Package: reslr (via r-universe)

October 12, 2026

Title Modelling Relative Sea Level Data

Version 0.2.0

URL https://github.com/maeveupton/reslr, https://maeveupton.github.io/reslr/

BugReports https://github.com/maeveupton/reslr/issues

Language en-US

Description The Bayesian modelling of relative sea-level data using a comprehensive approach that incorporates various statistical models within a unifying framework. Details regarding each statistical models; linear regression (Ashe et al 2019) <doi:10.1016/j.quascirev.2018.10.032>, change point models (Cahill et al 2015) <doi:10.1088/1748-9326/10/8/084002>, integrated Gaussian process models (Cahill et al 2015) <doi:10.1214/15-AOAS824>, temporal splines (Upton et al 2025) <doi:10.32614/RJ-2024-018>, spatio-temporal splines (Upton et al 2025) <doi:10.32614/RJ-2024-018> and generalised additive models (Upton et al 2025) <doi:10.1093/jrsssc/qlae044>. This package facilitates data loading, model fitting and result summarisation. Notably, it accommodates the inherent measurement errors found in relative sea-level data across multiple dimensions, allowing for their inclusion in the statistical models.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Depends R (>= 2.10)

LazyData true

Imports data.table, dplyr, fastDummies, fields, geosphere, ggplot2, magrittr, ncdf4, plyr, posterior, purrr, R2jags, stringr, tidybayes, tidyr

Suggests knitr, rmarkdown, testthat, vdiffr

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Maeve Upton [cph, aut, cre] (ORCID:
<https://orcid.org/0000-0002-3185-7731>), Andrew Parnell [aut],
Niamh Cahill [aut]

Maintainer Maeve Upton <uptonmaeve010@gmail.com>

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

Date/Publication 2026-05-26 10:20:02 UTC

RemoteUrl https://github.com/cran/reslr

RemoteRef HEAD

RemoteSha a3f61e231cfd51facb708e2b25a5e80d730382ff

Contents

cross_val_check 2
NAACproxydata 4
plot.reslr_input 5
plot.reslr_output 6
print.reslr_input 7
print.reslr_output 8
reslr_load 8
reslr_mcmc 10
summary.reslr_output 12

Index **13**

`cross_val_check` *Cross validation check for spline in time, spline in space time and GAM in order to select the most appropriate number of knots when creating basis functions.*

Description

Cross validation check for spline in time, spline in space time and GAM in order to select the most appropriate number of knots when creating basis functions.

Usage

```
cross_val_check(
  data,
  prediction_grid_res = 50,
  spline_nseg = NULL,
  spline_nseg_t = 20,
```

```
  spline_nseg_st = 6,
  n_iterations = 1000,
  n_burnin = 100,
  n_thin = 5,
  n_chains = 2,
  model_type,
  n_fold = 5,
  seed = NULL,
  CI = 0.95
)
```

Arguments

data	Raw input data
prediction_grid_res	Resolution of grid. Predictions over every 50 years(default) can vary based on user preference, as larger values will reduce computational run time.
spline_nseg	This setting is focused on the Noisy Input Spline model. It provides the number of segments used to create basis functions.
spline_nseg_t	This setting is focused on the Noisy Input Generalised Additive Model. It provides the number of segments used to create basis functions.
spline_nseg_st	This setting is focused on the Noisy Input Generalised Additive Model. It provides the number of segments used to create basis functions.
n_iterations	Number of iterations. Increasing this value will increase the computational run time.
n_burnin	Size of burn-in. This number removes a certain number of samples at the beginning.
n_thin	Amount of thinning.
n_chains	Number of MCMC chains. The number of times the model will be run.
model_type	The user selects their statistical model type. The user can select a Noisy Input Spline in Time using "ni_spline_t". The user can select a Noisy Input Spline in Space Time using "ni_spline_st". The user can select a Noisy Input Generalised Additive Model using "ni_gam_decomp".
n_fold	Number of folds required in the cross validation. The default is 5 fold cross validation.
seed	If the user wants reproducible results, seed stores the output when random selection was used in the creation of the cross validation.
CI	Size of the credible interval required by the user. The default is 0.95 corresponding to 95%.

Value

A list containing the model comparison measures, e.g. Root Mean Square Error (RMSE), and plot of true vs predicted values

Examples

```
data <- NAACproxydata %>% dplyr::filter(Site == "Cedar Island")
cross_val_check(data = data, model_type = "ni_spline_t",n_fold = 2)
```

`NAACproxydata` *Relative Sea level example dataset*

Description

An example dataset for 22 proxy sites along the Atlantic coast of North America

Usage

```
NAACproxydata
```

Format

A data frame with 1715 rows and 8 columns:

Region All regions along Atlantic coast of North America

Site All sites along Atlantic coast of North America

Latitude Latitude of the data site

Longitude Longitude of the data site

RSL Relative Sea level in meters

RSL_err Error in RSL

Age Age in years CE

Age_err Error in Age

Source

Check out the vignettes to see the sources

plot.reslr_input *Plot raw data with measurement uncertainty.*

Description

In this function, the raw data is plotted with grey uncertainty boxes representing the uncertainty associated with the input and the output. The function allows the user to plot the proxy record data and tide gauge data together or separately.

Usage

```
## S3 method for class 'reslr_input'
plot(
  x,
  title = "",
  xlab = "Year (CE)",
  ylab = "Relative Sea Level (m)",
  plot_tide_gauges = FALSE,
  plot_proxy_records = TRUE,
  plot_caption = TRUE,
  ...
)
```

Arguments

x	An object created via the function reslr_load
title	Title of the plot
xlab	Labeling the x-axis
ylab	Labeling the y-axis
plot_tide_gauges	Plotting the tide gauge data with the proxy records
plot_proxy_records	Plotting the proxy records on their own and this is the default
plot_caption	Plotting an informed caption with the number of tide gauges and proxy sites.
...	Not used

Value

Plot of the raw data with the measurement uncertainty.

Examples

```
full_dataset <- reslr_load(NAACproxydata)
plot(full_dataset)
```

`plot.reslr_output` *Plotting the results for each statistical model from the* `reslr_mcmc` *function.*

Description

Depending on the model chosen in the `reslr_mcmc` function, the package produces a range of output plots. Total posterior model fit plot with the raw data and measurement uncertainty are created for each statistical model. The rate of change plots are created for the EIV IGP and the NI spline regression models. For the NI GAM decomposition, each individual component of the model is plotted. Also, the regional and the non-linear local component, an associated rate plot is produced. If tide gauges are used in the model, the user has the ability plot the output with or without this additional data source.

Usage

```
## S3 method for class 'reslr_output'
plot(
  x,
  plot_proxy_records = TRUE,
  plot_tide_gauges = FALSE,
  title = "",
  plot_type = c("model_fit_plot"),
  plot_caption = TRUE,
  xlab = "Year (CE)",
  ylab = "Relative Sea Level (m)",
  y_rate_lab = "Rate of change (mm per year)",
  ...
)
```

Arguments

`x`	An object of class `reslr_output` and `model_type` created via `reslr_mcmc`
`plot_proxy_records`	Plotting the proxy records on their own and this is the default
`plot_tide_gauges`	Plotting the tide gauge data as well as proxy data
`title`	Plotting a title on the output plots
`plot_type`	The user can select the type of output plot they require from the following: "rate_plot", "model_fit_plot", "regional_plot", "regional_rate_plot", "linear_local_plot", "non_linear_local_plot", "non_linear_local_rate_plot", "nigam_component_plot"
`plot_caption`	Plotting an informed caption with the number of tide gauges and proxy sites.
`xlab`	Labeling the x-axis
`ylab`	Labeling the y-axis
`y_rate_lab`	Labeling the y-axis for rate of change plots
`...`	Not used

Value

Plot of model fit and the rate of change depending on the statistical model in question.

Examples

```
data <- NAACproxydata %>% dplyr::filter(Site == "Cedar Island")
x <- reslr_load(data = data)
jags_output <- reslr_mcmc(x, model_type = "eiv_slr_t")
plot(x = jags_output)
```

---

print.reslr_input *Print a reslr output object which is created from the* reslr_load *function.*

Description

In this function, the reslr input object is printed. This is a high-level summary which provides the number of observations and the number of sites utilised in the dataset.

Usage

```
## S3 method for class 'reslr_input'
print(x, ...)
```

Arguments

x	An object of class reslr_input
...	Other arguments (not supported)

Value

A neat presentation of your input reslr object

Examples

```
data <- NAACproxydata %>% dplyr::filter(Site == "Cedar Island")
reslr_input <- reslr_load(data = data)
print(x = reslr_input)
```

print.reslr_output *Print a reslr output object which is created by the* reslr_mcmc *function.*

Description

This will be very high level printing that the user can use to obtain information about the MCMC run using JAGS. The number of iterations and chains used by the user is printed In addition, the type of statistical model is printed.

Usage

```
## S3 method for class 'reslr_output'
print(x, ...)
```

Arguments

x	An object of class reslr_output
...	Other arguments (not supported)

Value

Returns high level information about the reslr_output object, i.e. the number of iterations and chains used.

Examples

```
data <- NAACproxydata %>% dplyr::filter(Site == "Cedar Island")
input_data <- reslr_load(data = data)
jags_output <- reslr_mcmc(input_data = input_data, model_type = "eiv_slr_t")
print(x = jags_output)
```

reslr_load *Loading in data for the* reslr *package*

Description

In this function, the data provided by the user is loaded into the package. The prerequisites of the input data structure has been given in the vignettes. The user can choose to include data from tide gauges which is sourced from the Permanent Service for Mean Sea Level (PSMSL) online database.

Usage

```
reslr_load(
  data,
  prediction_grid_res = 50,
  include_tide_gauge = FALSE,
  include_linear_rate = FALSE,
  list_preferred_TGs = NULL,
  TG_minimum_dist_proxy = FALSE,
  all_TG_1deg = FALSE,
  input_age_type = "CE",
  sediment_average_TG = 10,
  detrend_data = FALSE,
  core_col_year = NULL,
  cross_val = FALSE,
  test_set
)
```

Arguments

data	The input data as a dataframe.
prediction_grid_res	Resolution of grid. Predictions over every 50 years(default) can vary based on user preference, as larger values will reduce computational run time.
include_tide_gauge	Including tide gauge data from PSMSL website. The tide gauge data undergo a cleaning process in this function where flagged stations are removed as recommended by the online database. Next, the tide gauge data is averaged over period defined by sediment_average_TG which default is 10 years corresponding to accumulation rates of proxy records. Then, the user selects their preferred tide gauge based on three criteria: 1.nearest tide gauge to the proxy site; 2. User supplies a list of names of preferred tide gauges; 3. all tide gauges within 1 degree are chosen.
include_linear_rate	User decides to include linear_rate and linear_rate_err associated. This relates to linear rate which corresponds to an important physical process that impacts sea level observations which is glacial isostatic adjustment (GIA). For the linear_rate and its associated linear_rate_err, the user can provide these values as additional columns in the input dataframe. If they prefer, the package will calculate the linear_rate and the linear_rate_err using the data.
list_preferred_TGs	The user can supply the name or names of the preferred tide gauges from the PSMSL database.
TG_minimum_dist_proxy	The package finds the tide gauge closest to the proxy site
all_TG_1deg	The package finds all tide gauges within 1 degree of the proxy site
input_age_type	The inputted age in years "CE" or year "BP". The default is "CE" and is the preferred structure of the package. The package has the ability to use Before Present ("BP") observations.

`sediment_average_TG`
: Average the tide gauge data to make it comparable to accumulation rates of proxy records. The default averaging period for tide gauges is 10 years and the user can alter this.

`detrend_data`
: Detrend the data using the linear rate provided to remove this component.

`core_col_year`
: The year the sediment core was collected in order to the data to be detrended.

`cross_val`
: For the spline in time, spline in space time and the NIGAM the user can undertake cross validation to examine the

`test_set`
: The test set dataframe for cross validation

Value

A list containing data frame of data and prediction grid. The output of this function is two data frames, one with the data and one with the data_grid which represent a grid with evenly spaced time points. If tide gauge data is used, an ID column is included in the two output dataframes displaying the data source, "ProxyRecord" or "TideGaugeData".

Examples

```
data <- NAACproxydata %>% dplyr::filter(Site == "Cedar Island")
reslr_load(data = data)
```

---

reslr_mcmc *Run a* `reslr_input` *object through the main reslr Markov chain Monte Carlo (MCMC) function using a chosen statistical model*

Description

In this function, a variety of statistical models can be run depending on the requirements of the user (see reslr vignette for extension list of options). All models are written within a Bayesian framework and use JAGS (Just Another Gibbs Sampler) to calculate Markov Chain Monte Carlo (MCMC) simulation to obtain estimates of unknown parameters. The user has the ability to alter the number of iterations, the number of burnin, the number of chains and the thinning. These options relate to the amount of MCMC simulations required and should be reviewed by the user to ensure model convergence is achieved without excessively long run times. The user chooses their 'model_type' and as a range of models to choose from.

Usage

```
reslr_mcmc(
  input_data,
  model_type,
  n_cp = 1,
  igp_smooth = 0.2,
  n_iterations = 5000,
  n_burnin = 1000,
```

```
  n_thin = 4,
  n_chains = 3,
  CI = 0.95,
  spline_nseg = NULL,
  spline_nseg_t = 20,
  spline_nseg_st = 6
)
```

Arguments

input_data	Input data from the reslr_load function
model_type	The user selects their statistical model type. The user can select a Errors in Variable Simple Linear Regression using "eiv_slr_t". The user can select a Errors in Variable Change Point Regression using "eiv_cp_t". The user can select a Errors in Variable Integrated Gaussian Process using "eiv_igp_t". The user can select a Noisy Input Spline in Time using "ni_spline_t". The user can select a Noisy Input Spline in Space Time using "ni_spline_st". The user can select a Noisy Input Generalised Additive Model using "ni_gam_decomp".
n_cp	This setting is focused on the Errors in Variables Change Point model. The user can select the number of change points 1,2 or 3.
igp_smooth	This setting is focused on the Errors in Variables Integrated Gaussian Process model. It informs the prior for the smoothness (correlation) parameter if model = "igp" is chosen. Choose a value between 0 and 1. Closer to 1 will increase smoothness.
n_iterations	Number of iterations. Increasing this value will increase the computational run time.
n_burnin	Size of burn-in. This number removes a certain number of samples at the beginning.
n_thin	Amount of thinning.
n_chains	Number of MCMC chains. The number of times the model will be run.
CI	Size of the credible interval required by the user. The default is 0.95 corresponding to 95%.
spline_nseg	This setting is focused on the Noisy Input Spline model. It provides the number of segments used to create basis functions. See Advanced vignette for more details.
spline_nseg_t	This setting is focused on the Noisy Input Generalised Additive Model. It provides the number of segments used to create basis functions. See Advanced vignette for more details.
spline_nseg_st	This setting is focused on the Noisy Input Generalised Additive Model. It provides the number of segments used to create basis functions. See Advanced vignette for more details.

Value

A list containing the input data, the JAGS output and output dataframes used for final plots.

Examples

```
data <- NAACproxydata %>% dplyr::filter(Site == "Cedar Island")
input_data <- reslr_load(data = data)
reslr_mcmc(input_data = input_data, model_type = "eiv_slr_t")
```

`summary.reslr_output` *Produces summaries and convergence diagnostics for an object created with* `reslr_mcmc`.

Description

A warning message will appear if the model has not been converge. If this appears the user is recommended to re-run the model and alter the `reslr_mcmc` function default iteration and MCMC settings. Also, it provides high-level summaries of the estimated parameters.

Usage

```
## S3 method for class 'reslr_output'
summary(object, ...)
```

Arguments

`object`	Output object from the `reslr_mcmc`
`...`	Not in use

Value

A list containing convergence diagnostics and parameter estimates for the output.

Examples

```
data <- NAACproxydata %>% dplyr::filter(Site == "Cedar Island")
input_data <- reslr_load(data = data)
jags_output <- reslr_mcmc(input_data = input_data, model_type = "eiv_slr_t")
summary(object = jags_output)
```

Index

∗ **datasets**
NAACproxydata, 4

cross_val_check, 2

NAACproxydata, 4

plot.reslr_input, 5
plot.reslr_output, 6
print.reslr_input, 7
print.reslr_output, 8

reslr_load, *5*, 8
reslr_mcmc, *6*, 10, *12*

summary.reslr_output, 12