Package: ezECM (via r-universe)

October 18, 2024

Title Event Categorization Matrix Classification for Nuclear Detonations

Version 1.0.0

Description Implementation of an Event Categorization Matrix (ECM) detonation detection model and a Bayesian variant. Functions are provided for importing and exporting data, fitting models, and applying decision criteria for categorizing new events. This package implements methods described in the paper ``Bayesian Event Categorization Matrix Approach for Nuclear Detonations'' Koermer, Carmichael, and Williams (2024) available on arXiv at <doi:10.48550/arXiv.2409.18227>.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.2

Suggests knitr, rmarkdown, kableExtra, testthat (>= 3.0.0), LaplacesDemon

Config/testthat/edition 3

Imports ellipse, graphics, grDevices, klaR, lhs, MCMCpack, Rdpack, stats, utils, mvnfast

RdMacros Rdpack

VignetteBuilder knitr

URL `https://github.com/lanl/ezECM`

BugReports `https://github.com/lanl/ezECM/issues`

NeedsCompilation no

Author Scott Koermer [aut, cre, cph] (<`https://orcid.org/0000-0002-0325-7034`>)

Maintainer Scott Koermer <`skoermer@lanl.gov`>

Repository CRAN

Date/Publication 2024-10-17 17:10:01 UTC

Contents

`BayesECM` *Bayesian Event Matrix Categorization*

Description

Training a Bayesian ECM (B-ECM) model

Usage

```
BayesECM(
  Y,
  BT = c(100, 1000),
  priors = "default",
  verb = FALSE,
  transform = "logit"
)
```

Arguments

`Y`	`data.frame` of training data, with rows corresponding to N individual observations, and columns corresponding to p discriminants. An additional column named `"event"` is required, which labels each row with the known event category. Missing data is specified with `NA`. `dim(Y)` must equal `c(N, p + 1)`.
`BT`	integer vector of length 2, stipulating the number of `c("Burn-in", "Total")` Markov-Chain Monte-Carlo samples drawn.
`priors`	list of parameters to be used in prior distributions. See details.
`verb`	logical. A setting of `TRUE` prints a progress bar as samples are drawn, as well as warnings when they occur.
`transform`	character string specifying the transform to use on the elements of `Y` before fitting the model. Options are `"logit"` and `"arcsin"` with `"logit"` being the default. See details.

Details

The output of `BayesECM()` provides a trained Bayesian Event Categorization Matrix (B-ECM) model, utilizing the data and prior parameter settings . If there are missing values in `Y`, these values are imputed. A trained `BayesECM` model is then used with the `predict.BayesECM()` function to calculate expected category probabilities.

Data Preparation:

Before the data in `Y` is used with the model, the p-values $\in (0, 1]$ are transformed in an effort to better align the data with some properties of the normal distribution. When `transform == "logit"` the inverse of the logistic function $Y_{N \times p} = \log{(\texttt{Y})} - \log{(1 - \texttt{Y})}$ maps the values to the real number line. Values of `Y` exactly equal to 0 or 1 cannot be used when `transform == "logit"`. Setting the argument `transform == "arcsin"` uses the transformation $Y_{N \times p} = 2/\pi \times \mathrm{arcsin}\sqrt{Y}$ further described in Anderson et al. (2007). From here forward, the variable $Y_{N \times p}$ should be understood to be the transformation of `Y`, where N is the total number of rows in `Y` and p is the number of discriminant columns in `Y`.

The Model:

The B-ECM model structure can be found in a future publication, with some details from this publication are reproduced here.

B-ECM assumes that all data is generated using a mixture of K normal distributions, where K is equal to the number of unique event categories. Each component of the mixture has a unique mean of μ_k, and covariance of Σ_k, where $k \in \{1, \ldots, K\}$ indexes the mixture component. The likelihood of the i^{th} event observation y^i_p of p discriminants can be written as the sum below.

$$\sum_{k=1}^{K} \pi_k \mathcal{N}(y^i_p; \mu_k, \Sigma_k)$$

Each Gaussian distribution in the sum is weighted by the scalar variable π_k, where $\sum_{k=1}^{K} \pi_k = 1$ so that the density integrates to 1.

There are prior distributions on each μ_k, Σ_k, and π, where π is the vector of mixture weights $\{\pi_1, \ldots, \pi_K\}$. These prior distributions are detailed below. These parameters are important for understanding the model, however they are integrated out analytically to reduce computation time, resulting in a marginal likelihood $p(Y_{N_k \times p} | \eta_k, \Psi_k, \nu_k)$ which is a mixture of matrix t-distributions. $Y_{N_k \times p}$ is a matrix of the total data for the k^{th} event category containing N_k total event observations for training. The totality of the training data can be written as $Y_{N \times p}$, where $N = N_1 + \cdots + N_K$.

`BayesECM()` can handle observations where some of the p discriminants of an observation are missing. The properties of the conditional matrix t-distribution are used to impute the missing values, thereby accounting for the uncertainty related to the missing data.

Prior Distributions:

The posterior distributions $p(\mu_k | Y_{N_k \times p}, \eta_k)$, $p(\Sigma_k | Y_{N_k \times p}, \Psi_k, \nu_k)$, and $p(\pi | Y_{N \times p}, \alpha)$ are dependent on the specifications of prior distributions $p(\mu_k | \Sigma_k, \eta_k)$, $p(\Sigma_k | \Psi_k, \nu_k)$, and $p(\pi | \alpha)$.

$p(\mu_k | \Sigma_k, \eta_k)$ is a multivariate normal distribution with a mean vector of η_k and is conditional on the covariance Σ_k. $p(\Sigma_k | \Psi_k, \nu_k)$ is an Inverse Wishart distribution with degrees of freedom parameter ν_k, or `nu`, and scale matrix Ψ_k, or `Psi`. $p(\pi | \alpha)$ is a Dirichlet distribution with the parameter vector α of length K.

The ability to use `"default"` priors has been included for ease of use with various settings of the `priors` function argument. The default prior hyperparameter values differ for the argument of `transform` used, and the values can be inspected by examining the output of the `BayesECM()` function. Simply setting `priors = "default"` provides the same default values for all η_k, Ψ_k, ν_k in the mixture. If all prior parameters are to be shared between all event categories, but some non-default values are desirable then supplying a list of a similar structure as `priors = list(eta = rep(0, times = ncol(Y) - 1), Psi = "default", nu = "default", alpha = 10)` can be used, where setting a list element `"default"` can be exchanged for the correct data structure for the relevant data structure.

If one wishes to use some default values, but not share all parameter values between each event category, or wishes to specify each parameter value individually with no defaults, we suggest running and saving the output `BayesECM(Y = Y, BT = c(1,2))$priors`. Note that when specifying `eta` or `Psi` it is necessary that the row and column order of the supplied values corresponds to the column order of `Y`.

Value

Returns an object of `class("BayesECM")`. If there are missing data in the supplied argument `Y` the object contains Markov-chain Monte-Carlo samples of the imputed missing data. Prior distribution parameters used are always included in the output. The primary use of an object returned from `BayesECM()` is to later use this object to categorize unlabeled data with the `predict.BayesECM()` function.

Examples

```
csv_use <- "good_training.csv"
file_path <- system.file("extdata", csv_use, package = "ezECM")
training_data <- import_pvals(file = file_path, header = TRUE, sep = ",", training = TRUE)

trained_model <- BayesECM(Y = training_data)
```

`becm_decision` *B-ECM performance metrics*

Description

Outputs batch performance metrics of decisions using the output of `predict.BayesECM()` when the true category of testing events are known. Can be used for empirically comparing different model fits.

Usage

```
becm_decision(
  bayes_pred = NULL,
  vic = NULL,
  cat_truth = NULL,
  alphatilde = 0.05,
  pn = TRUE,
  C = matrix(c(0, 1, 1, 0), ncol = 2),
  rej = NULL
)
```

Arguments

bayes_pred	An object of class "BayesECMpred" returned from the predict.BayesECM() function. Data and order of data provided to predict.BayesECM() must match the order of the cat_truth argument.
vic	Character string, indicating the "Very Important Category" (vic) used for calculating categorization metrics. The return of becm_decision() provides information on if new observations should be categorized into vic or the remainder of the training categories grouped together.
cat_truth	Vector of the same length as nrow(bayes_pred$Ytilde), where Ytilde was previously supplied to the predict.BayesECM() function. Used for accuracy calculations of a BayesECM fit and decision criteria when the true category of the new observations are known.
alphatilde	Numeric scalar between 0 and 1 used for the significance level of typicality indices.
pn	Logical. Acronym for "false Positive, false Negative". pn == FALSE indicates that only accuracy of categorizations should be returned, while pn == TRUE indicates that false positives and false negatives should be returned in addition to accuracy.
C	Square matrix of dimension 2 or the number of categories. Used as the loss function in the decision theoretic framework. The default is 0-1 loss for binary categorization.
rej	data.frame of rejection via typicality index retrieved from a previous call to becm_decision(). Useful for saving computation time when comparing the results from different supplied matrices for argument C, while using a constant value of alphatilde.

Details

The matrix C specifies the loss for a set of categorization actions for a single new observation $\tilde{y}_{\tilde{p}}$ given the probability of $\tilde{y}_{\tilde{p}}$ belonging to each of K categories. Actions are specified as the columns, and the event category random variables are specified as the rows. See the vignette vignette("syn-data-code", package = "ezECM") for more mathematical details.

The dimension of matrix C specifies the categorization type. A dimension of 2 is binary categorization, with the first column and row always corresponding to the category chosen as the vic argument. Otherwise, when the dimension of C is equal to the number of categories, the indices of the rows and columns of C are in the same order as the categories listed for names(bayes_pred$BayesECMfit$Y).

Value

A list of two data frames of logicals. The rows in each data frame correspond to the rows in bayes_pred$Ytilde. The first data frame, named results, has three columns named correct, fn, and fp. The results column indicates if the categorization is correct. fn and fp stand for false negatives and false positives respectively. fn and fp are found under binary categorization. Values of NA are returned when a false positive or false negative is not relevant. The second data frame, named rej, indicates the rejection of each new observation from each category via typicality index. One can use rej to inspect if the typicality index played a role in categorization, or to supply to another call of becm_decision.

Examples

```
csv_use <- "good_training.csv"
file_path <- system.file("extdata", csv_use, package = "ezECM")
training_data <- import_pvals(file = file_path, header = TRUE, sep = ",", training = TRUE)

trained_model <- BayesECM(Y = training_data, BT = c(10,1000))

csv_use <- "good_newdata.csv"
file_path <- system.file("extdata", csv_use, package = "ezECM")
new_data <- import_pvals(file = file_path, header = TRUE, sep = ",", training = TRUE)

bayespred <- predict(trained_model,  Ytilde = new_data)

accuracy <- becm_decision(bayes_pred = bayespred, vic = "explosion", cat_truth = new_data$event)
```

---

cECM *Multiple Discriminant Analysis*

Description

Fits a regularized discriminant analysis model to labeled training data and generates an aggregate p-value for categorizing newly obtained data.

Usage

```
cECM(x, newdata = NULL, rda_params = NULL, transform = TRUE)
```

Arguments

x	Either a data.frame of training data or the list output from previous use of the cECM function. If a data.frame is supplied, each row contains one point of training data. Additionally, one column, with the name of event, must contain labels for the event of each row. The remaining columns contain the observed ncol(x) - 1 discriminant related p-value data.
newdata	a data.frame containing new data for event categorization. Must use a subset of the discriminants used in the training data. If for a particular event a certain discriminant is unavailable, specifying NA in place of the p-value will allow for calculation of the aggregate p-value.

`rda_params`	a `list` of arguments passed to the `klaR::rda()` function. If arguments `rda_params$x` or `rda_params$grouping` are supplied, they are ignored.
`transform`	Logical indicating if the supplied p-values should be transformed by the function $2/\pi \times \mathrm{asin}\sqrt{X}$. Ignored if a `list` is supplied as the argument `x`.

Details

Details on regularized discriminant analysis (RDA) can be found in Friedman (1989). Details on related implementation found in Anderson et al. (2007).

Value

A list. Any returned objects contain a list element indicating the value of `transform` supplied to the `cECM` function call, as well as a `klaR::rda()` object related to relevant training data. In addition if `newdata` argument is supplied, the returned list contains a `data.frame` specifying aggregate p-values for each new event (rows) for related event category (columns).

References

Anderson DN, Fagan DK, Tinker MA, Kraft GD, Hutchenson KD (2007). “A mathematical statistics formulation of the teleseismic explosion identification problem with multiple discriminants.” *Bulletin of the Seismological Society of America*, **97**(5), 1730–1741.

Friedman JH (1989). “Regularized discriminant analysis.” *Journal of the American statistical association*, **84**(405), 165–175.

Examples

```
x <- pval_gen(sims = 20, pwave.arrival = list(optim.starts = 5))
s <- sample(1:20, size = 2)

newdata <- x[s,]
newdata <- newdata[,-which(names(newdata) == "event")]

x <- x[-s,]

pval_cat <- cECM(x = x, transform = TRUE)

pval_cat <- cECM(x = pval_cat, newdata = newdata)
```

`cECM_decision` *Decision Function for the C-ECM Model*

Description

Returns category decisions for uncategorized events. When the true category of such events are known performance metrics are returned.

Usage

```
cECM_decision(pval = NULL, alphatilde = NULL, vic = NULL, cat_truth = NULL)
```

Arguments

`pval`	Class `"cECM"` object obtained from providing training and testing data to the `cECM()` function.
`alphatilde`	Scalar numeric between 0 and 1, used as the significance level for hypothesis testing.
`vic`	Character vector of length one, indicating the Very Important Category (VIC). Used for judging accuracy, false negatives, and false positives for binary categorization. Required when `cat_truth` is supplied.
`cat_truth`	Character vector corresponding to the true group of each row in the `newdata` argument, previously provided to the `cECM()` function. When supplied, along with `vic`, binary categorization accuracy is returned.

Details

When `is.null(cat_truth) == TRUE`, categorization over all event categories is used, using the same framework seen in (Anderson et al. 2007). The return value of `"indeterminant"` happens when there is a failure to reject multiple event categories. `"undefined"` is returned when all event categories are rejected.

When the arguments `cat_truth` and `vic` are included, binary categorization is utilized instead of categorization over all training categories. The definition of accuracy is more ambiguous when categorizing over all training categories and the user is encouraged to develop their own code for such a case. The goal of binary categorization is to estimate if an uncategorized observation is or is not in the event category stipulated by `vic`. Uncategorized events which are `"indeterminant"` or `"undefined"` are deemed to not be in the `vic` category.

Value

When `is.null(cat_truth) == TRUE` a vector providing the categorization over all event categories is returned. When the `cat_truth` and `vic` arguments are supplied a list is returned containing a `data.frame` detailing if each event was categorized accurately in a binary categorization framework, was a false positive, was a false negative, and the estimated event category. A vector stating overall categorization accuracy, false positive rate, and false negative rate is included with the list.

References

Anderson DN, Fagan DK, Tinker MA, Kraft GD, Hutchenson KD (2007). “A mathematical statistics formulation of the teleseismic explosion identification problem with multiple discriminants.” *Bulletin of the Seismological Society of America*, **97**(5), 1730–1741.

Examples

```
file_path <- system.file("extdata", "good_training.csv", package = "ezECM")
training_data <- import_pvals(file = file_path, header = TRUE, sep = ",", training = TRUE)
```

```
newdata <- training_data[1:10,]
cat_truth <- newdata$event
newdata$event <- NULL
training_data <- training_data[-(1:10),]

pval <- cECM(training_data, transform = TRUE, newdata = newdata)

binary_decision <- cECM_decision(pval = pval, alphatilde = 0.05,
vic = "explosion", cat_truth = cat_truth)

decision <- cECM_decision(pval = pval, alphatilde = 0.05)
```

`export_ecm` *Saving and loading fitted ECM models*

Description

`export_ecm()` saves an ECM model fit to training data for later use. The object can be the output of the `cECM()` or `BayesECM()` functions. Object saved in user specified location.

`import_ecm()` loads a previously exported ECM model into the global environment.

Usage

```
export_ecm(x = NULL, file = stop("'file' must be specified"), verb = TRUE)

import_ecm(file = NULL, verb = TRUE)
```

Arguments

`x`	Name of ECM model to be exported. `class(x)` should be either `"cECM"` or `"BayesECM"`.
`file`	Character string containing absolute or relative path of `*.rda` file to be exported or imported.
`verb`	Logical indicating if a message indicating success should be printed to the console.

Value

Saves file or imports file into global environment.

Examples

```
x <- pval_gen(sims = 20, pwave.arrival = list(optim.starts = 5))
pval_cat <- cECM(x = x, transform = TRUE)

export_ecm(x = pval_cat, file = "example-ecm.rda", verb = FALSE)
```

```
reload_pval_cat <- import_ecm(file = "example-ecm.rda")

## The below code shouldn't be used,
## it deletes the file created during the example

unlink("example-ecm.rda")
```

---

import_pvals *Import p-values*

Description

Imports and organizes observed p-values located in a `*.csv` file for training or categorization using an existing model.

Usage

```
import_pvals(file = NULL, header = TRUE, sep = ",", training = TRUE)
```

Arguments

`file`	Character string providing name of `*.csv` file to be imported. File name within current working directory or absolute path are acceptable. Argument passed directly to `utils::read.csv()`, see `utils::read.csv()` for details.
`header`	Logical indicating if first row of supplied `*.csv` file contains column names. See `utils::read.csv()` for details.
`sep`	Character string indicating the field separator character for the supplied `*.csv` file.
`training`	Logical indicating if the supplied `*.csv` file is to be used as training data. Only serves to suppress warnings.

Details

The purpose of this function is to give the user a way to import p-value data in which the data will be organized for use with the `cECM()` and `BayesECM()` functions. Warnings are printed when potential issues may arise with the supplied file, and the function attempts to detect and correct simple formatting issues.

Ideally, the user supplies a `*.csv` file which contains a header row labeling the columns. Each column contains the p-values of a particular discriminant, and each row must correspond to a single event. If training data is to be imported, the column containing known event categories is labeled `"event"`. If new data is imported to be used with an existing model fit, the order of the columns in the `new.data` file must be the same as the `*.csv` file containing training data.

Value

A `base::data.frame()` of p-values with each row corresponding to a single event and each column corresponding to a particular discriminant. If data labels are correctly provided in the supplied `*.csv` file, an additional column labeled `event` will hold these values.

Examples

```
file_path <- system.file("extdata", "good_training.csv", package = "ezECM")
training_data <- import_pvals(file = file_path, header = TRUE, sep = ",", training = TRUE)
```

---

plot.cECM — *Plot the data and output of `cECM()` categorization*

Description

Plot the data and output of `cECM()` categorization

Usage

```
## S3 method for class 'cECM'
plot(x, discriminants = c(1, 2), thenull = NULL, alphatilde = 0.05, ...)
```

Arguments

`x`	an object of which is of class "cECM", retrieved from the output of the `cECM()` function. The "cECM" object may or may not contain unlabeled data.
`discriminants`	character or integer vector of length two. If a character vector is provided, the character strings must match a subset of the column names for the training data, ie. `all(discriminants %in% names(x$x))` must be true. If an integer vector is given, the elements of the vector select the column indices of the data to be plotted.
`thenull`	character string or `NULL`. When unlabeled data is found within an "cECM" object, the name of one of the event categories can be provided as this argument. When "thenull" is provided, unlabled data where the category hypothesis is rejected is colored red.
`alphatilde`	numeric value specifying hypothesis testing significance level. Used in conjunction with `thenull`, aggregate p-values less than `alphatilde` are rejected and colored accordingly.
`...`	arguments passed to `base::plot()`

Details

The plot generated from plot.ecm() is first dependent on if the provided `x$newdata` contains a data frame of unlabled data.

If unlabled data is not part of the `"cECM"` object, the labled data is simply plotted with the 0.68 and 0.95 confidence levels obtained from the distribution fits returned from `cECM()`.

If unlabled data **is** part of the `"cECM"` object, the unlabled data is plotted in addition to the distribution plots. Each unlabled data point appears on the plot as an integer, which indexes the corresponding row of `x$newdata`.

Value

Plot illustrating results of `cECM()`

Examples

```
x <- pval_gen(sims = 20, pwave.arrival = list(optim.starts = 5))
s <- sample(1:20, size = 2)

newdata <- x[s,]
newdata <- newdata[,-which(names(newdata) == "event")]

x <- x[-s,]

pval_cat <- cECM(x = x, transform = TRUE)

pval_cat <- cECM(x = pval_cat, newdata = newdata)

plot(x = pval_cat, thenull = "explosion")
```

`predict.BayesECM` *New Event Categorization With Bayesian Inference*

Description

New Event Categorization With Bayesian Inference

Usage

```
## S3 method for class 'BayesECM'
predict(object, Ytilde, thinning = 1, mixture_weights = "training", ...)
```

Arguments

`object`	an object of class `"BayesECM"` obtained as the trained model output using the `BayesECM()` function.

`Ytilde`	`data.frame` of unlabeled observations to be categorized. Must contain the same discriminant names as the training data used in the provided "BayesECM" object. Each row is an individual observation. Missing data is specified with `NA`
`thinning`	integer, scalar. Values greater than one can be provided to reduce computation time. See details.
`mixture_weights`	character string describing the weights of the distributions in the mixture to be used for prediction. The default,`"training"` will utilize weights according to likelihood and prior specifications, while supplying the string `"equal"` will assume the marginal predictive distribution of each category is independent of the data and utilize equal weights.
`...`	not used

Details

The data in `Ytilde` should be the p-values $\in (0, 1]$. The transformation applied to the data used to generate `object` is automatically applied to `Ytilde` within the `predict.BayesECM()` function.

For a given event with an unknown category, a Bayesian ECM model seeks to predict the expected value of the latent variable $\tilde{\mathbf{z}}_K$, where $\tilde{\mathbf{z}}_K$ is a vector of the length K, and K is the number of event categories. A single observation of $\tilde{\mathbf{z}}_K$ is a draw from a Categorical Distribution.

The expected probabilities stipulated within the categorical distribution of $\tilde{\mathbf{z}}_K$ are conditioned on any imputed missing data, prior hyperparameters, and individually each row of `Ytilde`. The output from `predict.BayesECM()` are draws from the distribution of $\mathbf{E}[\tilde{\mathbf{z}}_K|\tilde{\mathbf{y}}_{\tilde{p}}, \mathbf{Y}^+, \eta, \mathbf{\Psi}, \nu, \alpha] = p(\tilde{\mathbf{z}}_K|\tilde{\mathbf{y}}_{\tilde{p}}, \mathbf{Y}^+, \eta, \mathbf{\Psi}, \nu, \alpha)$, where $\mathbf{Y}^+$ represents the observed values within the training data.

The argument `mixture_weights` controls the value of $p(\tilde{\mathbf{z}}_K|\mathbf{Y}_{N\times p}, \alpha)$, the probability of each $\tilde{z}_k = 1$, before $\tilde{\mathbf{y}}_{\tilde{p}}$ is observed. The standard result is obtained from the prior hyperparameter values in α and the number of unique events in each $\mathbf{Y}_{N_k\times p}$. Setting `mixture_weights = "training"` will utilize this standard result in prediction. If the frequency of the number events used for each category in training is thought to be problematic, providing the argument `mixture_weights = "equal"` sets $p(\tilde{z}_1 = 1|\mathbf{Y}_{N\times p}) = \cdots = p(\tilde{z}_K = 1|\mathbf{Y}_{N\times p}) = 1/K$. If the user wants to use a set of $p(\tilde{z}_k = 1|\mathbf{Y}_{N\times p})$ which are not equal but also not informed by the data, we suggest setting the elements of the hyperparameter vector α equal to values with a large magnitude and in the desired ratios for each category. However, this can cause undesirable results in prediction if the magnitude of some elements of α are orders larger than others.

To save computation time, the user can specify an integer value for `thinning` greater than one. Every `thinning`th Markov-chain Monte-Carlo sample is used for prediction. This lets the user take a large number of samples during the training step, allowing for better mixing. See details in a package vignette by running `vignette("syn-data-code", package = "ezECM")`

Value

Returns a `list`. The list element `epz` is a matrix with `nrow(Ytilde)` rows, corresponding to each event used for prediction, and `K` named columns. Each column of `epz` is the expected category probability of the row stipulated event. The remainder of the list elements hold data including `Ytilde`, information about additonal variables passed to `predict.BayesECM`, and data related to the previous `BayesECM()` fit.

Examples

```
csv_use <- "good_training.csv"
file_path <- system.file("extdata", csv_use, package = "ezECM")
training_data <- import_pvals(file = file_path, header = TRUE, sep = ",", training = TRUE)

trained_model <- BayesECM(Y = training_data, BT = c(10,1000))

csv_use <- "good_newdata.csv"
file_path <- system.file("extdata", csv_use, package = "ezECM")
new_data <- import_pvals(file = file_path, header = TRUE, sep = ",", training = TRUE)

bayes_pred <- predict(trained_model,  Ytilde = new_data)
```

`pval_gen` *Simulate p-values from earthquakes and explosions*

Description

p-values are simulated for depth and first polarity discriminants to use in testing classification models.

Usage

```
pval_gen(
  sims = 100,
  grid.dim = c(800, 800, 30),
  seismometer = list(N = 100, max.depth = 2, Sig = NULL, sig.draws = c(15, 2)),
  explosion = list(max.depth = 5, prob = 0.4),
  pwave.arrival = list(H0 = 5, V = 5.9, optim.starts = 15),
  first.polarity = list(read.err = 0.95)
)
```

Arguments

`sims`	numeric stipulating the number of individual events to generate.
`grid.dim`	numeric 3-vector providing the extent of the coordinate system in `c(X,Y,Z)` to be used in km.
`seismometer`	list stipulating variables relating to the seismometers. Providing an incomplete list reverts to the default values. List elements are: • `N` a numeric stipulating the number of seismometers • `max.depth` is a numeric providing the maximum depth for a seismometer location in km • `Sig` supplying a numeric vector of length `N` to this argument will stipulate the variance in the error in observed arrival time of each of the `N` seismometers.

- `sig.draws` a numeric 2-vector, if `Sig` is not provided the variance in arrival time at each station is drawn from `MCMCpack::rinvgamma()` using shape = `sig.draws[1]` and scale = `sig.draws[2]`.

`explosion` list stipulating variables regarding a detonation event. Providing an incomplete list reverts to the default values. List elements are:

- `max.depth` is a numeric providing the maximum depth of an explosion in km
- `prob` is the probability of an explosion, controlling the fraction of events which are explosions. Value provided must be in the interval $[0, 1]$

`pwave.arrival` list stipulating variables regarding the depth from p-wave arrival discriminant. Providing an incomplete list reverts to the default values. List elements are:

- `H0` a numeric providing the value of depth in km for the null hypothesis
- `V` a numeric stipulating the velocity of p-waves in km. Used for simulating p-wave arrival times as an argument of `time_fn()`.
- `optim.starts` number of `stats::optim()` starts used to maximize likelihood of event location.

`first.polarity` list stipulating variables regarding the depth from the polarity of first motion discriminant. List elements are:

- `read.err` numeric in the interval $[0, 1]$ providing the probability of an error in reading the true first polarity.

Details

Methods are adapted from the discriminants listed in (Anderson et al. 2007).

Value

A data frame, with the number of rows equal to `sims`. Columns contain the p-values observed for each simulation along with the true event type.

Depth Discriminant

The equation below is used to model p-wave arrival time t_i at seismometer i.

$$t_i = t_0 + T(S_i, S_0) + \epsilon_i$$

Where t_0 is the time of the event, $T()$ is a function modeling the arrival time (in this case time_fn), S_i is the location of seismometer i, S_0 is the location of the event, and ϵ_i is normally distributed error with known variance σ_i^2. Given `N` seismometers, the MLE of the event time $\hat{t}_0$ can be solved as the following:

$$\hat{t}_0 = \frac{\mathrm{tr}\left(\Sigma^{-1} T_i\right) - \mathrm{tr}\left(\Sigma^{-1} T(S, S_0)\right)}{\mathrm{tr}(\Sigma^{-1})}$$

Where $\mathrm{tr}()$ is the matrix trace operation, Σ is a matrix containing the elements of each σ_i^2 on the diagonal, T_i is a diagonal matrix containing each t_i, and $T(S, S_0)$ is a diagonal matrix containing each $T(S_i, S_0)$. This result is then plugged back into the first equation, which is then used in a normal likelihood function. Derivatives are taken of the likelihood so that a fast gradient based approch can be used to find the maximum likelihood estimate (MLE) of S_0.

The remainder of the calculation of the p-value is consistent with the *Depth from P-Wave Arrival Times* section of (Anderson et al. 2007). First note S_0 is equal to the 3-vector $(X_0, Y_0, Z_0)^\top$. Given the null hypothesis for the depth of $\mathrm{H}_0 : Z_0 \leq z_0$, the MLE $(\hat{X}_0, \hat{Y}_0)$ given $Z_0 = z_0$ is found. The sum of squared errors (SSE) is calculated as follows:

$$\mathrm{SSE}(S_0, t_0) = \sum_{i=1}^{N} \left(\frac{t_i - t_0 - T(S_i, S_0)}{\sigma_i} \right)^2$$

If H_0 is true then the following statistic has a central F distribution with 1 and $N - 4$ degrees of freedom.

$$F_{1,N-4} = \frac{\mathrm{SSE}(S_0, t_0 | Z_0 = z_0) - \mathrm{SSE}(S_0, t_0)}{\mathrm{SSE}(S_0, t_0)}$$

Because the test has directionality, the F statistic is then converted to a T statistic as such:

$$T_{N-4} = \mathrm{sign}(\hat{Z}_0 - z_0)\sqrt{F_{1,n-4}}$$

This T statistic is then used to compute the p-value

Polarity of First Motion

Under the null hypothesis that the event is an explosion, (and therefore the true polarity of first motion is always one), the error rate for mistakenly identifying the polarity of first motion is stipulated as the argument `first.polarity$read.err`. For an error rate of θ the p-value can then be calculated as follows:

$$\sum_{i=0}^{n} \binom{N}{i} \theta^i (1 - \theta)^{N-i}$$

Where n is the number of stations where a positive first motion was observed, and N is the total number of stations.

References

Anderson DN, Fagan DK, Tinker MA, Kraft GD, Hutchenson KD (2007). “A mathematical statistics formulation of the teleseismic explosion identification problem with multiple discriminants.” *Bulletin of the Seismological Society of America*, **97**(5), 1730–1741.

Examples

```
test.data <- pval_gen(sims = 5)
```

`P_wave_gen` *Generation of noisy p-wave arrival times*

Description

Similar utility to time_fn, however multiple seismometer locations can be provided simultaneously and normally distributed noise is added to the arrival time.

Usage

```
P_wave_gen(
  Si = NULL,
  S0 = NULL,
  Sig = NULL,
  neg.obs = TRUE,
  eps = sqrt(.Machine$double.eps)
)
```

Arguments

`Si`	Numeric matrix providing seismometer locations. Must contain 3 columns corresponding to (X,Y) corrdinates and depth.
`S0`	Numeric 3 element vector stipulating the location of an event, elements correspond to (X, Y, Z)
`Sig`	Numeric vector, or diagonal matrix, providing the variance in observed arrival times at each seismometer.
`neg.obs`	Logical indicating whether to allow negative observations of time (eg. the observed time of p-wave arrival is before the true time for the event).
`eps`	Numeric. If `neg.obs = FALSE` sets of observations are redrawn until all $t_i - t_0 \leq$ `eps`.

Value

Numeric vector of observation times that correspond to the rows of `Si`

Examples

```
pwave.obs <- P_wave_gen(Si = c(100,200,3), S0 = c(400, 500, 4), Sig = 0.05)
```

`summary.BayesECMpred` *Summary of Unlabeled Event Categorization*

Description

Tabulates results from the `predict.BayesECM()` function for quick analysis.

Usage

```
## S3 method for class 'BayesECMpred'
summary(object, index = 1, category = 1, C = 1 - diag(2), ...)
```

Arguments

`object`	an object of `class` `"BayesECMpred"` obtained as the output from the `predict.BayesECM()` function.
`index`	integer stipulating the event of interest. Value corresponds to the row index of `Ytilde` previously supplied to `predict.BayesECM()`
`category`	integer for the index of the category of interest for hypothesis testing. Alternatively, a character string naming the category of interest can be provided.
`C`	square matrix of dimension 2, providing loss values to be used in hypothesis testing. See Details.
`...`	not used

Details

Expected Loss:

`summary.BayesECMpred()` prints expected loss for the binary hypothesis stipulated by `category`. Expected loss is calculated using the loss matrix specified with argument `C`. The default values for `C` result in 0-1 loss being used. Format details for the loss matrix can be found in `vignette("syn-data-code")`.

Typicality:

Typicality indices are used in `cECM_decision()` as part of the decision criteria. Here, we have adapted typicality indices for use with a Bayesian ECM model for outlier detection, when a new observation may not be related to the categories used for training. Probability of the p-value being less than a significance level of 0.05 is reported. If no missing data is used for training, this probability is either 0 or 1.

Value

Prints a summary including probability of each category for the event stipulated by `index`, minimum expected loss for binary categorization, and probability of a-typicality of the event for the category specified by `category`.

Examples

```
csv_use <- "good_training.csv"
file_path <- system.file("extdata", csv_use, package = "ezECM")
training_data <- import_pvals(file = file_path, header = TRUE, sep = ",", training = TRUE)

trained_model <- BayesECM(Y = training_data, BT = c(10,1000))

csv_use <- "good_newdata.csv"
file_path <- system.file("extdata", csv_use, package = "ezECM")
new_data <- import_pvals(file = file_path, header = TRUE, sep = ",", training = TRUE)

bayespred <- predict(trained_model,  Ytilde = new_data)
```

`time_fn` *P-wave arrival times simulation*

Description

Simulates the arrival time of p-waves (without noise) using a mean velocity and euclidian distance, without taking the curvature into account.

Usage

```
time_fn(X = NULL, X0 = NULL, V = 7)
```

Arguments

`X`	numeric three element vector stipulating the location of a seismometer. Units of km
`X0`	numeric three element vector stipulating the location of an event. Units of km
`V`	numeric scalar supplying the p-wave velocity in km/s. A quick google search (Jan 31st 2023) shows typical values range from 5 to 8 km/s.

Details

Used for estimating event location, given a series of seismometer observations.

Value

numeric scalar providing the time in seconds for a p-wave to arrive at a seismometer.

Examples

```
  arrival.time <- time_fn(X = c(100,200,10), X0  = c(500, 80, 25), V = 5)
```

Index