

Package: PEcAn.benchmark (via r-universe)

October 16, 2026

Type Package

Title PEcAn Functions Used for Benchmarking

Version 1.7.5.9000

Author Michael Dietze, Betsy Cowdery, David LeBauer, Rob Kooper, Toni Viskari

Maintainer Mike Dietze <dietze@bu.edu>

Description The Predictive Ecosystem Carbon Analyzer (PEcAn) is a scientific workflow management tool that is designed to simplify the management of model parameterization, execution, and analysis. The goal of PEcAn is to streamline the interaction between data and models, and to improve the efficacy of scientific investigation. The PEcAn.benchmark package provides utilities for comparing models and data, including a suite of statistical metrics and plots.

URL <https://pecanproject.github.io>

BugReports <https://github.com/PecanProject/pecan/issues>

Imports dplyr, ggplot2, gridExtra, lubridate (>= 1.6.0), magrittr, ncd4 (>= 1.15), PEcAn.DB, PEcAn.logger, PEcAn.settings, PEcAn.utils, reshape2, rlang, SimilarityMeasures, stringr, tidyr, tidyselect, units, utils, grDevices, XML (>= 3.98-1.4), zoo, yaml

Suggests knitr, quarto, rmarkdown, scoringRules, PEcAn.data.land, testthat (>= 2.0.0)

VignetteBuilder knitr

License BSD_3_clause + file LICENSE

Copyright Authors

LazyLoad yes

LazyData FALSE

Encoding UTF-8

RoxygenNote 7.3.3

X-schema.org-keywords model-evaluation, model-data-comparison, PEcAn

Config/pak/sysreqs

cmake make libicu-dev libuv1-dev libxml2-dev libnetcdf-dev libssl-dev libudunits2-dev

Repository <https://pecanproject.r-universe.dev>

Date/Publication 2026-10-16 08:17:39 UTC

RemoteUrl <https://github.com/PecanProject/pecan>

RemoteRef HEAD

RemoteSha fdb9b89d6133f78473728f63b7c1b3deb4ee29ec

RemoteSubdir modules/benchmark

Contents

add_workflow_info	3
align_by_first_observation	4
align_by_time	5
align_data	5
align_data_to_data_pft	6
align_pft	8
bm_settings2pecan_settings	10
bm_validate	10
calc_benchmark	11
calc_metrics	11
check_BRR	12
check_if_legal_table	12
check_if_list_of_pfts	13
check_if_species_list	13
clean_settings_BRR	14
compute_metrics	14
create_BRR	15
define_benchmark	15
efi_long_to_array	16
format_wide2long	16
generate_validation_report	17
get_species_list_standard	18
load_and_map_data	18
load_csv	19
load_data	19
load_rds	20
load_tab_separated_values	21
load_x_netcdf	21
match_timestep	22
mean_over_larger_timestep	22
metric_AME	23
metric_Bias	23
metric_cor	24

<i>add_workflow_info</i>	3
metric_Coverage	24
metric_CRPS	25
metric_Frechet	25
metric_lmDiag_plot	26
metric_MAE	26
metric_MSE	27
metric_PMU	27
metric_PPMC	28
metric_R2	28
metric_RAE	29
metric_residual_plot	29
metric_RMSE	30
metric_run	30
metric_scatter_plot	31
metric_timeseries_plot	31
pecan_metric_registry	32
read_settings_BRR	32
register_metric	33
run_benchmark	33
Index	35

<code>add_workflow_info</code>	<i>Add workflow specific info to settings list for benchmarking</i>
--------------------------------	---

Description

Add workflow specific info to settings list for benchmarking

Usage

`add_workflow_info(settings, bety)`

Arguments

- | | |
|-----------------------|----------------------------------|
| <code>settings</code> | settings or multisettings object |
| <code>bety</code> | connection to the database |

Author(s)

Betsy Cowdery

```
align_by_first_observation
      align_first_observation
```

Description

align_first_observation

Usage

```
align_by_first_observation(observation_one, observation_two, custom_table)
```

Arguments

observation_one
a vector of plant functional types, or species. Provides species/pft names.

observation_two
another vector of plant functional types, or species. Provides the order.

custom_table
a table that either maps two pft's to one another or maps custom species codes to bety id codes. In the second case, must be passable to match_species_id.

Value

vector Returns a vector of PFT's/species from observation_one that matches the order of observation_two

Author(s)

Tempest McCabe

Examples

```
observation_one<-c("AMCA3", "AMCA3", "AMCA3", "AMCA3")
observation_two<-c("a", "b", "a", "a")

table<-list()
table$plant_functional_type_one<- c("AMCA3", "AMCA3", "ARHY", "ARHY")
table$plant_functional_type_two<- c('a','a','b', 'b') # PFT groupings
table<-as.data.frame(table)

aligned <- align_by_first_observation(
  observation_one = observation_one,
  observation_two = observation_two,
  custom_table = table)

# aligned should be a vector '[1] "AMCA3" "ARHY" "AMCA3" "AMCA3"'
```

align_by_time	<i>Align model predictions and observations by time</i>
---------------	---

Description

For each observation in 'obs_df', finds the nearest model prediction in 'model_df' within 'tolerance_secs'. Guarantees that each observation is paired with at most one model prediction.

Usage

```
align_by_time(model_df, obs_df, tolerance_secs = 3600)
```

Arguments

model_df	data.frame containing model predictions with a 'time' column (POSIXct) and 'value' or 'model' column.
obs_df	data.frame containing observations with a 'time' column (POSIXct) and 'value' or 'obvs' column.
tolerance_secs	maximum allowable time difference in seconds between paired observation and model prediction.

Value

data.frame with aligned rows containing model and observation columns, where each observation corresponds to exactly one model prediction.

align_data	<i>Align timeseries data</i>
------------	------------------------------

Description

Align timeseries data

Usage

```
align_data(model.calc, obvs.calc, var, align_method = "match_timestep")
```

Arguments

model.calc	data.frame
obvs.calc	data.frame
var	data.frame
align_method	name of function to use for alignment

Value

dat

Author(s)

Betsy Cowdery

align_data_to_data_pft	<i>align_data_to_data_pft</i>
------------------------	-------------------------------

Description

align_data_to_data_pft

Usage

```
align_data_to_data_pft(  
  con,  
  observation_one,  
  observation_two,  
  custom_table = NULL,  
  format_one,  
  format_two,  
  subset_is_ok = FALSE  
)
```

Arguments

con	database connection
observation_one	a vector of plant functional types, or species
observation_two	another vector of plant functional types, or species
custom_table	a table that either maps two pft's to one another or maps custom species codes to bety id codes. In the second case, must be passable to match_species_id.
format_one	The output of query.format.vars() of observation one of the form output\$vars\$bety_names
format_two	The output of query.format.vars() of observation two of the form output\$vars\$bety_names
subset_is_ok	When aligning two species lists, this allows for alignment when species lists aren't identical. set to FALSE by default.

Details

Aligns vectors of Plant Functional Type and species. Can align: - two vectors of plant functional types (pft's) if a custom map is provided - a list of species (usda, fia, or latin_name format) to a plant functional type - a list of species in a custom format, with a table mapping it to bety_species_id's

Will return a list of what was originally provided, bety_species_codes if possible, and an aligned output. Because some alignment is order-sensitive, alignment based on observation_one and observation_two are both provided.

Value

list containing the following columns:

`$original` Will spit back out original vectors pre-alignment

`$aligned$aligned_by_observation_one` Where possible, will return a vector of observation_one pft's/species in the order of observation_two

`species` Where possible, will return a vector of observation_two's pft's/species in the order of observation_one

`$bety_species_id` Where possible, will return the bety_species_id's for one or both observations

`$bety_species_intersection` Where possible, will return the intersection of two aligned lists of species. `subset_is_ok` must be set to TRUE.

Author(s)

Tempest McCabe

Examples

Not run:

```
con <- db.open(list(host = "postgres", user = "bety", password = "carya"))
```

```
observation_one <- c("AMCA3", "AMCA3", "AMCA3", "AMCA3")
```

```
observation_two <- c("a", "b", "a", "a")
```

```
table <- list()
```

```
table$plant_functional_type_one <- c("AMCA3", "AMCA3", "ARHY", "ARHY")
```

```
table$plant_functional_type_two <- c('a', 'a', 'b', 'b') # PFT groupings
```

```
table <- as.data.frame(table)
```

```
format_one <- "species_USDA_symbol"
```

```
format_two <- "plant_functional_type"
```

```
aligned <- align_data_to_data_pft(
```

```
  con = con,
```

```
  observation_one = observation_one, observation_two = observation_two,
```

```
  format_one = format_one, format_two = format_two,
```

```
  custom_table = table)
```

End(Not run)

align_pft

*Align vectors of Plant Functional Type and species.***Description**

Align vectors of Plant Functional Type and species.

Usage

```
align_pft(
  con,
  observation_one,
  observation_two,
  custom_table = NULL,
  format_one,
  format_two,
  subset_is_ok = FALSE,
  comparison_type = "data_to_data",
  ...
)
```

Arguments

con	database connection
observation_one	a vector of plant functional types, or species
observation_two	another vector of plant functional types, or species
custom_table	a table that either maps two pft's to one another or maps custom species codes to bety id codes. In the second case, must be passable to match_species_id.
format_one	The output of query.format.vars() of observation one of the form output\$vars\$bety_names
format_two	The output of query.format.vars() of observation two of the form output\$vars\$bety_names
subset_is_ok	When aligning two species lists, this allows for alignment when species lists aren't identical. set to FALSE by default.
comparison_type	one of "data_to_model", "data_to_data", or "model_to_model"
...	other arguments, currently ignored

Details

Can align: - two vectors of plant functional types (pft's) if a custom map is provided - a list of species (usda, fia, or latin_name format) to a plant functional type - a list of species in a custom format, with a table mapping it to bety_species_id's

Will return a list of what was originally provided, bety_species_codes if possible, and an aligned output. Because some alignment is order-sensitive, alignment based on observation_one and observation_two are both provided.

comparison_type can be one of the following:

data_to_data Will align lists of pfts and species. Must be associated with inputs.

data_to_model Not yet implemented

model_to_model Not yet implemented

Value

list containing the following columns:

\$original Will spit back out original vectors pre-alignment

\$aligned\$aligned_by_observation_one Where possible, will return a vector of observation_one pft's/species in the order of observation_two

species Where possible, will return a vector of observation_two's pft's/species in the order of observation_one

\$bety_species_id Where possible, will return the bety_species_id's for one or both observations

Author(s)

Tempest McCabe

Examples

```
## Not run:
```

```
con <- db.open(list(host = "postgres", user = "bety", password = "carya"))
```

```
#----- A species to PFT alignment -----
```

```
observation_one <- c("AMCA3", "AMCA3", "AMCA3", "AMCA3")
```

```
observation_two <- c("a", "b", "a", "a")
```

```
format_one <- "species_USDA_symbol"
```

```
format_two <- "plant_functional_type"
```

```
table <- list()
```

```
table$plant_functional_type_one <- c("AMCA3", "AMCA3", "ARHY", "ARHY")
```

```
table$plant_functional_type_two <- c('a', 'a', 'b', 'b') # PFT groupings
```

```
table <- as.data.frame(table)
```

```
aligned <- align_pft(
```

```
  con = con,
```

```
  observation_one = observation_one, observation_two = observation_two,
```

```
  custom_table = table,
```

```
  format_one = format_one, format_two = format_two)
```

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

bm_settings2pecan_settings

Move benchmarking settings back in to original pecan settings object

Description

Move benchmarking settings back in to original pecan settings object

Usage

```
bm_settings2pecan_settings(bm.settings)
```

Arguments

bm.settings settings or multisettings object

Author(s)

Betsy Cowdery

bm_validate

Validate benchmark input dataframes

Description

Validate benchmark input dataframes

Usage

```
bm_validate(model_df, obs_df)
```

Arguments

model_df data.frame with columns: time (POSIXct), value (numeric)

obs_df data.frame with columns: time (POSIXct), value (numeric)

Value

invisible(TRUE)

calc_benchmark	<i>Calculate benchmarking statistics</i>
----------------	--

Description

For each benchmark id, calculate metrics and update benchmarks_ensemble_scores

Usage

```
calc_benchmark(settings, bety, start_year = NA, end_year = NA)
```

Arguments

settings	settings object describing the run to calculate
bety	database connection
start_year, end_year	time range to read. If NA, these are taken from 'settings'

Author(s)

Betsy Cowdery

calc_metrics	<i>calc_metrics</i>
--------------	---------------------

Description

calc_metrics

Usage

```
calc_metrics(model.calc, obsv.calc, var, metrics, ensemble.id, bm_dir)
```

Arguments

model.calc	model data
obvs.calc	observational data
var	variables to be used
metrics	metrics to be used
ensemble.id	id of ensemble run
bm_dir	directory where benchmarking outputs will be saved

Author(s)

Betsy Cowdery

 check_BRR

Check whether a run has been registered as a reference run in BETY

Description

Check whether a run has been registered as a reference run in BETY

Usage

```
check_BRR(settings_xml, con)
```

Arguments

settings_xml	cleaned settings to be compared with BRR in the database
con	database connection

Author(s)

Betsy Cowdery

 check_if_legal_table *check_if_legal_table*

Description

check_if_legal_table

Usage

```
check_if_legal_table(table, observation_one, observation_two)
```

Arguments

table	a table that either maps two pft's to one another or maps custom species codes to bety id codes. In the second case, must be passable to match_species_id.
observation_one	a vector of plant functional types, or species
observation_two	another vector of plant functional types, or species

Details

Checks if custom_table: 1. is formatted correctly 2. is complete (has all of the species/pft's in both observations) 3. is condense-able (Could be represented as a hierachry)

Value

boolean

Author(s)

Tempest McCabe

check_if_list_of_pfts	<i>check_if_list_of_pfts</i>
-----------------------	------------------------------

Description

Checks if format contains a variable named "plant_functional_type"

Usage

check_if_list_of_pfts(vars)

Arguments

vars	names to check
------	----------------

Value

boolean

Author(s)

Tempest McCabe

check_if_species_list	<i>check_if_species_list</i>
-----------------------	------------------------------

Description

check_if_species_list

Usage

check_if_species_list(vars, custom_table = NULL)

Arguments

vars	format
custom_table	a table that either maps two pft's to one another or maps custom species codes to bety id codes. In the second case, must be passable to match_species_id.

Details

Checks if format contains a species list in a known format, or a declared custom format.

Value

boolean

Author(s)

Tempest McCabe

clean_settings_BRR	<i>Cleans PEcAn settings file and prepares the settings to be saved in a reference run record in BETY</i>
--------------------	---

Description

Cleans PEcAn settings file and prepares the settings to be saved in a reference run record in BETY

Usage

```
clean_settings_BRR(inputfile)
```

Arguments

inputfile the PEcAn settings file to be used.

Author(s)

Betsy Cowdery

compute_metrics	<i>Compute benchmark metrics</i>
-----------------	----------------------------------

Description

Compute benchmark metrics

Usage

```
compute_metrics(aligned, metrics = c("RMSE", "MAE", "R2"))
```

Arguments

aligned data.frame with columns: model, obvs, time
metrics character vector of metric names

Value

data.frame in wide format with columns 'Site' and each requested metric column per site.

create_BRR	<i>Create benchmark reference run and ensemble</i>
------------	--

Description

For each benchmark id, calculate metrics and update benchmarks_ensemble_scores

Usage

```
create_BRR(ens_wf, con, user_id = "")
```

Arguments

ens_wf	table made from joining ensemble and workflow tables
con	database connection
user_id	Optional user id to use for this record in reference_runs table

Author(s)

Betsy Cowdery

define_benchmark	<i>Benchmark Definition: Retrieve or Create Bety Benchmarking Records</i>
------------------	---

Description

Creates records for benchmarks, benchmarks_benchmarks_reference_runs, benchmarks_metrics

Usage

```
define_benchmark(settings, bety)
```

Arguments

settings	settings list
bety	database connection

Value

updated settings list

Author(s)

Betsy Cowdery

efi_long_to_array	<i>Reshape EFI Long Format DataFrame to Matrix of Ensemble Members</i>
-------------------	--

Description

Takes an EFI long format dataset (containing scenario, datetime/time, site_id/site, parameter/ensemble/member, variable, prediction/value) and reshapes it into a matrix of predictions with rows corresponding to timesteps and columns corresponding to ensemble members.

Usage

```
efi_long_to_array(df, var = NULL, site = NULL)
```

Arguments

df	data.frame in EFI long format.
var	character string specifying the target variable to filter (optional).
site	character string specifying the target site_id to filter (optional).

Value

A matrix with timesteps as rows and ensemble members as columns, with a ‘time’ attribute (POSIXct vector).

format_wide2long	<i>Function to convert wide format to long format</i>
------------------	---

Description

Function to convert wide format to long format

Usage

```
format_wide2long(out, format, vars_used, time.row)
```

Arguments

out	wide format data
format	as returned by query.format.vars
vars_used	data frame mapping ‘input_name’ to ‘bety_name’
time.row	ignored; value in output is set from ‘format\$vars\$storage_type’

Value

list of updated values

Author(s)

Istem Fer

generate_validation_report

Generate Validation Benchmark Report

Description

Generate Validation Benchmark Report

Usage

```
generate_validation_report(  
  benchmark_results,  
  output_file = "Validation_report.html",  
  template = NULL  
)
```

Arguments

benchmark_results	A list containing 'metrics' (data.frame), 'aligned_data' (data.frame), and 'plots' (list of ggplot objects) returned by the validation pipeline.
output_file	The path where the compiled report should be saved (e.g., "validation_report.html").
template	The path to the Quarto template. Defaults to the one provided in the package 'inst/reports/Validation_report.qmd'.

Author(s)

PEcAn Project

```
get_species_list_standard
    get_species_list_standard
```

Description

Returns the format type for convenience of use with match_species_id

Usage

```
get_species_list_standard(vars)
```

Arguments

vars format to be matched

Value

character Returns "usda", "latin_name", "fia" or "custom"

Author(s)

Tempest McCabe

```
load_and_map_data      Load and standardize arbitrary tabular data using a YAML mapping
                        configuration
```

Description

Load and standardize arbitrary tabular data using a YAML mapping configuration

Usage

```
load_and_map_data(data.path, mapping.path)
```

Arguments

data.path character, file path to the tabular data (e.g. .csv)
mapping.path character, file path to the YAML mapping configuration

Value

A standardized data frame with column names mapped to PEcAn standard vocabulary

load_csv	<i>load_csv</i>
----------	-----------------

Description

load_csv

Usage

```
load_csv(data.path, format, site, vars = NULL)
```

Arguments

data.path	character, file path to the .csv file
format	list, config list containing • header, numeric for number of header rows in file • skip, numeric for skip argument of <code>utils::read.csv</code> • na.strings, character vector for na.strings argument of <code>utils::read.csv</code>
site	list, currently ignored
vars	column names to return. If NULL, returns all columns

Value

a data frame with the processed data of the .csv file

Author(s)

Betsy Cowdery

load_data	<i>load_data</i>
-----------	------------------

Description

Generic function to convert input files containing observational data to a common PEcAn format.

Usage

```
load_data(
  data.path,
  format,
  start_year = NA,
  end_year = NA,
  site = NA,
  vars.used.index = NULL,
  ...
)
```

Arguments

data.path	character
format	list
start_year	numeric
end_year	numeric
site	list
vars.used.index	
	which variables to use? If NULL, these are taken from ‘format’
...	further arguments, currently ignored

Author(s)

Betsy Cowdery, Istem Fer, Joshua Mantooth

load_rds	<i>load_rds</i>
----------	-----------------

Description

load_rds

Usage

load_rds(data.path, format, site, vars = NULL)

Arguments

data.path	character
format	list, not used, for compatibility
site	not used, for compatibility
vars	optional variable names to load. if NULL, returns all variables in file

Author(s)

Istem Fer

`load_tab_separated_values`*Load files with mime-type 'text/tab-separated-values'*

Description

Load files with mime-type 'text/tab-separated-values'

Usage

```
load_tab_separated_values(data.path, format, site = NULL, vars = NULL)
```

Arguments

<code>data.path</code>	character
<code>format</code>	list
<code>site</code>	list
<code>vars</code>	variable names to load. If NULL, loads all columns

Author(s)

Betsy Cowdery, Mike Dietze

`load_x_netcdf`*Load from netCDF*

Description

Load from netCDF

Usage

```
load_x_netcdf(data.path, format, site, vars = NULL)
```

Arguments

<code>data.path</code>	character vector or list of paths to NetCDF files
<code>format</code>	list containing format info (e.g. na.strings)
<code>site</code>	list site info (optional)
<code>vars</code>	character vector of variables to load

Value

A data.frame with the requested variables and a 'time' column (POSIXct)

Author(s)

Istem Fer, Anshul Jain

match_timestep	<i>Match time step</i>
----------------	------------------------

Description

Match time step

Usage

```
match_timestep(date.coarse, date.fine, data.fine)
```

Arguments

date.coarse	numeric
date.fine	numeric
data.fine	matrix

Author(s)

Istem Fer

mean_over_larger_timestep	<i>Calculate benchmarking statistics</i>
---------------------------	--

Description

Calculate benchmarking statistics

Usage

```
mean_over_larger_timestep(date.coarse, date.fine, data.fine)
```

Arguments

date.coarse	numeric
date.fine	numeric
data.fine	data.frame

Author(s)

Betsy Cowdery, Michael Dietze

metric_AME	<i>Absolute Maximum Error</i>
------------	-------------------------------

Description

Absolute Maximum Error

Usage

```
metric_AME(dat, ...)
```

Arguments

dat	dataframe
...	ignored

Author(s)

Betsy Cowdery

metric_Bias	<i>Mean Bias Error</i>
-------------	------------------------

Description

Mean Bias Error (BIAS)

Usage

```
metric_Bias(dat, ...)
```

Arguments

dat	dataframe with columns 'model' and 'obvs'
...	ignored

Details

Computes the mean bias error between model predictions and observations.

Value

A numeric value representing the mean bias (mean(model - obvs)).

metric_cor	<i>Correlation Coefficient</i>
------------	--------------------------------

Description

Correlation Coefficient

Usage

```
metric_cor(dat, ...)
```

Arguments

dat	dataframe
...	ignored

Author(s)

Mike Dietze

metric_Coverage	<i>Prediction Interval Coverage</i>
-----------------	-------------------------------------

Description

Prediction Interval Coverage

Usage

```
metric_Coverage(dat, ...)
```

Arguments

dat	dataframe with columns 'model_q05' and 'model_q95' (or 'model_q025' and 'model_q975')
...	ignored

Details

Measures the fraction of observations that fall within the model's stated prediction interval (defaults to 90)

Value

A numeric value representing the fraction of observations that fall within the prediction interval.

`metric_CRPS`*Continuous Ranked Probability Score*

Description

Continuous Ranked Probability Score (CRPS)

Usage

```
metric_CRPS(dat, ...)
```

Arguments

<code>dat</code>	dataframe with column 'obvs' and either an 'ensemble_matrix' attribute, wide ensemble member columns ('member_1', 'member_2', ...), or matrix column 'model_ensemble'.
<code>...</code>	ignored

Details

Computes the Continuous Ranked Probability Score (CRPS) to evaluate model uncertainty and ensemble skill against observation data. Uses 'scoringRules::crps_sample()' when available, with a pure R analytical sample score fallback when 'scoringRules' is not present.

Value

A numeric value representing the mean CRPS across observations.

`metric_Frechet`*Frechet Distance*

Description

Frechet Distance

Usage

```
metric_Frechet(metric_dat, ...)
```

Arguments

<code>metric_dat</code>	dataframe
<code>...</code>	ignored

Author(s)

Betsy Cowdery

metric_lmDiag_plot	<i>Linear Regression Diagnostic Plot</i>
--------------------	--

Description

Linear Regression Diagnostic Plot

Usage

```
metric_lmDiag_plot(metric_dat, var, filename = NA, draw.plot = FALSE)
```

Arguments

metric_dat	data.frame
var	ignored
filename	path to save plot, or NA to not save
draw.plot	logical: return plot object?

Author(s)

Betsy Cowdery

metric_MAE	<i>Mean Absolute Error</i>
------------	----------------------------

Description

Mean Absolute Error

Usage

```
metric_MAE(dat, ...)
```

Arguments

dat	dataframe
...	ignored

Author(s)

Betsy Cowdery

metric_MSE	<i>Mean Square Error</i>
------------	--------------------------

Description

Mean Square Error

Usage

metric_MSE(dat, ...)

Arguments

dat	dataframe
...	ignored

Author(s)

Betsy Cowdery

metric_PMU	<i>Predictive Model Uncertainty (PMU)</i>
------------	---

Description

Predictive Model Uncertainty (PMU)

Usage

metric_PMU(dat, ...)

Arguments

dat	dataframe with columns 'obs_se' and 'obs_n'
...	ignored

Details

Calculates the pooled measurement uncertainty. Requires 'obs_se' (standard error) and 'obs_n' (replicate counts) in the observation data.

Value

A numeric value representing the pooled measurement uncertainty.

metric_PPMC	<i>Pearson Product Moment Correlation</i>
-------------	---

Description

Pearson Product Moment Correlation

Usage

metric_PPMC(metric_dat, ...)

Arguments

metric_dat	dataframe
...	ignored

Author(s)

Betsy Cowdery

metric_R2	<i>Squared Pearson Correlation (R2)</i>
-----------	---

Description

Squared Pearson Correlation (R2)

Usage

metric_R2(metric_dat, ...)

Arguments

metric_dat	dataframe with columns model and obvs
...	ignored

Details

Computes R-squared using the correlation-based formula: $R^2 = \left(\frac{\sum (obs - \bar{obs})(mod - \bar{mod})}{\sqrt{\sum (obs - \bar{obs})^2} \cdot \sqrt{\sum (mod - \bar{mod})^2}} \right)^2$

Note: Because this is a correlation-based R2, it is invariant to bias and slope. A model that is perfectly correlated but biased (e.g., model = obs + 100) will still score 1. This is distinct from variance explained or Nash-Sutcliffe Efficiency (NSE).

Author(s)

Betsy Cowdery

metric_RAE	<i>Relative Absolute Error</i>
------------	--------------------------------

Description

Relative Absolute Error

Usage

```
metric_RAE(metric_dat, ...)
```

Arguments

metric_dat	dataframe
...	ignored

Author(s)

Betsy Cowdery

metric_residual_plot	<i>Residual Plot</i>
----------------------	----------------------

Description

Residual Plot

Usage

```
metric_residual_plot(
  metric_dat,
  var,
  unit = NULL,
  filename = NA,
  draw.plot = is.na(filename)
)
```

Arguments

metric_dat	dataframe to plot, with at least columns 'time', 'model', 'obvs'
var	variable name, used as plot title
unit	measurement unit for the variable, added to the y-axis label; NULL to omit
filename	path to save plot, or NA to not save
draw.plot	logical: Return the plot object?

Author(s)

Betsy Cowdery

metric_RMSE	<i>Root Mean Square Error</i>
-------------	-------------------------------

Description

Root Mean Square Error

Usage

```
metric_RMSE(dat, ...)
```

Arguments

dat	dataframe
...	ignored

Author(s)

Betsy Cowdery

metric_run	<i>Model Run Check</i>
------------	------------------------

Description

Model Run Check

Usage

```
metric_run(settings)
```

Arguments

settings	list
----------	------

Author(s)

Betsy Cowdery

metric_scatter_plot	<i>Scatter Plot</i>
---------------------	---------------------

Description

Scatter Plot

Usage

```
metric_scatter_plot(  
  metric_dat,  
  var,  
  unit = NULL,  
  filename = NA,  
  draw.plot = is.na(filename)  
)
```

Arguments

metric_dat	dataframe to plot, with at least columns 'model' and 'obvs'
var	title for the plot
unit	measurement unit for the variable, added to the axis labels; NULL to omit
filename	path to save plot, or NA to not save
draw.plot	logical: Return the plot object?

Author(s)

Betsy Cowdery

metric_timeseries_plot	<i>Timeseries Plot</i>
------------------------	------------------------

Description

Timeseries Plot

Usage

```
metric_timeseries_plot(  
  metric_dat,  
  var,  
  unit = NULL,  
  filename = NA,  
  draw.plot = is.na(filename)  
)
```

Arguments

metric_dat	dataframe to plot, with at least columns ‘time’, ‘model’, ‘obvs’
var	variable name, used as plot title
unit	measurement unit for the variable, added to the y-axis label; NULL to omit
filename	path to save plot, or NA to not save
draw.plot	logical: Return the plot object?

Author(s)

Betsy Cowdery

pecan_metric_registry	<i>Metric Registry for PEcAn.benchmark</i>
-----------------------	--

Description

Metric Registry for PEcAn.benchmark

Usage

pecan_metric_registry

Format

An object of class environment of length 9.

read_settings_BRR	<i>Read settings from database using reference run id</i>
-------------------	---

Description

For each benchmark entry in a (multi)settings object, get run settings using reference run id and add to the settings object

Usage

read_settings_BRR(settings)

Arguments

settings	settings or multisettings object
----------	----------------------------------

Author(s)

Betsy Cowdery

register_metric	<i>Register a new benchmark metric</i>
-----------------	--

Description

Register a new benchmark metric

Usage

```
register_metric(name, func)
```

Arguments

name	Character name of the metric
func	Function that takes an aligned dataframe and returns a numeric value

run_benchmark	<i>Run a simple benchmark pipeline</i>
---------------	--

Description

Takes two validated dataframes, aligns by time, computes metrics, and returns a results table with a plot.

Usage

```
run_benchmark(
  model_df,
  obs_df,
  metrics = c("RMSE", "MAE"),
  tolerance_secs = 3600,
  method = "nearest"
)
```

Arguments

model_df	data.frame with columns: time (POSIXct), value (numeric)
obs_df	data.frame with columns: time (POSIXct), value (numeric)
metrics	character vector of metrics to compute. Options: "RMSE", "MAE"
tolerance_secs	nearest-neighbor time tolerance in seconds (default 1 hour)
method	alignment method: "nearest" or "interpolate"

Value

list with: metrics (data.frame), aligned (data.frame), plot (ggplot)

Author(s)

Anshul Jain

Index

- * **datasets**
 - pecan_metric_registry, [32](#)
- add_workflow_info, [3](#)
- align_by_first_observation, [4](#)
- align_by_time, [5](#)
- align_data, [5](#)
- align_data_to_data_pft, [6](#)
- align_pft, [8](#)
- bm_settings2pecan_settings, [10](#)
- bm_validate, [10](#)
- calc_benchmark, [11](#)
- calc_metrics, [11](#)
- check_BRR, [12](#)
- check_if_legal_table, [12](#)
- check_if_list_of_pfts, [13](#)
- check_if_species_list, [13](#)
- clean_settings_BRR, [14](#)
- compute_metrics, [14](#)
- create_BRR, [15](#)
- define_benchmark, [15](#)
- efi_long_to_array, [16](#)
- format_wide2long, [16](#)
- generate_validation_report, [17](#)
- get_species_list_standard, [18](#)
- load_and_map_data, [18](#)
- load_csv, [19](#)
- load_data, [19](#)
- load_rds, [20](#)
- load_tab_separated_values, [21](#)
- load_x_netcdf, [21](#)
- match_timestep, [22](#)
- mean_over_larger_timestep, [22](#)
- metric_AME, [23](#)
- metric_Bias, [23](#)
- metric_cor, [24](#)
- metric_Coverage, [24](#)
- metric_CRPS, [25](#)
- metric_Frechet, [25](#)
- metric_lmDiag_plot, [26](#)
- metric_MAE, [26](#)
- metric_MSE, [27](#)
- metric_PMU, [27](#)
- metric_PPMC, [28](#)
- metric_R2, [28](#)
- metric_RAE, [29](#)
- metric_residual_plot, [29](#)
- metric_RMSE, [30](#)
- metric_run, [30](#)
- metric_scatter_plot, [31](#)
- metric_timeseries_plot, [31](#)
- pecan_metric_registry, [32](#)
- read_settings_BRR, [32](#)
- register_metric, [33](#)
- run_benchmark, [33](#)