logical. If FALSE then see status outputs during the model run (default=TRUE).
`csv.output`	Logical. If TRUE then enable writing of csv output files (default=FALSE).

Details

The function solves a network of Ordinary Differential Equations using the lsoda function in the R deSolve package. The equations represent a shelf-sea food web and its connections to the physical and chemical environment, and to a set of fishing fleets.

The outputs from the run are time series (daily intervals) of the masses of each of the state variables, the fluxes between all state variable, and the fluxes in and out of the model including fishery landings. In addition, a range of derived quantities are generated for the final year of the run, including annual averages, maxima and minima of state variables, annual fluxes between state variables, annual landings and discards of each guild of taxa by each fleet of fishing gears, and a set of network indices generated by the R NetIndices package.

Value

Model outputs as an R-list object and optionally csv files.

See Also

`e2e_ls`, `e2e_read`, `e2e_copy`, `e2e_plot_ts`

Examples

```
# Load the 2003-2013 version of the North Sea model supplied with the package:
    model <- e2e_read("North_Sea", "2003-2013")

# Run the model for 2 years and generate the results object
    results <- e2e_run(model,nyears=2)
# Set csv.output=TRUE to save results to csv files to the directory
# specified in the e2e_read() function call.

# Time series plot of state varaiables over the full length of the run
    e2e_plot_ts(model, results,selection="ECO")
    dev.new()
    e2e_plot_ts(model, results,selection="CATCH")
```

---

e2e_run_mc — *Run a Monte Carlo simulation with StrathE2E and derive centiles of credible values of model outputs.*

Description

The Monte Carlo scheme provided with the StrathE2E2 package has two modes of operation. The first is referred to as 'baseline mode'. This involves generating a list of ecology model parameter sets and, for each set, determining the likelihood of observational data consistent with the environmental and fishery drivers. The model outputs generated with each set are then weighted by the likelihood before deriving quantiles of their distribution. The quantile ranges then represent credible intervals of the model outputs.

Usage

```
e2e_run_mc(
  model,
  nyears = 50,
  baseline.mode = TRUE,
  use.example.baseparms = FALSE,
  baseparms.ident = "",
  begin.sample = 1,
  n_iter = 1000,
  csv.output = FALSE,
  runtime.plot = TRUE,
  postprocess = TRUE
)
```

Arguments

model	R-list object defining the model configuration (baseline or scenario), compiled by the e2e_read() function.
nyears	Number of years to run each instance of the model. Needs to be long enough to allow the model to attain a stationary state (default=50).
baseline.mode	Logical. If TRUE then the simulation adopts a baseline mode. If FALSE then scenario mode (default=TRUE).

`use.example.baseparms`
: Logical. Value required only if baseline.mode=FALSE. If TRUE then the baseline parameters set is drawn from example data provided with the package. If FALSE then a user generated baseline parameter set is expected.

`baseparms.ident`
: Default = "". Value required if baseline.run=FALSE and use.example.baseparms=FALSE, in which case this is the model.ident string of the a user generated baseline parameter set. Remember to include the phrase within "" quotes.

`begin.sample`
: Default = 1. Value required if baseline.mode=FALSE. Value sets the first row in the file of baseline parameter data from which the sequence of parameters sets to be used in this run will be taken. Set a value > 1 if this run is part of a parallel set of runs.

`n_iter`
: Number of iterations of the model (default=1000). If baseline.mode=FALSE and the number of sets in the supplied baseline parameter file beyond the value of begin.sample is less than n_iter, then n_iter is reset to the remaining number available.

`csv.output`
: Logical. If TRUE then enable writing of csv output files (default=FALSE).

`runtime.plot`
: Logical. If FALSE then disable runtime plotting of the progress of the run - useful for testing (default=TRUE).

`postprocess`
: Logical. If FALSE then disable post-processing of the cumulative outputs, for example in parallel mode where multiple runs are going to be combined later (default=TRUE).

Details

The second mode of operation is referred to as the 'scenario mode'. In this case, the parameter sets and their associated likelihoods from a baseline mode simulation, are used to generate model outputs for scenarios of environmental or fishing drivers – e.g. increased activity by selected gears, or warmer sea temperatures. These scenario outputs are then weighted by the baseline mode likelihoods before derivation of quantiles and credible intervals.

In baseline mode, the Monte Carlo scheme generates an ensemble of model runs by sampling the ecology model parameters from uniform distibutions centred on an initial (baseline) parameter set. Ideally, this should be the maximum likelihood parameter set arrived at by prior application the various optimization functions in the package to 'fit' the model to a suite of observational data on the state of ecosystem (target data for fitting are in /Modelname/Variantname/Target/annual_observed_*.csv). Each iteration in the Monte Carlo scheme then generates a unique time series of model outputs at daily intervals, together with the overall likelihood of the obervational target data. From these data, the function then derives credible intervals of each output by weighting the values from each parameter set by the associated likelihood.

Running a scenario mode simulation requires a baseline model to have been previously completed, in order to generate a list of parameter sets and associated likelihoods. In scenario mode, model driving data (e.g. temperatures or fishing activities) are varied from the baseline configuration by editing the model object generated by a e2e_read() function call, and the model is then run with the existing baseline model parameter sets, and the results weighted by the baseline mode likelihoods in order to derive the credible intervals.

The choice of baseline vs scenario mode of simulations is determined in the function by a logical argument setting 'baseline.mode'.

In baseline mode, the coefficients of variation for jiggling the ecology parameter can be varied in real-time during the run by editing the file "monte_carlo.csv" in the folder /Param/control/ of the model version. However, it is recommended to leave the setting constant for the duration of a run.

A CV of 0.10 to 0.15 is recommended. If comparing the credible intervals for two different baseline models then it is important to use the same CV in both cases.

In scenario model, the parameter sets are pre-ordained by a prior baseline simulation, so the CV setting is ineffective.

On completion of all the iterations of the model, whether in baseline or scenarion mode, for each model output variable in turn, values from the individual runs and their associated model likelihoods are assembled as a list of paired values. The list is then sorted by ascending values of the model variable, and the cumulative likelihoods with increasing value of model variable is calculated. Values for the model variable at standard proportions (0.005, 0.25, 0.5, 0.75, 0.995) of the maximum cumulative likelhood are then extracted by interpolation. These represent the credible intervals of model output given uncertainty in the ecology model parameters.

Outputs from the e2e_run_mc() function depend on whether post-processing is selected. If not, then raw data are saved as csv files (provided that the argument csv.output=TRUE), and the function also returns a dataframe of the parameter sets used in the run and their corresponding likelihoods. In scenario mode this is just a replica of the baseline mode data that have been used to run the simulation. If post-processing is selected, then both the raw and processed credible interval data are saved to csv files (provided that the argument csv.output=TRUE), and the function returns a list object which includes the parameter dataframe plus all the processed credible interval data structures. csv files are saved in the folder /results/Modelname/Variantname/CredInt/ with an identifier for the simulation (model.ident) created by the e2e_read() function. There are two types of output. First is simply an accumulation of all the standard outputs from StrathE2E on a run-by-run basis. The second is a set of files containing the derived credible intervals.

The function displays a real-time graphic to show the progress of the simulation. During the StrathE2E-running phase of the process an x-y graph is updated after each iteration (x-axis=iterations, y-axis=Likelihood of the target data), with a horizontal grey line showing the baseline (maximum liklihood) result, and black symbols showing the likelihood for each iteration based on the parameter values sampled from the baseline. The y-axis limits for the graph can be varied in real-time during the run by editing the file "monte_carlo.csv" in the folder /Param/control/ of the model version.

During the post-processing phase, a message is displayed as each output variable is processed.

WARNING - the scheme can take a long time to run (~2 days with the default settings), and generate large output files (11 files total 3.2 Mb per iteration). Check the memory capacity of you machine before starting a long run. It can make sense to parallelize the process by splitting across different processors and merging the results afterwards. The package contains a function for merging cumulative output files from multiple runs and post-processing the combined data.

In baseline mode, parallel instances of the function can be implemented on different processors, all starting from the same initial parameter set. During the subsequent merging process, the outputs from the first parameter set, of all but the first instance, are stripped away and discarded.

Impementing parallel runs in scenario mode requires a different approach. Here, the parameter sets are pre-ordained and it is important to avoid using duplicate sets in the simulations. Hence, the function includes an argument 'begin.sample' to select a pointer in the baseline parameter set to begin sampling in any scenario mode run. For example, in the first instance, begin.sample=1, and the number of iterations (n_iter) might be set to e.g. 200. For the second instance the pointer (begin.sample) would then be set to 201, and so on. If begin.sample > 1, the function runs the first baseline parameter set (sample 1) first before proceeding to the pointer begin.sample and completing the requested number of iterations, so there will be one extra iteration in the output files. If the requested number of iterations means that sampling would over the end of available number of parameter sets then the number of iterations is reduced accordingly.

Although the e2e_run_mc() function generates large raw data files (11 files total 3.2 Mb per iteration), the processed data are much smaller (13 files total ~4 Mb regardless of the nunber of iterations).

Value

Depends on argument settings. List object containing processed data if postprocess=TRUE, otherwise a NULL object; csv files of raw and if processed data if csv.output=TRUE; real-time graphical displays during the simulation if runtime.plot=TRUE

See Also

e2e_read, e2e_merge_sens_mc, e2e_process_sens_mc, e2e_plot_sens_mc

Examples

```
# The examples provided here are illustration of how to set up and run Monte Carlo
# analyses. Even though they are stripped-down minimalist examples they each still
# take 10-15 minutes to run.

# ------------------------------------------------------------------------


# A quick demonstration of baseline mode with postprocessing, but csv output disabled.
# Load the 1970-1999 version of the North Sea model supplied with the package
# Run the Monte Carlo process and generate some data but disable csv output.
#    model <- e2e_read("North_Sea", "1970-1999")
#    demo <- e2e_run_mc(model,nyears=2,baseline.mode=TRUE,n_iter=5,csv.output=FALSE)
# View the structure of the returned list:
#    str(demo,max.level=1)
# View the structure of a returned list element containing sub-sets:
#    str(demo$CI_annual_avmass,max.level=1)
# View the structure of a returned list element containing sub-sets:
#    str(demo$CI_annual_fluxes,max.level=1)
# View the top few rows on the whole domain data on annual averge mass:
#    head(demo$CI_annual_avmass$whole)


# ------------------------------------------------------------------------

# Dummy example to illustrate a more meaningful run. Output directed to
# "../Folder/results" relative to the current working directory (REPLACE with your
# own results.path before running):
#    model <- e2e_read("North_Sea", "1970-1999",results.path="Folder/results")
#   mc_results <- e2e_run_mc(model,baseline.mode=TRUE,nyears=50,n_iter=1000,csv,output=TRUE)
# WARNING: This run will take about 48h to complete, much better to split up and spread
# across multiple processors !

# ------------------------------------------------------------------------


# A quick demonstration run in baseline mode, save the data to a temporary folder:
    basemodel <- e2e_read("North_Sea", "1970-1999",model.ident="mcbaseline")
    basedemo <- e2e_run_mc(basemodel,nyears=2,baseline.mode=TRUE,
                           n_iter=5,csv.output=TRUE)


# ------------------------------------------------------------------------

# Then a quick demonstration run in scenario mode using the saved baseline parameter data,
# from the previous example, and save to csv. It is assumed that the baseline parameter
```

```
# data are in the temporary folder in which they were created, and that the same temporary
# folder is used for this example.
# First create an extreme fishing scenario - quadruple some gear activities, run for 10 years
    scenariomodel<-basemodel
    scenariomodel$setup$model.ident <- "mcscenario"
# Gear 1 (Pelagic trawls) activity rate rescaled to 4*baseline:
    scenariomodel$data$fleet.model$gear_mult[1] <- 4
# Gear 4 (Beam_Trawl_BT1+BT2) activity rate rescaled to 4*baseline:
    scenariomodel$data$fleet.model$gear_mult[4] <- 4
    scendemo <- e2e_run_mc(scenariomodel,nyears=2,baseline.mode=FALSE,
                           use.example.baseparms=FALSE, baseparms.ident="mcbaseline",
                           begin.sample=1, n_iter=5,csv.output=TRUE)

# Compare the results of the baseline and scenario:
    basemodel <- e2e_read("North_Sea", "1970-1999", model.ident="mcbaseline")
    scenariomodel <- e2e_read("North_Sea","1970-1999",model.ident="mcscenario")
   e2e_compare_runs_box(selection="ANNUAL", model1=basemodel, ci.data1=TRUE, use.saved1=TRUE,
                        model2=scenariomodel, ci.data2=TRUE, use.saved2=TRUE )
    dev.new()
   e2e_compare_runs_box(selection="MONTHLY", model1=basemodel, ci.data1=TRUE, use.saved1=TRUE,
                        model2=scenariomodel, ci.data2=TRUE, use.saved2=TRUE )


# ------------------------------------------------------------------------


# Quick demonstration of parallelizing the process in baseline mode, with output folder
# to a temporary folder. To explore further details of results.path="YourFolder"
# relative to the current working directory.
# Launch batch 1 (on processor 1):
    model1 <- e2e_read("North_Sea", "1970-1999",  model.ident="BATCH1")
    results1 <- e2e_run_mc(model1,nyears=2,baseline.mode=TRUE,
                           n_iter=5,csv.output=TRUE,postprocess=FALSE)
# Launch batch 2 (on processor 2):
    model2 <- e2e_read("North_Sea", "1970-1999", model.ident="BATCH2")
    results2 <- e2e_run_mc(model2,nyears=2,baseline.mode=TRUE,
                           n_iter=6,csv.output=TRUE,postprocess=FALSE)
# Note that these two runs return only raw data since postprocess=FALSE

# Note that Batch 2 requests 6 iterations, rather that 5 in Batch 1.
# The number of iterations do not have to be the same in each batch.
# However, the first in each batch has to use the initial parameter set from the model setup,
# as this is the parent for all the subsequent samples generated during the run.
# When we come to merge the data from separate batches, data from the first iteration are
# stripped off and discarded for all but the first batch as we do not want to include duplicate
# data in the combined files. Hence we choose 6 iterations here in Batch 2 to make the point,
# and we expect the combined data to include 10 iterations.
#
# Then, afterwards, merge the two raw results files with text-tags BATCH1 and BATCH2,
# and post process the combined file:
    model3 <- e2e_read("North_Sea", "1970-1999", model.ident="COMBINED")
    processed_data <- e2e_merge_sens_mc(model3, selection="MC",
                     ident.list<-c("BATCH1","BATCH2"), postprocess=TRUE, csv.output=TRUE)
# or...
    combined_data <- e2e_merge_sens_mc(model3, selection="MC",
                    ident.list<-c("BATCH1","BATCH2"), postprocess=FALSE, csv.output=TRUE)
    processed_data<-e2e_process_sens_mc(model3,selection="MC",use.example=FALSE,
```

```
                                        csv.output=TRUE)

# Plot the parameter likelihood distributions from the combined data
    e2e_plot_sens_mc(model3, selection="MC")


# ------------------------------------------------------------------------


# Example of parallelizing the process in scenario mode, using the baseline parameter
# sets 'COMBINED' from above (assuming that this is sitting in the same temporary folder
# in which it was created in the example above.
# The activity of all fishing gears is reduced to zero to create a no-fishing scenario.
# Run each batch for 10 years as a relatively quick demo - a real run would need to
# run for at least 40 year
# Launch batch 1 (in processor 1):
    model1s <- e2e_read("North_Sea", "1970-1999", model.ident="BATCH1_S")
# Activity rates of all 12 gears rescaled to 0*baseline:
    model1s$data$fleet.model$gear_mult[1:12] <- 0
   results1s <- e2e_run_mc(model1s,nyears=2,baseline.mode=FALSE, baseparms.ident="COMBINED",
                 begin.sample=1, n_iter=5,csv.output=TRUE,postprocess=FALSE)
# Launch batch 2 (on processore 2):
    model2s <- e2e_read("North_Sea", "1970-1999", model.ident="BATCH2_S")
# Activity rates of all 12 gears rescaled to 0*baseline:
    model2s$data$fleet.model$gear_mult[1:12] <- 0
   results2s <- e2e_run_mc(model2s,nyears=2,baseline.mode=FALSE, baseparms.ident="COMBINED",
                 begin.sample=6, n_iter=5,csv.output=TRUE,postprocess=FALSE)
# Note that Batch 1 samples rows 1:5 of the baseline mode parameter set archive "COMBINED"
# (begin.sample=1, n_iter=5), so Batch 2 needs to start sampling at row 6 (begin.sample=6).
# The baseline archive contains 10 rows, so Batch 2 has the capacity to undertake up to 5
# sample iterations (rows 6:10). If we select more than 5 iterations (e.g. n_iter=8) then
# the code will automatically restrict to 5. Note that in fact, to be consistent with the
# format of output files from the baseline mode, each scenario mode run where
# 'begin.sample' > 1 will complete n_iter+1 iterations, by adding an initial run using the
# parameter values from row 1 of the baseline parameter set - which is then stripped off
# during merging.
#
# Then, merge the two raw results files with text-tags BATCH1_S and BATCH2_S, and post
# process the combined file:
    model3s <- e2e_read("North_Sea", "1970-1999", model.ident="COMBINED_S")
    processed_datas <- e2e_merge_sens_mc(model3s, selection="MC",
                ident.list<-c("BATCH1_S","BATCH2_S"), postprocess=TRUE, csv.output=TRUE)

# Finally plot comparisons of the baseline and scenario model runs:
   e2e_compare_runs_box(selection="ANNUAL", model1=model3, ci.data1=TRUE, use.saved1=TRUE,
                        model2=model3s, ci.data2=TRUE, use.saved2=TRUE )
    dev.new()
   e2e_compare_runs_box(selection="MONTHLY", model1=model3, ci.data1=TRUE, use.saved1=TRUE,
                        model2=model3s, ci.data2=TRUE, use.saved2=TRUE )


# ------------------------------------------------------------------------
```

---

e2e_run_sens *Perform a parameter sensitivity analysis on the StrathE2E model.*

Description

Performs a one-at-a-time parameter sensitivity analysis on the StrathE2E model using the Morris Method for factorial sampling of the physical configuration parameters, ecology model parameters, the fishing fleet model parameters, and the environmental forcings.

Usage

```
e2e_run_sens(
  model,
  nyears = 50,
  n_traj = 16,
  trajsd = 0.0075,
  n_setoflevels = 4,
  v_setoflevels = 0.1,
  coldstart = TRUE,
  quiet = TRUE,
  postprocess = TRUE,
  csv.output = FALSE,
  runtime.plot = TRUE,
  outID = 0
)
```

Arguments

model	R-list object generated by the e2e_read() function which defines the parent model configuration.
nyears	Number of years to run the model in each iteration (default=50).
n_traj	Number of trajectories of parameter sets (default=16).
trajsd	Coefficient of variation used to set the standard deviation for the gaussian distribution from which new paramater values are drawn to create each trajectoy baseline from the initial parent values (default=0.0075).
n_setoflevels	Number of fixed levels of coefficient of variation used to generate the individual parameter values in each level-run. Must be an even number (default=4).
v_setoflevels	Maximum coefficient of variation for the set of levels (default=0.1, i.e. -10 percent to +10 percent).
coldstart	Logical. If TRUE then the run is starting from cold - which means that the first trajectory baseline is the parent configuration as specified in the 'model' list object. If FALSE then signifies that this is a parallel run whuch will later be merged with the 'coldstart=TRUE' run. In this case the first trajectory baseline is a derivative of the parent. Default=TRUE.
quiet	Logical. If TRUE then suppress informational messages at the start of each iteration (default=TRUE).
postprocess	Logical. If TRUE then process the results through to a final sorted list of parameter sensitivies for plotting. If FALSE just produce the raw results. The reason for NOT processing would be if the job has been shared across multiple machines/processors and several raw result files need to be merged before processing. Default=TRUE.
csv.output	Logical. If TRUE then enable writing of csv output files (default=FALSE).
runtime.plot	Logical. If FALSE then disable runtime plotting of the progress of the run - useful for testing (default=TRUE).

outID — Numeric value in the range 0 to 247. Selects the output criterion to be used as the basis for the analysis. Default=0 corresponds to the likelihood of observed target data. Other values obtainable by running e2e_get_senscrit().

Details

The basis for the method is a scheme for sampling the model parameters, one at a time, from distributions around baseline sets, and testing the effects on the performance of the model against some criterion. The default criterion is the likelihood of the observational data on the state of the ecosystem that are used as the target for parameter optimization in the various simulated annealing functions supplied with the package. However, the criterion can in principle be any metric output from the model, e.g. a state variable value at some point in time or averaged over a time interval, or one of the computed fluxes.

The process requires an initial set of parameters for the model. We refer to this as the 'parent' parameter set. It is recommended that this should be the parameters producing the maximum likelihood of the observational target data (as estimated by e.g. the e2e_optimize_eco() function). The MODEL_SETUP.csv file in the folder /Models/Modelname/Modelvariant/ should be configured to point to the relevant files, and then loaded with the e2e_read() function.

From this parent set, a series of 'child' parameter sets are generated by applying a separate random increment to each parameter drawn from a gaussian distribution of mean 0 and standard deviation given by a fixed coefficient of variation applied to the parent-set value of each parameter.

For each of the child-sets of parameters, the chosen output criterion is saved following runs of StrathE2E to stationary state. We refer to these as trajectory baselines.

Then, for each trajectory, the parameters are varied in turn, one at a time, by adding a fixed proportionality increment to the trajectory baseline values, the model re-run, and the output criterion computed. We refer to these as 'level runs'. The proportionality increment is the same for all of the level runs within a given trajectory, and is drawn at random from a set of fixed levels distributed symetrically around 0 (e.g. -10, -5, +5, +10 percent, i.e. proportions of the trajectory baseline values = 0.9, 0.95, 1.05, 1.10).

For each level run, the 'Elementary Effect (EE)' of the given parameter is calculated from the difference between the level run criterion value and the corresponding trajectory baseline criterion value.

On completion of all the trajectories, the raw results are (optionally) post-processed to generate the mean and standard deviations of all the EE values for each parameter. EE_mean is an index of the magnitude of the sensitivity, and EE_sd is an index of the extent of interaction with other parameters.

During the run the function produces a real-time plot for each trajectory, in which the x-axis represents the sequence of parameters, and the y-axis is the likelihood of the target data. A horizontal red line indicates the likelihood of the parent parameter set, horizontal grey line indicates the likelihood for each trajectory baseline and each level-run likelihood is shown by a symbol. The y-axis range can be changed in real-time by editing the setup file "/Models/Modelname/Modelvariant/Param/control/sensitivity.csv"

The outputs from the function are saved as list objects and directed to csv files (provided that the argument csv.output=TRUE) in the "results" folder sepcified in an e2e_read() function call. The outputs are:

- Table of parameter values applied in each run of the model (OAT_parameter_values-*.csv, where * = model.ident as defined by e2e_read())
- Table of the criterion value and EE value for each trajectory/level run (OAT_results-*.csv)
- If post-processing is selected, then a table of Mean EE and standard deviation of EE for each parameter, sorted by the absolute value of EE_mean (sorted_parameter_elemental_effects-*.csv)

As mentioned above, the default criterion for assessing the model sensitivity is the likelihood of the observed target data set on the state of the ecosystem given each set of model drivers and parameters. However, a function argument allows other criteria to be chosen as the basis for the analysis from the list of annually averaged or integrated variables saved in the output objects:

- results$final.year.output$mass_results_wholedomain (whole-domain annual averages of stage variables over the final year of a model run), and
- results$final.year.output$annual_flux_results_wholedomain (whole-domian annual integrals of fluxes between state variables over the final year of a model run).

The criterion is chosen by setting a value for the argument outID. The default outID=0 selects the likelihood of the observed target data. Other values in the range 1 to 247 select annualy averaged mass or annually integrated flux outputs from a list viewable by running the function e2e_get_senscrit().

WARNING - The e2e_run_sens() function will take several days to run to completion on a single processor with even a modest number of iterations. The total number of model runs required to support the analysis is r*(n+1) where r is the number of trajectories and n is the number of parameters. The function incorporates all of the physical configuration parameters, fixed and fitted ecology model parameters, the fishing fleet model parameters, and the environmental forcings into the analysis, so n = 450. Each model run needs to be sufficiently long to achieve a stationary state and as a consequnce a typical runtime will be around 10h per trajectory. The mininimum recommended number of trajectories is 15, so the function can take several days to complete.

However, it is possible to spread the load over multiple processor/machines with arguments in the function allowing for management of this parallelization. Afterwards, the raw results files are combined into a single data set using the e2e_merge_sens_mc() function, and then processed using the function e2e_process_sens_mc().

A separate function e2e_plot_sens_mc() produces a graphical representation of the EE_mean and EE_sd results.

Value

Depends on the settings of arguments 'postprocess' and 'csv.ouptut': If postprocess=TRUE and csv.output=TRUE then outputs are csv files of raw parameter vectors, likelihoods and Elementary Effects for each run, and parameter list sorted by EE_mean, plus the function returns the data of sorted parameters. If postprocess=FALSE and csv.output=FALSE then the function simply returns a dataframe of likelihoods and Elementary Effects for each run.

References

For details on the sensitivity analysis method see: Morris, M.D. (1991). Factorial sampling plans for preliminary computational experiments. Technometrics, 33, 161-174.

For a review of sensitivity analysis methods including the Morris Method see: Wu, J. et al. (2013). Sensitivity analysis of infectious disease models: methods, advances and their application. J R Soc Interface 10: 20121018, 14pp.

See Also

e2e_get_senscrit, e2e_read, e2e_merge_sens_mc, e2e_process_sens_mc, e2e_plot_sens_mc

Examples

```
# The examples provided here are illustration of how to set up and run sensitivity
# analyses. Even though they are stripped-down minimalist examples, they each still
# take about 45 min to run.

# ------------------------------------------------------------------------

## Not run:
# Load the 2003-2013 version of the North Sea model supplied with the package:
    model <- e2e_read("North_Sea", "1970-1999")
#
# Run the sensitivity analysis process (a quick demonstration):
# WARNING - Running a full sensitivity analysis takes days of computer time on a single
# machine/processor because it involves a huge number of model runs.
# The example below is just a (relatively) quick minimalist demonstration and should NOT
# be taken as the basis for any analysis or conclusions.
# Even so, this minimalist demonstration run could take 45 min to complete because it
# involves 1353 model runs.
# This examples uses the likelihood of observed target data as the criterion for assessing
# model sensitivity (i.e. outID is not set and so assumes the default value of 0).
    sens_results <- e2e_run_sens(model, nyears=1, n_traj=3, csv.output=FALSE)
# View the top few rows of the results dataframe:
    head(sens_results)

## End(Not run)

# ------------------------------------------------------------------------

# To view the list of available criteria for use as the basis for the sensitivity analysis:
    e2e_get_senscrit()

# ------------------------------------------------------------------------

## Not run:
# This examples uses the annual average mass of planktivorous fish as the criterion for
assessing model sensitivity (i.e. outID = 24).
    sens_results <- e2e_run_sens(model, nyears=1, n_traj=3, csv.output=FALSE, outID=24)
# View the top few rows of the results dataframe:
    head(sens_results)

## End(Not run)

# ------------------------------------------------------------------------

# This is a dummy example to illustrate a more realistic sensitivity analysis:
#   sens_results <- e2e_run_sens(model, nyears=50, n_traj=16, postprocess=TRUE,csv.output=TRUE)
# DO NOT launch this configuration unless you are prepared to wait many days for the results

# ------------------------------------------------------------------------

## Not run:
# Example of parallelizing the process:
# Launch two (or more) runs separately on different processors, with results directed to a
# temporary folder. Set results.path="Yourfolder" to run operataionally.
# Launch batch 1 (on processor 1):
    model1 <- e2e_read("North_Sea", "1970-1999", model.ident="BATCH1")
```

```
    sens_results1 <- e2e_run_sens(model1, nyears=1, n_traj=3, coldstart=TRUE,
                   postprocess=FALSE, csv.output=TRUE)
# Note that coldstart=TRUE for the first batch only.
# Launch batch 2 (on processor 2):
    model2 <- e2e_read("North_Sea", "1970-1999", model.ident="BATCH2")
    sens_results1 <- e2e_run_sens(model2, nyears=1, n_traj=3, coldstart=FALSE,
                                  postprocess=FALSE, csv.output=TRUE)
# Note that these two runs return only raw data since postprocess=FALSE

# Then, afterwards, merge the two raw results files with text-tags BATCH1 and BATCH2,
# and post process the combined file:
    model3 <- e2e_read("North_Sea", "1970-1999", model.ident="COMBINED")
    processed_data <- e2e_merge_sens_mc(model3, selection="SENS",
                     ident.list<-c("BATCH1","BATCH2"), postprocess=TRUE, csv.output=TRUE)
# or...
    combined_data <- e2e_merge_sens_mc(model3, selection="SENS",
                    ident.list<-c("BATCH1","BATCH2"), postprocess=FALSE, csv.output=TRUE)
    processed_data <- e2e_process_sens_mc(model3, selection="SENS",
                     use.example=FALSE, csv.output=TRUE)

# Plot a diagram of parameter sensitivities from the combined data
    e2e_plot_sens_mc(model3, selection="SENS", use.example=FALSE)

## End(Not run)

# ------------------------------------------------------------------------
```

`e2e_run_ycurve` *Run a set of models to generate fishery yield curve data for either planktivorous or demersal fish.*

Description

Perform a set of StrathE2E model runs along a sequence of values of either planktivorous or demersal fish harvest ratio multiplier, saving the annual average whole-domain biomass, annual landings and annual discards of all exploitable food web guilds from each run plus the annual average biomasses of all the other living components of the food web.

Usage

```
e2e_run_ycurve(
  model,
  selection,
  nyears = 50,
  HRvector = c(0, 0.5, 1, 1.5, 2, 2.5, 3),
  HRfixed = 1,
  csv.output = FALSE
)
```

Arguments

`model`	R-list object defining a baseline model and configuration compiled by the e2e_read() function.

selection	Text string from a list identifying the fish guild for which a yield curve is to be generated. Select from: "PLANKTIV", "DEMERSAL", Remember to include the phrase within "" quotes.
nyears	Number of years to run the StrathE2E model at each level of harvest ratio. Default = 50.
HRvector	A vector of ascending order unique multiplier values to be applied to the baseline fish harvest ratio of the guild to be analysed. The values do not have to be evenly spaced (default = c(0,0.5,1.0,1.5,2.0,2.5,3.0)).
HRfixed	Single value of the multiplier to be applied as the alternative (planktivorous or demersal) fish harvest ratio for all runs (default=1.0).
csv.output	Logical. If TRUE then enable writing of csv output files (default=FALSE).

Details

The baseline for the sequence of runs (harvest ratio multiplier = 1.0) is a model name and variant as loaded by the e2e_read() function.

The planktivorous or demersal fish yield curve can be generated for a given fixed setting of the other (demersal or planktivorous) fish harvest ratio multiplier (default = 1.0). All other conditions are held constant as in the baseline model configuration.

The yield curve represents the catch that would be generated from the stationary state of the model attained with long-term repeating annual cycles of all driving data. Hence it is important that each simulation is run for long enough that the model attains its stationary state, which may be some distance from the baseline model initial conditions. It is recommended that each run is at least 50 years.

The data on annual average biomass and annual integrated catches stored in the returned data object (and optionally a csv file) can subsequently be plotted using the function e2e_plot_ycurve(). Users can easily plot any of the other saved data using their own plotting code.

If csv output is selected then the resulting files in the current user results folder have names Yield_curve_data_PFHRmult-*.csv or Yield_curve_data_DFHRmult-*.csv, depending on the 'selection' argument, and * represents the model.ident text set in the prior e2e_read() function call.

Value

Dataframe of annual average biomss, annual landings and annual discards of all exploitable guilds, plus average biomasses of other food web components, for each level of harvest ratio multiplier.

See Also

e2e_read, e2e_run, e2e_plot_ycurve

Examples

```
# Load the 1970-1999 version of the North Sea model supplied with the package :
    model <- e2e_read("North_Sea", "1970-1999", model.ident="70-99base")
# In this example csv output is directed to a temporary folder since results.path os not set.

# In this illustrative example the StrathE2E() model is run for only 3 years to enable quick
# return of results. In a real simulation nyear would be at least 50.
# This example illustrates that the vector of planktivorous fish harvest ratio multiplers
# does not have to be evenly spaced.
    hr <- c(0,0.5,0.75,1.0,1.25,2.0,3.0)
```

```
    pf_yield_data <- e2e_run_ycurve(model,selection="PLANKTIV", nyears=3, HRvector=hr,
                                    HRfixed=1,csv.output=FALSE)

# View the column names of the results dataframe:
    names(pf_yield_data)

# Plotting the results...
# The planktivorous fish yield curve can be plotted using the function:
    e2e_plot_ycurve(model, selection="PLANKTIV", results=pf_yield_data,
                    title="Planktivorous yield with baseline demersal fishing")
```

`StrathE2E2` *StrathE2E2 Package*

Description

StrathE2E2 is a dynamic model of the 'big-picture', whole ecosystem effects of hydrodynamics, temperature, nutrient additions, and fishing on continental shelf marine food webs.

Details

The StrathE2E2 model has two linked parts - a fishing fleet model and an ecology model. The fishing model integrates harvesting, discarding and seabed disturbance rates across a range of gears and passes the results into the ecology model.

The ecology model is a network of coupled ordinary differential equations representing the rates of change in organic detritus, dissolved inorganic nutrient, and coarse guilds of living biomass spanning microbes to megafauna. The equations include representations of feeding, metabolism, reproduction, active migrations, advection and mixing. Environmental driving data include temperature, irradiance, hydrodynamics, and nutrient inputs from rivers, atmosphere and ocean boundaries.

The package includes functions for parameter optimization, global sensitivity analysis, and Monte Carlo estimation of credible intervals for model outputs.

A fully developed and documented implementation for the North Sea is included in the package.

Documentation

List of all vignettes and documentation supplied with the package:
`../doc/index.html`

CheatSheet - quick-reference guide to StrathE2E2 functions ... `../doc/StrathE2E2_CheatSheet.pdf`
or ... from an R session ... vignette("StrathE2E2_CheatSheet")

Package User Manual (html) - guide to developing and running models

Package website: `https://marineresourcemodelling.gitlab.io/index.html`

Download documentation from the package website:
Technical Manual - documentation on the structure of input and output R-objects and files
Origin of the model (previous versions)
Overview of the model concepts and design
Ecology model description

Fishing fleet model description
Optimization, sensitivity and Monte Carlo methods
Documentation on the North Sea implementation of StrathE2E2 which is provided with the package

Repository for contributed models: (https://marineresourcemodelling.gitlab.io/resources/index.html)

Example data package

Example outputs from the more computationally intensive model functions are provided as a supplementary data package.

The data package auto-installs when example data are first invoked. Alternatively, to install manually:
... from an R session ... install.packages("StrathE2E2examples", repos="https://marineresourcemodelling.gitlab.io/sran")

Once installed, load the package using ... library(StrathE2E2examples)
Open documentation on the datasets using ... help(StrathE2E2examples)

See Also

Model management:

- e2e_ls: List the available models in a designated workspace.
- e2e_copy: Make a copy of a named model/variant in a user defined workspace.
- e2e_get_parmdoc: Download parameter documentation as a dataframe.

Basic model operations:

- e2e_read: Load a model setup from a given workspace.
- e2e_run: Run StrathE2E for a prescribed number of years with a given setup.
- e2e_extract_start: Create a new initial values file from the end of a model run.
- e2e_extract_hr: Extract the values of harvest ratios generated by the fishing fleet model.
- e2e_plot_ts: Time-series plots of model outputs for the full duration of a model run.

Parameter estimation:

- e2e_optimize_eco: Maximize the likelihood of observed ecosystem data by optimizing ecology model parameters.
- e2e_optimize_hr: Maximize the likelihood of observed ecosystem data by optimizing harvest ratio scaling parameters.
- e2e_optimize_act: Maximize the likelihood of observed ecosystem data or known harvest ratios by optimizing fishing activity parameters.
- e2e_plot_opt_diagnostics: Plot diagnostic data from optimization runs.
- e2e_calculate_hrscale: Calculate fishing fleet model parameters which link effort to harvest ratios.

Sensitivity and Monte Carlo analyses:

- `e2e_run_sens`: Run a global parameter sensitivity analysis with a given model setup.
- `e2e_run_mc`: Run a Monte Carlo analysis with a given model setup.
- `e2e_merge_sens_mc`: Merge parallel processing files from sensitivity or Monte Carlo runs.
- `e2e_process_sens_mc`: Process raw output data from sensitivity or Monte Carlo runs.
- `e2e_plot_sens_mc`: Plots diagnostic results from sensitivity and Monte Carlo analyses.
- `e2e_get_senscrit`: List the model outputs available as the basis for sensitivity analysis.

Compare model runs:

- `e2e_compare_obs`: Box-plot comparisons between observations and model outputs.
- `e2e_compare_runs_box`: Box-plot comparisons between two different model runs.
- `e2e_compare_runs_bar`: Tornado bar-plot comparisons between two different model runs.

Fishery yield analysis:

- `e2e_run_ycurve`: Perform a set of model runs to generate fishery yield curve data.
- `e2e_plot_ycurve`: Plot fishery yield curve data.

Visualize model inputs:

- `e2e_plot_edrivers`: Plot a climatological year of environmental driving data.
- `e2e_plot_fdrivers`: Plot distributions of fishery related driving data.

Visualize model outputs:

- `e2e_plot_eco`: Plot annual cycles of ecology model variables in the final year of a run.
- `e2e_plot_catch`: Plot annual landings and discards in the final year of a run.
- `e2e_plot_migration`: Plot annual cycles of active migration fluxes in the final year of a run.
- `e2e_plot_trophic`: Plot annual mean trophic level and omnivory indices.
- `e2e_plot_biomass`: Plot zonal distributions of annual average biomass densities.

Index

e2e_calculate_hrscale, 2, *72*
e2e_compare_obs, 3, *9*, *73*
e2e_compare_runs_bar, 5, *9*, *73*
e2e_compare_runs_box, 8, *73*
e2e_copy, 10, *16*, *57, 58*, *72*
e2e_extract_hr, 11, *72*
e2e_extract_start, 12, *72*
e2e_get_parmdoc, 13, *42*, *47*, *72*
e2e_get_senscrit, 14, *67*, *73*
e2e_ls, *3*, *11*, 15, *22*, *26*, *29*, *57, 58*, *72*
e2e_merge_sens_mc, 16, *55*, *62*, *67*, *73*
e2e_optimize_act, 19, *26*, *29*, *42*, *72*
e2e_optimize_eco, *22*, 24, *29*, *42*, *72*
e2e_optimize_hr, *22*, *26*, 27, *42*, *72*
e2e_plot_biomass, 30, *33*, *35*, *41*, *49*, *73*
e2e_plot_catch, *31*, 32, *35*, *41*, *49, 50*, *73*
e2e_plot_eco, *31*, *33*, 33, *41*, *49, 50*, *73*
e2e_plot_edrivers, 35, *39*, *73*
e2e_plot_fdrivers, *36*, 37, *73*
e2e_plot_migration, *31*, *33*, *35*, 39, *49*, *73*
e2e_plot_opt_diagnostics, *22*, *26*, *29*, 41, *72*
e2e_plot_sens_mc, *17*, 45, *55*, *62*, *67*, *73*
e2e_plot_trophic, *31*, *33*, *35*, *41*, 48, *73*
e2e_plot_ts, 50, *58*, *72*
e2e_plot_ycurve, 51, *70*, *73*
e2e_process_sens_mc, *17*, 53, *62*, *67*, *73*
e2e_read, *3, 4*, *7*, *9*, *11–14*, *16, 17*, *22*, *26*, *29*, *31*, *33*, *35, 36*, *39*, *41, 42*, *47*, *49, 50*, *52*, *55*, 55, *58*, *62*, *67*, *70*, *72*
e2e_run, *4*, *7*, *9*, *12, 13*, *31*, *33*, *35, 36*, *39*, *41*, *49, 50*, *52*, *57*, 58, *70*, *72*
e2e_run_mc, *4*, *9*, *17*, *31*, *33*, *35*, *41*, *47*, *49*, *55*, 59, *73*
e2e_run_sens, *15*, *17*, *47*, *55*, 64, *73*
e2e_run_ycurve, *52*, 69, *73*

StrathE2E2, 71