Package 'GaSP'

December 8, 2023

Type Package

Title Train and Apply a Gaussian Stochastic Process Model

Version 1.0.5

Description Train a Gaussian stochastic process model of an unknown function, possibly observed with error, via maximum likelihood or maximum a posteriori (MAP) estimation, run model diagnostics, and make predictions, following Sacks, J., Welch, W.J., Mitchell, T.J., and Wynn, H.P. (1989) ``Design and Analysis of Computer Experiments'', Statistical Science, <doi:10.1214/ss/1177012413>. Perform sensitivity analysis and visualize low-order effects, following Schonlau, M. and Welch, W.J. (2006), ``Screening the Input Variables to a Computer Model Via Analysis of Variance and Visualization'', <doi:10.1007/0-387-28014-6_14>.

Depends R (>= 3.5.0)

Suggests markdown, rmarkdown, knitr, testthat

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.2.3

VignetteBuilder knitr

NeedsCompilation yes

Author William J. Welch [aut, cre, cph]
(<https://orcid.org/0000-0002-4575-3124>),
Yilin Yang [aut] (<https://orcid.org/0000-0003-0885-6017>)

Maintainer William J. Welch <will@stat.ubc.ca>

Repository CRAN

Date/Publication 2023-12-08 06:30:03 UTC

R topics documented:

`borehole` *Data for the borehole function*

Description

Training and test data for the borehole function; see source for background.

Usage

```
borehole
```

Format

A list with the following four data frames:

x 8-dimensional input for 40 training runs.

y Output (the flow) for the 40 training runs in `x`.

x_pred 8-dimensional input for 1000 test runs at which to predict `y`.

y_true Output for the 1000 runs in `x_pred`.

Source

https://www.sfu.ca/~ssurjano/borehole.html

CrossValidate *Cross-validated predictions for a* GaSPModel *object.*

Description

Compute leave-one-out cross-validated predictions for a GaSPModel object.

Usage

```
CrossValidate(GaSP_model)
```

Arguments

GaSP_model	Object of class GaSPModel.

Value

A data frame with two columns: the cross-validated predictions Pred and their standard errors SE.

Note

RMSE computes the root mean squared error of the predictions. PlotPredictions and PlotResiduals plot the predictions or their residuals; PlotStdResiduals and PlotQQ plot the stanadardized residuals.

Examples

```
borehole_cv <- CrossValidate(borehole_fit)
```

DescribeX *Describe the input variables.*

Description

Describe the input variables to set up integration or summation ranges for Visualize.

Usage

```
DescribeX(
  x_names,
  x_min,
  x_max,
  support = NULL,
  num_levels = NULL,
  distribution = NULL
)
```

Arguments

x_names	A vector of character strings containing the names of the input variables.
x_min, x_max	Vectors of the same length as x_names containing the minima and maxima, respectively, of the input variables.
support	Optional vector of character strings of the same length as x_names. Valid strings for a variable are: "Continuous" (continuous between the input's x_min and x_max); "Fixed" (the input's x_min must equal its x_max); and "Grid" (which requires the next argument).
num_levels	An optional vector of integers for the number of levels of each input; must be present if the support argument includes "Grid". An input's number of levels is 0 if it is "Continuous", 1 if it is "Fixed", or > 1 if it is "Grid" to define an equally spaced grid inclusive of the input's x_min and x_max.
distribution	An optional vector of character strings of the same length as x_names to define the weight distributions of the input variables. Valid strings are "Uniform" or "Normal" (ignored for "Fixed" inputs).

Value

A data frame with the following columns: Variable (containing x_names), Min (containing x_min), and Max (containing x_max), plus the optional columns Support (from support), NumberLevels (from num_levels), and Distribution (from distribution).

Note

Does not check against GaSPModel and all characters are CASE SENSITIVE.

Examples

```
borehole_x_names <- colnames(borehole$x)
borehole_min <- c(0.05, 100.00, 63070.00, 990.00, 63.10, 700.00, 1120.00, 9855.00)
borehole_max <- c(0.15, 50000.00, 115600.00, 1110.00, 116.00, 820.00, 1680.00, 12045.00)
borehole_x_desc <- DescribeX(borehole_x_names, borehole_min, borehole_max)
```

Fit *Fit a GaSP model.*

Description

Fit (train) a GaSP model.

Usage

```
Fit(
  x,
  y,
  reg_model,
  sp_model = NULL,
  cor_family = c("PowerExponential", "Matern"),
  cor_par = data.frame(0),
  random_error = c(FALSE, TRUE),
  sp_var = -1,
  error_var = -1,
  nugget = 1e-09,
  tries = 10,
  seed = 500,
  fit_objective = c("Likelihood", "Posterior"),
  theta_standardized_min = 0,
  theta_standardized_max = .Machine$double.xmax,
  alpha_min = 0,
  alpha_max = 1,
  derivatives_min = 0,
  derivatives_max = 3,
  log_obj_tol = 1e-05,
  log_obj_diff = 0,
  lambda_prior = 0.1,
  model_comparison = c("Objective", "CV")
)
```

Arguments

`x`	A data frame containing the input (explanatory variable) training data.
`y`	A vector or a data frame with one column containing the output (response) training data.
`reg_model`	The regression model, specified as a formula, but note the left-hand side of the formula is unused; see example.
`sp_model`	An optional stochastic process model, specified as a formula, but note the left-hand side of the formula and the intercept are unused. The default `NULL` uses all column names in `x`.
`cor_family`	A character string specifying the (product, anisoptropic) correlation-function family: "PowerExponential" for the power-exponential family or "Matern" for the Matern family.
`cor_par`	An optional data frame containing the correlation parameters with one row per `sp_model` term and two columns set up as described in `GaSPModel` Details; only used to start the first objective optimization (see Details).
`random_error`	A boolean for the presence or not of a random (measurement, white-noise) error term.

`sp_var, error_var`
: Starting values of the stochastic process and error variances for the first try to optimize the objective (see Details); valid (i.e., nonnegative) values will only be used if `random_error = TRUE`. The invalid default value of -1 indicates that a starting value will be chosen by `Fit`.

`nugget`
: For numerical stability the proportion of the total variance due to random error is fixed at this value (`random_error = FALSE`) or bounded below by it (`random_error = TRUE`).

`tries`
: Number of optimizations of the objective from different random starting points.

`seed`
: The random-number seed to generate starting points.

`fit_objective`
: The objective that `Fit` attempts to optimize: "Likelihood" (maximum likelihood estimation) or "Posterior" (Bayesian maximum a posteriori estimation).

`theta_standardized_min, theta_standardized_max`
: The minimum and maximum of the standardized θ parameter (see Details).

`alpha_min, alpha_max`
: The minimum and maximum of the α parameter of power-exponential.

`derivatives_min, derivatives_max`
: The minimum and maximum of the δ parameter of Matern.

`log_obj_tol`
: An absolute tolerance for terminating the optimization of the log of the objective.

`log_obj_diff`
: The critical value for the change in the log objective for informal tests during optimization of correlation parameters. No testing is done with the default of 0; a larger critical value such as 2 may give a more parsimonious model.

`lambda_prior`
: The rate parameter of an exponential prior for each θ parameter; used only if `fit_objective = "Posterior"`.

`model_comparison`
: The criterion used to select from multiple solutions when `tries`> 1: the objective function ("Objective") or leave-one-out cross validation ("CV").

Details

Fit numerically optimizes the profile objective function with respect to the correlation parameters; the mean and overall variance parameters are estimated in closed form given the correlation parameters.

A `cor_par` data frame supplied by the user is the starting point for the first optimization try. If `random_error = TRUE`, then `sp_var` / (`sp_var` + `error_var`) is another correlation parameter to be optimized; `sp_var` and `error_var` values supplied by the user will initialize this parameter for the first try.

Set `random_error = TRUE` to estimate the variance of the random (measurement, white-noise) error; a small `nugget` error variance is for numerical stability.

For term j in the stochastic-process model, the estimate of θ_j is constrained between `theta_standardized_min` / r_j^2 and `theta_standardized_max` / r_j^2, where r_j is the range of term j. Note that `Fit` returns unscaled estimates relating to the original, unscaled inputs.

Value

A GaSPModel object, which is a list with the following components:

x	The data frame containing the input training data.
y	The training output data, now as a vector.
reg_model	The regression model, now in the form of a data frame.
sp_model	The stochastic process model, now in the form of a data frame.
cor_family	The correlation family.
cor_par	A data frame for the estimated correlation parameters.
random_error	The boolean for the presence or not of a random error term.
sp_var	The estimated stochastic process variance.
error_var	The estimated random error variance.
beta	A data frame holding the estimated regression-model parameters.
objective	The maximum value found for the objective function: the log likelihood (fit_objective = "Likelihood") or the log posterior (fit_objective = "Posterior").
cond_num	The condition number.
CVRMSE	The leave-one-out cross-validation root mean squared error.

References

Sacks, J., Welch, W.J., Mitchell, T.J., and Wynn, H.P. (1989) "Design and Analysis of Computer Experiments", *Statistical Science*, 4, pp. 409-423, doi:10.1214/ss/1177012413.

Examples

```
x <- borehole$x
y <- borehole$y
borehole_fit <- Fit(
  reg_model = ~1, x = x, y = y, cor_family = "Matern",
  random_error = FALSE, nugget = 0, fit_objective = "Posterior"
)
```

GaSPModel *Create a* GaSPModel *object.*

Description

Return a template for a GaSPModel object.

Usage

```
GaSPModel(
  x,
  y,
  reg_model,
  sp_model = NULL,
  cor_family = c("PowerExponential", "Matern"),
  cor_par,
  random_error = c(FALSE, TRUE),
  sp_var,
  error_var = 0
)
```

Arguments

x	A data frame containing the input (explanatory variable) training data.
y	A vector or a data frame with one column containing the output (response) training data.
reg_model	The regression model, specified as a formula, but note the left-hand side of the formula is unused; see example.
sp_model	An optional stochastic process model, specified as a formula, but note the left-hand side of the formula and the intercept are unused. The default NULL uses all column names in x.
cor_family	A character string specifying the (product, anisoptropic) correlation-function family: "PowerExponential" for the power-exponential family or "Matern" for the Matern family.
cor_par	A data frame containing the correlation parameters with one row per sp_model term and two columns (see Details).
random_error	A boolean for the presence or not of a random (measurement, white-noise) error term.
sp_var	The stochastic process variance.
error_var	The random error variance, with default 0.

Details

The data frame cor_par contains one row for each term in the stochastic process model. There are two columns. The first is named Theta, and the second is either Alpha (power-exponential) or Derivatives (Matern). Let h_j be a distance between points for term j in the stochastic-process model. For power-exponential, the contribution to the product correlation from term j depends on a distance-scale parameter θ_j from the Theta column and a smoothness parameter α_j from the Alpha column; the contribution is $exp(-\theta_j h_j^{2-\alpha_j})$. For example, $\alpha_j = 0$ gives the squared-exponential (Gaussian) correlation. The contribution to the product correlation for Matern also depends on θ_j, and the second parameter is the number of derivatives $\delta_j = 0, 1, 2, 3$ from the Derivatives column. The contribution is $exp(-\theta_j h_j)$ for $\delta_j = 0$ (the exponential correlation), $exp(-\theta_j h_j)(\theta_j h_j + 1)$ for $\delta_j = 1$, $exp(-\theta_j h_j)((\theta_j h_j)^2/3 + \theta_j h_j + 1)$ for $\delta_j = 2$, and $exp(-\theta_j h_j^2)$ for $\delta_j = 3$ (the squared-exponential correlation). Note that $\delta_j = 3$ codes for a limiting infinite number of derivatives. This

is not the usual parameterization of the Matern, but it is consistent with power-exponential for the exponential and squared-exponential special cases common to both.

A value should be given to `error_var` if the model has a random-error term (`random_error = TRUE`), and a small "nugget" such as 10^{-9} may be needed for improved numerical conditioning.

Value

A `GaSPModel` object, which is a list with the following components:

`x`	The data frame containing the input training data.
`y`	The training output data, now as a vector.
`reg_model`	The regression model, now in the form of a data frame.
`sp_model`	The stochastic process model, now in the form of a data frame.
`cor_family`	The correlation family.
`cor_par`	The data frame containing the correlation parameters.
`random_error`	The boolean for the presence or not of a random error term.
`sp_var`	The stochastic process variance.
`error_var`	The random error variance.
`beta`	A placeholder for a data frame to hold the regression-model parameters.
`objective`	A placeholder for the maximum fit objective.
`cond_num`	A placeholder for the condition number.
`CVRMSE`	A placeholder for the model's cross-validated root mean squared error.

Note

This function does not excecute `Fit` and is intended for `CrossValidate`, `Predict` and `Visualize` with models trained otherwise by the user. Placeholders do not need to be specified to excecute these further functions, as they are always recomputed as needed.

References

Sacks, J., Welch, W.J., Mitchell, T.J., and Wynn, H.P. (1989) "Design and Analysis of Computer Experiments", *Statistical Science*, 4, pp. 409-423, doi:10.1214/ss/1177012413.

Examples

```
x <- borehole$x
y <- borehole$y
theta <- c(
  5.767699e+01, 0.000000e+00, 0.000000e+00, 1.433571e-06,
  0.000000e+00, 2.366557e-06, 1.695619e-07, 2.454376e-09
)
alpha <- c(
  1.110223e-16, 0.000000e+00, 0.000000e+00, 0.000000e+00,
  0.000000e+00, 0.000000e+00, 2.494862e-03, 0.000000e+00
)
cor_par <- data.frame(Theta = theta, Alpha = alpha)
```

```
rownames(cor_par) <- colnames(borehole$x)
sp_var <- 38783.7
borehole_gasp <- GaSPModel(
  x = borehole$x, y = borehole$y,
  reg_model = ~1, cor_family = "PowerExponential",
  cor_par = cor_par, random_error = FALSE,
  sp_var = sp_var
)
```

PlotAll *Execute* PlotPredictions, PlotResiduals, PlotStdResiduals, PlotMainEffects, *and* PlotJointEffects.

Description

Execute PlotPredictions, PlotResiduals and PlotStdResiduals (all applied to cross validation only), PlotMainEffects, and PlotJointEffects.

Usage

```
PlotAll(
  GaSP_model,
  cross_validation,
  visualization,
  y_name = "y",
  y_units = "",
  x_units = NULL,
  se_plot = TRUE,
  y_values = NULL,
  se_values = NULL,
  pch = 1
)
```

Arguments

GaSP_model	Object of class GaSPModel, the entire model will be verified but only x and y will be used.
cross_validation	A data frame returned by CrossValidate.
visualization	A list object returned by Visualize.
y_name	An optional character string containing the output variable name (for labels).
y_units	An optional character string containing the units of the output variable (for labels).
x_units	An optional vector of character strings containing the units of the input variables (for labels).
se_plot	An optional boolean indicating whether to make standard-error contour plots.

y_values	An optional vector of contour values for the estimated joint effects.
se_values	An optional vector of contour values for the standard errors.
pch	Plotting symbol for plot; default is open circle.

Value

No return value, generates plots.

Examples

```
PlotAll(borehole_fit, borehole_cv, borehole_vis)
```

PlotJointEffects *Plot the estimated joint effects.*

Description

Plot the estimated joint effects.

Usage

```
PlotJointEffects(
  joint_effect,
  anova_percent,
  x_units = NULL,
  y_name = "y",
  y_units = "",
  se_plot = TRUE,
  y_values = NULL,
  se_values = NULL
)
```

Arguments

joint_effect	A data frame from Visualize with plotting coordinates for the estimated joint effects.
anova_percent	A data frame from Visualize of ANOVA percentages.
x_units	An optional vector of character strings containing the units of the input variables (for labels).
y_name	An optional character string containing the output variable name (for labels).
y_units	An optional character string containing the units of the output variable (for labels).
se_plot	An optional boolean indicating whether to make standard-error contour plots.
y_values	An optional vector of contour values for the estimated joint effects.
se_values	An optional vector of contour values for the standard errors.

Details

Plots are sent to the active device.

Value

No return value, generates plots.

Examples

```
PlotJointEffects(borehole_vis$joint_effect, borehole_vis$anova_percent)
```

`PlotMainEffects` *Plot the estimated main effects.*

Description

Plot the estimated main effects.

Usage

```
PlotMainEffects(
  main_effect,
  anova_percent,
  x_units = NULL,
  y_name = "y",
  y_units = ""
)
```

Arguments

`main_effect`	A data frame from `Visualize` with plotting coordinates for the estimated main effects.
`anova_percent`	A data frame from `Visualize` of `ANOVA` `percentages`.
`x_units`	An optional vector of character strings containing the units of the input variables (for labels).
`y_name`	An optional character string containing the output variable name (for labels).
`y_units`	An optional character string containing the units of the output variable (for labels).

Details

Plots are sent to the active device. Each plot shows an estimated main effect (red solid line) and pointwise approximate 95% confidence limits (green dashed line).

Value

No return value, generates plots.

Examples

```
PlotMainEffects(borehole_vis$main_effect, borehole_vis$anova_percent)
```

`PlotPredictions` *Plot true versus predicted output.*

Description

Plot true versus predicted output (response) made by `Predict` or `CrossValidate`.

Usage

```
PlotPredictions(
  y_pred,
  y,
  y_name = "y",
  y_units = "",
  title = c("Predict", "CrossValidate"),
  pch = 1
)
```

Arguments

`y_pred`	A data frame of predicted output values made by `Predict` or `CrossValidate`.
`y`	A vector of length equal to the number of rows in `y_pred` containing the true output values.
`y_name`	An optional character string containing the output variable name (for labels).
`y_units`	An optional character string constaining the units of the output variable (for labels).
`title`	A character string for the name of the function generating the predictions (for an appropriate title): "Predict" from `Predict` or "CrossValidate" from `CrossValidate`; "" for no title.
`pch`	Plotting symbol for `plot`; default is open circle.

Value

No return value, generates plots.

Examples

```
PlotPredictions(borehole_cv, y, title = "CrossValidate")

PlotPredictions(borehole_pred$y_pred, borehole$y_true, title = "Predict")
```

`PlotQQ` *Normal quantile-quantile (Q-Q) plot.*

Description

Normal quantile-quantile (Q-Q) plot of the standardized residuals of predictions from `Predict` or `CrossValidate`.

Usage

```
PlotQQ(y_pred, y, y_name = "y")
```

Arguments

`y_pred`	A data frame of predicted output values made by `Predict` or `CrossValidate`.
`y`	A vector of length equal to the number of rows in `y_pred` containing the true output values.
`y_name`	An optional character string containing the output variable name (for labels).

Value

No return value, generates plots.

Examples

```
PlotQQ(borehole_cv, y)
```

`PlotResiduals` *Plot residuals versus each input variable.*

Description

Plot residuals versus each input variable.

Usage

```
PlotResiduals(
  x,
  y_pred,
  y,
  x_units = NULL,
  y_name = "y",
  y_units = "",
  pch = 1
)
```

Arguments

x	A data frame with number of rows equal to the number of rows in y_pred containing the input (explanatory) variables.
y_pred	A data frame of predicted output values made by Predict or CrossValidate.
y	A vector of length equal to the number of rows in y_pred containing the true output values.
x_units	An optional vector of character strings containing the units of the input variables in x (for labels).
y_name	An optional character string containing the output variable name (for labels).
y_units	An optional character string constaining the units of the output variable (for labels).
pch	Plotting symbol for plot; default is open circle.

Value

No return value, generates plots.

Examples

```
PlotResiduals(x, borehole_cv, y)
```

PlotStdResiduals *Plot standardized residuals versus predictions.*

Description

Plot standardized residuals versus predictions made by Predict or CrossValidate.

Usage

```
PlotStdResiduals(
  y_pred,
  y,
  y_name = "y",
  y_units = "",
  title = c("Predict", "CrossValidate"),
  pch = 1
)
```

Arguments

y_pred	A data frame of predicted output values made by Predict or CrossValidate.
y	A vector of length equal to the number of rows in y_pred containing the true output values.
y_name	An optional character string containing the output variable name (for labels).
y_units	An optional character string constaining the units of the output variable (for labels).
title	A character string for the name of the function generating the predictions (for an appropriate title): "Predict" from Predict or "CrossValidate" from CrossValidate; "" for no title.
pch	Plotting symbol for plot; default is open circle.

Value

No return value, generates plots.

Examples

```
PlotStdResiduals(borehole_cv, y, title = "CrossValidate")
```

Predict *Predict from a* GaSPModel *object.*

Description

Predict from a GaSPModel object.

Usage

```
Predict(GaSP_model, x_pred, generate_coefficients = c(FALSE, TRUE))
```

Arguments

GaSP_model	Object of class GaSPModel.
x_pred	A data frame containing the values of the input variables at which to predict the output.
generate_coefficients	A boolean indicating whether coefficients for further external predictions are generated.

Value

A list with the following elements:

y_pred	A data frame with two columns: the predictions Pred and their standard errors SE.
pred_coeffs	A vector of coefficients for further predictions; NULL if generate_coefficients is FALSE.

Note

The vector of prediction coefficients in pred_coeffs can be used as follows. Let c denote the coefficients and let r denote a vector with element i containing the correlation between the output at a given new point and the output at training point i. Then the prediction for the output at the new point is the dot product of c and r.

RMSE computes the root mean squared error of the predictions. PlotPredictions and PlotResiduals plot the predictions or their residuals; PlotStdResiduals and PlotQQ plot the standardized residuals.

Examples

```
borehole_pred <- Predict(
  GaSP_model = borehole_fit,
  x_pred = borehole$x_pred,
  generate_coefficients = TRUE
)
```

RMSE *Calculate the root mean squared error (RMSE) of prediction*

Description

Calculate the root mean squared error (RMSE) of prediction

Usage

```
RMSE(y_pred, y_true, normalized = FALSE)
```

Arguments

y_pred	A vector of predicted output values.
y_true	A vector of true output values.
normalized	An optional boolean: if TRUE, the RMSE is normalized by dividing it by the standard deviation of y_true.

Value

The RMSE or normalized RMSE.

Examples

```
RMSE(borehole_pred$y_pred$Pred, borehole$y_true)

RMSE(borehole_cv$Pred, y)
```

Visualize *Visualize a* GaSPModel *object.*

Description

Carry out a functional analysis of variance (ANOVA) of a GaSPModel object and generate plotting coordinates for its estimated main and 2-input joint effects.

Usage

```
Visualize(GaSP_model, x_description, main_percent = 0, interaction_percent = 0)
```

Arguments

GaSP_model	Object of class GaSPModel.
x_description	A data frame describing the input variables. See DescribeX.
main_percent	An optional minimum percentage of variation explained by an input's main effect to return the effect's plotting coordinates; the default of zero gives plotting coordinates for all inputs.
interaction_percent	An optional minimum percentage of variation explained by the interaction effect of a pair of inputs to return the plotting coordinates for their joint effect (main effects plus interaction effect); the default of zero gives plotting coordinates for all pairs of inputs.

Details

If there are many inputs, to avoid excessive plotting of many trivial joint effects set interaction_percent = 1 say.

Value

A list with the following elements:

`anova_percent`	A data frame containing the ANOVA percentages for all main effects and 2-input interaction effects.
`main_effect`	A data frame with plotting coordinates for the estimated main effects.
`joint_effect`	A data frame with plotting coordinates for the estimated 2-input joint effects.
`total_percent`	Total percentage of the prediction variation accounted for by all main effects and 2-input interaction effects.
`average`	Overall average of the prediction function, averaged with respect to all inputs.
`SE_average`	Standard error of the overall average.

References

Schonlau, M. and Welch, W.J. (2006), "Screening the Input Variables to a Computer Model Via Analysis of Variance and Visualization", in *Screening: Methods for Experimentation in Industry, Drug Discovery, and Genetics,* Dean. A. and Lewis, S., eds., pp. 308-327, Springer, New York, doi:10.1007/0-387-28014-6_14.

Examples

```
borehole_vis <- Visualize(borehole_fit, borehole_x_desc)
```

Index