Package ‘DySS’

October 12, 2022

Type Package

Title Dynamic Screening Systems

Version 1.0

Date 2022-07-04

Maintainer Lu You <Lu.You@epi.usf.edu>

Description In practice, we will encounter problems where the longitudinal performance of processes needs to be monitored over time. Dynamic screening systems (DySS) are methods that aim to identify and give signals to processes with poor performance as early as possible. This package is designed to implement dynamic screening systems and the related methods.

References:

License GPL-2 | GPL-3

Imports Rcpp (>= 1.0.0), utils, stats, graphics, ggplot2, gridExtra

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 7.1.2

Encoding UTF-8

Suggests knitr, rmarkdown

VignetteBuilder knitr

Depends R (>= 2.10)

LazyData true

NeedsCompilation yes

Author Lu You [aut, cre],
Peihua Qiu [aut]
**Description**

The function `calculate_ATS` calculates the average time to signals (ATS) given a control chart matrix and a specified control limit (CL). ATS is defined as the average time from the start of process monitoring to signal times.

**Usage**

```r
calculate_ATS(
    chart_matrix,
    time_matrix,
    nobs,
    starttime,
    endtime,
    design_interval,
    n_time_units,
    time_unit,
    CL,
    no_signal_action = "omit"
)
```
Arguments

chart_matrix: charting statistic values arranged as a numeric matrix.
chart_matrix[i,j] is the jth charting statistic of the ith subject.

time_matrix: observation times arranged as a numeric matrix.
time_matrix[i,j] is the jth observation time of the ith subject, corresponding
to the time the charting statistic chart_matrix[i,j] is computed.

nobs: number of observations arranged as an integer vector.
nobs[i] is the number of observations for the ith subject.

starttime: a numeric vector that gives the start times.
starttime[i] is the time that the ith subject starts to be monitored.

endtime: a numeric vector that gives the end times.
endtime[i] is the time that the ith subject is lost to be monitored.

design_interval: a numeric vector of length two that gives the left- and right- limits of the design
interval. By default, design_interval=range(time_matrix,na.rm=TRUE).

n_time_units: an integer value that gives the number of basic time units in the design time interval.
The design interval will be discretized to
seq(design_interval[1],design_interval[2],length.out=n_time_units)
time_unit: an optional numeric value of basic time unit. Only used when n_time_units is missing.
The design interval will be discretized to
seq(design_interval[1],design_interval[2],by=time_unit)

CL: a numeric value specifying the control limit.
CL is the control limit, signals will be given if charting statistics are greater than
the control limit.

no_signal_action: a character specifying the method to use when a signal is not given to a process.
If no_signal_action="omit" take averages by omitting the processes with no
signals, namely, average only the processes with signals.
If no_signal_action="maxtime" impute the signal times by the maximum
time, which is the right limit of design time interval.
If no_signal_action="endtime" impute the signal times by the end times.

Details

Calculate ATS

Value

a numeric value, the ATS given the charting statistics and the control limit.

References

methods. Technometrics, 62(2).
Examples

data("data_example_long_1d")

result_pattern<-estimate_pattern_long_1d(
  data_matrix=data_example_long_1d$data_matrix_IC,
  time_matrix=data_example_long_1d$time_matrix_IC,
  nob=0.1,
  design_interval=data_example_long_1d$design_interval,
  n_time_units=data_example_long_1d$n_time_units,
  estimation_method="meanvar",
  smoothing_method="local linear",
  bw_mean=0.1,
  bw_var=0.1)

result_monitoring<-monitor_long_1d(
  data_matrix_new=data_example_long_1d$data_matrix_OC,
  time_matrix_new=data_example_long_1d$time_matrix_OC,
  nob_new=0.1,
  pattern=result_pattern,
  side="upward",
  chart="CUSUM",
  method="standard",
  parameter=0.5)

result_ATS<-calculate_ATS(
  chart_matrix=result_monitoring$chart,
  time_matrix=data_example_long_1d$time_matrix_OC,
  nob=data_example_long_1d$nobs_OC,
  start_time=rep(0,nrow(data_example_long_1d$time_matrix_OC)),
  end_time=rep(1,nrow(data_example_long_1d$time_matrix_OC)),
  design_interval=data_example_long_1d$design_interval,
  n_time_units=data_example_long_1d$n_time_units,
  CL=2.0)

calculate_signal_times

**Calculate Signal Times**

Description

The function `calculate_signal_times` calculates the time to signals given a control chart matrix and a specified control limit (CL).

Usage

calculate_signal_times(
  chart_matrix,
  time_matrix,
calculate_signal_times

nobs,
starttime,
endtime,
design_interval,
n_time_units,
time_unit,
CL
)

Arguments

chart_matrix a matrix of charting statistic values.  
chart_matrix[i,j] is the jth charting statistic of the ith subject.

time_matrix a matrix of observation times.  
time_matrix[i,j] is the jth observation time of the ith subject, corresponding to the time the charting statistic chart_matrix[i,j] is computed.

nobs number of observations arranged as an integer vector.  
nobs[i] is the number of observations for the ith subject.

starttime a vector of times from the start of monitoring.  
starttime[i] is the time that the ith subject starts to be monitored.

endtime a vector of times from the start of monitoring.  
endtime[i] is the time that the ith subject is lost to be monitored.

design_interval a numeric vector of length two that gives the left- and right- limits of the design interval. By default, design_interval=range(time_matrix,na.rm=TRUE).

n_time_units an integer value that gives the number of basic time units in the design time interval.  
The design interval will be discretized to  
seq(design_interval[1],design_interval[2],length.out=n_time_units)

time_unit an optional numeric value of basic time unit. Only used when n_time_units is missing.  
The design interval will be discretized to  
seq(design_interval[1],design_interval[2],by=time_unit)

CL a numeric value specifying the control limit.  
CL is the control limit, signals will be given if charting statistics are greater than the control limit.

Details

Calculate Signal Times

Value

A list of two vectors:

signal_times times to signals, a numeric vector.

signals whether the subject received signals, a logical vector.
References

Examples

```r
data("data_example_long_1d")

result_pattern<-estimate_pattern_long_1d(
  data_matrix=data_example_long_1d$data_matrix_IC,
  time_matrix=data_example_long_1d$time_matrix_IC,
  nobs=data_example_long_1d$nobs_IC,
  design_interval=data_example_long_1d$design_interval,
  n_time_units=data_example_long_1d$n_time_units,
  estimation_method="meanvar",
  smoothing_method="local linear",
  bw_mean=0.1,
  bw_var=0.1)

result_monitoring<-monitor_long_1d(
  data_matrix_new=data_example_long_1d$data_matrix_OC,
  time_matrix_new=data_example_long_1d$time_matrix_OC,
  nobs_new=data_example_long_1d$nobs_OC,
  pattern=result_pattern,
  side="upward",
  chart="CUSUM",
  method="standard",
  parameter=0.5)

result_signal_times<-calculate_signal_times(
  chart_matrix=result_monitoring$chart,
  time_matrix=data_example_long_1d$time_matrix_OC,
  nobs=data_example_long_1d$nobs_OC,
  starttime=rep(0,nrow(data_example_long_1d$time_matrix_OC)),
  endtime=rep(1,nrow(data_example_long_1d$time_matrix_OC)),
  design_interval=data_example_long_1d$design_interval,
  n_time_units=data_example_long_1d$n_time_units,
  CL=2.0)
```

---

**data_example_long_1d**  
A simulated dataset with univariate data

**Description**

A simulated univariate longitudinal dataset for demonstration.

**Usage**

data(data_example_long_1d)
**data_example_long_md**

**Format**
An object of class `list` of length 9.

**Details**
Data Example: Univariate Longitudinal Data

**Value**
A list of the following components

- `$data_matrix_IC`: The data matrix for IC data.
- `$time_matrix_IC`: The time matrix for IC data.
- `$nobs_IC`: Number of observations for each IC process.
- `$data_matrix_OC`: The data matrix for OC data.
- `$time_matrix_OC`: The time matrix for OC data.
- `$nobs_OC`: Number of observations for each OC process.
- `$design_interval`: The design interval.
- `$n_time_units`: Number of time units in the design interval.
- `$time_unit`: The time unit.

**Examples**
```
data(data_example_long_1d)
```

**Description**
A simulated univariate longitudinal dataset for demonstration.

**Usage**
```
data(data_example_long_md)
```

**Format**
An object of class `list` of length 9.
Details

Data Example: Multivariate Longitudinal Data

Value

A list of the following components

- `$data_array_IC`: The data array for IC data.
- `$time_matrix_IC`: The time matrix for IC data.
- `$nobs_IC`: Number of observations for each IC process.
- `$data_array_OC`: The data array for OC data.
- `$time_matrix_OC`: The time matrix for OC data.
- `$nobs_OC`: Number of observations for each OC process.
- `$design_interval`: The design interval.
- `$n_time_units`: Number of time units in the design interval.
- `$time_unit`: The time unit.

Examples

```
data(data_example_long_md)
```

---

**data_example_long_surv**

*A simulated dataset with longitudinal and survival data*

Description

A simulated univariate longitudinal dataset for demonstration.

Usage

```
data(data_example_long_surv)
```

Format

An object of class `list` of length 15.

Details

Data Example: Longitudinal and Survival Data
Value
A list of the following components

$\text{data\_array\_IC}$  The data array for IC data.
$\text{time\_matrix\_IC}$  The time matrix for IC data.
$\text{nobs\_IC}$  Number of observations for each IC process.
$\text{starttime\_IC}$  Start time of monitoring for IC processes.
$\text{survtime\_IC}$  End time of monitoring for IC processes.
$\text{survevent\_IC}$  Survival events of IC processes.
$\text{data\_array\_OC}$  The data array for OC data.
$\text{time\_matrix\_OC}$  The time matrix for OC data.
$\text{nobs\_OC}$  Number of observations for each OC process.
$\text{starttime\_OC}$  Start time of monitoring for OC processes.
$\text{survtime\_OC}$  End time of monitoring for OC processes.
$\text{survevent\_OC}$  Survival events of OC processes.
$\text{design\_interval}$  The design interval.
$\text{n\_time\_units}$  Number of time units in the design interval.
$\text{time\_unit}$  The time unit.

Examples

data(data\_example\_long\_surv)

---

**data\_stroke**  \hspace{10cm} A real data example on stroke

Description
In this dataset, there are 27 subjects with stroke and 1028 subjects without stroke. Three risk factors, systolic blood pressures, diastolic blood pressures, cholesterol levels, are collected over time at different ages.

Usage

data(data\_stroke)

Format
An object of class list of length 8.
Details

Real Data Example: Stroke Data

Value

A list of the following components

$systolic_ctrl$  A matrix of systolic blood pressures for controls. The [i,j] element is the jth observation of the ith control.

$diastolic_ctrl$  A matrix of diastolic blood pressures for controls. The [i,j] element is the jth observation of the ith control.

$cholesterol_ctrl$  A matrix of cholesterol levels for controls. The [i,j] element is the jth observation of the ith control.

$age_ctrl$  A matrix of the age of observations for controls. The [i,j] element is the age of jth observation for the ith control.

$systolic_case$  A matrix of systolic blood pressures for cases. The [i,j] element is the jth observation of the ith case.

$diastolic_case$  A matrix of diastolic blood pressures for cases. The [i,j] element is the jth observation of the ith case.

$cholesterol_case$  A matrix of cholesterol levels for cases. The [i,j] element is the jth observation of the ith case.

$age_case$  A matrix of the age of observations for cases. The [i,j] element is the age of jth observation for the ith case.

Examples

```r
data(data_stroke)
```

---

estimate_pattern_long_1d

*Estimate the Regular Longitudinal Pattern of Univariate Data*

Description

Function `estimate_pattern_long_1d` estimate the regular longitudinal pattern of a univariate variable from a dataset of n subjects. This is usually the first step of dynamic screening. The pattern can be described by mean, variance, covariance, and distribution depending on the estimation method. When the estimated pattern is used for monitoring new subjects, the collected data from new subjects are compared to the estimated pattern for monitoring abnormality.
Usage

```r
estimate_pattern_long_1d(
  data_matrix,
  time_matrix,
  nobs,
  design_interval,
  n_time_units,
  time_unit,
  estimation_method,
  smoothing_method = "local linear",
  bw_mean,
  bw_var,
  bw_cov,
  bw_t,
  bw_y
)
```

Arguments

data_matrix observed data arranged in a numeric matrix format.
data_matrix[i,j] is the jth observation of the kth dimension of the ith subject.
time_matrix observation times arranged in a numeric matrix format.
time_matrix[i,j] is the jth observation time of the ith subject.
data_matrix[i,j] is observed at time_matrix[i,j].
nobs number of observations arranged as an integer vector.
nobs[i] is the number of observations for the ith subject.
design_interval a numeric vector of length two that gives the left- and right- limits of the design interval. By default, design_interval=range(time_matrix, na.rm=TRUE).
n_time_units an integer value that gives the number of basic time units in the design time interval.
The design interval will be discretized to seq(design_interval[1], design_interval[2], length.out=n_time_units)
time_unit an optional numeric value of basic time unit. Only used when n_time_units is missing.
The design interval will be discretized to seq(design_interval[1], design_interval[2], by=time_unit)
estimation_method a character specifying the estimation method.
If estimation_method="meanvar", the function will estimate the mean and variance functions using local smoothing (c.f., Qiu and Xiang, 2014). Parameters bw_mean and bw_var are required.
If estimation_method="meanvarcov", the function will estimate the mean, variance and covariance functions using local smoothing (c.f., Li and Qiu, 2016). Parameters bw_mean, bw_var and bw_cov are required.
If estimation_method="meanvarcovmean", the function will estimate the mean, variance and covariance functions (c.f., Li and Qiu, 2016). In the last step,
the mean function will be updated using the covariance function. Parameters bw_mean, bw_var and bw_cov are required.
If estimation_method="distribution", the function will estimate the distribution function (c.f., You and Qiu, 2020). Parameters bw_t and bw_y are required.
If estimation_method="distributionvarcov", the function will estimate the distribution function and the covariance function of standardized values (c.f., You and Qiu 2020). Parameters bw_cov, bw_t and bw_y are required.

smoothing_method
a character value specifying the smoothing method.
If smoothing_method="local constant", apply local constant approximation.
If smoothing_method="local linear", apply local linear approximation.

bw_mean
a numeric value.
The bandwidth parameter for estimating mean function.

bw_var
a numeric value.
The bandwidth parameter for estimating variance function.

bw_cov
a numeric value.
The bandwidth parameter for estimating covariance function.

bw_t
a numeric value.
The bandwidth parameter in time axis for estimating distribution function.

bw_y
a numeric value.
The bandwidth parameter in y-axis for estimating distribution function.

Details
Estimate the Regular Longitudinal Pattern of Univariate Data

Value
a list that stores the estimated longitudinal pattern and model parameters.
If estimation_method="meanvar", returns a list of class pattern_long_1d_meanvar
If estimation_method="meanvarcov" or "meanvarcovmean", returns a list of class pattern_long_1d_meanvarcov
If estimation_method="distribution", returns a list of class pattern_long_1d_distribution
If estimation_method="distributionvarcov", returns a list of class pattern_long_1d_distributionvarcov

$grid Discretized design interval.
$mean_est Estimated mean function.
$var_est Estimated variance function.
$cov_est Estimated covariance function.

References

Examples

data("data_example_long_1d")

result_pattern<-estimate_pattern_long_1d(
    data_matrix=data_example_long_1d$data_matrix_IC,
    time_matrix=data_example_long_1d$time_matrix_IC,
    nobs=data_example_long_1d$nobs_IC,
    design_interval=data_example_long_1d$design_interval,
    n_time_units=data_example_long_1d$n_time_units,
    estimation_method="meanvar",
    smoothing_method="local linear",
    bw_mean=0.1,
    bw_var=0.1)

---

estimate_pattern_long_md

Estimate the Regular Longitudinal Pattern of Multivariate Data

Description

Function estimate_pattern_long_md estimate the regular longitudinal pattern of multivariate processes from a dataset of n subjects. This is usually the first step of dynamic screening. The pattern can be described by mean, variance, covariance, and distribution depending on the estimation method. When the estimated pattern is used for monitoring new subjects, the collected data from new subjects are compared to the estimated pattern for monitoring abnormality.

Usage

estimate_pattern_long_md(  data_array,  time_matrix,  nobs,  design_interval,  n_time_units,  time_unit,  estimation_method,  bw_mean,  bw_var,  bw_cov)
**Arguments**

- **data_array**
  - observed data arranged in a 3d array format.
  - data_array[i,j,k] is the jth observation of the kth dimension of the ith subject.

- **time_matrix**
  - observation times arranged in a numeric matrix format.
  - time_matrix[i,j] is the jth observation time of the ith subject.
  - data_array[i,j,k] is observed at time_matrix[i,j].

- **nobs**
  - number of observations arranged as an integer vector.
  - nobs[i] is the number of observations for the ith subject.

- **design_interval**
  - a numeric vector of length two that gives the left- and right- limits of the design interval. By default, design_interval=range(time_matrix, na.rm=TRUE).

- **n_time_units**
  - an integer value that gives the number of basic time units in the design time interval.
  - The design interval will be discretized to seq(design_interval[1], design_interval[2], length.out=n_time_units)

- **time_unit**
  - an optional numeric value of basic time unit. Only used when n_time_units is missing.
  - The design interval will be discretized to seq(design_interval[1], design_interval[2], by=time_unit)

- **estimation_method**
  - a string.
  - If estimation_method="meanvar", the function will estimate the mean function (E[y(t)]), and variance function (Var(y(t))). Parameters bw_mean_int and bw_var_int are needed.
  - If estimation_method="meanvarcov", the function will estimate the mean function (E[y(t)]), variance function (Var(y(t))), and covariance function (Cov(y(s), y(t))). Parameters bw_mean_int, bw_var_int and bw_cov_int.

- **bw_mean**
  - a numeric value.
  - The bandwidth parameter for estimating mean function.

- **bw_var**
  - a numeric value.
  - The bandwidth parameter for estimating variance function.

- **bw_cov**
  - a numeric value.
  - The bandwidth parameter for estimating covariance function.

**Details**

Estimate the Regular Longitudinal Pattern of Multivariate Data

**Value**

- an object that stores the estimated longitudinal pattern and model parameters.
- If estimation_method="meanvar", returns an object of class pattern_long_md_meanvar.
- If estimation_method="meanvarcov", returns an object of class pattern_long_md_meanvarcov.
**$grid**
Discretized design interval.

**$mean_est**
Estimated mean function.

**$var_est**
Estimated variance function.

**$cov_est**
Estimated covariance function.

**References**


**Examples**

```r
data("data_example_long_md")

result_pattern<-estimate_pattern_long_md(
  data_array=data_example_long_md$data_array_IC,
  time_matrix=data_example_long_md$time_matrix_IC,
  nobs=data_example_long_md$nobs_IC,
  design_interval=data_example_long_md$design_interval,
  n_time_units=data_example_long_md$n_time_units,
  estimation_method="meanvar",
  bw_mean=0.1,
  bw_var=0.1)
```

---

**estimate_pattern_long_surv**

*Estimate the Pattern of Longitudinal and Survival Data*

**Description**

Function `estimate_pattern_long_surv` estimate the pattern of longitudinal and survival data from a dataset of n subjects. This is usually the first step of dynamic screening. The risk of a subject to event is quantified by a linear combination of longitudinal data by a Cox model. The risk pattern can be described by mean and variance depending on the estimation method. When the estimated pattern is used for monitoring new subjects, the collected data from new subjects are compared to the estimated pattern for monitoring abnormality.

**Usage**

```r
estimate_pattern_long_surv(
  data_array,
  time_matrix,
```
nobs,  
starttime,  
survtime,  
survevent,  
design_interval,  
n_time_units,  
time_unit,  
estimation_method = "risk",  
smoothing_method = "local linear",  
bw_beta,  
bw_mean,  
bw_var  
)

Arguments

data_array observed data arranged in a 3d array format.
data_array[i,j,k] is the jth observation of the kth dimension of the ith subject.

time_matrix observation times arranged in a numeric matrix format.
time_matrix[i,j] is the jth observation time of the ith subject.
data_array[i,j,] is observed at time_matrix[i,j].

nobs number of observations arranged as an integer vector.
nobs[i] is the number of observations for the ith subject.

starttime a vector of entry times
starttime[i] is the entry time of the ith subject.

survtime a vector of survival times
survtime[i] is the survival time of the ith subject.

survevent a logical vector of survival events
If survevents[i]==TRUE, then a survival event is observed at survtime[i].
If survevents[i]==FALSE, then no survival event is observed at survtime[i].

design_interval a numeric vector of length two that gives the left- and right- limits of the design interval. By default, design_interval=range(time_matrix,na.rm=TRUE).

n_time_units an integer value that gives the number of basic time units in the design time interval.
The design interval will be discretized to seq(design_interval[1],design_interval[2],length.out=n_time_units).

time_unit an optional numeric value of basic time unit. Only used when n_time_units is missing.
The design interval will be discretized to seq(design_interval[1],design_interval[2],by=time_unit).

estimation_method a string.
If estimation_method="risk", apply the risk monitoring method (c.f., You and Qiu 2020).
(Currently only the method "risk" is available.)
estimate_pattern_long_surv

smoothing_method
    a string.
    If smoothing_method="local constant", apply local constant smoothing
    If smoothing_method="local linear", apply local linear smoothing

bw_beta     an integer value.
    The bandwidth parameter for estimating the regression coefficients beta in the Cox model.

bw_mean     an integer value.
    The bandwidth parameter for estimating mean function.

bw_var      an integer value.
    The bandwidth parameter for estimating variance function.

Details

Estimate the Pattern of Longitudinal and Survival Data

Value

an object that stores the estimated longitudinal pattern and model parameters.
If estimation_method="risk", returns an object of class pattern_long_surv_risk.

$grid           discretized design interval.
$beta_est       Estimated regression coefficients.
$mean_risk_est  Estimated mean function.
$var_risk_est   Estimated variance function.

References


Examples

data("data_example_long_surv")

result_pattern<-estimate_pattern_long_surv(
  data_array=data_example_long_surv$data_array_IC,
  time_matrix=data_example_long_surv$time_matrix_IC,
  nobs=data_example_long_surv$nobs_IC,
  starttime=data_example_long_surv$starttime_IC,
  survtime=data_example_long_surv$survtime_IC,
  survevent=data_example_long_surv$survevent_IC,
  design_interval=data_example_long_surv$design_interval,
  n_time_units=data_example_long_surv$n_time_units,
  estimation_method="risk",
  smoothing_method="local linear",
  bw_beta=0.05,
evaluate_control_chart_one_group

Evaluate Control Charts (in a single dataset)

Description

The function `evaluate_control_chart_one_group` evaluates a control chart when the in-control (IC) and out-of-control (OC) charting statistics are supplied together in one matrix `chart_matrix`. The logical vector `status` indicates if the ith subject is IC or OC.

Usage

```r
evaluate_control_chart_one_group(
  chart_matrix,  # charting statistics arranged as a numeric matrix. chart_matrix[i,j] is the jth charting statistic of the ith subject.
  time_matrix,   # observation times arranged as a numeric matrix. time_matrix[i,j] is the jth observation time of the ith subject. chart_matrix[i,j] is the charting statistic of the ith subject at time_matrix[i,j].
  nobs,          # number of observations arranged as an integer vector. nobs[i] is the number of observations for the ith subject.
  starttime,     # a numeric vector. starttime[i] is the time when monitoring starts for ith subject.
  endtime,       # a numeric vector. endtime[i] is the time when monitoring ends for ith subject.
  status,        # a logical vector. status[i]=FALSE if the ith subject is IC, while status[i]=TRUE indicates the ith subject is OC.
  design_interval, # a numeric vector of length two that gives the left- and right- limits of the design interval. By default, design_interval=range(time_matrix, na.rm=TRUE).
  n_time_units,  #
  time_unit,     #
  no_signal_action = "omit"  #
)
```

Arguments

- `chart_matrix`: charting statistics arranged as a numeric matrix. `chart_matrix[i,j]` is the jth charting statistic of the ith subject.
- `time_matrix`: observation times arranged as a numeric matrix. `time_matrix[i,j]` is the jth observation time of the ith subject. `chart_matrix[i,j]` is the charting statistic of the ith subject at `time_matrix[i,j]`.
- `nobs`: number of observations arranged as an integer vector. `nobs[i]` is the number of observations for the ith subject.
- `starttime`: a numeric vector. `starttime[i]` is the time when monitoring starts for ith subject.
- `endtime`: a numeric vector. `endtime[i]` is the time when monitoring ends for ith subject.
- `status`: a logical vector. `status[i]=FALSE` if the ith subject is IC, while `status[i]=TRUE` indicates the ith subject is OC.
- `design_interval`: a numeric vector of length two that gives the left- and right- limits of the design interval. By default, `design_interval=range(time_matrix, na.rm=TRUE)`. 
- `n_time_units`: 
- `time_unit`: 
- `no_signal_action`: "omit"
evaluate_control_chart_one_group

n_time_units  an integer value that gives the number of basic time units in the design time interval. The design interval will be discretized to seq(design_interval[1],design_interval[2],length.out=n_time_units).

time_unit  an optional numeric value of basic time unit. Only used when n_time_units is missing. The design interval will be discretized to seq(design_interval[1],design_interval[2],by=time_unit).

no_signal_action  a character value specifying how to set signal times when processes with no signals. If no_signal_action="omit", the signal time is set to be missing. If no_signal_action="maxtime", the signal time is set to be the time from start time to the end of the design interval. If no_signal_action="endtime", the signal time is set to be the time from start time to the end time.

Details

Evaluate Control Charts

Value

an list that stores the evaluation measures.

$thres  A numeric vector. Threshold values for control limits.
$FPR  A numeric vector. False positive rates.
$TPR  A numeric vector. True positive rates.
$ATS0  A numeric vector. In-control ATS.
$ATS1  A numeric vector. Out-of-control ATS.

References


Examples

result_pattern<-estimate_pattern_long_surv(
data_array=data_example_long_surv$data_array_IC,
time_matrix=data_example_long_surv$time_matrix_IC,
nobs=data_example_long_surv$nobs_IC,
starttime=data_example_long_surv$starttime_IC,
survevent=data_example_long_surv$survevent_IC,
design_interval=data_example_long_surv$design_interval,
n_time_units=data_example_long_surv$n_time_units,
evaluate_control_chart_two_groups

**Evaluate Control Charts**

**Description**

The function `evaluate_control_chart_two_groups` evaluates control charts when the in-control (IC) and out-of-control (OC) charting statistics are supplied separately in two matrices `chart_matrix_IC` and `chart_matrix_OC`.

**Usage**

```r
evaluate_control_chart_two_groups(
  chart_matrix_IC, 
  time_matrix_IC, 
  nobs_IC, 
  starttime_IC, 
  endtime_IC, 
  chart_matrix_OC, 
  time_matrix_OC, 
  nobs_OC, 
  starttime_OC, 
  endtime_OC,
  estimation_method="risk", 
  smoothing_method="local linear", 
  bw_beta=0.05, 
  bw_mean=0.1, 
  bw_var=0.1)
```

```r
result_monitoring<-monitor_long_surv(
  data_array_new=data_example_long_surv$data_array_IC, 
  time_matrix_new=data_example_long_surv$time_matrix_IC, 
  nobs_new=data_example_long_surv$nobs_IC, 
  pattern=result_pattern, 
  method="risk", 
  parameter=0.5)
```

```r
output_evaluate<-evaluate_control_chart_one_group(
  chart_matrix=result_monitoring$chart[1:200,], 
  time_matrix=data_example_long_surv$time_matrix_IC[1:200,], 
  nobs=data_example_long_surv$nobs_IC[1:200], 
  starttime=rep(0,200), 
  endtime=rep(1,200), 
  status=data_example_long_surv$survevent_IC[1:200], 
  n_time_units=data_example_long_surv$n_time_units, 
  no_signal_action="maxtime")
```
design_interval,
n_time_units,
time_unit,
no_signal_action = "omit"
)

Arguments

chart_matrix_IC, chart_matrix_OC
charting statistics arranged as a numeric matrix.
chart_matrix_IC[i,j] is the jth charting statistic of the ith IC subject.
chart_matrix_OC[i,j] is the jth charting statistic of the ith OC subject.

time_matrix_IC, time_matrix_OC
observation times arranged as a numeric matrix.
time_matrix_IC[i,j] is the jth observation time of the ith IC subject.
time_matrix_OC[i,j] is the jth observation time of the ith OC subject.
chart_matrix_IC[i,j] is the charting statistic of the ith IC subject at time_matrix[i,j].
chart_matrix_OC[i,j] is the charting statistic of the ith OC subject at time_matrix[i,j].

nobs_IC, nobs_OC
number of observations arranged as an integer vector.
nobs_IC[i] is the number of observations for the ith subject.
nobs_OC[i] is the number of observations for the ith subject.

starttime_IC, starttime_OC
a numeric vector that gives the start times.
starttime_IC[i] is the time that the ith IC subject starts to be monitored.
starttime_OC[i] is the time that the ith OC subject starts to be monitored.

endtime_IC, endtime_OC
a numeric vector that gives the end times.
endtime_IC[i] is the time that the ith IC subject is lost to be monitored.
endtime_OC[i] is the time that the ith OC subject is lost to be monitored.

design_interval
a numeric vector of length two that gives the left- and right- limits of the design interval. By default, design_interval=range(time_matrix,na.rm=TRUE).
n_time_units
an integer value that gives the number of basic time units in the design time interval. The design interval will be discretized to seq(design_interval[1],design_interval[2],length.out=n_time_units).
time_unit
an optional numeric value of basic time unit. Only used when n_time_units is missing. The design interval will be discretized to seq(design_interval[1],design_interval[2],by=time_unit).
no_signal_action
a character value specifying how to set signal times when processes with no signals.
If no_signal_action="omit", the signal time is set to be missing.
If no_signal_action="maxtime", the signal time is set to be the time from start time to the end of the design interval.
If no_signal_action="endtime", the signal time is set to be the time from start time to the end time.
Details

Evaluate Control Charts

Value

an list that stores the evaluation measures.

$thres$   A numeric vector. Threshold values for control limits.
$FPR$     A numeric vector. False positive rates.
$TPR$     A numeric vector. True positive rates.
$ATS0$    A numeric vector. In-control ATS.
$ATS1$    A numeric vector. Out-of-control ATS.

References


Examples

```r
pattern<-estimate_pattern_long_1d(
  data_matrix=data_example_long_1d$data_matrix_IC,
  time_matrix=data_example_long_1d$time_matrix_IC,
  nobs=data_example_long_1d$nobs_IC,
  design_interval=data_example_long_1d$design_interval,
  n_time_units=data_example_long_1d$n_time_units,
  estimation_method="meanvar",
  smoothing_method="local linear",
  bw_mean=0.1,
  bw_var=0.1)

chart_IC_output<-monitor_long_1d(
  data_example_long_1d$data_matrix_IC,
  data_example_long_1d$time_matrix_IC,
  data_example_long_1d$nobs_IC,
  pattern=pattern,side="upward",chart="CUSUM",
  method="standard",parameter=0.2)

chart_OC_output<-monitor_long_1d(
  data_example_long_1d$data_matrix_OC,
  data_example_long_1d$time_matrix_OC,
  data_example_long_1d$nobs_OC,
  pattern=pattern,side="upward",chart="CUSUM",
  method="standard",parameter=0.2)

output_evaluate<-evaluate_control_chart_two_groups(
  chart_matrix_IC=chart_IC_output$chart[1:50],
  
```
monitor_long_1d

Monitor Univariate Longitudinal Data

Description
Monitor Univariate Longitudinal Data

Usage

monitor_long_1d(
  data_matrix_new,
  time_matrix_new,
  nobs_new,
  pattern,
  side = "upward",
  chart = "CUSUM",
  method = "standard",
  parameter = 0.5,
  CL = Inf
)

Arguments

data_matrix_new
observed data arranged in a numeric matrix format.
data_matrix_new[i,j] is the jth observation of the ith subject.

time_matrix_new
observation times arranged in a numeric matrix format.
time_matrix_new[i,j] is the jth observation time of the ith subject.
data_matrix_new[i,j] is observed at time_matrix_new[i,j].

nobs_new
number of observations arranged as an integer vector.
nobs_new[i] is the number of observations for the ith subject.

pattern
the estimated regular longitudinal pattern
side
a character value specifying the sideness/direction of process monitoring
If side="upward" apply control charts that aim to detect upward shifts.
If side="downward" apply control charts that aim to detect downward shifts.
If side="both" apply control charts that aim to detect shifts in both sides

chart
a string specifying the control charts to use. If chart="CUSUM" apply CUSUM charts.
If chart="EWMA" apply EWMA charts.

method
a string
If method="standard", standardize observations by mean and variance (cf., Qiu and Xiang, 2014).
If method="decorrelation", standardize and decorrelate observations by mean and covariance (cf., Li and Qiu, 2016).
If method="sprint", standardize and decorrelate observations within sprint length by mean and covariance (cf., You and Qiu 2018).
If method="distribution and standard", standardize observations by distribution (cf., You and Qiu, 2020).
If method="distribution and decorrelation", standardize observations by distribution and covariance (cf., You and Qiu, 2020).
If method="distribution and sprint", standardize and decorrelate observations within sprint length by distribution and covariance (cf., You and Qiu, 2020).
method="nonparametric and standard" currently not supported.
method="nonparametric and decorrelation" currently not supported

parameter
a numeric value
If chart="CUSUM", parameter is the allowance constant in the control chart.
If chart="EWMA", parameter is the weighting in the control chart.

CL
a numeric value specifying the control limit.
A signal will be given if charting statistics are larger than the control limit.
(Note: in this package, signs of charting statistics may be reversed such that larger values of charting statistics indicate worse performance of processes.)
After the signal is given, the algorithm stops calculating the charting statistics for the remaining observation times. The default value of control limit is infinity, which means we will calculate the charting statistics for all observation times.

Value
a list that stores the result.

$chart
a numeric matrix, $chart[i,j] is the jth charting statistic of the ith subject.

$standardized_values
a numeric matrix, $standardized_values[i,j] is the standardized value of the jth observation of the ith subject.

References

Examples

```r
data("data_example_long_1d")

result_pattern<-estimate_pattern_long_1d(
    data_matrix=data_example_long_1d$data_matrix_IC,
    time_matrix=data_example_long_1d$time_matrix_IC,
    nobs=data_example_long_1d$nobs_IC,
    design_interval=data_example_long_1d$design_interval,
    n_time_units=data_example_long_1d$n_time_units,
    estimation_method="meanvar",
    smoothing_method="local linear",
    bw_mean=0.1,
    bw_var=0.1)

result_monitoring<-monitor_long_1d(
    data_matrix_new=data_example_long_1d$data_matrix_OC,
    time_matrix_new=data_example_long_1d$time_matrix_OC,
    nobs_new=data_example_long_1d$nobs_OC,
    pattern=result_pattern,
    side="upward",
    chart="CUSUM",
    method="standard",
    parameter=0.5)
```

---

**monitor_long_md**  
**Monitor Multivariate Longitudinal Data**

**Description**

Monitor Multivariate Longitudinal Data

**Usage**

```r
monitor_long_md(
    data_array_new,
    time_matrix_new,
    nobs_new,
    pattern,
```
side = "both",
method = "multivariate EWMA",
parameter = 0.5,
CL = Inf
)

Arguments

data_array_new an array of longitudinal observations.
data_array_new[i,j,k] is the jth observation of the kth dimension of the ith subject.

time_matrix_new a matrix of observation times.
time_matrix_new[i,j] is the jth observation time of the ith subject.
data_array_new[i,j,k] is observed at time_matrix[i,j].
nobs_new an integer vector for number of observations.
nobs_new[i] is the number of observations for the ith subject.
pattern the estimated regular longitudinal pattern
side a string
If side="upward", control charts aim to detect upward shifts.
If side="downward", control charts aim to detect downward shifts.
If side="both", control charts aim to detect shifts in both sides.
method a string
If method="simultaneous CUSUM", apply simultaneous CUSUM charts. (See SIMUL in You et al, 2020.)
If method="simultaneous EWMA", apply simultaneous EWMA charts. (See SIMUL in You et al, 2020.)
If method="multivariate CUSUM", apply multivariate CUSUM charts.
If method="multivariate EWMA", apply multivariate EWMA charts. (See Qiu and Xiang, 2015 or QX-1S/QS-2S in You et al, 2020.)
If method="decorrelation CUSUM", apply decorrelation CUSUM charts. (See Li and Qiu, 2017 or LQ-1S/LQ-2S in You et al, 2020)
If method="decorrelation EWMA", apply decorrelation EWMA charts. (See Li and Qiu, 2017 or LQ-1S/LQ-2S in You et al, 2020)
If method="nonparametric CUSUM"
If method="nonparametric EWMA"

parameter a numeric value.
parameter is the allowance constant if method is a CUSUM chart.
parameter is the weighting parameter if method is an EWMA chart.

CL a numeric value
CL is the control limit. A signal will be given if charting statistics are larger than the control limit. (Note: in this package, signs of charting statistics may be reversed such that larger values of charting statistics indicate worse performance of processes.) After the signal is given, the algorithm stops calculating the charting statistics for the remaining observation times. The default value of control limit is infinity, which means we will calculate the charting statistics for all observation times.
Value

- a list that stores the result.

$\text{chart}$

- a numeric matrix, $\text{chart}[i,j]$ is the jth charting statistic of the ith subject calculated at time $\text{time_matrix_new}[i,j]$.

$\text{SSijk}$

- a numeric array, the multivariate statistics used in the calculation of control charts. $\text{SSijk}[i,j]$ is the jth multivariate statistic for the ith subject.

$\text{standardized_values}$

- a numeric array. $\text{standardized_values}[i,j]$ is the jth standardized vector for the ith subject.

References


Examples

data("data_example_long_md")

result_pattern<-estimate_pattern_long_md(
  data_array=data_example_long_md$data_array_IC,
  time_matrix=data_example_long_md$time_matrix_IC,
  nobs=data_example_long_md$nobs_IC,
  design_interval=data_example_long_md$design_interval,
  n_time_units=data_example_long_md$n_time_units,
  estimation_method="meanvar",
  bw_mean=0.1,
  bw_var=0.1)

result_monitoring<-monitor_long_md(
  data_array_new=data_example_long_md$data_array_OC,
  time_matrix_new=data_example_long_md$time_matrix_OC,
  nobs_new=data_example_long_md$nobs_OC,
  pattern=result_pattern,
  side="both",
  method="multivariate EWMA",
  parameter=0.5)

result_ATS<-calculate_ATS(
  chart_matrix=result_monitoring$chart_matrix,
  time_matrix=data_example_long_md$time_matrix_OC,
  nobs=data_example_long_md$nobs_OC,
monitor long surv

monitor_long_surv

Monitor Longitudinal Data for Survival Outcomes

Usage

monitor_long_surv(
  data_array_new,
  time_matrix_new,
  nobs_new,
  pattern,
  method,
  parameter = 0.5,
  CL = Inf
)

Arguments

data_array_new observed data arranged in a numeric array format.
data_array_new[i,j,k] is the jth observation of the kth dimension of the ith subject.

time_matrix_new observation times arranged in a numeric matrix format.
time_matrix_new[i,j] is the jth observation time of the ith subject.
data_array_new[i,j,k] is observed at time_matrix[i,j].
nobs_new number of observations arranged as an integer vector.
nobs_new[i] is the number of observations for the ith subject.

pattern the estimated longitudinal and survival pattern from estimate_pattern_long_surv.

method a character value specifying the smoothing method
If method="risk", apply the risk monitoring method by You and Qiu (2020).

parameter a numeric value.
The weighting parameter in the modified EWMA charts.

CL a numeric value specifying the control limit

Description

Monitor Longitudinal Data for Survival Outcomes

Usage

monitor_long_surv(
  data_array_new,
  time_matrix_new,
  nobs_new,
  pattern,
  method,
  parameter = 0.5,
  CL = Inf
)

Arguments

data_array_new observed data arranged in a numeric array format.
data_array_new[i,j,k] is the jth observation of the kth dimension of the ith subject.

time_matrix_new observation times arranged in a numeric matrix format.
time_matrix_new[i,j] is the jth observation time of the ith subject.
data_array_new[i,j,k] is observed at time_matrix[i,j].
nobs_new number of observations arranged as an integer vector.
nobs_new[i] is the number of observations for the ith subject.

pattern the estimated longitudinal and survival pattern from estimate_pattern_long_surv.

method a character value specifying the smoothing method
If method="risk", apply the risk monitoring method by You and Qiu (2020).

parameter a numeric value.
The weighting parameter in the modified EWMA charts.

CL a numeric value specifying the control limit
Value

a list that stores the result.

$chart    charting statistics arranged in a matrix.
$standardized_values    standardized values arranged in a matrix.

References


Examples

data("data_example_long_surv")

result_pattern<-estimate_pattern_long_surv(
  data_array=data_example_long_surv$data_array_IC,
  time_matrix=data_example_long_surv$time_matrix_IC,
  nobs=data_example_long_surv$nobs_IC,
  starttime=data_example_long_surv$starttime_IC,
  survtime=data_example_long_surv$survtime_IC,
  survevent=data_example_long_surv$survevent_IC,
  design_interval=data_example_long_surv$design_interval,
  n_time_units=data_example_long_surv$n_time_units,
  estimation_method="risk",
  smoothing_method="local linear",
  bw_beta=0.05,
  bw_mean=0.1,
  bw_var=0.1)

result_monitoring<-monitor_long_surv(
  data_array_new=data_example_long_surv$data_array_OC,
  time_matrix_new=data_example_long_surv$time_matrix_OC,
  nobs_new=data_example_long_surv$nobs_OC,
  pattern=result_pattern,
  method="risk",
  parameter=0.5)

plot_evaluation

Evaluate and Visualize Control Charts by ROC curves

Description

Evaluate and Visualize Control Charts by ROC curves

Usage

plot_evaluation(evaluate_control_chart)
**plot_evaluation**

**Arguments**

evaluate_control_chart

- an object of class evaluate_control_chart.

evaluate_control_chart is an output from evaluate_control_chart_one_group
or evaluate_control_chart_two.

**Value**

No return value, called for drawing two ROC plots.

**Examples**

```r
result_pattern<-estimate_pattern_long_surv(
  data_array=data_example_long_surv$data_array_IC,
  time_matrix=data_example_long_surv$time_matrix_IC,
  nobs=data_example_long_surv$nobs_IC,
  startime=data_example_long_surv$startime_IC,
  survtime=data_example_long_surv$survevent_IC,
  design_interval=data_example_long_surv$design_interval,
  n_time_units=data_example_long_surv$n_time_units,
  estimation_method="risk",
  smoothing_method="local linear",
  bw_beta=0.05,
  bw_mean=0.1,
  bw_var=0.1)

result_monitoring<-monitor_long_surv(
  data_array_new=data_example_long_surv$data_array_IC,
  time_matrix_new=data_example_long_surv$time_matrix_IC,
  nobs_new=data_example_long_surv$nobs_IC,
  pattern=result_pattern,
  method="risk",
  parameter=0.5)

output_evaluate<-evaluate_control_chart_one_group(
  chart_matrix=result_monitoring$chart,
  time_matrix=data_example_long_surv$time_matrix_IC,
  nobs=data_example_long_surv$nobs_IC,
  startime=rep(0,nrow(data_example_long_surv$time_matrix_IC)),
  endtime=rep(1,nrow(data_example_long_surv$time_matrix_IC)),
  status=data_example_long_surv$survevent_IC,
  design_interval=data_example_long_surv$design_interval,
  n_time_units=data_example_long_surv$n_time_units,
  no_signal_action="maxtime")

plot_evaluation(output_evaluate)
plot_PMROC(output_evaluate)
```
Evaluate and Visualize Control Charts by PM-ROC curves

Description
Evaluate and Visualize Control Charts by PM-ROC curves

Usage
plot_PMROC(evaluate_control_chart)

Arguments
- evaluate_control_chart
  - an object of class evaluate_control_chart.
  - evaluate_control_chart is an output from evaluate_control_chart_one_group or evaluate_control_chart_two_group.

Value
No return value, called for drawing one PM-ROC plot.

Examples

```r
pattern<-estimate_pattern_long_1d(
  data_matrix=data_example_long_1d$data_matrix_IC,
  time_matrix=data_example_long_1d$time_matrix_IC,
  nobs=data_example_long_1d$nobs_IC,
  design_interval=data_example_long_1d$design_interval,
  n_time_units=data_example_long_1d$n_time_units,
  estimation_method="meanvar",
  smoothing_method="local linear",
  bw_mean=0.1,
  bw_var=0.1)

chart_IC_output<-monitor_long_1d(
  data_example_long_1d$data_matrix_IC,
  data_example_long_1d$time_matrix_IC,
  data_example_long_1d$nobs_IC,
  pattern=pattern,side="upward",chart="CUSUM",
  method="standard",parameter=0.2)

chart_OC_output<-monitor_long_1d(
  data_example_long_1d$data_matrix_OC,
  data_example_long_1d$time_matrix_OC,
  data_example_long_1d$nobs_OC,
  pattern=pattern,side="upward",chart="CUSUM",
  method="standard",parameter=0.2)

output_evaluate<-evaluate_control_chart_two_groups(
```
search_CL

Description

Given a chart matrix, the function search_CL searches the control limit (CL) so that the specified average time to signals (ATS) can be attained.

Usage

search_CL(
  chart_matrix, time_matrix, nobs, starttime, endtime, design_interval, n_time_units, time_unit, ATS_nominal, CL_lower, CL_step, CL_upper, no_signal_action = "omit", ATS_tol, CL_tol
)
Arguments

chart_matrix  charting statistics arranged as a numeric matrix.
chart_matrix[i,j] is the jth charting statistic of the ith subject.

time_matrix observation times arranged as a numeric matrix.
time_matrix[i,j] is the jth observation time of the ith subject, corresponding
to the time the charting statistic chart_matrix[i,j] is computed.

nobs number of observations arranged as an integer vector.
nobs[i] is the number of observations for the ith subject.

starttime a vector of times from the start of monitoring.
starttime[i] is the time that the ith subject starts to be monitored.

endtime a vector of times from the start of monitoring.
endtime[i] is the time that the ith subject is lost to be monitored.

design_interval a numeric vector of length two that gives the left- and right- limits of the design
interval. By default, design_interval=range(time_matrix,na.rm=TRUE).

n_time_units an integer value that gives the number of basic time units in the design time
interval. The design interval will be discretized to
seq(design_interval[1],design_interval[2],length.out=n_time_units)

time_unit an optional numeric value of basic time unit. Only used when n_time_units is
missing. The design interval will be discretized to
seq(design_interval[1],design_interval[2],by=time_unit)

ATS_nominal a numeric value.
ATS_nominal is the nominal (or say targeted) ATS that is intended to achieve.

CL_lower, CL_step, CL_upper three numeric values.
The control limit will be searched within the interval [CL_lower,CL_upper].
When applying grid search, the algorithm will use a step size of CL_step.
(Namely, the algorithm will start with CL_lower, and search through the se-
quences CL_lower,CL_lower+CL_step,CL_lower+2*CL_step,... until CL_upper.)

no_signal_action a character specifying the method to use when a signal is not given to a process.
If no_signal_action="omit" take averages by omitting the processes with no
signals, namely, average only the processes with signals.
If no_signal_action="maxtime" impute the signal times by the maximum
time, which is the right limit of design time interval.
If no_signal_action="endtime" impute the signal times by the end times.

ATS_tol a numeric value.
Error tolerance for ATS.

CL_tol a numeric value.
Error tolerance for control limit.

Details

Search Control Limit
Value

a numeric value, the control limit that gives the desired ATS.

Examples

result_pattern <- estimate_pattern_long_1d(
  data_matrix = data_example_long_1d$data_matrix_IC,
  time_matrix = data_example_long_1d$time_matrix_IC,
  nobs = data_example_long_1d$nobs_IC,
  design_interval = data_example_long_1d$design_interval,
  n_time_units = data_example_long_1d$n_time_units,
  estimation_method = "meanvar",
  smoothing_method = "local linear",
  bw_mean = 0.1,
  bw_var = 0.1)

result_monitoring <- monitor_long_1d(
  data_matrix_new = data_example_long_1d$data_matrix_IC,
  time_matrix_new = data_example_long_1d$time_matrix_IC,
  nobs_new = data_example_long_1d$nobs_IC,
  pattern = result_pattern,
  side = "upward",
  chart = "CUSUM",
  method = "standard",
  parameter = 0.5)

CL <- search_CL(
  chart_matrix = result_monitoring$chart,
  time_matrix = data_example_long_1d$time_matrix_IC,
  nobs = data_example_long_1d$nobs_IC,
  starttime = rep(0, nrow(data_example_long_1d$time_matrix_IC)),
  endtime = rep(1, nrow(data_example_long_1d$time_matrix_IC)),
  design_interval = data_example_long_1d$design_interval,
  n_time_units = data_example_long_1d$n_time_units,
  ATS_nominal = 200, CL_lower = 0, CL_upper = 5)
Index

* datasets
  data_example_long_1d, 6
  data_example_long_md, 7
  data_example_long_surv, 8
  data_stroke, 9

calculate_ATS, 2
calculate_signal_times, 4

data_example_long_1d, 6
data_example_long_md, 7
data_example_long_surv, 8
data_stroke, 9

estimate_pattern_long_1d, 10
estimate_pattern_long_md, 13
estimate_pattern_long_surv, 15
evaluate_control_chart_one_group, 18
evaluate_control_chart_two_groups, 20

monitor_long_1d, 23
monitor_long_md, 25
monitor_long_surv, 28

plot_evaluation, 29
plot_PMROC, 31

search_CL, 32