Package 'fdaoutlier'

October 1, 2023

Title Outlier Detection Tools for Functional Data Analysis

Version 0.2.1

Description A collection of functions for outlier detection in functional data analysis. Methods implemented include directional outlyingness by Dai and Genton (2019) <doi:10.1016/j.csda.2018.03.017>, MS-plot by Dai and Genton (2018) <doi:10.1080/10618600.2018.1473781>, total variation depth and modified shape similarity index by Huang and Sun (2019) <doi:10.1080/00401706.2019.1574241>, and sequential transformations by Dai et al. (2020) <doi:10.1016/j.csda.2020.106960 among others. Additional outlier detection tools and depths for functional data like functional boxplot, (modified) band depth etc., are also available.

License GPL-3

URL https://github.com/otsegun/fdaoutlier

BugReports https://github.com/otsegun/fdaoutlier/issues

Encoding UTF-8

LazyData true

Suggests testthat (>= 2.1.0), covr, spelling, knitr, rmarkdown

RoxygenNote 7.2.3

Imports MASS

Depends R (>= 2.10)

Language en-US

VignetteBuilder knitr

NeedsCompilation yes

Author Oluwasegun Taiwo Ojo [aut, cre, cph] (<https://orcid.org/0000-0001-9629-6990>), Rosa Elvira Lillo [aut], Antonio Fernandez Anta [aut, fnd]

Maintainer Oluwasegun Taiwo Ojo <seguntaiwoojo@gmail.com>

Repository CRAN

Date/Publication 2023-09-30 22:40:08 UTC

R topics documented:

`band_depth` *Compute the band depth for a sample of curves/observations.*

Description

This function computes the band depth of López-Pintado and Romo (2009) doi:10.1198/jasa.2009.0108. Bands of 2 functions are always considered using the fast algorithm of Sun et al. (2012) doi:10.1002/sta4.8.

Usage

```
band_depth(dt)
```

Arguments

`dt`	A matrix or data frame of size n observations/curves by p domain/evaluation points.

Value

A numeric vector of size `nrow(dt)` containing the band depth values of each curve.

References

López-Pintado, S., & Romo, J. (2009). On the Concept of Depth for Functional Data. *Journal of the American Statistical Association*, 104(486), 718-734.

Sun, Y., Genton, M. G., & Nychka, D. W. (2012). Exact fast computation of band depth for large functional datasets: How quickly can one million curves be ranked?. *Stat*, 1(1), 68-74.

Examples

```
dt1 <- simulation_model1()
bd2 <- band_depth(dt = dt1$data)
```

`directional_quantile` *Compute directional quantile outlyingness for a sample of discretely observed curves*

Description

The directional quantile is a measure of outlyingness based on a scaled pointwise deviation from the mean. These deviations are usually scaled by the deviation of the mean from the 2.5% upper and lower quantiles depending on if the (pointwise) observed value of a function is above or below the (pointwise) mean. Directional quantile was mentioned in Myllymäki et al. (2015) doi:10.1016/j.spasta.2014.11.004, Myllymäki et al. (2017) doi:10.1111/rssb.12172 and Dai et al. (2020) doi:10.1016/j.csda.2020.106960.

Usage

```
directional_quantile(dt, quantiles = c(0.025, 0.975))
```

Arguments

`dt`	A matrix or dataframe of size n observations/curves by p domain/evaluation points.
`quantiles`	A numeric vector of length 2 specifying the probabilities of the lower and upper quantiles. Values must be between 0 and 1. Defaults to `c(0.025, 0.975)` as specified in Dai et al. (2020) doi:10.1016/j.csda.2020.106960.

Details

The method computes the directional quantile of a sample of curves discretely observed on common points. The directional quantile of a function/curve $X_i(t)$ is the maximum pointwise scaled outlyingness of $X_i(t)$. The scaling is done using the pointwise absolute difference between the 2.5% mean and the lower (and upper) quantiles. See Dai et al. (2020) doi:10.1016/j.csda.2020.106960 and Myllymäki et al. (2017) doi:10.1111/rssb.12172 for more details.

Value

A numeric vector containing the the directional quantiles of each observation of `dt`.

Author(s)

Oluwasegun Taiwo Ojo

References

Dai, W., Mrkvička, T., Sun, Y., & Genton, M. G. (2020). Functional outlier detection and taxonomy by sequential transformations. *Computational Statistics & Data Analysis*, 106960.

Myllymäki, M., Mrkvička, T., Grabarnik, P., Seijo, H., & Hahn, U. (2017). Global envelope tests for spatial processes. *J. R. Stat. Soc. B*, 79:381-404.

Examples

```
dt1 <- simulation_model1()
dq <- directional_quantile(dt1$data)
```

---

dir_out *Dai & Genton (2019) Directional outlyingness for univariate or multivariate functional data.*

Description

Compute the directional outlyingness of a univariate or multivariate functional data based on Dai and Genton (2019) doi:10.1016/j.csda.2018.03.017 and Dai and Genton (2018) doi:10.1080/10618600.2018.1473781.

Usage

```
dir_out(
  dts,
  data_depth = "random_projections",
  n_projections = 200L,
  seed = NULL,
  return_distance = TRUE,
  return_dir_matrix = FALSE
)
```

Arguments

`dts`	A matrix (or data frame) for univariate functional data (of size `n` observations by `p` domain points) or a 3-dimensional array for multivariate functional data (of size `n` observations by `p` domain points by `d` dimension).

`data_depth` The method for computing the depth. The random projection depth is always used as suggested in Dai and Genton (2018) doi:10.1080/10618600.2018.1473781. Support for more depth methods will be added.

`n_projections` The number of directions for computing random projection depth. By default, 200 random directions are generated from a scaled uniform distribution between -1 and 1.

`seed` An integer indicating the seed to set when generating the random directions. Defaults to NULL in which case a seed is not set.

`return_distance`
A logical value. If TRUE, returns: a matrix whose columns are the mean and variation of directional outlyingness, the mahalanobis distance of the observations of this matrix, and the robust estimate of the mean and covariance of this matrix (computed using the minimum covariance determinant method).

`return_dir_matrix`
A logical value. If TRUE, returns the directional outlyingness matrix (or array for multivariate functional data). Computed from the chosen `depth_depth`.

Details

The directional outlyingness, as defined in Dai and Genton (2019) doi:10.1016/j.csda.2018.03.017 is

$$O(Y, F_Y) = (1/d(Y, F_Y) - 1).v$$

where d is a depth notion, and v is the unit vector pointing from the median of F_Y to Y. For univariate and multivariate functional data, the projection depth is always used as suggested by Dai and Genton (2019) doi:10.1016/j.csda.2018.03.017.

Value

Returns a list containing:

`mean_outlyingness`
an n x d matrix of the mean of directional outlyingness.

`var_outlyingness`
a vector of length n containing the variation of directional outlyingness.

`ms_matrix` if`return_distance` = T, an n x `(d+1)` matrix whose columns are the mean and variation of directional outlyingness.

`distance` if `return_distance` = T, a vector of distance computed from the `ms_matrix` using a robust estimate of the mean and covariance matrix.

`mcd_obj` if `return_distance` = T, a list containing the robust (minimum covariance determinant) estimate of the mean and covariance of the `ms_matrix`.

`dirout_matrix` if `return_dir_matrix` = T, an n x p (or n x p x d) matrix (or array) containing the directional outlyingness values for the univariate (or multivariate) functional `dts`.

Author(s)

Oluwasegun Taiwo Ojo.

References

Dai, W., and Genton, M. G. (2018). Multivariate functional data visualization and outlier detection. *Journal of Computational and Graphical Statistics*, 27(4), 923-934.

Dai, W., and Genton, M. G. (2019). Directional outlyingness for multivariate functional data. *Computational Statistics & Data Analysis*, 131, 50-65.

Zuo, Y. (2003). Projection-based depth functions and associated medians. *The Annals of Statistics*, 31(5), 1460-1490.

See Also

`msplot` for outlier detection using msplot and `projection_depth` for multivariate projection depth.

Examples

```
# univariate magnitude model in Dai and Genton (2018).
dt4 <- simulation_model4()
dirout_object <- dir_out(dts = dt4$data, return_distance = TRUE)
```

---

extremal_depth *Compute extremal depth for functional data*

Description

Compute extremal depth for functional data

Usage

```
extremal_depth(dts)
```

Arguments

`dts`	A numeric matrix or dataframe of size n observations/curves by p domain/evaluation points.

Details

This function computes the extremal depth of a univariate functional data. The extremal depth of a function g with respect to a set of function S denoted by $ED(g, S)$ is the proportion of functions in S that is more extreme than g. The functions are ordered using depths cumulative distribution functions (d-CDFs). Extremal depth like the name implies is based on extreme outlyingness and it penalizes functions that are outliers even for a small part of the domain. Proposed/mentioned in Narisetty and Nair (2016) doi:10.1080/01621459.2015.1110033.

Value

A vector containing the extremal depths of the rows of `dts`.

Author(s)

Oluwasegun Ojo

References

Narisetty, N. N., & Nair, V. N. (2016). Extremal depth for functional data and applications. *Journal of the American Statistical Association*, 111(516), 1705-1714.

@seealso `total_variation_depth` for functional data.

Examples

```
dt3 <- simulation_model3()
ex_depths <- extremal_depth(dts = dt3$data)
# order functions from deepest to most outlying
order(ex_depths, decreasing = TRUE)
```

`extreme_rank_length` *Compute the Extreme Rank Length Depth.*

Description

This function computes the extreme rank length depth (ERLD) of a sample of curves or functions. Functions have to be discretely observed on common domain points. In principle, the ERLD of a function X_i is the proportion of functions in the sample that is considered to be more extreme than X_i, an idea similar to `extremal_depth`. To determine which functions are more extreme, pointwise ranks of the functions are computed and compared pairwise.

Usage

```
extreme_rank_length(
  dts,
  type = c("two_sided", "one_sided_left", "one_sided_right")
)
```

Arguments

`dts`	A matrix or data frame of size n observations/curves by p domain/evaluation points.
`type`	A character value. Can be one of `"two_sided"`, `"one_sided_left"` or `"one_sided_right"`. If `"two_sided"` is specified, small and large values in `dts` will be considered extreme. If `"one_sided_left"` is specified, then only small values in `dts` are considered to be extreme while for `"one_sided_right"`, only large values in `dts` are considered to be extreme. `"two_sided"` is the default. See `Details` for more details.

Details

There are three possibilities in the (pairwise) comparison of the pointwise ranks of the functions. First possibility is to consider only small values as extreme (when `type = "one_sided_left"`) in which case the raw pointwise ranks r_{ij} are used. The second possibility is to consider only large values as extreme (when `type = "one_sided_right"`) in which case the pointwise ranks used are computed as $R_{ij} = n + 1 - r_{ij}$ where r_{ij} is the raw pointwise rank of the function i at design point j and n is the number of functions in the sample. Third possibility is to consider both small and large values as extreme (when `type = "two_sided"`) in which case the pointwise ranks used is computed as $R_{ij} = min(r_ij, n + 1 - r_{ij})$. In the computation of the raw pointwise ranks r_{ij}, ties are broken using an average. See Dai et al. (2020) doi:10.1016/j.csda.2020.106960 and Myllymäki et al. (2017) doi:10.1111/rssb.12172 for more details.

Value

A numeric vector containing the depth of each curve

Author(s)

Oluwasegun Ojo

References

Dai, W., Mrkvička, T., Sun, Y., & Genton, M. G. (2020). Functional outlier detection and taxonomy by sequential transformations. *Computational Statistics & Data Analysis*, 106960.

Myllymäki, M., Mrkvička, T., Grabarnik, P., Seijo, H., & Hahn, U. (2017). Global envelope tests for spatial processes. *J. R. Stat. Soc. B*, 79:381-404.

Examples

```
dt3 <- simulation_model3()
erld <- extreme_rank_length(dt3$data)
```

`functional_boxplot` *Functional Boxplot for a sample of functions.*

Description

This function finds outliers in a sample of curves using the functional boxplot by Sun and Genton (2011) doi:10.1198/jcgs.2011.09224. Unlike the name suggests, the function does not actually produce a plot but is only used as support in finding outliers in other functions. Different depth and outlyingness methods are supported for ordering functions. Alternatively, the depth values of the functions can be supplied directly.

Usage

```
functional_boxplot(
  dts,
 depth_method = c("mbd", "tvd", "extremal", "dirout", "linfinity", "bd", "erld", "dq"),
  depth_values = NULL,
  emp_factor = 1.5,
  central_region = 0.5,
  erld_type = NULL,
  dq_quantiles = NULL
)
```

Arguments

`dts`	A matrix or data frame of size n observations/curves by p domain/evaluation points for univariate functional data.
`depth_method`	A character value specifying the method to use for computing the depth values (if `depth_values` is not supplied) used in ordering the functions. The following methods are are supported: **"mbd":** The modified band depth with bands defined by 2 functions. Uses the algorithm of Sun et al. (2012)doi:10.1002/sta4.8. **"tvd"** The total variation depth of Huang and Sun (2019) doi:10.1080/00401706.2019.1574241. **"extremal"** The extremal depth of Narisetty and Nair (2016) doi:10.1080/01621459.2015.1110033. **"dirout"** Uses the robust distance of the mean and variation of directional outlyingness (`dir_out`) defined in Dai and Genton (2019) doi:10.1016/j.csda.2018.03.017. Since this method is a measure of outlyingness of a function the negative of the computed robust distance is used in ordering the functions. **"linfinity"** The L-infinity depth defined in Long and Huang (2015) <arXiv:1506.01332 is used in ordering functions. **"bd"** Uses the band depth with bands defined by 2 functions according to the algorithm of Sun et al. (2012) doi:10.1002/sta4.8. **erld** Uses the extreme rank length depth used in Dai et al. (2020) doi:10.1016/j.csda.2020.106960. **"dq"** Uses the directional quantile (DQ) used in Dai et al. (2020) doi:10.1016/j.csda.2020.106960. Since DQ is a measure of outlyingness, the negative of the DQ values is used in ordering the functions. The default method is `"mbd"`. Alternatively, the `depth_values` of the functions can be supplied in which case the depths are not computed and `depth_method` is ignored.
`depth_values`	A numeric vector containing the depth values of the functions in `dts` to use for ordering functions. `length(depth_values)` must be equal to number of rows of `dts`. If `depth_values` is specified, the depth is not computed and any method specified in `depth_method` is ignored.
`emp_factor`	A numeric value specifying the empirical factor for the boxplot. Defaults to 1.5.
`central_region`	A numeric value between 0 and 1 indicating the probability of central region. Defaults to 0.5.

erld_type	If depth_method = "erld", the type of ordering to use in computing the extreme rank length depth. Can be one of "two_sided", "one_sided_left" or "one_sided_right". A "two_sided" ordering is used by default if erld_type is not specified. See extreme_rank_length for more details.
dq_quantiles	If depth_method = "dq", a numeric vector of length 2 specifying the probabilities of upper and lower quantiles. Defaults to c(0.025, 0.975) for the upper and lower 2.5% quantiles. See directional_quantile for details.

Value

A list containing:

outliers	The indices of the functions/curves flagged as outliers.
depth_values	The depths of the functions/curves in dts.
median_curve	The index of the median curve, which is the curve with the largest depth value (or smallest outlyingness value).

References

Sun, Y., & Genton, M. G. (2011). Functional boxplots. *Journal of Computational and Graphical Statistics*, 20(2), 316-334.

See Also

seq_transform for functional outlier detection using sequential transformation.

Examples

```
dt1 <- simulation_model1()
fbplot_obj <- functional_boxplot(dt1$data, depth_method = "mbd")
fbplot_obj$outliers
```

---

hardin_factor_numeric *Compute F distribution factors for approximating the tail of the distribution of robust MCD distance.*

Description

Computes asymptotically, the factors for F approximation cutoff for (MCD) robust mahalanobis distances according to Hardin and Rocke (2005) doi:10.1198/106186005X77685.

Usage

```
hardin_factor_numeric(n, dimension)
```

Arguments

n	A numeric value indicating the number of observations of the data.
dimension	A numeric value indicating the number of variables of the data.

Details

This function computes the two factors needed for the determining an appropriate cutoff for robust mahalanobis distances computed using the MCD method.

The F approximation according to Hardin and Rocke (2005) doi:10.1198/106186005X77685 is given by:

$$c(m-p+1)/(pm) * RMD^2 \; F_{p,m-p+1}$$

where m is a parameter for finding the degree of freedom of the F distribution, c is a scaling constant and p is the dimension. The first factor returned by this function (`factor1`) is $c(m-p+1)/(pm)$ and the second factor (`factor2`) is $F_{p,m-p+1}$.

Value

Returns a list containing:

`factor1`	then estimated value of $c(m-p+1)/(pm)$ based on `n` and `dimension`.
`factor2`	the value of $F_{p,m-p+1}$.

References

Hardin, J., and Rocke, D. M. (2005). The distribution of robust distances. *Journal of Computational and Graphical Statistics*, 14(4), 928-946.

`linfinity_depth` *Compute the L-infinity depth of a sample of curves/functions.*

Description

The L-infinity depth is a simple generalization of the L^p multivariate depth to functional data proposed in Long and Huang (2015) <arXiv:1506.01332 and also used in Dai et al. (2020) doi:10.1016/j.csda.2020.106960.

Usage

```
linfinity_depth(dt)
```

Arguments

`dt`	A matrix or data frame of size n functions/curves by p domain/evaluation points.

Value

A numeric vector of size `nrow(dt)` containing the band depth values of each curve.

References

Long, J. P., & Huang, J. Z. (2015). A study of functional depths. *arXiv preprint arXiv*:1506.01332.

Dai, W., Mrkvička, T., Sun, Y., & Genton, M. G. (2020). Functional outlier detection and taxonomy by sequential transformations. *Computational Statistics & Data Analysis*, 106960.

Examples

```
dt1 <- simulation_model1()
linf <- linfinity_depth(dt1$data)
```

`modified_band_depth` *Compute the modified band depth for a sample of curves/functions.*

Description

This function computes the modified band depth of López-Pintado and Romo (2009) doi:10.1198/jasa.2009.0108. Bands of 2 functions are always used and the fast algorithm of Sun et al. (2012) doi:10.1002/sta4.8 is used in computing the depth values.

Usage

```
modified_band_depth(dt)
```

Arguments

`dt`	A matrix or data frame of size n functions/curves by p domain/evaluation points.

Value

A numeric vector of size `nrow(dt)` containing the band depth values of each curve.

References

López-Pintado, S., & Romo, J. (2009). On the Concept of Depth for Functional Data. *Journal of the American Statistical Association*, 104(486), 718-734.

Sun, Y., Genton, M. G., & Nychka, D. W. (2012). Exact fast computation of band depth for large functional datasets: How quickly can one million curves be ranked?. *Stat*, 1(1), 68-74.

Examples

```
dt1 <- simulation_model1()
mbd2 <- modified_band_depth(dt1$data)
```

`msplot` *Outlier Detection using Magnitude-Shape Plot (MS-Plot) based on the directional outlyingness for functional data.*

Description

This function finds outliers in univariate and multivariate functional data using the MS-Plot method described in Dai and Genton (2018) doi:10.1080/10618600.2018.1473781. Indices of observations flagged as outliers are returned. In addition, the scatter plot of VO against MO (||MO||) can be requested for univariate (multivariate) functional data.

Usage

```
msplot(
  dts,
  data_depth = c("random_projections"),
  n_projections = 200,
  seed = NULL,
  return_mvdir = TRUE,
  plot = TRUE,
  plot_title = "Magnitude Shape Plot",
  title_cex = 1.5,
  show_legend = T,
  ylabel = "VO",
  xlabel
)
```

Arguments

`dts`	A matrix/data frame for univariate functional data (of size n observations by p domain points) or a $3-$dimensional array for multivariate functional data (of size n observations by p domain points by d dimension).
`data_depth`	The depth used in the computation of the directional outlyingness of `dts`. The projection depth is always used. Support for other depth methods will be added.
`n_projections`	The number of random directions to generate for computing the random projection depth. By default 200 directions are generated.
`seed`	An integer indicating the seed to set when generating random directions for computing the random projection depth. NULL by default in which case no seed is set.
`return_mvdir`	A logical value indicating whether to return the mean and variation of directional outlyingness (MO and VO). For univariate functional data, MO and VO are vectors. For multivariate functional data, VO is a vector while MO is a matrix of size nxd.
`plot`	A logical indicating whether to make the msplot of VO against MO. In the case of multivariate functional data, a plot of VO against $\|\|MO\|\|$ is made.

plot_title	The title of the plot. Set to "Magnitude Shape Plot" by default. Ignored if plot = FALSE.
title_cex	Numerical value indicating the size of the plot title relative to the device default. Set to 1.5 by default. Ignored if plot = FALSE.
show_legend	A logical indicating whether to add legend to plot if plot = TRUE.
ylabel	The label of the y-axis. Set to "VO" by default.
xlabel	The label of the x-axis if plot = TRUE. If not specified (default), set to "MO" for univariate functional data and "||MO||" for multivariate functional data.

Details

MS-Plot finds outliers by computing the mean and variation of directional outlyingness (MO and VO) described in Dai and Genton (2019) doi:10.1016/j.csda.2018.03.017. A multivariate data whose columns are the computed MO and VO is then constructed and the robust mahalanobis distance(s) of the rows of this matrix are computed (using the minimum covariate determinant estimate of the location and scatter). The tail of the distribution of these distances is approximated using the F distribution according to Hardin and Rocke (2005) doi:10.1198/106186005X77685 to get the cutoff. The projection depth is always used for computing the directional outlyingness (as suggested by Dai and Genton (2019) doi:10.1016/j.csda.2018.03.017).

Value

Returns a list containing:

outliers_index	an integer vector containing the indices of the outliers.
median_curve	the index of the median function (which is the function with the smallest robust mahalanobis distance computed from the matrix whose columns are made up of MO and VO).
mean_outlyingness	if return_mvdir = TRUE, a numeric vector of the mean of directional outlyingness for univariate functional data or an nxd matrix of the mean of directional outlyingness for multivariate functional data.
var_outlyingness	if return_mvdir = TRUE, a numeric vector of length n observations containing the variation of directional outlyingness.

Author(s)

Oluwasegun Taiwo Ojo.

References

Dai, W., and Genton, M. G. (2018). Multivariate functional data visualization and outlier detection. *Journal of Computational and Graphical Statistics*, 27(4), 923-934.

Dai, W., and Genton, M. G. (2019). Directional outlyingness for multivariate functional data. *Computational Statistics & Data Analysis*, 131, 50-65.

Hardin, J., and Rocke, D. M. (2005). The distribution of robust distances. *Journal of Computational and Graphical Statistics*, 14(4), 928-946.

See Also

dir_out for directional outlyingness and projection_depth for multivariate projection depth.

Examples

```
# Univariate magnitude model in Dai and Genton (2018).
dt1 <- simulation_model1()
msplot_object <- msplot(dts = dt1$data)
msplot_object$outliers_index
msplot_object$mean_outlyingness
msplot_object$var_outlyingness
```

---

muod *Massive Unsupervised Outlier Detection (MUOD)*

Description

MUOD finds outliers by computing for each functional data, a magnitude, amplitude and shape index. Outliers are then detected in each set of index and outliers found are classified as either a magnitude, shape or amplitude outlier.

Usage

```
muod(dts, cut_method = c("boxplot", "tangent"))
```

Arguments

dts	a matrix or dataframe of size n observation by p domain points.
cut_method	a character value indicating method to use for finding indices cutoff. Must be either "boxplot" or "tangent".

Details

MUOD was proposed in Azcorra et al. (2020) doi:10.1038/s41598018248742 as a support method for finding influential users in a social network data. It was also mentioned in Vinue and Epiphano (2020) doi:10.1007/s11634020004129 where it was compared with other functional outlier detection methods.

MUOD computes for each curve three indices: amplitude, magnitude and shape indices. Then a cutoff is determined for each set of indices and outliers are identified in each set of index. Outliers identified in the magnitude indices are flagged as magnitude outliers. The same holds true for the amplitude and shape indices. Thus, the outliers are not only identified but also classified.

Value

Returns a list containing the following

outliers	a vector containing the indices of outliers identified.
indices	a dataframe containing the shape, magnitude and amplitude indices

References

Azcorra, A., Chiroque, L. F., Cuevas, R., Anta, A. F., Laniado, H., Lillo, R. E., Romo, J., & Sguera, C. (2018). Unsupervised scalable statistical method for identifying influential users in online social networks. Scientific reports, 8(1), 1-7.

Examples

```
dt1 <- simulation_model1()
md <- muod(dts = dt1$data)
str(md$outliers)
dim(md$indices)
```

`plot_dtt` *Plot Data from simulation models*

Description

Support function for plotting data generated by any of the simulation model functions `simulation_model1()` - `simulation_model9()`.

Usage

```
plot_dtt(
  y,
  grid_points,
  p,
  true_outliers,
  show_legend,
  plot_title,
  title_cex,
  ylabel,
  xlabel,
  legend_pos = "bottomright"
)
```

Arguments

`y`	A matrix of n observations by p domain points
`grid_points`	A vector of the evaluation/domain points of `y`
`p`	A value indicating the number of evaluation/domain points
`true_outliers`	An integer vector indicating the indices of the true outliers
`show_legend`	A logical indicating whether to add legend to plot if `plot = TRUE`.
`plot_title`	Title of plot if `plot` is `TRUE`
`title_cex`	Numerical value indicating the size of the plot title relative to the device default. Set to 1.5 by default. Ignored if `plot = FALSE`.

ylabel	The label of the y-axis. Set to "" by default.
xlabel	The label of the x-axis if plot = TRUE. Set to "gridpoints" by default.
legend_pos	A character value indicating the position of the legend. Can be one of "bottomright", "bottom", "bottomleft", "left" "topleft", "top", "topright", "right", ""center.

`projection_depth` *Random projection for multivariate data*

Description

Helper function to compute the random projection depth of multivariate point(s) with respect to a multivariate data.

Usage

```
projection_depth(dts, dt = dts, n_projections = 500L, seed = NULL)
```

Arguments

dts	A matrix or data frame of size m observations by d dimension or vector of length d. Contains the observation(s) whose depth is to be computed.
dt	A matrix or dataframe of size n observations by d dimension. Equals to dts by default.
n_projections	The number of directions for random projections. By default, 500 random directions for projection are generated from a scaled uniform distribution between -1 and 1.
seed	The random seed to set when generating the random directions. Defaults to NULL.

Value

A vector containing the depth values of dts with respect to dt.

Author(s)

Oluwasegun Taiwo Ojo

See Also

msplot for outlier detection using msplot and dir_out for directional outlyingness.

Examples

```
projection_depth(dts = iris[1:5, -5], dt = iris[1:10, -5], n_projection = 7, seed = 20)
```

`seq_transform` *Find and classify outliers functional outliers using Sequential Transformation*

Description

This method finds and classify outliers using sequential transformations proposed in Algorithm 1 of Dai et al. (2020) doi:10.1016/j.csda.2020.106960. A sequence of transformations are applied to the functional data and after each transformation, a functional boxplot is applied on the transformed data and outliers flagged by the functional data are noted. A number of transformations mentioned in Dai et al. (2020) doi:10.1016/j.csda.2020.106960 are supported including vertical alignment ("T1(X)(t)"), normalization ("T2(X)(t)"), one order of differencing ("D1(X)(t)" and "D2(X)(t)") and point-wise outlyingness data ("O(X)(t)"). The feature alignment transformation based on warping/curve registration is not yet supported.

Usage

```
seq_transform(
  dts,
  sequence = c("T0", "T1", "T2"),
 depth_method = c("mbd", "tvd", "extremal", "dirout", "linfinity", "bd", "erld", "dq"),
  save_data = FALSE,
  emp_factor = 1.5,
  central_region = 0.5,
  erld_type = NULL,
  dq_quantiles = NULL,
  n_projections = 200L,
  seed = NULL
)
```

Arguments

`dts`	A matrix for univariate functional data (of size `n` observations by `p` domain points) or a 3-dimensional array for multivariate functional data (of size `n` observations by `p` domain points by `d` dimension). Only the outlyingness transformation ("O(X)(t)") supports multivariate functional data so the sequence of transformation must always start with outlyingness ("O(X)(t)") whenever a multivariate functional data is parsed to `dts`.
`sequence`	A character vector usually of length between 1 and 6 containing any of the strings: `"T0"`, `"D0"`, `"T1"`, `"T2"`,`"D1"`, `"D2"` and `"O"` (in any order). These sequence of strings specifies the sequence of transformations to be applied on the data and their meanings are described as follows: `"T0"` **and** `"D0"` Functional boxplot applied on raw data (no transformation is applied). `"T1"` Apply vertical alignment on data, i.e. subtract from each curve its expectation over the domain of evaluation.

"T2" Apply normalization on data, i.e. divide each curve by its L-2 norm.

"D1" **and** "D2" Apply one order of differencing on data.

"O" Find the pointwise outlyingness of data. For multivariate functional data, this transformation replaces the multivariate functional data with a univariate functional data of pointwise outlyingness.

Examples of sequences of transformations include: "T0", c("T0", "T1", "D1"), c("T0", "T1", "T2"), c("T0", "D1", "D2") and c("T0", "T1", "T2", "D1", "D2"). See Details for their meaning.

depth_method A character value specifying depth/outlyingness method to use in the functional boxplot applied after each stage of transformation. Note that the same depth/outlyingness method is used in the functional boxplot applied after each transformation in the sequence. The following methods are currently supported:

"mbd": The modified band depth with bands defined by 2 functions. Uses the algorithm of Sun et al. (2012).

"tvd" The total variation depth of Huang and Sun (2019).

"extremal" The extremal depth of Narisetty and Nair (2016).

"dirout" Uses the robust distance of the mean and variation of directional outlyingness (dir_out) defined in Dai and Genton (2018). Since this method is a measure of outlyingness of a function the negative of the computed robust distance is used in ordering the functions.

"linfinity" The L-infinity depth defined in Long and Huang (2015) is used in ordering functions.

"bd" Uses the band depth with bands defined by 2 functions according to the algorithm of Sun et al. (2012)

erld Uses the extreme rank length depth defined in Myllymäki et al. (2017) and mentioned in Dai et al. (2020).

"dq" Uses the directional quantile (DQ) defined in Myllymäki et al. (2017) and mentioned in Dai et al. (2020). Since DQ is a measure of outlyingness, the negative of the DQ values is used in ordering the functions.

save_data A logical. If TRUE, the intermediate transformed data are returned in a list.

emp_factor The empirical factor for functional boxplot. Defaults to 1.5.

central_region A value between 0 and 1 indicating the central region probability for functional_boxplot. Defaults to 0.5.

erld_type If depth_method = "erld", the type of ordering to use in computing the extreme rank length depth (ERLD). Can be one of "two_sided", "one_sided_left" or "one_sided_right". A "two_sided" ordering is used by default if erld_type is not specified and depth_method = "erld". The "one_sided_right" ERLD is especially useful for ordering functions of outlyingness (the output of the "O" transformation) since it considers only large values as extreme. See extreme_rank_length for details.

dq_quantiles If depth_method = "dq", a numeric vector of length 2 specifying the probabilities of upper and lower quantiles. Defaults to c(0.025, 0.975) for the upper and lower 2.5% quantiles. See directional_quantile for details.

`n_projections` An integer indicating the number of random projections to use in computing the point-wise outlyingness if a 3-d array is specified in `dts` i.e. (multivariate functional data), and the transformation `"O"` is part of the sequence of transformations parsed to `sequence`. Defaults to 200L.

`seed` The random seed to set when generating the random directions in the computation of the point-wise outlyingness. Defaults to NULL. in which case a seed is not set.

Details

This function implements outlier detection using sequential transformations described in Algorithm 1 of Dai et al. (2020) doi:10.1016/j.csda.2020.106960. A sequence of transformations are applied consecutively with the functional boxplot applied on the transformed data after each transformation. The following example sequences (and their meaning) suggested in Dai et al. (2020) doi:10.1016/j.csda.2020.106960 can be parsed to argument `sequence`.

`"T0"` Apply functional boxplot on raw data (no transformation is applied).

`c("T0", "T1", "D1")` Apply functional boxplot on raw data, then apply vertical alignment on data followed by applying functional boxplot again. Finally apply one order of differencing on the vertically aligned data and apply functional boxplot again.

`c("T0", "T1", "T2")` Apply functional boxplot on raw data, then apply vertical alignment on data followed by applying functional boxplot again. Finally apply normalization using L-2 norm on the vertically aligned data and apply functional boxplot again.

`c("T0", "D1", "D2")` Apply functional boxplot on raw data, then apply one order of difference on data followed by applying functional boxplot again. Finally apply another one order of differencing on the differenced data and apply functional boxplot again. Note that this sequence of transformation can also be (alternatively) specified by `c("T0", "D1", "D1")`, `c("T0", "D2", "D2")`, and `c("T0", "D2", "D1")` since `"D1"` and `"D2"` do the same thing which is to apply one order lag-1 difference on the data.

`"O"` Find the pointwise outlyingness of the multivariate or univariate functional data and then apply functional boxplot on the resulting univariate functional data of pointwise outlyingness. Care must be taken to specify a one sided ordering function (i.e. "one_sided_right" extreme rank length depth) in the functional boxplot used on the data of point-wise outlyingness. This is because only large values should be considered extreme in the data of the point-wise outlyingness.

For multivariate functional data (when a 3-d array is supplied to `dts`), the sequence of transformation must always begin with `"O"` so that the multivariate data can be replaced with the univariate data of point-wise outlyingness which the functional boxplot can subsequently process because the `functional_boxplot` function only supports univariate functional data.

If repeated transformations are used in the sequence (e.g. when `sequence = c("T0", "D1", "D1")`), a warning message is thrown and the labels of the output list are changed (e.g. for `sequence = c("T0", "D1", "D1")`, the labels of the output lists become `"T0"`, `"D1_1"`, `"D1_2"`, so that outliers are accessed with `output$outlier$D1_1` and `output$outlier$D1_2`). See examples for more.

Value

A list containing two lists are returned. The contents of the returned list are:

`outliers:` A named list of length `length(sequence)` containing the index of outliers found after each transformation. The names of the elements of this list are the sequence strings supplied to `sequence` and the outliers found after each stage of transformation are not necessarily mutually exclusive.

`transformed_data` If `save_data = TRUE` a named list of length `length(sequence)` containing the transformed matrix after each transformation. The names of the elements of this list are the sequence strings supplied to `sequence`. `NULL` otherwise (if `save_data = FALSE`).

Examples

```
# same as running a functional boxplot
dt1 <- simulation_model1()
seqobj <- seq_transform(dt1$data, sequence = "T0", depth_method = "mbd")
seqobj$outliers$T0
functional_boxplot(dt1$data, depth_method = "mbd")$outliers

# more sequences
dt4 <- simulation_model4()
seqobj <- seq_transform(dt4$data, sequence = c("T0", "D1", "D2"), depth_method = "mbd")
seqobj$outliers$T0 # outliers found in raw data
seqobj$outliers$D1 # outliers found after differencing data the first time
seqobj$outliers$D2 # outliers found after differencing the data the second time

# saving transformed data
seqobj <- seq_transform(dt4$data, sequence = c("T0", "D1", "D2"),
 depth_method = "mbd", save_data = TRUE)
seqobj$outliers$T0 # outliers found in raw data
head(seqobj$transformed_data$T0)  # the raw data
head(seqobj$transformed_data$D1) # the first order differenced data
head(seqobj$transformed_data$D2) # the 2nd order differenced data

# double transforms e.g. c("T0", "D1", "D1")
seqobj <- seq_transform(dt4$data, sequence = c("T0", "D1", "D1"),
 depth_method = "mbd", save_data = TRUE) # throws warning
seqobj$outliers$T0 # outliers found in raw data
seqobj$outliers$D1_1 #found after differencing data the first time
seqobj$outliers$D1_2 #found after differencing data the second time
head(seqobj$transformed_data$T0)  # the raw data
head(seqobj$transformed_data$D1_1) # the first order differenced data
head(seqobj$transformed_data$D1_2) # the 2nd order differenced data

# multivariate data
dtm <- array(0, dim = c(dim(dt1$data), 2))
dtm[,,1] <- dt1$data
dtm[,,2] <- dt1$data
seqobj <- seq_transform(dtm, sequence = "O", depth_method = "erld",
```

```
 erld_type = "one_sided_right", save_data = TRUE)
seqobj$outliers$O # multivariate outliers
head(seqobj$transformed_data$O) # univariate outlyingness data
```

`simulation_model1` *Convenience function for generating functional data*

Description

This is a typical magnitude model in which outliers are shifted from the normal' non-outlying observations. The main model is of the form:

$$X_i(t) = \mu t + e_i(t),$$

and the contamination model model is of the form:

$$X_i(t) = \mu t + qk_i + e_i(t)$$

where $t \in [0, 1]$, $e_i(t)$ is a Gaussian process with zero mean and covariance function of the form:

$$\gamma(s, t) = \alpha \exp(-\beta|t - s|^{\nu}),$$

$k_i \in \{-1, 1\}$ (usually with $P(k_i = -1) = P(k_i = 1) = 0.5$), and q is a constant controlling how far the outliers are from the mean function of the data, usually, $q = 6$ or $q = 8$. The domain of the generated functions is over the interval $[0, 1]$. Please see the simulation models vignette with `vignette("simulation_models", package = "fdaoutlier")` for more details.

Usage

```
simulation_model1(
  n = 100,
  p = 50,
  outlier_rate = 0.05,
  mu = 4,
  q = 8,
  kprob = 0.5,
  cov_alpha = 1,
  cov_beta = 1,
  cov_nu = 1,
  deterministic = TRUE,
  seed = NULL,
  plot = F,
  plot_title = "Simulation Model 1",
  title_cex = 1.5,
  show_legend = T,
  ylabel = "",
  xlabel = "gridpoints"
)
```

Arguments

n	The number of curves to generate. Set to 100 by default.
p	The number of evaluation points of the curves. Curves are usually generated over the interval $[0, 1]$. Set to 50 by default.
outlier_rate	A value between $[0, 1]$ indicating the percentage of outliers. A value of 0.06 indicates about 6% of the observations will be outliers depending on whether the parameter deterministic is TRUE or not. Set to 0.05 by default.
mu	The mean value of the functions in the main and contamination model. Set to 4 by default.
q	A value indicating the shift of the outliers from the mean function, i.e., the q in the contamination model. Used to control how far the outliers are from the mean function. Set to 8 by default.
kprob	A value between 0 and 1 indicating the probability that an outlier will be above or below the mean function, i.e., $P(k_i = 1)$ in the contamination model. Can be used to control the amount of outliers above or below the mean. Set to 0.5 by default.
cov_alpha	A value indicating the coefficient of the exponential function of the covariance matrix, i.e., the α in the covariance function. Set to 1 by default.
cov_beta	A value indicating the coefficient of the terms inside the exponential function of the covariance matrix, i.e., the β in the covariance function. Set to 1 by default.
cov_nu	A value indicating the power to which to raise the terms inside the exponential function of the covariance matrix, i.e., the ν in the covariance function. Set to 1 by default.
deterministic	A logical value. If TRUE, the function will always return round(n*outlier_rate) outliers and consequently the number of outliers is always constant. If FALSE, the number of outliers are determined using n Bernoulli trials with probability outlier_rate, and consequently the number of outliers returned is random. TRUE by default.
seed	A seed to set for reproducibility. NULL by default in which case a seed is not set.
plot	A logical value indicating whether to plot data.
plot_title	Title of plot if plot is TRUE
title_cex	Numerical value indicating the size of the plot title relative to the device default. Set to 1.5 by default. Ignored if plot = FALSE.
show_legend	A logical indicating whether to add legend to plot if plot = TRUE.
ylabel	The label of the y-axis. Set to "" by default.
xlabel	The label of the x-axis if plot = TRUE. Set to "gridpoints" by default.

Value

A list containing:

data	a matrix of size n by p containing the simulated data set
true_outliers	a vector of integers indicating the row index of the outliers in the generated data.

Examples

```
dt <- simulation_model1(n = 50, plot = TRUE)
dim(dt$data)
dt$true_outliers
```

`simulation_model2` *Convenience function for generating functional data*

Description

This model generates non-persistent magnitude outliers, i.e., the outliers are magnitude outliers for only a portion of the domain of the functional data. The main model is of the form:

$$X_i(t) = \mu t + e_i(t),$$

with contamination model of the form:

$$X_i(t) = \mu t + q k_i I_{T_i \leq t \leq T_i + l} + e_i(t)$$

where: $t \in [0, 1]$, $e_i(t)$ is a Gaussian process with zero mean and covariance function of the form:

$$\gamma(s, t) = \alpha \exp(-\beta |t - s|^{\nu}),$$

$k_i \in \{-1, 1\}$ with $P(k_i = -1) = P(k_i = 1) = 0.5$, q is a constant controlling how far the outliers are from the mass of the data, I is an indicator function, T_i is a uniform random variable between an interval $[a, b] \subset [0, 1]$, and l is a constant specifying for how much of the domain the outliers are away from the mean function. Please see the simulation models vignette with `vignette("simulation_models", package = "fdaoutlier")` for more details.

Usage

```
simulation_model2(
  n = 100,
  p = 50,
  outlier_rate = 0.05,
  mu = 4,
  q = 8,
  kprob = 0.5,
  a = 0.1,
  b = 0.9,
  l = 0.05,
  cov_alpha = 1,
  cov_beta = 1,
  cov_nu = 1,
  deterministic = TRUE,
  seed = NULL,
  plot = F,
  plot_title = "Simulation Model 2",
```

```
  title_cex = 1.5,
  show_legend = T,
  ylabel = "",
  xlabel = "gridpoints"
)
```

Arguments

n	The number of curves to generate. Set to 100 by default.
p	The number of evaluation points of the curves. Curves are usually generated over the interval $[0, 1]$. Set to 50 by default.
outlier_rate	A value between $[0, 1]$ indicating the percentage of outliers. A value of 0.06 indicates about 6% of the observations will be outliers depending on whether the parameter `deterministic` is `TRUE` or not. Set to 0.05 by default.
mu	The mean value of the functions. Set to 4 by default.
q	A value indicating the shift of the outliers from the mean function. Used to control how far the outliers are from the mean function. Set to 8 by default.
kprob	A value between 0 and 1 indicating the probability that an outlier will be above or below the mean function. Can be used to control the amount of outliers above or below the mean. Set to 0.5 by default.
a, b	values values specifying the interval $[a, b]$ for the uniform distribution from which T_i is drawn in the contamination model.
l	the value of l in the contamination model
cov_alpha	A value indicating the coefficient of the exponential function of the covariance matrix, i.e., the α in the covariance function. Set to 1 by default.
cov_beta	A value indicating the coefficient of the terms inside the exponential function of the covariance matrix, i.e., the β in the covariance function. Set to 1 by default.
cov_nu	A value indicating the power to which to raise the terms inside the exponential function of the covariance matrix, i.e., the ν in the covariance function. Set to 1 by default.
deterministic	A logical value. If `TRUE`, the function will always return `round(n*outlier_rate)` outliers and consequently the number of outliers is always constant. If `FALSE`, the number of outliers are determined using `n` Bernoulli trials with probability `outlier_rate`, and consequently the number of outliers returned is random. `TRUE` by default.
seed	A seed to set for reproducibility. `NULL` by default in which case a seed is not set.
plot	A logical value indicating whether to plot data.
plot_title	Title of plot if `plot` is `TRUE`
title_cex	Numerical value indicating the size of the plot title relative to the device default. Set to 1.5 by default. Ignored if `plot = FALSE`.
show_legend	A logical indicating whether to add legend to plot if `plot = TRUE`.
ylabel	The label of the y-axis. Set to `""` by default.
xlabel	The label of the x-axis if `plot = TRUE`. Set to `"gridpoints"` by default.

Value

A list containing:

`data`	a matrix of size `n` by `p` containing the simulated data set
`true_outliers`	a vector of integers indicating the row index of the outliers in the generated data.

Examples

```
dtt <- simulation_model2(plot = TRUE)
dtt$true_outliers
dim(dtt$data)
```

`simulation_model3` *Convenience function for generating functional data*

Description

This model generates outliers that are magnitude outliers for a part of the domain. The main model is of the form:

$$X_i(t) = \mu t + e_i(t),$$

with contamination model of the form:

$$X_i(t) = \mu t + q k_i I_{T_i \leq t} + e_i(t)$$

where: $t \in [0, 1]$, $e_i(t)$ is a Gaussian process with zero mean and covariance function of the form:

$$\gamma(s, t) = \alpha \exp(-\beta |t - s|^{\nu}),$$

$k_i \in \{-1, 1\}$ with $P(k_i = -1) = P(k_i = 1) = 0.5$, q is a constant controlling how far the outliers are from the mass of the data, I is an indicator function, and T_i is a uniform random variable between an interval $[a, b] \subset [0, 1]$. Please see the simulation models vignette with `vignette("simulation_models", package = "fdaoutlier")` for more details.

Usage

```
simulation_model3(
  n = 100,
  p = 50,
  outlier_rate = 0.05,
  mu = 4,
  q = 6,
  a = 0.1,
  b = 0.9,
  kprob = 0.5,
  cov_alpha = 1,
  cov_beta = 1,
  cov_nu = 1,
```

```
  deterministic = TRUE,
  seed = NULL,
  plot = F,
  plot_title = "Simulation Model 3",
  title_cex = 1.5,
  show_legend = T,
  ylabel = "",
  xlabel = "gridpoints"
)
```

Arguments

n	The number of curves to generate. Set to 100 by default.
p	The number of evaluation points of the curves. Curves are usually generated over the interval $[0, 1]$. Set to 50 by default.
outlier_rate	A value between $[0, 1]$ indicating the percentage of outliers. A value of 0.06 indicates about 6% of the observations will be outliers depending on whether the parameter `deterministic` is `TRUE` or not. Set to 0.05 by default.
mu	The mean value of the functions. Set to 4 by default.
q	A value indicating the shift of the outliers from the mean function. Used to control how far the outliers are from the mean function. Set to 8 by default.
a, b	values values specifying the interval $[a, b]$ for the uniform distribution from which T_i is drawn in the contamination model.
kprob	A value between 0 and 1 indicating the probability that an outlier will be above or below the mean function. Can be used to control the amount of outliers above or below the mean. Set to 0.5 by default.
cov_alpha	A value indicating the coefficient of the exponential function of the covariance matrix, i.e., the α in the covariance function. Set to 1 by default.
cov_beta	A value indicating the coefficient of the terms inside the exponential function of the covariance matrix, i.e., the β in the covariance function. Set to 1 by default.
cov_nu	A value indicating the power to which to raise the terms inside the exponential function of the covariance matrix, i.e., the ν in the covariance function. Set to 1 by default.
deterministic	A logical value. If `TRUE`, the function will always return `round(n*outlier_rate)` outliers and consequently the number of outliers is always constant. If `FALSE`, the number of outliers are determined using `n` Bernoulli trials with probability `outlier_rate`, and consequently the number of outliers returned is random. `TRUE` by default.
seed	A seed to set for reproducibility. `NULL` by default in which case a seed is not set.
plot	A logical value indicating whether to plot data.
plot_title	Title of plot if `plot` is `TRUE`
title_cex	Numerical value indicating the size of the plot title relative to the device default. Set to 1.5 by default. Ignored if `plot = FALSE`.
show_legend	A logical indicating whether to add legend to plot if `plot = TRUE`.
ylabel	The label of the y-axis. Set to `""` by default.
xlabel	The label of the x-axis if `plot = TRUE`. Set to `"gridpoints"` by default.

Value

A list containing:

`data`	a matrix of size `n` by `p` containing the simulated data set
`true_outliers`	a vector of integers indicating the row index of the outliers in the generated data.

Examples

```
dt <- simulation_model3(plot = TRUE)
dt$true_outliers
dim(dt$data)
```

`simulation_model4` *Convenience function for generating functional data*

Description

This models generates outliers defined on the reversed time interval of the main model. The main model is of the form:

$$X_i(t) = \mu t(1-t)^m + e_i(t),$$

with contamination model of the form:

$$X_i(t) = \mu(1-t)t^m + e_i(t)$$

Please see the simulation models vignette with `vignette("simulation_models", package = "fdaoutlier")` for more details.

Usage

```
simulation_model4(
  n = 100,
  p = 50,
  outlier_rate = 0.05,
  mu = 30,
  m = 3/2,
  cov_alpha = 0.3,
  cov_beta = (1/0.3),
  cov_nu = 1,
  deterministic = TRUE,
  seed = NULL,
  plot = F,
  plot_title = "Simulation Model 4",
  title_cex = 1.5,
  show_legend = T,
  ylabel = "",
  xlabel = "gridpoints"
)
```

Arguments

n	The number of curves to generate. Set to 100 by default.
p	The number of evaluation points of the curves. Curves are usually generated over the interval $[0, 1]$. Set to 50 by default.
outlier_rate	A value between $[0, 1]$ indicating the percentage of outliers. A value of 0.06 indicates about 6% of the observations will be outliers depending on whether the parameter `deterministic` is `TRUE` or not. Set to 0.05 by default.
mu	The mean value of the functions. Set to `30` by default.
m	the constant m in the main and contamination model. Set to $3/2$ by default.
cov_alpha	A value indicating the coefficient of the exponential function of the covariance matrix, i.e., the α in the covariance function. Set to 1 by default.
cov_beta	A value indicating the coefficient of the terms inside the exponential function of the covariance matrix, i.e., the β in the covariance function. Set to 1 by default.
cov_nu	A value indicating the power to which to raise the terms inside the exponential function of the covariance matrix, i.e., the ν in the covariance function. Set to 1 by default.
deterministic	A logical value. If `TRUE`, the function will always return `round(n*outlier_rate)` outliers and consequently the number of outliers is always constant. If `FALSE`, the number of outliers are determined using `n` Bernoulli trials with probability `outlier_rate`, and consequently the number of outliers returned is random. `TRUE` by default.
seed	A seed to set for reproducibility. `NULL` by default in which case a seed is not set.
plot	A logical value indicating whether to plot data.
plot_title	Title of plot if `plot` is `TRUE`
title_cex	Numerical value indicating the size of the plot title relative to the device default. Set to 1.5 by default. Ignored if `plot = FALSE`.
show_legend	A logical indicating whether to add legend to plot if `plot = TRUE`.
ylabel	The label of the y-axis. Set to `""` by default.
xlabel	The label of the x-axis if `plot = TRUE`. Set to `"gridpoints"` by default.

Value

A list containing:

data	a matrix of size `n` by `p` containing the simulated data set
true_outliers	a vector of integers indicating the row index of the outliers in the generated data.

Examples

```
dt <- simulation_model4(plot = TRUE)
dt$true_outliers
dim(dt$data)
```

`simulation_model5` *Convenience function for generating functional data*

Description

This models generates shape outliers with a different covariance structure from that of the main model. The main model is of the form:

$$X_i(t) = \mu t + e_i(t),$$

contamination model of the form:

$$X_i(t) = \mu t + \tilde{e}_i(t),$$

where $t \in [0, 1]$, and $e_i(t)$ and $\tilde{e}_i(t)$ are Gaussian processes with zero mean and covariance function of the form:

$$\gamma(s, t) = \alpha \exp(-\beta |t - s|^{\nu})$$

Please see the simulation models vignette with `vignette("simulation_models", package = "fdaoutlier")` for more details.

Usage

```
simulation_model5(
  n = 100,
  p = 50,
  outlier_rate = 0.05,
  mu = 4,
  cov_alpha = 1,
  cov_beta = 1,
  cov_nu = 1,
  cov_alpha2 = 5,
  cov_beta2 = 2,
  cov_nu2 = 0.5,
  deterministic = TRUE,
  seed = NULL,
  plot = F,
  plot_title = "Simulation Model 5",
  title_cex = 1.5,
  show_legend = T,
  ylabel = "",
  xlabel = "gridpoints"
)
```

Arguments

`n`	The number of curves to generate. Set to 100 by default.
`p`	The number of evaluation points of the curves. Curves are usually generated over the interval $[0, 1]$. Set to 50 by default.

`outlier_rate` A value between $[0, 1]$ indicating the percentage of outliers. A value of 0.06 indicates about 6% of the observations will be outliers depending on whether the parameter `deterministic` is `TRUE` or not. Set to 0.05 by default.

`mu` The mean value of the functions. Set to 4 by default.

`cov_alpha, cov_alpha2`
A value indicating the coefficient of the exponential function of the covariance matrix, i.e., the α in the covariance function. `cov_alpha` is for the main model while `cov_alpha2` is for the covariance function of the contamination model. `cov_alpha` is set to 1 by default while `cov_alpha2` is set to 5 by default.

`cov_beta, cov_beta2`
A value indicating the coefficient of the terms inside the exponential function of the covariance matrix, i.e., the β in the covariance function. `cov_beta` is for the main model while `cov_beta2` is for the covariance function of the contamination model. `cov_beta` is set to 1 by default while `cov_beta2` is set to 2 by default.

`cov_nu, cov_nu2`
A value indicating the power to which to raise the terms inside the exponential function of the covariance matrix, i.e., the ν in the covariance function. `cov_nu` is for the main model while `cov_nu2` is for the covariance function of the contamination model. `cov_nu` is set to 1 by default while `cov_nu2` is set to 0.5 by default.

`deterministic` A logical value. If `TRUE`, the function will always return `round(n*outlier_rate)` outliers and consequently the number of outliers is always constant. If `FALSE`, the number of outliers are determined using `n` Bernoulli trials with probability `outlier_rate`, and consequently the number of outliers returned is random. `TRUE` by default.

`seed` A seed to set for reproducibility. `NULL` by default in which case a seed is not set.

`plot` A logical value indicating whether to plot data.

`plot_title` Title of plot if `plot` is `TRUE`

`title_cex` Numerical value indicating the size of the plot title relative to the device default. Set to 1.5 by default. Ignored if `plot = FALSE`.

`show_legend` A logical indicating whether to add legend to plot if `plot = TRUE`.

`ylabel` The label of the y-axis. Set to `""` by default.

`xlabel` The label of the x-axis if `plot = TRUE`. Set to `"gridpoints"` by default.

Value

A list containing:

`data` a matrix of size `n` by `p` containing the simulated data set

`true_outliers` a vector of integers indicating the row index of the outliers in the generated data.

Examples

```
dt <- simulation_model5(plot = TRUE)
dt$true_outliers
dim(dt$data)
```

`simulation_model6` *Convenience function for generating functional data*

Description

This models generates shape outliers that have a different shape for a portion of the domain. The main model is of the form:

$$X_i(t) = \mu t + e_i(t),$$

with contamination model of the form:

$$X_i(t) = \mu t + (-1)^u q + (-1)^{(1-u)}(\frac{1}{\sqrt{r\pi}}) \exp(-z(t-v)^w) + e_i(t)$$

where: $t \in [0, 1]$, $e_i(t)$ is a Gaussian process with zero mean and covariance function of the form:

$$\gamma(s,t) = \alpha \exp(-\beta|t-s|^\nu),$$

u follows Bernoulli distribution with probability $P(u = 1) = 0.5$; q, r, z and w are constants, and v follows a Uniform distribution between an interval $[a, b]$ and m is a constant. Please see the simulation models vignette with `vignette("simulation_models", package = "fdaoutlier")` for more details.

Usage

```
simulation_model6(
  n = 100,
  p = 50,
  outlier_rate = 0.1,
  mu = 4,
  q = 1.8,
  kprob = 0.5,
  a = 0.25,
  b = 0.75,
  cov_alpha = 1,
  cov_beta = 1,
  cov_nu = 1,
  pi_coeff = 0.02,
  exp_pow = 2,
  exp_coeff = 50,
  deterministic = TRUE,
  seed = NULL,
  plot = F,
  plot_title = "Simulation Model 6",
  title_cex = 1.5,
  show_legend = T,
  ylabel = "",
  xlabel = "gridpoints"
)
```

Arguments

n	The number of curves to generate. Set to 100 by default.
p	The number of evaluation points of the curves. Curves are usually generated over the interval $[0, 1]$. Set to 50 by default.
outlier_rate	A value between $[0, 1]$ indicating the percentage of outliers. A value of 0.06 indicates about 6% of the observations will be outliers depending on whether the parameter deterministic is TRUE or not. Set to 0.05 by default.
mu	The mean value of the functions in the main and contamination model. Set to 4 by default.
q	The constant term q in the contamination model. Set to 1.8 by default.
kprob	The probability $P(u = 1)$. Set to 0.5 by default.
a, b	Values specifying the interval of from which v in the contamination model is drawn. Set to 0.25 and 0.75 respectively.
cov_alpha	A value indicating the coefficient of the exponential function of the covariance matrix, i.e., the α in the covariance function. Set to 1 by default.
cov_beta	A value indicating the coefficient of the terms inside the exponential function of the covariance matrix, i.e., the β in the covariance function. Set to 1 by default.
cov_nu	A value indicating the power to which to raise the terms inside the exponential function of the covariance matrix, i.e., the ν in the covariance function. Set to 1 by default.
pi_coeff	The constant r in the contamination model i.e., the coefficient of of pi. Set to 0.02 by default.
exp_pow	The constant w in the contamination model i.e., the power of the term in the exponential function of the contamination model. Set to 2.
exp_coeff	The constant z in the contamination model i.e., the coefficient term in the exponential function of the contamination model. Set to 50 by default.
deterministic	A logical value. If TRUE, the function will always return round(n*outlier_rate) outliers and consequently the number of outliers is always constant. If FALSE, the number of outliers are determined using n Bernoulli trials with probability outlier_rate, and consequently the number of outliers returned is random. TRUE by default.
seed	A seed to set for reproducibility. NULL by default in which case a seed is not set.
plot	A logical value indicating whether to plot data.
plot_title	Title of plot if plot is TRUE
title_cex	Numerical value indicating the size of the plot title relative to the device default. Set to 1.5 by default. Ignored if plot = FALSE.
show_legend	A logical indicating whether to add legend to plot if plot = TRUE.
ylabel	The label of the y-axis. Set to "" by default.
xlabel	The label of the x-axis if plot = TRUE. Set to "gridpoints" by default.

Value

A list containing:

`data`	a matrix of size `n` by `p` containing the simulated data set
`true_outliers`	a vector of integers indicating the row index of the outliers in the generated data.

Examples

```
dt <- simulation_model6(n = 50, plot = TRUE)
dim(dt$data)
dt$true_outliers
```

`simulation_model7` *Convenience function for generating functional data*

Description

This model generates pure shape outliers that are periodic. The main model is of the form:

$$X_i(t) = \mu t + e_i(t),$$

with contamination model of the form:

$$X_i(t) = \mu t + k \sin(r\pi(t + \theta)) + e_i(t),$$

where: $t \in [0, 1]$, and $e_i(t)$ is a Gaussian process with zero mean and covariance function of the form:

$$\gamma(s, t) = \alpha \exp -\beta |t - s|^{\nu},$$

θ is uniformly distributed in an interval $[a, b]$ and k, r are constants. Please see the simulation models vignette with `vignette("simulation_models", package = "fdaoutlier")` for more details.

Usage

```
simulation_model7(
  n = 100,
  p = 50,
  outlier_rate = 0.05,
  mu = 4,
  sin_coeff = 2,
  pi_coeff = 4,
  a = 0.25,
  b = 0.75,
  cov_alpha = 1,
  cov_beta = 1,
  cov_nu = 1,
  deterministic = TRUE,
  seed = NULL,
```

```
  plot = F,
  plot_title = "Simulation Model 7",
  title_cex = 1.5,
  show_legend = T,
  ylabel = "",
  xlabel = "gridpoints"
)
```

Arguments

n	The number of curves to generate. Set to 100 by default.
p	The number of evaluation points of the curves. Curves are usually generated over the interval $[0, 1]$. Set to 50 by default.
outlier_rate	A value between $[0, 1]$ indicating the percentage of outliers. A value of 0.06 indicates about 6% of the observations will be outliers depending on whether the parameter `deterministic` is `TRUE` or not. Set to 0.05 by default.
mu	The mean value of the functions in the main and contamination model. Set to `4` by default.
sin_coeff	The coefficient k in the contamination model, i.e, the coefficient of the sine term in the contamination model. Set to 2 by default.
pi_coeff	The coefficient r in the contamination model, i.e., the coefficient of pi in the contamination model. Set to 4 by default.
a, b	Values indicating the interval of the uniform distribution from which θ should be drawn. Set by default to 0.25 and 0.75 respectively.
cov_alpha	A value indicating the coefficient of the exponential function of the covariance matrix, i.e., the α in the covariance function. Set to 1 by default.
cov_beta	A value indicating the coefficient of the terms inside the exponential function of the covariance matrix, i.e., the β in the covariance function. Set to 1 by default.
cov_nu	A value indicating the power to which to raise the terms inside the exponential function of the covariance matrix, i.e., the ν in the covariance function. Set to 1 by default.
deterministic	A logical value. If `TRUE`, the function will always return `round(n*outlier_rate)` outliers and consequently the number of outliers is always constant. If `FALSE`, the number of outliers are determined using `n` Bernoulli trials with probability `outlier_rate`, and consequently the number of outliers returned is random. `TRUE` by default.
seed	A seed to set for reproducibility. `NULL` by default in which case a seed is not set.
plot	A logical value indicating whether to plot data.
plot_title	Title of plot if `plot` is `TRUE`
title_cex	Numerical value indicating the size of the plot title relative to the device default. Set to 1.5 by default. Ignored if `plot = FALSE`.
show_legend	A logical indicating whether to add legend to plot if `plot = TRUE`.
ylabel	The label of the y-axis. Set to `""` by default.
xlabel	The label of the x-axis if `plot = TRUE`. Set to `"gridpoints"` by default.

Value

A list containing:

`data`	a matrix of size `n` by `p` containing the simulated data set
`true_outliers`	a vector of integers indicating the row index of the outliers in the generated data.

Examples

```
dt <- simulation_model7(n = 50, plot = TRUE)
dim(dt$data)
dt$true_outliers
```

`simulation_model8` *Convenience function for generating functional data*

Description

This model generates pure shape outliers that are periodic. The main model is of the form:

$$X_i(t) = k\sin(r\pi t) + e_i(t),$$

with contamination model of the form:

$$X_i(t) = k\sin(r\pi t + v) + e_i(t),$$

where $t \in [0, 1]$, and $e_i(t)$ is a Gaussian processes with zero mean and covariance function of the form:

$$\gamma(s, t) = \alpha \exp -\beta|t - s|^{\nu}$$

and k, r, v are constants. Please see the simulation models vignette with `vignette("simulation_models", package = "fdaoutlier")` for more details.

Usage

```
simulation_model8(
  n = 100,
  p = 50,
  outlier_rate = 0.05,
  pi_coeff = 15,
  sin_coeff = 2,
  constant = 2,
  cov_alpha = 1,
  cov_beta = 1,
  cov_nu = 1,
  deterministic = TRUE,
  seed = NULL,
  plot = F,
  plot_title = "Simulation Model 8",
```

```
  title_cex = 1.5,
  show_legend = T,
  ylabel = "",
  xlabel = "gridpoints"
)
```

Arguments

n	The number of curves to generate. Set to 100 by default.
p	The number of evaluation points of the curves. Curves are usually generated over the interval $[0, 1]$. Set to 50 by default.
outlier_rate	A value between $[0, 1]$ indicating the percentage of outliers. A value of 0.06 indicates about 6% of the observations will be outliers depending on whether the parameter `deterministic` is `TRUE` or not. Set to 0.05 by default.
pi_coeff	The coefficient r in the main and contamination model. Set to 15 by default.
sin_coeff	The coefficient k in the main and contamination model. Set to 2 by default.
constant	The value of the constant v in the contamination model. Set to 2 by default.
cov_alpha	A value indicating the coefficient of the exponential function of the covariance matrix, i.e., the α in the covariance function. Set to 1 by default.
cov_beta	A value indicating the coefficient of the terms inside the exponential function of the covariance matrix, i.e., the β in the covariance function. Set to 1 by default.
cov_nu	A value indicating the power to which to raise the terms inside the exponential function of the covariance matrix, i.e., the ν in the covariance function. Set to 1 by default.
deterministic	A logical value. If `TRUE`, the function will always return `round(n*outlier_rate)` outliers and consequently the number of outliers is always constant. If `FALSE`, the number of outliers are determined using `n` Bernoulli trials with probability `outlier_rate`, and consequently the number of outliers returned is random. `TRUE` by default.
seed	A seed to set for reproducibility. `NULL` by default in which case a seed is not set.
plot	A logical value indicating whether to plot data.
plot_title	Title of plot if `plot` is `TRUE`
title_cex	Numerical value indicating the size of the plot title relative to the device default. Set to 1.5 by default. Ignored if `plot = FALSE`.
show_legend	A logical indicating whether to add legend to plot if `plot = TRUE`.
ylabel	The label of the y-axis. Set to `""` by default.
xlabel	The label of the x-axis if `plot = TRUE`. Set to `"gridpoints"` by default.

Value

A list containing:

data	a matrix of size `n` by `p` containing the simulated data set
true_outliers	a vector of integers indicating the row index of the outliers in the generated data.

Examples

```
dt <- simulation_model8(plot = TRUE)
dim(dt$data)
dt$true_outliers
```

`simulation_model9` *Convenience function for generating functional data*

Description

Periodic functions with outliers of different amplitude. The main model is of the form

$$X_i(t) = a_{1i} \sin \pi + a_{2i} \cos \pi + e_i(t),$$

with contamination model of the form

$$X_i(t) = (b_{1i} \sin \pi + b_{2i} \cos \pi)(1 - u_i) + (c_{1i} \sin \pi + c_{2i} \cos \pi)u_i + e_i(t),$$

where $t \in [0, 1]$, $\pi \in [0, 2\pi]$, a_{1i}, a_{2i} follows uniform distribution in an interval $[a_1, a_2]$ b_{1i}, b_{i1} follows uniform distribution in an interval $[b_1, b_2]$; c_{1i}, c_{i1} follows uniform distribution in an interval $[c_1, c_2]$; u_i follows Bernoulli distribution and $e_i(t)$ is a Gaussian processes with zero mean and covariance function of the form

$$\gamma(s, t) = \alpha \exp -\beta |t - s|^{\nu}$$

Please see the simulation models vignette with `vignette("simulation_models", package = "fdaoutlier")` for more details.

Usage

```
simulation_model9(
  n = 100,
  p = 50,
  outlier_rate = 0.05,
  kprob = 0.5,
  ai = c(3, 8),
  bi = c(1.5, 2.5),
  ci = c(9, 10.5),
  cov_alpha = 1,
  cov_beta = 1,
  cov_nu = 1,
  deterministic = TRUE,
  seed = NULL,
  plot = F,
  plot_title = "Simulation Model 9",
  title_cex = 1.5,
  show_legend = T,
  ylabel = "",
  xlabel = "gridpoints"
)
```

Arguments

n	The number of curves to generate. Set to 100 by default.
p	The number of evaluation points of the curves. Curves are usually generated over the interval $[0, 1]$. Set to 50 by default.
outlier_rate	A value between $[0, 1]$ indicating the percentage of outliers. A value of 0.06 indicates about 6% of the observations will be outliers depending on whether the parameter `deterministic` is `TRUE` or not. Set to 0.05 by default.
kprob	The probability $P(u_i = 1)$. Set to 0.5 by default.
ai	A vector of two values containing a_{1i} and a_{2i} in the main model. Set to `c(3, 8)` by default.
bi	A vector of 2 values containing b_{1i} and b_{2i} in the contamination model. Set to `c(1.5, 2.5)` by default.
ci	A vector of 2 values containing c_1i and c_2i in the contamination model. Set to `c(9, 10.5)` by default.
cov_alpha	A value indicating the coefficient of the exponential function of the covariance matrix, i.e., the α in the covariance function. Set to 1 by default.
cov_beta	A value indicating the coefficient of the terms inside the exponential function of the covariance matrix, i.e., the β in the covariance function. Set to 1 by default.
cov_nu	A value indicating the power to which to raise the terms inside the exponential function of the covariance matrix, i.e., the ν in the covariance function. Set to 1 by default.
deterministic	A logical value. If `TRUE`, the function will always return `round(n*outlier_rate)` outliers and consequently the number of outliers is always constant. If `FALSE`, the number of outliers are determined using `n` Bernoulli trials with probability `outlier_rate`, and consequently the number of outliers returned is random. `TRUE` by default.
seed	A seed to set for reproducibility. `NULL` by default in which case a seed is not set.
plot	A logical value indicating whether to plot data.
plot_title	Title of plot if `plot` is `TRUE`
title_cex	Numerical value indicating the size of the plot title relative to the device default. Set to 1.5 by default. Ignored if `plot = FALSE`.
show_legend	A logical indicating whether to add legend to plot if `plot = TRUE`.
ylabel	The label of the y-axis. Set to `""` by default.
xlabel	The label of the x-axis if `plot = TRUE`. Set to `"gridpoints"` by default.

Value

A list containing:

data	a matrix of size `n` by `p` containing the simulated data set
true_outliers	a vector of integers indicating the row index of the outliers in the generated data.

Examples

```
dt <- simulation_model9(plot = TRUE)
dim(dt$data)
dt$true_outliers
```

`spanish_weather` *Spanish Weather Data*

Description

A dataset containing daily temperature, log precipitation and wind speed of 73 spanish weather stations in Spain between 1980 - 2009.

Usage

```
spanish_weather
```

Format

A list containing :

`$station_info`: A dataframe containing geographic information from the 73 weather stations with the following variables:

`ind`: id of weather station

`name`: name of weather station

`province`: province of weather station

`altitude`: altitude in meters of the station

`year.ini`: start year

`year.end`: end year

`longitude`: longitude of the coordinates of the weather station (in decimal degrees)

`longitude`: latitude of the coordinates of the weather station (in decimal degrees)

`$temperature`: A matrix of size 73 (stations) by 365 (days) containing average daily temperature for the period 1980-2009 (in degrees Celsius, marked with UTF-8 string). Leap years temperatures for February 28 and 29 were averaged.

`$wind_speed`: A matrix of size 73 (stations) by 365 (days) containing average daily wind speed for the period 1980-2009 (in m/s).

`$logprec`: A matrix of size 73 (stations) by 365 (days) containing average daily log precipitation for the period 1980-2009 (in log mm). Negligible precipitation (less than 1 tenth of mm) is replaced by 0.05 and no precipitation (`0.0` mm) is replaced by `0.01` after which the logarithm was applied.

Details

This is a stripped down version of the popular `aemet` spanish weather data available in the `fda.usc` doi:10.18637/jss.v051.i04 package. See the documentation of `fda.usc` for more details about data.

Source

Data obtained from the `fda.usc` doi:10.18637/jss.v051.i04 package.

Examples

```
data(spanish_weather)
names(spanish_weather)
names(spanish_weather$station_info)
```

`total_variation_depth` *Total Variation Depth and Modified Shape Similarity Index*

Description

This function computes the total variation depth (tvd) and the modified shape similarity index (mss) proposed in Huang and Sun (2019) doi:10.1080/00401706.2019.1574241.

Usage

```
total_variation_depth(dts)
```

Arguments

`dts`	A matrix or dataframe of size n observations/curves by p domain/evaluation points.

Details

This function computes the total variation depth (TVD) and modified shape similarity (MSS) index of a univariate functional data. The definition of the estimates of TVD and MSS can be found in Huang and Sun (2019) doi:10.1080/00401706.2019.1574241.

Value

Returns a list containing the following

`tvd`	the total variation depths of the observations of `dts`
`mss`	the modified shape similarity index of the observations of `dts`

Author(s)

Oluwasegun Ojo

References

Huang, H., & Sun, Y. (2019). A decomposition of total variation depth for understanding functional outliers. *Technometrics*, 61(4), 445-458.

See Also

tvd_mss for outlier detection using TVD and MSS.

Examples

```
dt6 <- simulation_model6()
tvd_object <- total_variation_depth(dt6$data)
```

---

tvd_mss *Outlier detection using the total variation depth and modified shape similarity index.*

Description

Find shape and magnitude outliers using the Total Variation Depth and Modified Shape Similarity Index proposed in Huang and Sun (2019) doi:10.1080/00401706.2019.1574241.

Usage

```
tvd_mss(
  data,
  emp_factor_mss = 1.5,
  emp_factor_tvd = 1.5,
  central_region_tvd = 0.5
)

tvdmss(
  dts,
  emp_factor_mss = 1.5,
  emp_factor_tvd = 1.5,
  central_region_tvd = 0.5
)
```

Arguments

emp_factor_mss	The empirical factor of the classical boxplot used on the modified shape similarity index. Defaults to 1.5.
emp_factor_tvd	The empirical factor of the functional boxplot used on the TVD of observations. Defaults to 1.5.
central_region_tvd	A number between 0 and 1 indicating the central region probability of the functional boxplot used on the TVD of the observations. Defaults to 0.5. See also details.
dts, data	A matrix or dataframe of size n observations/curves by p domain/evaluation points.

Details

This method uses a combination of total variation depth (TVD) and modified shape similarity (MSS) index defined in Huang and Sun (2019) doi:10.1080/00401706.2019.1574241 to find magnitude and shape outliers. The TVD and MSS of all the observations are first computed and a classical boxplot is then applied on the MSS. Outliers detected by the boxplot of MSS are flagged as shape outliers. The shape outliers are then removed from the data and the TVD of the remaining observations are used in a functional boxplot to detect magnitude outliers. The central region of this functional boxplot (`central_region_tvd`) is w.r.t. to the original number of curves. Thus if 8 shape outliers are found out of 100 curves, specifying `central_region_tvd` = 0.5 will ensure that 50 observations are used as the central region in the functional boxplot on the remaining 92 observations.

Value

Returns a list containing the following

`outliers`	the indices of the (shape and magnitude) outliers
`shape_outliers`	the indices of the shape outliers
`magnitude_outliers`	the indices of the magnitude outliers
`tvd`	the total variation depths of the observations of `data`
`mss`	the modified shape similarity index of the observations of `data`

Functions

- `tvd_mss()`: Deprecated function. Use `tvdmss` instead.

Author(s)

Oluwasegun Ojo

References

Huang, H., & Sun, Y. (2019). A decomposition of total variation depth for understanding functional outliers. *Technometrics*, 61(4), 445-458.

See Also

`msplot` for outlier detection using msplot.

Examples

```
dt6 <- simulation_model6()
res <- tvdmss(dt6$data)
res$outliers
```

`world_population` *World Population Data by Countries*

Description

This is the world population data, revision 2010, by countries used in the paper Nagy et al. (2016) doi:10.1080/10618600.2017.1336445 and Dai et al. (2020) doi:10.1016/j.csda.2020.106960. It contains population (both sexes) of countries as of July 1 in the years 1950 - 2010. The data have been pre-processed as described in Nagy et al. (2016) and hence contains only the 105 countries with population in the range of one million and fifteen million on July 1, 1980.

Usage

```
world_population
```

Format

A matrix of size 105 rows by 61 columns.

Details

Data included for illustration and testing purposes.

Source

Data originally available in the `depth.fd` package.

Examples

```
data(world_population)
str(world_population)
```

Index