Package: SSMSE (via r-universe)

September 21, 2024

Type Package

Title Management Strategy Evaluation (MSE) using Stock Synthesis (SS)

Version 0.2.8

Description An R package for performing Management Strategy Evaluation (MSE) using Stock Synthesis (SS). SS is used as the Operating Model (OM) and, if the user desires, the Estimation model (EM). SSMSE allows existing SS models to be used as the basis for an OM. These OMs are used in the MSE framework provided by SSMSE to evaluate the implications of management actions on a population given uncertainty.

License MIT + file LICENSE.md

URL https://github.com/nmfs-fish-tools/SSMSE

BugReports https://github.com/nmfs-fish-tools/SSMSE/issues

Depends R (>= 4.0)

Imports assertive.base, assertive.properties, assertive.types, doParallel, dplyr, foreach, ggplot2, magrittr, parallel, r4ss, ss3sim, stats, tidyr, utils

Suggests testthat

Remotes cran/assertive.base@master, cran/assertive.properties@master, cran/assertive.types@master, r4ss/r4ss@main, ss3sim/ss3sim@main

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.1

Repository https://noaa-fisheries-integrated-toolbox.r-universe.dev

RemoteUrl https://github.com/nmfs-fish-tools/SSMSE

RemoteRef HEAD

RemoteSha 1f12b27f65403062d914a79d6e239e5ccb3e9dc0

Contents

`add_dev_changes` *Add the deviation changes from the list obj to an existing df*

Description

Add the deviation changes from the list obj to an existing df

Usage

```
add_dev_changes(fut_list, scen, iter, parlist, dat, vals_df, nyrs, ctl)
```

Arguments

fut_list	A single change input
scen	The scenario name
iter	The iteration name
parlist	A parameter file as read in by r4ss::SS_readpar_3.30
dat	A data file as read in by r4ss::SS_readdat.
vals_df	The dataframe with future om values
nyrs	The number of years to extend the model forward
ctl	A control file as read in by r4ss::SS_readctl.

Value

A modified version of vals_df with the new changes applied.

Author(s)

Kathryn Doering

---

add_new_dat *Add new data to an existing EM dataset*

Description

This should be used for the feedback loops when an EM is used.

Usage

```
add_new_dat(
  OM_dat,
  EM_datfile,
  sample_struct,
  EM_dir,
  nyrs_assess,
  do_checks = TRUE,
  new_datfile_name = NULL,
  verbose = FALSE
)
```

Arguments

OM_dat	An valid SS data file read in using r4ss. In particular, this should be sampled data.
EM_datfile	Datafile name run in previous iterations with the EM. Assumed to exist in EM_dir.

sample_struct	A optional list including which years, seasons, and fleets should be added from the OM into the EM for different types of data. If NULL, the data structure will try to be inferred from the pattern found for each of the datatypes within the EM datafiles. Include this structure for the number of years to extend the model out. Note that the data should be specified using the list component names and column names as in would be used in `r4ss::SS_readdat()`. The `run_SSMSE_iter` function examples give an example of what this structure should be. Running the function `create_sample_struct()` will also produce a sample_struct object in the correct form. Can be NULL only when MS is not EM.
EM_dir	Absolute or relative path to the Estimation model directory.
nyrs_assess	The number of years between assessments. E.g., if an assessment is conducted every 3 years, put 3 here. A single integer value.
do_checks	Should checks on the data be performed? Defaults to TRUE.
new_datfile_name	An optional name of a file to write the new datafile to. If NULL, a new datafile will not be written.
verbose	Want verbose output? Defaults to FALSE.

Value

A new SS datafile containing the data in EM_datfile with new data from OM_dat appended

Author(s)

Kathryn Doering

`add_OM_devs` *Add in future parameter values*

Description

Add in future parameter values

Usage

```
add_OM_devs(ctl, dat, parlist, timeseries, future_om_dat)
```

Arguments

ctl	A control file as read in by `r4ss::SS_readctl`.
dat	A data file as read in by `r4ss::SS_readdat`.
parlist	A parameter file as read in by `r4ss::SS_readpar_3.30`
timeseries	The timeseries table from `r4ss::SS_output()`.
future_om_dat	A data frame with random sample data for future parameter

Author(s)

Nathan Vaughan

`add_sample_struct` *Add in years of sampling data needed*

Description

Add in years of sampling data needed

Usage

```
add_sample_struct(sample_struct, dat)
```

Arguments

`sample_struct`	A optional list including which years, seasons, and fleets should be added from the OM into the EM for different types of data. If NULL, the data structure will try to be inferred from the pattern found for each of the datatypes within the EM datafiles. Include this structure for the number of years to extend the model out. Note that the data should be specified using the list component names and column names as in would be used in `r4ss::SS_readdat()`. The `run_SSMSE_iter` function examples give an example of what this structure should be. Running the function `create_sample_struct()` will also produce a sample_struct object in the correct form. Can be NULL only when MS is not EM.
`dat`	A datafile as read in by r4ss::SS_readdat

`calc_comp_var` *Calculate uncertainty and biases in historic composition data*

Description

Calculate uncertainty and biases in historic composition data

Usage

```
calc_comp_var(
  data_obs,
  data_exp,
  bins,
  fleets = NULL,
  years = NULL,
  seasons = NULL,
  merge_genders = TRUE,
  genders = NULL,
  merge_seasons = TRUE,
  merge_fleets = FALSE
)
```

Arguments

data_obs	A data frame of observed composition data extracted from SS .dat file
data_exp	A data frame of the expected composition data as estimated by an SS assessment model
bins	A vector object including the composition bins
fleets	A vector of the fleet numbers to analyze composition uncertainty for (Default is all fleets if NULL)
years	A vector of the years to include when calculating composition uncertainty (Default is all years if NULL)
seasons	A vector of the seasons to include when calculating composition uncertainty (Default is all years if NULL)
merge_genders	TRUE/FALSE should genders be merged to calculate variance and biases (Defaults to TRUE)
genders	A vector of the genders to analyze composition uncertainty for (Default is all genders if NULL)
merge_seasons	TRUE/FALSE should seasons be merged to calculate variance and biases (Defaults to TRUE)
merge_fleets	TRUE/FALSE should fleets be merged to calculate variance and biases (Defaults to FALSE)

Value

A list object with uncertainty and bias characteristics to inform data simulation.

Author(s)

Nathan R. Vaughan

---

calc_par_trend *Calculate the parameter trend*

Description

Calculate the parameter trend

Usage

```
calc_par_trend(
  val_info,
  val_line = c("mean", "sd", "cv", "ar_1_phi"),
  ref_parm_value,
  vals_df,
  parname,
  parlist,
```

```
  ctl,
  par_section,
  dat
)
```

Arguments

val_info	The line in the input df containing info about the parameter.
val_line	Which line in val info to use.
ref_parm_value	This is the historic parameter that the end trend value. Can be NA if the there is no line in val_info for the given parameter
vals_df	The dataframe of the parameter values by year. Use to get start val and last year
parname	Name of the parameter with devs from the SS model. will reference, if using a relative method.
parlist	A parameter file as read in by `r4ss::SS_readpar_3.30`
ctl	A control file as read in by `r4ss::SS_readctl`.
par_section	Which parameter section should this variabile be in?
dat	A data file as read in by `r4ss::SS_readdat`.

Value

A vector of values with length ncol(vals_df), the number of future years.

Author(s)

Kathryn Doering

---

change_dat *Change dataset from OM into format for EM*

Description

Change dataset from OM into format for EM

Usage

```
change_dat(OM_datfile, EM_datfile, EM_dir, do_checks = TRUE, verbose = FALSE)
```

Arguments

OM_datfile	Filename of the datfile produced by the OM within the EM_dir.
EM_datfile	Filename of the datfile from the original EM within the EM_dir.
EM_dir	Absolute or relative path to the Estimation model directory.
do_checks	Should checks on the data be performed? Defaults to TRUE.
verbose	Want verbose output? Defaults to FALSE.

Value

the new EM data file. Side effect is saving over the OM_dat file in EM_dir.

Author(s)

Kathryn Doering

Examples

```
## Not run:
# TODO: Add example

## End(Not run)
```

`change_yrs_fcast` *Change the years in the forecast file*

Description

This is both to increment years forward and/or to change absolute years to relative years.

Usage

```
change_yrs_fcast(
  fore,
  make_yrs_rel = TRUE,
  nyrs_increment = NULL,
  nyrs_fore = NULL,
  mod_styr,
  mod_endyr
)
```

Arguments

`fore`	A forecasting file read into R using r4ss::SS_readforecast()
`make_yrs_rel`	Should the absolute years in the forecast file be changed to relative years? Defaults to TRUE.
`nyrs_increment`	The number of years to increment forecasting period years. If NULL (the default value), will not be incremented.
`nyrs_fore`	The number of years of forecasting to do. If NULL, do not change the number of forecasting years already specified in `fore`
`mod_styr`	The first year of the model
`mod_endyr`	The last year of the model `fore` assumes when read in. Note that the assumed model year will be different for the output if nyrs_increment is not NULL.

Value

A forecasting file as an R list object

Author(s)

Kathryn Doering

`check_avail_dat` *check all index years/fleets in EM available in OM. (but not vice versa) a general function that can be used*

Description

check all index years/fleets in EM available in OM. (but not vice versa) a general function that can be used

Usage

```
check_avail_dat(
  EM_dat,
  OM_dat,
  list_item = "CPUE",
  colnames = c("year", "seas", "index")
)
```

Arguments

`EM_dat`	An SS data file read in using r4ss for an EM
`OM_dat`	An SS data file read in using r4ss for an OM
`list_item`	A component in both EM_dat and OM_dat to check values for. This should be a single string value.
`colnames`	The column names of data to append together.

Author(s)

Kathryn Doering

check_catch_df *Check the catch dataframe*

Description

Ensure the catch data frame has the correct column names in the correct order and the correct number of column names.

Usage

```
check_catch_df(df)
```

Arguments

df	The catch dataframe to test

Author(s)

Kathryn Doering

check_convergence *Flag potential convergence issues in SS3 model runs*

Description

Does basic checks for convergance of estimation model runs from `run_SSMSE()` simulations. This function 1) warns if there are parameters on bounds; 2) warns if the SSB in the EM is 2x as large or half as small as the OM. Note these warnings may not mean that the models have not converged, but can flag potential issues that can be investigated further

Usage

```
check_convergence(summary, min_yr, max_yr)
```

Arguments

summary	Summary returned from running `SSMSE_summary_all()`
min_yr	The first year of SSB checked
max_yr	The last year of SSB checked

Value

A tibble containing the SSB values in the EM relative to the OM by model run of each iteration of each scenario.

Examples

```
## Not run:
check_convergance(SSMSE_summary, min_yr = 101, max_yr = 120)

## End(Not run)
```

`check_dir` *Check that the directory for an OM is valid*

Description

Check that the directory contains starter and forecast SS files.

Usage

```
check_dir(dir)
```

Arguments

`dir`	Input to check. Should be a directory name that should contain an SS model that can be used as an OM. @author Kathryn Doering

---

`check_EM_forecast` *Check structure of forecast is suitable to use in the EM*

Description

Check structure of forecast is suitable to use in the EM

Usage

```
check_EM_forecast(fore, n_flts_catch = NULL)
```

Arguments

`fore`	A forecast list read in using r4ss::SS_readforecast
`n_flts_catch`	The number of fleets with catch. If NULL, this function will skip a check requiring this input.

Value

Function mainly used for side effects, but returns TRUE invisibly if no errors created.

Author(s)

Kathryn Doering

check_future_catch *Check future catch smaller than the last year's population size.*

Description

Note that it could still be possible to take out too much catch from the population, so this may not catch all instances of too much catch

Usage

```
check_future_catch(catch, OM_dir, catch_units = "bio", datfile = NULL)
```

Arguments

catch	A dataframe of catch values and its associated information to add to the OM. The column names are the same as in an SS data file (e.g., year, season, fleet, catch, catch_se). length of the number of years (only works when catch is for 1 fleet)
OM_dir	The full path to the OM directory.
catch_units	What units is the catch in? "bio" for biomass or "num" for numbers? Defaults to "bio".
datfile	The optional name (as a character string) of the datafile, presumed to exist in OM_dir. Defaults to NULL, and if is NULL, the function will get the datfile name from the starter.ss file in OM_dir.

Author(s)

Kathryn Doering

check_future_om_list_str *Check the general structure of a future OM list and standardize values*

Description

Checks that a future OM list is valid. If any values are implicit, then add these values. Does not check against arguments in the scenario, just the generic structure

Usage

```
check_future_om_list_str(future_om_list)
```

Arguments

future_om_list	An optional list of lists including changes that should be made after the end year of the input model. Each first level list element outlines 1 change to be made to the operating model. To see an example, try running create_future_om_list. Defaults to NULL, which implies that the model will be extended forward in time assuming the original model structure.

Value

The future_om_list with implicit arguments made explicit

---

check_future_om_list_vals

Check structure of a future OM list against the scen_list and standardize output

---

Description

Checks that a future OM list is valid when compared with the scen_list inputs

Usage

```
check_future_om_list_vals(future_om_list, scen_list)
```

Arguments

future_om_list	An optional list of lists including changes that should be made after the end year of the input model. Each first level list element outlines 1 change to be made to the operating model. To see an example, try running create_future_om_list. Defaults to NULL, which implies that the model will be extended forward in time assuming the original model structure.
scen_list	The list object of scenarios specifying inputs created by SSMSE::create_scen_list.

Value

The future_om_list with implicit arguments made explicit

check_OM_dat *check that an OM data set has at least the same data as an estimation model*

Description

check that an OM data set has at least the same data as an estimation model

Usage

```
check_OM_dat(OM_dat, EM_dat)
```

Arguments

OM_dat	A data set read in using r4ss::SS_readdat from an operating model. Note that it should span the same years as EM_dat.
EM_dat	A data set read in using r4ss::SS_readdata from an estimation model. Note that it should span the same years as EM_dat

Author(s)

Kathryn Doering

check_sample_struct *Check sample_struct_list*

Description

Check that list object sample_struct_list has the expected form, including the correct names, correct column names (as in r4ss), and that all values in the dataframes are integer or numeric. This does not check for if numeric or interger values make sense given the model used.

Usage

```
check_sample_struct(
  sample_struct,
  valid_names = list(catch = c("Yr", "Seas", "FltSvy", "SE"), CPUE = c("Yr", "Seas",
   "FltSvy", "SE"), lencomp = c("Yr", "Seas", "FltSvy", "Sex", "Part", "Nsamp"), agecomp
   = c("Yr", "Seas", "FltSvy", "Sex", "Part", "Ageerr", "Lbin_lo", "Lbin_hi", "Nsamp"),
   meanbodywt = c("Yr", "Seas", "FltSvy", "Part", "Type", "Std_in"), MeanSize_at_Age_obs
    = c("Yr", "Seas", "FltSvy", "Sex", "Part", "Ageerr", "N_"))
)
```

Arguments

sample_struct	A optional list including which years, seasons, and fleets should be added from the OM into the EM for different types of data. If NULL, the data structure will try to be inferred from the pattern found for each of the datatypes within the EM datafiles. Include this structure for the number of years to extend the model out. Note that the data should be specified using the list component names and column names as in would be used in `r4ss::SS_readdat()`. The `run_SSMSE_iter` function examples give an example of what this structure should be. Running the function `create_sample_struct()` will also produce a sample_struct object in the correct form. Can be NULL only when MS is not EM.
valid_names	The list to compare sample_struct to.

Author(s)

Kathryn Doering

---

check_scen_list *Check structure of the object scen_list*

Description

Check the structure that is input to `run_SSMSE`.

Usage

```
check_scen_list(list, verbose = FALSE)
```

Arguments

list	A list to check
verbose	Want verbose output? Defaults to FALSE.

Author(s)

Kathryn Doering

clean_init_mod_files *clean the initial model files*

Description

clean the initial model files

Usage

```
clean_init_mod_files(
  OM_out_dir,
  EM_out_dir = NULL,
  MS = "EM",
  overwrite = FALSE
)
```

Arguments

OM_out_dir	The full path to the directory in which the OM is run.
EM_out_dir	Relative or absolute path to the estimation model, if using a model outside of the SSMSE package.
MS	The management strategy to use. Current options are: "last_yr_catch" which uses the previous year's catch; "no_catch" which uses 0 catch; "EM" which uses an stock synthesis model as the estimation method and the management strategy as defined in the forecast file of the stock synthesis estimation method; "Interim" to modify catch based on survey predictions between assessments. Users can also specify their own management strategies as a function. For example, if the function is called "my_ms" then the user should specify MS = "my_ms" and specify the path to the file containing the function in custom_MS_source.
overwrite	Allow existing files to be overwritten?

combine_cols *function that creates a combined column to the list_item of interest*

Description

function that creates a combined column to the list_item of interest

Usage

```
combine_cols(dat_list, list_item, colnames)
```

Arguments

dat_list	An SS data file as a list read in using r4ss
list_item	List item in dat_list to extract and return a modified version of this value
colnames	Column names in list_item

---

convert_future_om_list_to_devs_df

Create the devs dataframe for a scenario and iteration from user input

---

Description

This function parses user inputs to convert it into a dataframe of deviations.

Usage

```
convert_future_om_list_to_devs_df(
  future_om_list,
  scen_name,
  niter,
  om_mod_path,
  nyrs,
  global_seed = 123
)
```

Arguments

future_om_list	An optional list of lists including changes that should be made after the end year of the input model. Each first level list element outlines 1 change to be made to the operating model. To see an example, try running create_future_om_list. Defaults to NULL, which implies that the model will be extended forward in time assuming the original model structure.
scen_name	The scenario name
niter	The iteration number
om_mod_path	Path to the OM files. Used to reference parameter names.
nyrs	The total number of years that the model will be extended forward.
global_seed	A global seed to set, then pull new seeds from. Defaults to 123.

Value

A list including 3 dataframes and one list: devs_df, the additive deviations relative to the base values; base_df, the base values of the parameter with deviations; abs_df, the absolute future values by year (first col) and parameter (parameterss in different cols). Also includes a modified version of the future_om_list which includes the seed applied to each list component (note that this is not the ultimate seed used for sampling, as addiitonal) seeds are generated from this seed based on the scenario, iteration, and option for randomness (replicate across scenarios or randomize across scenarios). Note that no OM files are modified or created as part of this function (i.e., it does not have side effects).

Author(s)

Kathryn Doering

`convert_to_r4ss_names` *Convert user input to r4ss data names*

Description

Convert user input to r4ss data names

Usage

```
convert_to_r4ss_names(
  sample_struct,
 convert_key = data.frame(df_name = c(rep("catch", 4), rep("CPUE", 4), rep("lencomp",
   6), rep("agecomp", 9), rep("meanbodywt", 6), rep("MeanSize_at_Age_obs", 7)),
  r4ss_name = c("year", "seas", "fleet", "catch_se", "year", "seas", "index", "se_log",
  "Yr", "Seas", "FltSvy", "Gender", "Part", "Nsamp", "Yr", "Seas", "FltSvy", "Gender",
    "Part", "Ageerr", "Lbin_lo", "Lbin_hi", "Nsamp", "Year", "Seas", "Fleet",
  "Partition", "Type", "Std_in", "Yr", "Seas", "FltSvy", "Gender", "Part", "AgeErr",
    "N_"), sample_struct_name = c("Yr",
     "Seas", "FltSvy", "SE", "Yr", "Seas",
   "FltSvy", "SE", "Yr", "Seas", "FltSvy", "Sex", "Part", "Nsamp", "Yr", "Seas",
  "FltSvy", "Sex", "Part", "Ageerr", "Lbin_lo", "Lbin_hi", "Nsamp", "Yr", "Seas",
  "FltSvy", "Part", "Type", "Std_in", "Yr", "Seas", "FltSvy", "Sex", "Part", "Ageerr",
    "N_"), stringsAsFactors = FALSE)
)
```

Arguments

`sample_struct`	A optional list including which years, seasons, and fleets should be added from the OM into the EM for different types of data. If NULL, the data structure will try to be inferred from the pattern found for each of the datatypes within the EM datafiles. Include this structure for the number of years to extend the model out. Note that the data should be specified using the list component names and column names as in would be used in `r4ss::SS_readdat()`. The `run_SSMSE_iter` function examples give an example of what this structure should be. Running the function `create_sample_struct()` will also produce a sample_struct object in the correct form. Can be NULL only when MS is not EM.
`convert_key`	Data frame defining how r4ss names relate to the sample_struct names. For now, a 1:1 relationship is assumed.

copy_model_files *Copy OM and EM model files*

Description

Copy OM and EM model files from input to output location.

Usage

```
copy_model_files(
  OM_in_dir = NULL,
  OM_out_dir = NULL,
  EM_in_dir = NULL,
  EM_out_dir = NULL,
  verbose = FALSE
)
```

Arguments

OM_in_dir	Relative or absolute path to the operating model, if using a model outside of the SSMSE package. Should be a string.
OM_out_dir	The full path to the directory in which the OM is run.
EM_in_dir	Relative or absolute path to the estimation model,
EM_out_dir	Relative or absolute path to the estimation model, if using a model outside of the SSMSE package.
verbose	Want verbose output? Defaults to FALSE.

Value

TRUE, if copying is successful

create_future_om_list *Helper function to create future om list objects*

Description

The future_om_list objects specify changes to make in the future to the OM as the OM is extended forward in time. In particular, this function helps users create these objects. For now, just returns examples based on cod model that comes with SSMSE. To learn more about the options available for future_om_list object, see the structure of future_om_list section of the SSMSE user manual.

Usage

```
create_future_om_list(
  example_type = c("model_change", "custom"),
  list_length = 1
)
```

Arguments

example_type	Type of example future_om_list object to create. Options are "model_change" or "custom". Defaults to "model_change".
list_length	The length of the example list to create. Defaults to 1. For now, just replicates the same list.

Examples

```
example_future_om_list <-
  create_future_om_list(example_type = "custom", list_length = 2)
```

---

create_OM *Create the OM*

Description

This function manipulates the OM as needed so that it can be used as an operating model.

Usage

```
create_OM(
  OM_out_dir,
  overwrite = TRUE,
  writedat = TRUE,
  verbose = FALSE,
  nyrs = NULL,
  nyrs_assess = NULL,
  nscen = 1,
  scen_name = NULL,
  niter = 1,
  future_om_dat = NULL,
  verify_OM = TRUE,
  sample_struct_hist = NULL,
  sample_struct = NULL,
  seed = NULL
)
```

Arguments

OM_out_dir	The full path to the directory in which the OM is run.
overwrite	Allow existing files to be overwritten?
writedat	Should a new datafile be written?
verbose	Want verbose output? Defaults to FALSE.
nyrs	Number of years beyond the years included in the OM to run the MSE. A single integer value.
nyrs_assess	The number of years between assessments. This is used to structure the forecast file for use in the OM.
nscen	The scenario number
scen_name	The scenario name
niter	the iteration number
future_om_dat	An optional data_frame including changes that should be made after the end year of the input model. Including parameter variations, recruitment deviations, and implementation errors.
verify_OM	Should the model be run without estimation and some basic checks done to verify that the OM can run? Defaults to TRUE.
sample_struct_hist	An optional list including which years should be sampled for the historical period for the data generated from the OM. If this is left as NULL, then the same sampling scheme will be used as in the OM's data file. If it is not NULL, then each year.
sample_struct	A optional list including which years, seasons, and fleets should be added from the OM into the EM for different types of data. If NULL, the data structure will try to be inferred from the pattern found for each of the datatypes within the EM datafiles. Include this structure for the number of years to extend the model out. Note that the data should be specified using the list component names and column names as in would be used in `r4ss::SS_readdat()`. The `run_SSMSE_iter` function examples give an example of what this structure should be. Running the function `create_sample_struct()` will also produce a sample_struct object in the correct form. Can be NULL only when MS is not EM.
seed	A random seed so that reproducible results are possible.

Value

A modified datafile

Author(s)

Kathryn Doering & Nathan Vaughan

create_out_dirs *create the OM directory*

Description

Create an OM directory within the out_dir specified (named by the value of niter)

Usage

```
create_out_dirs(
  out_dir,
  niter,
  OM_name,
  OM_in_dir,
  EM_name = NULL,
  EM_in_dir = NULL
)
```

Arguments

out_dir	The directory to which to write output. IF NULL, will default to the working directory.
niter	The number iteration
OM_name	Name of the operating model (OM).
OM_in_dir	Relative or absolute path to the operating model, if using a model outside of the SSMSE package. Should be a string.
EM_name	Name of the EM model
EM_in_dir	Relative or absolute path to the estimation model,

Value

A list with 2 named components each of length 1 characters. The components are: OM_dir, where OM will be run, and OM_in_dir, where the model files will be copied from.

create_sample_struct *Create the sample_struct list*

Description

Create a sampling structure list using the pattern in a data file and a year range. NAs are added if no pattern is found (and rm_NAs = FALSE). The types of structure that are added to this list (given their presence in the dat file) with their names as called in the list object in parentheses are: catch (catch), relative indices (CPUE), length composition (lencomp), age composition (agecomp), mean body weight (meanbodywt), and mean size at age (MeanSize_at_Age_obs). Details for creating the sample structure list are available in the sampling options section of the SSMSE user manual.

Usage

```
create_sample_struct(dat, nyrs, rm_NAs = FALSE)
```

Arguments

dat	An r4ss list object read in using r4ss::SS_readdat() or the path (relative or absolute) to an SS data file to read in.
nyrs	Number of years beyond the years included in the dat file to run the MSE. A single integer value.
rm_NAs	Should all NAs be removed from dataframes? Defaults to FALSE.

Value

A sample_struct list object, where each list element is a dataframe containing sampling values. If there were no data for the type, NA is returned for the element.

Author(s)

Kathryn Doering

Examples

```
OM_path <- system.file("extdata", "models", "cod", "ss3.dat", package = "SSMSE")
# note there is a warning for lencomp because it does not have a consistent pattern
sample_struct <- create_sample_struct(OM_path, nyrs = 20)
print(sample_struct)
```

---

create_scen_list *Create scen_list object to use in run_SSMSE function.*

Description

Function to create parameter `scen_list` in `run_SSMSE`, but also could be used by users to construct their list prior to using `run_SSMSE`. Note that there is no error checking in this function, so getting output does not insure that this output can be used as input to run_SSMSE.

Usage

```
create_scen_list(
  scen_name_vec,
  out_dir_scen_vec = NULL,
  iter_vec = NULL,
  OM_name_vec = NULL,
  OM_in_dir_vec = NULL,
  EM_name_vec = NULL,
  EM_in_dir_vec = NULL,
  MS_vec = NULL,
```

```
  use_SS_boot_vec = NULL,
  nyrs_vec = NULL,
  nyrs_assess_vec = NULL,
  sample_struct_list = NULL,
  sample_struct_hist_list = NULL,
  sample_catch_vec = NULL,
  interim_struct_list = NULL
)
```

Arguments

scen_name_vec	A vector containing names of the scenarios. The each string will be a directory containing all the model runs for a scenario.s
out_dir_scen_vec	The directory to which to write output. IF NULL, will default to the working directory.
iter_vec	The number of iterations per scenario. A vector of integers in the same order as scen_name_vec.
OM_name_vec	Names of a valid Stock Synthesis stock assessment model. To see the names of built-in models, type `list.dirs(system.file("extdata", "models", package = "SSMSE"),full.names = FALSE, recursive = FALSE)` into the R console.
OM_in_dir_vec	Vector of relative or absolute paths to the operating model, if using a model outside of the SSMSE package.
EM_name_vec	Should be NULL unless `MS = "EM"`. Name of a valid Stock Synthesis stock assessment model to use as an EM. If the value of EM_name is NULL and `MS = "EM"`, then SSMSE will look for the estimation model in the path specified in EM_in_dir. valid inputs for EM_name are: `"cod"` or `NULL`.
EM_in_dir_vec	Relative or absolute path to the estimation model, if using model outside of the SSMSE package. Note that this value should be NULL if `MS` has a value other than "EM".
MS_vec	Vector of management strategies. Current options are: `"last_yr_catch"` which uses the previous year's catch; `"no_catch"` which uses 0 catch; `"EM"` which uses an stock synthesis model as the estimation method and the management strategy as defined in the forecast file of the stock synthesis estimation method; `"Interim"` to modify catch based on survey predictions between assessments. Users can also specify their own managment strategies as an function. To use the function, it must be available in the global enviroment and specified by name in MS. For example, if the function is called "my_ms" then the user should make it available in the global environment and specify "my_ms" as a component of MS_vec.
use_SS_boot_vec	Should a bootstrapped data set generated by SS be used? Defaults to TRUE.
nyrs_vec	Number of years beyond the years included in the OM to run the MSE. A single integer value.
nyrs_assess_vec	The number of years between assessments. E.g., if an assessment is conducted every 3 years, put 3 here. A single integer value. (NOTE: This could be made

more flexible by instead reading in a vector of assessment years, so users could specify irregular numbers of yrs between assessments.)

`sample_struct_list`
: A optional list of lists including which years, seasons, and fleets should be added from the OM into the EM for different types of data. If NULL, the data structure will try to be infered from the pattern found for each of the datatypes within the EM datafiles. Include this strucutre for the number of years to extend the model out.

`sample_struct_hist_list`
: An optional list of lists including which years should be sampled for the historical period for the data generated from the OM. If this is left as NULL, then the same sampling scheme will be used as in the OM's data file. If it is not NULL, then each year

`sample_catch_vec`
: Should catch be sampled or fixed at the OM values? This can be a single Boolean (TRUE or FALSE) to apply to all scenarios or a vector of the same length as the number of scenarios. Defaults to FALSE.

`interim_struct_list`
: A optional list of parameters to control an interim assessment with an example structure below, where Beta=a positive value that is inversely proportional to risk, MA_years= the number of years to average index observations of when calculating deviations, assess_freq=the number of years between full assessments during with an interim assessment will happen every year, and Index_weights is a vector of length n indexes that weights all indexes for multi index inference. `interim_struct_list<-list(Beta=1,MA_years=3,assess_freq=5,Index_weights=rep(1,max(re`

Author(s)

Kathryn Doering

Examples

```
scen_list <- create_scen_list(
  scen_name_vec = c("scen 1", "scen_2"),
  out_dir_scen_vec = file.path("path", "to", "dir"),
  iter_vec = list(1:2, 5:7),
  OM_name_vec = "cod",
  OM_in_dir_vec = NULL,
  EM_name_vec = "cod",
  EM_in_dir_vec = NULL,
  MS_vec = "EM",
  use_SS_boot_vec = TRUE,
  nyrs_vec = 6,
  nyrs_assess_vec = 3,
  sample_struct_list = NULL
)
```

develop_OMs *Develop different operating models*

Description

This is a utility to help a user create new operating models starting from the same model. For now, it is only possible to adjust 1 parameter value

Usage

```
develop_OMs(
  OM_name = NULL,
  OM_in_dir = NULL,
  out_dir = getwd(),
  par_name,
  par_vals,
  refit_OMs = TRUE,
  hess = FALSE
)
```

Arguments

OM_name	Name of the operating model (OM).
OM_in_dir	Relative or absolute path to the operating model, if using a model outside of the SSMSE package. Should be a string.
out_dir	Path where the new models will be written. Defaults to the current working directory.
par_name	Name of the parameter to modify
par_vals	Vector of parameter values to modify in the OM. Assume these will be fixed so phase will be set as negative.
refit_OMs	Should the models be refit to data? Defaults to TRUE
hess	Should the hessian be estimated if reffiting the OMs? defaults to FALSE

EM *Use EM as the management strategy option.*

Description

Use EM as the management strategy option.

Usage

```
EM(
  EM_out_dir = NULL,
  init_loop = TRUE,
  OM_dat,
  verbose = FALSE,
  nyrs_assess,
  dat_yrs,
  sample_struct = NULL,
  seed = NULL,
  OM_out_dir,
  ...
)
```

Arguments

EM_out_dir	Relative or absolute path to the estimation model, if using a model outside of the SSMSE package.
init_loop	Logical. If this is the first initialization loop of the MSE, init_loop should be TRUE. If it is in further loops, it should be FALSE.
OM_dat	An valid SS data file read in using r4ss. In particular, this should be sampled data.
verbose	Want verbose output? Defaults to FALSE.
nyrs_assess	The number of years between assessments. E.g., if an assessment is conducted every 3 years, put 3 here. A single integer value.
dat_yrs	Which years should be added to the new model? Ignored if init_loop is TRUE.
sample_struct	A optional list including which years, seasons, and fleets should be added from the OM into the EM for different types of data. If NULL, the data structure will try to be inferred from the pattern found for each of the datatypes within the EM datafiles. Include this structure for the number of years to extend the model out. Note that the data should be specified using the list component names and column names as in would be used in r4ss::SS_readdat(). The run_SSMSE_iter function examples give an example of what this structure should be. Running the function create_sample_struct() will also produce a sample_struct object in the correct form. Can be NULL only when MS is not EM.
seed	A random seed so that reproducible results are possible.
OM_out_dir	The full path to the directory in which the OM is run.
...	Any additional parameters

Author(s)

Kathryn Doering

get_avg_catch *Example Performance Metric: Calculate average catch over a range of years*

Description

Example performance metric that calculates average catch over a range of years in a Stock Synthesis data file. This function aggregates across fleets, so may not be appropriate for models with multiple fleets.

Usage

```
get_avg_catch(datfile, yrs)
```

Arguments

datfile	Path to the Stock Synthesis data file containing catch
yrs	A vector containing a range of years. Years are as defined in the Stock Synthesis data file.

Value

The average catch, a number.

Examples

```
## Not run:
avg_catch <- function(datfile = "ss3model/dat.ss", yrs = 30:75) {
  avg_catch
}

## End(Not run)
```

get_bin *Get SS3 binary/executable location in package*

Description

Get the binary/executable location in the package SSMSE. This function is from ss3sim.

Usage

```
get_bin(bin_name = "ss")
```

Arguments

bin_name	Name of SS3 binary, defaults to "ss"

Value

The path to an SS binary. If using the GitHub version of the package, this will be an internal binary. Otherwise, this function will search for a version of the binary in your path. See the ss3sim vignette.

Examples

```
## Not run:
get_bin()

## End(Not run)
```

`get_catch_cv` *Example Performance Metric: Calculate the coefficient of variation of catch*

Description

Example performance metric that calculates the coefficient of variation (CV) of catch over a range of years in a Stock Synthesis data file. This function aggregates across fleets, so may not be appropriate for models with multiple fleets.

Usage

```
get_catch_cv(datfile, yrs)
```

Arguments

`datfile`	Path to the Stock Synthesis data file containing catch
`yrs`	A vector containing a range of years. Years are as defined in the Stock Synthesis data file.

Value

The catch coefficient of variation, a number.

Examples

```
## Not run:
catch_cv <- get_catch_cv(datfile = "mod/dat.ss", yrs = c(20:50, 75:100))
catch_cv

## End(Not run)
```

get_catch_sd *Example Performance Metric: Calculate Standard Deviation of Catch*

Description

Example performance metric that calculates the standard deviation of catch over a range of years in a Stock Synthesis data file. This function aggregates across fleets, so may not be appropriate for models with multiple fleets.

Usage

```
get_catch_sd(datfile, yrs)
```

Arguments

datfile	Path to the Stock Synthesis data file containing catch
yrs	A vector containing a range of years. Years are as defined in the Stock Synthesis data file.

Value

The catch standard deviation, a number.

Examples

```
## Not run:
catch_sd <- get_catch_sd(datfile = "mod/dat.ss", yrs = c(20:50, 75:100))
catch_sd

## End(Not run)
```

get_dead_catch *Get dead catch from the timeseries Report.sso table*

Description

Get dead catch from the timeseries Report.sso table

Usage

```
get_dead_catch(timeseries, units_of_catch)
```

Arguments

timeseries	The timeseries table from `r4ss::SS_output()`.
units_of_catch	From datalist, the catch units. A named list where the names are the fleets (to provide an extra check)

Value

a data frame with retained catch by Yr, Era, Seas, Fleet, and units (long format)

---

get_EM_catch_df *Get the EM catch data frame*

Description

Get the data frame of catch for the next iterations when using a Stock Synthesis Estimation model from the Report.sso file.

Usage

```
get_EM_catch_df(EM_dir, dat)
```

Arguments

EM_dir	Path to the EM files
dat	A SS datfile read into R using `r4ss::SS_readdat()`

Value

A data frame of future catch

Author(s)

Kathryn Doering

---

get_EM_dat *Change the OM data to match the format of the original EM data*

Description

This does the technical part of changing the EM data. Note this may be unnecessary

Usage

```
get_EM_dat(OM_dat, EM_dat, do_checks = TRUE)
```

Arguments

OM_dat	An SS data file read in by as a list read in using r4ss from the operating model
EM_dat	An SS data file read in by as a list read in using r4ss from the estimation model
do_checks	Should checks on the data be performed? Defaults to TRUE.

Value

A data list in the same format that can be read/written by r4ss that has index. lcomps, and age comps from OM_dat, but with the same structure as EM_dat.

Author(s)

Kathryn Doering

`get_F` *Get the Fishing mortality from the timeseries Report.sso table*

Description

Get the Fishing mortality from the timeseries Report.sso table

Usage

```
get_F(timeseries, fleetnames, fleetnames_all)
```

Arguments

`timeseries`	The timeseries table from `r4ss::SS_output()`.
`fleetnames`	A vector of fleet names, in the order they appear in the ss model.
`fleetnames_all`	A vector of ALL fleet names that are in the model in the order that they are specified in the model. This vector helps the function know which order the fleets appear in the model.

Value

a list containing: F_df, a long dataframe with F by Yr, Era, Seas, and fleet; F_rate, a data frame with F for the time frame of the model only by Yr, Seas, and Fleet, ordered as the ss.par file expects; init_F, a named vector of initial F values by Season and Fleet, ordered (and named) as SS expects; and F_rate_fcast, a dataframe of forecasted F by Yr, Seas, and fleet, ordered as SS would expect in F_rate.

get_full_sample_struct

Get the full sample structure from user input

Description

Get the full sample structure from user input by looking at the OM data. If it cannot be unambiguously determined, this function will return an error describing what additional user input is required.

Usage

```
get_full_sample_struct(sample_struct, OM_out_dir)
```

Arguments

sample_struct	A optional list including which years, seasons, and fleets should be added from the OM into the EM for different types of data. If NULL, the data structure will try to be inferred from the pattern found for each of the datatypes within the EM datafiles. Include this structure for the number of years to extend the model out. Note that the data should be specified using the list component names and column names as in would be used in `r4ss::SS_readdat()`. The `run_SSMSE_iter` function examples give an example of what this structure should be. Running the function `create_sample_struct()` will also produce a sample_struct object in the correct form. Can be NULL only when MS is not EM.
OM_out_dir	The full path to the directory in which the OM is run.

Value

A list of the full sample structure, using names as input by the user input by the user (not r4ss names).

get_impl_error_matrix *Put implementation error of 0 into a matrix*

Description

Put implementation error of 0 into a matrix

Usage

```
get_impl_error_matrix(yrs)
```

Arguments

yrs	a vector of years

Value

A length(yrs) row, 2 column matrix containing the years in the first column and the implementation error values in the second.

`get_init_samp_scheme` *Get the sampling scheme in a data file.*

Description

Determine what the default sampling scheme is for a given data file. Produces a list object with the sampling scheme, which can be modified, if desired.

Usage

```
get_init_samp_scheme(
  dat,
 dat_types = c("CPUE", "lencomp", "agecomp", "meanbodywt", "MeanSize_at_Age_obs")
)
```

Arguments

`dat`	An SS data file
`dat_types`	Types of data to include

`get_input_value` *return a value from a data frame*

Description

Return a single value from a column of a dataframe using the method specified

Usage

```
get_input_value(data, method = "most_common_value", colname, group = NULL)
```

Arguments

`data`	A dataframe which has a column that matches (at least partially) colname
`method`	How should the value to be returned be selected? Current options include "most_common_value", where the most common input uncertainty value will be returned and "only_value" where all input values must be the same in data; if they are, this value will be returned. Otherwise, an error will be generated.
`colname`	Column name as a string in `data`. Note that partial matching and regular expressions can be used.
`group`	Column name as a string in `data` used to group the data before calculating the input value to use. Defaults to NULL.

Details

Note that this function was created intially to return a value to use as the input uncertainty, but it should be generalizable to pulling a value from a column in any data frame using the method specified.

Value

A value of the same type as `data[, colname]` if `group` is NULL, or a data.frame if `group` is specified.

Author(s)

Kathryn Doering

Examples

```
dfr <- data.frame(
  "year" = 1:5,
  "value" = c(2, 2, 2, 3, 3),
  "se_log" = 0.2
)
SSMSE:::get_input_value(
  data = dfr, method = "most_common_value", colname = "se_log",
  group = "value"
)
SSMSE:::get_input_value(data = dfr, method = "most_common_value", colname = "value")
SSMSE:::get_input_value(data = dfr, method = "only_value", colname = "se_log")
# generates an error:
# SSMSE:::get_input_value(data = dfr, method = "only_value", colname = "value")
```

---

get_no_EM_catch_df *Get the data frame of catch for the next iterations when not using an estimation model.*

Description

Get the data frame of catch for the next iterations when not using an estimation model.

Usage

```
get_no_EM_catch_df(OM_dir, yrs, MS = "last_yr_catch")
```

Arguments

`OM_dir`	The full path to the OM directory.
`yrs`	A vector of years

MS — The management strategy to use. Current options are: "last_yr_catch" which uses the previous year's catch; "no_catch" which uses 0 catch; "EM" which uses an stock synthesis model as the estimation method and the management strategy as defined in the forecast file of the stock synthesis estimation method; "Interim" to modify catch based on survey predictions between assessments. Users can also specify their own management strategies as a function. For example, if the function is called "my_ms" then the user should specify MS = "my_ms" and specify the path to the file containing the function in custom_MS_source.

Value

A dataframe of future catch.

Author(s)

Kathryn Doering

---

get_performance_metrics

get basic data to calculate performance metrics

Description

get basic data to calculate performance metrics

Usage

```
get_performance_metrics(
  dir = getwd(),
  use_SSMSE_summary_all = TRUE,
  quantities = c("catch", "SpawnBio")
)
```

Arguments

dir — Path to the directory containing the scenarios, either relative or absolute. Defaults to the working directory.

use_SSMSE_summary_all — If it exists, should the summmary files generated by SSMSE_summary_all be used? Defaults to TRUE.

quantities — Quantites from the operating model to add

get_rel_SSB_avg *Example Performance Metric: Calculate the avg relative SSB (SSB/SSB unfished) over a range of years for each iteration*

Description

Example performance metric that calculates the average Spawning Stock Biomass SSB (units as in the simulations) relative to the unfished SSB over a range of years for each iteration of each scenario in the SSMSE simulation run.

Usage

```
get_rel_SSB_avg(summary, min_yr, max_yr)
```

Arguments

summary	Summary returned from running SSMSE_summary_all()
min_yr	The first year to include in the average
max_yr	The last year to include in the average

Value

A tibble containing the relative avg SSB per year by iteration and scenario.

Examples

```
## Not run:
rel_avg_ssb <- get_rel_SSB_avg(run_SSMSE_summary, min_yr = 10, max_yr = 105)
rel_avg_ssb

## End(Not run)
```

get_retained_catch *Get retained catch from the timeseries Report.sso table*

Description

Get retained catch from the timeseries Report.sso table

Usage

```
get_retained_catch(timeseries, units_of_catch)
```

Arguments

timeseries	The timeseries table from r4ss::SS_output().
units_of_catch	From datalist, the catch units. A named list where the names are the fleets (to provide an extra check)

Value

a data frame with retained catch by Yr, Era, Seas, Fleet, and units (long format)

---

get_SSB_avg *Example Performance Metric: calculate the average SSB over a range of years for each iteration*

Description

Example performance metric that calculates the average Spawning Stock Biomass SSB (units as in the simulations) over a range of years for each iteration of each scenario in the SSMSE simulation run.

Usage

```
get_SSB_avg(summary, min_yr, max_yr)
```

Arguments

summary	Summary returned from running SSMSE_summary_all()
min_yr	The first year to include in the average
max_yr	The last year to include in the average

Value

A tibble containing the average SSB by iteration and scenario.

Examples

```
## Not run:
avg_ssb <- get_SSB_avg(run_SSMSE_summary, min_yr = 10, max_yr = 105)
avg_ssb

## End(Not run)
```

get_total_catch *Example Performance Metric: Calculate total catch over a range of years*

Description

Example performance metric that calculates total catch over a range of years in a Stock Synthesis data file. This function aggregates catch across fleets, so may not be appropriate for models with multiple fleets.

Usage

```
get_total_catch(datfile, yrs)
```

Arguments

datfile	Path to the Stock Synthesis data file containing catch
yrs	A vector containing a range of years. Years are as defined in the Stock Synthesis data file.

Value

The total catch, a number.

Examples

```
## Not run:
total_catch <- get_total_catch(datfile = "ss3model/dat.ss", yrs = 25:100)
total_catch

## End(Not run)
```

Interim *Interim assessment management strategy*

Description

Interim assessment management strategy

Usage

```
Interim(
  EM_out_dir = NULL,
  EM_init_dir = NULL,
  init_loop = TRUE,
  OM_dat,
  OM_out_dir = NULL,
  verbose = FALSE,
  nyrs_assess,
  dat_yrs,
  future_om_list = NULL,
  sample_struct = NULL,
  interim_struct = NULL,
  seed = NULL,
  ...
)
```

Arguments

EM_out_dir	Relative or absolute path to the estimation model, if using a model outside of the SSMSE package.
EM_init_dir	Initialization director that retains the reference files for interim assessments
init_loop	Logical. If this is the first initialization loop of the MSE, init_loop should be TRUE. If it is in further loops, it should be FALSE.
OM_dat	An valid SS data file read in using r4ss. In particular, this should be sampled data.
OM_out_dir	The full path to the directory in which the OM is run.
verbose	Want verbose output? Defaults to FALSE.
nyrs_assess	The number of years between assessments. E.g., if an assessment is conducted every 3 years, put 3 here. A single integer value.
dat_yrs	Which years should be added to the new model? Ignored if init_loop is TRUE.
future_om_list	An optional list of lists including changes that should be made after the end year of the input model. Each first level list element outlines 1 change to be made to the operating model. To see an example, try running create_future_om_list. Defaults to NULL, which implies that the model will be extended forward in time assuming the original model structure.
sample_struct	A optional list including which years, seasons, and fleets should be added from the OM into the EM for different types of data. If NULL, the data structure will try to be inferred from the pattern found for each of the datatypes within the EM datafiles. Include this structure for the number of years to extend the model out. Note that the data should be specified using the list component names and column names as in would be used in r4ss::SS_readdat(). The run_SSMSE_iter function examples give an example of what this structure should be. Running the function create_sample_struct() will also produce a sample_struct object in the correct form. Can be NULL only when MS is not EM.

interim_struct	An optional including how many years to average over, fleet weights, the scaling rate (Beta) of catch relative to the index change for each fleet, and the reference year for each fleet (either a fixed year or <=0 relative to end_yr, fixed year will stay constant during simulation while relative year will progress with simulation).
seed	A random seed so that reproducible results are possible.
...	Any additional parameters

Author(s)

Nathan Vaughan

---

last_yr_catch *Last year catch used in the future for management strategy*

Description

Last year catch used in the future for management strategy

Usage

```
last_yr_catch(OM_out_dir, OM_dat, dat_yrs, ...)
```

Arguments

OM_out_dir	The full path to the directory in which the OM is run.
OM_dat	An valid SS data file read in using r4ss. In particular, this should be sampled data.
dat_yrs	Which years should be added to the new model? Ignored if init_loop is TRUE.
...	Any additional parameters

Author(s)

Kathryn Doering

locate_in_dirs *Locate the OM model files*

Description

Locate the OM model files

Usage

```
locate_in_dirs(OM_name = NULL, OM_in_dir = NULL)
```

Arguments

OM_name	Name of the operating model (OM).
OM_in_dir	Relative or absolute path to the operating model, if using a model outside of the SSMSE package. Should be a string.

Value

A list with on comonent, OM_in_dir, which contains the model location

match_parname *Match parameter name to parameter names in the par file*

Description

Match parameter name to parameter names in the par file

Usage

```
match_parname(list_pars, parlist)
```

Arguments

list_pars	the parameter names to find
parlist	A parameter file as read in by `r4ss::SS_readpar_3.30`

Value

A dataframe containing the parameter name and which object it is in in the par object.

Author(s)

Kathryn Doering

no_catch *No Catch in the future management strategy*

Description

No Catch in the future management strategy

Usage

```
no_catch(OM_out_dir, OM_dat, dat_yrs, ...)
```

Arguments

OM_out_dir	The full path to the directory in which the OM is run.
OM_dat	An valid SS data file read in using r4ss. In particular, this should be sampled data.
dat_yrs	Which years should be added to the new model? Ignored if init_loop is TRUE.
...	Any additional parameters

Author(s)

Kathryn Doering

parse_MS *Parse management strategy options*

Description

This function matches each management strategy with its correct method. And checks for errors.

Usage

```
parse_MS(
  MS,
  EM_out_dir = NULL,
  EM_init_dir = NULL,
  init_loop = TRUE,
  OM_dat,
  OM_out_dir = NULL,
  verbose = FALSE,
  nyrs_assess,
  dat_yrs,
  future_om_list = NULL,
  sample_struct = NULL,
  interim_struct = NULL,
  seed = NULL
)
```

Arguments

MS	The management strategy to use. Current options are: "last_yr_catch" which uses the previous year's catch; "no_catch" which uses 0 catch; "EM" which uses an stock synthesis model as the estimation method and the management strategy as defined in the forecast file of the stock synthesis estimation method; "Interim" to modify catch based on survey predictions between assessments. Users can also specify their own management strategies as a function. For example, if the function is called "my_ms" then the user should specify MS = "my_ms" and specify the path to the file containing the function in custom_MS_source.
EM_out_dir	Relative or absolute path to the estimation model, if using a model outside of the SSMSE package.
EM_init_dir	Initialization director that retains the reference files for interim assessments
init_loop	Logical. If this is the first initialization loop of the MSE, init_loop should be TRUE. If it is in further loops, it should be FALSE.
OM_dat	An valid SS data file read in using r4ss. In particular, this should be sampled data.
OM_out_dir	The full path to the directory in which the OM is run.
verbose	Want verbose output? Defaults to FALSE.
nyrs_assess	The number of years between assessments. E.g., if an assessment is conducted every 3 years, put 3 here. A single integer value.
dat_yrs	Which years should be added to the new model? Ignored if init_loop is TRUE.
future_om_list	An optional list of lists including changes that should be made after the end year of the input model. Each first level list element outlines 1 change to be made to the operating model. To see an example, try running create_future_om_list. Defaults to NULL, which implies that the model will be extended forward in time assuming the original model structure.
sample_struct	A optional list including which years, seasons, and fleets should be added from the OM into the EM for different types of data. If NULL, the data structure will try to be inferred from the pattern found for each of the datatypes within the EM datafiles. Include this structure for the number of years to extend the model out. Note that the data should be specified using the list component names and column names as in would be used in r4ss::SS_readdat(). The run_SSMSE_iter function examples give an example of what this structure should be. Running the function create_sample_struct() will also produce a sample_struct object in the correct form. Can be NULL only when MS is not EM.
interim_struct	An optional including how many years to average over, fleet weights, the scaling rate (Beta) of catch relative to the index change for each fleet, and the reference year for each fleet (either a fixed year or <=0 relative to end_yr, fixed year will stay constant during simulation while relative year will progress with simulation).
seed	A random seed so that reproducible results are possible.

Author(s)

Kathryn Doering & Nathan Vaughan

plot_comp_sampling *Plot comp data, expected values, and sampled data for 1 scenario*

Description

Creates a plot that can be used to see how sampling lines up with data and expected values for the index of abundance

Usage

```
plot_comp_sampling(dir = getwd(), comp_type = c("agecomp", "lencomp"))
```

Arguments

dir	Path to the directory containing 1 scenario. Defaults to the current working directory.
comp_type	Type of composition data, age or length. Defaults to age.

Value

A list containing 2 components: 1) the ggplot object and 2) the dataframe used to make the ggplot object

Author(s)

Kathryn Doering

plot_index_sampling *Plot index data, expected values, and sampled data for 1 scenario*

Description

Creates a plot that can be used to see how sampling lines up with data and expected values for the index of abundance

Usage

```
plot_index_sampling(dir = getwd())
```

Arguments

dir	Path to the directory containing 1 scenario. Defaults to the current working directory.

Value

A list containing 2 components: 1) the ggplot object and 2) the dataframe used to make the ggplot object

Author(s)

Kathryn Doering

---

r4ss_obj_err *Error if object is not an r4ss object*

Description

Error if object is not an r4ss object

Usage

```
r4ss_obj_err(obj_name = "object ", type = "list")
```

Arguments

obj_name	Object name that is not an r4ss object to print in the error
type	Type that obj_name was expected to be, but is not,

Author(s)

Kathryn Doering

---

rm_sample_struct_hist *Remove the historical sampling structure*

Description

Remove the historical sampling structure

Usage

```
rm_sample_struct_hist(sample_struct_hist, dat)
```

Arguments

sample_struct_hist	An optional list including which years should be sampled for the historical period for the data generated from the OM. If this is left as NULL, then the same sampling scheme will be used as in the OM's data file. If it is not NULL, then each year.
dat	The data file, as read in using r4ss

rm_vals *remove vals in 2 list components with the same name*

Description

From 2 list components with the same name, remove vals that aren't in the compare object

Usage

```
rm_vals(return_obj, compare_obj, name_in_obj, colnames)
```

Arguments

return_obj	the object (containing list component of name in obj) that will be modified. Only combinations of the columns found in compare object will be retained
compare_obj	the object (containing list component of name_in_obj) that return_obj will be compared to
name_in_obj	the name of the list elements to use; the same name must be in return_obj and compare_obj. This list element must be a data frame with the same column names
colnames	The column names within the name_in_obj list components to compare.

Value

return_obj[[name_in_obj]], modified to only include elements present in compare_obj[[name_in_obj]].

Author(s)

Kathryn Doering

run_EM *Run the estimation model*

Description

Runs the estimation model and performs checks if desired.

Usage

```
run_EM(
  EM_dir,
  hess = FALSE,
  check_converged = TRUE,
  set_use_par = FALSE,
  verbose = FALSE
)
```

Arguments

EM_dir	Absolute or relative path to the estimation model directory
hess	Get the hessian during model run? Defaults to FALSE. Not estimating the hessian will speed up the run, but no estimates of error will be generated.
check_converged	Perform checks to see if the model converged? Defaults to TRUE.
set_use_par	Should input values be read from the .par file? If TRUE, will change setting in the starter file; otherwise, will use the setting already in the starter file, which may or may not read from the .par file.
verbose	Want verbose output? Defaults to FALSE.

Author(s)

Kathryn Doering

---

run_OM *Initial run of the OM*

Description

This function is used to initialize the OM and get either expected values or bootstrap.

Usage

```
run_OM(
  OM_dir,
  boot = TRUE,
  nboot = 1,
  init_run = FALSE,
  verbose = FALSE,
  debug_par_run = TRUE,
  sample_catch = FALSE,
  seed = NULL
)
```

Arguments

OM_dir	The full path to the OM directory.
boot	Return the bootstrap dataset? If TRUE, function returns the number bootstrapped dataset specified in nboot. If FALSE, it returns the expected values.
nboot	The number bootstrapped data set. This value is only used if boot = TRUE. Note that this numbering does NOT correspond with the numbering in section of r4ss::SS_readdat. E.g., specifying section = 3 in SS_readdat is equivalent to specifying nboot = 1.
init_run	Is this the initial iteration of the OM? Defaults to FALSE.

verbose	Want verbose output? Defaults to FALSE.
debug_par_run	If set to TRUE, and the run fails, a new folder called error_check will be created, and the model will be run from control start values instead of ss.par. The 2 par files are then compared to help debug the issue with the model run. Defaults to TRUE.
sample_catch	Should catch be sampled or fixed at the OM values? Defaults to FALSE.
seed	A random seed so that reproducible results are possible.

Author(s)

Kathryn Doering

`run_SSMSE` *run an MSE using SS OMs*

Description

High level function to run a management strategy evaluation using Stock Synthesis as the Operating model(s). For more examples and information on how to use SSMSE, see the SSMSE user manual.

Usage

```
run_SSMSE(
  scen_name_vec,
  out_dir_scen_vec = NULL,
  iter_vec,
  OM_name_vec = NULL,
  OM_in_dir_vec = NULL,
  EM_name_vec = NULL,
  EM_in_dir_vec = NULL,
  run_EM_last_yr = FALSE,
  MS_vec = c("EM", "no_catch", "Interim"),
  custom_MS_source = NULL,
  use_SS_boot_vec = TRUE,
  nyrs_vec,
  nyrs_assess_vec,
  sample_struct_list = NULL,
  future_om_list = NULL,
  sample_struct_hist_list = NULL,
  sample_catch_vec = FALSE,
  interim_struct_list = NULL,
  verbose = FALSE,
  seed = NULL,
  n_F_search_loops = 20,
  tolerance_F_search = 0.001,
  run_parallel = FALSE,
  n_cores = NULL
)
```

Arguments

scen_name_vec	A vector containing names of the scenarios. The each string will be a directory containing all the model runs for a scenario.s
out_dir_scen_vec	The directory to which to write output. IF NULL, will default to the working directory.
iter_vec	The number of iterations per scenario. A vector of integers in the same order as scen_name_vec.
OM_name_vec	Names of a valid Stock Synthesis stock assessment model. To see the names of built-in models, type `list.dirs(system.file("extdata", "models", package = "SSMSE"),full.names = FALSE, recursive = FALSE)` into the R console.
OM_in_dir_vec	Vector of relative or absolute paths to the operating model, if using a model outside of the SSMSE package.
EM_name_vec	Should be NULL unless `MS = "EM"`. Name of a valid Stock Synthesis stock assessment model to use as an EM. If the value of EM_name is NULL and `MS = "EM"`, then SSMSE will look for the estimation model in the path specified in EM_in_dir. valid inputs for EM_name are: `"cod"` or `NULL`.
EM_in_dir_vec	Relative or absolute path to the estimation model, if using model outside of the SSMSE package. Note that this value should be NULL if `MS` has a value other than "EM".
run_EM_last_yr	Should the MS be implemented to get future catch if the last year is an assessment year? TRUE/FALSE option, so the same for all scenarios and iterations. Defaults to FALSE.
MS_vec	Vector of management strategies. Current options are: `"last_yr_catch"` which uses the previous year's catch; `"no_catch"` which uses 0 catch; `"EM"` which uses an stock synthesis model as the estimation method and the management strategy as defined in the forecast file of the stock synthesis estimation method; `"Interim"` to modify catch based on survey predictions between assessments. Users can also specify their own managment strategies as an function. To use the function, it must be available in the global enviroment and specified by name in MS. For example, if the function is called "my_ms" then the user should make it available in the global environment and specify "my_ms" as a component of MS_vec.
custom_MS_source	A file location with the source code for any custom MS models to be used in the simulation. SSMSE will source this file which should contain a function/s whose name/s match each custom MS models included in MS_vec. To learn more about using custom management strategies, see the using a Custom Management Stratey/Procedure section in the SSMSE User Manual.
use_SS_boot_vec	Should a bootstrapped data set generated by SS be used? Defaults to TRUE.
nyrs_vec	Number of years beyond the years included in the OM to run the MSE. A single integer value.
nyrs_assess_vec	The number of years between assessments. E.g., if an assessment is conducted every 3 years, put 3 here. A single integer value. (NOTE: This could be made

more flexible by instead reading in a vector of assessment years, so users could specify irregular numbers of yrs between assessments.)

`sample_struct_list`
: A optional list of lists including which years, seasons, and fleets should be added from the OM into the EM for different types of data. If NULL, the data structure will try to be infered from the pattern found for each of the datatypes within the EM datafiles. Include this strucutre for the number of years to extend the model out.

`future_om_list`
: An optional list of lists including changes that should be made after the end year of the input model. Each first level list element outlines 1 change to be made to the operating model. To see an example, try running `create_future_om_list`. Defaults to NULL, which implies that the model will be extended forward in time assuming the original model structure.

`sample_struct_hist_list`
: An optional list of lists including which years should be sampled for the historical period for the data generated from the OM. If this is left as NULL, then the same sampling scheme will be used as in the OM's data file. If it is not NULL, then each year

`sample_catch_vec`
: Should catch be sampled or fixed at the OM values? This can be a single Boolean (TRUE or FALSE) to apply to all scenarios or a vector of the same length as the number of scenarios. Defaults to FALSE.

`interim_struct_list`
: A optional list of parameters to control an interim assessment with an example structure below, where Beta=a positive value that is inversely proportional to risk, MA_years= the number of years to average index observations of when calculating deviations, assess_freq=the number of years between full assessments during with an interim assessment will happen every year, and Index_weights is a vector of length n indexes that weights all indexes for multi index inference. `interim_struct_list<-list(Beta=1,MA_years=3,assess_freq=5,Index_weights=rep(1,max(re`

`verbose`
: Want verbose output? Defaults to FALSE.

`seed`
: Input a fixed seed to replicate previous simulation runs. Seed can be a single value for a global seed, n_scenarios+1 length vector for scenario specific and a global seed, n_iterations+n_scenarios+1 length vector for iteration scenario and global seeds. Can also be a list object with a single value under `seed[["global"]]`, a vector under `seed[["scenario"]]`, and a multiple vectors for iteration specific seeds under `seed[["iter"]][[1:n_scenarios]]`.

`n_F_search_loops`
: Number of times to try to find an F that achieves the catches input in the OM. Defaults to 20.

`tolerance_F_search`
: How far apart the input catch and achieved catch can be in tried to find an F that achieves the catch input in the OM. Defaults to 0.001.

`run_parallel`
: Option to use parallel processing on iterations. Defaults to FALSE

`n_cores`
: how many cores to use if running in parallel defaults to n_cores available - 1 (also capped at one less than the number of cores available)

Author(s)

Kathryn Doering & Nathan Vaughan

Examples

```
## Not run:
my_dir <- file.path(tempdir(), "ex-run_SSMSE")
dir.create(my_dir)
# For the EM, use the specified data structure
my_sample_struct_list <- list(
  NULL,
  list(
    catch = data.frame(
      Yr = 101:106,
      Seas = 1,
      FltSvy = 1,
      SE = 0.05
    ),
    CPUE = data.frame(
      Yr = c(102, 105),
      Seas = 7,
      FltSvy = 2,
      SE = 0.01
    ),
    lencomp = data.frame(
      Yr = c(102, 105), Seas = 1,
      FltSvy = 1, Sex = 0,
      Part = 0, Nsamp = 100
    ),
  )
)
# Use the default parameter values, except for the once specified.
# Note that the scen_list, either specified or internally created in the
# function is returned.
input_list <- run_SSMSE(
  scen_name_vec = c("scen_1", "scen_2"),
  out_dir_scen_vec = my_dir,
  iter_vec = c(2, 2),
  OM_name_vec = c("cod", "cod"),
  OM_in_dir_vec = NULL,
  EM_name_vec = c(NA, "cod"),
  EM_in_dir_vec = NULL,
  MS_vec = c("no_catch", "EM"),
  use_SS_boot_vec = TRUE,
  nyrs_vec = 6,
  nyrs_assess_vec = 3,
  scope = c("2", "1", "3"),
  rec_dev_pattern = c(
    "none", "rand", "AutoCorr_rand",
    "AutoCorr_Spec", "vector"
  ),
  rec_dev_pars = NULL,
```

```
  impl_error_pattern = c("none", "rand", "user"),
  impl_error_pars = NULL,
  verbose = FALSE,
  seed = NULL,
  sample_struct_list = my_sample_struct_list
)
unlink(my_dir, recursive = TRUE)

## End(Not run)
```

`run_SSMSE_iter` *Run one iteration of an MSE using SS OM*

Description

High level function to run 1 iteration of a scenario for a management strategy evaluation using Stock Synthesis as the Operating model.

Usage

```
run_SSMSE_iter(
  out_dir = NULL,
  OM_name = "cod",
  OM_in_dir = NULL,
  EM_name = NULL,
  EM_in_dir = NULL,
  run_EM_last_yr = FALSE,
  MS = "last_yr_catch",
  custom_MS_source = NULL,
  use_SS_boot = TRUE,
  nyrs = 100,
  nyrs_assess = 3,
  nyrs_lag = 0,
  nscen = 1,
  scen_name = NULL,
  niter = 1,
  iter_seed = NULL,
  sample_struct = NULL,
  future_om_list = NULL,
  sample_struct_hist = NULL,
  sample_catch = FALSE,
  interim_struct = NULL,
  n_F_search_loops = 20,
  tolerance_F_search = 0.001,
  verbose = FALSE
)
```

Arguments

out_dir	The directory to which to write output. IF NULL, will default to the working directory.
OM_name	Name of the operating model (OM).
OM_in_dir	Relative or absolute path to the operating model, if using a model outside of the SSMSE package. Should be a string.
EM_name	Name of a valid Stock Synthesis stock assessment model to use as an EM. If the value of EM_name is NULL and MS = "EM", then SSMSE will look for the estimation model in the path specified in EM_in_dir. or NULL.
EM_in_dir	Relative or absolute path to the estimation model,
run_EM_last_yr	Should the MS be implemented to get future catch if the last year is an assessment year? TRUE/FALSE option, so the same for all scenarios and iterations. Defaults to FALSE.
MS	The management strategy to use. Current options are: "last_yr_catch" which uses the previous year's catch; "no_catch" which uses 0 catch; "EM" which uses an stock synthesis model as the estimation method and the management strategy as defined in the forecast file of the stock synthesis estimation method; "Interim" to modify catch based on survey predictions between assessments. Users can also specify their own management strategies as a function. For example, if the function is called "my_ms" then the user should specify MS = "my_ms" and specify the path to the file containing the function in custom_MS_source.
custom_MS_source	A file location with the source code for any custom MS models to be used in the simulation. SSMSE will source this file which should contain a function/s whose name/s match each custom MS models included in MS_vec. To learn more about using custom management strategies, see the using a Custom Management Stratey/Procedure section in the SSMSE User Manual.
use_SS_boot	Should a bootstrapped data set generated by SS be used? Defaults to TRUE.
nyrs	Number of years beyond the years included in the OM to run the MSE. A single integer value.
nyrs_assess	The number of years between assessments. E.g., if an assessment is conducted every 3 years, put 3 here. A single integer value. (NOTE: we could make this more flexible by instead reading in a vector of assessment years, so users could specify irregular numbers of yrs between assessments.)
nyrs_lag	number of years of lag in obtaining data. i.e. the number of years post EM assessment end yr before advice can be implemented. defaults to 0.
nscen	Which scenario is this. Integer value >=1
scen_name	Name of the scenario. The directory containing all the model runs the scenario will be stored within a folder of this name.
niter	The iteration number, which is also the name of the folder the results will be written to.
iter_seed	List containing fixed seeds for this iteration.

`sample_struct` A optional list including which years, seasons, and fleets should be added from the OM into the EM for different types of data. If NULL, the data structure will try to be inferred from the pattern found for each of the datatypes within the EM datafiles. Include this structure for the number of years to extend the model out. Note that the data should be specified using the list component names and column names as in would be used in `r4ss::SS_readdat()`. The `run_SSMSE_iter` function examples give an example of what this structure should be. Running the function `create_sample_struct()` will also produce a sample_struct object in the correct form. Can be NULL only when MS is not EM.

`future_om_list` An optional list of lists including changes that should be made after the end year of the input model. Each first level list element outlines 1 change to be made to the operating model. To see an example, try running `create_future_om_list`. Defaults to NULL, which implies that the model will be extended forward in time assuming the original model structure.

`sample_struct_hist`
An optional list including which years should be sampled for the historical period for the data generated from the OM. If this is left as NULL, then the same sampling scheme will be used as in the OM's data file. If it is not NULL, then each year.

`sample_catch` Should catch be sampled or fixed at the OM values? Defaults to FALSE.

`interim_struct` A optional list of parameters to control an interim assessment with an example structure below, where Beta=a positive value that is inversely proportional to risk, MA_years= the number of years to average index observations of when calculating deviations, assess_freq=the number of years between full assessments during with an interim assessment will happen every year, and Index_weights is a vector of length n indexes that weights all indexes for multi index inference.

```
interim_struct<-list(Beta = 1,
                     MA_years = 3,
                     assess_freq = 5,
                    Index_weights = rep(1, max(ref_index[, 3])))
```

`n_F_search_loops`
Number of times to try to find an F that achieves the catches input in the OM. Defaults to 20.

`tolerance_F_search`
How far apart the input catch and achieved catch can be in tried to find an F that achieves the catch input in the OM. Defaults to 0.001.

`verbose` Want verbose output? Defaults to FALSE.

Author(s)

Kathryn Doering & Nathan Vaughan

Examples

```
## Not run:
# Create a temporary folder for the output
temp_path <- file.path(tempdir(), "run_SSMSE_iter-ex")
```

```
dir.create(temp_path)

# run 1 iteration and 1 scenario of SSMSE
run_SSMSE_iter(
  OM_name = "cod",
  MS = "no_catch",
  out_dir = temp_path,
  nyrs = 6,
  nyrs_assess = 3
)
unlink(file.path(temp_path, "1"), recursive = TRUE)
# run 1 iteration and 1 scenario of SSMSE using an EM.
run_SSMSE_iter(
  OM_name = "cod",
  MS = "EM",
  out_dir = temp_path,
  EM_name = "cod",
  nyrs = 6,
  nyrs_assess = 3,
  sample_struct = list(
    catch = data.frame(Yr = 101:106, Seas = 1, FltSvy = 1, SE = 0.05),
    CPUE = data.frame(Yr = c(102, 105), Seas = 7, FltSvy = 2, SE = 0.01),
    lencomp = data.frame(
      Yr = c(102, 105), Seas = 1, FltSvy = 1,
      Sex = 0, Part = 0, Nsamp = 100
    ),
    agecomp = data.frame(
      Yr = c(102, 105), Seas = 1, FltSvy = 2,
      Sex = 0, Part = 0, Ageerr = 1,
      Lbin_lo = -1, Lbin_hi = -1, Nsamp = 50
    )
  )
)
unlink(temp_path, recursive = TRUE)

## End(Not run)
```

`run_SSMSE_scen` *Run an MSE scenario using SS OM*

Description

High level function to run 1 scenario (but potentially many iterations) for a management strategy evaluation using Stock Synthesis as the Operating Model

Usage

```
run_SSMSE_scen(
  scen_name = "scen_1",
  nscen = 1,
```

```
  out_dir_scen = NULL,
  iter = 2,
  OM_name = "cod",
  OM_in_dir = NULL,
  EM_name = NULL,
  EM_in_dir = NULL,
  run_EM_last_yr = FALSE,
  MS = "no_catch",
  custom_MS_source = NULL,
  use_SS_boot = TRUE,
  nyrs = 100,
  nyrs_assess = 3,
  scen_seed = NULL,
  sample_struct = NULL,
  future_om_list = NULL,
  sample_struct_hist = NULL,
  sample_catch = FALSE,
  interim_struct = NULL,
  verbose = FALSE,
  run_parallel = FALSE,
  n_cores = NULL,
  n_F_search_loops = 20,
  tolerance_F_search = 0.001
)
```

Arguments

scen_name	Name of the scenario. The directory containing all the model runs the scenario will be stored within a folder of this name.
nscen	Which scenario is this. Integer value >=1
out_dir_scen	The directory to which to write output. IF NULL, will
iter	The number of iterations for the scenario. A single integer value.
OM_name	Name of the operating model (OM).
OM_in_dir	Relative or absolute path to the operating model, if using a model outside of the SSMSE package. Should be a string.
EM_name	Name of a valid Stock Synthesis stock assessment model to use as an EM. If the value of EM_name is NULL and MS = "EM", then SSMSE will look for the estimation model in the path specified in EM_in_dir. or NULL.
EM_in_dir	Relative or absolute path to the estimation model,
run_EM_last_yr	Should the MS be implemented to get future catch if the last year is an assessment year? TRUE/FALSE option, so the same for all scenarios and iterations. Defaults to FALSE.
MS	The management strategy to use. Current options are: "last_yr_catch" which uses the previous year's catch; "no_catch" which uses 0 catch; "EM" which uses an stock synthesis model as the estimation method and the management strategy as defined in the forecast file of the stock synthesis estimation method;

"Interim" to modify catch based on survey predictions between assessments. Users can also specify their own management strategies as a function. For example, if the function is called "my_ms" then the user should specify MS = "my_ms" and specify the path to the file containing the function in custom_MS_source.

custom_MS_source
: A file location with the source code for any custom MS models to be used in the simulation. SSMSE will source this file which should contain a function/s whose name/s match each custom MS models included in MS_vec. To learn more about using custom management strategies, see the using a Custom Management Stratey/Procedure section in the SSMSE User Manual.

use_SS_boot
: Should a bootstrapped data set generated by SS be used? Defaults to TRUE.

nyrs
: Number of years beyond the years included in the OM to run the MSE. A single integer value.

nyrs_assess
: The number of years between assessments. E.g., if an assessment is conducted every 3 years, put 3 here. A single integer value. (NOTE: we could make this more flexible by instead reading in a vector of assessment years, so users could specify irregular numbers of yrs between assessments.)

scen_seed
: List containing fixed seeds for this scenario and its iterations.

sample_struct
: A optional list including which years, seasons, and fleets should be added from the OM into the EM for different types of data. If NULL, the data structure will try to be inferred from the pattern found for each of the datatypes within the EM datafiles. Include this structure for the number of years to extend the model out. Note that the data should be specified using the list component names and column names as in would be used in r4ss::SS_readdat(). The run_SSMSE_iter function examples give an example of what this structure should be. Running the function create_sample_struct() will also produce a sample_struct object in the correct form. Can be NULL only when MS is not EM.

future_om_list
: An optional list of lists including changes that should be made after the end year of the input model. Each first level list element outlines 1 change to be made to the operating model. To see an example, try running create_future_om_list. Defaults to NULL, which implies that the model will be extended forward in time assuming the original model structure.

sample_struct_hist
: An optional list including which years should be sampled for the historical period for the data generated from the OM. If this is left as NULL, then the same sampling scheme will be used as in the OM's data file. If it is not NULL, then each year.

sample_catch
: Should catch be sampled or fixed at the OM values? Defaults to FALSE.

interim_struct
: A optional list of parameters to control an interim assessment with an example structure below, where Beta=a positive value that is inversely proportional to risk, MA_years= the number of years to average index observations of when calculating deviations, assess_freq=the number of years between full assessments during with an interim assessment will happen every year, and Index_weights is a vector of length n indexes that weights all indexes for multi index inference.

```
interim_struct<-list(Beta = 1,
```

```
                                        MA_years = 3,
                                        assess_freq = 5,
                                      Index_weights = rep(1, max(ref_index[, 3])))
```

verbose — Want verbose output? Defaults to FALSE.

run_parallel — Option to use parallel processing on iterations. Defaults to FALSE

n_cores — how many cores to use if running in parallel defaults to n_cores available - 1 (also capped at one less than the number of cores available)

n_F_search_loops — Number of times to try to find an F that achieves the catches input in the OM. Defaults to 20.

tolerance_F_search — How far apart the input catch and achieved catch can be in tried to find an F that achieves the catch input in the OM. Defaults to 0.001.

Author(s)

Kathryn Doering & Nathan Vaughan

Examples

```
## Not run:
# Create a temporary folder for the output and set the working directory:
temp_path <- file.path(tempdir(), "run_SSMSE_scen-example")
dir.create(temp_path, showWarnings = FALSE)

# run 2 iteration and 1 scenario of SSMSE
run_SSMSE_scen(
  scen_name = "no_catch",
  iter = 1:2,
  OM_name = "cod",
  MS = "no_catch",
  out_dir_scen = temp_path,
  nyrs = 6,
  nyrs_assess = 3
)
unlink(temp_path, recursive = TRUE)

## End(Not run)
```

---

run_ss_model *Run an operating or estimation model*

Description

This function takes care of calling SS. Importantly, it parses whether the user is on Unix or Windows and calls the binary correctly. This lower-level function is meant to be called by higher level functions. Modified from run_ss3model in ss3sim.

Usage

```
run_ss_model(
  dir,
  admb_options = "",
  ss_bin = "ss",
  ignore.stdout = TRUE,
  admb_pause = 0.05,
  show.output.on.console = FALSE,
  check_run = TRUE,
  debug_par_run = FALSE,
  verbose = FALSE,
  ...
)
```

Arguments

dir	The full or relative path to the model directory
admb_options	Any options to pass to SS command. Should be of the form '-option'. Note that no checks are done to ensure this is a valid ADMB command
ss_bin	Name of the SS executable. Defaults to "ss"
ignore.stdout	Passed to system. If TRUE then ADMB output is not printed on screen. This will be slightly faster. Set to FALSE to help with debugging.
admb_pause	A length of time (in seconds) to pause after running the simulation model. This can be necessary on certain computers where file writing can be slightly delayed. For example, on computers where the files are written over a network connection. If the output files haven't finished writing before R starts looking for the output then the simulation will crash with an error about missing files. The default value is set to 0.01 seconds, just to be safe.
show.output.on.console	Logical: passed on to system.
check_run	Should it be checked that the model ran by deleting the data.ss_new file if one exists and then checking if one was created? Defaults to TRUE.
debug_par_run	If set to TRUE, and the run fails, a new folder called error_check will be created, and the model will be run from control start values instead of ss.par. The 2 par files are then compared to help debug the issue with the model run. Defaults to FALSE.
verbose	Want verbose output? Defaults to FALSE.
...	Anything else to pass to system.

Author(s)

Sean C. Anderson, Kathryn Doering

sample_vals — *Sample vals from normal random, lognormal random, or modified AR-1 process.*

Description

Sample vals from normal random, lognormal random, or modified AR-1 process.

Usage

```
sample_vals(mean, sd, ar_1_phi = 0, ndevs, dist = c("normal", "lognormal"))
```

Arguments

mean	A single value or vector of mean parameters
sd	A single value or vector of sd parameter
ar_1_phi	The phi (coefficient) value for an ar 1 process. Should be between -1 and 1. 0 means an AR 1 process will NOT be used. 1 indicates a random walk. A single value or vector.
ndevs	The number of sampled values to expect
dist	The distribution to sample from.

Author(s)

Kathryn Doering

set_MSE_seeds — *Set the initial global, scenario, and iteration seeds*

Description

Set the initial global, scenario, and iteration seeds

Usage

```
set_MSE_seeds(seed = NULL, iter_vec)
```

Arguments

seed	Input a fixed seed to replicate previous simulation runs. Seed can be a single value for a global seed, n_scenarios+1 length vector for scenario specific and a global seed, n_iterations+n_scenarios+1 length vector for iteration scenario and global seeds. Can also be a list object with a single value under `seed[["global"]]`, a vector under `seed[["scenario"]]`, and a multiple vectors for iteration specific seeds under `seed[["iter"]][[1:n_scenarios]]`.
iter_vec	The number of iterations per scenario. A vector of integers in the same order as scen_name_vec.

Value

A list of length 3 with 1) the global seed value; 2) the scenario seed values; and 3) the iteration seed values.

Examples

```
seeds <- set_MSE_seeds(seed = seq(10, 80, by = 10), iter_vec = c(2, 3))
```

Sim_comp *Calculate uncertainty and biases in historic composition data*

Description

Calculate uncertainty and biases in historic composition data

Usage

```
Sim_comp(
  Comp_uncert,
  data_exp,
  bins,
  years = NULL,
  seasons = NULL,
  fleets = NULL,
  genders = NULL
)
```

Arguments

Comp_uncert	A list object representing the output from the calc_comp_var function
data_exp	A vector representing the expected composition values for which to draw a random observation dataset
bins	A vector object including the composition bins
years	A vector of the years to simulate data for. default is all years in data_exp if NULL.
seasons	A vector of the seasons to simulate data for. default is all seasons in data_exp if NULL.
fleets	A vector of the fleets to simulate data for. default is all fleets in data_exp if NULL.
genders	A vector of the genders to simulate data for. default is all genders in data_exp if NULL.

Value

A list object with uncertainty and bias characteristics to inform data simulation.

Author(s)

Nathan R. Vaughan

`SSMSE` *SSMSE: A package for Management Strategy Evaluation (MSE) using Stock Synthesis (SS)*

Description

SSMSE is an R package for performing Management Strategy Evaluation (MSE) using Stock Synthesis (SS). SS is used as the Operating Model (OM) and, if the user desires, the Estimation model (EM). SSMSE allows existing SS models to be used as the basis for an OM. These OMs are used in the MSE framework provided by SSMSE to evaluate the implications of management actions on a population given uncertainty.

Author(s)

Maintainer: Kathryn Doering <kathryn.doering@noaa.gov>

Authors:

- Nathan Vaughan <nathan.vaughan@noaa.gov>

See Also

Useful links:

- https://github.com/nmfs-fish-tools/SSMSE
- Report bugs at https://github.com/nmfs-fish-tools/SSMSE/issues

---

`SSMSE_summary_all` *Get results in a list for 1 scenario*

Description

Get results in a list for 1 iteration, using ss3sim::get_results_iter

Usage

```
SSMSE_summary_all(
  dir = getwd(),
  scenarios = NULL,
  run_parallel = FALSE,
  n_cores = NULL,
  overwrite = FALSE
)
```

Arguments

dir	Path to the directory containing the scenarios, either relative or absolute. Defaults to the working directory.
scenarios	A character vector of scenarios in dir from which to extract summaries. If left as NULL, the summaries will be extracted from all folders in dir.
run_parallel	Option to use parallel processing on iterations. Defaults to FALSE
n_cores	how many cores to use if running in parallel defaults to n_cores available - 1 (also capped at one less than the number of cores available)
overwrite	Allow existing files to be overwritten?

Value

A list of 3 data frames called scalar, ts, and dq (for derived quantities). These lists contain information for multiple model runs (estimation models and operating models) for 1 iteration.Also writes 3 .csv files with the contents of this list of dataframes to dir and 3.csv files with scenario specific results in each of the scenario foldurs..

See Also

get_results_all

SSMSE_summary_iter *Get results in a list for 1 iteration*

Description

Get results in a list for 1 iteration, using ss3sim::get_results_iter

Usage

```
SSMSE_summary_iter(dir)
```

Arguments

dir	Path to the directory for 1 iteration of 1 scenario.

Value

A list of 3 data frames called scalar, timeseries, and derived (for derived quantities). These lists contain information for multiple model runs (estimation models and operating models) for 1 iteration.

See Also

get_results_iter

SSMSE_summary_scen *Get results in a list for 1 scenario*

Description

Get results in a list for 1 iteration, using ss3sim::get_results_iter

Usage

```
SSMSE_summary_scen(dir = getwd())
```

Arguments

dir	Path to the directory for 1 scenario, either relative or absolute. Defaults to the working directory.

Value

A list of 3 data frames called scalar, ts, and dq (for derived quantities). These lists contain information for multiple model runs (estimation models and operating models) for 1 iteration.Also writes 3 .csv files with the contents of this list of dataframes to dir.

See Also

get_results_scenario

test_no_par *Change a model from running with par to running without par*

Description

The intention of this function is to help troubleshooting issues with the par file. It is intended mostly to help troubleshooting while developing the SSMSE package, but may also be helpful with runtime testing.

Usage

```
test_no_par(orig_mod_dir, new_mod_dir)
```

Arguments

orig_mod_dir	The original model directory
new_mod_dir	The new model directory (folder need not exist)

update_basevals_blocks *Update a sequence of base parameter annual values to account for a time varying block effects*

Description

Update a sequence of base parameter annual values to account for a time varying block effects

Usage

```
update_basevals_blocks(
  base_vals,
  base_years,
  temp_block,
  current_par,
  ctl,
  dat,
  temp_ctl,
  base_range,
  baseparm,
  base_bounds
)
```

Arguments

base_vals	A vector of base parameter values that will be updated to include the impact of a time varying block change
base_years	A vector of years for which the base values are needed
temp_block	The timevarying parameter lines for the block effects on the base parameter
current_par	The index of the current parameter being updated
ctl	A control file as read in by r4ss::SS_readctl.
dat	A data file as read in by r4ss::SS_readdat.
temp_ctl	A subset of the control file representing the parameter section of interest (i.e. MG, SR, Q, or Selectivity)
base_range	the difference between the base parameters max and min bounds
baseparm	The value of the base parameter
base_bounds	The min and max bounds of the base parameter

Value

A modified parameter value series that incorporates the appropriate time varying block effects.

Author(s)

Nathan Vaughan

`update_basevals_dev` *Update a sequence of base parameter annual values to account for a time varying deviation effects*

Description

Update a sequence of base parameter annual values to account for a time varying deviation effects

Usage

```
update_basevals_dev(
  base_vals,
  temp_dev,
  dev_seq,
  current_par,
  temp_ctl,
  base_range,
  base_bounds
)
```

Arguments

`base_vals`	A vector of base parameter values that will be updated to include the impact of a time varying deviations
`temp_dev`	The time varying parameter lines for the deviations on the base parameter
`dev_seq`	A vector of the parameter deviations to be applied to the base values
`current_par`	The index of the current parameter being updated
`temp_ctl`	A subset of the control file representing the parameter section of interest (i.e. MG, SR, Q, or Selectivity)
`base_range`	the difference between the base parameters max and min bounds
`base_bounds`	The min and max bounds of the base parameter

Value

A modified parameter series that incorporates the appropriate deviations.

Author(s)

Nathan Vaughan

`update_basevals_env` *Update a sequence of base parameter annual values to account for a time varying environmental effects*

Description

Update a sequence of base parameter annual values to account for a time varying environmental effects

Usage

```
update_basevals_env(
  base_vals,
  base_years,
  temp_env,
  current_par,
  timeseries,
  temp_ctl,
  dat,
  base_range,
  base_bounds,
  parlist
)
```

Arguments

`base_vals`	A vector of base parameter values that will be updated to include the impact of a time varying environmental effects
`base_years`	A vector of years for which the base values are needed
`temp_env`	The time varying parameter lines for the environmental effects on the base parameter
`current_par`	The index of the current parameter being updated
`timeseries`	The timeseries table from `r4ss::SS_output()`.
`temp_ctl`	A subset of the control file representing the parameter section of interest (i.e. MG, SR, Q, or Selectivity)
`dat`	A datafile as read in by r4ss::SS_readdat
`base_range`	the difference between the base parameters max and min bounds
`base_bounds`	The min and max bounds of the base parameter
`parlist`	A parameter file as read in by `r4ss::SS_readpar_3.30`

Value

A modified parameter series that incorporates the appropriate time varying environmental effects.

Author(s)

Nathan Vaughan

`update_OM` *Extend the OM forward using next years' catch*

Description

Add the EM defined catch values for the next years.

Usage

```
update_OM(
  OM_dir,
  catch = NULL,
  harvest_rate = NULL,
  catch_basis = NULL,
  F_limit = NULL,
  EM_pars = NULL,
  write_dat = TRUE,
  impl_error = NULL,
  verbose = FALSE,
  seed = NULL,
  n_F_search_loops = 20,
  tolerance_F_search = 0.001
)
```

Arguments

`OM_dir`	The full path to the OM directory.
`catch`	A dataframe of catch values and its associated information to add to the OM. The column names are the same as in an SS data file (e.g., year, season, fleet, catch, catch_se). Must input either a catch and/or a harvest rate data frame. If both are input the catch will override harvest rate as the management unit but harvest rate will be used as a starting guess for search.
`harvest_rate`	A dataframe of harvest rate (F) values and associated information to add to the OM. The column names are as in an SS datafile. If harvest rate is input without a corresponding catch the OM will assume effort based management an use harvest rate directly with implementation error added.
`catch_basis`	data frame with columns year, seas, fleet, basis that specifies if catch should reference retained biomass (1) or dead biomass (2). Any year/season/fleet not listed will assume a value of 1 referencing retained biomass. Entering -99 for any of year, season, or fleet will apply the basis across all values of that variable (i.e. a single row with -99, -99, -99, 1 would implement retained biomass for all cases)

F_limit — data frame with columns year, fleet, season, limit that specifies a maximum F allowed in the OM or a negative value to specify a multiple of the historic maximum F. Any year/season/fleet not listed will assume a value of 1.5. Entering -99 for any of year, season, or fleet will apply the limit across all values of that variable (i.e. a single row with -99, -99, -99, -2 would implement a cap of twice the historic maximum F for all cases)

EM_pars — a dataframe of parameter value updates to modify OM

write_dat — Should the datafile be overwritten? Defaults to TRUE.

impl_error — The implementation error

verbose — Want verbose output? Defaults to FALSE.

seed — A random seed so that reproducible results are possible.

n_F_search_loops — Number of times to try to find an F that achieves the catches input in the OM. Defaults to 20.

tolerance_F_search — How far apart the input catch and achieved catch can be in tried to find an F that achieves the catch input in the OM. Defaults to 0.001.

Value

A new dat list object (format as created by r4ss::SS_readdat) that has been extended forward as if read in by r4ss function SS_readdat

Author(s)

Kathryn Doering & Nathan Vaughan

Index