Package: kwb.heatsine (via r-universe)

August 30, 2024

Title R Package for Calculating Hydraulic Travel Times Based on Sinus Temperature Fitting

Version 0.1.5

Description Requires daily temperature times series in a surface water body and one groundwater observation well (in case of an production well this data needs to be cleaned in order to reduce temperature fluctuations due to the operation scheme!).

License MIT + file LICENSE

URL https://github.com/KWB-R/kwb.heatsine

BugReports https://github.com/KWB-R/kwb.heatsine/issues

Depends R (>= 2.10)

Imports dplyr (>= 1.0.2), forcats (>= 0.5.0), ggplot2 (>= 3.3.2), hydroGOF (>= 0.4.0), kwb.utils (>= 0.7.0), lubridate (>= 1.7.9), magrittr (>= 1.5), plotly (>= 4.9.2.1), readr (>= 1.4.0), rlang (>= 0.4.8), stringr (>= 1.4.0), tibble (>= 3.0.4), tidyr (>= 1.1.2)

Suggests covr (>= 3.5.1), htmlwidgets (>= 1.5.2), knitr (>= 1.30), rmarkdown (>= 2.4), sessioninfo (>= 1.1.1), testthat (>= 2.3.2), withr (>= 2.3.0)

VignetteBuilder knitr

Remotes github::kwb-r/kwb.utils

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.1.1

Repository https://kwb-r.r-universe.dev

RemoteUrl https://github.com/KWB-R/kwb.heatsine

RemoteRef HEAD

RemoteSha dff1238327f1e04a0c99dba66e116e3f19a23693

Contents

`extdata_file` *Get Path to File in This Package*

Description

Get Path to File in This Package

Usage

```
extdata_file(...)
```

Arguments

`...`	parts of path passed to `system.file`

`get_predictions` *Get Predictions*

Description

Get Predictions

Usage

```
get_predictions(sinusfit_sw, sinusfit_gw, retardation_factor = 2)
```

Arguments

sinusfit_sw	as retrieved by optimise_sinus_variablePeriod with surface water temperature data
sinusfit_gw	as retrieved by optimise_sinus_variablePeriod with groundwater temperature data
retardation_factor	hydraulic retardation factor (default: 2)

Value

list with sim/observation data ("data") fit parameters ("paras"), goodness-of-fit values ("gof") traveltimes ("traveltimes") and special (min, max, turning) points ("points")

---

get_tidy_traveltimes *Get tidy traveltimes*

Description

Get tidy traveltimes

Usage

```
get_tidy_traveltimes(traveltimes)
```

Arguments

traveltimes	traveltimes object as retrieved by get_predictions

Value

data frame with tidy traveltimes

---

get_travel_time *Helper function: get traveltime*

Description

Helper function: get traveltime

Usage

```
get_travel_time(sinusfit_sw, sinusfit_gw, retardation_factor = 1.8)
```

Arguments

sinusfit_sw	as retrieved by optimise_sinus_variablePeriod with surface water temperature data
sinusfit_gw	as retrieved by optimise_sinus_variablePeriod with groundwater temperature data
retardation_factor	hydraulic retardation factor (default: 2)

Value

data frame with travel times for min/max and turning points

---

load_temperature_from_csv *Load Temperature Data From CSV*

Description

Load Temperature Data From CSV

Usage

```
load_temperature_from_csv(path)
```

Arguments

path	path to csv file with temperature data and columns: "date" (YYYY-MM-DD) and "value"

Value

tibble with temperature data and columns "date" and "value"

Examples

```
path <- kwb.heatsine::extdata_file("temperature_groundwater_Txxxx3.csv")
gw_data <- kwb.heatsine::load_temperature_from_csv(path)
gw_data
```

`optimise_sinus_fixedPeriod` *Optimise Sinus Fit for Fixed Period*

Description

Optimise Sinus Fit for Fixed Period

Usage

```
optimise_sinus_fixedPeriod(df, period_length = 365.25)
```

Arguments

`df`	data frame with temperature data and columns "date" (YYYY-MM-DD) and "value"
`period_length`	period length (default: 365.25)

Value

list with fit parameters ("paras"), goodness-of-fit values ("gof"), special points, i.e. min/max/turning-points ("points), fit model ("lm_model") and input data ("data")

References

https://stats.stackexchange.com/questions/77543/how-do-i-get-the-amplitude-and-phase-for-sine-wave-from-lm-summary

`optimise_sinus_variablePeriod` *Optimise Sinus Fit for Variable Period*

Description

Optimise Sinus Fit for Variable Period

Usage

```
optimise_sinus_variablePeriod(
  temp_df,
  optFunc = opt_func,
  opt_limits = c(100, 500),
  opt_tolerance = 0.001,
  opt_debug = FALSE
)
```

Arguments

temp_df	data frame with temperature data and columns "date" (YYYY-MM-DD) and "value"
optFunc	optimisation function (default: opt_func)
opt_limits	optimisation limits for "period_length" (default: c(100,500))
opt_tolerance	(default: 0.001)
opt_debug	show debug information (default: FALSE)

Value

list with fit parameters ("paras"), goodness-of-fit values ("gof"), special points, i.e. min/max/turning-points ("points), fit model ("lm_model") and input data ("data")

References

https://stats.stackexchange.com/questions/77543/how-do-i-get-the-amplitude-and-phase-for-sine-wave-from-lm-summary

---

opt_func *Optimise Sinus Fit Function*

Description

Optimise Sinus Fit Function

Usage

```
opt_func(period, df, opt_criteria = "RMSE", debug = TRUE)
```

Arguments

period	period length
df	data frame with temperature data and columns "date" (YYYY-MM-DD) and "value"
opt_criteria	(default: "RMSE"), for other options check: ?hydroGOF::gof
debug	show debug messages (default: TRUE)

Value

scalar with optimisation result

plot_prediction_interactive *Plot Prediction Interactive*

Description

Plot Prediction Interactive

Usage

```
plot_prediction_interactive(predictions)
```

Arguments

predictions	as retrieved by get_predictions()

Value

interactive prediction plot

plot_temperature_interactive *Plot Temperature Interactive*

Description

Plot Temperature Interactive

Usage

```
plot_temperature_interactive(df)
```

Arguments

df	data frame with temperature data and columns "date" (YYYY-MM-DD) and "value"

Value

plot with interactive temperature data

Examples

```
path <- kwb.heatsine::extdata_file("temperature_groundwater_Txxxx3.csv")
gw_data <- kwb.heatsine::load_temperature_from_csv(path)
kwb.heatsine::plot_temperature_interactive(gw_data)
```

run_optimisation *Wrapper function for sinus optimisation*

Description

Wrapper function for sinus optimisation

Usage

```
run_optimisation(
  data_sw_selected,
  data_gw_selected,
  retardation_factor = 2,
  sw_monitoring_id = ifelse(!is.null(attr(data_sw_selected, "monitoring_id")),
    attr(data_sw_selected, "monitoring_id"), "surface-water monitoring point"),
  gw_monitoring_id = ifelse(!is.null(attr(data_gw_selected, "monitoring_id")),
    attr(data_gw_selected, "monitoring_id"), "groundwater monitoring point"),
  limits = c(100, 500),
  tolerance = 0.001,
  debug = FALSE
)
```

Arguments

data_sw_selected	data.frame with daily data temperature data of surface water monitoring point with columns "date" (format: "YYYY-MM-DD") and "value" (format: double, temperature in degree Celsius) for selected time period
data_gw_selected	data.frame with daily data temperature data of groundwater monitoring point with columns "date" (format: "YYYY-MM-DD") and "value" (format: double, temperature in degree Celsius) for selected time period
retardation_factor	hydraulic retardation factor (default: 2)
sw_monitoring_id	optional label for surface water monitoring id (default: "surface-water monitoring point" or attr(data_sw_selected, "monitoring_id") if data imported with load_temperature_from_csv), otherwise can be any user-defined character string to be used as label for the monitoring point
gw_monitoring_id	optional label for groundwater monitoring id (default: "surface-water monitoring point" or attr(data_sw_selected, "monitoring_id") if data imported with load_temperature_from_csv), otherwise can be any user-defined character string to be used as label for the monitoring point
limits	minimum/maximum period length for sinus optimisation in days (default: c(100, 500))

tolerance	the desired accuracy (default: 0.001
debug	show debug messages (default: FALSE)

Value

list with sim/observation data ("data") fit parameters ("paras"), goodness-of-fit values ("gof") travel-times ("traveltimes") and special (min, max, turning) points ("points") as returned by get_predictions

Examples

```
load_temp <- function(base_name) {
 kwb.heatsine::load_temperature_from_csv(
 kwb.heatsine::extdata_file(base_name)
 )
}

data_sw <- load_temp("temperature_surface-water_Txxsxx-mxxxxsxxx.csv")
data_gw <- load_temp("temperature_groundwater_Txxxx3.csv")

data_sw_selected <- kwb.heatsine::select_timeperiod(
 data_sw,
 date_start = "2015-10-10",
 date_end = "2016-10-14"
)

data_gw_selected <- kwb.heatsine::select_timeperiod(
 data_gw,
 date_start = "2015-12-28",
 date_end = "2016-12-26"
)

kwb.heatsine::run_optimisation(data_sw_selected = data_sw_selected,
data_gw_selected = data_gw_selected,
retardation_factor = 1.8,
sw_monitoring_id = attr(data_sw_selected, "monitoring_id"),
gw_monitoring_id = attr(data_gw_selected, "monitoring_id"),
limits = c(100, 500),
tolerance = 0.001,
debug = FALSE)
```

---

select_timeperiod *Helper function: select timeperiod*

Description

Helper function: select timeperiod

Usage

```
select_timeperiod(
  df,
  date_start,
  date_end = as.Date(date_start) + 365.25,
  col_date = "date"
)
```

Arguments

df	data frame with date (defined in parameter "col_date)
date_start	start date of selection
date_end	end date of selection. If no value is given 365.25 days after "date_start" will be used (default: as.Date(date_start) + 365.25)
col_date	column for dates (default: "date")

Value

data frame with selected dates

Examples

```
path <- kwb.heatsine::extdata_file("temperature_groundwater_Txxxx3.csv")
gw_data <- kwb.heatsine::load_temperature_from_csv(path)
gw_data_selected <- kwb.heatsine::select_timeperiod(gw_data,
date_start = "2015-12-28", date_end = "2016-12-26")
gw_data_selected
```

Index